# Intusoft Newsletter

Personal Computer Circuit & System Design Tools

intusoft

Copyright © Intusoft, All Rights Reserved

Issue #80 Sept. 2007
Tel. (310) 329-3295
Fax (310) 329-9864

## New ICAP/4 8.x.11 Build 3090 Release

Intusoft's latest 8.x.11 Build 3090 for the ICAP/4 and Test Designer products is mature and feature-rich. In this release numerous improvements have been implemented that further the software's ease of use, notably in the more advance applications. Two popular new features are: (1) the ability to export waveforms in a tab-delimited file that can easily be used in other programs such as Microsoft excel, and (2) a Pspice-to-IsSpice4 model converter. Further, Build 3090 is compatible with Windows Vista and 64-bit Windows operating systems.

### In This Issue

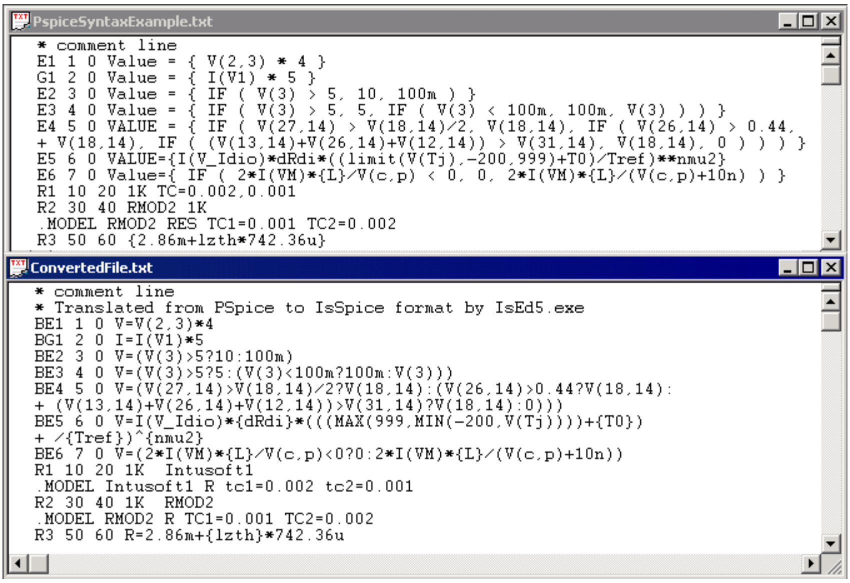
A detailed features list for Build 3090 can be found at: http://www.intusoft.com/featurechanges.htm.

ICAP/4Windows and higher offerings have always been attractive due to valuable features like interactive parameter modification, and advance multi-run analysis like Monte Carlo, Sensitivity, Optimize, EVA, WCS, RSS, GFT, nested parametric sweeping and fault simulation. Build 3090 adds to only these packages the new Pspice-to-IsSpice4 Converter, and an improved automated part-database management system (Library Manager). If you are an existing ICAP/4Rx or Consumer user and frequently add or modify parts from different sources, then you'll want to upgrade to a ICAP/4Windows or higher offering. You can upgrade at anytime by only paying the price difference. Please read the rest of the newsletter to decide if it is time for you to upgrade to a higher offering and the latest Build 3090. You won't be disappointed.

## Pspice-to-IsSpice4 Converter

Many SPICE models are available in Pspice format. The majority of the analog syntax for both IsSpice4 and Pspice is exactly the same, but there are a few small differences that would make the Pspice model inoperable in IsSpice4 until it was translated to proper syntax. A manual conversion could be done, but this would take time and is error prone.

Intusoft has created a Pspice-to-IsSpice4 Converter that instantly changes the analog Pspice netlist into a format used by IsSpice4. Simply open the Pspice netlist file in IsEd, or paste it in a blank IsEd window. From the edit menu select "Convert Pspice to IsSpice" or press the <Ctrl>+I hotkey. Another window will come to view with the netlist converted to IsSpice4 syntax. Note that the converter works on analog parts only. No Pspice digital parts will be translated.


PspiceSyntaxExample.txt

```
* comment line
E1 1 0 Value = { V(2,3) * 4 }
G1 2 0 Value = { I(V1) * 5 }
E2 3 0 Value = { IF ( V(3) > 5, 10, 100m ) }
E3 4 0 Value = { IF ( V(3) > 5, 5, IF ( V(3) < 100m, 100m, V(3) ) ) }
E4 5 0 VALUE = { IF ( V(27,14) > V(18,14)/2, V(18,14), IF ( V(26,14) > 0.44,
+ V(18,14), IF ( (V(13,14)+V(26,14)+V(12,14)) > V(31,14), V(18,14), 0 ) ) ) }
E5 6 0 VALUE={I(V_Idio)*dRdi*((limit(V(Tj),-200,999)+T0)/Tref)**nmu2}
E6 7 0 Value={ IF ( 2*I(VM)*{L}/V(c,p) < 0, 0, 2*I(VM)*{L}/(V(c,p)+10n) ) }
R1 10 20 1K TC=0.002,0.001
R2 30 40 RMOD2 1K
.MODEL RMOD2 RES TC1=0.001 TC2=0.002
R3 50 60 {2.86m+lzth*742.36u}
```

