PARAMETER EXTRACTION WITH Spicepipe

R.M. Biernacki, J.W. Bandler, S.H. Chen and P.A. Grobelny

SOS-93-9-R

May 1993

PARAMETER EXTRACTION WITH Spicepipe

R.M Biernacki,* J.W. Bandler,* S.H. Chen* and P.A. Grobelny

Abstract

In this report we present a new technique to extract DC and small-signal AC parameters of device models featured in SPICE utilizing the optimization capabilities of OSA90/hope™ and Spicepipe. Spicepipe is a user-friendly interface interconnecting SPICE-PAC, an optimization structured version of the popular circuit simulator SPICE, with the OSA90/hope design environment. Our method utilizes the OSA90/hope expression processing capabilities to convert the results of SPICE-PAC circuit analysis into small-signal multiport parameters to be matched with measurement data. The technique makes it possible to use the OSA90/hope state-of-the-art optimizers, such as ℓ_1, ℓ_2, minimax or the recently added, very powerful Huber optimizer, to extract parameters of popular and widely accepted SPICE device models. We also optimize multiport parameters of circuits working under small-signal AC conditions and simulated by SPICE-PAC. This application is particularly powerful if applied to large circuits partitioned into subcircuits. Furthermore, each of the subcircuits can contain device models available either in SPICE-PAC or in OSA90/hope. We demonstrate SPICE device model parameter extraction by extracting parameters of the SPICE-PAC model of a bipolar transistor working under DC and small-signal AC conditions. An LC transformer circuit example is used to illustrate optimization of small-signal multiport parameters of a circuit simulated by SPICE-PAC.

---

* R.M. Biernacki, J.W. Bandler and S.H. Chen are also with Optimization Systems Associates Inc., P.O. Box 8083, Dundas, Ontario, Canada L9H 5E7.

This work was supported in part by Optimization Systems Associates Inc. and in part by the Natural Sciences and Engineering Research Council of Canada under Grants OGP0042444, OGP0007239 and STR0117819.

I. INTRODUCTION

In this report we exemplify our open software architecture approach [1, 2] to building general purpose CAD systems with Spicepipe [2, 3]. Spicepipe is a user friendly interface interconnecting the SPICE-like simulator SPICE-PAC [4] to the friendly optimization environment of OSA90/hope [5]. We concentrate on utilizing Spicepipe to perform parameter extraction of device models featured in SPICE-PAC. We also extract and subsequently optimize small-signal multiport parameters of entire circuits, possibly divided into subcircuits, simulated by SPICE-PAC. Through Spicepipe, a variety of OSA90/hope state-of-the-art optimizers including ℓ_1, ℓ_2, minimax or the recently added Huber optimizer, can be used to extract parameters of SPICE-PAC device models. For other applications of Spicepipe, including design optimization, Monte Carlo analysis and yield-driven design in both the frequency and the time domains see [3].

SPICE-PAC, similarly to SPICE, provides as output of circuit analysis either time waveforms for the time-domain analysis or complex values of voltages and/or currents in the circuit for the small-signal analysis. Multiport small-signal responses are not available directly as results of SPICE-PAC analysis, which makes parameter extraction of device models difficult, especially if the measurement data is available in the form of S, H or any other multiport parameters. We employ the OSA90/hope expression processing capability to convert SPICE-PAC analysis results into the small-signal parameters of interest. Also, if the circuit to be analyzed by SPICE-PAC is partitioned into subcircuits we use OSA90/hope to combine the multiport parameters of the subcircuits and obtain the parameters of the overall circuit.

In this report we assume that the reader is familiar with the basics of OSA90/hope, SPICE-PAC and Spicepipe as well as with the principles of the Datapipe technology [5, 6] utilized in Spicepipe. In Section II we briefly introduce Spicepipe. Section III is devoted to a discussion on converting SPICE-PAC small-signal responses into multiport parameters. In Section IV we present our approach to parameter extraction of SPICE-PAC device models. We illustrate this approach extracting parameters of the SPICE-PAC Bipolar Junction Transistor (BJT) model. In

Section V we present our approach to optimizing small-signal multiport parameters of circuits simulated by SPICE-PAC. Optimization of the modulus of the input reflection coefficient of an LC transformer circuit demonstrates small-signal AC parameter optimization capabilities of Spicepipe. Section VI contains our conclusions.

II. Spicepipe [2, 3]

Spicepipe is a user friendly interface interconnecting SPICE-PAC to OSA90/hope. SPICE-PAC is an extended, optimization-structured version of the circuit simulator SPICE [7] featuring, among others, excellent time-domain simulation and SPICE device models. OSA90/hope is a circuit design environment providing powerful design optimization and statistical capabilities, including Monte Carlo analysis and yield-driven design. It also has built-in frequency-domain circuit simulation including harmonic balance. All these features are embedded in a language supporting scalar, vector and matrix expression processing.

By connecting SPICE-PAC to OSA90/hope we created a mixed frequency-time-domain design environment supporting the circuit designer with versatile simulation and optimization, statistical analysis and yield-driven design in both domains. Also, SPICE-PAC and OSA90/hope's device models became simultaneously available through Spicepipe. This makes it possible for the designer to choose the most suitable device models from either the well established, widely used but rigid in terms of extendability, SPICE device models, or from the robust and much more flexible OSA90/hope device models.

Spicepipe is based on our Datapipe [6] technology which utilizes UNIX pipes as communication channels used to exchange information between otherwise independent programs. We call these communication channels Datapipes. We use SPICE-PAC's variables, described in detail in [3, 8], to dynamically update sweep, optimization and/or statistical parameters during the analysis to bypass expensive SPICE-PAC circuit file parsing in the second and subsequent sweep or optimization iterations. Spicepipe invokes SPICE-PAC in a manner completely transparent to the user.

III. CONVERTING SPICE RESULTS INTO MULTIPORT PARAMETERS

In order to obtain multiport small-signal parameters of a circuit simulated by SPICE-PAC we have to postprocess SPICE-PAC small-signal analysis results.

Let us, for example, assume that the admittance parameters Y of a two-port circuit are sought. From SPICE-PAC simulation we obtain port voltages and port currents as

$$\boldsymbol{V} = [V_1\ V_2]^T \tag{1}$$

and

$$\boldsymbol{I} = [I_1\ I_2]^T \tag{2}$$

as indicated in Fig. 1. Y parameters are defined as

$$I_1 = Y_{11}V_1 + Y_{12}V_2 \tag{3a}$$

$$I_2 = Y_{21}V_1 + Y_{22}V_2 \tag{3b}$$

or in matrix notation

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \tag{4}$$

where all the voltages, currents and Y parameters are, in general, complex. Equation (4) can be rewritten as

$$\begin{bmatrix} I_1 \\ I_2 \end{bmatrix} = \begin{bmatrix} V_1 & V_2 & 0 & 0 \\ 0 & 0 & V_1 & V_2 \end{bmatrix} \begin{bmatrix} Y_{11} \\ Y_{12} \\ Y_{21} \\ Y_{22} \end{bmatrix} \tag{5}$$

Clearly, the system in (5) is underdetermined. To make the voltage matrix in (5) of full rank we need two more independent equations relating port voltages and currents through Y parameters.

These can be obtained by simulating the device under a different load condition. Changing the load impedance in Fig. 1 from Y_L to Y_L' and resimulating we get

$$I'_1 = Y_{11}V'_1 + Y_{12}V'_2 \tag{6a}$$

$$I'_2 = Y_{21}V'_1 + Y_{22}V'_2 \tag{6b}$$

Combining equations in (3) and (6) we extend (5) to

$$\begin{bmatrix} I_1 \\ I_2 \\ I_1' \\ I_2' \end{bmatrix} = \begin{bmatrix} V_1 & V_2 & 0 & 0 \\ 0 & 0 & V_1 & V_2 \\ V_1' & V_2' & 0 & 0 \\ 0 & 0 & V_1' & V_2' \end{bmatrix} \begin{bmatrix} Y_{11} \\ Y_{12} \\ Y_{21} \\ Y_{22} \end{bmatrix} \tag{7}$$

If Y_L is different from Y_L' the system in (7) is of full rank for most practical circuits. The solution of (7) results in the sought Y parameters. In addition, if we use the driving source and load constraints to express currents $\boldsymbol{I}$ and $\boldsymbol{I'}$, where $\boldsymbol{I'} = [I_1' \; I_2']^T$, in terms of the source and load parameters and voltages $\boldsymbol{V}$ and $\boldsymbol{V'}$, where $\boldsymbol{V'} = [V_1' \; V_2']^T$, we can eliminate $\boldsymbol{I}$ and $\boldsymbol{I'}$ from (7) and rewrite (7) as

$$\begin{bmatrix} I_S - V_1 Y_S \\ -V_2 Y_L \\ I_S' - V_1' Y_S' \\ -V_2' Y_L' \end{bmatrix} = \begin{bmatrix} V_1 & V_2 & 0 & 0 \\ 0 & 0 & V_1 & V_2 \\ V_1' & V_2' & 0 & 0 \\ 0 & 0 & V_1' & V_2' \end{bmatrix} \begin{bmatrix} Y_{11} \\ Y_{12} \\ Y_{21} \\ Y_{22} \end{bmatrix} \tag{8}$$

or

$$\boldsymbol{I}_c = \boldsymbol{V}_c \boldsymbol{Y}_c \tag{9}$$

where $\boldsymbol{I}_c$ corresponds to the left-hand side current vector, $\boldsymbol{V}_c$ corresponds to the right-hand side matrix with voltage entries and $\boldsymbol{Y}_c$ is a vector of Y parameters. To solve the complex system (8) or (9) we can solve an equivalent real system

$$\begin{bmatrix} Re\{\boldsymbol{I}_c\} \\ Im\{\boldsymbol{I}_c\} \end{bmatrix} = \begin{bmatrix} Re\{\boldsymbol{V}_c\} & -Im\{\boldsymbol{V}_c\} \\ Im\{\boldsymbol{V}_c\} & Re\{\boldsymbol{V}_c\} \end{bmatrix} \begin{bmatrix} Re\{\boldsymbol{Y}_c\} \\ Im\{\boldsymbol{Y}_c\} \end{bmatrix} \tag{10}$$

In order to solve (10) we need SPICE-PAC to return as its output only V and V'. It is worth noticing that the source parameters I_S, Y_S and I_S', Y_S' can take arbitrary values. In particular, they can be real and unchanged, that is I_S and Y_S can be equal to I_S' and Y_S', respectively. The above approach to determining Y parameters is similar to determining Y parameters from measurements where for a two-port network one would have to take two sets of independent measurements as well.

The derivation of the formula expressing the Y parameters as functions of port voltages and terminations for networks with three and more ports is analogous. In general, n sets of independent results (simulated or measured) are needed to determine the Y parameters, where n is the number of ports in the network. If the network is known to be lossless, reciprocal or both lossless and reciprocal the number of the data sets required is correspondingly smaller.

Once we know the Y parameters of the circuit any other parameters can be easily obtained. In fact, instead of deriving Y we could derive Z, H or any other parameters with the other parameters following immediately. Which parameters we should use would be more important if the parameters were to be obtained by measurements. Since we deal with computer simulation any of the parameters can be obtained in an equally easy way.

IV. PARAMETER EXTRACTION OF SPICE-PAC DEVICE MODELS

Using Spicepipe the user defines the SPICE-PAC circuit file and designates the model parameters to be extracted within the OSA90/hope design environment. The results of SPICE-PAC analysis become available as outputs of the Spicepipe Datapipe. Error functions for optimization follow immediately as the difference between the device responses and the measurements. Spicepipe syntax and synchronisation requirements assuring proper flow of information between OSA90/hope and SPICE-PAC are detailed in [3].

