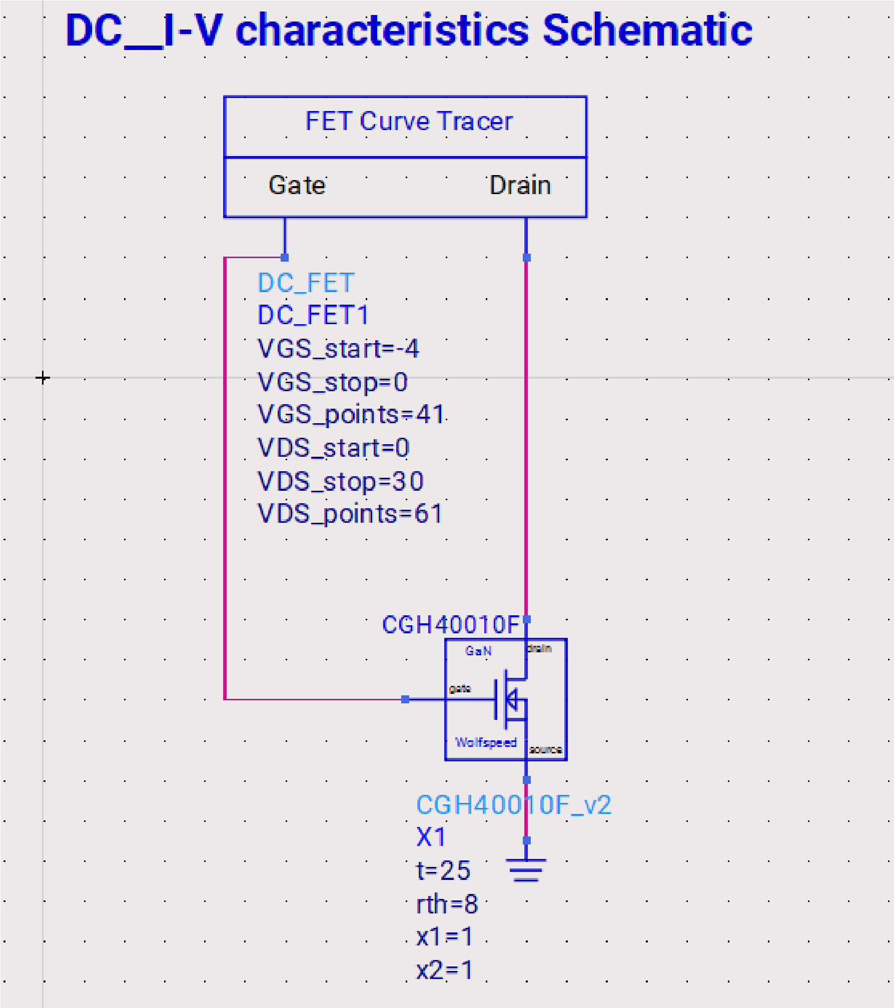
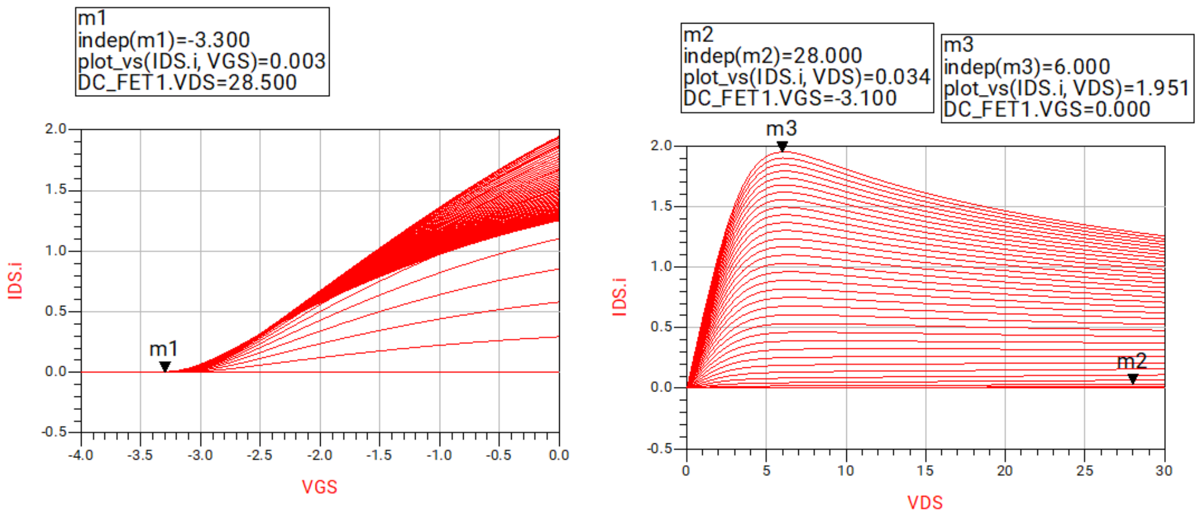
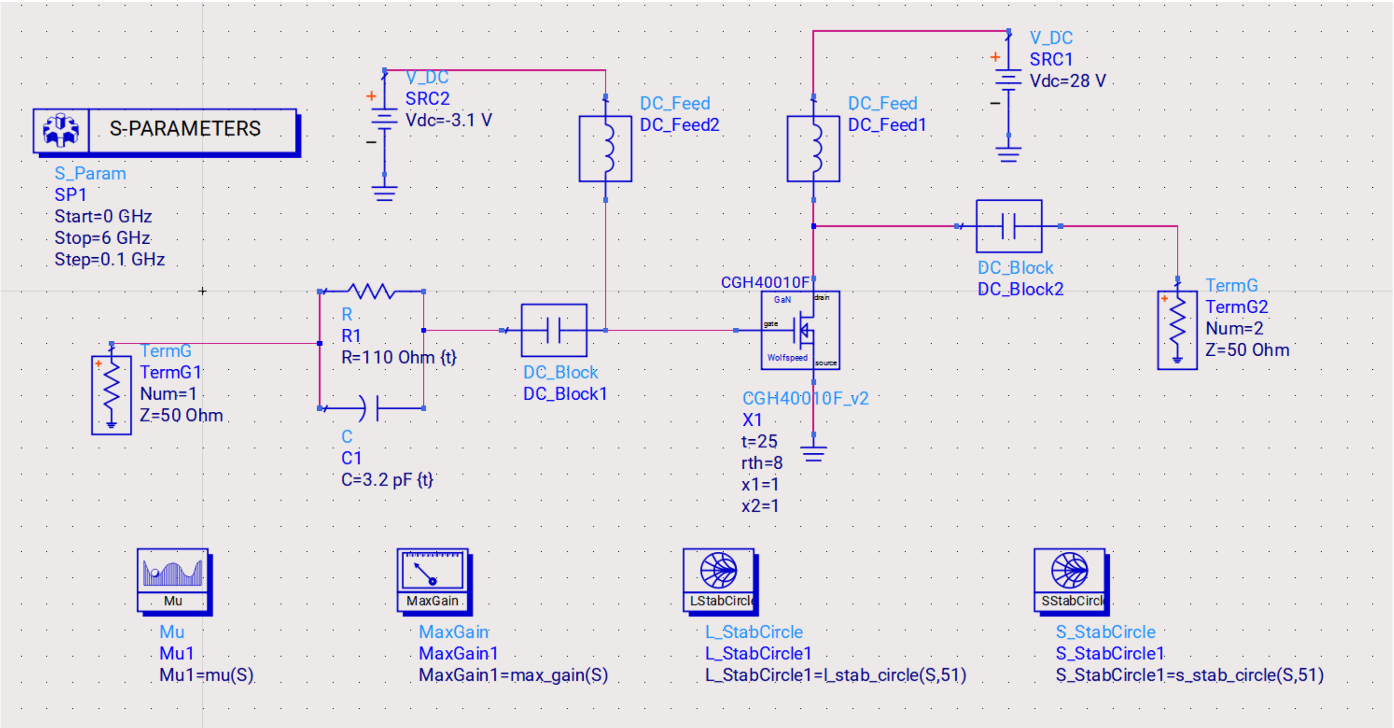


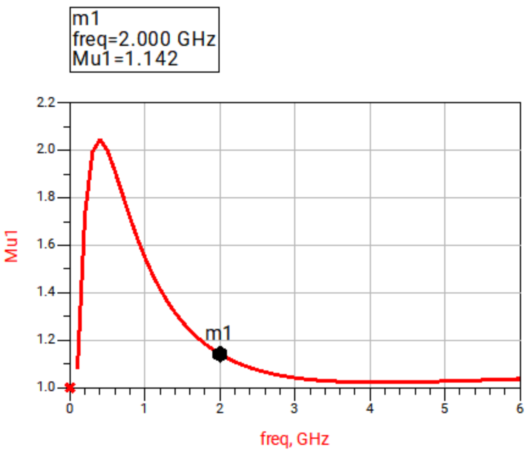
DC__I-V characteristics Results

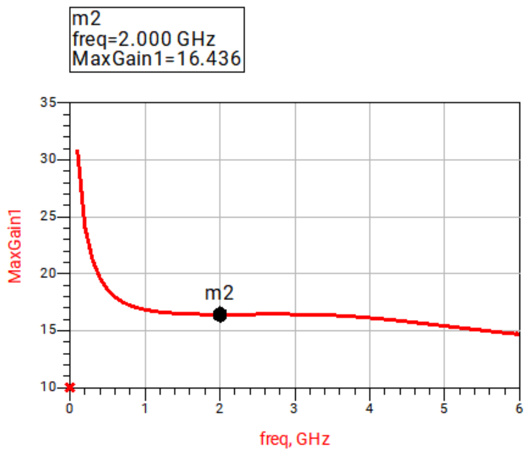


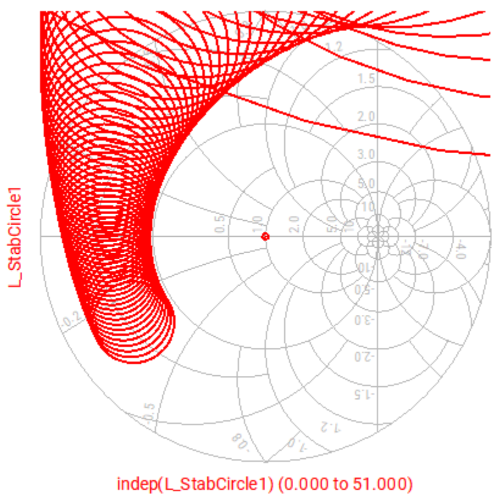
Stability Schematic:

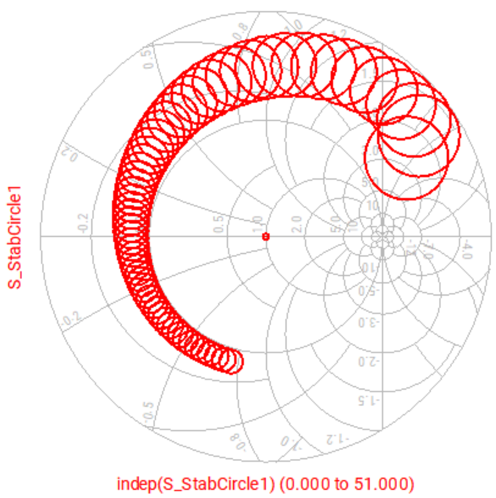


Stability Results:

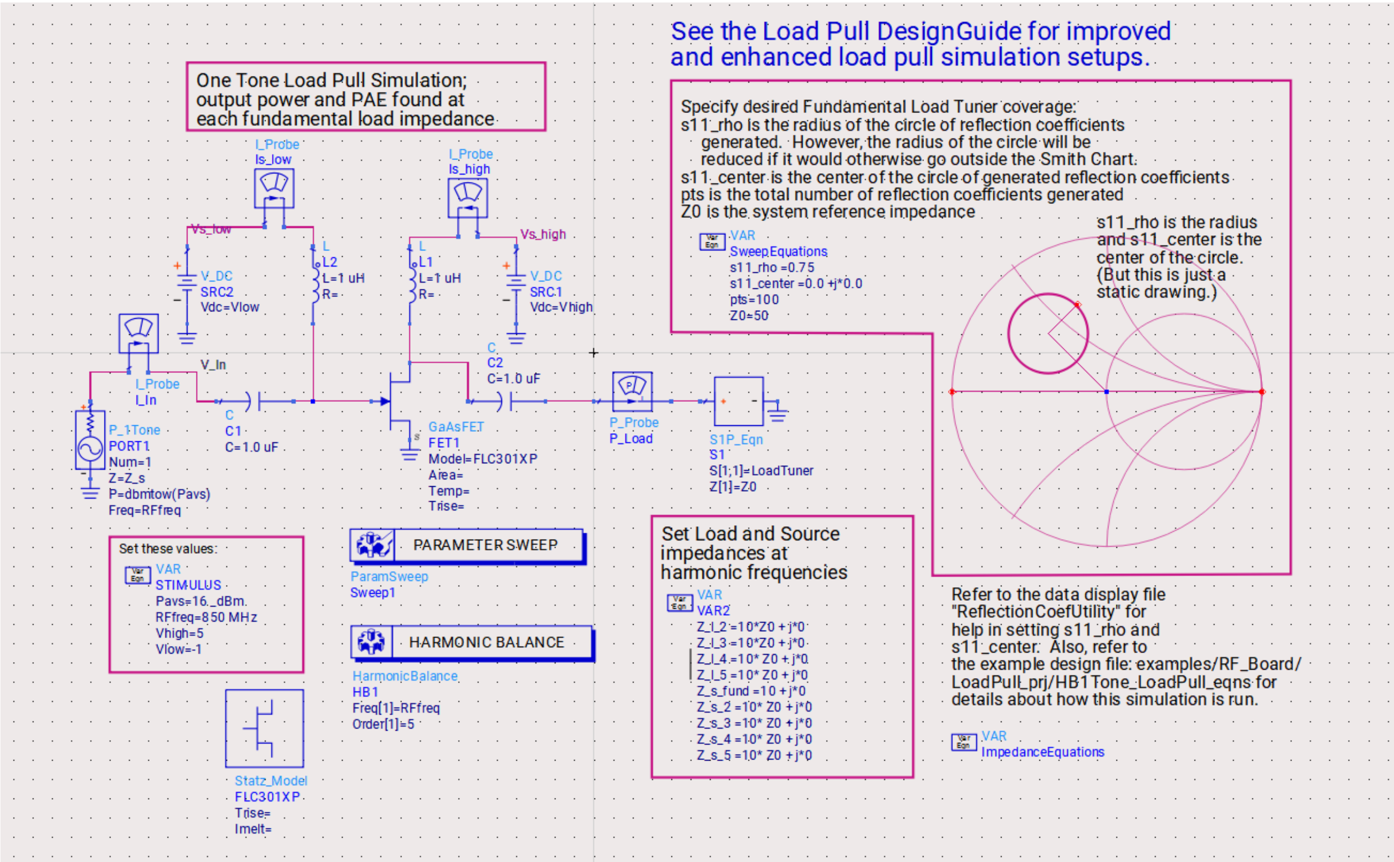




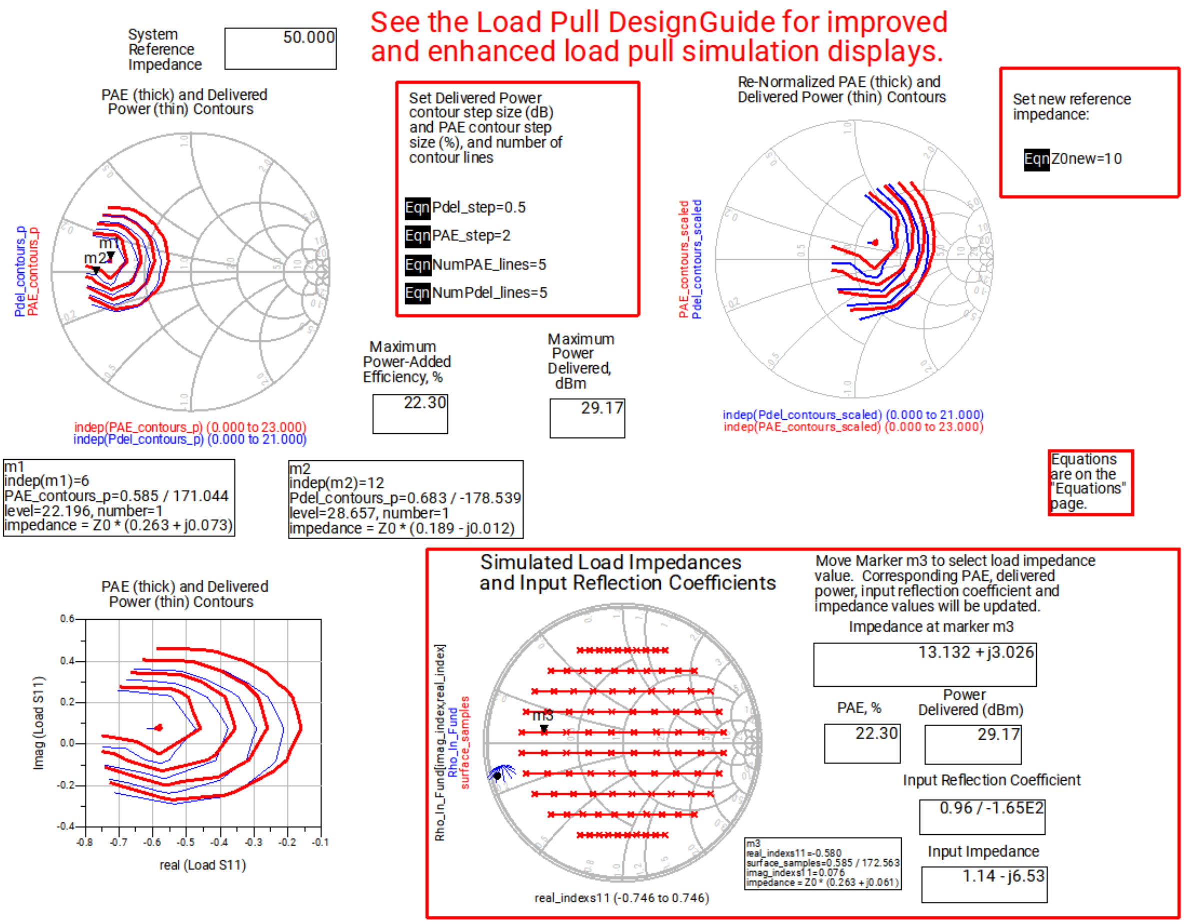




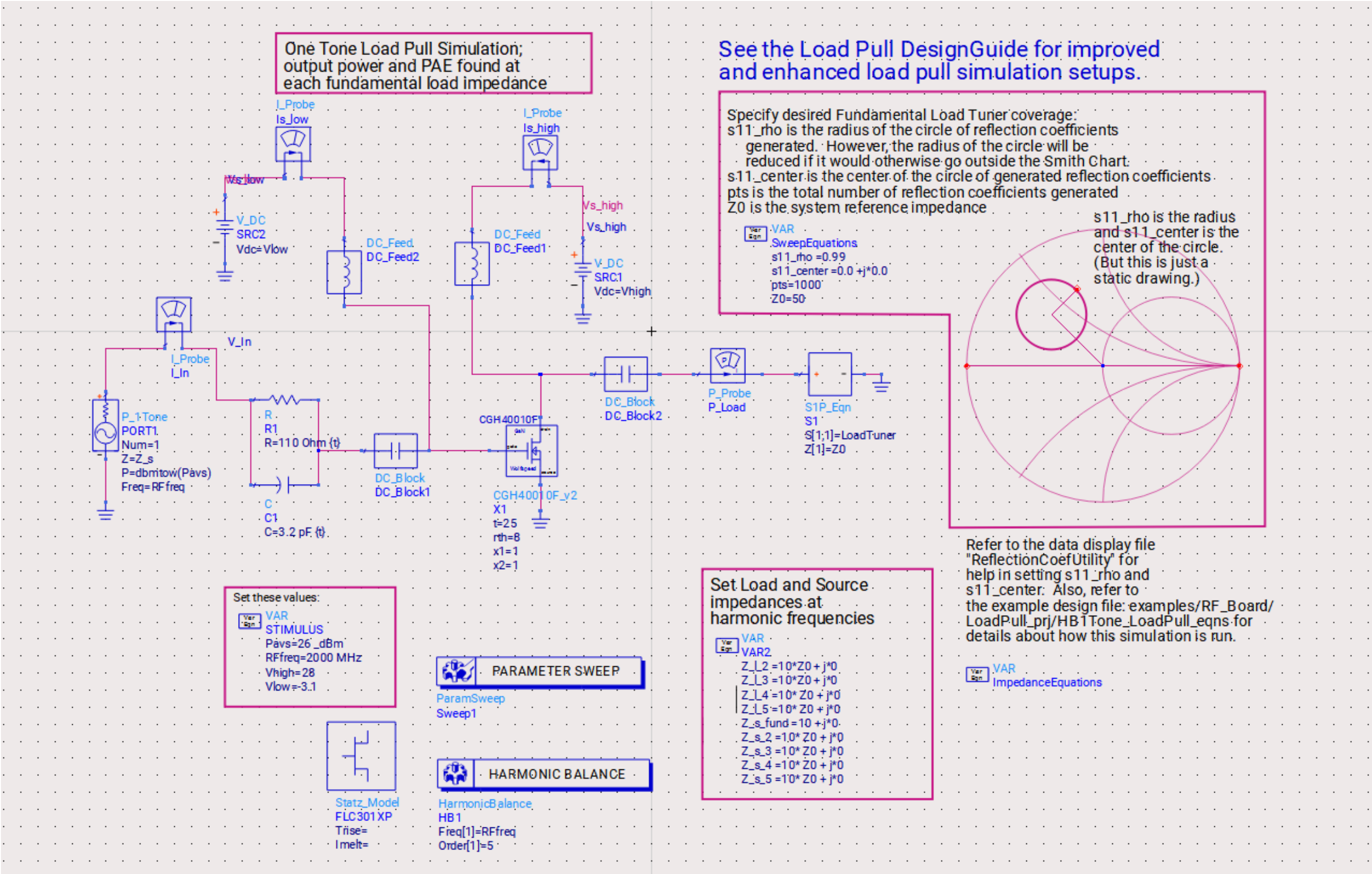
Load Pull Design Guide Schematic Design:



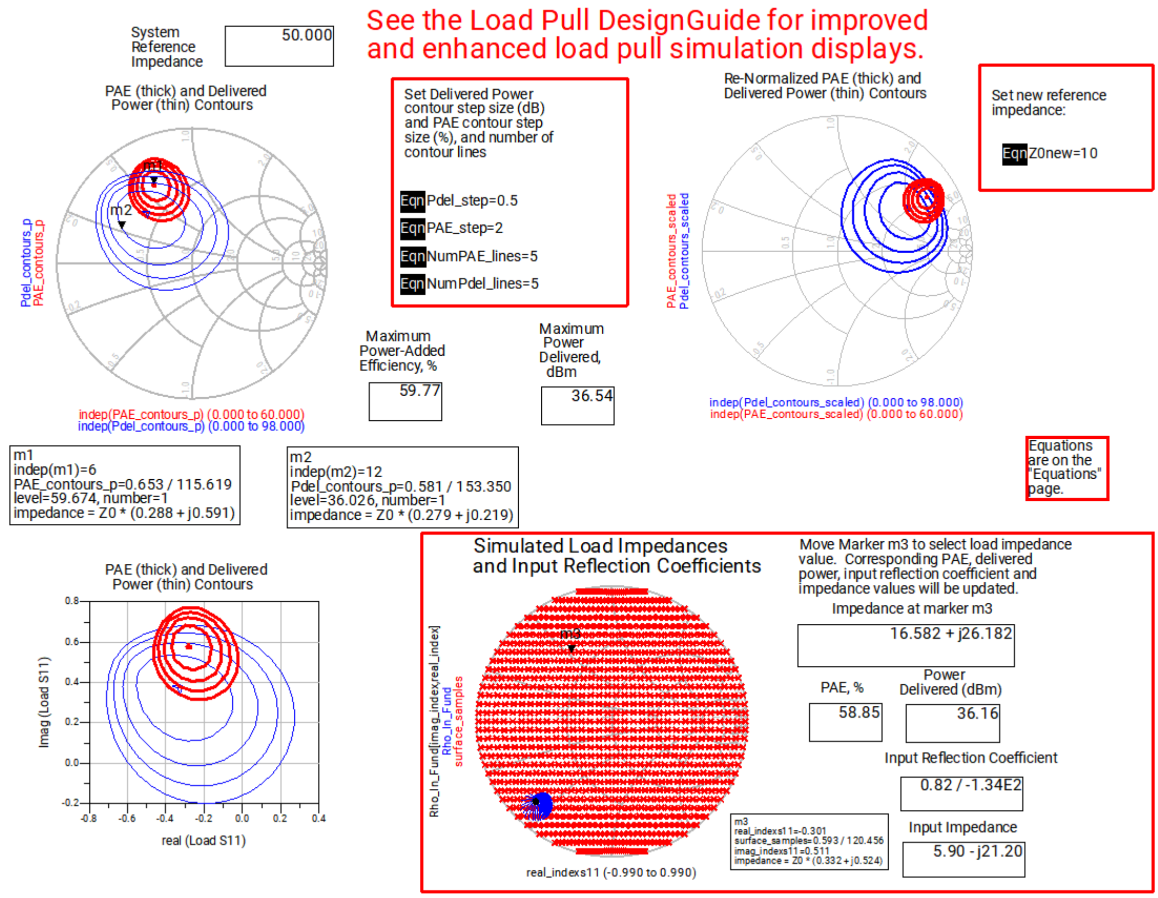
Load Pull Design Guide Schematic Design Results:



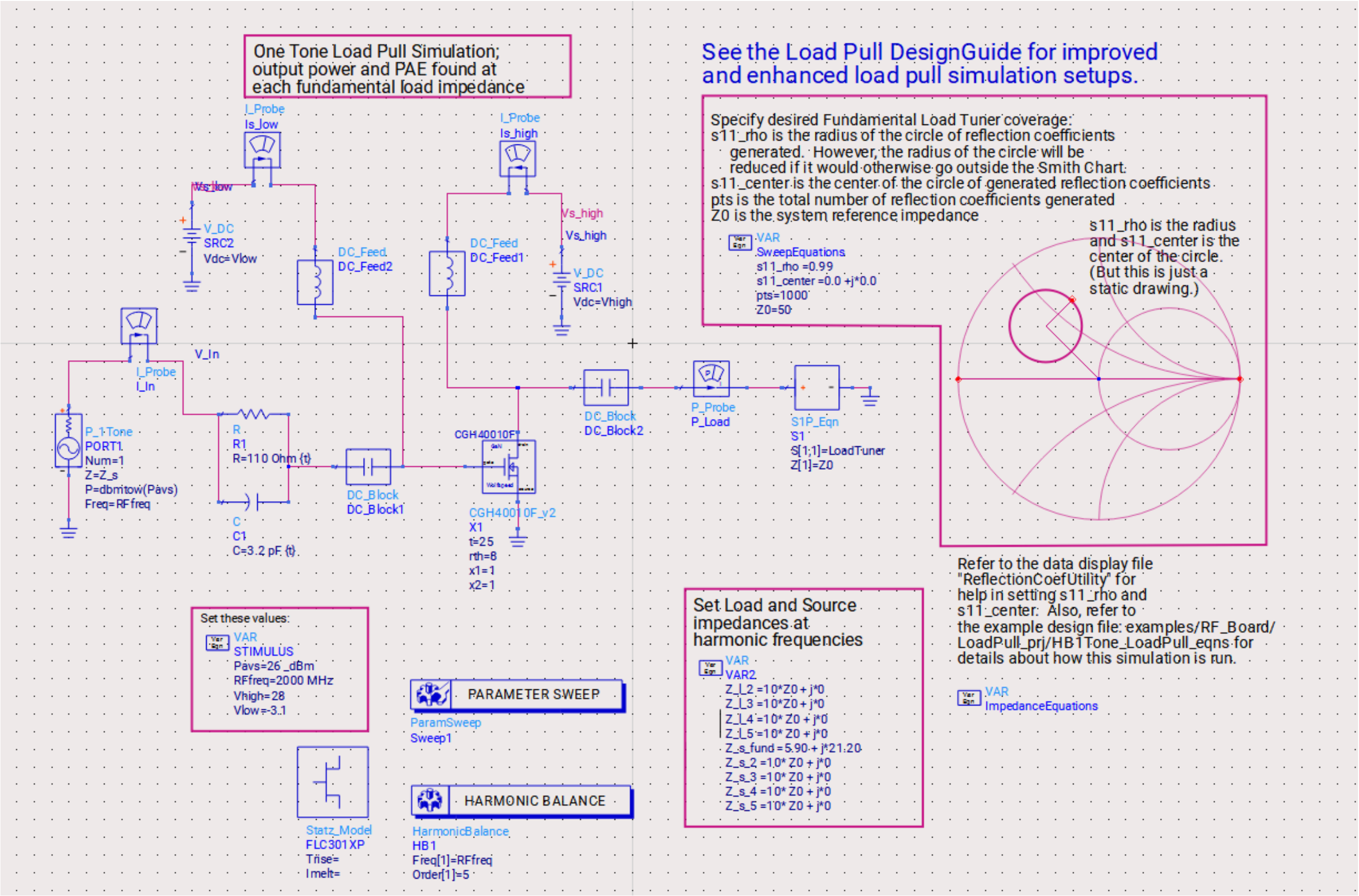
Load Pull Design Guide Replace with Stability Components Schematic:



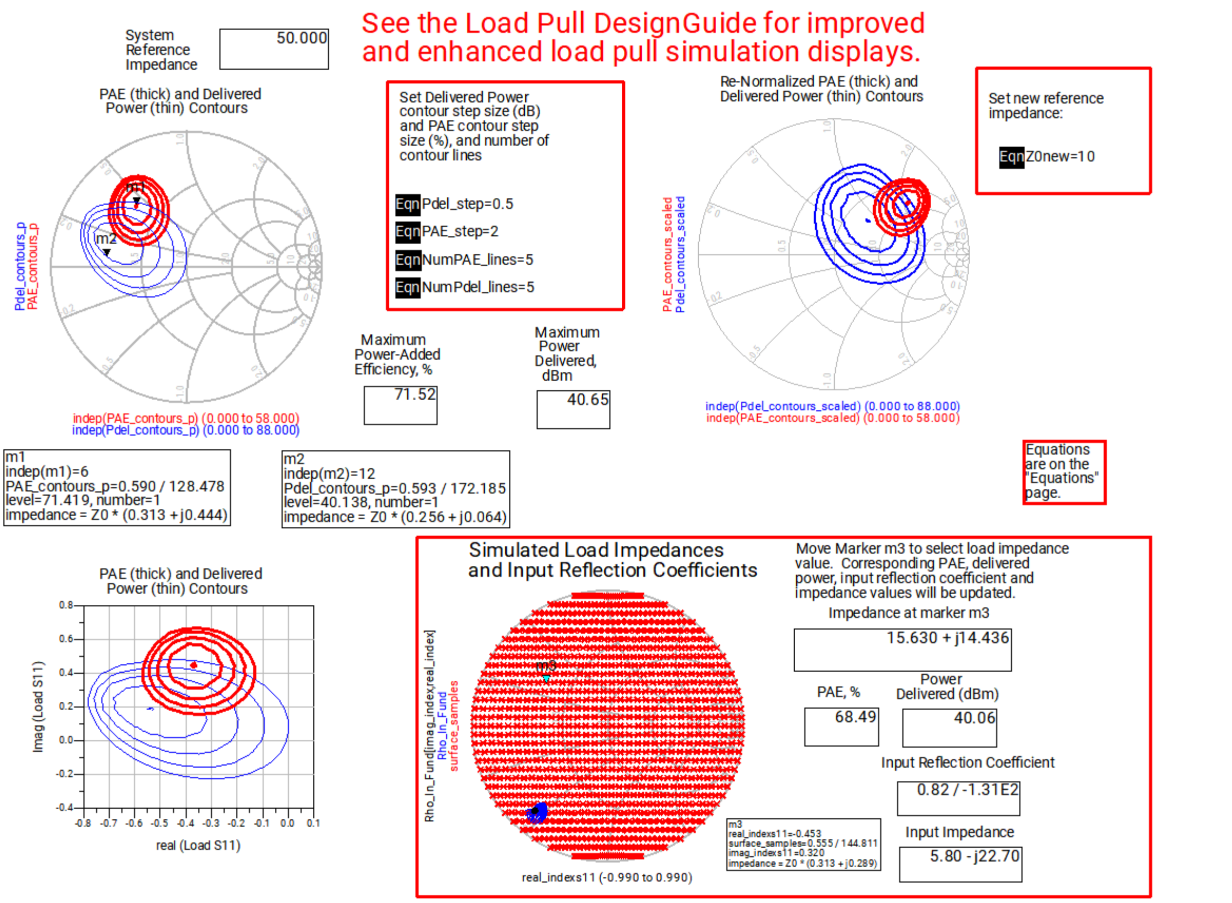
Load Pull Design Guide Replace with Stability Components Results:



Load Pull Design Guide Replace with Stability Components and Zin Value:



Load Pull Design Guide Replace with Stability Components and Matched With PAE % and Power Delivered (dBm):



DC Power Calculations
The exists() function checks to be sure the corresponding piece of data is in the dataset. If it is not, then the function returns 0.

Eqn Vs_l=exists("real(Vs_low[0])")

Eqn Vs_h=exists("real(Vs_high[0])")

Eqn Is_h=exists("real(Is_high.i[0])")

Eqn Is_l=exists("real(Is_low.i[0])")

Eqn Pdc=Is_h*Vs_h +Is_l*Vs_l +1e-20

Power Delivered and Power-Added Efficiency Calculations

Eqn Pdel_Watts=P_Load.p[1]

Pavs is the available source power, set on the schematic, and passed into the dataset using the Harmonic Balance controller.

Eqn Pavs_Watts=10**((Pavs[0,0]-30)/10)

Eqn ~~PAE=100*(Pdel_Watts-Pavs_Watts)/Pdc~~

Eqn P_In_Watts=0.5*real(V_In[1]*conj(I_In.i[1]))

This power-added efficiency equation that uses the power absorbed at the input is considered to be a better, more accurate figure of merit than the one that uses the power available from the source.

Eqn PAE=100*(Pdel_Watts-P_In_Watts)/Pdc

Eqn Pdel_dBm = Pdel_Watts>0? 10*log10(Pdel_Watts)+30 : -200

Power-Added Efficiency contour calculations

Eqn PAEmax=max(max(PAE))

Eqn PAE_contours=contour(PAE,PAEmax-0.1-[0::(NumPAE_lines-1)]*PAE_step)

This -0.1 term is included so the contour at the maximum is a small circle, rather than a single point.

Conversion of the contours to polar coordinates for plotting on Smith Chart

Eqn PAE_contours_p=[indep(PAE_contours)+j*PAE_contours]

Conversion of the contours (which are actually curves of complex reflection coefficients that produce a constant PAE) to impedances

Eqn Zl_PAE=zin(PAE_contours_p,Z0[0,0])

Z0 is the reference impedance, set on the schematic, and passed into the dataset using the Harmonic Balance controller.

Re-scaling the contours for plotting on a Smith Chart with a different reference impedance

Eqn PAE_contours_scaled=(Zl_PAE-conj(Z0new))/(Zl_PAE+Z0new)

surface_samples are the actually-simulated reflection coefficients from which the contours are generated. real_indexs11 and imag_indexs11 are the swept variables in the simulation.

Eqn surface_samples=real_indexs11+j*expand(imag_indexs11)

Eqn Z_In_Fund=V_In[1]/I_In.i[1]

Eqn Rho_In_Fund=(Z_In_Fund-conj(Z0))/(Z_In_Fund+Z0)

Power Delivered contour calculations

Eqn Pdelmax = max(max(Pdel_dBm))

Eqn Pdel_contours=contour(Pdel_dBm,Pdelmax-.01-[0::(NumPdel_lines-1)]*Pdel_step)

Conversion of the contours to polar coordinates for plotting on Smith Chart

Eqn Pdel_contours_p=[indep(Pdel_contours) +j*Pdel_contours]

Conversion of the contours (which are actually curves of complex reflection coefficients that produce a constant delivered power) to impedances

Eqn Zl_Pdel=zin(Pdel_contours_p,Z0[0,0])

Re-scaling the contours for plotting on a Smith Chart with a different reference impedance

Eqn Pdel_contours_scaled=(Zl_Pdel-conj(Z0new))/(Zl_Pdel+Z0new)

imag_index and real_index are used to calculate the PAE and power delivered that correspond to the reflection coefficient at the m3 marker location.

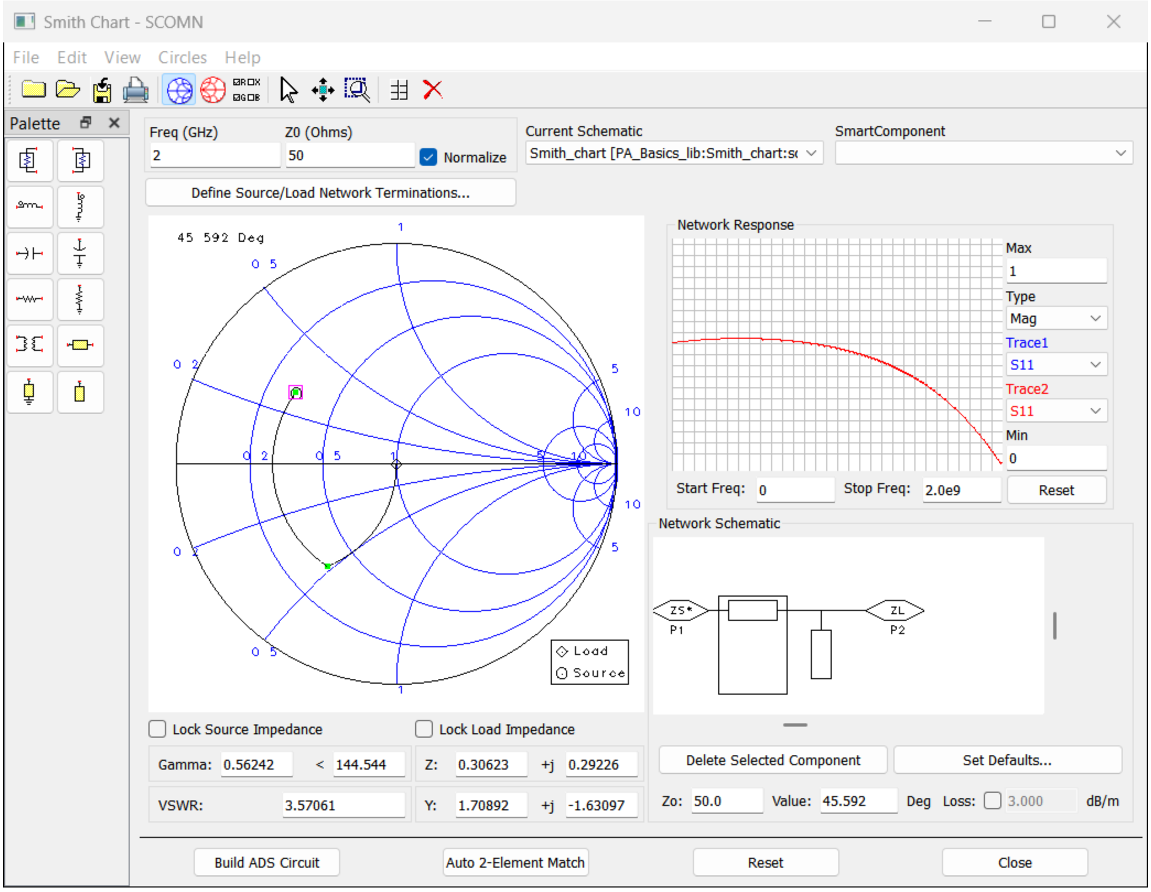
Eqn imag_index=find_index(imag_indexs11,imag(m3))

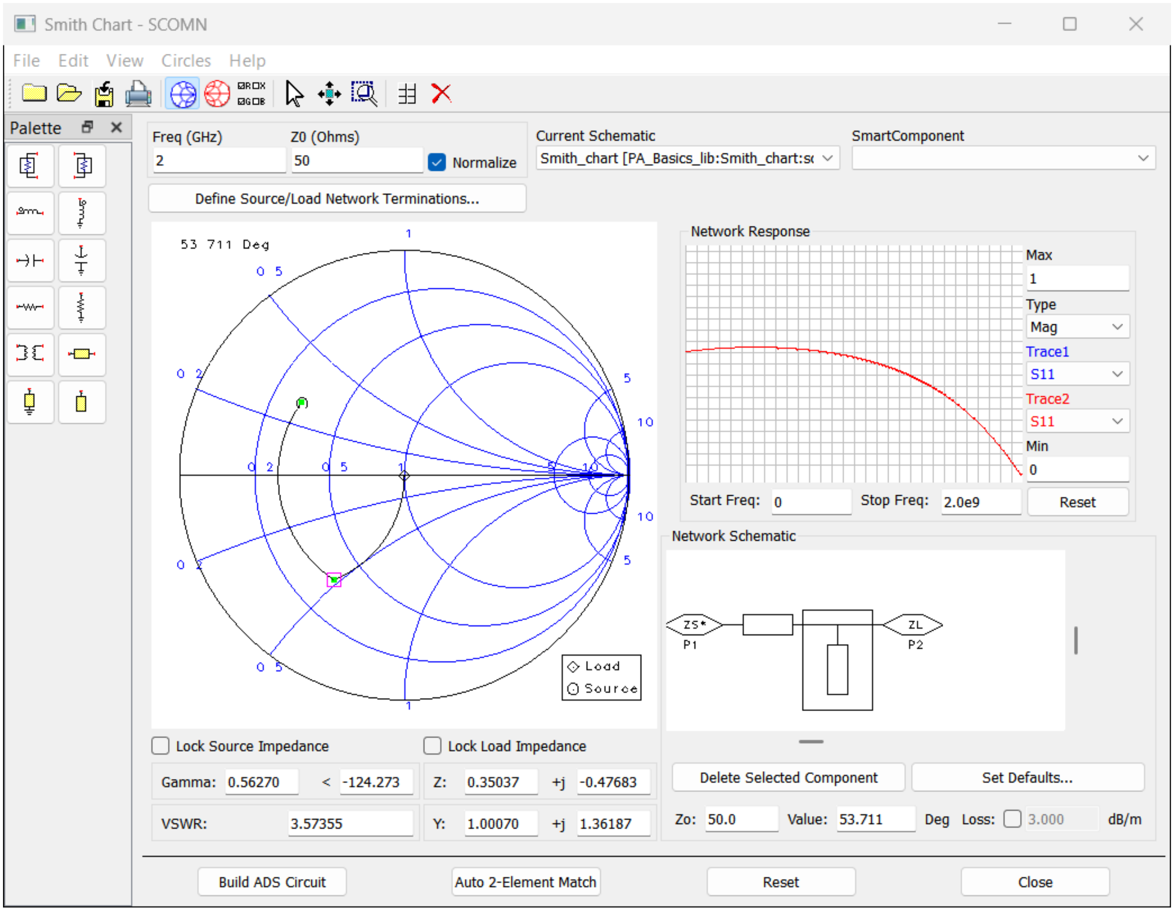
Eqn real_index=find_index(real_indexs11[imag_index,::],real(m3))

This calculates the impedance at marker m3.