ConvertedFile.txt

```
* comment line
* Translated from PSpice to IsSpice format by IsEd5.exe
BE1 1 0 V=V(2,3)*4
BG1 2 0 I=I(V1)*5
BE2 3 0 V=(V(3)>5?10:100m)
BE3 4 0 V=(V(3)>5?5:(V(3)<100m?100m:V(3)))
BE4 5 0 V=(V(27,14)>V(18,14)/2?V(18,14):(V(26,14)>0.44?V(18,14):
+ (V(13,14)+V(26,14)+V(12,14))>V(31,14)?V(18,14):0)))
BE5 6 0 V=I(V_Idio)*{dRdi}*(((MAX(999,MIN(-200,V(Tj))))+{T0})
+ /{Tref})^{nmu2}
BE6 7 0 V=(2*I(VM)*{L}/V(c,p)<0?0:2*I(VM)*{L}/(V(c,p)+10n))
R1 10 20 1K  Intusoft1
.MODEL Intusoft1 R tc1=0.002 tc2=0.001
R2 30 40 1K  RMOD2
.MODEL RMOD2 R TC1=0.001 TC2=0.002
R3 50 60 R=2.86m+{lzth}*742.36u
```

**Figure 1:** Example of the type of conversion that the Pspice-to-IsSpice4 Converter performs.

Everything in the selected file will be converted at once. You can also have a file with multiple models. To be compatible with SPICE model library files (.LIB) and simulation netlist files (.CIR), the first line of the file is always ignored and treated as a comment line. If you want a quick one-line conversions then remember that the first line will not be evaluated for conversion.

| Function | Pspice syntax | IsSpice syntax |
|---|---|---|
| Common logarithm (base 10) | LOG10(X) | LOG(X) |
| Natural logarithm (base e) | LOG(X) | LN(X) |
| Exponentiation | X**Y | X^Y |
| Boolean XOR | A^B | (({A\|B})&(~A\|~B)) |
| LIMIT (x,min,max) | LIMIT (A,B,C) | (MAX(C,MIN(B,A))) |
| If Exp is true then T else F | IF(Exp,T,F) | (Exp?T:F) |

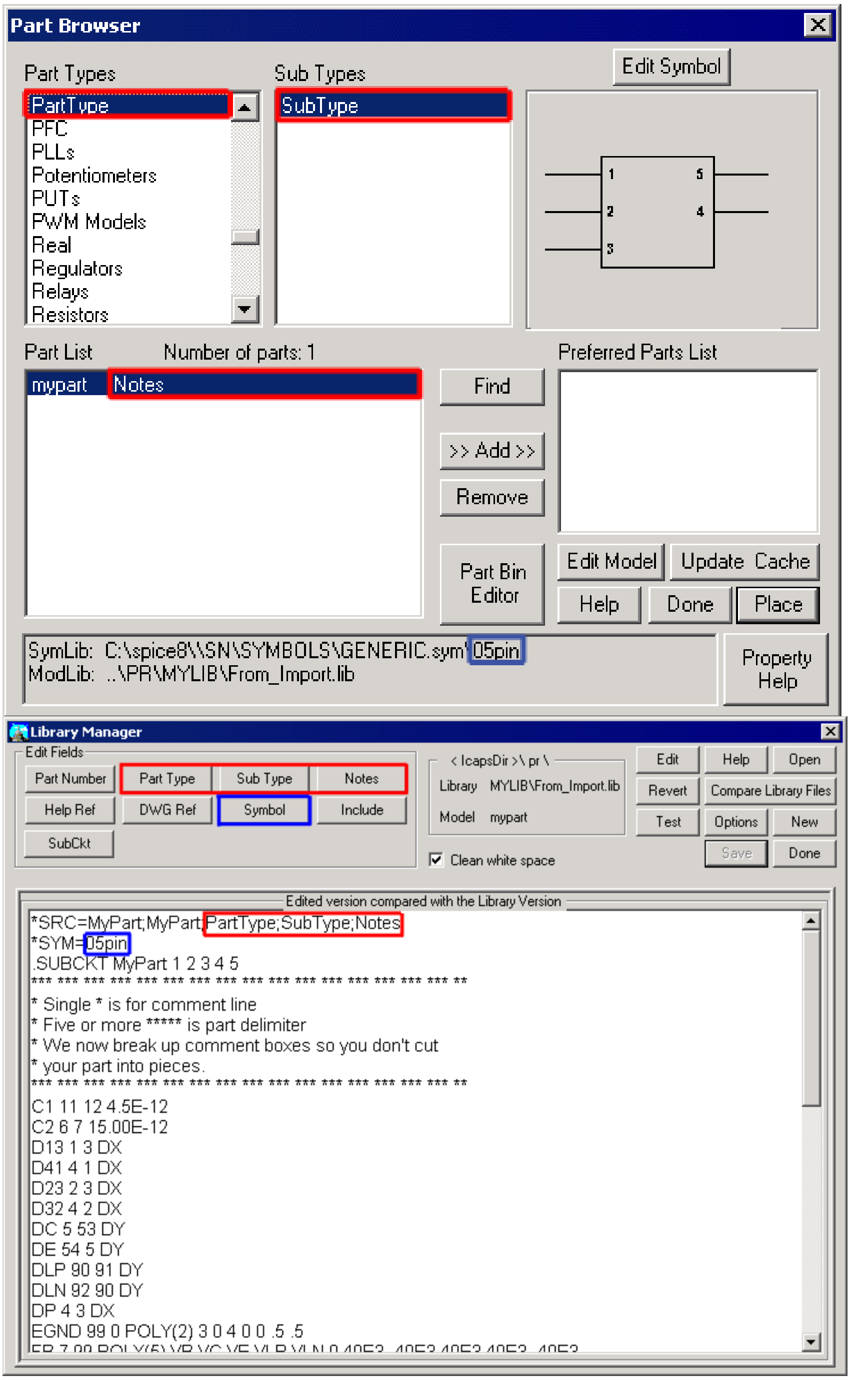
**Figure 2:** The things to watch out for regarding expressions is that IsSpice4 uses ^ for exponentiation, while Pspice uses ** for exponentiation and ^ for XOR. For natural log, Pspice uses log(x) and IsSpice4 uses ln(x).

