TELKOMNIKA, Vol.14, No.1, March 2016, pp. 64~71
ISSN: 1693-6930, accredited A by DIKTI, Decree No: 58/DIKTI/Kep/2013
DOI: 10.12928/TELKOMNIKA.v14i1.2832

SPICE Engine Analysis and Circuit Simulation Application Development

Bing Chen*[1], Gang Lu[2], Yuehai Wang[3]
[1]College of Electronic Engineering, Naval University of Engineering,WuHan,430033, China
[2]Equipment Department of the Navy, Beijing,100055, China
[3]School of electrical information engineering, North China University of Technology, Beijing, 100144, China
*Corresponding author, e-mail: chenbing0710@126.com

Abstract

Electrical design automation plays an important role in nowadays electronic industry. Various commercial Simulation Program with Integrated Circuit Emphasis (SPICE) packages, such as pSpice Or CAD, have become the standard computer program for electrical simulation, with numerous copies in use worldwide. The customized simulation software with copyright need the understanding and using of SPICE engine which was open-source shortly after its birth. The inner workings of SPICE, including algorithms, data structure and code structure of SPICE were analyzed, and a engine package and application development approach were proposed. The experiments verified its feasibility and accuracy.

Keywords*: SPICE engine analysis, Circuit Simulation Application, Simulation analysis*

1. Introduction

Knowledge of the behavior of electrical circuits requires the simultaneous solution of a number of equations.The model creation [1], circuit simulation [2] and testing [3], and application of analog circuit and VLSI [4] require the assisstance of simulation tools. As a general-purpose circuit-simulation program that solves the network equations for the node voltage for nonlinear DC, nonlinear transient, and linear AC analysis, SPICE in its different versions has been the main computer-aided analysis program used in analog design, and diagnosis for researchers, printing circuit board and electrical device manufacture for engineering and in universities and colleges for student education for over 40 years. This widely used spice-softwares, such as pSpice, hSpice, ngSpice, shares same engine which was re-developed or modified on the base of Berkeley's kernel and to provide various interfaces and functions.

However, this commercial software cannot be used to develop customized software free. In fact, it is well known that SPICE engine is open-source and free use by researchers all over the world. To make customized simulation software, researchers had various customer modification after analysis of the simulation algorithms [5],data structure and code structure. In 1980s, SPICE with version 2 had already rewrited in C code and ported in PC [6]. The 1990s saws various expansion of SPICE with multimedia technology [7] and networks [8, 9] with its application in engineer and education. In the later two decades, the simulation algorithms were continously modified to speed the convergence [10, 11] and improve the accuracy [12]. In the new centery, various virtual circuit labaroty [13, 14] and studies [15-17] had been build by the aid of these customized spice-like simulations. To the author's knowledge, there is little information aviable about the computation principle and its simulations of the SPICE engine. Further, the detail information about the re-development of the simulation software with spice engine and C++ is also helpful to programmers.

In this paper, we conducted this study to analyze the inner workings of SPICE engine and customized simulation software development based on the spice kernel. The experiments verified the feasibility and accuracy of the proposed approach.

Received October 12, 2014; Revised February 16, 2016; Accepted February 27, 2016

2. Inside of SPICE

2.1. Introduction of SPICE

SPICE stands for Simulation Program with Integrated Circuit Emphasis. It was born as a class project at the University of California, Berkeley during the mid-1970s and first released in 1971. After its first presented at a conference in 1973, this invaluable program soon evolved to universally used software and then to become the worldwide standard integrated circuit simulator in 1980s. Now SPICE is a general-purpose, open source analog electronic circuit simulator used in integrated circuit and board-level design to check the integrity of circuit designs, to predict circuit behavior, and even to locate the potential errors in circuit board.

2.2. SPICE Algorithm Overview

Figure 1 has given a simplified block diagram of the main SPICE program flow. We can see from Figure 1, when a circuit description was input to SPICE, its initial operating point was calculated first, and the linear companion models was then created for non-linear devices. Then SPICE entered its first kernel process--block 3 and 4, which stand for Nodal Analysis accomplished by formulating the Nodal Matrix and solving the nodal equations for the circuit voltages. SPICE finds the solution for Non-linear circuits in the inner loop (2-6). It may take many iterations before the calculations converge to a solution. The outer loop (7-9), together with the inner loop, performs a Transient Analysis creating equivalent linear models for energy-storage components for capacitors, inductors, etc. and selecting the best time points.

