Intusoft Newsletter

Personal Computer Circuit & System Design Tools

intusoft

Issue #71 Aug 2003
Tel. (310) 329-3295
Fax (310) 329-9864

SPICE 1 2 3 and beyond...

In This Issue

SPICE (***S****imulation* ***P****rogram with* ***I****ntegrated* ***C****ircuit* ***E****mphasis*) became the industry standard for integrated circuit simulation with the release of version 2 from U.C. Berkeley in 1975. It was quickly adopted for all types of electronic circuit simulation including RF and Power.

However, the foundations for SPICE began long before with Gustav Kirchhoff in 1845. Building on the work of Ohm and Newton, Kirchhoff formulated two laws used to describe an electrical circuit using algebraic equations. Kirchhoff's Current Law (KCL), forms the basis for electronic circuit simulation [1]. Meanwhile, working in parallel, Carl Friedrich Gauss was developing many of the mathematical principals [2] that provided methods to solve Kirchhoff's equations.

Moving forward to the origins of digital computing, Alan Turing in 1948 invented the LU decomposition method used today for solving an equation matrix. As digital computers became ubiquitous in the 60's, the quest for a mechanized analysis intensified. Using the FORTRAN language and IBM mainframes, businesses lead by IBM began producing software programs that attacked various parts of the problem. By the 1970's, the University of California at Berkeley became a major contributing force starting with its CANCER [3] simulator. Then in 1972, the first SPICE simulator became available.

There are several reasons for the success of SPICE; first it was placed in the public domain. More importantly, the performance of each of the key numerical algorithms was compared with various rivaling techniques. Ultimately, SPICE was an amalgam of the best technology. It's longevity lies in its centuries long scientific heritage. But a new discipline was emerging, Software Engineering.

Fuelled by the new C programming language, the awkward FORTRAN code used in SPICE was replaced and in 1985 SPICE3 was released in C. At the same time, the SPICE2 code made its way into the new personal computer. SPICE3 was not very stable and it took a number of years to flush out its major bugs. No sooner was the Berkeley project finished, than Bjarne Stroustrup's C++ language entered the picture. After several years, the C++ language became the building block for the Microsoft Foundation Class (MFC) library. C++ is based on the Simula language that was developed for simulation modeling. It's as though the C++ language and MFC were designed to be used in a circuit simulator.

On the other side of the country; the Georgia Institute of Technology was adding an event driven engine into SPICE3. The combined kernels ultimately became XSPICE. Event driven simulation allows both digital and analog circuits to run efficiently together. It turns out that the XSPICE addition can also be used for ordinary SPICE models. Intusoft makes that capability available to users with its Code Model Software Development Kit (CMSDK).

By 1994, Intusoft had made a stable XSPICE version using the C programming language. Then in 1997, Intusoft's version 8 software using MFC was released. Since then, both Georgia Tech. and U.C. Berkeley have stopped active research in their respective versions of SPICE. A feature new to SPICE3 was a built-in scripting language. As originally shipped from Berkeley, the script language was very unstable but rich in promise. Intusoft has gone on to fix the original Berkeley software and has extended its capability. See the article on page 7 for a script language tutorial.

Intusoft has gone on to include new primitive models using the CMSDK and also by direct implementation in the SPICE3 code. Intusoft is able to respond rapidly to incorporate new primitive models into its IsSpice kernel, such as the BSIM4, EKV and MESFET, because most researchers use the SPICE3 kernel. Here are some of the key features added to IsSpice over time, beginning with the SPICE3 release:

Predictor/Corrector step: Added in SPICE3, it predicts the state vector at the next time step and initializes the solution vector to that value. For most cases, a single iteration is all that's needed at each time step.

Gmin Stepping: Added in SPICE3, it doesn't actually step Gmin, but instead adds a large value in the main diagonal of the matrix and progressively reduces the displacement in order to steer the solution to a stable value. Intusoft corrected a defect in the implementation by accounting for the sign of the values in the admittance matrix so that there wouldn't be an abrupt sign change on the last step.

