IEEE Int. Conference on Computer Design (ICCD'84); Port Chester, NY, October 8-11, 1984, pp.492-496.

SPICE-PAC, A PACKAGE OF SUBROUTINES FOR INTERACTIVE SIMULATION AND OPTIMIZATION OF CIRCUITS

W.M. Zuberek

Department of Computer Science
Memorial University of Newfoundland
St. John's, NL, Canada A1C–5S7

Abstract. SPICE-PAC is a package (or a set) of subroutines which is functionally equivalent to the SPICE 2G circuit simulation program, i.e., it accepts the same circuit description and performs all the analyses which are available in the SPICE 2G programs, but also provides an access to internal values of circuit elements, dynamic definitions of parameters and outputs, hierarchical naming scheme for subcircuit elements, parameterized subcircuit expansions, an interface to libraries of standard modules, and a set of run-time diagnostics to control the package actions. Two immediate applications of the package are: (1) interactive circuit simulation in which an interactive driver controls the simulation subroutines according to user commands, and (2) circuit optimization in which an optimization package is interfaced with the simulation package by user-supplied subroutines that evaluate objective functions, constrains, etc. Some results of very simple examples are included as an illustration.

1. INTRODUCTION

Computer-aided circuit analysis or circuit simulation, which matured in the 1970's, has established itself as a significant tool for analysis and design of integrated circuits. The SPICE-2 program [1,3,4] developed at the Department of Electrical Engineering and Computer Sciences, University of California in Berkeley, has become one of the most popular "second-generation" circuit simulators. It provides several linear and nonlinear analyses, including DC operating point, nonlinear DC transfer curves, nonlinear transient, small-signal frequency domain, noise, distortion and Fourier. Circuits may contain resistors, capacitors, inductors and mutual inductors, independent linear and nonlinear voltage and current sources, four types of dependent sources, transmission lines, and the four most common semiconductor devices: diodes, bipolar junction transistors, junction field effect transistors and metal-oxide-semiconductor field effect transistors (MOSFETs). SPICE has built-in models for semiconductor devices and the users need to specify only the pertinent model parameter values, moreover, if different semiconductor devices use the same model, the model parameters can be specified once only.

The SPICE-2 program execution consists of two basic phases. The first phase reads all the input data (i.e., the circuit description and parameters of required analyses), while the second phase performs all the simulations and prints the results. The consequence of such a program organization is that even a minor change in any of element descriptions or parameter values requires a new, independent run of the simulator. This makes optimization problems almost intractable because of inefficiency and complexity of interfacing optimization packages with the SPICE program, which can be done at the input/output file level only.

To overcome these difficulties a new structure of the circuit simulator is required, in which different analyses (for the same circuit) can be performed selectively, and in which there is an access to internal representation of circuit elements in order to update their values during optimization. The simulator should have the structure of a set (or a package) of subroutines rather than a program with one, fixed sequence of operations.

SPICE-PAC 2G6a.84.05 is a package of simulation subroutines obtained by redesigning the SPICE 2G.6 simulation program.

2. SPICE-PAC 2G6a

The SPICE-PAC version 2G6a provides:

- all the analyses available in the SPICE 2G programs,
- access to circuit variables as required in circuit optimization (circuit variables can be defined at the main circuit level as well as in subcircuits),
- dynamic declarations of output variables (output variables can be indicated at the main circuit level as well as in subcircuits),
- dynamic definitions of parameters for all analyses,
- parameterized subcircuit calls (subcircuit element definitions can be redefined by parameters included in subcircuit calls),
- an interface to libraries of standard modules (standard modules are in the form of subcircuits, and can be accessed by (parameterized) module calls).

SPICE-PAC does not provide the "main" program which must be suplied by the user to "drive" the subroutines, i.e., to call the subroutines which define parameters and perform analyses, as required by a particular application.

