HSPICE

For other uses, see Spice (disambiguation).

SPICE 1	
Original author(s)	Laurence Nagel
Initial release	1973; 47 years ago
Written in	Fortran
Type	Electronic circuit simulation
License	Public-domain software
Website	bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE/

SPICE 2	
Initial release	1975; 45 years ago
Stable release	2G.6 / 1983
Written in	Fortran
Type	Electronic circuit simulation
License	BSD 3 Clause
Website	bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE/

SPICE 3	
Original author(s)	Thomas Quarles
Initial release	1989; 31 years ago
Stable release	3f.5 / July 1993
Written in	C
Type	Electronic circuit simulation
License	BSD license
Website	bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE/

SPICE ("**Simulation Program with Integrated Circuit Emphasis**")[1][2] is a general-purpose, open-source analog electronic circuit simulator. It is a program used in integrated circuit and board-level design to check the integrity of circuit designs and to predict circuit behavior.

Contents

Introduction

Unlike board-level designs composed of discrete parts, it is not practical to breadboard integrated circuits before manufacture. Further, the high costs of photolithographic masks and other manufacturing prerequisites make it essential to design the circuit to be as close to perfect as possible before the integrated circuit is first built. Simulating the circuit with SPICE is the industry-standard way to verify circuit operation at the transistor level before committing to manufacturing an integrated circuit.

Board-level circuit designs can often be breadboarded for testing. Even with a breadboard, some circuit properties may not be accurate compared to the final printed wiring board, such as parasitic resistances and capacitances. These parasitic components can often be estimated more accurately using SPICE simulation. Also, designers may want more information about the circuit than is available from a single mock-up. For instance, circuit performance is affected by component manufacturing tolerances. In these cases it is common to use SPICE to perform Monte Carlo simulations of the effect of component variations on performance, a task which is impractical using calculations by hand for a circuit of any appreciable complexity.

Circuit simulation programs, of which SPICE and derivatives are the most prominent, take a text netlist describing the circuit elements (transistors, resistors, capacitors, etc.) and their connections, and translate[3] this description into equations to be solved. The general equations produced are nonlinear differential algebraic equations which are solved using implicit integration methods, Newton's method and sparse matrix techniques.

Origins

SPICE was developed at the Electronics Research Laboratory of the University of California, Berkeley by Laurence Nagel with direction from his research advisor, Prof. Donald Pederson. SPICE1 is largely a derivative of the CANCER program,[4] which Nagel had worked on under Prof. Ronald Rohrer. CANCER is an acronym for "Computer Analysis of Nonlinear Circuits, Excluding Radiation," a hint to Berkeley's liberalism in the 1960s:[5] at these times many circuit simulators were developed under contracts with the United States Department of Defense that required the capability to evaluate the radiation hardness of a circuit. When Nagel's original

advisor, Prof. Rohrer, left Berkeley, Prof. Pederson became his advisor. Pederson insisted that CANCER, a proprietary program, be rewritten enough that restrictions could be removed and the program could be put in the public domain.[6]

SPICE1 was first presented at a conference in 1973.[7] SPICE1 is coded in FORTRAN and uses nodal analysis to construct the circuit equations. Nodal analysis has limitations in representing inductors, floating voltage sources and the various forms of controlled sources. SPICE1 has relatively few circuit elements available and uses a fixed-timestep transient analysis. The real popularity of SPICE started with SPICE2[8] in 1975. SPICE2, also coded in FORTRAN, is a much-improved program with more circuit elements, variable timestep transient analysis using either the trapezoidal (second order Adams-Moulton method) or the Gear integration method (also known as BDF), equation formulation via modified nodal analysis[9] (avoiding the limitations of nodal analysis), and an innovative FORTRAN-based memory allocation system developed by another graduate student, Ellis Cohen. The last FORTRAN version of SPICE is 2G.6 in 1983. SPICE3[10] was developed by Thomas Quarles (with A. Richard Newton as advisor) in 1989. It is written in C, uses the same netlist syntax, and added X Window System plotting.

As an early public domain software program with source code available,[11] SPICE was widely distributed and used. Its ubiquity became such that "to SPICE a circuit" remains synonymous with circuit simulation.[12] SPICE source code was from the beginning distributed by UC Berkeley for a nominal charge (to cover the cost of magnetic tape). The license originally included distribution restrictions for countries not considered friendly to the US, but the source code is currently covered by the BSD license.