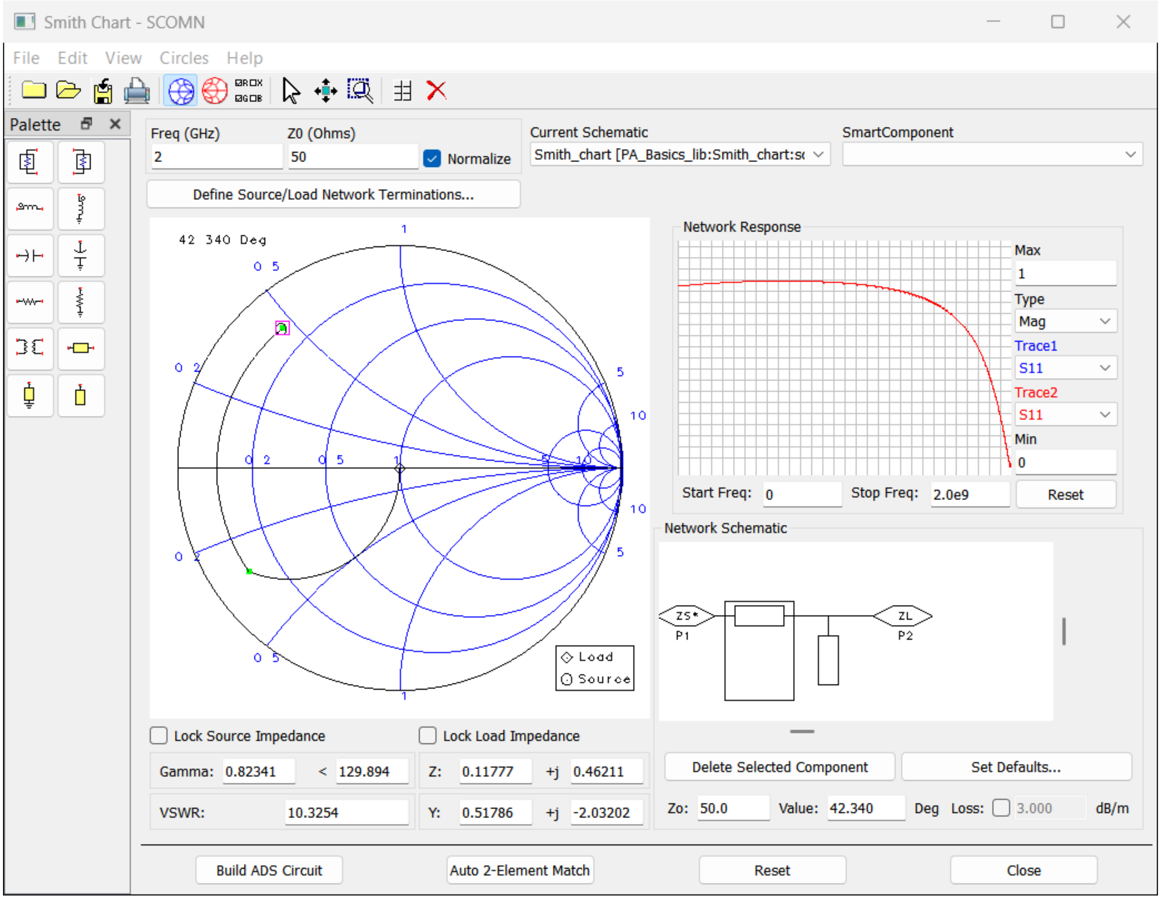
Eqn Z_at_m3=(conj(Z0[0,0])+Z0[0,0]*m3)/(1-m3)

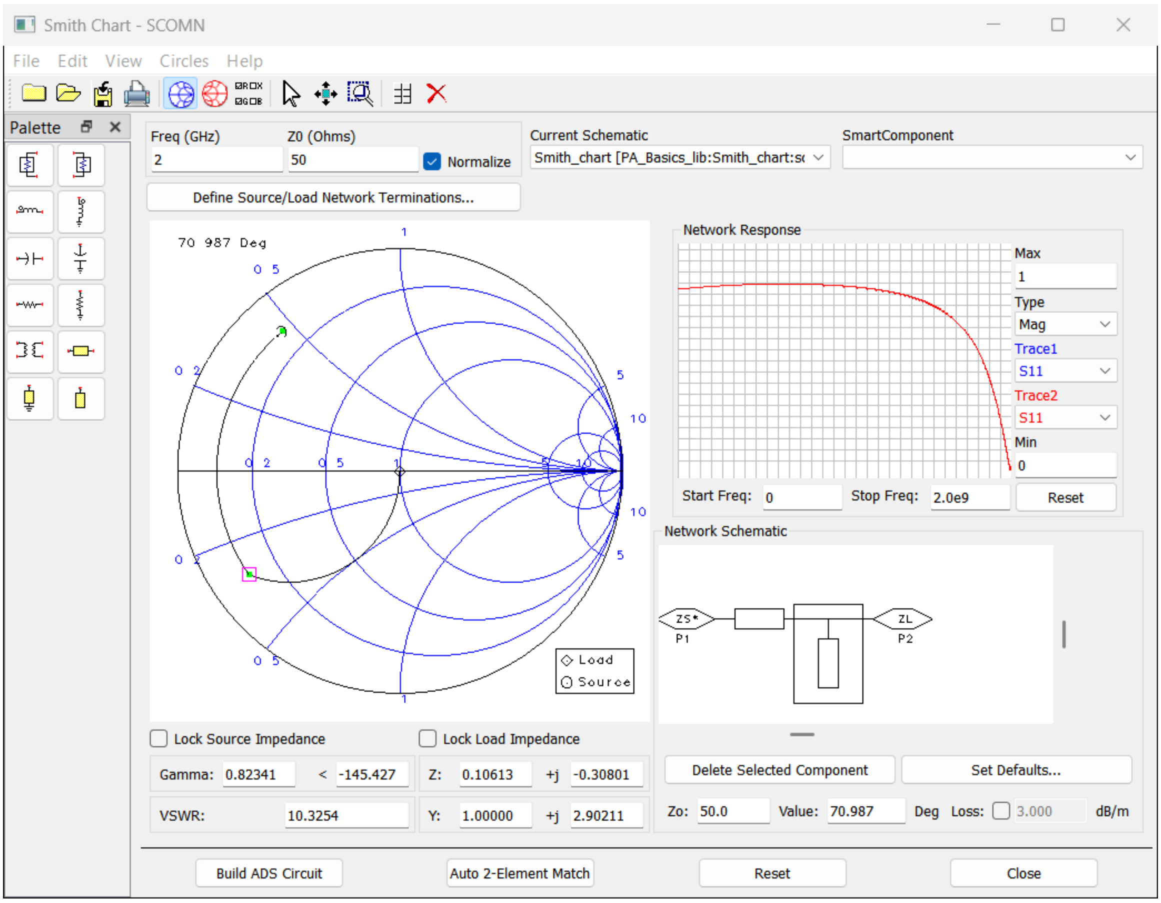
OUTPUT MATCHING NETWORK AT 2GHz:



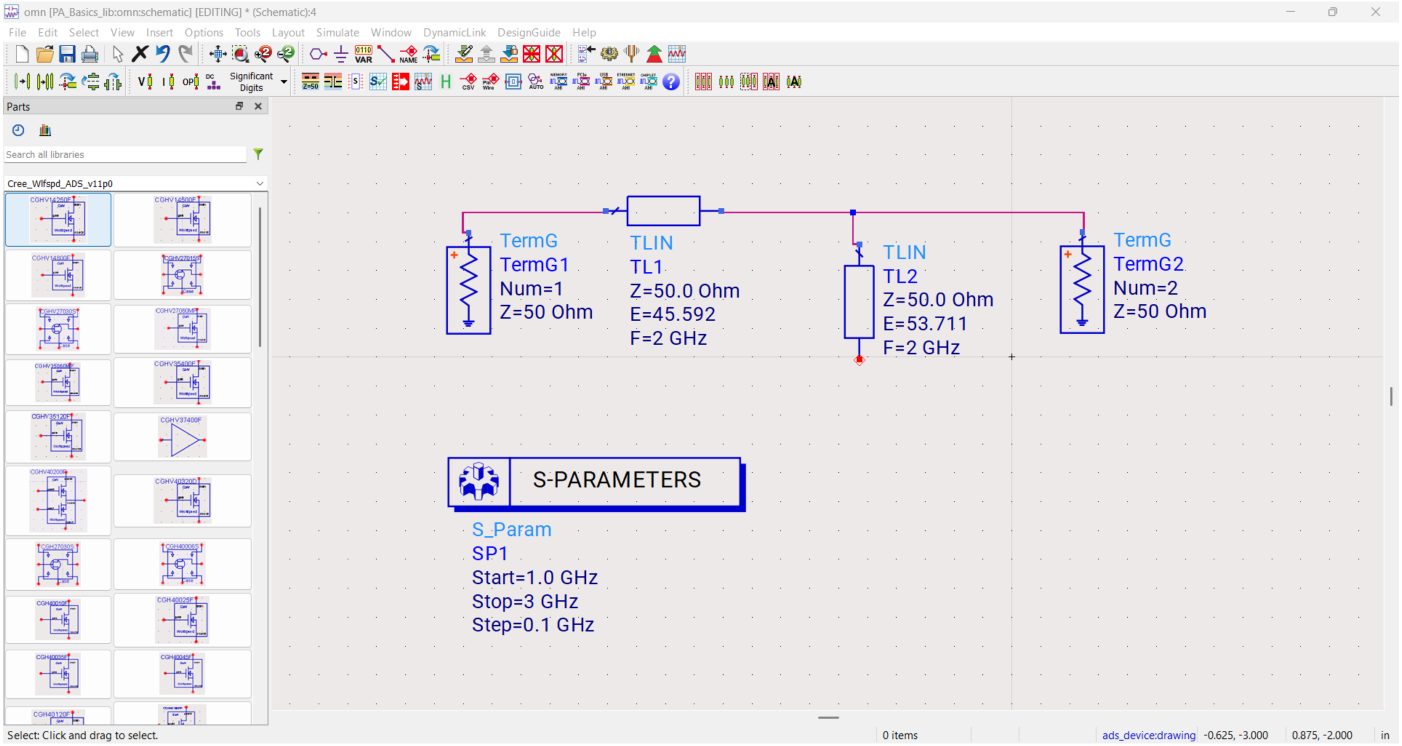


INPUT MATCHING NETWORK AT 2GHz:

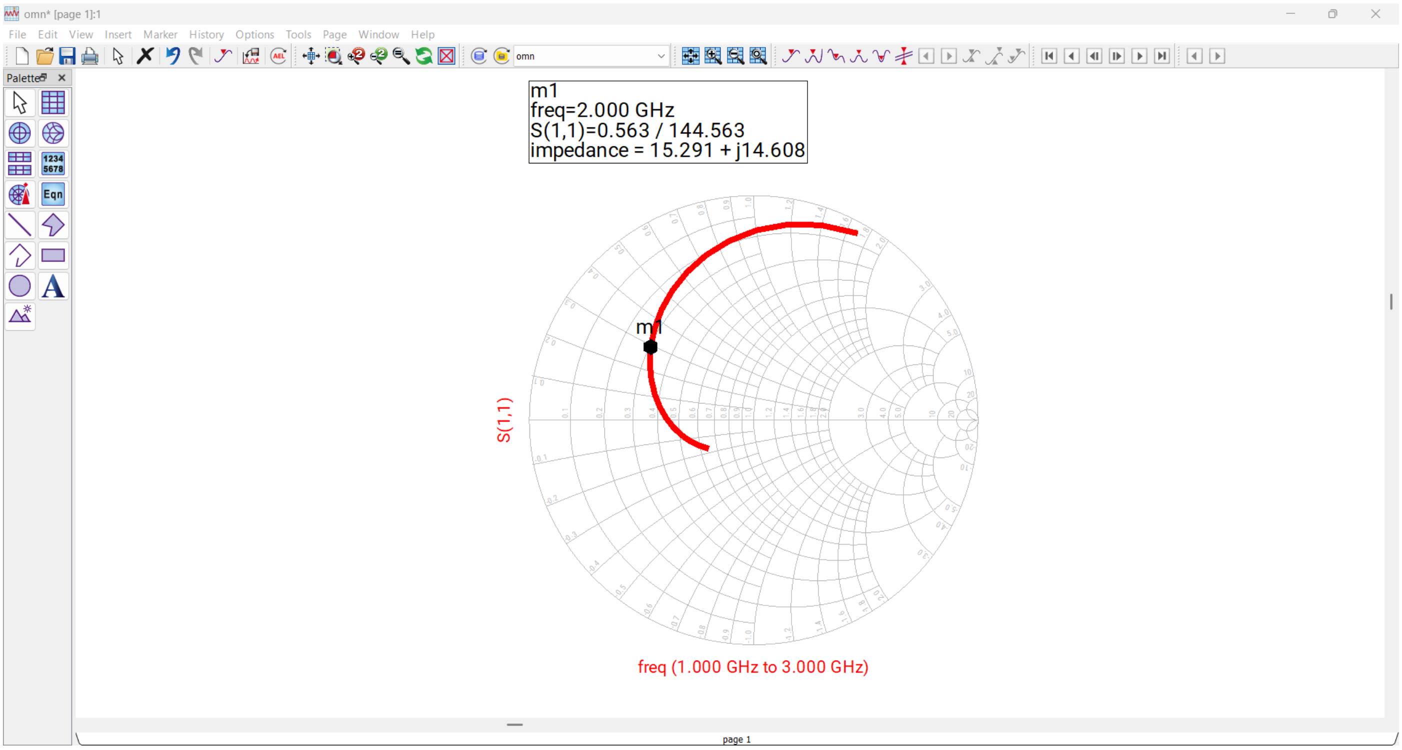




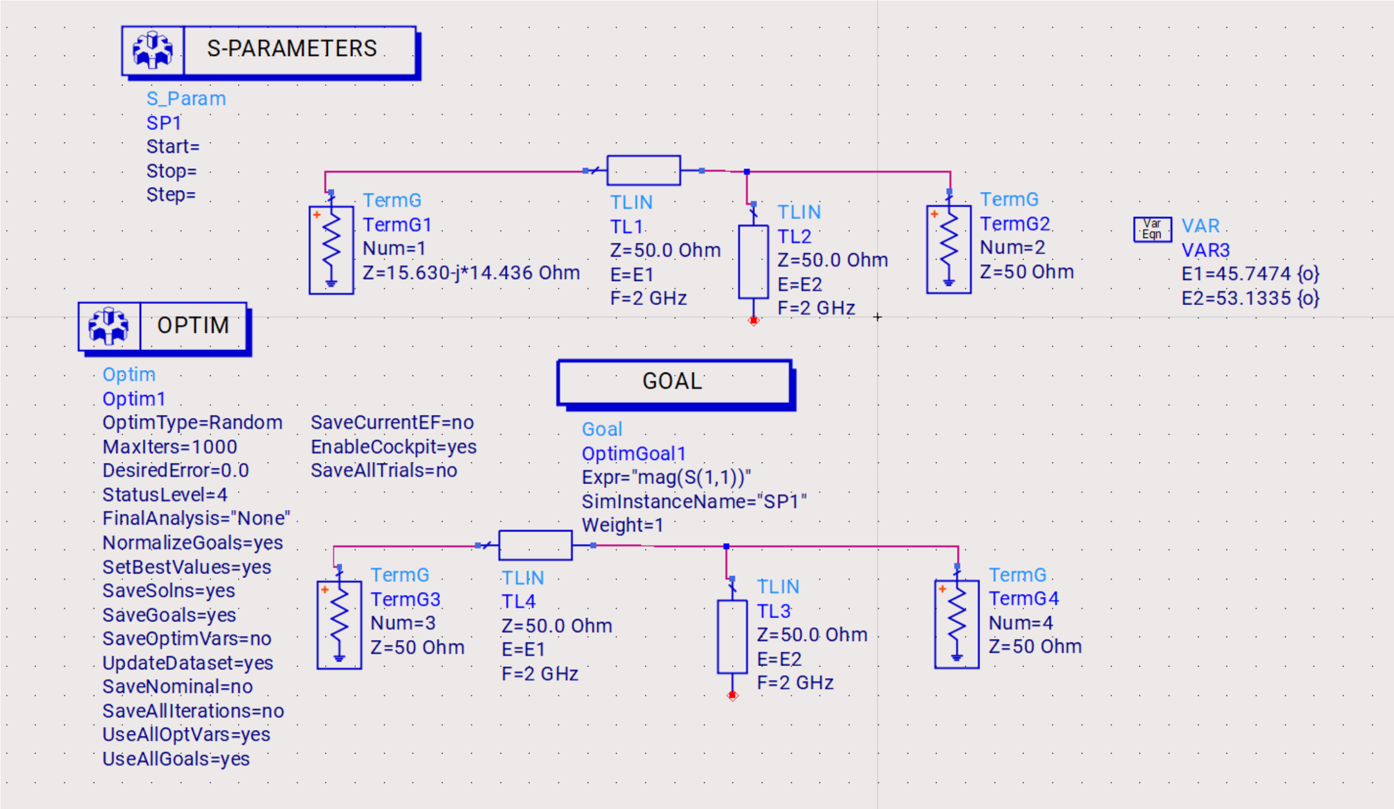
OUTPUT MATCHING NETWORK (WITHOUT OPTIMIZE) SCHEMATICS At 2GHz:



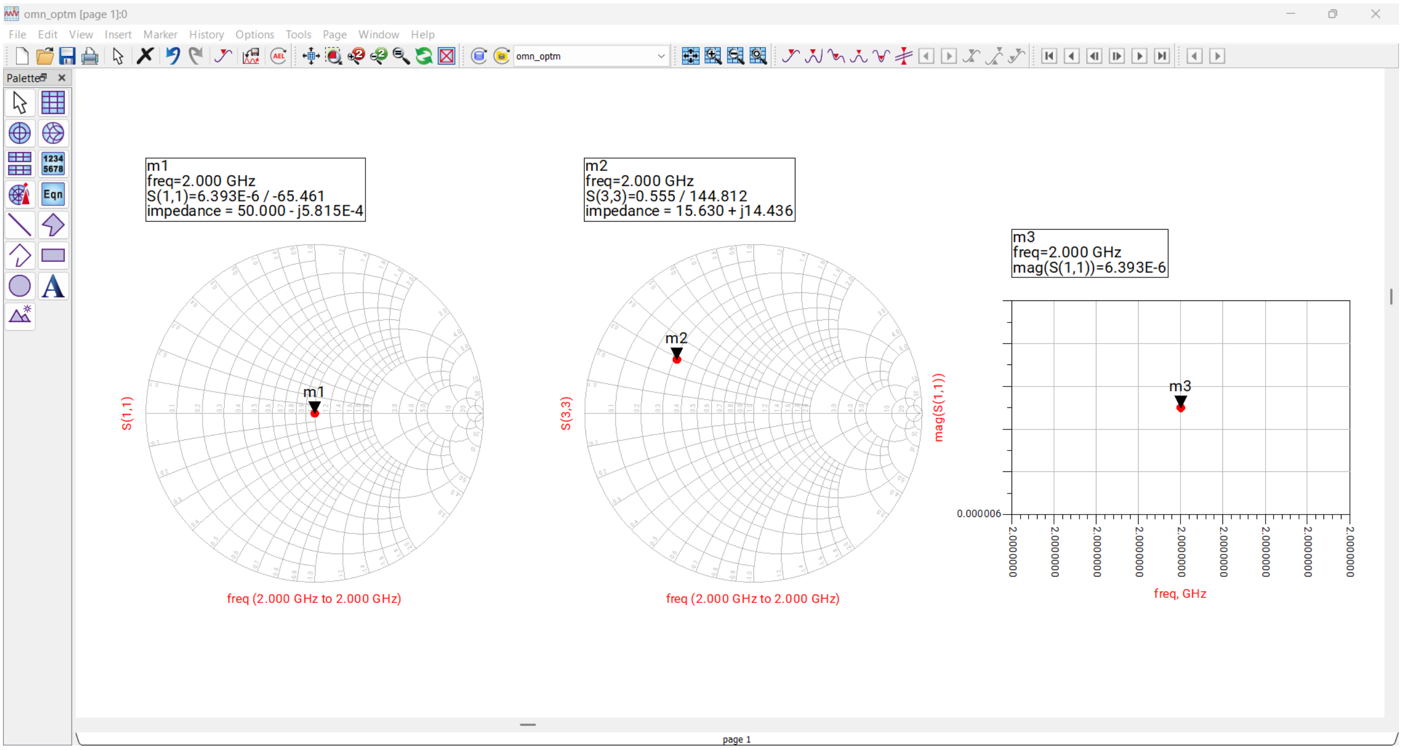
OUTPUT MATCHING NETWORK (WITHOUT OPTIMIZE) RESULTS At 2GHz:



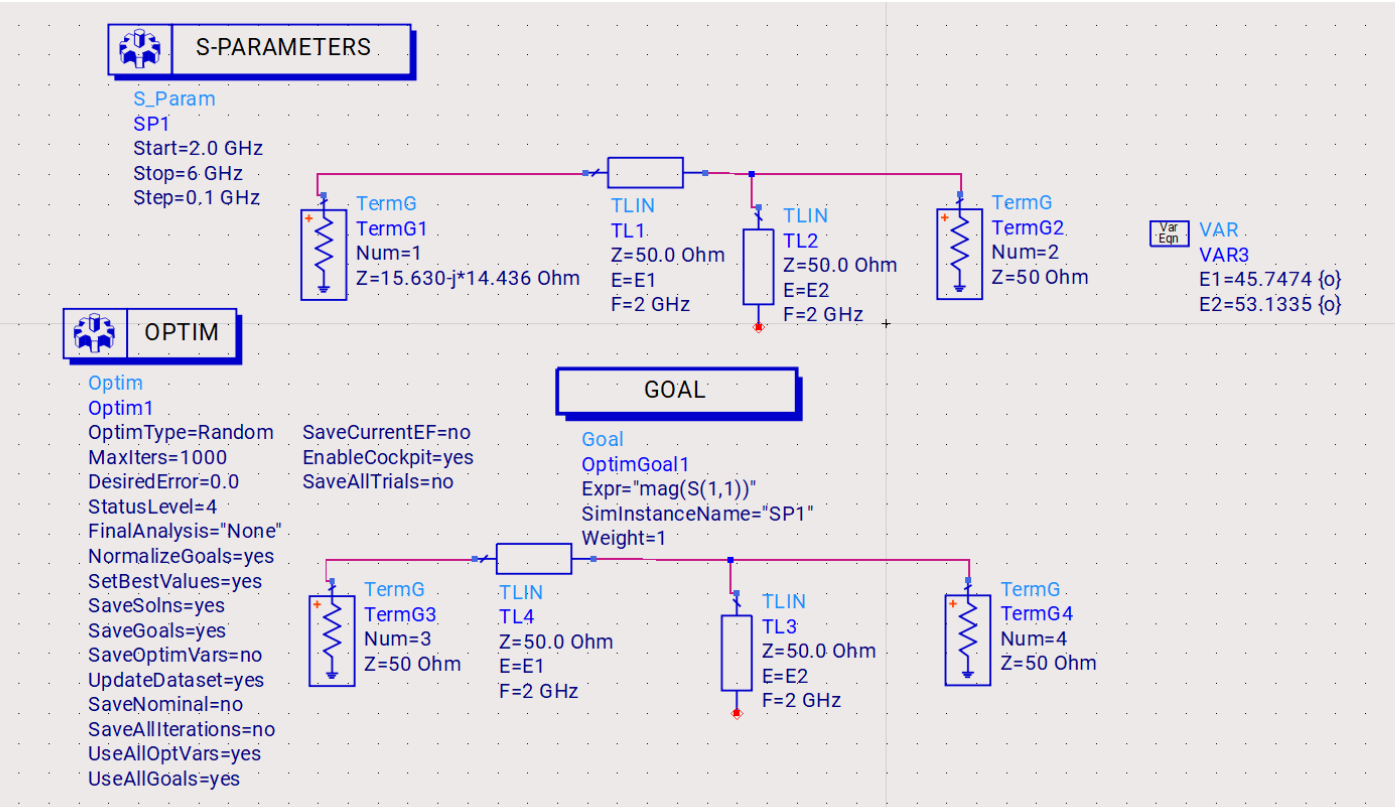
OUTPUT MATCHING NETWORK (WITH OPTIMIZE) SCHEMATICS AT SINGLE FREQUENCY 2GHz:



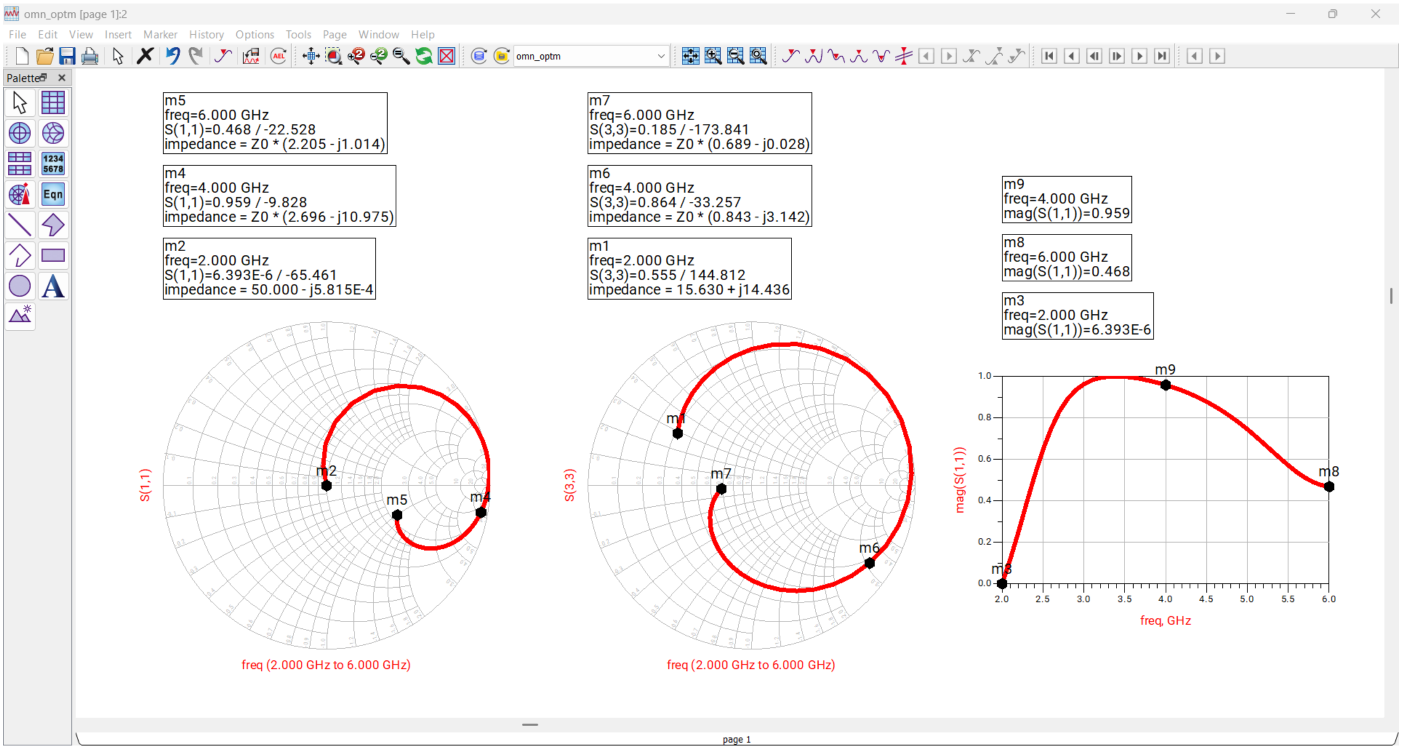
OUTPUT MATCHING NETWORK (WITH OPTIMIZE) RESULTS AT SINGLE FREQUENCY 2GHz:



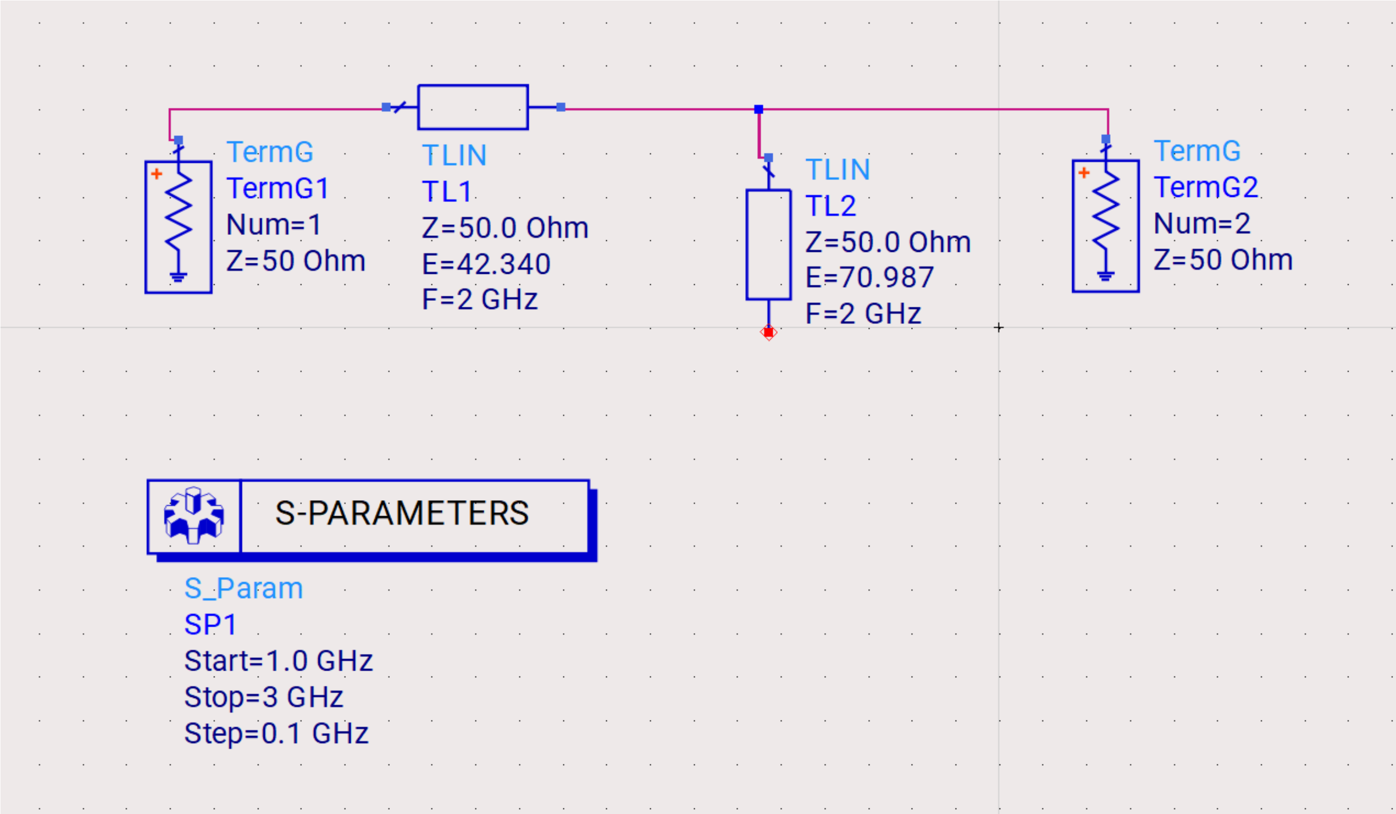
OUTPUT MATCHING NETWORK (WITH OPTIMIZE) SCHEMATICS AT FREQUENCY 2GHz,4GHz and 6GHz:



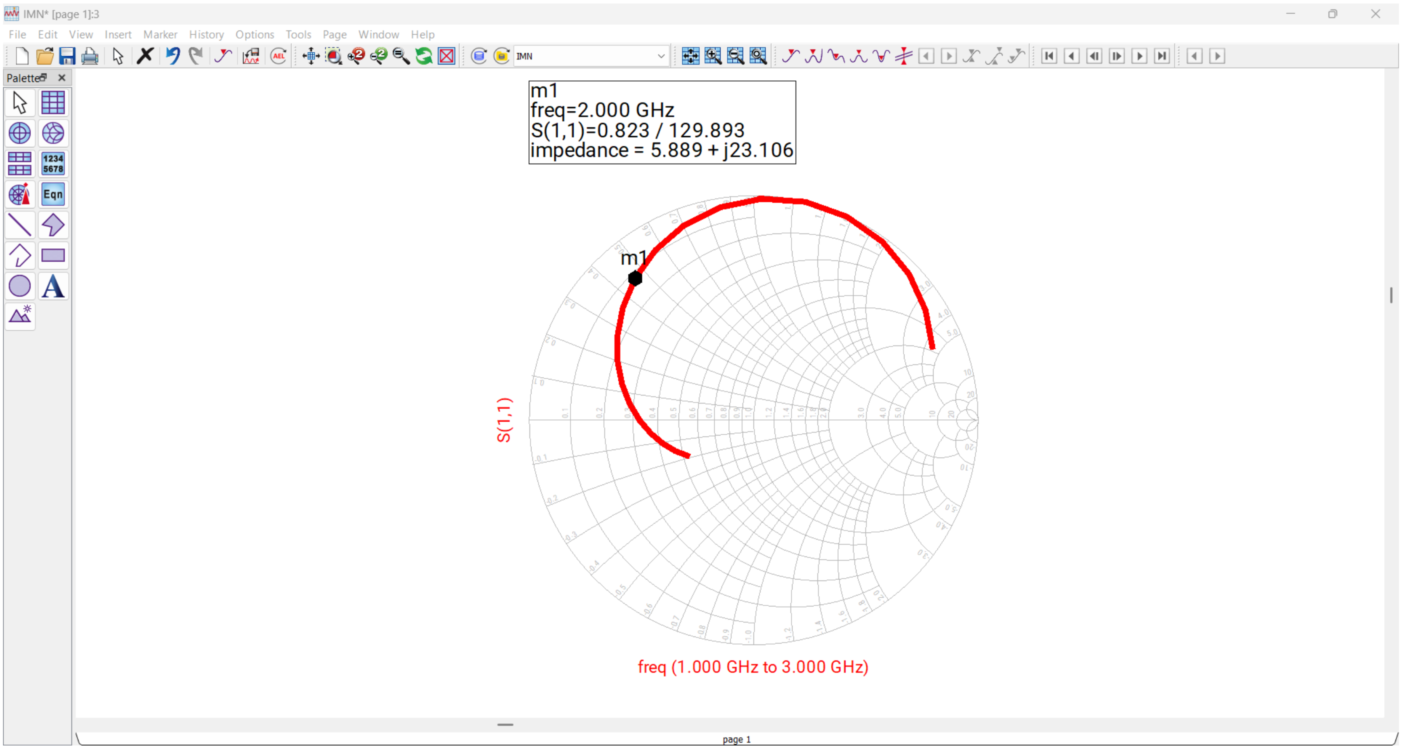
OUTPUT MATCHING NETWORK (WITH OPTIMIZE) RESULTS AT FREQUENCY 2GHz,4GHz and 6GHz:



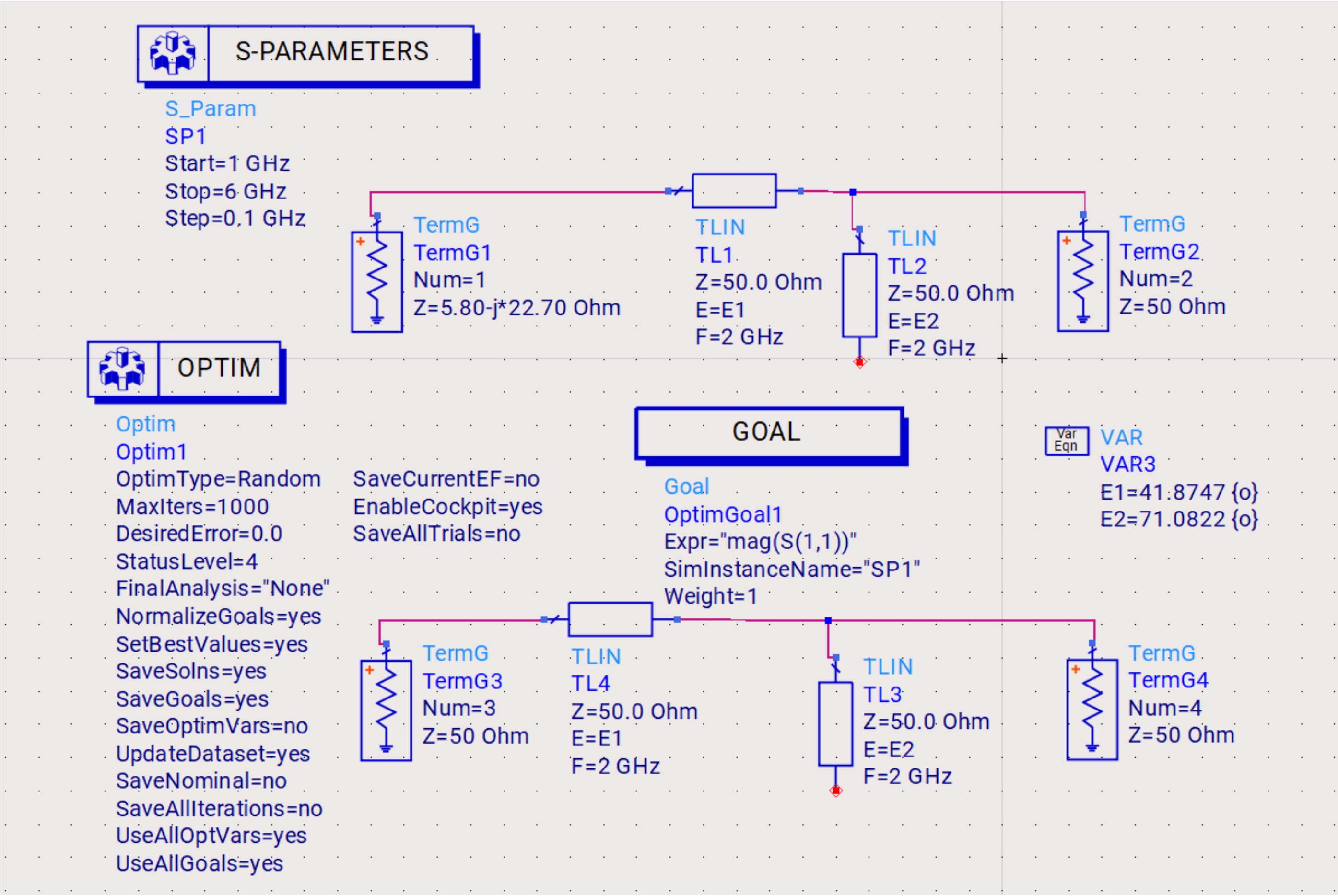
INPUT MATCHING NETWORK (WITHOUT OPTIMIZE) SCHEMATICS At 2GHz:



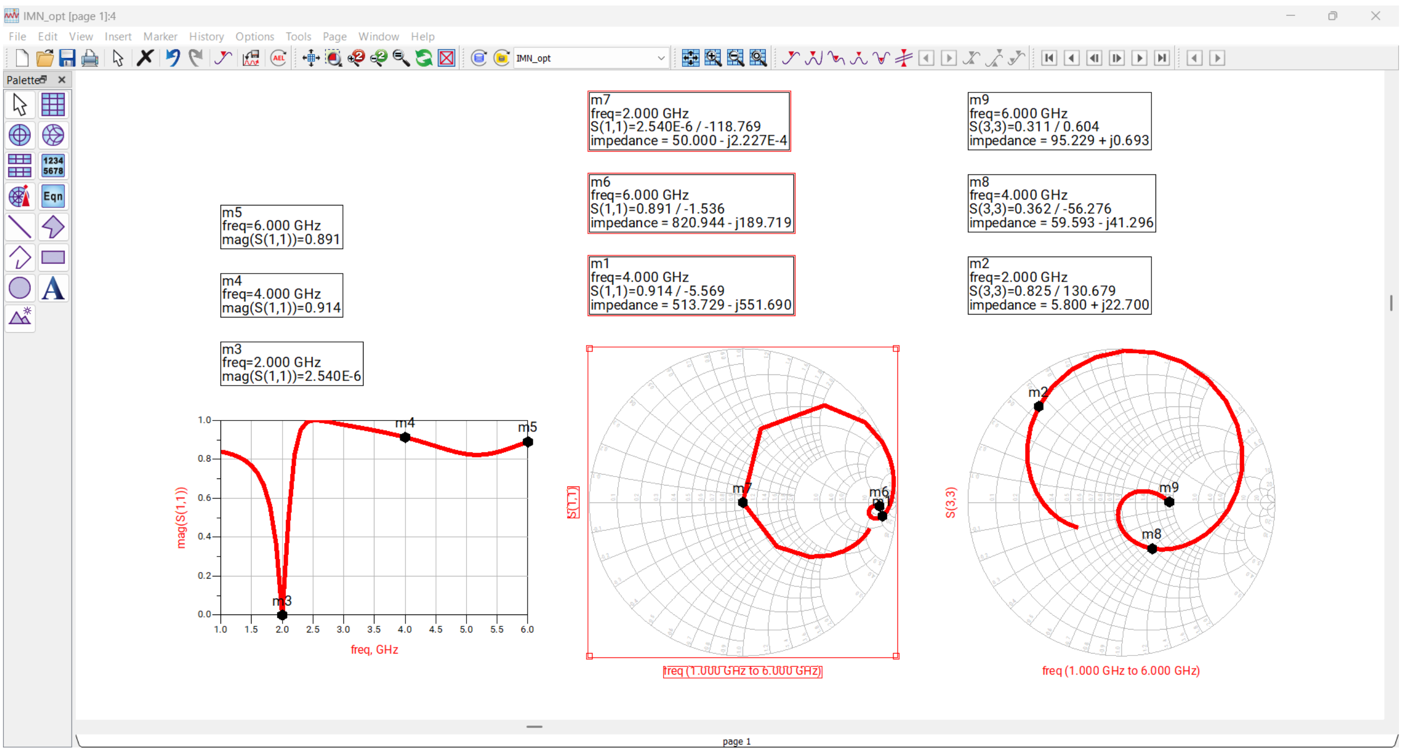
INPUT MATCHING NETWORK (WITHOUT OPTIMIZE) RESULTS At 2GHz:



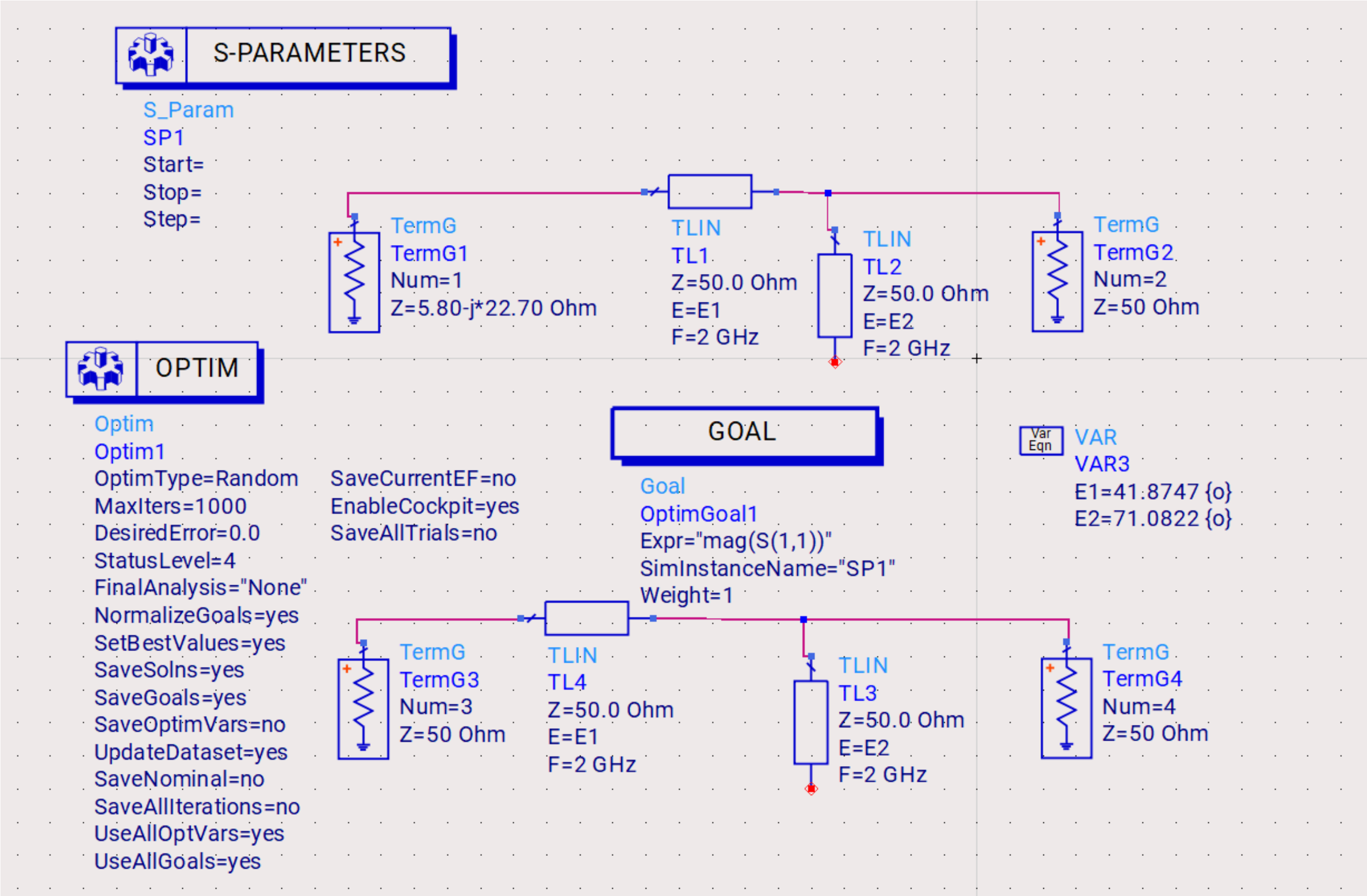
INPUT MATCHING NETWORK (WITH OPTIMIZE) SCHEMATICS AT FREQUENCY 2GHz,4GHz and 6GHz:



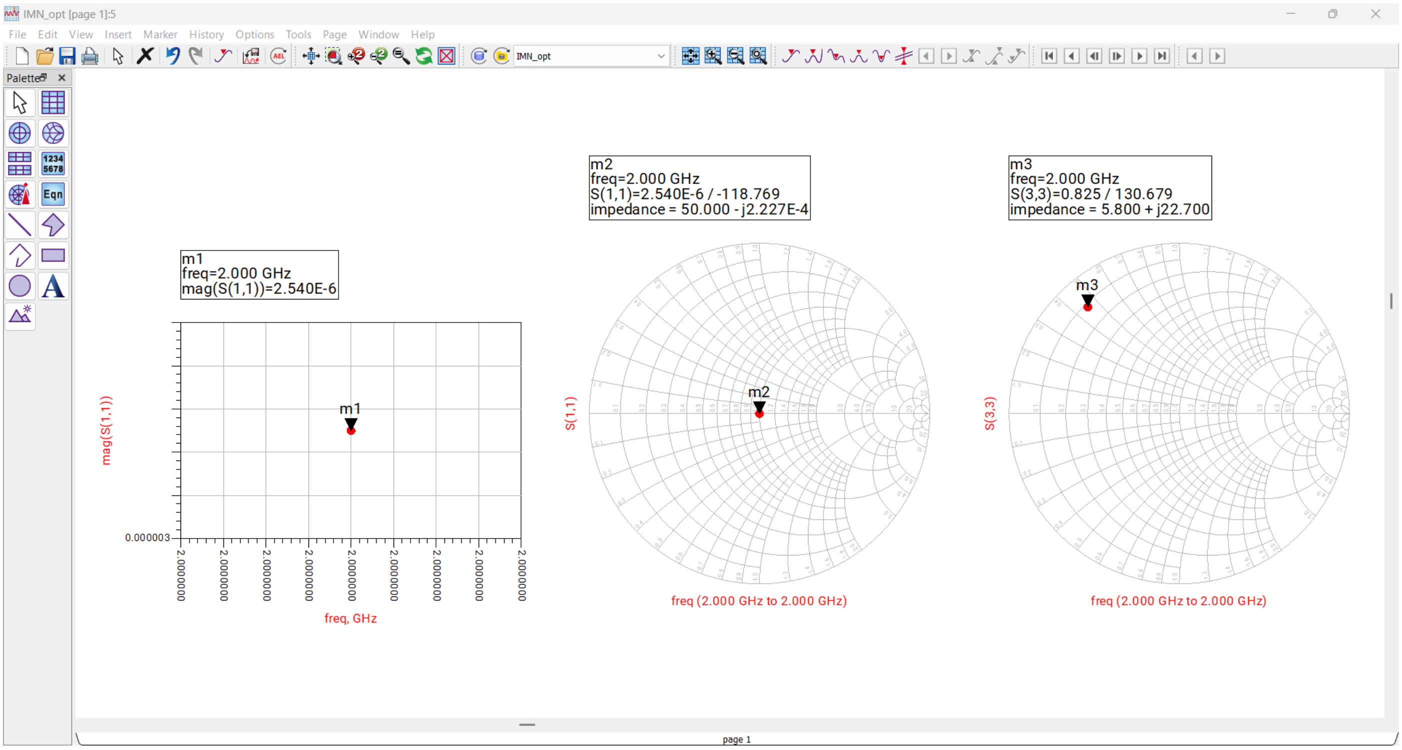
INPUT MATCHING NETWORK (WITH OPTIMIZE) RESULTS AT FREQUENCY 2GHz,4GHz and 6GHz:



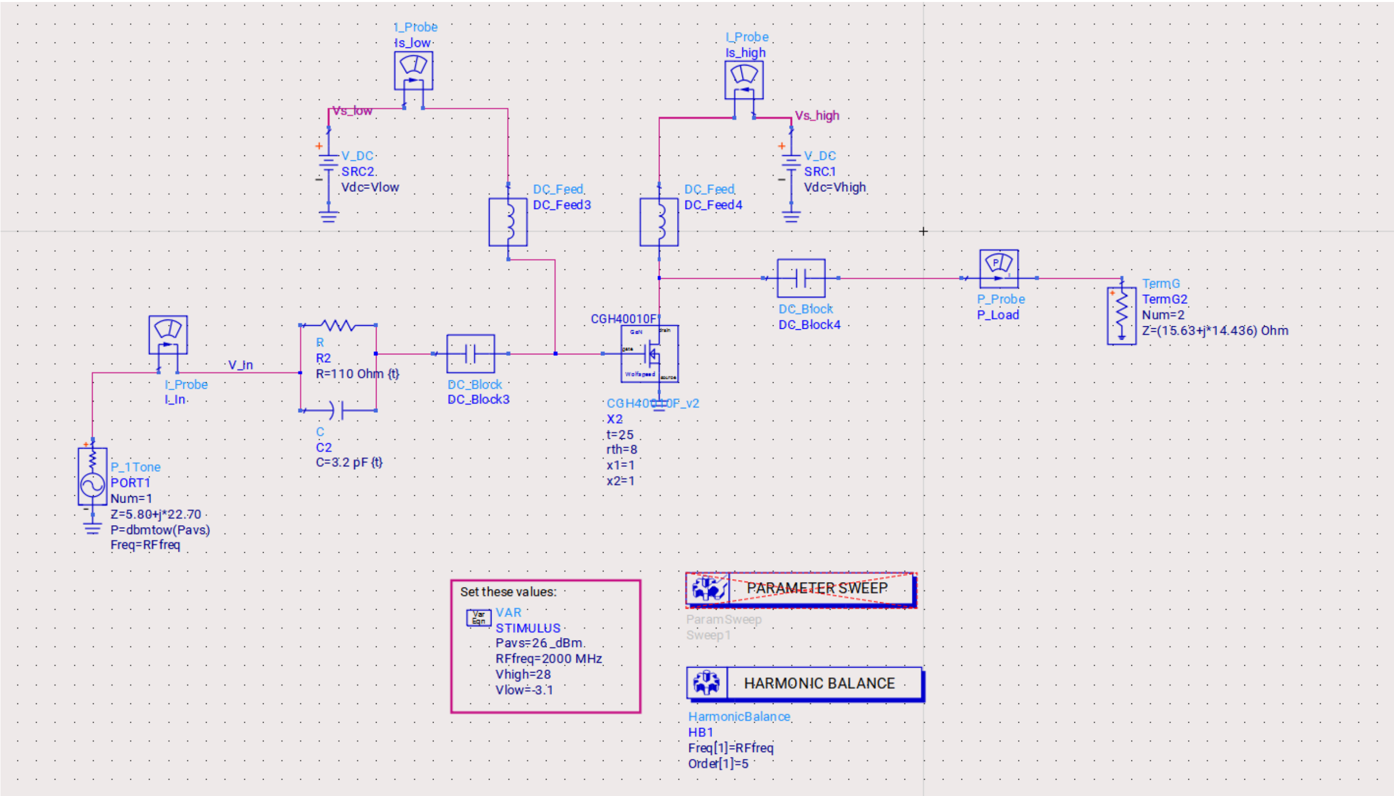
INPUT MATCHING NETWORK (WITH OPTIMIZE) SCHEMATICS AT SINGLE FREQUENCY 2GHz:



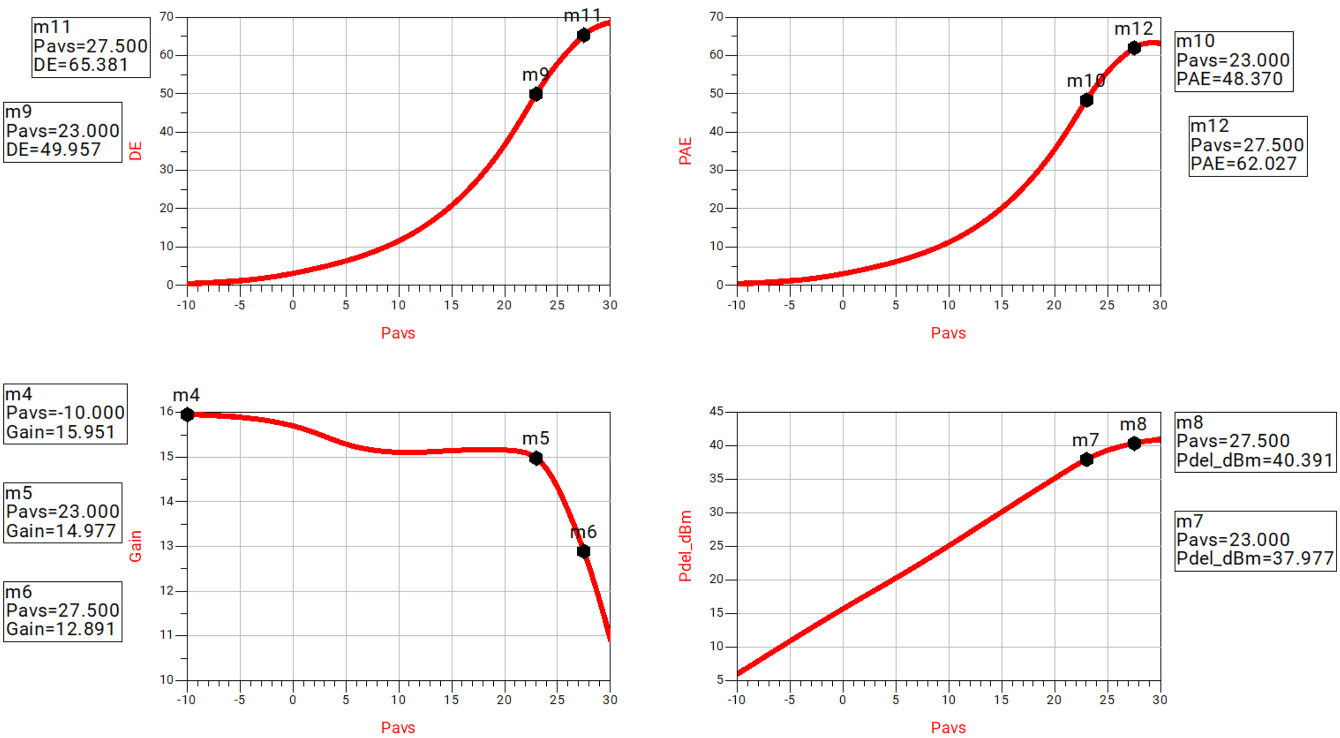
INPUT MATCHING NETWORK (WITH OPTIMIZE) RESULTS AT SINGLE FREQUENCY 2GHz:



Load Pull Testing with Harmonic Balance Schematic:



Load Pull Testing with Harmonic Balance Drain Efficiency, Power-Added Efficiency, Gain and Power Delivered (dBm) Results:



DC Power Calculations
The exists() function checks to be sure the corresponding piece of data is in the dataset. If it is not, then the function returns 0.