The procedures of extracting the DC and small-signal AC parameters are essentially the same except that the small-signal AC parameter extraction involves, in general, postprocessing of SPICE-PAC responses as described in the previous section.

SPICE-PAC BJT Model Parameter Extraction

To illustrate SPICE-PAC device model parameter extraction we extract parameters of the SPICE-PAC BJT model. We separately extract the parameters of the model for the DC and small-signal AC operating conditions. In both cases we try to match the model response with synthetic data obtained from SPICE-PAC BJT model simulation. We used the BJT model suggested in [9] of the 2N2222A transistor to generate the data. The 2N2222A transistor SPICE-PAC model parameters as well as the DC and small-signal AC data are listed in Appendix A. SPICE-PAC BJT equivalent static model, small-signal circuit equivalent model as well as the formulas describing these models are given in [10].

A. DC BJT Model Parameter Extraction

Here we match the I-V characteristics of the model with the data. For optimization we use the OSA90/hope ℓ_2 optimizer. Table I lists the values of the model parameters both at the starting point and at the solution. SPICE-PAC BJT model parameters not listed in Table I assume their default values. 6 parameters were optimized. Figs. 2 and 3 show the match between the model response and the data before and after optimization, respectively. A good match between the extracted model response and the data can be observed.

B. Small-Signal AC BJT Model Parameter Extraction

We want to match the small-signal AC parameters of the transistor to the S parameter data. We use the technique described in Section III to calculate the Y and subsequently S parameters of the transistor in each optimization iteration. We use the OSA90/hope ℓ_1 optimizer. The SPICE-PAC model parameters used are listed in Table II together with the optimization results. 15 BJT model parameters were optimized. Parameters not listed assume their default values. Figs.

4 and 5 show the S parameters match before and after optimization, respectively. A very good match of model response to the data can be observed for the extracted BJT model.

OSA90/hope circuit files used in the above experiments can be found in Appendix B.

V. SMALL-SIGNAL AC OPTIMIZATION WITH Spicepipe

Having created a mechanism converting SPICE-PAC small-signal AC analysis results into multiport parameters we can use it in optimization of circuits.

Consider a situation where we want to optimize some or all of the multiport parameters of a large circuit. Let us further partition the circuit under consideration into subcircuits. We can treat all of the subcircuits as independent multiport networks interconnected together, as shown in Fig. 6. We simulate and extract the multiport parameters of each of the subcircuits independently. Subsequently we let OSA90/hope derive the multiport parameters of the overall circuit by combining the parameters of the subcircuits.

Optimization of an LC Transformer

An LC transformer circuit is chosen to demonstrate optimization of small-signal multiport parameters of a partitioned circuit simulated by SPICE-PAC. We want to optimize the modulus of the input reflection coefficient MS11 for the transformer of Fig. 7. We use 21 equally spaced points in the frequency range from 0.0796 GHz to 0.188 GHz. All L and C elements in the circuit are optimizable. Input resistance R_{in} = 3 Ω and output resistance R_{out} = 1 Ω.

We first optimize the entire transformer as a one port network. Next we partition the transformer into two subcircuits, as illustrated in Fig. 8, each of which is simulated by SPICE-PAC and described by its small-signal two-port parameters. We let OSA90/hope combine these subcircuits together with proper source (R_{in} = 3 Ω) and termination (R_{out} = 1 Ω), see Fig. 8, and calculate the one port input reflection coefficient MS11. We optimize the partitioned transformer again. The optimization results with and without partitioning agree to 5 decimal digits. Values of the L and C elements before and after optimization are listed in Table III. The maximum value of MS11 is decreased by the OSA90/hope minimax optimizer from 0.66 to 0.076. The diagrams of MS11 as function of frequency before and after optimization are shown in Fig. 9.

The circuit files used in this experiment are listed in Appendix C.

VI. CONCLUSIONS

We have presented a new technique utilizing Spicepipe to extract DC and small-signal AC parameters of SPICE-PAC device models. The technique makes it possible to apply various optimizers of OSA90/hope to extract parameters of SPICE-PAC device models. We have demonstrated SPICE-PAC device model parameter extraction extracting DC and small-signal AC parameters of SPICE-PAC npn BJT model. We have also presented an approach to optimize multiport parameters of circuits simulated by SPICE-PAC and working under small-signal AC conditions. The optimization of small-signal multiport parameters has been illustrated with optimization of an LC transformer circuit.

REFERENCES

[1] J.W. Bandler, R.M. Biernacki and S.H. Chen, "Design optimization with external simulators," *Proc. 8th Conf. on the Computation of Electromagnetic Fields COMPUMAG* (Sorrento, Italy), pp. 1061-1063, 1991.

[2] R.M. Biernacki, J.W. Bandler, S.H. Chen and P.A. Grobelny, " Spicepipe - a case study of flexible circuit optimization," *Proc. European Conf. on Circuit Theory and Design,* Davos, Switzerland, 1993.

[3] R.M. Biernacki, J.W. Bandler, S.H. Chen and P.A. Grobelny, "Integrating the SPICE-PAC simulator with the OSA90/hope™ design environment," Department of Electrical and Computer Engineering, McMaster University, Hamilton, Canada, Report SOS-92-5-R, 1992.

[4] W.M. Zuberek, "SPICE-PAC version 2G6c user's guide", Department of Computer Science, Memorial University of Newfoundland, St. John's, Newfoundland, Canada A1C 5S7, Technical Report 8902, 1989.

[5] *OSA90/hope™ Version 2.0 User's Manual*, Optimization Systems Associates, Inc., P.O. Box 8083, Dundas, Ontario, Canada L9H 5E7, 1992.

[6] J.W. Bandler, Q.J. Zhang, G. Simpson and S.H. Chen, "IPPC: a library for inter-program pipe communication," Department of Electrical and Computer Engineering, McMaster University, Hamilton, Canada, Report SOS-90-10-U, 1990.

[7] A. Vladimirescu, K. Zhang, A.R. Newton, D.O. Pederson and A.L. Sangiovanni-Vincentelli, "SPICE Version 2G - User's guide," Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, 1981.

[8] W.M. Zuberek, "SPICE-PAC version 2G6c an overview," Department of Computer Science, Memorial University of Newfoundland, St. John's, Newfoundland, Canada A1C 5S7, Technical Report 8903, 1989.

[9] T.W. Thorpe, *Computerized Circuit Analysis with Spice*. John Wiley & Sons, 1992.

[10] P. Antognetti and G. Massobrio, *Semiconductor Device Modeling with Spice*. New York: McGraw-Hill, 1988.

TABLE I
PARAMETERS OF THE SPICE-PAC BJT MODEL
FOR DC I-V CHARACTERISTIC FITTING

Parameters	Before Optimization	After Optimization
IS(A)	10×10^{-15}	9.33×10^{-15}
VAF(V)	100	86.54
BF	300	220.25
NE	1.5	1.33
ISE(A)	10×10^{-15}	1.1×10^{-15}
BR	10	5.95
IKF(A)	0.2*	0.2*
XTB	1*	1*
NC	2*	2*
RC(Ω)	1*	1*
RB(Ω)	10*	10*
RE(Ω)	1*	1*

* Parameters not optimized.

TABLE II
PARAMETERS OF THE SPICE-PAC BJT MODEL
FOR SMALL-SIGNAL AC DATA FITTING

Parameters	Before Optimization	After Optimization
IS(A)	10×10^{-15}	25.26×10^{-15}
VAF(V)	100	262.21
BF	300	354.9
NE	1.5	1.22
ISE(A)	10×10^{-15}	7.83×10^{-15}
BR	0.01	0.04
IKF(A)	0.2*	0.2*
XTB	1*	1*
NC	2*	2*
RC(Ω)	10	1.05
CJC(F)	5×10^{-12}	4.75×10^{-12}
MJC	0.5	0.16
CJE(F)	51×10^{-12}	42.8×10^{-12}
MJE	0.5	0.076
TR(S)	10×10^{-9}	35.97×10^{-9}
TF(S)	100×10^{-12}	101.93×10^{-12}
RB(Ω)	100	9.92
RE(Ω)	1	0.021

* Parameters not optimized.

TABLE III
LC TRANSFORMER CIRCUIT:
L AND *C* ELEMENT VALUES BEFORE AND AFTER OPTIMIZATION

Element	L_1 (nH)	C_2 (nF)	L_3 (nH)	C_4 (nF)	L_5 (nH)	C_6 (nF)
Value Before Optimization	1	1	1	1	1	1
Value After Optimization	1.041	0.979	2.340	0.780	2.937	0.347

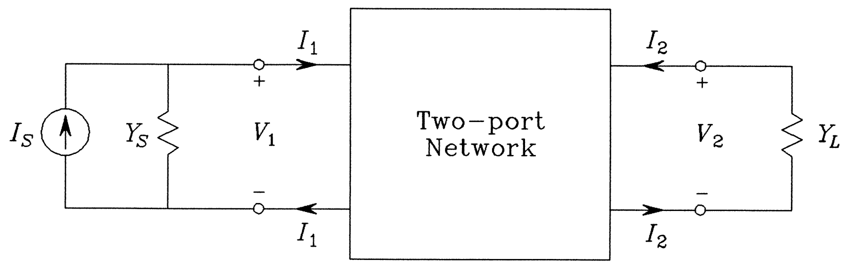


Fig. 1. Two-port network voltage polarity and current direction convention.