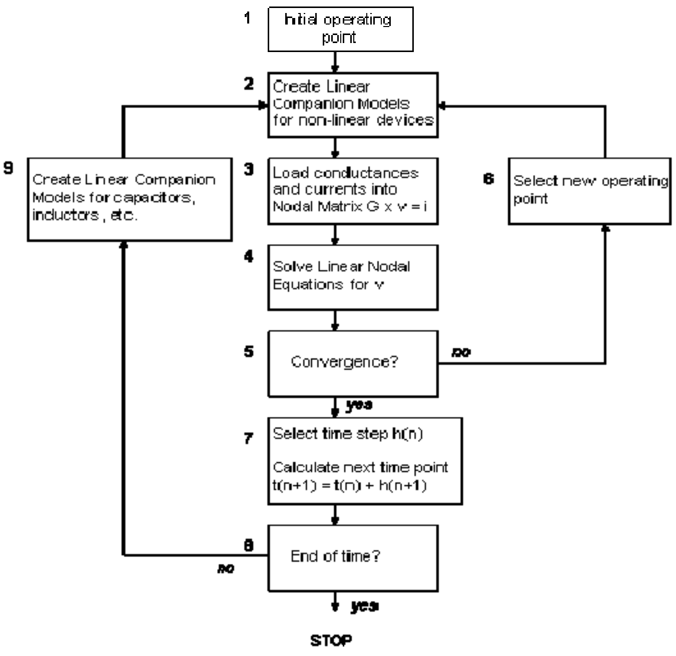

1
Initial operating point
2
Create Linear Companion Models for non-linear devices
9
Create Linear Companion Models for capacitors, inductors, etc.
3
Load conductances and currents into Nodal Matrix G x v = i
6
Select new operating point
4
Solve Linear Nodal Equations for v
5
Convergence?
no
yes
7
Select time step h(n)
Calculate next time point t(n+1) = t(n) + h(n+1)
8
End of time?
no
yes
STOP

Figure 1. The Flowchart of SPICE Algorithm

2.2.1. Nodal Analysis

At the core of the SPICE engine is a basic technique called Nodal Analysis that calculates the voltage at any node given all resistances (conductances) and current sources of the circuit. When a circuit was input, as shown in Figure 2, the SPICE engine calculate the the set of nodal equations according Kirchoff Law. These set of equations will be conveted to matrix form using mathmatical analysis and numeric computing. The frequently used method is Gaussian Elimination and LU Factorization. At last, the equation is solved for the node voltages and this equation is the central mechanism of the SPICE algorithm.

2.2.2. Non-Linear DC Analysis

SPICE can find the voltage at every node in a DC linear circuit by only executing mode 3 and 4. If any non-linear components appear in the circuit, SPICE will first transforming the non-linear components into some equivalence circuit suitable for Nodal Analysis, and then find a solution as if it was a linear circuit, but using a little guessing and a lot more trips. This process covers step 2-5, i.e. the inner loop.

As an example, the circuit shown in Figure 2 includes a very non-linear component, a diode.

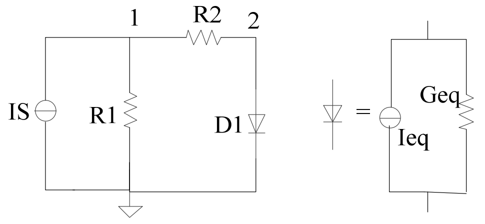


Figure 2. A Simple Non-linear Diode Circuit and part of its equivalent circuit

Diode's current versus voltage relationship is described by:

$$Id = IS(\exp(\frac{Vd}{Vt} - 1)) \quad (1)$$

Where *IS* is the saturation current and V_t is the thermal voltage. A diode can be modeled using linear components -- a parallel combination of a conductance G_{eq} and a current source I_{eq}, where G_{eq} is simply the slope of the tangentat the operating point V_{do}.

$$Geq = \frac{dId}{dVd} = \frac{IS}{Vt}\exp(\frac{Vdo}{Vt}) \quad (2)$$

And I_{eq} is the point where the tangent slices through the y-axis.

$$Ieq = Ido - Geq \quad (3)$$

After this equivalent substition, the nodal equations can be developed as:

$$\begin{pmatrix} \frac{1}{R1}+\frac{1}{R2} & -\frac{1}{R2} \\ -\frac{1}{R2} & \frac{1}{R2}+Geq \end{pmatrix}\begin{pmatrix} V1 \\ V2 \end{pmatrix} = \begin{pmatrix} IS \\ -Ieq \end{pmatrix} \quad (4)$$

The whole solution looks like this:
1) Guess the diode's initial trial operating point.
2) Create linear companion models.
3) Solve Nodal Equations.
4) Convergence?

