AC and Noise Simulation for an Ethernet Phy

Gerhard Noessing, Villach

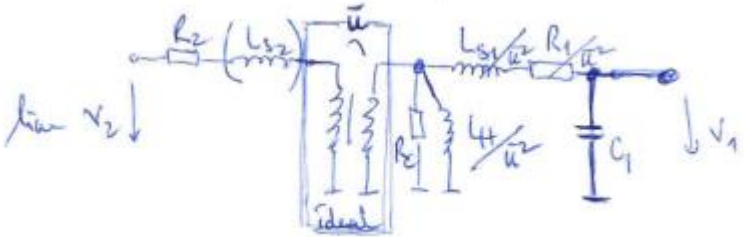
AGENDA

- Frequency Domain simulation Matlab or SystemC-AMS ?
- Noise simulation with SystemC-AMS
- Compare Time Domain with Frequency Domain Simulation
- Simulation Results
- Conclusion

Traditional Approach of Frequency Domain Model of Mixed Signal Blocks in Matlab

- Setup equations
- Solve equations via symbolic toolbox
- Output can be embedded into Matlab Frequency domain simulation

Matlab Equation Generation



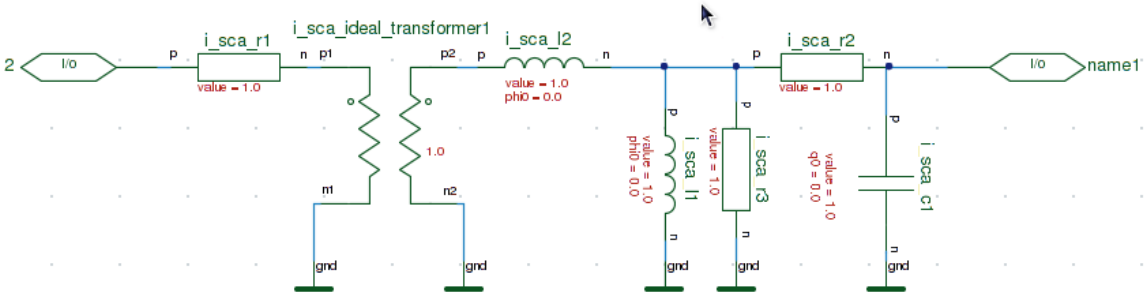

```
A11_trafo =
(1+(s.*Ls1./u_tr.^2+R1./u_tr.^2).*u_tr.^2./s./Lh+(s.*Ls1./u_tr.^2+R1./u_tr.^2)./Rc)./u_tr;

A12_trafo =
(1+(s.*Ls1./u_tr.^2+R1./u_tr.^2).*u_tr.^2./s./Lh+(s.*Ls1./u_tr.^2+R1./u_tr.^2)./Rc).*R2+(1+(s.*Ls1.
/u_tr.^2+R1./u_tr.^2).*u_tr.^2./s./Lh+(s.*Ls1./u_tr.^2+R1./u_tr.^2)./Rc).*s.*Ls2+s.*Ls1./u_tr.^2+R1
./u_tr.^2).*u_tr;

A21_trafo =
(s.*C1+(s.^2.*C1.*Ls1./u_tr.^2+s.*C1.*R1./u_tr.^2+1).*u_tr.^2./s./Lh+(s.^2.*C1.*Ls1./u_tr.^2+s.*C
1.*R1./u_tr.^2+1)./Rc)./u_tr;

A22_trafo =
((s.*C1+(s.^2.*C1.*Ls1./u_tr.^2+s.*C1.*R1./u_tr.^2+1).*u_tr.^2./s./Lh+(s.^2.*C1.*Ls1./u_tr.^2+s.*
C1.*R1./u_tr.^2+1)./Rc).*R2+(s.*C1+(s.^2.*C1.*Ls1./u_tr.^2+s.*C1.*R1./u_tr.^2+1).*u_tr.^2./s./Lh
+(s.^2.*C1.*Ls1./u_tr.^2+s.*C1.*R1./u_tr.^2+1)./Rc).*s.*Ls2+s.^2.*C1.*Ls1./u_tr.^2+s.*C1.*R1./u
_tr.^2+1).*u_tr;
```

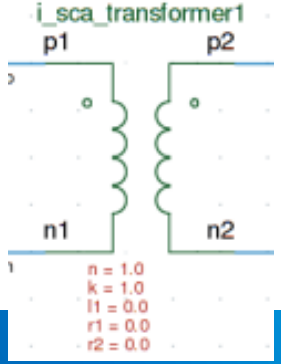
Huge Formulas for more complex Simulation