Bypass: Bypasses computation of certain model parameters if the model state hasn't changed very much. Feature didn't work properly until SPICE3.

VSECTOL: A new option for time step control added by Intusoft. Similar to CHGTOL but uses the product of voltage and time instead of current and time. It's extremely important for modeling circuit-switching events.

AUTOTOL: A new option added by Intusoft to make separate VNTOL and ABSTOL entries for each node voltage or branch current; it runs automatically during the operating point calculation.

New Primitive Models in SPICE3

Behavioral Element: A major breakthrough in SPICE3 to provide user algebraic equations. Intusoft added the if-then-else capability along with Boolean expressions and made TIME, FREQ and TEMP from the simulator available as variables

BSIM3, 4: New MOSFET models consistent with shorter channels in modern IC fabrication.

Hysteresis Switch: The switch becomes a primitive instead of a subcircuit.

Lossy Transmission Line: Added in SPICE3 to include skin effect.

RC Transmission line: Added in SPICE3 for short IC interconnects.

New Primitive Models in IsSpice

MESFET: Added by Intusoft for GaAs parts.

EKV MOSFET: Added by Intusoft in 2001, used for improved convergence with similar accuracy as the BSIM models.

Vardelay: An Intusoft CMSDK model for a variable delay element working in AC, DC and TRAN simulations.

Model Revisions

MOS3: Intusoft corrected a coding error that results in an unstable limit cycle in the transient simulation as Vds approaches zero.

ASRC: Intusoft added current and power vector measurement.

R,L,C: Intusoft added Behavioral (ASRC) model equations.

Some XSPICE code models

Magnetic Core
Differentiator
SOI MOSFET
Hysteresis Block Reluctance Model
Inductive Coupling into the reluctance model
Limiter

Controlled One-Shot
Table Models
Laplace (s-Domain) Transfer Function
Slew Rate Block
Controlled Sine Wave Oscillator
Controlled Square Wave Oscillator
Controlled Triangle Wave Oscillator
Smooth Transition Switch
Repeating Piece-Wise Linear Source
Controlled Digital Oscillator
Z-Transform Block (Real)

Simulation Control

Voltage Source PWL: added a repeat mode.

Simulation Templates: Added capability for complex scripted analysis using the Spice3 Interactive Control Language, ICL. These templates currently include Monte Carlo, Worst Case, Extreme Value, Optimization, RSS and implement the General Feedback Theorem.

[1] The second law, Kirchhoff's Voltage Law, KVL, is rarely used in computer simulation because the KVL equation set is not unique for a given circuit while there is only one correct KCL formulation.
[2] Unbeknownst to Gauss, some of his work was actually developed centuries earlier in China.
[3] CANCER: Computer Analysis of Non-Linear Circuits Excluding Radiation.

Simple Oscillator Comparison

Figure 1 shows a simple oscillator using SPICE3 technology as augmented by Intusoft in IsSpice. The circuit is a behavioral level model of a timer commonly used by IC designers. This circuit was first described by Christophe Basso, EDN, 8/16/2001. Basso also showed how to make PSpice do the same thing by adding several behavioral expressions to model the hysteresis as shown in Figure 2. This circuit is useful in studying the capability of various simulators because it's relatively easy to compute the oscillator frequency and calculate simulation errors.