SPICE-PAC contains 25 main (or interfacing) subroutines:

SPICEA - initializes the package and reads circuit description,

SPICEB - defines circuit variables,

SPICEC - sets internal structures and performs initial processing,

SPICED - defines parameters for DC analysis,

SPICEE - defines execution-time limit,

SPICEF - defines frequencies for AC, NOISE and DISTORTION analyses,

SPICEG - defines parameters for DISTORTION analysis,

SPICEH - defines parameters for FOURIER analysis,

SPICEI - defines initial conditions (as node and/or device voltages,

SPICEJ - sets and resets internal flags,

SPICEK - defines parameters for DC TRANSFER FUNCTION analysis,

SPICEL - activates definitions of parameters and outputs,

SPICEM - defines the temperature for subsequent analyses,

SPICEN - defines parameters for NOISE analysis,

SPICEO - defines outputs for different analyses,

SPICEP - determines internal pointers for circuit element names,

SPICEQ - defines output variables,

SPICER - performs DC, TRANSIENT, AC, NOISE, DISTORTION, FOURIER and DC TRANSFER FUNCTION analyses, and sets OP-POINT data,

SPICES - performs DC SENSITIVITY analysis,

SPICET - defines parameters for TRANSIENT analysis,

SPICEU - updates circuit variables,

SPICEV - retrieves actual values of circuit variables,

SPICEW - retrieves SPICE-PAC execution times,

SPICEX - defines parameters and outputs using symbolic form,

SPICEY - retrieves the names of output variables, circuit elements and circuit variables.

A more detailed description of the subroutines is given in [5].

SPICE-PAC follows the SPICE 2G.6 program with the following exceptions:

- ".TEMPERATURE" lines, for SPICE-PAC, can define one temperature only (for subsequent definitions of the temperature the SPICEM subroutine should be used),
- ".DC" lines, for SPICE-PAC, can describe one set of the DC transfer curve source and sweep limits (for subsequent definitions of DC analysis parameters the SPICED subroutine should be used),
- ".ALTER" sections and
- ".PLOT" lines, in SPICE-PAC, are simply ignored.

It should be noted that all the control lines in circuit descriptions (".DC ...", ".AC ...", etc.) are used to define parameters of the corresponding analyses only, and the analyses are performed (selectively) by calling the SPICER and SPICES subroutines with appropriate arguments. Moreover, all the parameters defined by the control lines in the circuit description can be redefined by corresponding subroutines of the package (SPICED, SPICEF, etc.), or can be replaced by parameters "predefined" in an extended circuit description and activated (when required) by the SPICEL subroutine.

Extended circuit description is an optional part of the SPICE input file which can contain:

- definitions of circuit variables,
- definitions of parameters and outputs for different analyses,
- circuit element names which are converted into equivalent internal pointers as required by some SPICE-PAC subroutines (SPICER/OP-POINT, SPICES),
- definitions of monitoring and break-points.

All information provided by the extended circuit description can also be obtained by appropriate calls of SPICE-PAC subroutines. In most cases, however, extended circuit description can significantly simplify the use of SPICE-PAC, and, moreover, it allows the "main" program to be more general and more flexible since all the specific information can be placed in the data file rather than incorporated directly into the program.

Extended circuit description is separated from the (basic) circuit description by the line:

```
.END/EXT
```

and is terminated by the line:

```
.END
```

Circuit variables are defined by the lines:

```
.VAR  variable-name
```

where "variable-name" is either a simple element name for those elements which have one attribute only (usually it is the "value" of the circuit element), or a composite name which is used for multi-attribute circuit elements to indicate:

- polynomial coefficients of nonlinear conductors and inductors (e.g. C15'#3),
- polynomial coefficients of dependent voltage and current sources (e.g. E1'#0),
- DC and AC parameters of independent voltage and current sources (e.g. VIN'DC),
- parameters of time-dependent source functions of independent voltage and current sources (e.g., if an independent voltage source is described as "VIN 3 0 PULSE(-1,1,2NS,3NS,3NS)" then VIN'#4 denotes a circuit variable which corresponds to the "fall time" of VIN),
- parameters of semiconductor devices (e.g. Q1'AREA),
- parameters of (common) device models (e.g. MOD'RB),
- parameters of models associated with (particular) semiconductor devices (e.g. Q2:RB); in this case "common" model parameters are not influenced by changes of "individual" device model parameters.