## Improved Import SPICE Model

If you have a SPICE part in IsSpice4 compatible format then you can easily import the part and add it to the part browser. Highlight the individual part you want to import and copy it to your clipboard. In SpiceNet's File menu select "Import Spice Model" and press the "From clipboard" button. If the model name on the .subckt line does NOT already exist in the part database, the New Model dialog will come to view. The unique model name will be shown in the bottom right field under "Enter the new model name." By default the "From_Import.lib" file will be selected, but you can specify your own library file. Note that all user added parts are placed in the <ICAPSdir>\pr\mylib folder to keep user parts separate from Intusoft parts.

If the part already exists in the part database, Library Manager is immediately brought to view with a character by character diff of an existing part and imported part. If you don't want the existing part in the part database replaced with your imported model, then exit Library Manager without saving changes, and re-import the part again with a slightly different name on the .subckt line.

Every fourth asterisk in a row has been automatically replaced with a space. This prevents the problem of imported parts with comment boxes made up of asterisks from being split into pieces. If a line starts with an asterisk and has 5 asterisks in a row, we know that the beginning or end of the part has been reached. Even if you only have one part in a library, you need a top and bottom row of asterisks part delimiter.



SymLib: C:\spice8\\SN\SYMBOLS\GENERIC.sym\05pin
ModLib: ..\PR\MYLIB\From_Import.lib

Library MYLIB\From_Import.lib
Model mypart

```
*SRC=MyPart;MyPart;PartType;SubType;Notes
*SYM=05pin
.SUBCKT MyPart 1 2 3 4 5
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** **
* Single * is for comment line
* Five or more ***** is part delimiter
* We now break up comment boxes so you don't cut
* your part into pieces.
*** *** *** *** *** *** *** *** *** *** *** *** *** *** *** *** **
C1 11 12 4.5E-12
C2 6 7 15.00E-12
D13 1 3 DX
D41 4 1 DX
D23 2 3 DX
D32 4 2 DX
DC 5 53 DY
DE 54 5 DY
DLP 90 91 DY
DLN 92 90 DY
DP 4 3 DX
EGND 99 0 POLY(2) 3 0 4 0 0 .5 .5
```

**Figure 3:** Use Library Manager to make the imported part show up in the correct part browser category, and with the correct symbol.

If you bring up the hierarchical Graphical Parts Browser you will find the recently imported part under PartType then SubType. You will want to move your part so it is under one of the existing categories. Use the Edit Fields buttons at the top left corner of the Library Manager dialog to change the text shown in the part browser. If the text is the same as an existing category then it will be placed in that category. Further, instead of using a generic box symbol, a more appropriate existing symbol can be used. First select a similar existing part in the part browser and look at the symbol name shown at the end of the SymLib: line, located at the bottom of the part browser. Then in the Library Manager click the “symbol” button and type the existing symbol name.

Changing the *SRC= or *SYM= line will flag Library Manager to ask if want to perform an “update part database” when you try to close the dialog. This is necessary so your changes will ultimately be found in the part browser. If Library Manager launches Makedb, it will automatically close the part browser and re-open it with the new updated part database after it is done compiling.

If the vendor who provided this part revises it, you can then re-import the part. Your modified *SRC= and *SYM= lines will not be changed. You will see a character by character diff and know exactly what was changed between revisions.

## Improved Export Schematic as SubCkt

Prepare a schematic circuit configuration with only the circuitry that you want to include in the subcircuit. This means you must create a special configuration with the test circuitry and stimulus sources removed. You can use wires, test points or continuation symbols on the nodes that are unconnected. Unconnected nodes are shown as red diamonds. All unconnected parts must have their pins resolved with node numbers or names shown.

If you want to pass parameters into your subcircuit then you need to copy your passed parameters to the parameters dialog for the configuration that you plan to export. These passed parameters will be added to the .subckt line for you.

Select Make Subdrawing from the Subdrawings drop-down menu. The Subdrawing will be given a name that is the same name as the current configuration. You may change it in the Subdrawing Name: field if you want your part to have a different name. Click on the node from the list of nodes in the dialog that you want to expose from inside of the subcircuit (external connection that will be on the .SUBCKT line). Click the Add button to add the selected node to the list of subcircuit pins. Repeat this operation for each node you want to expose.

You may arrange the node order using the Move Up/Move Down buttons. You can also assign the pins to be hidden. You will then be able to make connections to the hidden pins by using a continuation symbol with the same name as the hidden pin. The hidden pin name is assigned inside of the Parts Properties dialog.

If you use the symbol wizard to create your own symbol then remember to save your symbol before you click on the Finish Subdrawing button at the bottom left of the symbol editor. Also don't add the symbol to an existing symbol that Intusoft provides or it will be overwritten when you update the software. We have created a special folder to contain user-generated symbols: (C:\spice8\sn\mysym). If you save your symbol to ..\sn\mysym\mysym.sym then you won't have to do anything. If you save the symbol to a different symbol file then modify the c:\spice8\sn\sym.@@@ file to include the path to your new symbol file.