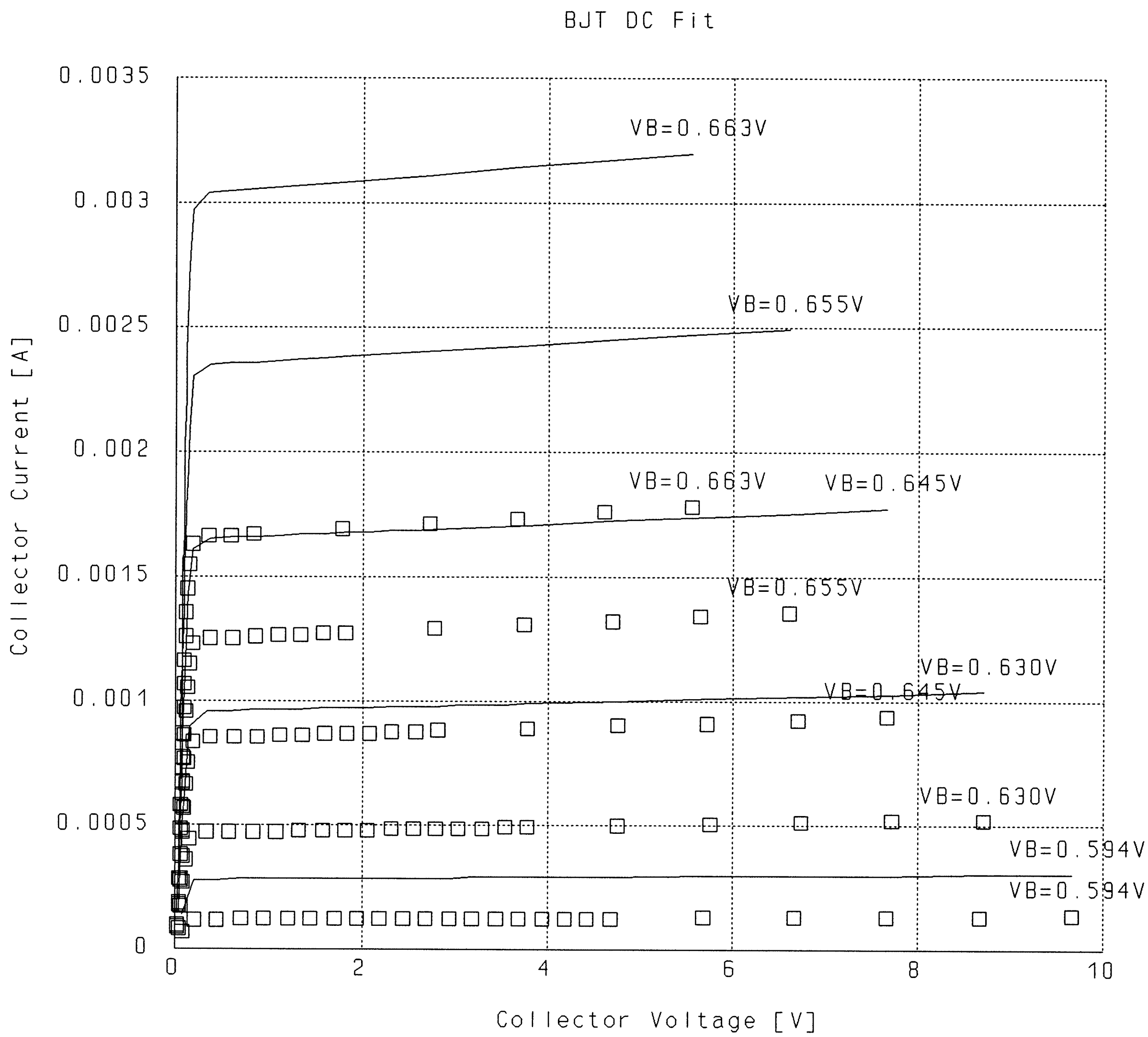


Fig. 2. DC I-V characteristic fit for the SPICE-PAC BJT model before optimization. Squares (□) correspond to the data, solid lines (——) correspond to the model responses.

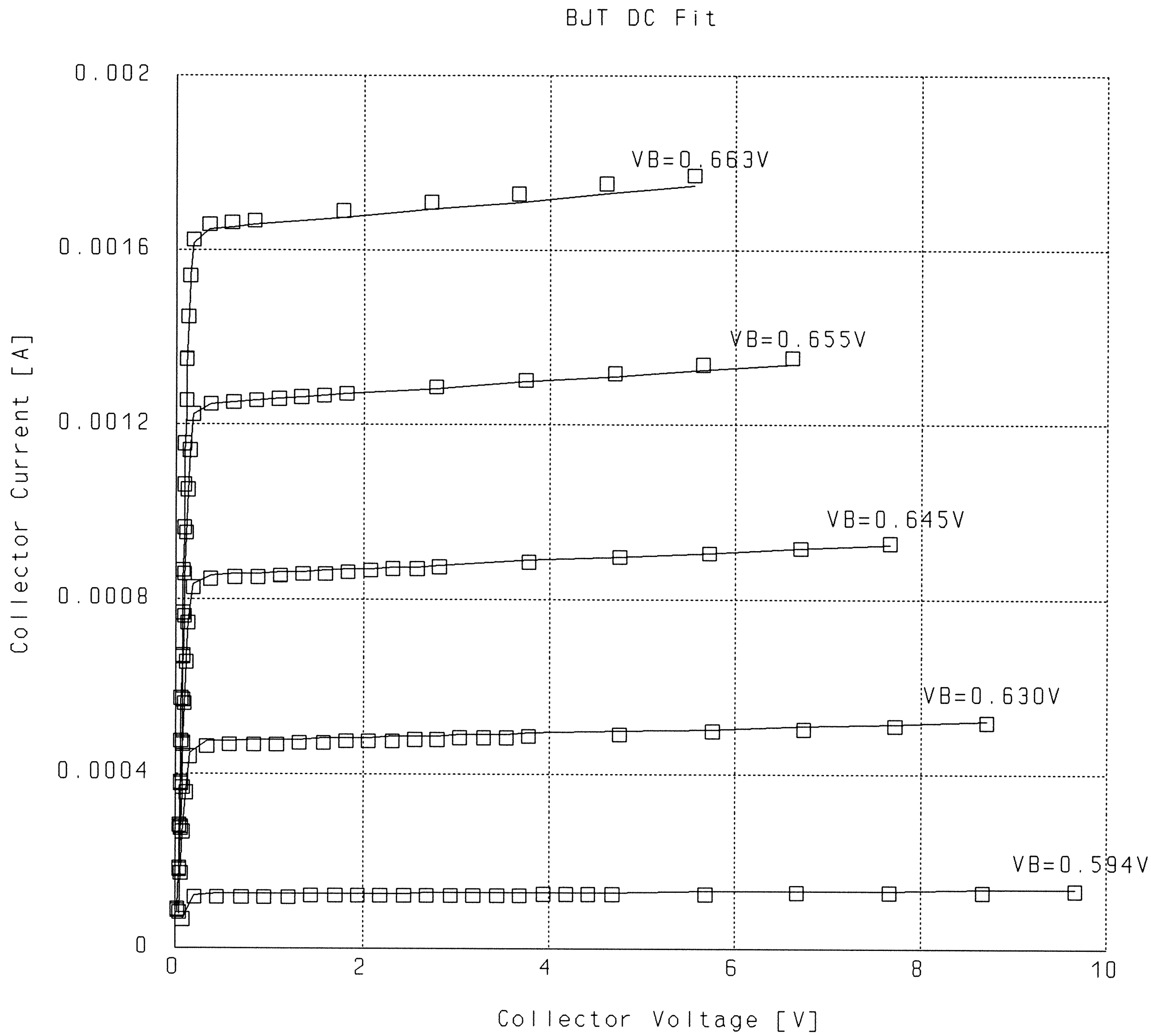


Fig. 3. DC I-V characteristic fit for the SPICE-PAC BJT model after optimization. Good fit to data can be observed. Squares (□) correspond to the data, solid lines (——) correspond to the model responses.

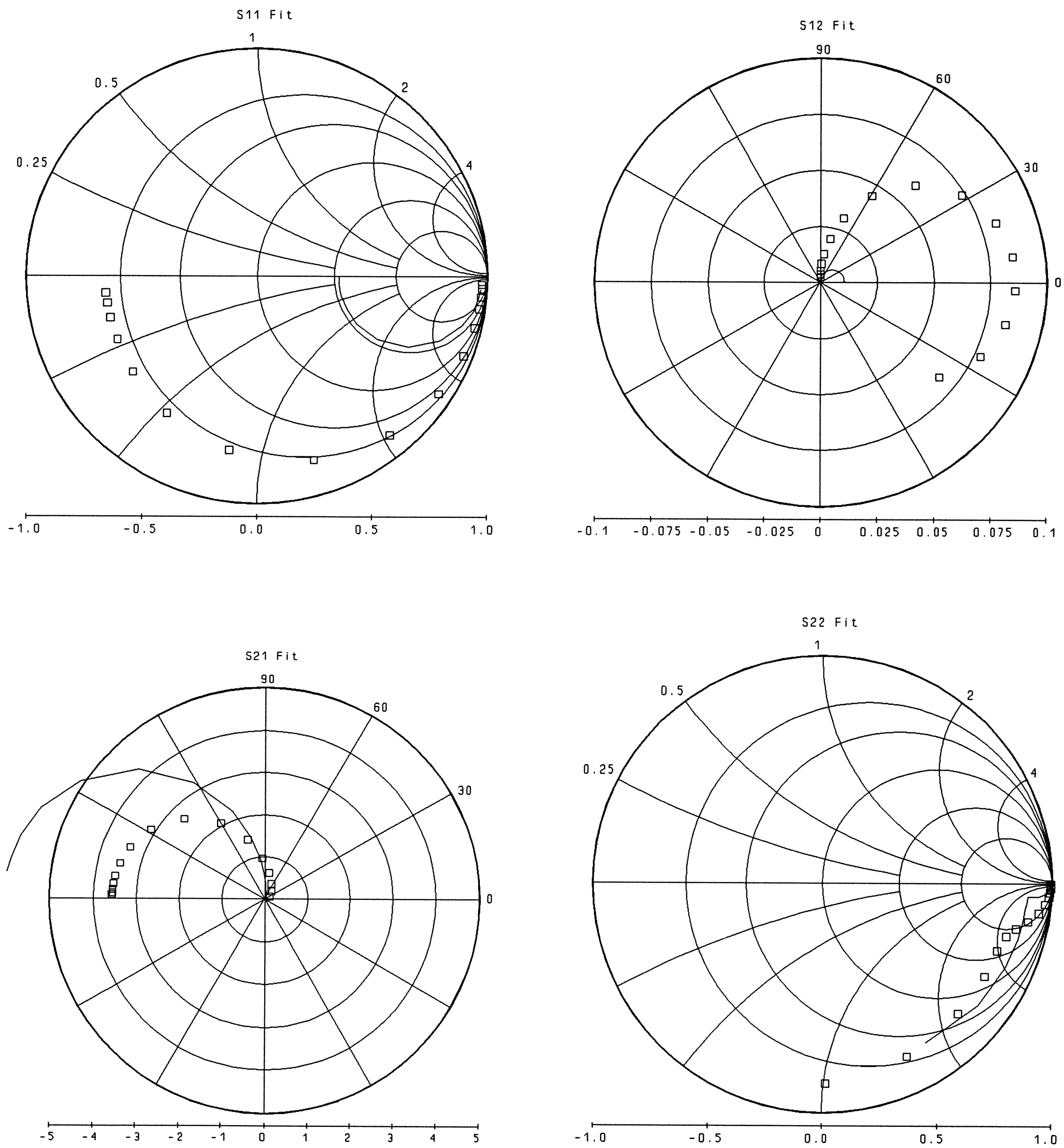


Fig. 4. Small-signal AC parameters fit for the SPICE-PAC BJT model before optimization. Squares (□) correspond to the data, solid lines (——) correspond to the model responses.