For charging:

$$Vs = Vm\left(1 - e^{\frac{-t}{RC}}\right)$$ where Vm = Vcc-.1, Vs=Vmax-.1

and for discharging:

$$Vs = Vxe^{\frac{-t}{R_d C}}$$ where Vx = 2.5 and Vs = .1

Then:

C=1n
Rd=50
R=10k

Tcharge = -R*C*ln(1-2.4/4.9) = 6.72944e-006

Tdischarge = -Rd*C*ln(.1/2.5) = 1.60944e-007

f=1/(Tcharge+Tdischarge) = 145.13k

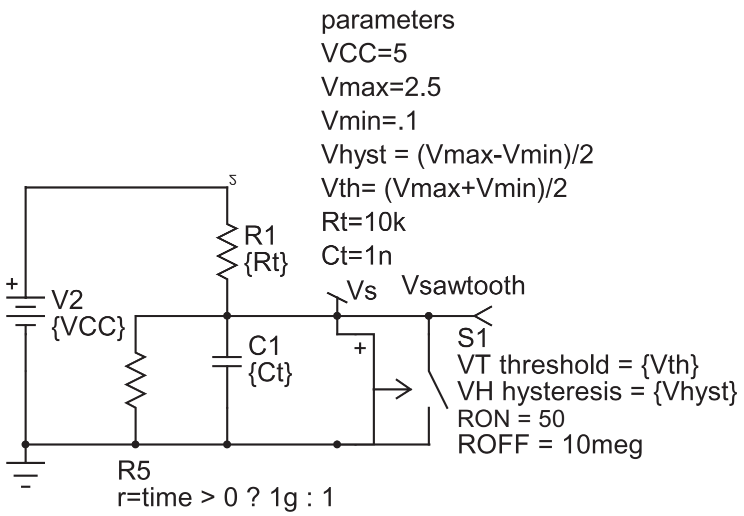

Figure 1. The IsSpice schematic uses a time varying R5 to establish the DC operating point, thus eliminating the need for making V2 a pulsed source.

This circuit illustrates a fundamental problem with SPICE2 and SPICE3 time step control. It's based on controlling integration error as estimated by the Local Truncation Error, LTE. However when a switching event occurs, the time step stays the same as it was prior to the event. Now the previous time step will be at its largest when S1 turns on and the resulting time step will be far too large to get a decent estimate of the C1 discharge time. The SPICE2/3 solution is to limit the maximum time step to achieve the desired accuracy. For this example you need to set the time step near 10n, which produces 100k-point waveforms for 1msec of simulation time. Moreover, the circuit has two stable states, making calculation of a DC operating point impossible. The DCOP problem can be solved by making the input voltage, V2, a pulse, or by using UIC (Use Initial Conditions) to describe the transient simulation and setting IC=0 for C1. Setting UIC is extremely undesirable because you then take responsibility for the initial condition of all parts in your circuit. That includes subcircuit models provided by vendors for which you have little or no control, or understanding. The pulse solution is probably better but may not represent correct operation in all cases. By using the IsSpice enhancements, resistor R5 can be switched from low impedance to high impedance at the beginning of the simulation. The advanced VSECTOL option in IsSpice is used to retain the variable time step capability of SPICE2/3. It does this by scaling back the time step when the product of the time step and change in the next predicted voltage at node Vs exceeds the value specified by VSECTOL. Now you still need a Tmax specification when many oscillation cycles occur so that the time step doesn't violate the Nyquist criteria.