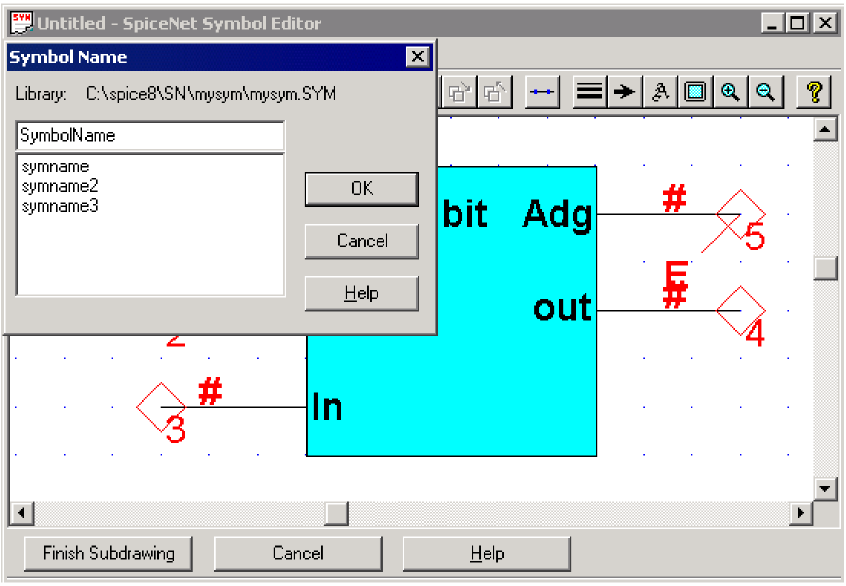


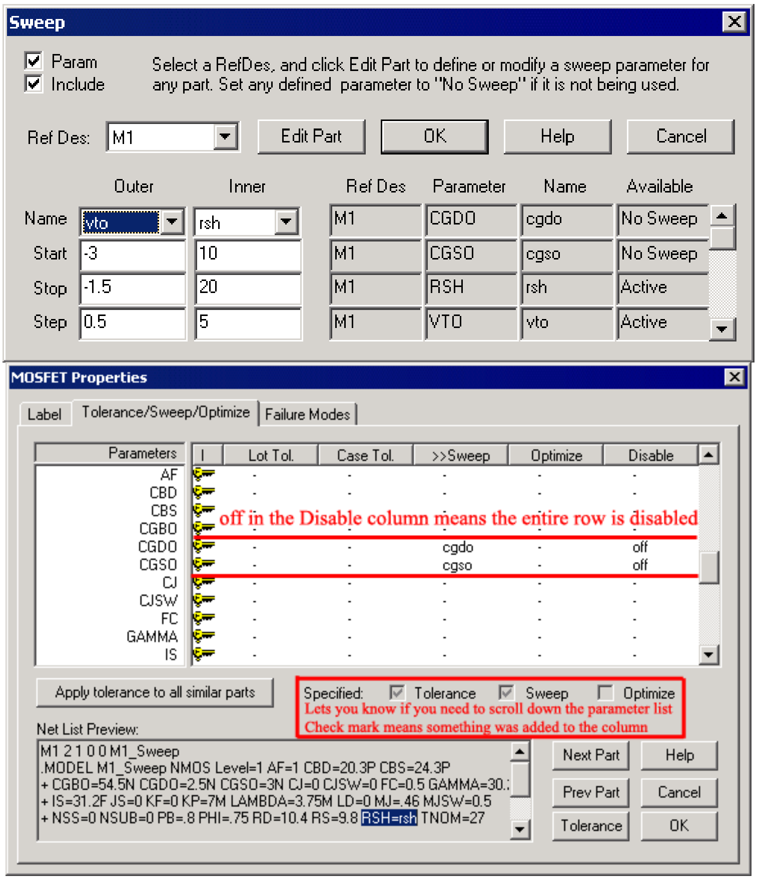
**Figure 4:** When finished editing the symbol choose "save symbol" from the file menu. Browse to the c:\spice8\sn\mysym folder, type "mysym.sym" and press the Open button. Remember the symbol name you use here. You will need to use this name on the *SYM= line in the SubCkt netlist that you export.

From the SpiceNet file drop-down menu select Export. Choose "subckt" in the drop-down list and press OK. By default, the From_Icaps.lib file will be selected, but you can specify your own library file. After makedb is done, Library Manager will come up ready for you to modify the *SRC= and *SYM= lines.

If you revise your schematic then re-export your SubCkt. The subdrawing step is only needed to decide which nodes in your circuit to expose on the symbol. Your modified *SRC= and *SYM= lines will not be changed by re-exporting your SubCkt.

## Improved Advance Analysis

The advance analysis that use tolerance/sweep/optimize tab have been made easier to setup, and we have moved the active/disabled state from the part level to individual parameters for each part.



☑ Param
☑ Include

Select a RefDes, and click Edit Part to define or modify a sweep parameter for any part. Set any defined parameter to "No Sweep" if it is not being used.