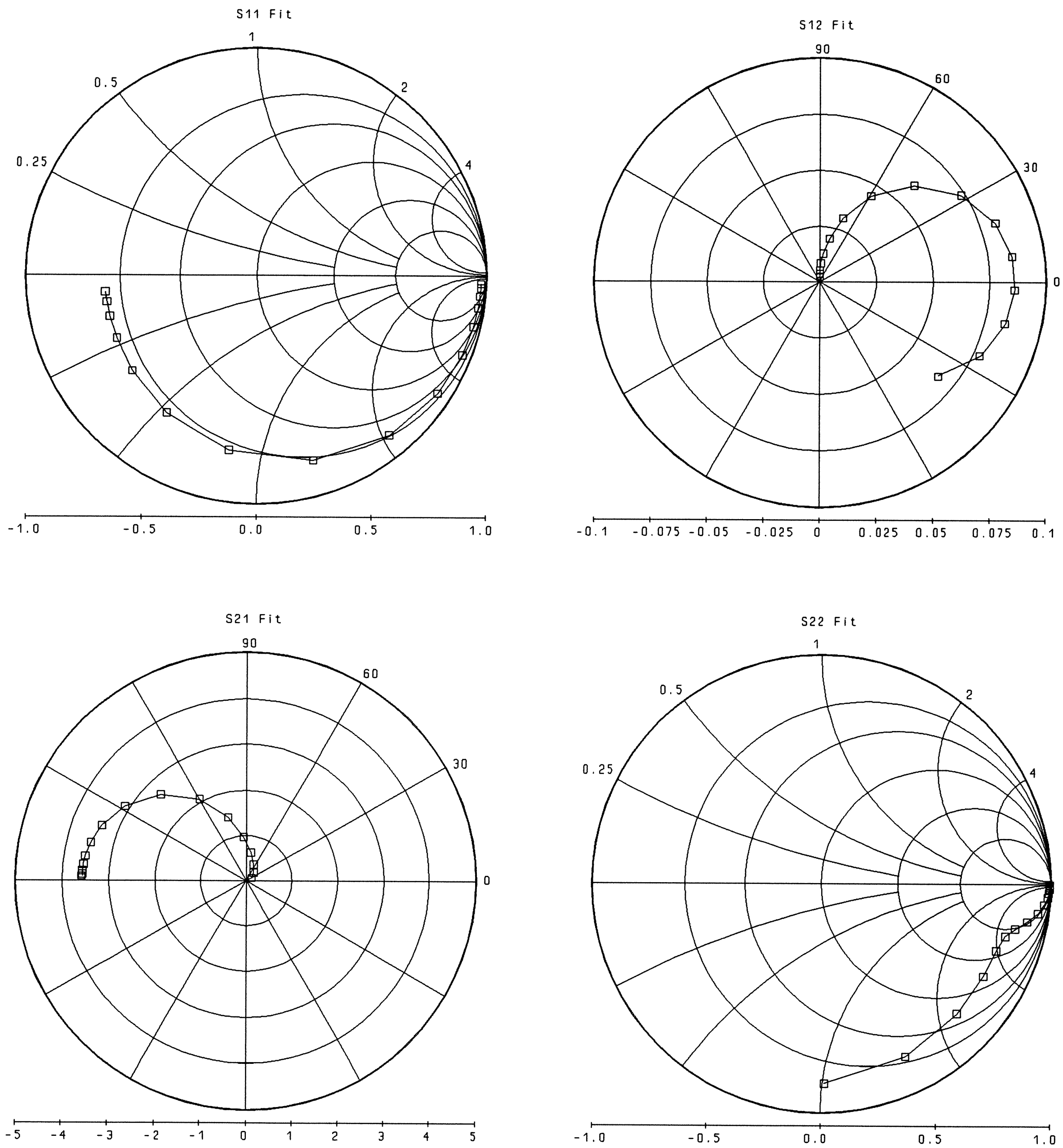


Fig. 5. Small-signal AC parameters fit for the SPICE-PAC BJT model after optimization. Very good fit to data can be observed. Squares (□) correspond to the data, solid lines (——) correspond to the model responses.

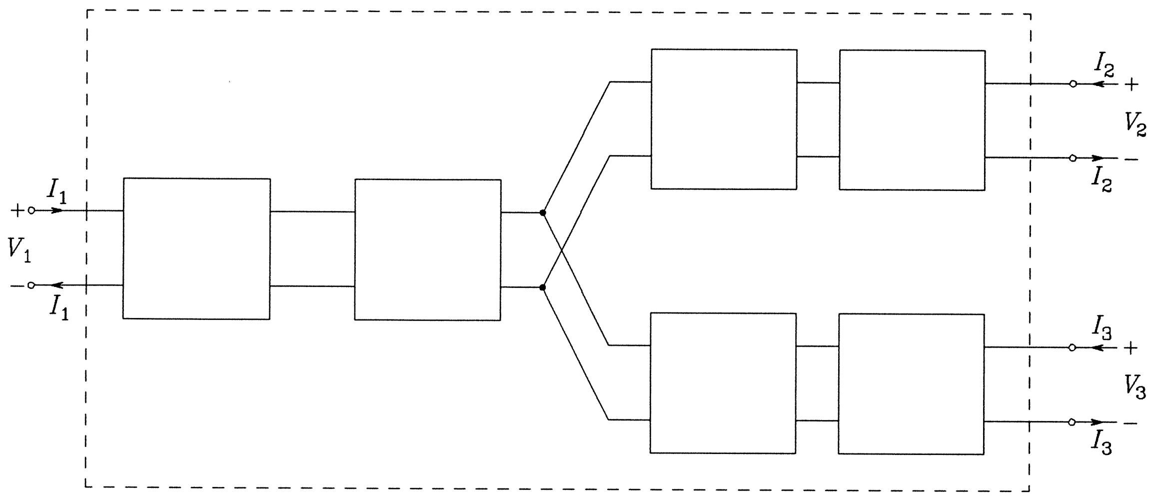


Fig. 6. An example illustrating partitioning of a large circuit into smaller subcircuits.

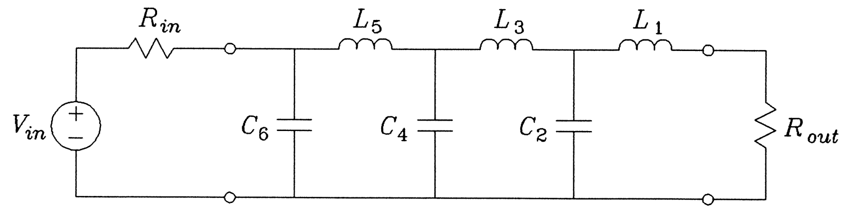


Fig. 7. LC transformer circuit. We optimize the input reflection coefficient MS11 of the transformer.

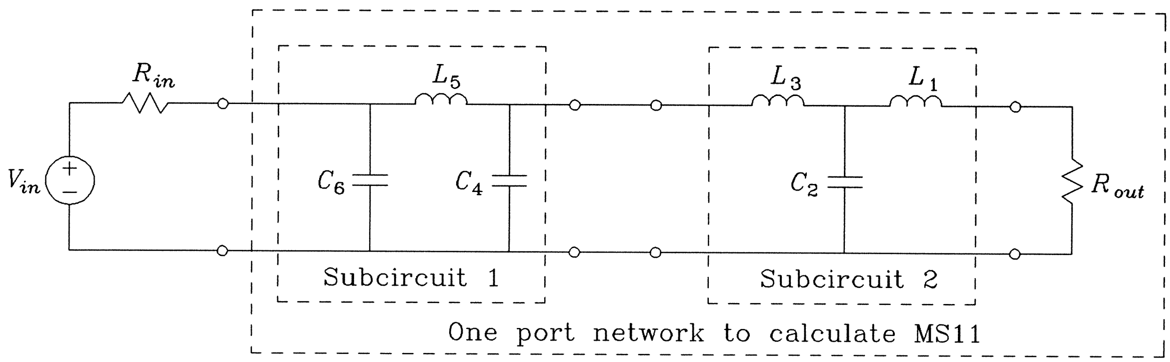
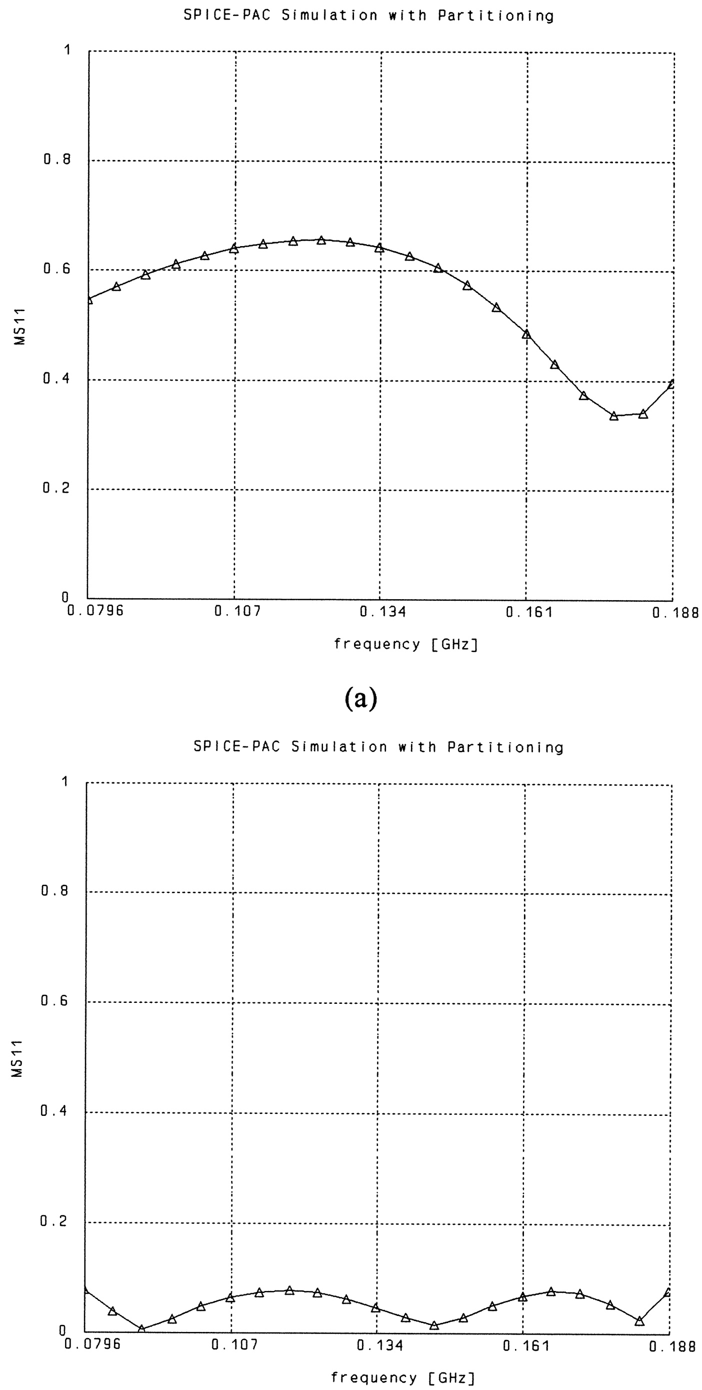


Fig. 8. LC transformer circuit. We partition the transformer into two two-port subcircuits. We use the Spicepipe connection to SPICE-PAC to simulate the subcircuits. Next, within the OSA90/hope design environment, we calculate small-signal two-port parameters of the subcircuits and let OSA90/hope compute the input reflection coefficient MS11 of the overall circuit.



(b)

Fig. 9. Optimization of the magnitude of the input reflection coefficient MS11 of the LC transformer. MS11 before (a) and after (b) minimax optimization. The triangles indicate points in the frequency sweep. The maximum value of MS11 is decreased from 0.66 in (a) to 0.076 in (b). Optimization results obtained with and without partitioning are the same.

APPENDIX A

In this appendix we list the SPICE-PAC (or SPICE) BJT model parameters used in [9] to model the 2N2222A transistor. We used that model to generate synthetic DC and small-signal AC data for the experiment described in Section IV. The data is also listed in this appendix.