$|V_n - V_{n-1}| < V_{limit}$ and $|I_n - I_{n-1}| < I_{limit}$?

5) No--Use the calculated diode voltage as new trial operating point for another iteration. Start again at Step 2.
6) Yes - Stop iterations and find the solution.
In SPICE, the limits of voltage and currentare actually calculated by[18].

$V_{limit} = V_n$ * RELLOT+VNTOL
$I_{limit} = I_n$ * RELLOT+ABSTOL

By default, RELTOL is set to 0.001 or 0.1 percent. So if the expected voltage is 5*V*, the V_{limit} should be set to 5 *mV* to reach a solution. However, if the excepted voltage swings near zero, V_{limit} would be ridiculous small and hard to be reached. That's where VNTOL enters the picture. The default of VNTOL is 1 μV to ensures that the limit doesn't get too small if voltages approach $0V$. RELTOL and ABSTOL play a similar role for the current change limits,and the default for ABSTOL is $1PA$.

2.2.3. Transient Analysis

SPICE executed the outer loop for the linear circuit transient analysis. If there is any energy-storage component, such as capacitor and inductor, SPICE first transformed the energy-storage components into their linear companion models, and used Nodal Analysis to find the solution. For example, for the capacitors, SPICE will use backward-Euler(BE) to predict the next approximate value at a future time point t_{n+1}. The idea of EU is predicting the next voltage with the slope at x_{n+1} as the following equation:

$$x_{n+1} \approx x_n + h\frac{dx_{n+1}}{dt} \tag{5}$$

Where $h = \Delta t = t(n+1) - t(n)$.

Graphically, it looks something like Figure 3.

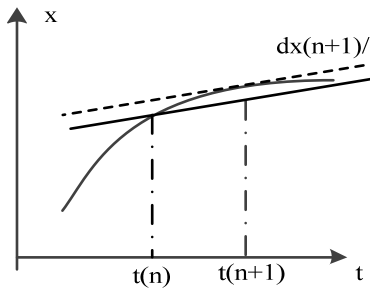


Figure 3. Numeric integration applied to approximate the next voltage

SPICE will first try to transform an energy-storage component into its equivalent linear component. For example, a capacitor is transformed using a two step process:

The relationships of a capacitor's voltage-current-charge can be described as:

$$I = C\frac{dV}{dt} \tag{6}$$

Next, apply the BE formula to predict the capacitor's voltage at the next time point.

$$V_{n+1} \approx V_n + h\frac{dV_{n+1}}{dt} \tag{7}$$

Finally, we can get an equation in terms of voltages and currents only:

$$I_{n+1} \approx \frac{C}{h}V_{n+1} + \frac{C}{h}V_n = G_{eq}V_{n+1} + G_{eq}V_n \tag{8}$$

At this time, we can transform a capacitor circuit shown as Figure 4 with its Linear companion model, where$I_{eq} = G_{eq}V_n, G_{eq} = \frac{C}{h}$.

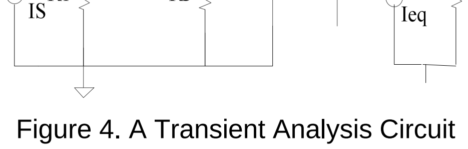


Figure 4. A Transient Analysis Circuit

With this equivalent substitution, SPICE can develop the nodal equations and find the every node voltages for certain time. Then, SPICE would forward a time step and repeat the same procedure to find the voltages at that time, and so on till the last time.

SPICE can set timestepdynamically to trade offsimulation accuracy and speed. If the change of current or voltage is rapid, SPICE would reduce its step to increase the precise of the simulation, and vice versa.