Ref Des: M1

| | Outer | Inner |
|---|---|---|
| Name | vto | rsh |
| Start | -3 | 10 |
| Stop | -1.5 | 20 |
| Step | 0.5 | 5 |

| Ref Des | Parameter | Name | Available |
|---|---|---|---|
| M1 | CGDO | cgdo | No Sweep |
| M1 | CGSO | cgso | No Sweep |
| M1 | RSH | rsh | Active |
| M1 | VTO | vto | Active |

| Parameters | I | Lot Tol. | Case Tol. | >>Sweep | Optimize | Disable |
|---|---|---|---|---|---|---|
| AF | | - | - | - | - | - |
| CBD | | - | - | - | - | - |
| CBS | | | | | | |
| CGBO | | - | - | - | - | - |
| CGDO | | - | - | cgdo | - | off |
| CGSO | | - | - | cgso | - | off |
| CJ | | - | - | - | - | - |
| CJSW | | - | - | - | - | - |
| FC | | - | - | - | - | - |
| GAMMA | | - | - | - | - | - |
| IS | | - | - | - | - | - |

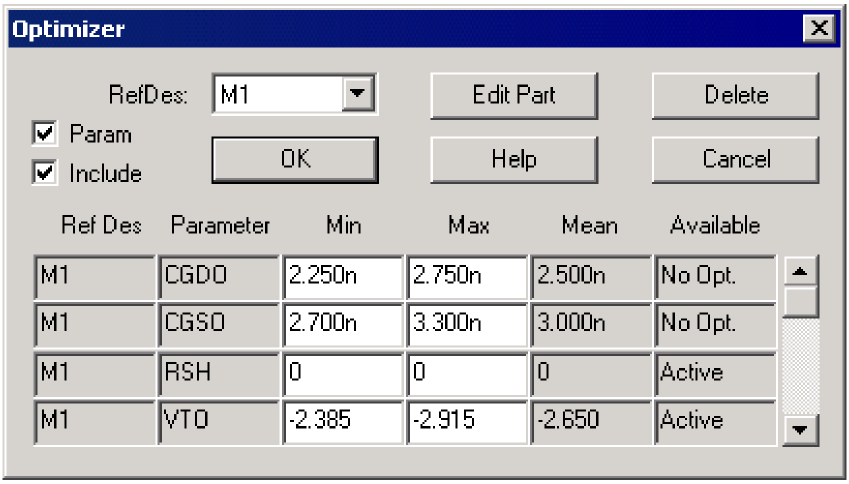
off in the Disable column means the entire row is disabled

Specified: ☑ Tolerance ☑ Sweep ☐ Optimize
Lets you know if you need to scroll down the parameter list
Check mark means something was added to the column

Net List Preview:

```
M1 2 1 0 0 M1_Sweep
.MODEL M1_Sweep NMOS Level=1 AF=1 CBD=20.3P CBS=24.3P
+ CGBO=54.5N CGDO=2.5N CGSO=3N CJ=0 CJSW=0 FC=0.5 GAMMA=30.:
+ IS=31.2F JS=0 KF=0 KP=7M LAMBDA=3.75M LD=0 MJ=.46 MJSW=0.5
+ NSS=0 NSUB=0 PB=.8 PHI=.75 RD=10.4 RS=9.8 RSH=rsh TNOM=27
```

**Figure 5:** Sweep parameters have been added under the Sweep column for CGDO, CGSO, RSH, and VTO in the MOSFET properties dialog for M1. The specified section informs you that something was defined and you need to scroll down the list. You are then able to disable individual sweep variables instead of all sweep variables in the part. The active or disabled states are shown directly in the sweep dialog, eliminating the need to check each part for its states.



| Ref Des | Parameter | Min | Max | Mean | Available |
|---|---|---|---|---|---|
| M1 | CGDO | 2.250n | 2.750n | 2.500n | No Opt. |
| M1 | CGSO | 2.700n | 3.300n | 3.000n | No Opt. |
| M1 | RSH | 0 | 0 | 0 | Active |
| M1 | VTO | -2.385 | -2.915 | -2.650 | Active |

**Figure 6:** Under the optimize column 10% has been entered for CGDO, CGSO, RSH, and VTO parameters. The optimizer dialog shows all active and disabled parameters saving you from having to check each part for its state. Notice that RSH comes up as 0 since a 10% tolerance has been placed on a parameter that is 0.

The sweep output file has been made more organized. Outer and Inner loop device parameter values are listed in association with any user-prescribed measurements, including the plot name associated with each value.

For Extreme Value Analysis, Device tolerances are railed to their maximum or minimum value, based on the sign of an initial sensitivity analysis, such that the result at any measured device or node specified by the user is maximized. A successive simulation is then run with the device's tolerance engaged in a maximum positive or negative direction. This former EVA routine is now called EVA_HI. A new EVA_LO routine similarly performs the above tasks, but device tolerances are railed to their extreme value based on the sensitivity analysis's minimizing the measured result of specified devices or nodes.

In Worst Case Analysis by Sensitivity, a reference simulation is first run on the design and a plot of the data is stored to save the simulation results, notably at nodes or devices specified by the user (i.e., Vout, iR2, etc.). Then a new simulation is run for each device containing tolerance parameters, perturbing the parameter by a small fraction. The difference between these two sets of measurements is saved. The absolute value of operations performed on the difference measurements are summed (for the WCS_HI analysis) or subtracted (for the new WCS_LO analysis), and saved in the IsSpice4 output file for viewing, and in a format that can be read back into the "Results" dialog accessed from SpiceNet's "Simulation Control" dialog.