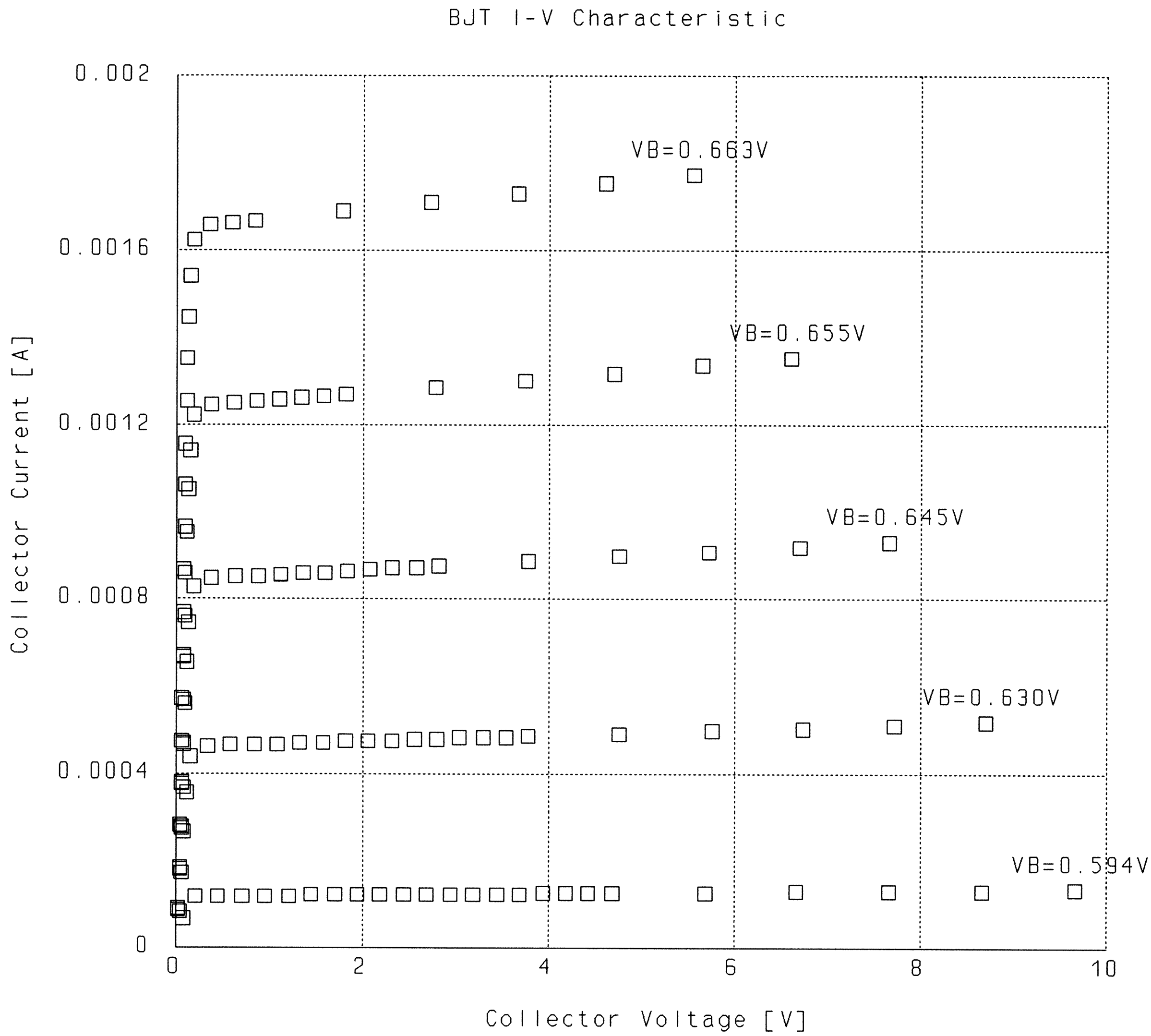
SPICE-PAC BJT Model Parameters

```
.MODEL Q2N2222A NPN(IS=14.34F XTI=3 EG=1.11 VAF=74.03 BF=255.9 NE=1.307
+ ISE=14.34F IKF=0.2847 XTB=1.5 BR=6.092 NC=2 ISC=0 IKR=0 RC=1
+ CJC=7.306P MJC=0.3416 VJC=0.75 FC=0.5 CJE=22.01P MJE=0.377 VJE=0.75
+ TR=46.92N TF=411.1P ITF=0.6 VTF=1.7 XTF=3 RB=10)
```

DC Data - Numerical Values

```
IC_DC_DATA[5,25]=[
! PARAMETER  VB=0.5942;
6.566e-05 0.0001154 0.0001164 0.0001168 0.0001172 0.0001176  0.000118 0.0001184 0.0001188
0.0001192 0.0001196 0.0001199 0.0001203 0.0001207 0.0001211 0.0001215 0.0001219 0.0001223
0.0001227 0.0001231 0.0001247 0.0001262 0.0001278 0.0001294 0.0001309
! PARAMETER  VB=0.6297;
8.104e-05 0.0001715 0.0002634 0.0003541 0.0004366 0.0004617 0.0004632 0.0004648 0.0004663
0.0004678 0.0004694 0.0004709 0.0004725  0.000474 0.0004755 0.0004771 0.0004786 0.0004802
0.0004817 0.0004832 0.0004894 0.0004956 0.0005017 0.0005079 0.0005141
! PARAMETER  VB=0.6453;
8.559e-05 0.0001784 0.0002729 0.0003682 0.0004636 0.0005587 0.0006529 0.0007447 0.0008256
0.0008446 0.0008474 0.0008502 0.0008529 0.0008557 0.0008585 0.0008613 0.0008641 0.0008669
0.0008696 0.0008724 0.0008836 0.0008947 0.0009058 0.0009169 0.0009281
! PARAMETER  VB=0.6553;
8.807e-05 0.0001819 0.0002774 0.0003736 0.0004702  0.000567 0.0006638 0.0007605  0.000857
0.0009529  0.001048   0.00114  0.001223  0.001245  0.001249  0.001253  0.001257  0.001261
 0.001265  0.001269  0.001285  0.001301  0.001317  0.001334   0.00135
! PARAMETER  VB=0.6627;
8.969e-05 0.0001843 0.0002802 0.0003769 0.0004739 0.0005713 0.0006688 0.0007663 0.0008639
0.0009615  0.001059  0.001156  0.001253  0.001349  0.001445  0.001538  0.001623  0.001655
  0.00166  0.001665  0.001687  0.001708  0.001729   0.00175  0.001772 ];

VC_DC_DATA[5,25]=[
! PARAMETER  VB=0.5942;
 0.08586  0.2116   0.459   0.708   0.957   1.206   1.455   1.704   1.953   2.202   2.451
     2.7   2.949   3.198   3.447   3.696   3.945   4.194   4.443   4.692   5.688   6.684
   7.681   8.677   9.673
! PARAMETER  VB=0.6297;
 0.04739 0.07115 0.09149  0.1148  0.1586  0.3458   0.592  0.8381   1.084    1.33   1.577
   1.823   2.069   2.315   2.561   2.807   3.053     3.3   3.546   3.792   4.776   5.761
   6.746    7.73   8.715
! PARAMETER  VB=0.6453;
 0.03602 0.05404 0.06764 0.07954 0.09104  0.1032  0.1176  0.1382  0.1861  0.3885  0.6316
  0.8746   1.118   1.361   1.604   1.847    2.09   2.333   2.576   2.819   3.791   4.763
   5.735   6.708    7.68
! PARAMETER  VB=0.6553;
 0.02982 0.04515 0.05654 0.06604 0.07455 0.08261 0.09053 0.09871  0.1075  0.1177  0.1304
  0.1495  0.1934  0.3885  0.6284  0.8683   1.108   1.348   1.588   1.828   2.787   3.747
   4.706   5.666   6.625
! PARAMETER  VB=0.6627;
 0.02578 0.03936 0.04949 0.05783 0.06513 0.07179 0.07806 0.08413 0.09016  0.0963  0.1027
  0.1096  0.1173  0.1264   0.138  0.1552   0.192  0.3625  0.5996  0.8363   1.783    2.73
   3.677   4.624   5.571];

!PARAMETER
IB_DC_DATA[5]=[1.015e-06 3.426e-06 5.887e-06 8.362e-06 1.084e-05];
VB_DC_DATA[5]=[0.5942    0.6297    0.6453    0.6553    0.6627];
```

DC Data - I-V Characteristics

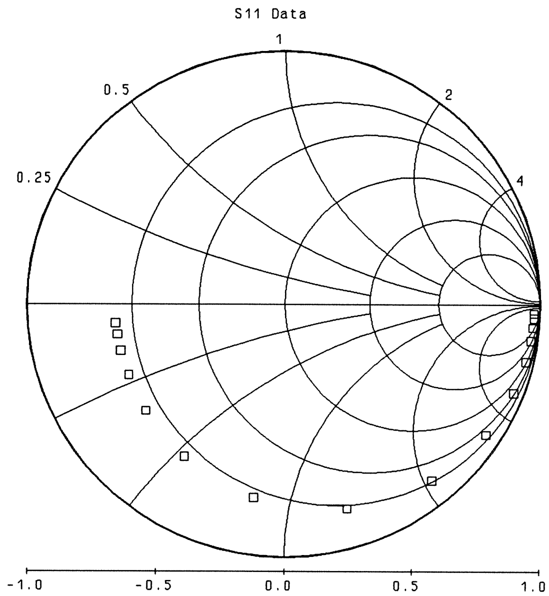


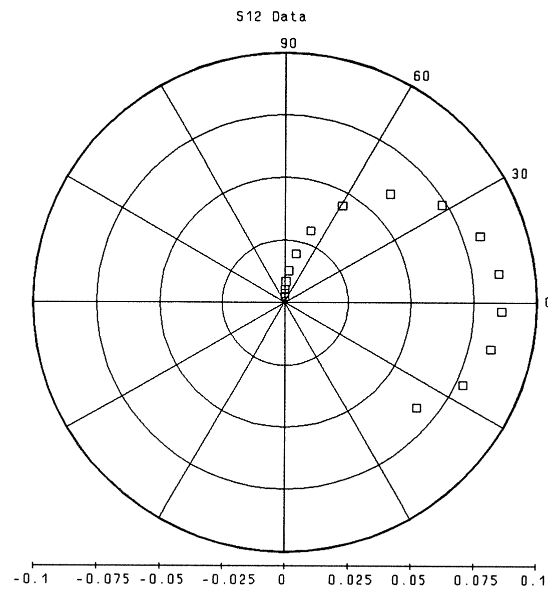
AC Data - Numerical Values

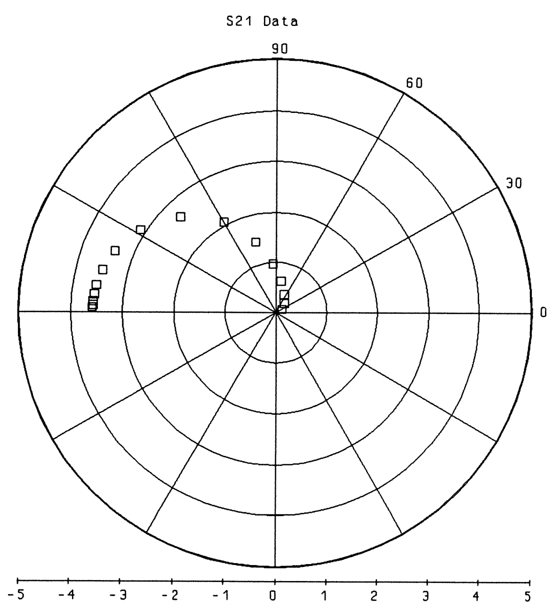
```
! FORMAT  FREQ;
!FREQS[16]=[1.5849e+06 2.51191e+06 3.98112e+06 6.30968e+06 1.00002e+07  1.5849e+07 2.51191e+07 3.98112e+07
!         6.30968e+07 1.00002e+08  1.5849e+08 2.51191e+08 3.98112e+08 6.30968e+08 1.00002e+09       1e+09];

SRI_AC_DATA[16,8]=[
! FORMAT   SR11          SR12          SR21          SR22          SI11          SI12          SI21          SI22;
! IB=5.887e06  VB=0.6453;
        0.978817  4.77822e-05      -3.55264      0.998743    -0.0369442    0.00203293     0.0855386  -0.00645005
        0.977555  0.000119931      -3.54971      0.998574    -0.0585073    0.00321939      0.135464   -0.0102166
        0.974395  0.000300668      -3.54237      0.998148    -0.0925483    0.00509211      0.214279   -0.0161684
        0.966511  0.000751581      -3.52407      0.997087     -0.145969    0.00802978      0.337962   -0.0255307
        0.947039   0.00186516      -3.47887      0.994464     -0.228566     0.0125672      0.529181   -0.0400937
        0.900132   0.00454761      -3.36998      0.988145     -0.351643     0.0193099      0.814062   -0.0621323
```