H_dac2line = -gdac.*Zin_afe.*(ZN_LD.*AD2.*Rout_pofi.*R1_pga.*Rout_pga.*ZH_buf+ZN_LD.*AD2.*Rout_pofi.*R1_pga.*Rout_pga.*ZH_line-
ZN_pofi.*AD1.*Rout_LD.*Rout_prefi.*R1_pga.*Zout_LD+ZN_pofi.*AD1.*Rout_LD.*R1_pga.*Rout_pga.*AD4.*ZH_buf+ZN_pofi.*AD1.*Rout_LD.*Rout_prefi.*R1_pga.*AD3.*ZH_buf-
ZN_pofi.*ZN_LD.*AD1.*AD2.*ZN_pga.*ZH_line.*Rout_pga+ZN_LD.*AD2.*Rout_pofi.*ZH_line.*ZH_buf.*Rout_pga.*AD4-ZN_pofi.*ZN_LD.*AD1.*AD2.*Rout_prefi.*Rout_pga.*ZH_buf-
ZN_pofi.*ZN_LD.*AD1.*AD2.*Rout_prefi.*ZN_pga.*ZH_line-
ZN_pofi.*ZN_LD.*AD1.*AD2.*R1_pga.*Rout_pga.*AD4.*ZH_line+ZN_pofi.*AD1.*Rout_LD.*ZN_prefi.*ZN_pga.*ZH_buf+ZN_LD.*AD2.*Rout_pofi.*R1_pga.*Rout_pga.*AD4.*ZH_buf+
ZN_pofi.*AD1.*Rout_LD.*Rout_prefi.*R1_pga.*ZH_line+ZN_pofi.*AD1.*Rout_LD.*Rout_prefi.*R1_pga.*ZH_buf-ZN_pofi.*ZN_LD.*AD1.*AD2.*R1_prefi.*R1_pga.*AD4.*AD3.*ZH_line-
ZN_pofi.*ZN_LD.*AD1.*AD2.*R1_prefi.*ZN_pga.*ZH_line+ZN_pofi.*ZN_LD.*AD1.*AD2.*Rout_prefi.*ZN_pga.*Zout_LD-
ZN_pofi.*AD1.*Rout_LD.*Rout_prefi.*R1_pga.*AD3.*Zout_LD+ZN_pofi.*ZN_LD.*AD1.*AD2.*Rout_prefi.*Rout_pga.*Zout_LD+ZN_pofi.*AD1.*Rout_LD.*Rout_prefi.*R1_pga.*AD3.*Z
H_line-ZN_pofi.*ZN_LD.*AD1.*AD2.*ZN_prefi.*ZH_buf.*AD3.*ZH_line-ZN_pofi.*ZN_LD.*AD1.*AD2.*ZN_prefi.*R1_pga.*AD3.*ZH_buf-
ZN_pofi.*ZN_LD.*AD1.*AD2.*R1_prefi.*R1_pga.*AD3.*ZH_buf+ZN_pofi.*AD1.*Rout_LD.*ZN_prefi.*Rout_pga.*ZH_buf-ZN_pofi.*ZN_LD.*AD1.*AD2.*ZN_prefi.*R1_pga.*ZH_buf-
ZN_pofi.*ZN_LD.*AD1.*AD2.*R1_prefi.*R1_pga.*AD4.*ZH_line-
ZN_pofi.*ZN_LD.*AD1.*AD2.*ZN_prefi.*R1_pga.*AD3.*ZH_line+ZN_pofi.*ZN_LD.*AD1.*AD2.*ZN_prefi.*R1_pga.*AD3.*Zout_LD+ZN_pofi.*ZN_LD.*AD1.*AD2.*ZN_prefi.*R1_pga.*Zo
ut_LD-ZN_pofi.*ZN_LD.*AD1.*AD2.*ZN_prefi.*R1_pga.*ZH_line-
ZN_LD.*AD2.*Rout_pofi.*ZN_pga.*Zout_LD.*Rout_pga.*AD4+ZN_LD.*AD2.*Rout_pofi.*ZN_pga.*ZH_buf.*Rout_pga+ZN_LD.*AD2.*Rout_pofi.*ZN_prefi.*R1_pga.*ZH_buf+Rout_pref
i.*ZN_pga.*Rout_LD.*Rout_pofi.*Zout_LD+ZN_pofi.*AD1.*Rout_LD.*R1_pga.*Rout_pga.*AD4.*ZH_line+ZN_LD.*AD2.*Rout_pofi.*ZN_pga.*ZH_line.*Rout_pga-
ZN_pofi.*ZN_LD.*AD1.*AD2.*R1_prefi.*ZH_buf.*ZH_line+ZN_pofi.*AD1.*Rout_LD.*ZN_prefi.*ZH_buf.*ZH_line+ZN_prefi.*Rout_pga.*Rout_LD.*Rout_pofi.*Zout_LD+R1_prefi.*R1_pg
a.*Rout_LD.*Rout_pofi.*Zout_LD+ZN_LD.*AD2.*Rout_pofi.*ZN_prefi.*R1_pga.*ZH_line-Rout_pofi.*ZH_buf.*Rout_LD.*Rout_pga.*ZH_line-
Rout_pofi.*ZH_buf.*Rout_LD.*Rout_pga.*AD4.*ZH_line+ZN_pofi.*AD1.*Rout_LD.*R1_prefi.*ZH_buf.*AD3.*ZH_line+ZN_pofi.*AD1.*Rout_LD.*ZN_prefi.*ZH_buf.*AD3.*ZH_line+ZN_p
ofi.*AD1.*Rout_LD.*R1_prefi.*ZH_buf.*AD4.*AD3.*ZH_line-
ZN_LD.*AD2.*Rout_pofi.*ZN_pga.*Zout_LD.*Rout_pga+ZN_LD.*AD2.*Rout_pofi.*ZN_pga.*ZH_line.*Rout_pga.*AD4+Rout_prefi.*Rout_pga.*Rout_LD.*Rout_pofi.*Zout_LD+Rout_pg
a.*R1_pga.*Rout_LD.*Rout_pofi.*Zout_LD-ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*AD3.*R1_prefi+Rout_pofi.*Zout_LD.*Rout_LD.*Rout_pga.*R1_prefi.*AD4-
ZH_line.*Rout_pofi.*Rout_LD.*ZN_pga.*R1_prefi-ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*Rout_prefi-ZH_line.*Rout_pofi.*ZN_pga.*Rout_LD.*Rout_pga-
ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*AD4.*R1_prefi+ZN_pofi.*ZN_LD.*AD1.*AD2.*Rout_prefi.*R1_pga.*AD3.*Zout_LD-ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*AD3.*R1_prefi-
ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*ZN_prefi-ZH_line.*Rout_pofi.*Rout_LD.*Rout_pga.*R1_prefi.*AD4-ZH_line.*Rout_pofi.*Rout_LD.*Rout_pga.*R1_prefi-
ZH_line.*Rout_pofi.*Rout_LD.*Rout_pga.*Rout_prefi-ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*Rout_pga-ZH_line.*Rout_pofi.*Rout_LD.*ZN_pga.*Rout_prefi-
ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*AD3.*ZN_prefi-ZH_line.*Rout_pofi.*Rout_LD.*ZN_pga.*AD4.*R1_prefi-ZH_line.*Rout_pofi.*ZN_pga.*Rout_LD.*Rout_pga.*AD4-
ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*AD4.*R1_prefi-ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*Rout_pga.*AD4-ZH_buf.*Rout_pofi.*Rout_LD.*Rout_pga.*Rout_prefi-
ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*R1_prefi-ZH_buf.*ZH_line.*Rout_pofi.*Rout_LD.*AD4.*R1_prefi-ZH_buf.*ZH_line.*Rout_pofi.*Rout_LD.*Rout_prefi-
ZH_buf.*ZH_line.*Rout_pofi.*Rout_LD.*AD3.*ZN_prefi-ZH_buf.*ZH_line.*Rout_pofi.*Rout_LD.*AD3.*AD4.*R1_prefi-ZH_buf.*ZH_line.*Rout_pofi.*Rout_LD.*AD3.*Rout_prefi-
ZH_buf.*Rout_pofi.*Rout_LD.*ZN_pga.*Rout_prefi-ZH_buf.*ZH_line.*Rout_pofi.*Rout_LD.*R1_prefi-ZH_buf.*Rout_pofi.*Rout_LD.*ZN_pga.*ZN_prefi-
ZH_buf.*ZH_line.*Rout_pofi.*Rout_LD.*ZN_prefi-ZH_buf.*Rout_pofi.*Rout_LD.*Rout_pga.*ZN_prefi-ZH_buf.*Rout_pofi.*Rout_LD.*ZN_pga.*R1_prefi-
ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*Rout_prefi-ZH_buf.*Rout_pofi.*ZN_pga.*Rout_LD.*Rout_pga-ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*AD3.*ZN_prefi-
ZH_buf.*Rout_pofi.*Rout_LD.*ZN_pga.*AD4.*R1_prefi-ZH_buf.*Rout_pofi.*ZN_pga.*Rout_LD.*Rout_pga.*AD4-ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*AD3.*Rout_prefi-
ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*AD3.*Rout_prefi+Rout_pofi.*Zout_LD.*Rout_LD.*R1_pga.*AD4.*R1_prefi-ZH_buf.*ZH_line.*Rout_pofi.*Rout_LD.*R1_pga.*AD3.*R1_prefi-
ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*AD3.*AD4.*R1_prefi-ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*ZN_prefi-ZH_buf.*Rout_pofi.*Rout_LD.*Rout_pga.*R1_prefi.*AD4-
ZH_buf.*Rout_pofi.*Rout_LD.*Rout_pga.*R1_prefi-ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*Rout_pga-ZH_buf.*Rout_pofi.*Rout_LD.*R1_pga.*Rout_pga.*AD4-