# Got Fault?

## Automated Fault Diagnosis Using SPICE Simulation for Analog & Mixed-Signal Design

### Introduction

Through the decades the design automation industry has touted accurate simulation of electronic products from concept to production readiness. SPICE simulation for analog and mixed-signal circuitry has been no exception, including its unique ability to mold designs for production compliance.  Analyses like Monte Carlo and worst-case simulation mimic variations in real-world component tolerances on a design, so it can be tweaked for acceptable performance. Such variation is valuable because it cannot otherwise be performed with a lab prototype board or production unit.

But establishing good design margins is not the only factor that helps ensure a product's performance merit.  What happens when a part goes sour in the field, such as a short, open or stuck-at value, or if a power supply faults?  What effect could this have on a product's end use?  The product designer should be privy to end-use risk (i.e., security monitor, backup power system, automobile ABS), but the cause and effect of failed parts is something that's best scrutinized early in the design process, not in the field.

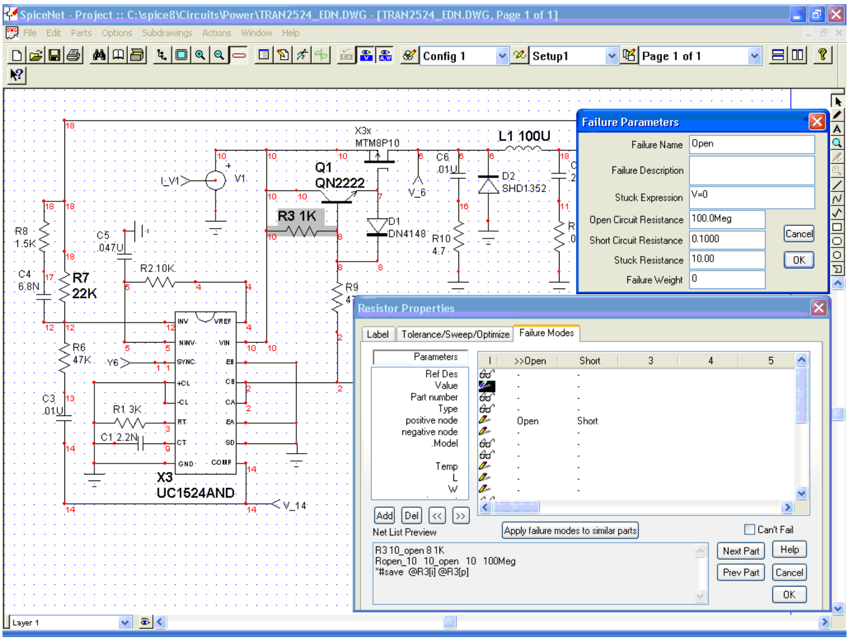
Historically, digital simulators have provided fault diagnosis to analyze the outcome of component failures, viewing the effects that faulty devices inflict on critical design signals. Today, such automated capability is also available for analog-based design (and with mixed digital) by way of Intusoft's automated fault simulation.  Traditionally without this capability, a designer had to employ any number of tricks to painstakingly insert circuit faults. A super-high resistance value (i.e., 1G ohm) could be inserted as an open condition, or a wire might be inserted in place of a component to impose a short. A certain value might be temporarily assigned to a transistor's temperature coefficient to force a "stuck at" condition, or an "open" from base to emitter.  Manually inducing thorough fault coverage in this fashion could take days-to-weeks to setup and simulate, including the need to track an immense amount of alteration to the design database.

Automated fault coverage on the other hand spares the designer from this lengthy and impractical task. In short it provides a convenient means of first assigning pass/fail limits to components' electrical properties (i.e., max power, RMS voltage, transistor collector current, rise time, propagation delay, etc.). Signal variation limits are similarly prescribed for desired lines. Then, shorts, opens and "stuck at" values are assigned to passive and active devices, as well as faulty levels for device temperature and power supply sources. After enabling any number of these conditions, fault simulation is run. Resulting electrical measurements are automatically recorded for desired components and signals lines, including readouts that chart numerical data and pass/fail results.

## Fault Simulation Setup and Implementation

You prepare a fault simulation (in time or frequency domain) by assigning fault conditions to any device (Figure 7). A type of fault can be prescribed in a "failure mode" dialog box for a part (or optionally all "like parts"), as well as power sources and device temperature. Next, Figure 8 illustrates a one-fault setup for a UC1524 advanced regulating PWM controller circuit. Using the mouse, L1 is enabled as "shorted" amongst a vast collection of other possible faults that were assigned throughout the design.

The engineer also selects which electrical properties of the design to monitor during a fault run. Several signals and devices were selected in Figure 9 for monitoring their "initial value" post a fault simulation. A variety of other selectable measurements are provided in the list. The fault run (time domain) was prescribed over a 1-millisecond period. Figure 10 illustrates the resulting measured "initial value" readings, including pass/fail. However, min/max est limits have not yet been assigned.