Simple and composite variable-names can be direct or qualified. The direct names are used for those elements which are at the "top" (or "main") level of circuit description (i.e., elements not belonging to subcircuits). The subcircuit elements must be identified by the qualified names in which the element name follows the full sequence of the subcircuit names separated by periods "." (starting from the "top" level, e.g., X1.X3.X2.Q12:RE is a composite qualified variable-name denoting emitter resistance of the bipolar transistor Q12 in the subcircuit X2 of the subcircuit X3 of the subcircuit X1).

Definitions of parameters for SPICE-PAC analyses have the general form:

```
.PAR/id  analysis(parameters)
```

where "id" is an unsigned integer number that is used as an (unique) identifier of the definition, "analysis" is "DC", "TR", "AC", "NO", "DS", "FO", "TF" or "SE" for DC transfer curve, transient, small-signal AC, noise, distortion, Fourier, DC transfer function and DC sensitivity analyses, respectively, and "parameters" is a list of corresponding parameters separated by commas ",", e.g.:

```
DC(VIN,-5.0,5.0,21)
AC(LOG(11,1.D2,1.D4),1.D5,LOG(11,1.D6,1.D8))
NO(VIN,V(X2.4),2)
```

Definitions of outputs for different analyses have the general form:

```
.OUT/id analysis(output-list)
```

where "id" ana "analysis" are as before, and "output-list" is a sequence of output variables separated by commas ",".

Library modules correspond to standard SPICE subcircuits stored in individual files within a file system. They are referenced by (parameterized) subcircuit calls in which subcircuit names are the names of corresponding module-files preceded by the dollar sign "$". To handle hierarchically structured file references, module-file names indicated in subcircuit calls are concatenated with library path-names defined by a new directive:

```
.LIBRARY path-name
```

where "path-name" is either a sequence of directory names separated by "/" and then the new "path-name" redefines the previous one, or a sequence of directory names preceded by the "+" sign and then the previous "path-name" is extended by the new one, or a sequence of directory names preceded by one or more "-" signs, in which case the previous "path-name" is reduced by one directory name for each "-" sign, and then extended by the new "path-name" (which may be empty). For example, the sequence:

```
.LIBRARY /a/b/c/d
X1 ... $module1
.LIBRARY --x/y
X2 ... $module2
```

references the module files:

```
/a/b/c/d/module1
```

and

```
/a/b/x/y/module2
```

Two immediate examples of SPICE-PAC applications are circuit optimization and interactive circuit simulation.

3. INTERACTIVE SIMULATION

The general structure of an interactive circuit simulator is shown in Fig.1. An interactive driver mainly handles communication with the user, i.e., it enters user commands and displays (in numerical or graphical form) the results, while interfacing subroutines convert user commands (and parameters) into sequences of SPICE-PAC subroutine calls.

Because of initial preprocessing performed by SPICE-PAC, all circuit variables must be defined at the initial stage of circuit analysis. In interactive simulation it should be possible to modify values of arbitrary circuit

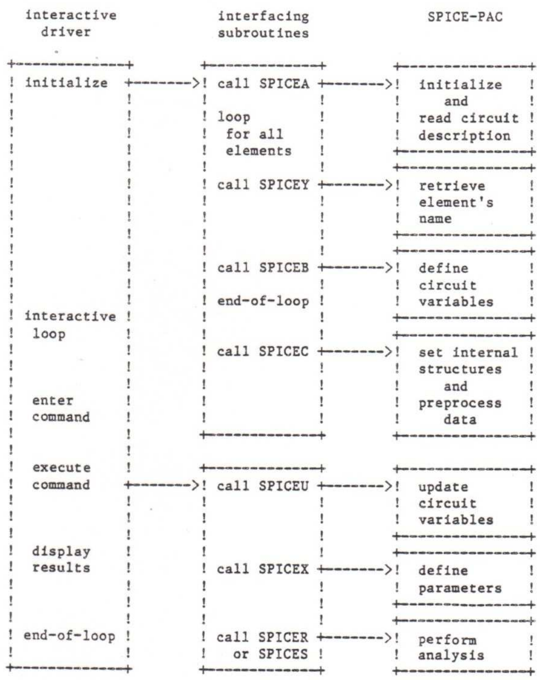

Fig.1. General structure of an interactive circuit simulator.

elements. To provide this flexibility, the initial phase of setting-up internal structures defines all circuit elements as circuit variables (using the subroutines SPICEY and SPICEB).