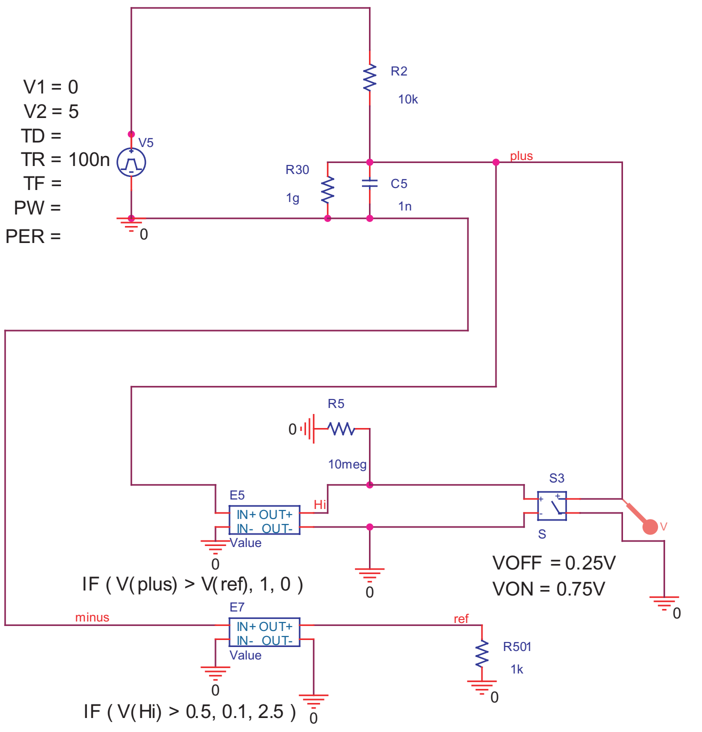

Figure 2. The PSpice schematic requires additional behavioral elements, E5 and E7 to introduce hysteresis. Christophe Basso, EDN, 8/16/2001.

Table 1 shows how the simulators stack up. Simulation time is reduced 4 fold in IsSpice and waveform size is reduced to less than 10%. What that means is that long simulations with many switching cycles can be performed quickly, allowing you to explore a greater variety of circuit options to perfect your design.

Table 1: How simulator options effect accuracy and speed using a 266MegHz Pentium II with NT-4 for all cases.

Options		Error		Simulation Time (sec)		Data Points	
Tmax	**VSECTOL**	**IsSpice**	**PSpice**	**IsSpice**	**PSpice**	**IsSpice**	**PSpice**
10n	-	.005%	0.18%	31.0	76.3	100.008k	201.336k
200n	1n	.087%	[a]	8.0	[a]	7.779k	[a]

[a] The VSECTOL option is not available using PSpice
note: Using built-in time step options result in unacceptable performance (the wrong waveform) for both simulators.

ICL Scripting

So far we have only looked at the core simulation engine. But there's really a lot more involved in getting data in and out of the simulator. Analog circuits are well described graphically, either by a schematic or a block diagram [5]. In ICAP/4, the schematic system (SpiceNet) controls the simulation [6]; freeing the user from dealing with netlists, thus removing a significant source of data input error. Further, the XSPICE event driven circuits are connected seamlessly via node bridges to the analog circuitry when the netlist is made.

What really distinguishes ICAP/4 from other simulators is the way the ICAP/4's SPICE3 scripting has been used to unify the schematic database, the IsSpice simulator and the IntuScope waveform viewer. Scripts were introduced with SPICE3 to allow interactive simulation and waveform viewing. Intusoft has expanded the role of scripts to embed measurement scripts and their results in the Schematic. The script language is common to all 3 applications; we call it the Interactive Control Language, ICL.

ICL Scripts consist of Commands, Functions and Operations. The ICL, like BASIC or JAVA, is an interpreter. There are two distinguishing features that make the ICL unique to SPICE3. First, there are no type declarations. If you enter y = 2, then y is automatically cast as a double. If you then take y = y * (1,2), where (1,2) is a complex number, then y is recast to the complex type. If you went on to y = y * vout, where vout is a real vector, then y becomes a complex vector. The second difference between the ICL and BASIC is the large number of commands that control the simulator and the waveform viewer. All of the SPICE3 "dot" commands can be run as scripts. You can even place a script in the input "deck" enclosed by ".control" and ".endc". Both Spice4.exe and Scope5.exe have an OLE [7] automation interface. The key automation interface is called DoScript, which uses a string argument that is the script to be executed. You can run scripts in either IsSpice or IntuScope and send scripts from the schematic to the waveform viewer by right clicking on script text within the schematic. There are certain functions that don't run in both environments; for example, the SPICE3 "dot" commands, like TRAN or OP, will only run from IsSpice, while plot related commands, like "plot" or "makesmithplot," only run from IntuScope.