Failure Parameters

| | |
|---|---|
| Failure Name | Open |
| Failure Description | |
| Stuck Expression | V=0 |
| Open Circuit Resistance | 100.0Meg |
| Short Circuit Resistance | 0.1000 |
| Stuck Resistance | 10.00 |
| Failure Weight | 0 |

Cancel OK

Resistor Properties

Label | Tolerance/Sweep/Optimize | Failure Modes

| Parameters | I | >>Open | Short | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| Ref Des | | - | - | | | |
| Value | | - | - | | | |
| Part number | | - | - | | | |
| Type | | - | - | | | |
| positive node | | Open | Short | | | |
| negative node | | - | - | | | |
| .Model | | - | - | | | |
| | | - | - | | | |
| Temp | | - | - | | | |
| L | | - | - | | | |
| W | | - | - | | | |

Add Del << >>

Apply failure modes to similar parts

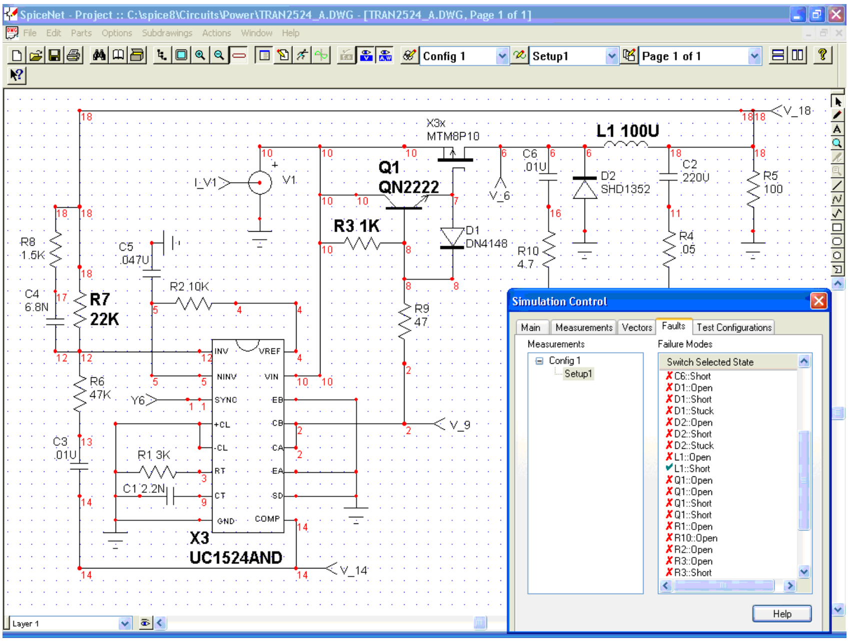
Can't Fail

Net List Preview

```
R3 10_open 8 1K
Ropen_10  10_open  10  100Meg
*#save  @R3[i] @R3[p]
```

Next Part Help Prev Part Cancel OK

**Figure 7:** Fault setup for R3


SpiceNet - Project :: C:\spice8\Circuits\Power\TRAN2524_A.DWG - [TRAN2524_A.DWG, Page 1 of 1]

Config 1 | Setup1 | Page 1 of 1

Simulation Control

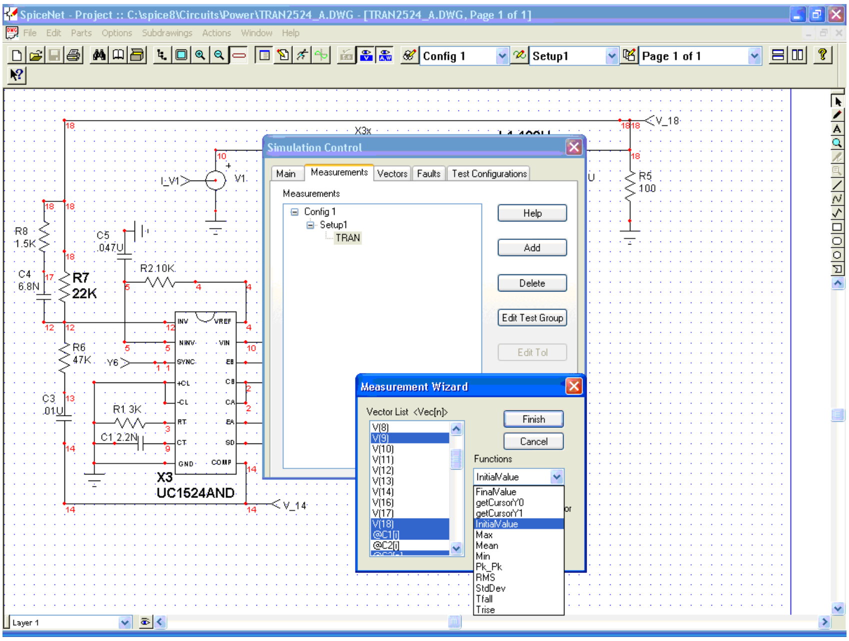
Main | Measurements | Vectors | Faults | Test Configurations

Measurements: Config 1 / Setup1

Failure Modes: Switch Selected State

✗C6::Short
✗D1::Open
✗D1::Short
✗D1::Stuck
✗D2::Open
✗D2::Short
✗D2::Stuck
✗L1::Open
✓L1::Short
✗Q1::Open
✗Q1::Open
✗Q1::Short
✗Q1::Short
✗R1::Open
✗R10::Open
✗R2::Open
✗R3::Open
✗R3::Short

**Figure 8:** Single Fault Enable (L1 Short)


SpiceNet - Project :: C:\spice8\Circuits\Power\TRAN2524_A.DWG - [TRAN2524_A.DWG, Page 1 of 1]