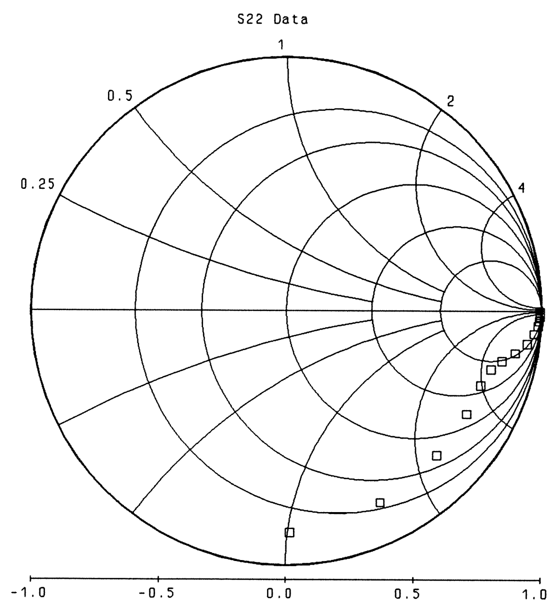
0.7936	0.0106387	-3.12267	0.973772	-0.519124	0.0284165	1.20153	-0.0933908
0.580963	0.0227909	-2.62906	0.944978	-0.701934	0.0381171	1.62378	-0.131935
0.248116	0.0417867	-1.85647	0.89942	-0.812756	0.0432449	1.87761	-0.169195
-0.116063	0.0624717	-1.01144	0.84775	-0.76838	0.0387766	1.76911	-0.198854
-0.386932	0.0775496	-0.383789	0.803649	-0.605204	0.0264052	1.38167	-0.232978
-0.536211	0.0850191	-0.0402731	0.763833	-0.424504	0.011374	0.948842	-0.295344
-0.605406	0.0863738	0.11297	0.706462	-0.281555	-0.00380488	0.597103	-0.405377
-0.635391	0.0821428	0.165698	0.592677	-0.183184	-0.0190075	0.341543	-0.568902
-0.648928	0.0709481	0.163164	0.370039	-0.119327	-0.0332592	0.163218	-0.755894
-0.656297	0.0524107	0.126391	0.0188129	-0.0784167	-0.0422878	0.0492175	-0.874062];

AC Data Graphical Display









APPENDIX B

Here, we list the OSA90/hope circuit files used in the experiments described in Section IV. The include files *npn_dc.dat* and *npn_ac.dat* contain DC and small-signal AC data listed in Appendix A. The *macros.inc* file includes OSA90/hope macro functions used to compute S parameters from SPICE-PAC responses. The *macros.inc* file is listed at the end of this appendix.

Circuit File for DC Parameter Extraction

```
#define CREATE_FILE         "create_file"
#define SPICEPIPE           "Spicepipe"
#define DUMPON             1
#define DUMPOFF            0
#define Point_NO1          20
#define Point_NO2          5
#define Point_NO           25

Expression
  char cir_contents[] =
"
Q1 5 4 0  Q2N2222A
.MODEL Q2N2222A NPN(IS=14.34F VAF=74.03 BF=255.9 NE=1.307
+ ISE=14.34F IKF=0.2847 XTB=1.5 BR=6.092 NC=2 RC=1 RB=10 RE=1F)
IBB   0  4  1
VCC   5  0  DC 10V
CDD   5  6  1M
VDD   6  0  1
.TEMP 25
.DC VDD 1 1 2
.PRINT DC I(VCC) V(5)
.END/EXT
.VAR IBB'DC
.VAR VCC'DC
.VAR Q2N2222A'IS
.VAR Q2N2222A'VAF
.VAR Q2N2222A'BF
.VAR Q2N2222A'NE
.VAR Q2N2222A'ISE
.VAR Q2N2222A'IKF
.VAR Q2N2222A'XTB
.VAR Q2N2222A'BR
.VAR Q2N2222A'NC
.VAR Q2N2222A'RC
.VAR Q2N2222A'RB
.VAR Q2N2222A'RE
.END
";
   j=1; k=1;
   #include "npn_dc.dat";
   char cir_name[]="npn.cir";
   char spp_out[]="npn.out";
   char cir_name1[]="npn.cir";
   char spp_out1[]="npn1.out";
   char dc[]=".dc";
   IBB=IB_DC_DATA[k];
   VCC=VC_DC_DATA[k,j];
! Starting point:
!   IS=?10e-15?; VAF=?100?; BF=?300?; NE=?1.5?; ISE=?10e-15?; IKF=0.2;
!   XTB=1; BR=?10?; NC=2; RC=1; RB=10; RE=1;
```

```
! Solution using L2:
   IS=?9.32736e-15?; VAF=?86.5449?; BF=?220.25?; NE=?1.3293?; ISE=?1.09935e-14?; IKF=0.2;
   XTB=1; BR=?5.94761?; NC=2; RC=1; RB=10; RE=1;

   Sp_input[14]=[ IBB VCC IS VAF BF NE ISE IKF XTB BR NC RC RB RE];

   Datapipe:       COM                        FILE = CREATE_FILE
                   N_INPUT = 2                INPUT = (cir_name, cir_contents)
                   N_OUTPUT = 1               OUTPUT = (char in_name[8]);
   DataPipe:       COM                        FILE = SPICEPIPE
                   N_INPUT = 18               INPUT = (in_name, DUMPOFF, spp_out, dc, Sp_input)
                   N_OUTPUT = 2               OUTPUT = (mIC1, VC1);

   IC1=-mIC1;
   Error=abs(IC1-IC_DC_DATA[k,j]);
End

Sweep
   Title="BJI DC Fit"
   k: from 1 to 5 step 1
   j: from 1 to Point_NO step 1  IC1 VC1 IC_DC_DATA[k,j]
      {PARAMETRIC  TITLE="BJT DC Fit"
              X_TITLE="VC [V]"
              Y_TITLE="IC [A]"
              k=ALL
              X=VC1 XMIN=0 XMAX=10 NXTICKS=5
              Y=IC1.green & IC_DC_DATA[k,j].square.green YMIN=0 YMAX=0.002 NYTICKS=5};
End

Specification
  k: 1 j: 1 2 5 10 15       from 21 to 25 step 2 Error w=10000;
  k: 2 j: 1 3 5 6 11 16     from 21 to 25 step 2 Error w=10000;
  k: 3 j: 1 4 8 9 13 17     from 21 to 25 step 2 Error w=10000;
  k: 4 j: 1 4 7 10 13 14 18 from 21 to 25 step 2 Error w=10000;
  k: 5 j: 1 5 10 15 17 18   from 21 to 25 step 2 Error w=10000;
End
```