The birth of SPICE was named an IEEE Milestone in 2011; the entry mentions that SPICE "evolved to become the worldwide standard integrated circuit simulator."[13] Nagel was awarded the *2019 IEEE Donald O. Pederson Award in Solid-State Circuits* for the development of SPICE.[14]

Successors

Open-source successors

No newer versions of Berkeley SPICE have been released after version 3f.5 in 1993.[15] Since then, the open-source or academic continuations of SPICE include: XSPICE,[16] developed at Georgia Tech, which added mixed analog/digital "code models" for behavioral simulation, CIDER[17] (previously CODECS), developed by UC Berkeley and Oregon State Univ., which added semiconductor device simulation, SPICE OPUS,[18][19] developed and maintained by the University of Ljubljana is based on SPICE 3f.4 and on XSPICE, ngspice, based on SPICE 3f.5, XSPICE and CIDER.[20][21]

Commercial versions and spinoffs

Berkeley SPICE inspired and served as a basis for many other circuit simulation programs, in academia, in industry, and in commercial products. The first commercial version of SPICE is

ISPICE,[22] an interactive version on a timeshare service, National CSS. The most prominent commercial versions of SPICE include HSPICE (originally commercialized by Ashawna and Kim Hailey of Meta Software, but now owned by Synopsys) and PSPICE (now owned by Cadence Design Systems). The integrated circuit industry adopted SPICE quickly, and until commercial versions became well developed many IC design houses had proprietary versions of SPICE.[23]

Today a few IC manufacturers, typically the larger companies, have groups continuing to develop SPICE-based circuit simulation programs. Among these are ADICE at Analog Devices, LTspice at Analog Devices (available to the public as freeware), Mica at Freescale Semiconductor, and TINA-TI[24] at Texas Instruments. Both LTspice and TINA-TI come bundled with models from their respective company.[25][26] Analog Devices offers a similar free tool called ADIsimPE (based on the SIMetrix/SIMPLIS[27] implementation of SPICE).[28] Other companies maintain internal circuit simulators which are not directly based upon SPICE, among them PowerSpice at IBM, TITAN at Infineon Technologies, Lynx at Intel Corporation, and Pstar at NXP Semiconductor.[*citation needed*]

Program features and structure

SPICE became popular because it contained the analyses and models needed to design integrated circuits of the time, and was robust enough and fast enough to be practical to use.[29] Precursors to SPICE often had a single purpose: The BIAS[30] program, for example, did simulation of bipolar transistor circuit operating points; the SLIC[31] program did only small-signal analyses. SPICE combined operating point solutions, transient analysis, and various small-signal analyses with the circuit elements and device models needed to successfully simulate many circuits.

Analyses

SPICE2 includes these analyses:

- AC analysis (linear small-signal frequency domain analysis)
- DC analysis (nonlinear quiescent point calculation)
- DC transfer curve analysis (a sequence of nonlinear operating points calculated while sweeping an input voltage or current, or a circuit parameter)
- Noise analysis (a small signal analysis done using an adjoint matrix technique which sums uncorrelated noise currents at a chosen output point)
- Transfer function analysis (a small-signal input/output gain and impedance calculation)
- Transient analysis (time-domain large-signal solution of nonlinear differential algebraic equations)

Since SPICE is generally used to model nonlinear circuits, the small signal analyses are necessarily preceded by a quiescent point calculation at which the circuit is linearized. SPICE2 also contains code for other small-signal analyses: sensitivity analysis, pole-zero analysis, and small-signal distortion analysis. Analysis at various temperatures is done by automatically updating semiconductor model parameters for temperature, allowing the circuit to be simulated at temperature extremes.

Other circuit simulators have since added many analyses beyond those in SPICE2 to address changing industry requirements. Parametric sweeps were added to analyze circuit performance with changing manufacturing tolerances or operating conditions. Loop gain and stability calculations were added for analog circuits. Harmonic balance or time-domain steady state analyses were added for RF and switched-capacitor circuit design. However, a public-domain circuit simulator containing the modern analyses and features needed to become a successor in popularity to SPICE has not yet emerged.[29]

It is very important to use appropriate analyses with carefully chosen parameters. For example, application of linear analysis to nonlinear circuits should be justified separately. Also, application of transient analysis with default simulation parameters can lead to qualitatively wrong conclusions on circuit dynamics.[32]

Device models

SPICE2 includes many semiconductor device compact models: three levels of MOSFET model, a combined Ebers–Moll and Gummel–Poon bipolar model, a JFET model, and a model for a junction diode. In addition, it had many other elements: resistors, capacitors, inductors (including coupling), independent voltage and current sources, ideal transmission lines, active components and voltage and current controlled sources.