To learn about the ICL, open IntuScope and make sure that both the "Commands" and "Output Record" windows are visible. If not, select them in the "Window" menu to turn them on. Next, select "Script Syntax" in the help menu. That brings up an html page that describes the ICL. If you use Netscape or Mozilla, you will need to switch to the browser page using <Alt+tab> or by using the Windows toolbar.

Press <Ctrl+R> to execute a script that is typed into the "Commands" window. You can enter complex numbers using (a,b) or a + j(b) where a is the real part and b is the imaginary part. Enter the following into the "Commands" window and press <Ctrl+R>.

Commands	**Output Record**
y = 2 * volt	y = 2.000000e+000, 6.000000e+000 volts
y = y * (1+j(3))	x = 3.00000 amperes
x = 3 * ampere	y*x = 6.000000e+000, 1.800000e+001 watts
y	y+x = 5.000000e+000, 6.000000e+000 unknown
x	
y*x	
y+x	

Notice that units were attached by multiplying the primitive unit definition by a real number. When scope encounters a line that can't otherwise be evaluated, it assumes it's a print command. Just entering a variable and pressing <Ctrl+R> will let you see the value in the "Output Record" window. Units are evaluated and simplified for each expression so that units for y * x become watts and y + x is unknown. This dimensional analysis capability can be used to debug complex equations, showing errors if units evaluate to unknown. You can view the primitive dimensions by executing the following script:

```
load rule.plt
display
```

Rule.plt is handled in a special way so that the dimensions of its variables are not simplified; for example, evaluate “watt” and you get “watt = 1.00000 volt amperes.” You can go ahead and add new units and definitions to rule.plt and save them. Rule.plt will be loaded into the constants plot when Scope5.exe starts up. IsSpice doesn’t do units calculations at this time; it’s planned for a future update.

When IsSpice runs, it saves results in plots. The plots are collections of simulation vectors and the result of ICL evaluations. IsSpice automatically names plots; for example the first transient simulation plot will be called TRAN1. If the ICL runs another simulation, then its results will be placed in a plot named TRAN2, and so forth. You can use the ICL to rename a plot; for example, calling the result of the first simulation ref, etc.

IntuScope connects to IsSpice using a set of filters you can find in the plugin folder. A filter can be selected using the “Browse…” button in the “Add Waveforms” window.

Plug in	Description	Availability
SpiceIn.dll	Gets data using the COM Interface to Spice4.exe	All versions
OutFileIn.dll	Reads data from the “.out” file.	All versions
CSDFin.dll	Reads CSDF formatted data from a “.csd” file.	Not available using the Demo version
TextIn.dll	Reads data from a text file.	Not available using the Demo version
TouchStoneIn.dll	Reads touchstone formatted data for smith charts	Not available using the Demo version

If you are interested in writing your own filters; for example, to get data from an oscilloscope, contact Intusoft to obtain a software development kit.

You can access the desired plot in IntuScope using the “Type” dropdown control in the “Add Waveform” dialog or “Plots” dropdown control in the IsSpice “Simulation Control” Dialog. These selections are synchronized between the programs. The display command in IsSpice will list all of the simulation data available for each plot. In IntuScope, only the vectors that have been created using IntuScope are shown using the display command. IntuScope can create vectors from the Source list in the Y-Axis and X-Axis lists provided by the selected plug in source. Each IntuScope drawing document can have multiple plots.

All SPICE programs use a variable time step. When you add vectors from different simulations or different filters, the length of the vectors will in general not be the same. Other waveform viewers, like PSpice's Probe, simply won't let you evaluate functions or perform operations involving data with different vector lengths. Intusoft has elected to automatically interpolate data between mixed data sources. The x-axis then becomes the union of the two x-axis data sets. Each y-axis value is then interpolated into the new vector space. That allows you to display the results of a second simulation in the same plot using the "Actions" menu and selecting "Add Updated Traces" or pressing <Ctrl+U>. You can then take the difference, measure the RMS error, etc. You can also use the waveform numbers as aliases or use the built-in previous and current aliases

```
error = rms(w1-w2)
print error
```