Frequency Domain Model with SystemC-AMS

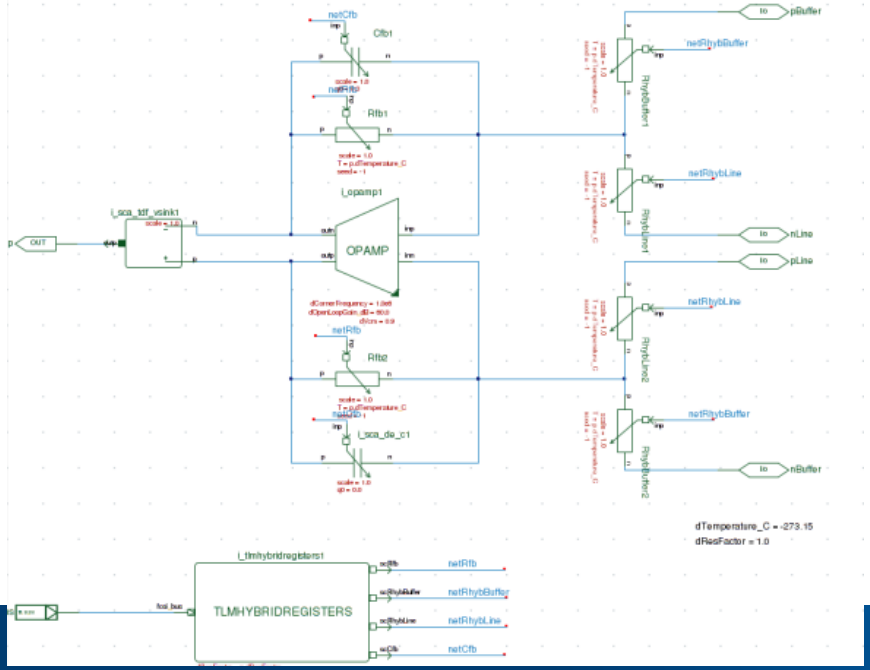
- Within SystemC-AMS I can use electrical elements or even better a predefined Transformer Symbol
- Coside provides the Netlister for SystemC and library elements
- SystemC is the simulation master and is used to verify the transfer function
- Registers are modelled in SystemC
- SystemC – AMS solves all the equations for the user