3. SPICE Code Strcture

The simplified block diagram of SPICE engine code structure can be shown in Figure 5, where seven Modules are included.

1) Command Parsing: The kernel of this block is comn structure, which includes the name of the commands and their corresponding functions. The program implemented an array of comn structure that defines the commands which SPICE can parse and its corresponding actions. Whenever a command was input, SPICE parses it and finds its match from the set of commands, and executes the corresponding action.
2) Circuit Description: The most important data structure of this block is CKT circuit structure, where contains the device type, simulation type, temperature, and node information, etc.
3) Sparse Matrix: This block defines SMP element structure to represent the non-zero items of sparse matrix where the items are stored with increasing order as they appear in the row or column of The matrix.
4) Numerical Computation: This block contains two typical numerical methods: Newton-Raphson iteration and numerical integration. In SPICE, they were used to transform the set of differential equations into a set of algebraic equations.

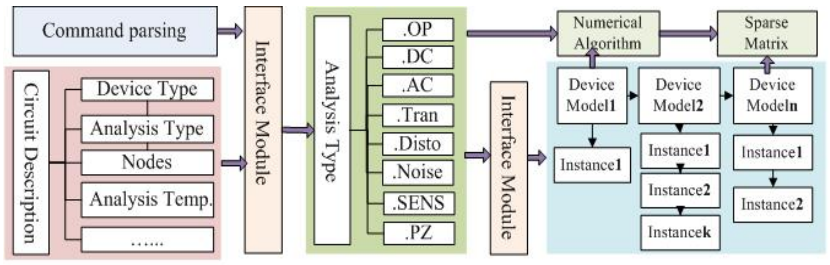


Figure 5. Code Structure of SPICE engine

6) Device Block contains two structures, DEV mode land SPICE dev.DEV model lists the models of the device types used in given circuit and SPICE dev contains the complete information of every component in the circuit.
7) Interface: this block is designed to be a generalized and open structure, and used to communicate between the Front and end of simulation system.

4. Application Interface of SPICE engine

SPICE engine can be package to a number of independent code procedures with prescribed interface. In this way, a block can call certain package to complete given work according to the interface. In addition, any block can be maintained, updated, even rewritten independently.

Basically, SPICE operates like this:

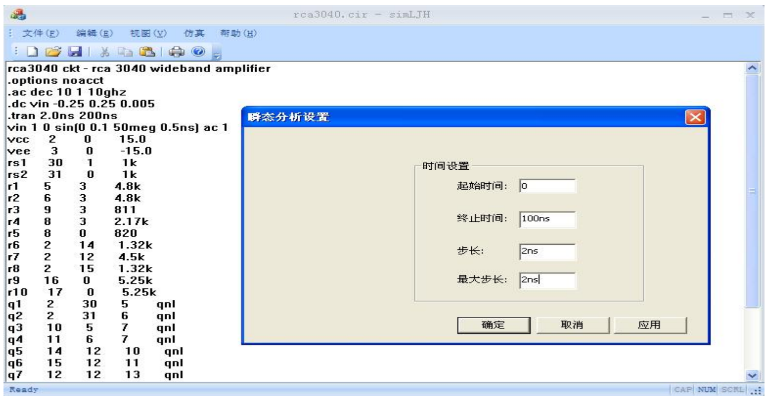
1) SPICE reads in a text circuit description file (".cir" extension) called a netlist, and then sets the corresponding Parameters.
2) SPICE sets the simulation options and performed certain analysis type as given (AC, Transient, DC, Noise, etc.) by calling the interface of analysis procedure.

Let's take transient (time) analysis for Linear Circuits as an example to investigate. SPICE would call fuctionDCTran first with the given circuit description. Inside this function, and device information would be found according to every device model. An operating point is required for the initial solution to a Transient Analysis. After the initial operating point was found, this function set the timestep and option parameter call the inner loop to solve the Nodal equation. The results is acquired from the sparse matrix and stored in the structure TRANan, where it can be print or draw by ".PRINT" or".PLOT" command.