or use after <Ctrl + U> for a single waveform

```
rms(current – prior)
```

Now, when a mathematical operation is performed between waveform vectors, the SPICE3 ICL parser will perform the operation for each element in the vector; unlike a BASIC program that must be made to iterate over each element in the vector array---resulting in dramatically faster operation. Of course you can access individual array elements, like Vout[n]; however, with variable time steps, the index n isn't very useful. You can extract a portion of a vector without having to iterate over the elements in the vector by using the "[[" operator.

```
Vmid = w1[[500u,600u]]
```

This function extracts data based on the scale vector range (time in this case) and makes a new vector out of the data between 500u and 600u.

You can make an accompanying time axis for a new plot: t56 = time[[(500u,600u)]], and then plot vmid vs t56. Of course you can subtract the offset from t56 using the following script: t56 = t56 - 500u.

Vectors can be constructed and plotted without connecting to any source data. For example, the following script plots a constant power load and a resistive load-line:

```
vmax = 150*volt
Rs = .5*ohm
imax = vmax/Rs
P = 1000*watt
v=vector(1000)/1000*vmax+vmax/1000
* constant power load
i=P/v
plot i v
setxlimits 0,vmax
setylimits 0,imax
```

```
* resistive load line
ir = (vmax-v)/Rs
plot ir
setylimits 0,imax
```

This script illustrates several important features of the ICL. First, the vector function was used to create an x-axis that goes from vmax/1000 to vmax. Second, the expressions for i and ir used the vector v, causing the automatic type casting of the two currents into vectors. Third, the "plot" command caused the traces to be displayed and set the default or x-axis vector to v.

The ICL commands include if-then-else control structures and various iteration schemes. There is no scope control operator (like {...} in the C language); the control statements terminate with an "end" command.

Data is stored as vectors in multiple plots. Measurements are also stored in the plots, however, most measurements are scalars (vector length = 1). Intusoft has added the concept of an alias to represent a vector or a plot. Then, using nextplot and nextvector you can loop through all of the vectors in each plot without knowing what measurements they represent. The following script fragment is taken from the Monte Carlo analysis. The sameplot tests are used to skip plots where we are storing data and the length test eliminates everything except scalar measurements. This fragment builds a new vector in the prob plot where each element comes from a simulation measurement of a particular case.

```
pl = nextplot(null)
while pl <> null
    if sameplot(ref.default) = 0
        if sameplot(constants.default) = 0
            if sameplot(prob.default) = 0
                printstatus -l pl
                nv = nextvector(null)
                while nv <> null
                    if length(nv) = 1
                        prob.nv[constants.plotnum] = nv
                    end
                    nv = nextvector(nv)
                end
                constants.plotnum = constants.plotnum + 1
            end
        end
    end
    pl = nextplot(pl)
end
```

These operations along with a few netlist control directives form the basis of Intusoft's Simulation Templates. These templates are accessed from the SpiceNet simulation control dialog. So far there are 10 of these templates. Most of them work for AC, DC and TRAN simulations.

[5] Digital circuits are best described by their bus or register transfer architecture. Combining analog and digital technologies is accomplished by introducing the notion of a bus in the schematic diagram to eliminate the clutter of wires. Spice itself has no analog/digital preference. It gives accurate results for both analog and digital circuits. The XSPICE extension reduces the accuracy burden for digital circuits.
[6] Many "high end" schematic packages were derived from UNIX at a time when the software vendors made their own graphical user interface (GUI). Windows, on the other hand, has a built-in GUI. The Windows operating system performs many of the operations these other packages had to place in their own GUI. Then when "high end" packages port to Windows there is confusion over where data is stored, and whether to access it using the Windows Operating System or from the application. This usually results in a program management layer that duplicates much of the Windows OS. ICAP/4 is built with several component programs that must interoperate. Control is maintained using the schematic and an invisible control program called icaps.exe. The function of the control program is to add pre-processing functions to build the IsSpice netlist; provide error communications; and to start, terminate and switch between the component programs.
[7] Microsoft keeps changing the name; it started as OLE, then became ActiveX and now its generally referred to as COM.