Circuit File for Small-Signal AC Parameter Extraction

```
#define CREATE_FILE         "create_file"
#define SPICEPIPE           "Spicepipe"
#define DUMPON             1
#define DUMPOFF            0
#define Point_NO           16

Control
  Non_Microwave_Units;
End

Expression
#include "macros.inc"; ! Macros for complex arithmetic and V2Y and Y2S conversions
#include "npn_ac.dat"; ! Array of S parameters (real/imaginary) SRI_AC_DATA[16,8]
  char cir_contents[] =
"
Q1 5 4 0  Q2N2222A
.MODEL Q2N2222A NPN(IS=14.34F XTI=3 EG=1.11 VAF=74.03 BF=255.9 NE=1.307
+ ISE=14.34F IKF=0.2847 XTB=1.5 BR=6.092 NC=2 ISC=0 IKR=0. RC=1
+ CJC=7.306P MJC=0.3416 VJC=0.75 FC=0.5 CJE=22.01P MJE=0.377 VJE=0.75
+ TR=46.92N TF=411.1P RB=10 RE=0.01)
IIN   0  2  AC 2U
RIN   2  0  50K
CBB   2  4  10M
IBB   0  4  DC 5.887U
RCC   5  6  2.5K
VCC   6  0  DC 10V
CCB   5  7  10M
RL    7  0  2.5K
.TEMP 25
```

```
.AC DEC 5 1MEG 1G
.OPTION CPTIME=9999;
.PRINT AC VR(2) VI(2) VR(7) VI(7)
.DC VCC 10 10 10
.PRINT DC V(4) V(5) I(VCC)
.END/EXT
.VAR IIN'ACM
.VAR RIN
.VAR RCC
.VAR RL
.VAR Q2N2222A'IS
.VAR Q2N2222A'VAF
.VAR Q2N2222A'BF
.VAR Q2N2222A'NE
.VAR Q2N2222A'ISE
.VAR Q2N2222A'IKF
.VAR Q2N2222A'BR
.VAR Q2N2222A'NC
.VAR Q2N2222A'RC
.VAR Q2N2222A'CJC
.VAR Q2N2222A'MJC
.VAR Q2N2222A'CJE
.VAR Q2N2222A'MJE
.VAR Q2N2222A'TR
.VAR Q2N2222A'TF
.VAR Q2N2222A'RB
.VAR Q2N2222A'RE
.END
";
   j=1;
   IIN=2UA;
   RIN=50KOH;
   RCC=2.5KOH;
   RL1=2.5KOH;
   RL2=5.0KOH;
   char cir_name[]="npn.cir";
   char spp_out[]="npn.out";
   char ac[]=".ac";
   char dc[]=".dc";

! Starting point:
!   IS=?10e-15?; VAF=?100?; BF=?300?; NE=?1.5?; ISE=?10e-15?; IKF=0.2; BR=?0.01?;
!   NC=2; RC=?10?; CJC=?5e-12?; MJC=?0.5?; CJE=?51e-12?; MJE=?0.5?; TR=?10e-9?;
!   TF=?100e-12?; RB=?100?; RE=?1?;

! Solution using L1:
   IS=?2.52593e-14?; VAF=?262.206?; BF=?354.898?; NE=?1.22454?; ISE=?7.83178e-15?; IKF=0.2; BR=?0.0395198?;
   NC=2; RC=?1.04947?; CJC=?4.75128e-12?; MJC=?0.158088?; CJE=?4.27953e-11?; MJE=?0.0758791?; TR=?3.59663e-08?;
   TF=?1.01932e-10?; RB=?9.91971?; RE=?0.0212473?;

   Sp_input[21]=[ IIN RIN RCC RL1 IS VAF BF NE ISE IKF BR NC RC CJC MJC CJE MJE TR TF RB RE];
   Datapipe:      COM                        FILE = CREATE_FILE
                  N_INPUT = 2                INPUT = (cir_name, cir_contents)
                  N_OUTPUT = 1               OUTPUT = (char in_name[8]);
   DataPipe:      COM                        FILE = SPICEPIPE
                  N_INPUT = 25               INPUT = (in_name, DUMPOFF, spp_out, ac, Sp_input)
                  N_OUTPUT = (4*Point_NO)    OUTPUT = (RV1m[Point_NO], IV1m[Point_NO],
                                                       RV2m[Point_NO], IV2m[Point_NO]);
   DataPipe:      COM                        FILE = SAME
                  N_INPUT = 25               INPUT = (in_name, DUMPOFF, spp_out, dc, Sp_input)
                  N_OUTPUT = 3               OUTPUT = (VB, VC, IC);
   Sp_input1[21]=[ IIN RIN RCC RL2 IS VAF BF NE ISE IKF BR NC RC CJC MJC CJE MJE TR TF RB RE];
   DataPipe:      COM                        FILE = SAME
                  N_INPUT = 25               INPUT = (in_name, DUMPOFF, spp_out, ac, Sp_input1)
                  N_OUTPUT = (4*Point_NO)    OUTPUT = (RV1'm[Point_NO], IV1'm[Point_NO],
                                                       RV2'm[Point_NO], IV2'm[Point_NO]);
   DataPipe:      COM                        FILE = SAME
                  N_INPUT = 25               INPUT = (in_name, DUMPOFF, spp_out, dc, Sp_input1)
                  N_OUTPUT = 3               OUTPUT = (VB', VC', IC');
```

```
! Define parameters for Y calculations:
  RV1=RV1m[j]; IV1=IV1m[j];
  RV2=RV2m[j]; IV2=IV2m[j];
  RV1'=RV1'm[j]; IV1'=IV1'm[j];  RV2'=RV2'm[j]; IV2'=IV2'm[j];
  RIs=IIN;       IIs=0;  RYs=1/RIN;       IYs=0;  RY1=(RCC+RL1)/(RCC*RL1); IY1=0;
  RIs'=IIN;      IIs'=0; RYs'=1/RIN;      IYs'=0; RY1'=(RCC+RL2)/(RCC*RL2);        IY1'=0;
!----------------------------------------------------------------------
! Extract Y parameters from voltages
   V2Y(RV1,  IV1,  RV2,  IV2,  RV1', IV1', RV2', IV2',
       RIs,  IIs,  RYs,  IYs,  RY1,  IY1,  RIs', IIs', RYs', IYs', RY1', IY1', Y);
!----------------------------------------------------------------------
! Convert Y parameters into S parameters
  RY0=1/50; IY0=0;
  Y2S(RY0, IY0, Y[1], Y[2], Y[3], Y[4], Y[5], Y[6], Y[7], Y[8], S);

  RI2MP(Y[1], Y[5], YM11, YP11);  RI2MP(Y[2], Y[6], YM12, YP12);
  RI2MP(Y[3], Y[7], YM21, YP21);  RI2MP(Y[4], Y[8], YM22, YP22);
  RI2MP(SR11, SI11, SM11, SP11);  RI2MP(SR12, SI12, SM12, SP12);
  RI2MP(SR21, SI21, SM21, SP21);  RI2MP(SR22, SI22, SM22, SP22);

  SS[8]=[SR11 SR12 SR21 SR22 SI11 SI12 SI21 SI22];
  SEL_SP_DATA[8]=row(SRI_AC_DATA, j);
  SP_ERROR[8]=abs(SS - SEL_SP_DATA);
End

Sweep
  Title="S Fit" j: from 1 to Point_NO step 1 SR11 SR12 SR21 SR22 SI11 SI12 SI21 SI22 SEL_SP_DATA
  {SMITH    TITLE="S11 Fit"
            RI=(SR11, SI11).green & (SEL_SP_DATA[1], SEL_SP_DATA[5]).green.square}
  {POLAR    TITLE="S12 Fit"
            RI=(SR12, SI12).green & (SEL_SP_DATA[2], SEL_SP_DATA[6]).green.square YMAX=0.1 NYTICKS=4}
  {POLAR    TITLE="S21 Fit"
            RI=(SR21, SI21).green & (SEL_SP_DATA[3], SEL_SP_DATA[7]).green.square YMAX=5 NYTICKS=5}
  {SMITH    TITLE="S22 Fit"
            RI=(SR22, SI22).green & (SEL_SP_DATA[4], SEL_SP_DATA[8]).green.square};
End

Specification
  j: from 1 to Point_NO step 1 SP_ERROR w=100;
End
```

OSA90/hope Macro Functions to Postprocess SPICE-PAC Results

```
!-------------------------------------------------------------------
! Complex addition
#define Cadd(sum, jsum, x, jx, y, jy){
         sum = x + y;    jsum = jx + jy;
}
!-------------------------------------------------------------------
! Complex subtraction
#define Csub(dif, jdif, x, jx, y, jy){
         dif = x - y;    jdif = jx - jy;
}
!-------------------------------------------------------------------
! Complex multiplication
#define Cmul(mul, jmul, x, jx, y, jy){
         mul = x*y - jx*jY;      jmul = x*jy + jx*y;
}
!-------------------------------------------------------------------
! Complex division
#define Cdiv(div, jdiv, x, jx, y, jy){
         div = (x*y + jx*jY)/(y*y + jy*jy);       jdiv = (jx*y - x*jy)/(y*y + jy*jy);
}
!-------------------------------------------------------------------
! Calculates S parameters from Y parameters
#define Y2S(RY0, IY0, RY11, RY12, RY21, RY22, IY11, IY12, IY21, IY22, $){
  Cadd($Rt1, $It1, RY11, IY11, RY0, IY0);
  Cadd($Rt2, $It2, RY22, IY22, RY0, IY0);
```

```
  Cmul($Rt3, $It3, RY12, IY12, RY21, IY21);
  Cmul($Rt4, $It4, $Rt1, $It1, $Rt2, $It2);
  Csub($RDY, $IDY, $Rt4, $It4, $Rt3, $It3);
! S11:
  Csub($Rt5, $It5, RY0, IY0, RY11, IY11);
  Cmul($Rt6, $It6, $Rt5, $It5, $Rt2, $It2);
  Cadd($Rt7, $It7, $Rt6, $It6, $Rt3, $It3);
  Cdiv($R11, $I11, $Rt7, $It7, $RDY, $IDY);
! S12:
  Cmul($Rt8, $It8, (-2*RY12), (-2*IY12), RY0, IY0);
  Cdiv($R12, $I12, $Rt8, $It8, $RDY, $IDY);
! S21:
  Cmul($Rt9, $It9, (-2*RY21), (-2*IY21), RY0, IY0);
  Cdiv($R21, $I21, $Rt9, $It9, $RDY, $IDY);
! S22:
  Csub($Rt10, $It10, RY0, IY0, RY22, IY22);
  Cmul($Rt11, $It11, $Rt1, $It1, $Rt10, $It10);
  Cadd($Rt12, $It12, $Rt11, $It11, $Rt3, $It3);
  Cdiv($R22, $I22, $Rt12, $It12, $RDY, $IDY);
}
!------------------------------------------------------------------
! Calculates Y parameters from voltages
#define V2Y(RV1,  IV1,  RV2,  IV2,  RV1', IV1', RV2', IV2',
            RIs,  IIs,  RYs,  IYs,  RYl,  IYl,  RIs', IIs', RYs', IYs', RYl', IYl', $){
  $RV[4,4]=[ RV1  RV2    0    0
               0    0  RV1  RV2
            RV1' RV2'    0    0
               0    0 RV1' RV2'];
  $IV[4,4]=[  IV1  IV2    0    0
               0    0  IV1  IV2
            IV1' IV2'    0    0
               0    0 IV1' IV2'];
! Set up the expanded V matrix
  $V[8,8]=[$RV  -$IV
           $IV   $RV];
! Define the real and imaginary right-hand side
  $RI[4]=[ (RIs - (RV1*RYs - IV1*IYs))
           -(RV2*RYl - IV2*IYl)
           (RIs' - (RV1'*RYs' - IV1'*IYs'))
           -(RV2'*RYl' - IV2'*IYl')];
  $II[4]=[ (IIs - (IV1*RYs + RV1*IYs))
           -(IV2*RYl + RV2*IYl)
           (IIs' - (IV1'*RYs' + RV1'*IYs'))
           -(IV2'*RYl' + RV2'*IYl') ];
! Set up the expanded left-hand side
  $I[8]=[ $RI $II ];
! Calculate the Y parameters by LU factorization
  $VLU[9,8]=LU($V);
  $[8]=subst($VLU, $I);
}
```

APPENDIX C

Here we list two circuit files used to optimize the LC transformer circuit described in Section V. In the first circuit file the LC transformer is simulated as one circuit. In the second circuit file the transformer circuit is partitioned into two two-port subcircuits.

The *macros.inc* include file used in both circuit files is listed at the end of Appendix B. The source code of the mirror Datapipe program is listed at the end of this appendix. We use this program to incorporate the small-signal multiport parameters of circuits simulated by SPICE-PAC into the OSA90/hope circuit description.

Spicepipe1 Datapipe executable program, used in the second circuit file, is a soft link to Spicepipe. By using Spicepipe1 rather than Spicepipe for the second subcircuit we eliminate unnecessary reparsing of the SPICE-PAC input file by SPICE-PAC in each optimization iteration. Such reparsing would be necessary for each subcircuit because OSA90/hope calls the subcircuits interchangeably.