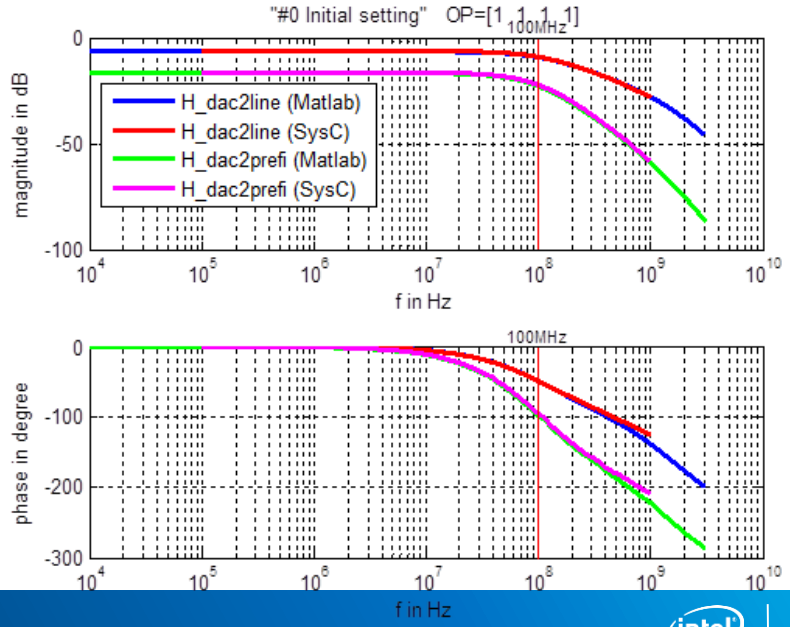




SystemC-AMS as Executable Specification for Mixed Signal Design Teams

- SystemC is the reference for Matlab model
- SystemC is the reference for Spice Simulation transistor model