SPICE3 added more sophisticated MOSFET models, which were required due to advances in semiconductor technology. In particular, the BSIM family of models were added, which were also developed at UC Berkeley.

Commercial and industrial SPICE simulators have added many other device models as technology advanced and earlier models became inadequate. To attempt standardization of these models so that a set of model parameters may be used in different simulators, an industry working group was formed, the Compact Model Council,[33] to choose, maintain and promote the use of standard models. The standard models today include BSIM3, BSIM4, BSIMSOI, PSP, HICUM, and MEXTRAM.

Exclusion for integrated photonic circuits

Traditional photonic device simulators apply direct methods to solve Maxwell's equations for the complete structure, whereas photonic circuit simulators are based on a segmentation into building blocks (BBs), each of which is represented at a logic level by a phothonic device, "coupled to other BBs by guided modes of optical waveguides". At the circuit-level modeling, a photonic integrated circuit (PIC) contain both electrical wires and optical signals, respectively described by voltage/current and by complex-valued envelope for the forward- and backward-propagating modes.[34]

The building block netlist of both the photonic and electronic circuits, including their net and port connections, can be expressed in a SPICE format with some schematic editors, like the ones used for electronic design automation.[35]

To reproduce the complete photonic signal information, without losing eventual optical phenomena, it is needed the real-time waveform of both the electric and the magnetic field for every mode or polarization in the waveguide.[*clarification needed*] While SPICE works with 10^{-15} time steps, timescale datacommunications of ≈10–100 10^{-12} are common. To make the amount of information tractable, the modulation increases of complexity, having to encode both amplitude and phase, in a way similar as in the simulation of RF circuits.[36]

However, Photonic Integrated Circuit simulators need to test multiple communication channels in match with different Carrier frequencies, or equivalently more amplitudes in any single channel, a type of sophisticated signal that is unsupported on the SPICE program features and structure as described above.[34] At 2019, SPICE can't be used to "simulate photonics and electronics together in a photonic circuit simulator",[37] and thus it isn't yet considered as a test simulator for photonic integrated circuits.

Input and output: Netlists, schematic capture and plotting

SPICE2 takes a text netlist as input and produces line-printer listings as output, which fits with the computing environment in 1975. These listings are either columns of numbers corresponding to calculated outputs (typically voltages or currents), or line-printer character "plots". SPICE3 retaines the netlist for circuit description, but allows analyses to be controlled from a command-line interface similar to the C shell. SPICE3 also added basic X plotting, as UNIX and engineering workstations became common.

Vendors and various free software projects have added schematic capture front-ends to SPICE, allowing a schematic diagram of the circuit to be drawn and the netlist to be automatically generated. Also, graphical user interfaces were added for selecting the simulations to be done and manipulating the voltage and current output vectors. In addition, very capable graphing utilities have been added to see waveforms and graphs of parametric dependencies. Several free versions of these extended programs are available, some as introductory limited packages, and some without restrictions.

Transient analysis

Since transient analysis is dependent on time, it uses different analysis algorithms, control options with different convergence-related issues and different initialization parameters than DC analysis. However, since a transient analysis first performs a DC operating point analysis (unless the UIC option is specified in the .TRAN statement), most of the DC analysis algorithms, control options, and initialization and convergence issues apply to transient analysis.

Initial conditions for transient analysis

Some circuits, such as oscillators or circuits with feedback, do not have stable operating point solutions. For these circuits, either the feedback loop must be broken so that a DC operating point can be calculated or the initial conditions must be provided in the simulation input. The DC operating point analysis is bypassed if the UIC parameter is included in the .TRAN statement. If UIC is included in the .TRAN statement, a transient analysis is started using node voltages

specified in an .IC statement. If a node is set to 5 V in a .IC statement, the value at that node for the first time point (time 0) is 5 V.

You can use the .OP statement to store an estimate of the DC operating point during a transient analysis.

```
.TRAN 1ns 100ns UIC .OP 20ns
```

The .TRAN statement UIC parameter in the above example bypasses the initial DC operating point analysis. The .OP statement calculates transient operating point at t = 20 ns during the transient analysis.

Although a transient analysis might provide a convergent DC solution, the transient analysis itself can still fail to converge. In a transient analysis, the error message "internal timestep too small" indicates that the circuit failed to converge. The convergence failure might be due to stated initial conditions that are not close enough to the actual DC operating point values.

See also

-  Electronics portal
-  Free and open-source software portal

- Comparison of EDA Software
- List of free electronics circuit simulators
- Input Output Buffer Information Specification (IBIS)
- Transistor models