5. Experiments of using SPICE Engine

SPICE engine can interactive with end user by command lines only. Therefore, it's convenient to encapsulate the engine to dynamic link libraries in the integrated development environment like visual studio 2010. These libraries were then package to be a engine class with exposing appropriate interfaces to the caller. For instance, the interface function for the transient analysis described previously may look like "BOOL tran (char* filename, int step, intend_t, intstart_t, intmaxstep)", where the input parameter can be the file name and path of circuit description, timestep, endtime, starttime (with the default value zero), and max timestep allowed. This function returns TURE when it completes the analysis and return FALSE when any error was found.

Then various applications about different circuit analysis and fault simulation can be developed with the SPICE engine that was packaged as a class. Here we build a single document MFC application that calls the SPICE engine. It reads a netlist file as input, and sets analysis type, simulation parameters by a dialogue. Then the application calls the engine to simulate the behavior of the inputting circuit. The simulation results were outputted as text file. Figure 6 shows the application running interface. It displays a operator amplified circuit in the window frame and reads user defined analysis parameters.



```
rca3040 ckt - rca 3040 wideband amplifier
.options noacct
.ac dec 10 1 10ghz
.dc vin -0.25 0.25 0.005
.tran 2.0ns 200ns
vin 1 0 sin(0 0.1 50meg 0.5ns) ac 1
vcc    2    0    15.0
vee    3    0    -15.0
rs1    30   1    1k
rs2    31   0    1k
r1     5    3    4.8k
r2     6    3    4.8k
r3     9    3    811
r4     8    3    2.17k
r5     8    0    820
r6     2    14   1.32k
r7     2    12   4.5k
r8     2    15   1.32k
r9     16   0    5.25k
r10    17   0    5.25k
q1     2    30   5    qnl
q2     2    31   6    qnl
q3     10   5    7    qnl
q4     11   6    7    qnl
q5     14   12   10   qnl
q6     15   12   11   qnl
q7     12   12   13   qnl
```

起始时间: 0
终止时间: 100ns
步长: 2ns
最大步长: 2ns

Figure 6. Simulation Application Example

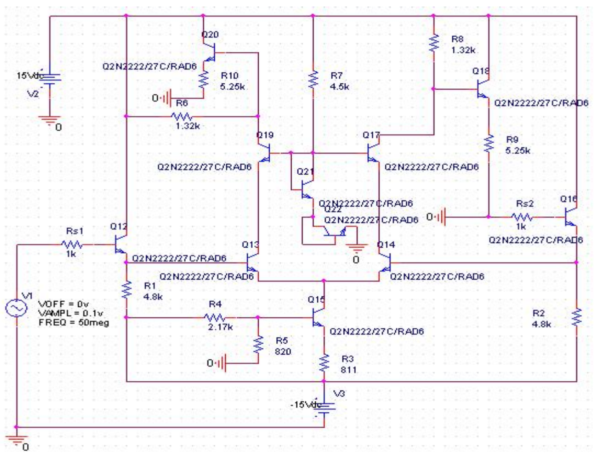


Figure 7. OP AMP circuit under testing

To verify the feasibility of this approach and investigate the accuracy of SPICE engine, the same circuit was build and analyzed with the same parameters and analysis type in OrCAD Capture, a famous business software. Figure 7 is schematic Of the testing circuit in orCAD Capture.

These two groups of simulation data were exported to Matlab to investigate the difference. Fig 8 is result given by Matlab where the solid line represent the results from OrCAD Capture, and the dot line represent the results from the testing application. It is clear that the results are in accord with each other in most case. There are also some difference due to the possible difference in device models, and maybe integration methods. The most significant difference lie to the maximumValue with only 2.13% difference.

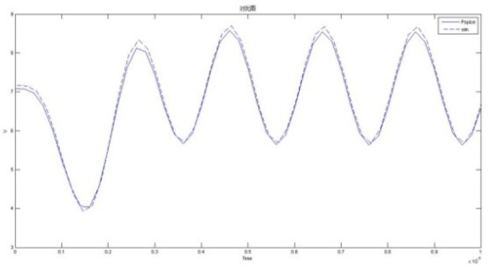


Figure 8. Experiment result

As the result, our approach of using SPICE engine turns out to be a suitable for further software development.

6. Conclusion

In this paper, the kernel algorithm and code structure of SPICE engine were analyzed, and an software architecture that package SPICE engine as class and called by other applications is proposed. The experiment verified the feasibility and accuracy of the proposed approach. This research can be used to development customized simulation software with fully simulation typies and all customered parameters aviable. Furthermore, it can easily used to develope the fault simulation by simply replace the function model of the devices as a fault-injected model.