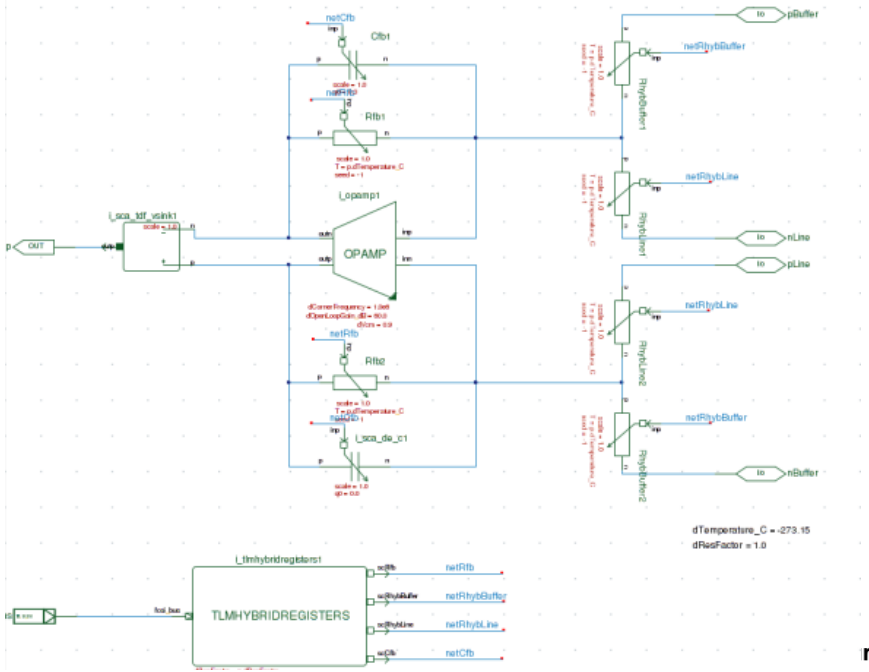


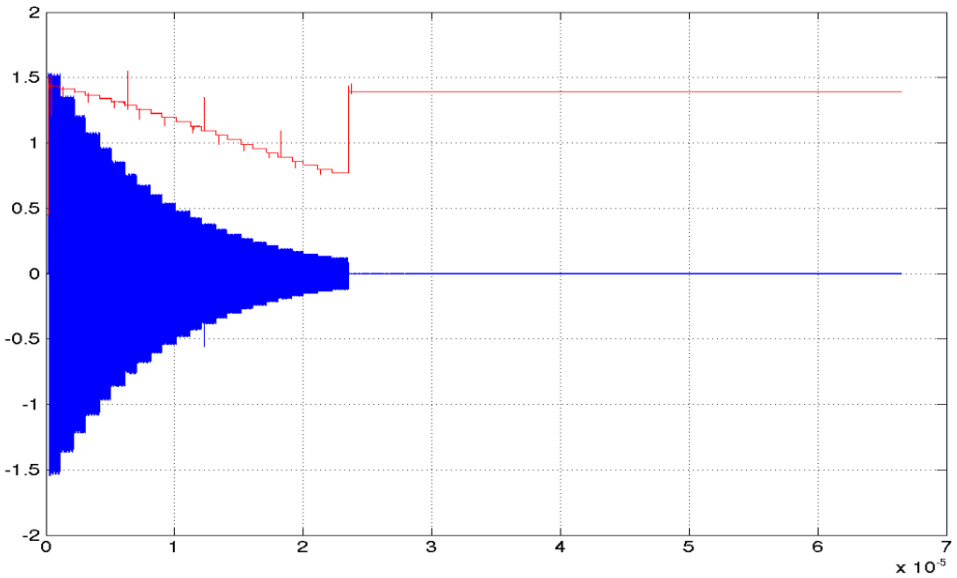


How to run AC simulation?

```
for (int sGain = 10; sGain >= -12; sGain--)
{
    std::stringstream Filename;
    std::stringstream Filename2;

    // set gain
    i_tlm_controller1->usTestCase = TLM_ASP_PGA_GAIN_E;
    i_tlm_controller1->sGain = sGain;

    sc_core::sc_start(0.5, sc_core::SC_US);

    // ac domain simulation
    tf2 -> set_mode(sca_ac_format(sca_util::SCA_AC_DB_DEG));
    Filename << "RESULTS/pga_tb/" << p_i_pga_tb.TrafoSelection << "_echo_" << sGain << ".dat";
    tf2 -> reopen(Filename.str());
    i_sin_src_tdf->p.ac_ampl = 1.0;
    i_sin_src_tdf2->p.ac_ampl = 0.0;
    //sca_ac_start (start, stop, number of samples, SCA_LIN or SCA_LOG)
    sca_ac_start(10.0e3, 1.0e9, 1000, sca_ac_analysis::SCA_LOG);

    tf2 -> set_mode(sca_ac_format(sca_util::SCA_AC_DB_DEG));
    Filename2 << "RESULTS/pga_tb/" << p_i_pga_tb.TrafoSelection << "_rxgain_" << sGain << ".dat";
    tf2 -> reopen(Filename2.str());
    i_sin_src_tdf->p.ac_ampl = 0.0;
    i_sin_src_tdf2->p.ac_ampl = 1.0;
    //sca_ac_start (start, stop, number of samples, SCA_LIN or SCA_LOG)
    sca_ac_start(10.0e3, 1.0e9, 1000, sca_ac_analysis::SCA_LOG);

    // time domain simulation
    tf2 -> reopen(osTracefile);

    sc_core::sc_start(0.5, sc_core::SC_US);

}
```