Modeling Tips

The power company's dilemma! What happens if you make a constant power load? With some finite line impedance, there are two stable solutions. The load line shown in Figure 3 illustrates the problem.

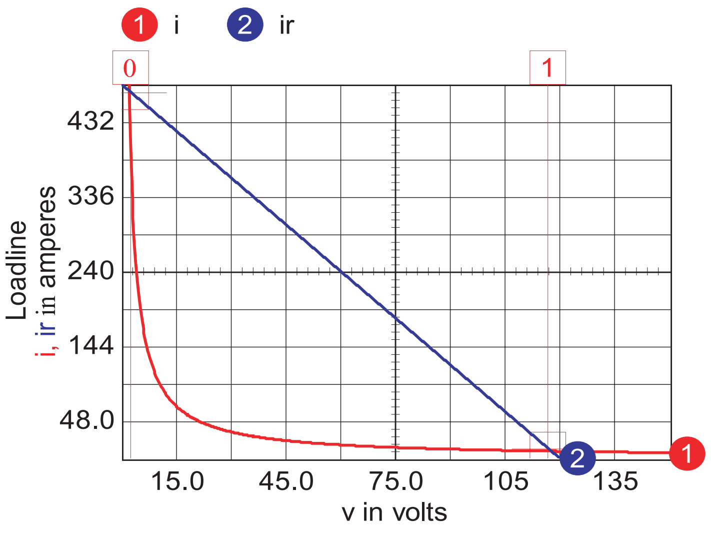

Figure 3. Constant power loads can have two stable solutions.

A very poor solution occurs at a low load voltage when a tremendous loss occurs delivering the power. But that's exactly what happens if you model power loss in IsSpice using a B element current source having the following equation:

```
I=P/Vin
```

You can get around the problem by arranging for the current at low voltage to go to zero before its intercepted by the load line. A convenient way to do that is by using the following equations in IntuScope.

```
vn = vmax/10
Pmax = 1000*watt
v=vector(1000)/1000*vmax+vmax/1000
Ploss = Pmax*(1-exp(-v/vn))
```

Figure 4 shows the new load model, which is accurate for higher voltage and eliminates the low voltage solution. To accomplish this in IsSpice, you need to add a B-element to calculate Ploss, then use Ploss to find the current, I=Ploss/Vin.

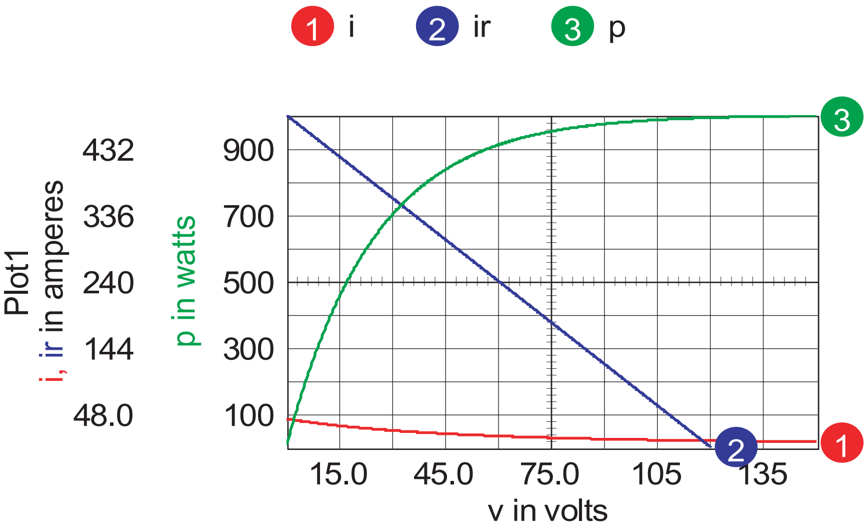

Figure 4. Making load power approach 0 at low voltage eliminates the undesired solution.