Fig.2 shows a family of frequency-response characteristics of a simple single-stage amplifier. Different curves correspond to different values of resistors and capacitors in the circuit, selected interactively to obtain the required gain and then the bandwidth of the amplifier. Similar plots can be obtained for all other SPICE-PAC analyses.

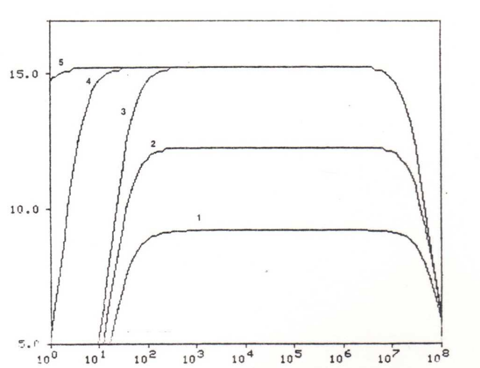

Fig.2. A family of frequency-response characteristics.

4. CIRCUIT OPTIMIZATION

The general structure of interfacing SPICE-PAC with an (abstract) optimization package OPTIM-PAC is sketched in Fig.3. MAIN and SUBR are user-supplied segments; MAIN initializes the packages and sets all the parameters while SUBR evaluates objective functions and constraints, as required by OPTIM-PAC, using SPICE-PAC subroutines for updating circuit variables and performing circuit analyses.

As an optimization example a single-stage CE amplifier in a self-biasing configuration is analysed, and it is to find the values of R1, R2 and RE such that for the midband frequency f=50kHz and for BETA.dc=80, 150, 250, the magnitude of the voltage gain is equal to 10 V/V, and the input resistance is not less than 10kohms.

The minimax optimization package WMBG2 used in this example is a modified version of linearly constrained minimax optimization technique due to Hald [2] combined with routines for numerical approximation of gradients [6].

In minimax formulation, there are three optimization variables R1, R2 and RE (additional circuit variable which corresponds to BETA.dc is used as an optimization parameter), and seven residual functions:

- the difference between 10K and the input resistance,
- the differences between the magnitude of the voltage gain and 10 V/V for beta.dc=80, 150, 250,
- the differences between 10 V/V and the magnitude of the voltage gain for beta.dc=80, 150, 250.

```
****  SPICE-PAC 2G6a.84.05   DATE : 18 MAY 84 10:27
****  INPUT LISTING          TEMP =    27.000 DEG C
**************************
* AMPLIFIER OPTIMIZATION *
**************************
VCC 5 0 12
VIN 1 0 AC 1
R1 2 5 350K
R2 2 0 40K
RC 4 5 5K
RE 3 0 400
CB 1 2 100UF
Q1 4 2 3 MOD
.MODEL MOD NPN(BF=150 VAF=50 IS=1.E-9 RB=100 CJC=1PF)
.PRINT AC V(4) V(2) I(VIN)
.AC 50K
.END/EXT
.VAR R1
.VAR R2
.VAR RE
.VAR Q1:BF
.END

OPTIMIZATION WITHOUT SCALING

PARAMETER :   Q1:BF

     8.00d+01 1.50d+02 2.50d+02

VARIABLES :

        R1       R2       RE
```

```
STARTING POINT :

     3.50d+05 4.00d+04 4.00d+02

LOWER AND UPPER BOUNDS :

     1.00d+04 5.00d+03 1.00d+02
     5.00d+05 1.00d+05 5.00d+02

ITERATIONS :

         R1       R2       RE     maxfun

  1  3.50d+05 4.00d+04 4.00d+02 2.31d+00
  2  3.50d+05 4.00d+04 4.00d+02 2.31d+00
  3  3.50d+05 4.00d+04 4.00d+02 2.31d+00
  4  3.50d+05 4.00d+04 4.00d+02 2.31d+00
  5  2.51d+05 5.09d+04 4.71d+02 1.91d+01
  6  3.26d+05 4.63d+04 4.64d+02 3.33d-01
  7  3.26d+05 4.74d+04 4.72d+02 1.80d-01
  8  3.25d+05 7.13d+04 4.78d+02 1.89d+01
  9  3.24d+05 5.21d+04 4.72d+02 1.96d-01
 10  3.18d+05 5.06d+04 4.72d+02 1.93d-01
 11  3.18d+05 5.06d+04 4.79d+02 1.66d-01
 12  3.16d+05 5.03d+04 4.74d+02 1.48d-01
 13  3.14d+05 5.02d+04 4.75d+02 1.25d-01
 14  3.13d+05 5.01d+04 4.75d+02 1.25d-01
 15  3.13d+05 5.02d+04 4.75d+02 1.24d-01
 16  3.32d+05 5.10d+04 4.74d+02 1.29d-01
 17  2.32d+05 4.67d+04 4.79d+02 1.91d+01
 18  3.26d+05 5.80d+04 4.77d+02 1.14d-01
 19  3.26d+05 5.80d+04 4.71d+02 1.48d-01
 20  3.25d+05 6.76d+04 4.78d+02 1.87d+01
 21  3.25d+05 6.04d+04 4.76d+02 1.63d+01
 22  3.25d+05 5.84d+04 4.76d+02 1.13d+00
 23  3.26d+05 5.84d+04 4.81d+02 1.96d-01
 24  3.25d+05 5.80d+04 4.76d+02 1.07d-01
 25  3.25d+05 5.82d+04 4.75d+02 9.71d-02
 26  3.25d+05 5.83d+04 4.75d+02 7.86d-02
 27  3.25d+05 5.83d+04 4.71d+02 6.13d+00
 28  3.25d+05 5.86d+04 4.75d+02 4.98d+00
 29  3.25d+05 5.84d+04 4.75d+02 7.48d-02
 30  3.25d+05 5.84d+04 4.75d+02 1.62d+00
 31  3.25d+05 5.85d+04 4.77d+02 1.09d-01
 32  3.25d+05 5.84d+04 4.75d+02 4.47d-01

SOLUTION :

     3.25d+05 5.84d+04 4.75d+02

     TYPE OF SOLUTION :         1
     NUMBER OF ITERATIONS :    22
     NUMBER OF SHIFTS :         1
```