Simulation result in Time domain





re 15 Transient i_elec_trafo

AC Simulation Result

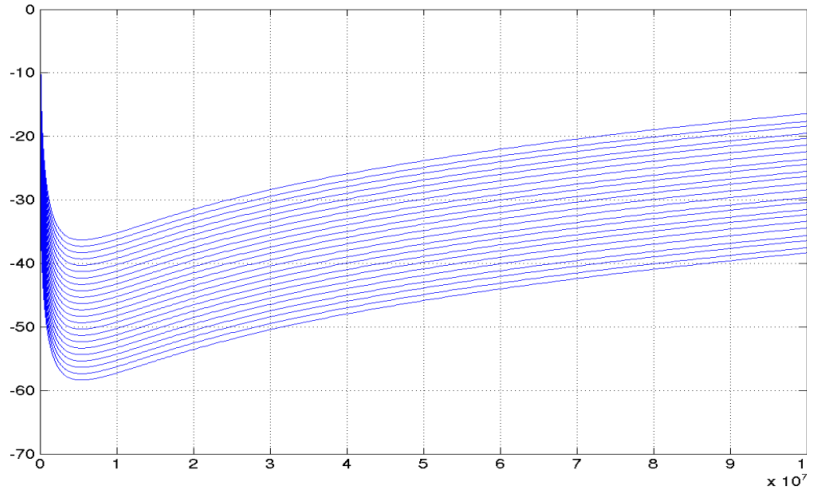


Figure 17 Echo Level i_elec_trafo

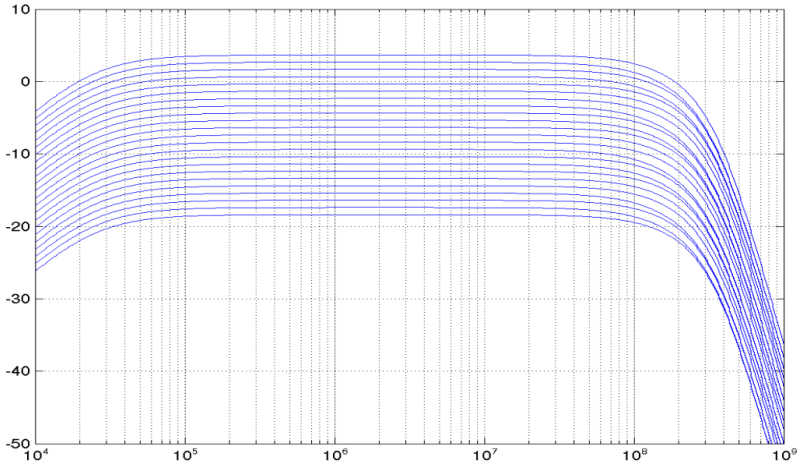


Figure 18 Receive Level i_elec_trafo

AC Simulation with different termination

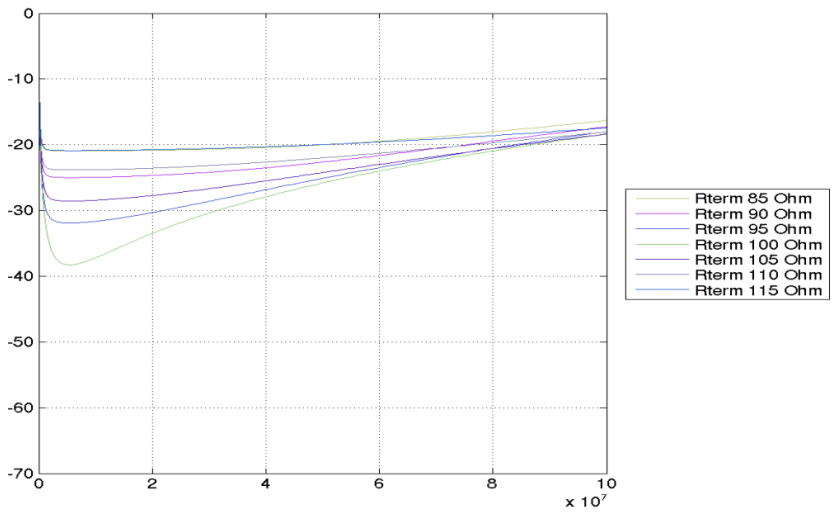
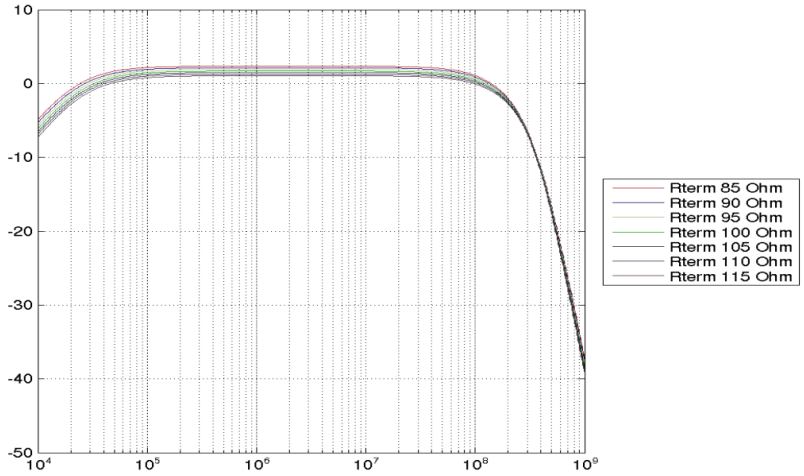


Figure 19 Echo Level vs. termination i_elec_trafo



20 Receive Level vs. termination i_elec_trafo

How to start a Noise Simulation

```
tf2 -> reopen("RESULTS/pga_tb/" + p_i_pga_tb.TrafoSelection + "_ac_noise.dat");
tf2 -> set_mode(sca_noise_format(sca_util::SCA_NOISE_ALL));
sca_ac_noise_start(1.0e6, 100.0e6, 100, sca_ac_analysis::SCA_LIN);
```

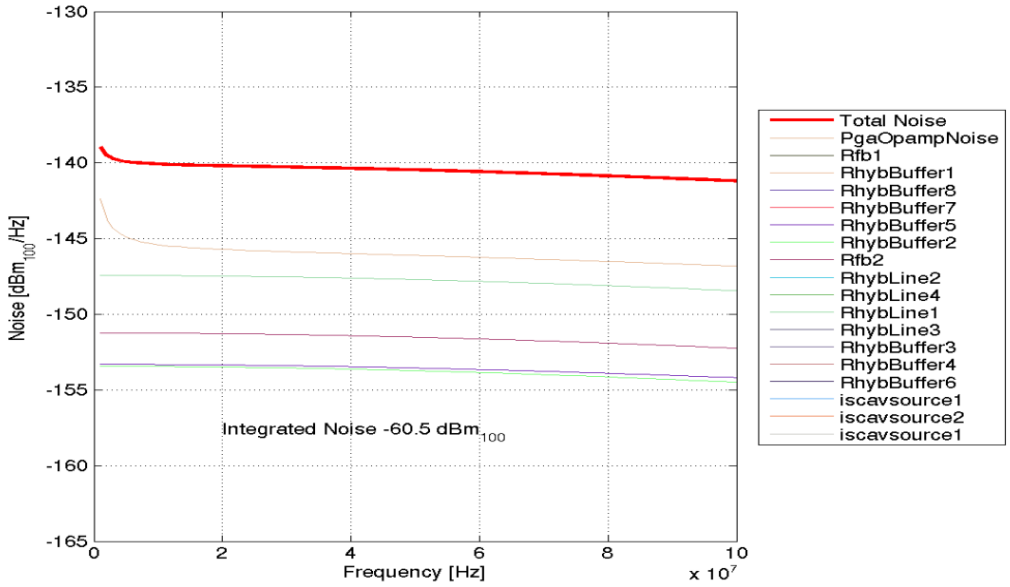
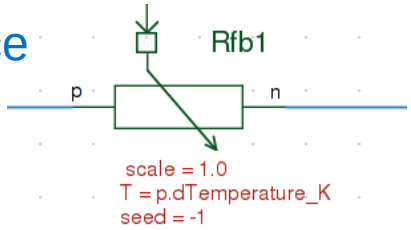