Using ICAP/4 to Run in a Client Server Environment

Intusoft sells an ICAP/4 license to run remotely as a network-based server. The purpose of this package is to perform simulations of existing circuits with some minor variations in order to illustrate the behavior of a third-party product. Frequently the server will be accessed using the Internet. For this technique to be responsive to the user, the circuit must simulated in several seconds and the data must be passed back in a short time. If you made the circuit using SpiceNet, then the ".cir" file would have been generated by the schematic. You can identify the portions of the circuit that can be changed using .param statements; for example,

```
.param Rload=20.
```

Then, the .param lines can be moved into a file called param.txt and replace in the ".cir" file with a single *include param.txt line. Then you need to connect an html form to alter the values in the param.txt file and make a cgi script for the server to run the simulation.

The syntax to make Spice4.exe and its preprocessor (icaps.exe) run without interactive graphics is:

```
\spice8\icaps –runspice –nographics sample.cir
```

Where \spice8 is the folder where you installed the ICAP/4 software and sample.cir is the name of the input "deck." The result will be in the ".out" file or a CSDF file if you choose that option. Then you must collect the data from those files and send back the appropriate result to the user. For an example go to http://www.intusoft.com/support.htm.

More New Features

In the last newsletter we mentioned new features to expect with our next release in mid September. Here is a list of even more features that we have recently added to this coming release.

1)Added to SpiceNet the ability to rotate or flip multiple parts as a "group select" about their collective center was added to SpiceNet.

2)Previously with IntuScope it was possible to select "Link X Axes" in the Scaling menu only when all displayed plots had identical x-axes. Now "Link X Axis" is available as long as the x axes of all plots have the same name. If their scale factors or log types are different, all are set to match those of the active plot.

3)IntuScope now provides the ability to perform an update using the currently selected analysis type, rather than the original analysis type. You can change between these two modes by selecting either "Use Original Analysis Type" or "Use Selected Analysis Type" in the update menu. You can have the update happen during mode change by checking "Update on Mode Change."

4)Holding the shift key while cross-probing produces the waveform on the schematic instead of in IntuScope, while not holding the Shift key will view waveforms directly in IntuScope. Also SpiceNet will automatically launch Scope5 in the background and place a trace in it if you try to cross-probe to it.

5)When in cross-probe mode you can now move the cross-probe cursor over the desired area to probe, press the right mouse button and then choose from the popup menu what type of waveform you want to plot to the schematic or IntuScope. All future cross-probe will now be in this cross-probe mode until you change it.

6)SpiceNet will automatically change the cross-probe cursor to a normal cursor while it is passing over a cross-probe waveform plot that has been placed on the schematic. This enables easy moving, deleting, and resizing of cross-probe plots without having to manually change cursors between operations.

7)SpiceNet has been modified to prevent saving a file with a name longer than 43 characters (including the .DWG extension) or a path longer than 240 characters (not counting the file name). Also, a warning is displayed when opening a file that exceeds the limits.

8)Added to IntuScope the option to “Include Cursors” in print outs.

9)Added PARAMDIGITS option that allows you to set the number of significant digits that will be printed on the netlist for a passed parameter/value. This option is only active in ICAPS parameter passing, and the default is 2 for backward compatibility.

10)Modified IntuScope to significantly reduce its requirement for memory when working with very large waveforms.

PCB from Bartels

Intusoft’s relationship with Bartels is in full swing as an intensive effort to fully integrate ICAP/4 with Bartels’ PCB layout and route system progresses. The first release is targeted for December of this year. A lower-end version will be provided at no charge for ICAP/4 users, with a modest upgrade cost for more functionality. Intusoft Development is completing the XML database (see NL 70) and working on the ICAP/4 interface to Bartels. More to come in the next newsletter.