Circuit File for the LC transformer Optimization without Partitioning

```
#define CREATE_FILE        "create_file"
#define SPICEPIPE          "Spicepipe"
#define MIRROR             "mirror"
#define DUMPON             1
#define DUMPOFF            0
#define Point_NO           21

Control
  Non_Microwave_Units;
End

Expression
#include "macros.inc"; ! Macros for complex arithmetics and V2Y and Y2S conversions
  char cir_contents[]=
"
VIN   1 0 AC 1
RIN   1 2 3
C6    2 0 1
C4    3 0 1
C2    4 0 1
L5    2 3 1
L3    3 4 1
L1    4 5 1
ROUT  5 0 1
.PRINT AC VR(2) VI(2) VR(5) VI(5)
.AC LIN 21 0.079578G 0.187644G
.END/EXT
.VAR VIN'ACM
.VAR RIN
.VAR ROUT
.VAR L1
```

```
.VAR L3
.VAR L5
.VAR C2
.VAR C4
.VAR C6
.END
";
   L1=1.04114NH; L3=2.34056NH; L5=2.93737NH; C2=0.979122NF; C4=0.780183NF; C6=0.347045NF;
   VIN=1V;
   RIN=3OH;
   ROUT1=1OH;
   char cir_name[]="lc6.cir";
   char out_name[]="lc6.out";
   char ac[]=".ac";
   input1[1:9]=[VIN, RIN, ROUT1, L1, L3, L5, C2, C4, C6];
   Datapipe:    COM                   FILE = CREATE_FILE
                N_INPUT = 2           INPUT = (cir_name, cir_contents)
                N_OUTPUT = 1          OUTPUT = (char in_name[8]);
   DataPipe:    COM                   FILE = SPICEPIPE
                N_INPUT = 13          INPUT = (in_name, DUMPOFF, out_name, ac, input1)
               N_OUTPUT = (4*Point_NO)  OUTPUT = (RV1m[Point_NO], IV1m[Point_NO],
                                            RV2m[Point_NO], IV2m[Point_NO]);
   ROUT2=2OH;
   input2[1:9]=[VIN, RIN, ROUT2, L1, L3, L5, C2, C4, C6];
   DataPipe:    COM                   FILE = SAME
                N_INPUT = 13          INPUT = (in_name, DUMPOFF, out_name, ac, input2)
               N_OUTPUT = (4*Point_NO)  OUTPUT = (RV1'm[Point_NO], IV1'm[Point_NO],
                                            RV2'm[Point_NO], IV2'm[Point_NO]);
  j=1;
! Define parameters for Y calculations:
  RV1=RV1m[j]; IV1=IV1m[j];  RV2=RV2m[j]; IV2=IV2m[j];
  RV1'=RV1'm[j]; IV1'=IV1'm[j];  RV2'=RV2'm[j]; IV2'=IV2'm[j];
  RIs=VIN/RIN;   IIs=0;  RYs=1/RIN;      IYs=0;  RYl=1/ROUT1;     IYl=0;
  RIs'=VIN/RIN;  IIs'=0; RYs'=1/RIN;     IYs'=0; RYl'=1/ROUT2;    IYl'=0;
!---------------------------------------------------------------------
! Extract Y parameters from voltages
   V2Y(RV1, IV1, RV2, IV2, RV1', IV1', RV2', IV2',
       RIs, IIs, RYs, IYs, RYl, IYl, RIs', IIs', RYs', IYs', RYl', IYl', Y);
!---------------------------------------------------------------------
! Convert Y parameters into S parameters
  RY0=1/50; IY0=0;
  Y2S(RY0, IY0, Y[1], Y[2], Y[3], Y[4], Y[5], Y[6], Y[7], Y[8], S);

  RI2MP(Y[1], Y[5], YM11, YP11);  RI2MP(Y[2], Y[6], YM12, YP12);
  RI2MP(Y[3], Y[7], YM21, YP21);  RI2MP(Y[4], Y[8], YM22, YP22);
  RI2MP(SR11, SI11, SM11, SP11);  RI2MP(SR12, SI12, SM12, SP12);
  RI2MP(SR21, SI21, SM21, SP21);  RI2MP(SR22, SI22, SM22, SP22);
End

Model
  DataPipe:      LINEAR          FILE = MIRROR    FORMAT=S;
                 N_INPUT = 8     INPUT = (SR11, SI11, SR12, SI12, SR21, SI21, SR22, SI22)
                 N_PORT = 2      NAME=lc6
  PORT 1 0 R=RIN;
  DATAPORT 1 2 0 DATA=lc6;
  RES 2 0 R=1OH;
  Circuit;
  st=(0.187644GHz-0.079578GHz)/20;
  fr=0.079578GHz+(j-1)*st;
End

Sweep
  AC: Title="AC MS11"  j: from 1 to Point_NO step 1 freq=fr Rref=3 fr MS11
      {PARAMETRIC  TITLE="MS11 of a Circuit Containing 2 Spice Subcircuits"
               X_TITLE="freq [Hz]"
               Y_TITLE="MS11"
               j=all
               X=fr
               Y=MS11.green.triangle & MS11.green YMIN=0 YMAX=1 NYTICKS=5};
```

```
End

Spec
  AC: j: from 1 to Point_NO step 1 freq=fr Rref=3 MS11<0;
End
```

Circuit File for the LC transformer Optimization with Partitioning

```
#define CREATE_FILE        "create_file"
#define SPICEPIPE          "Spicepipe"
#define SPICEPIPE1         "Spicepipe1"
#define MIRROR             "mirror"
#define DUMPON             1
#define DUMPOFF            0
#define Point_NO           21

Control
  Non_Microwave_Units;
End

Expression
#include "macros.inc"; ! Macros for complex arithmetics and V2Y and Y2S conversions
  char cir_contentsA[]=
"
VIN   1 0 AC 1
RIN   1 2 3
C6    2 0 1
L5    2 3 1
C4    3 0 1
ROUT  3 0 1
.PRINT AC VR(2) VI(2) VR(3) VI(3)
.AC LIN 21 0.079578G 0.187644G
.END/EXT
.VAR VIN'ACM
.VAR RIN
.VAR ROUT
.VAR L5
.VAR C4
.VAR C6
.END
";
  char cir_contentsB[]=
"
VIN   1 0 AC 1
RIN   1 2 3
L3    2 3 1
C2    3 0 1
L1    3 4 1
ROUT  4 0 1
.PRINT AC VR(2) VI(2) VR(4) VI(4)
.AC LIN 21 0.079578G 0.187644G
.END/EXT
.VAR VIN'ACM
.VAR RIN
.VAR ROUT
.VAR C2
.VAR L1
.VAR L3
.END
";
   L1=?1NH?; L3=?1NH?; L5=?1NH?; C2=?1NF?; C4=?1NF?; C6=?1NF?;
   VIN=1V;
   RIN=3OH;
   ROUT1=1OH;
   char cir_nameA[]="lc6A.cir";
   char out_nameA[]="lc6A.out";
   char ac[]=".ac";
   inputA1[1:6]=[VIN, RIN, ROUT1, L5, C4, C6];
```

```
   Datapipe:     COM                       FILE = CREATE_FILE
                 N_INPUT = 2               INPUT = (cir_nameA, cir_contentsA)
                 N_OUTPUT = 1              OUTPUT = (char in_nameA[9]);
   DataPipe:     COM                       FILE = SPICEPIPE
                 N_INPUT = 10              INPUT = (in_nameA, DUMPOFF, out_nameA, ac, inputA1)
                N_OUTPUT = (4*Point_NO)    OUTPUT = (RV1Am[Point_NO], IV1Am[Point_NO],
                                                   RV2Am[Point_NO], IV2Am[Point_NO]);
   ROUT2=2OH;
   inputA2[1:6]=[VIN, RIN, ROUT2, L5, C4, C6];
   DataPipe:     COM                       FILE = SAME
                 N_INPUT = 10              INPUT = (in_nameA, DUMPOFF, out_nameA, ac, inputA2)
                N_OUTPUT = (4*Point_NO)    OUTPUT = (RV1A'm[Point_NO], IV1A'm[Point_NO],
                                                   RV2A'm[Point_NO], IV2A'm[Point_NO]);
   char cir_nameB[]="lc6B.cir";
   char out_nameB[]="lc6B.out";
   inputB1[1:6]=[VIN, RIN, ROUT1, C2, L1, L3];
   Datapipe:     COM                       FILE = CREATE_FILE
                 N_INPUT = 2               INPUT = (cir_nameB, cir_contentsB)
                 N_OUTPUT = 1              OUTPUT = (char in_nameB[9]);
   DataPipe:     COM                       FILE = SPICEPIPE1
                 N_INPUT = 10              INPUT = (in_nameB, DUMPOFF, out_nameB, ac, inputB1)
                N_OUTPUT = (4*Point_NO)    OUTPUT = (RV1Bm[Point_NO], IV1Bm[Point_NO],
                                                   RV2Bm[Point_NO], IV2Bm[Point_NO]);
   inputB2[1:6]=[VIN, RIN, ROUT2, C2, L1, L3];
   DataPipe:     COM                       FILE = SAME
                 N_INPUT = 10              INPUT = (in_nameB, DUMPOFF, out_nameB, ac, inputB2)
                N_OUTPUT = (4*Point_NO)    OUTPUT = (RV1B'm[Point_NO], IV1B'm[Point_NO],
                                                   RV2B'm[Point_NO], IV2B'm[Point_NO]);
  j=1;
! Define parameters for Y calculations (circuit A):
  RV1A=RV1Am[j]; IV1A=IV1Am[j];      RV2A=RV2Am[j]; IV2A=IV2Am[j];
  RV1A'=RV1A'm[j]; IV1A'=IV1A'm[j];  RV2A'=RV2A'm[j]; IV2A'=IV2A'm[j];
  RIs=VIN/RIN;   IIs=0;  RYs=1/RIN;        IYs=0;  RY1=1/ROUT1;      IY1=0;
  RIs'=VIN/RIN;  IIs'=0; RYs'=1/RIN;       IYs'=0; RY1'=1/ROUT2;     IY1'=0;
!--------------------------------------------------------------------
! Extract Y parameters from voltages
   V2Y(RV1A, IV1A, RV2A, IV2A, RV1A', IV1A', RV2A', IV2A',
       RIs, IIs, RYs, IYs, RY1, IY1, RIs', IIs', RYs', IYs', RY1', IY1', YA);
!--------------------------------------------------------------------
! Convert Y parameters into S parameters
  RY0=1/50; IY0=0;
  Y2S(RY0, IY0, YA[1], YA[2], YA[3], YA[4], YA[5], YA[6], YA[7], YA[8], SA);

! Define parameters for Y calculations (circuit B):
  RV1B=RV1Bm[j]; IV1B=IV1Bm[j];  RV2B=RV2Bm[j]; IV2B=IV2Bm[j];
  RV1B'=RV1B'm[j]; IV1B'=IV1B'm[j];  RV2B'=RV2B'm[j]; IV2B'=IV2B'm[j];
!--------------------------------------------------------------------
! Extract Y parameters from voltages
   V2Y(RV1B, IV1B, RV2B, IV2B, RV1B', IV1B', RV2B', IV2B',
       RIs, IIs, RYs, IYs, RY1, IY1, RIs', IIs', RYs', IYs', RY1', IY1', YB);
!--------------------------------------------------------------------
! Convert Y parameters into S parameters
  Y2S(RY0, IY0, YB[1], YB[2], YB[3], YB[4], YB[5], YB[6], YB[7], YB[8], SB);
End

Model
   DataPipe:     LINEAR          FILE = MIRROR
                 N_INPUT = 8     INPUT = (SAR11, SAI11, SAR12, SAI12, SAR21, SAI21, SAR22, SAI22)
                N_PORT = 2       NAME=CircuitA FORMAT=S;
   DataPipe:     LINEAR          FILE = SAME
                 N_INPUT = 8     INPUT = (SBR11, SBI11, SBR12, SBI12, SBR21, SBI21, SBR22, SBI22)
                N_PORT = 2       NAME=CircuitB FORMAT=S;
   PORT 1 0 R=RIN;
   DATAPORT 1 2 DATA=CircuitA;
   DATAPORT 2 3 DATA=CircuitB;
   RES 3 0 R=1OH;
   CIRCUIT;
   st=(0.187644GHz-0.079578GHz)/20;
   fr=0.079578GHz+(j-1)*st;
```

```
End

Sweep
  AC: Title="AC MS11"  j: from 1 to Point_NO step 1 freq=fr Rref=3 fr MS11
      {PARAMETRIC  TITLE="MS11 of a Circuit Containing 2 Spice Subcircuits"
               X_TITLE="freq [Hz]"
               Y_TITLE="MS11"
               j=all
               X=fr
               Y=MS11.green.triangle & MS11.green YMIN=0 YMAX=1 NYTICKS=5};
End

Spec
  AC: j: from 1 to Point_NO step 1 freq=fr Rref=3 MS11<0;
End
```

Source Code of the Mirror Datapipe Program

```
#include <stdio.h>
#include "ippcv2.h"
void main()
{
  int    input_no, output_no, i, error=0;
  float  input[8];
  char*  error_str;

  for (;;)
  {
    pipe_initialize2();
    pipe_read2(&input_no, sizeof(int), 1);
    pipe_read2(&output_no, sizeof(int), 1);
    pipe_read2(input, sizeof(int), input_no);
    if(input_no!=output_no)
    {
      error_str="mirror: Number of inputs different from 2*N_PORT^2.";
      error=strlen(error_str)+1;
    }
    pipe_write2(&error, sizeof(int), 1);
    if(error)
      pipe_write2(error_str, 1, error);
    else
      pipe_write2(input, sizeof(float), output_no);
  }
}
```