Figure 21 Noise i_elec_trafo

How to add noise sources

- Automatically inserted noise source



- Manually inserted noise source

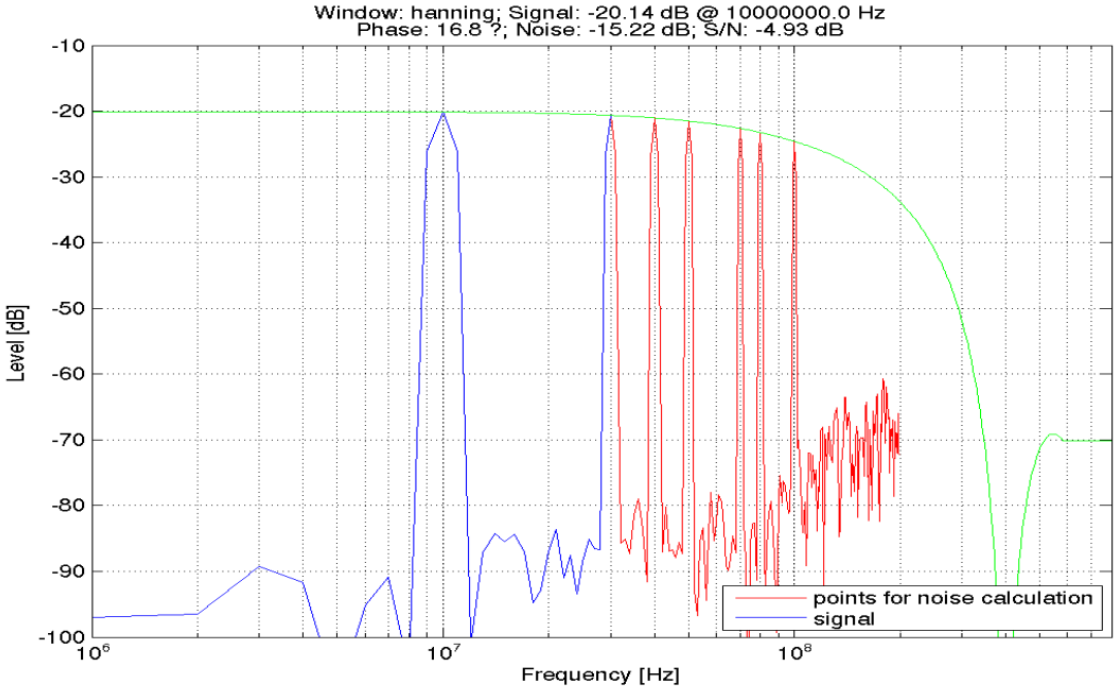
```
// remove wire from input pin and connect net_noise instead, return old wire
SC_RECONNECT_PORT(*input_pin, *net_noise, net_orig);

// new electrical source
sca_eln::sca_tdf_vsource* vnoise;
vnoise = new sca_eln::sca_tdf_vsource("vnoise");
vnoise->p.bind(*net_noise);
vnoise->n.bind(*net_orig);
vnoise->inp.bind(*tdf_noise);

// new noise source
upd_noise_src_tdf::params p_noise_src;
p_noise_src.nfft = 16384;
p_noise_src.psd_mask = "utilities/noise_files/pga/inputreffered_noise_inV2_HZ_Typ70C_VDCin645mV.dat";
p_noise_src.z0 = 100;

upd_noise_src_tdf *PgaOpampNoise;
PgaOpampNoise = new upd_noise_src_tdf("PgaOpampNoise", p_noise_src);
PgaOpampNoise->tdf_o.bind(*tdf_noise);
```

How to implement AC and Noise in Primitive Blocks

```
//////////////////////////////////////////////
// method ac_processing                     //
//////////////////////////////////////////////
void adc_comb::ac_processing()
{
    sc_time dTs;
    sca_complex h;

    dTs = sc_time(1.0/p.dAdcCombDataRate, SC_SEC);

    // third order, decimate by 4
    h = pow((1.0 - sca_ac_z(-4, dTs)) /
            (1.0 - sca_ac_z(-1, dTs)) / 4.0, 3.0);

    sca_ac(tdfout) = h * sca_ac(tdfin);

}
```

```
//////////////////////////////////////////////
// method ac_processing                     //
//////////////////////////////////////////////
void quant::ac_processing()
{
    sca_util::sca_complex cE;


    // for AC output = input
    sca_ac(tdfout) = sca_ac(tdfin);


    // calculate quantizing noise
    // The quantizing noise is distributed over FS/2
    // E = q^2 / 12 * 2 / FS
    // e = q * sqrt(2/12/FS)
    cE = s.dLSB * sqrt(2.0/12.0/s.dFs);

    sca_ac_noise(tdfout) = cE;

}
```