The solution, obtained with 32 function evaluations and 22 iteration steps, corresponds to the maximum residual function which is less than 0.1 (line 29). Also, it can be observed that the solution is in rather narrow "valley" (lines 27, 28, 30) which is quite "flat" (lines 7, 9, 18). In many cases, a less accurate solution, obtained in just few iteration steps, should be satisfactory.

5. CONCLUDING REMARKS

Interactive simulation and circuit optimization are the most straightforward applications of the circuit simulation package which, in a very similar way, can be linked with many other packages, for example statistical modeling, yield analysis and design centering, as well as higher level language processors, circuit extractors, etc. Moreover, the modular structure of the package allows (at least potentially) to replace or enhance some of the subroutines, and introduce new interfaces to other packages, or alternative numerical algorithms, or redefine the existing device models. This also means that a package of circuit simulation subroutines can be easily used as one of the basic components of more advanced computer-aided-design systems.

6. REFERENCES

1. E. Cohen, "Program reference for SPICE 2"; University of California, Berkeley, Memo ERL-M592, 1976.
2. J. Hald, "MMLA1Q, a Fortran subroutine for linearly constrained minimax optimization"; Inst. for Numerical Analysis, Technical University of Denmark, Lyngby, Denmark, Report NI-81-01, 1981.
3. D.O. Pederson, "A historical review of circuit simulation"; IEEE Tr. on Circuits and Systems 1984(31)1, pp.103-111.
4. A. Vladimirescu, K. Zhang, A.R. Newton, D.O. Pederson, A.Sangiovanni- Vincentelli, "SPICE Version 2G - User's Guide (10 Aug. 1981)"; Department of Electrical Engineering and Computer Sciences, University of California, Berkeley CA 94720.
5. W.M. Zuberek, "SPICE-PAC 2G6a.84.05 - User's Guide"; Department of Computer Science, Memorial University of Newfoundland, St. John's, Canada A1C 5S7.
6. W.M. Zuberek, "Numerical approximation of gradients for circuit optimization", Proc. 27 Midwest Symp. on Circuits and Systems, Morgantown WV, 1984.