Further works involves two parts. One is details of engine, such as its algorithm, its analysis, its parameter setting. Theother is result data processing, such as Fourier transform, wavelet analysis and featureextraction.

References

[1] Hamiane M, Ahlia U. A CMOSbased Analog Function Generator: HSPICE Modeling and Simulation. *International Journal of Electrical and Computer Engineering (IJECE)*. 2014; 4(4): 532-538.

[2] Xiaoyuan W, Zhe Y. Simulation of ZVS Converter and Analysis of Its Switching Loss Based on PSPICE. *International Journal of Power Electronics and Drive Systems (IJPEDS)*. 2012; 2(1): 19-24.

[3] Autee R. Dynamic Testing & Simulation of 4 KW 55 Hp Switched Reluctance Motor using PSpice. *International Journal of Power Electronics and Drive Systems (IJPEDS)*. 2011; 1(1): 36-40.

[4] Srivastava A, Palavali SR. *Integration of SPICE with TEK LV511 ASIC design verification system*. Proceedings of the 36th Midwest Symposium on Circuits and Systems, August 16, 1993 - August 18, 1993. Detroit, MI, USA. 1993.

[5] Nichols KG, Kazmierski TJ, Zwolinski M, Brown AD. *Overview of SPICE-like circuit simulation algorithms*. IEE Proceedings: Circuits, Devices and Systems. 1994; 141(4): 242-250.

[6] Zhao Y, Gao G. The Implementaton of SPICE2 with the Graphic Pre- and Post-Processors on PC. *Chinese Journal of Microelectronics and computer*. 1988; 09: 12-14.

[7] Conrad A. Interactive spice program organizes circuit designs. *Microwaves &amp RF*. 1995; 34(4): 183-184.

[8] Regnier J, Wilamowski B. SPICE simulation and analysis through Internet and intranet networks. *IEEE Circuits and Devices Magazine*. 1998; 14(3): 9-12.

[9] Wilamowski B, Regnier JR, Malinowski A. *SIP - Spice Intranet Package*. Proceedings of the 1998 International Symposium on Industrial Electronics, ISIE. Pretoria, South africa. 1998.

[10] Zuberek WM, Zuberek MS. *Table-driven circuit elements in SPICE-like simulation programs*. Twenty-Third Annual Asilomar Conference on Signals, Systems &amp; Computers. Pacific Grove, CA, USA. 1989.

[11] Zhou TY, Liu H, Zhou D, Tarim T. *A fast analog circuit analysis algorithm for design modification and verification*. SPECIAL SECTION ON THE 2010 INTERNATIONAL SYMPOSIUM ON PHYSICAL DESIGN. Piscataway, NJ. 2011.

[12] Zhang X, Liu G, Guo Y. A Circuit Optimization System Based on SPICE. *Chinese Journal of Hangzhou Institute of Electronic Engineering*. 2001; 03: 33-37.

[13] Mao S, Dong Y, Zhu M. Construction of the Virtual Experimental Platform of Electronic Circuits Based on Spice. *Chinese Journal of Computer Application and Software*. 2006; 23(03): 17-19.

[14] Yang Y, Zhao J, Wu W. Design of Circuit Simulation Software Based on Spice Secondary Development. *Chinese Journal of Computer Simulation*. 2007; 24(05): 268-323.

[15] Beams JD. *Adding SPICE to software development: A software development approach designed for rapidly changing environments*. Proceedings of the 1995 ACM Symposium on Applied Computing. Nashville, TN, USA. 1995.

[16] Zhuang H, He F, Lin X, Zhang L, Zhang J, Zhang X, et al. *A web-based platform for nanoscale non-classical device modeling and circuit performance simulation*. 2009 International Conference on Web Information Systems and Mining, WISM. Shanghai, China. 2009.

[17] Zhuang H, He J, Deng W, Zhang X, Zhu X, He Q, et al. *A web-based education platform for nanoscale device modeling and circuit simulation*. 4th Interdisciplinary Engineering Design Education Conference, IEDEC. Santa Clara, CA, United states. 2014.

[18] Nenzi P, Vogt H. Spice Users Manual. 2012.