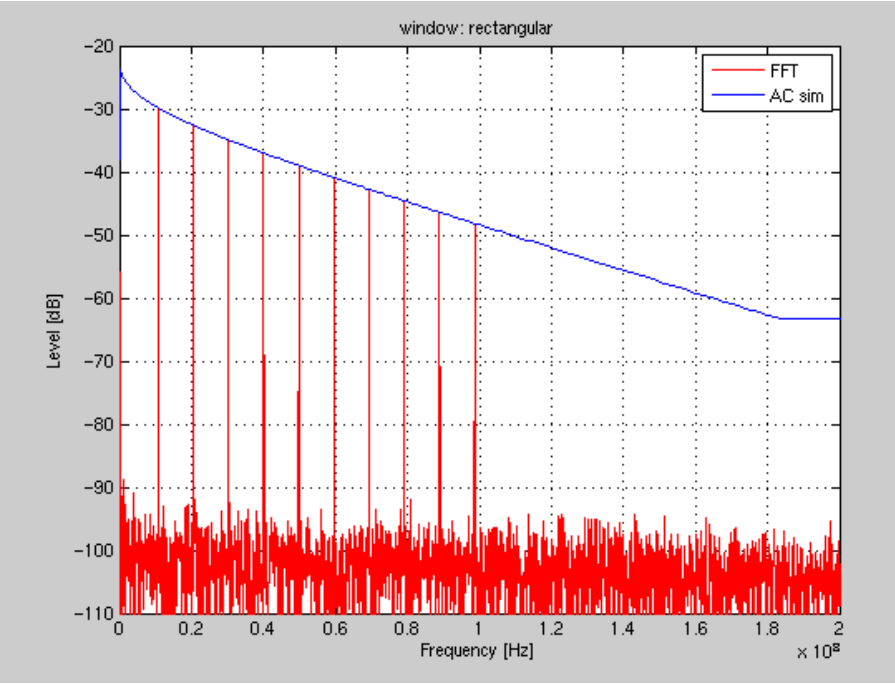
Compare Multitone time domain signal with AC simulation

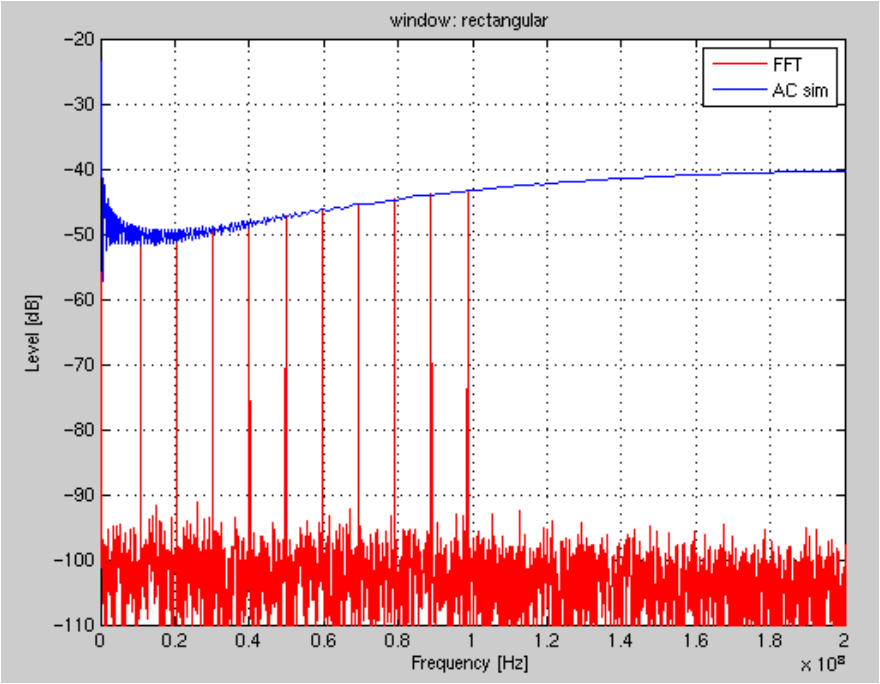


Line Model using Scattering Parameter

- Coside offers a Scattering parameter model
  - S-parameter can easily be generated out of existing Matlab models
  - Alternatively measurement results could be use
- The line model can be used to simulate in
  - Time Domain
  - Frequency Domain (AC simulation)

Simulation Results of Transfer functions with Line Model





Conclusion:
Simulation of AFE with SystemC-AMS

- AC and noise simulation
  - Powerful tool to analyze the Analog blocks
  - Reference for Matlab and Spice Simulation
  - Easy to use, fast build up
- SystemC is one of the standard solutions for System Simulation
  - With SystemC – TLM an easy integration of Bus models and Instruction Set Simulator is provided
  - With SystemC-AMS we extend the possibility to integrate Analog Models into the System Simulation
  - Possibility of end to end simulation including our AFE models
  - Check the FW code already in an early stage