Eqn Vs_l=exists("real(Vs_low[0])")

Eqn Vs_h=exists("real(Vs_high[0])")

Eqn Is_h=exists("real(Is_high.i[0])")

Eqn Is_l=exists("real(Is_low.i[0])")

Eqn Pdc=Is_h*Vs_h +Is_l*Vs_l +1e-20

Power Delivered and Power-Added Efficiency Calculations

Eqn Pdel_Watts=P_Load.p[1]

Pavs is the available source power, set on the schematic, and passed into the dataset using the Harmonic Balance controller.

Eqn Pavs_Watts=10**((Pavs[0,0]-30)/10)

Eqn ~~PAE=100*(Pdel_Watts-Pavs_Watts)/Pdc~~

Eqn P_In_Watts=0.5*real(V_In[1]*conj(I_In.i[1]))

This power-added efficiency equation that uses the power absorbed at the input is considered to be a better, more accurate figure of merit than the one that uses the power available from the source.

Eqn PAE=100*(Pdel_Watts-P_In_Watts)/Pdc

Eqn Pdel_dBm = Pdel_Watts>0? 10*log10(Pdel_Watts)+30 : -200

Power-Added Efficiency contour calculations

Eqn PAEmax=max(max(PAE))

Eqn PAE_contours=contour(PAE,PAEmax-0.1-[0::(NumPAE_lines-1)]*PAE_step)

This -0.1 term is included so the contour at the maximum is a small circle, rather than a single point.

Conversion of the contours to polar coordinates for plotting on Smith Chart

Eqn PAE_contours_p=[indep(PAE_contours)+j*PAE_contours]

Conversion of the contours (which are actually curves of complex reflection coefficients that produce a constant PAE) to impedances

Eqn Zl_PAE=zin(PAE_contours_p,Z0[0,0])

Z0 is the reference impedance, set on the schematic, and passed into the dataset using the Harmonic Balance controller.

Re-scaling the contours for plotting on a Smith Chart with a different reference impedance

Eqn PAE_contours_scaled=(Zl_PAE-conj(Z0new))/(Zl_PAE+Z0new)

surface_samples are the actually-simulated reflection coefficients from which the contours are generated. real_indexs11 and imag_indexs11 are the swept variables in the simulation.

Eqn surface_samples=real_indexs11+j*expand(imag_indexs11)

Eqn Z_In_Fund=V_In[1]/I_In.i[1]

Eqn Rho_In_Fund=(Z_In_Fund-conj(Z0))/(Z_In_Fund+Z0)

Eqn DE=100*(Pdel_Watts)/Pdc

Eqn Gain=Pdel_dBm-Pavs

Power Delivered contour calculations

Eqn Pdelmax = max(max(Pdel_dBm))

Eqn Pdel_contours=contour(Pdel_dBm,Pdelmax-.01-[0::(NumPdel_lines-1)]*Pdel_step)

Conversion of the contours to polar coordinates for plotting on Smith Chart

Eqn Pdel_contours_p=[indep(Pdel_contours) +j*Pdel_contours]

Conversion of the contours (which are actually curves of complex reflection coefficients that produce a constant delivered power) to impedances

Eqn Zl_Pdel=zin(Pdel_contours_p,Z0[0,0])

Re-scaling the contours for plotting on a Smith Chart with a different reference impedance

Eqn Pdel_contours_scaled=(Zl_Pdel-conj(Z0new))/(Zl_Pdel+Z0new)

imag_index and real_index are used to calculate the PAE and power delivered that correspond to the reflection coefficient at the m3 marker location.

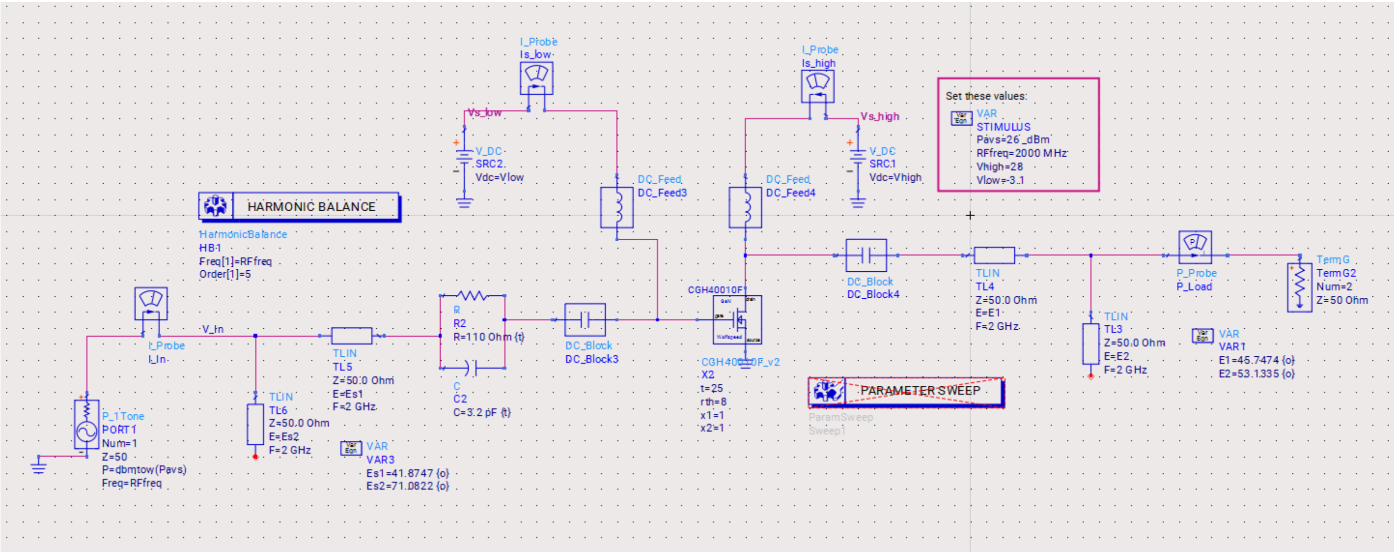
Eqn imag_index=find_index(imag_indexs11,imag(m3))

Eqn real_index=find_index(real_indexs11[imag_index,::],real(m3))

This calculates the impedance at marker m3.

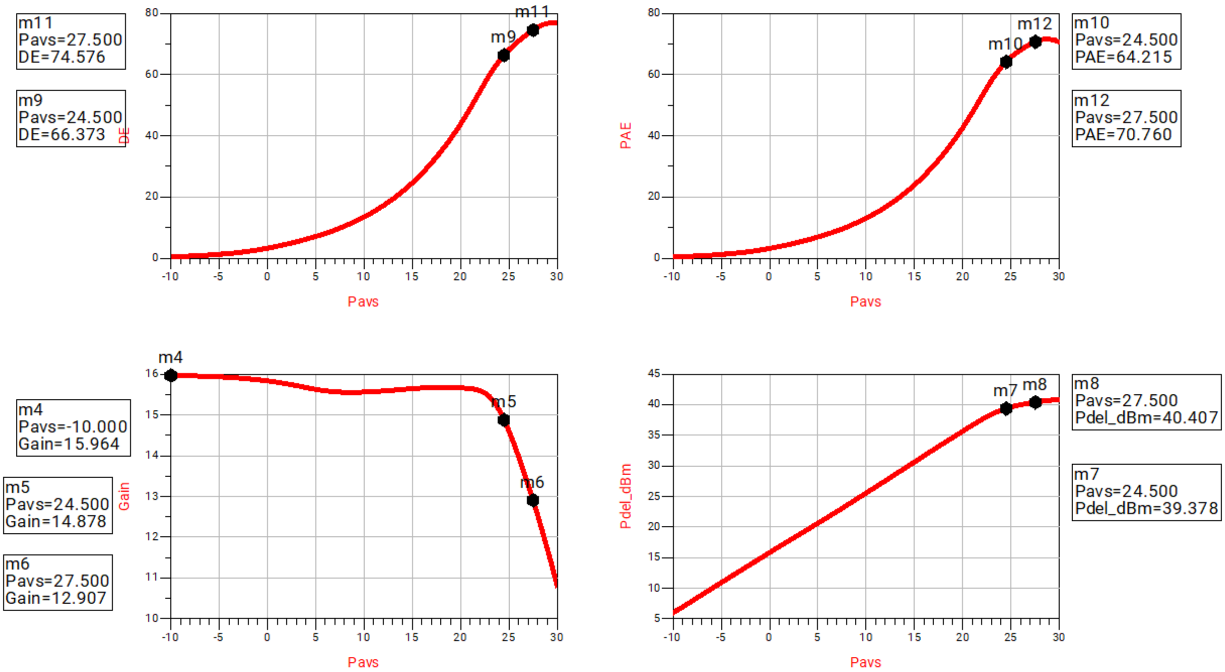
Eqn Z_at_m3=(conj(Z0[0,0])+Z0[0,0]*m3)/(1-m3)

Load Pull Matching Final Schematic Design:



Load Pull Final Matching Drain Efficiency, Power-Added

Efficiency, Gain and Power Delivered (dBm) Results:



DC Power Calculations
The exists() function checks to be sure the corresponding piece of data is in the dataset. If it is not, then the function returns 0.

Eqn Vs_l=exists("real(Vs_low[0])")

Eqn Vs_h=exists("real(Vs_high[0])")

Eqn Is_h=exists("real(Is_high.i[0])")

Eqn Is_l=exists("real(Is_low.i[0])")

Eqn Pdc=Is_h*Vs_h +Is_l*Vs_l +1e-20

Power Delivered and Power-Added Efficiency Calculations

Eqn Pdel_Watts=P_Load.p[1]

Pavs is the available source power, set on the schematic, and passed into the dataset using the Harmonic Balance controller.

Eqn Pavs_Watts=10**((Pavs[0,0]-30)/10)

Eqn ~~PAE=100*(Pdel_Watts-Pavs_Watts)/Pdc~~

Eqn P_In_Watts=0.5*real(V_In[1]*conj(I_In.i[1]))

This power-added efficiency equation that uses the power absorbed at the input is considered to be a better, more accurate figure of merit than the one that uses the power available from the source.

Eqn PAE=100*(Pdel_Watts-P_In_Watts)/Pdc

Eqn Pdel_dBm = Pdel_Watts>0? 10*log10(Pdel_Watts)+30 : -200

Power-Added Efficiency contour calculations

Eqn PAEmax=max(max(PAE))

Eqn PAE_contours=contour(PAE,PAEmax-0.1-[0::(NumPAE_lines-1)]*PAE_step)

This -0.1 term is included so the contour at the maximum is a small circle, rather than a single point.

Conversion of the contours to polar coordinates for plotting on Smith Chart

Eqn PAE_contours_p=[indep(PAE_contours)+j*PAE_contours]

Conversion of the contours (which are actually curves of complex reflection coefficients that produce a constant PAE) to impedances

Eqn Zl_PAE=zin(PAE_contours_p,Z0[0,0])

Z0 is the reference impedance, set on the schematic, and passed into the dataset using the Harmonic Balance controller.

Re-scaling the contours for plotting on a Smith Chart with a different reference impedance

Eqn PAE_contours_scaled=(Zl_PAE-conj(Z0new))/(Zl_PAE+Z0new)

surface_samples are the actually-simulated reflection coefficients from which the contours are generated. real_indexs11 and imag_indexs11 are the swept variables in the simulation.

Eqn surface_samples=real_indexs11+j*expand(imag_indexs11)

Eqn Z_In_Fund=V_In[1]/I_In.i[1]

Eqn Rho_In_Fund=(Z_In_Fund-conj(Z0))/(Z_In_Fund+Z0)

Power Delivered contour calculations

Eqn Pdelmax = max(max(Pdel_dBm))

Eqn Pdel_contours=contour(Pdel_dBm,Pdelmax-.01-[0::(NumPdel_lines-1)]*Pdel_step)

Conversion of the contours to polar coordinates for plotting on Smith Chart

Eqn Pdel_contours_p=[indep(Pdel_contours) +j*Pdel_contours]

Conversion of the contours (which are actually curves of complex reflection coefficients that produce a constant delivered power) to impedances

Eqn Zl_Pdel=zin(Pdel_contours_p,Z0[0,0])

Re-scaling the contours for plotting on a Smith Chart with a different reference impedance

Eqn Pdel_contours_scaled=(Zl_Pdel-conj(Z0new))/(Zl_Pdel+Z0new)

Eqn DE=100*(Pdel_Watts)/Pdc

Eqn Gain=Pdel_dBm-Pavs

imag_index and real_index are used to calculate the PAE and power delivered that correspond to the reflection coefficient at the m3 marker location.

Eqn imag_index=find_index(imag_indexs11,imag(m3))

Eqn real_index=find_index(real_indexs11[imag_index,::],real(m3))

This calculates the impedance at marker m3.

Eqn Z_at_m3=(conj(Z0[0,0])+Z0[0,0]*m3)/(1-m3)