Config 1 | Setup1 | Page 1 of 1

Simulation Control

Main | Measurements | Vectors | Faults | Test Configurations

Measurements: Config 1 / Setup1 / TRAN

Help | Add | Delete | Edit Test Group | Edit Tol

Measurement Wizard

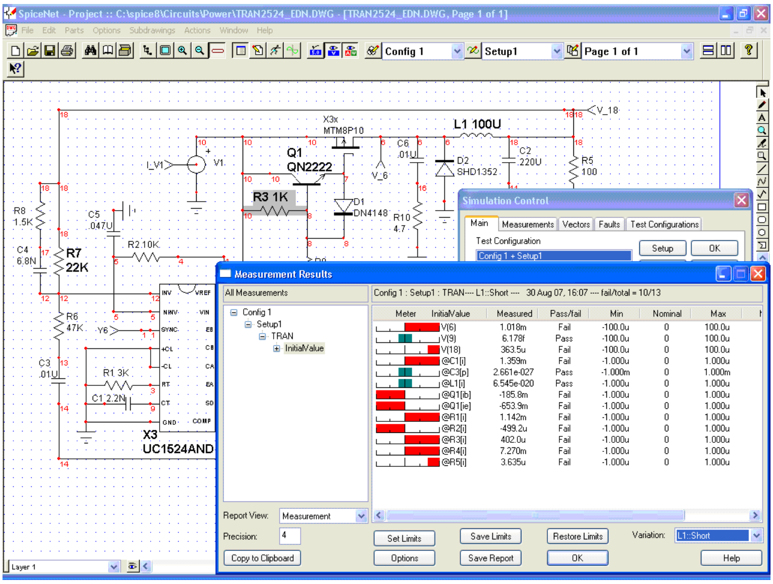
Vector List <Vec[n]>: V(8), V(9), V(10), V(11), V(12), V(13), V(14), V(16), V(17), V(18), @C1[i], @C2[i]

Functions: InitialValue

FinalValue, getCursorY0, getCursorY1, InitialValue, Max, Mean, Min, Pk_Pk, RMS, StdDev, Tfall, Trise

Finish | Cancel

**Figure 9:** Several signals and devices selected to monitor “initial value,” or any of the other measurements


SpiceNet - Project :: C:\spice8\Circuits\Power\TRAN2524_EDN.DWG - [TRAN2524_EDN.DWG, Page 1 of 1]

Measurement Results

All Measurements: Config 1 / Setup1 / TRAN / InitialValue

Config 1 : Setup1 : TRAN---- L1::Short ---- 30 Aug 07, 16:07 ---- fail/total = 10/13

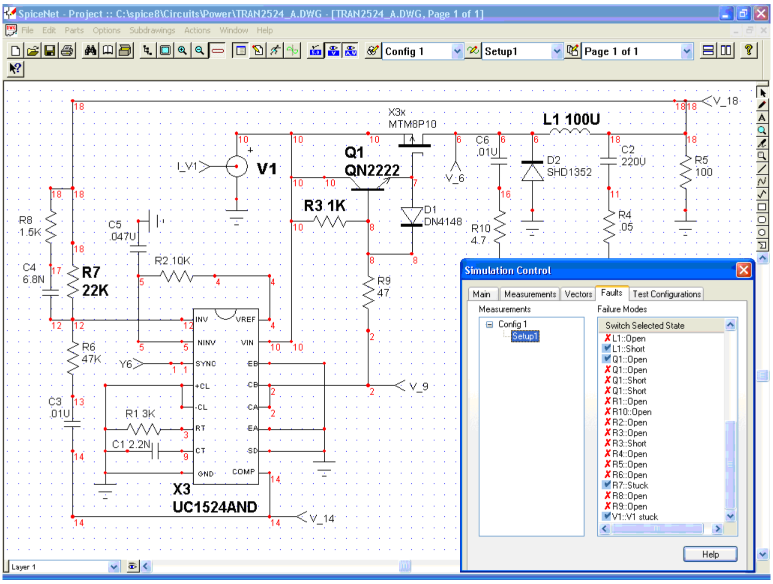
| Meter | InitialValue | Measured | Pass/fail | Min | Nominal | Max |
|---|---|---|---|---|---|---|
| 🟥 | V(6) | 1.018m | Fail | -100.0u | 0 | 100.0u |
| 🟩 | V(9) | 6.178f | Pass | -100.0u | 0 | 100.0u |
| 🟥 | V(18) | 363.5u | Fail | -100.0u | 0 | 100.0u |
| 🟥 | @C1[i] | 1.359m | Fail | -1.000u | 0 | 1.000u |
| 🟩 | @C3[p] | 2.661e-027 | Pass | -1.000m | 0 | 1.000m |
| 🟩 | @L1[i] | 6.545e-020 | Pass | -1.000u | 0 | 1.000u |
| 🟥 | @Q1[ib] | -185.8m | Fail | -1.000u | 0 | 1.000u |
| 🟥 | @Q1[ie] | -653.9m | Fail | -1.000u | 0 | 1.000u |
| 🟥 | @R1[i] | 1.142m | Fail | -1.000u | 0 | 1.000u |
| 🟥 | @R2[i] | -499.2u | Fail | -1.000u | 0 | 1.000u |
| 🟥 | @R3[i] | 402.0u | Fail | -1.000u | 0 | 1.000u |
| 🟥 | @R4[i] | 7.270m | Fail | -1.000u | 0 | 1.000u |
| 🟥 | @R5[i] | 3.635u | Fail | -1.000u | 0 | 1.000u |

Report View: Measurement
Precision: 4
Variation: L1::Short

Simulation Control — Test Configuration: Config 1 + Setup1

**Figure 10:** Measured results for signals and components selected by the designer, with an “L1 short” fault

Figure 11 displays multiple faults enabled for simulation. Note that three new faults were added (R7 stuck at 1K ohms, R3 short and source V1 stuck at 4.9v). Post simulation results are shown in Figure 12.


SpiceNet - Project :: C:\spice8\Circuits\Power\TRAN2524_A.DWG - [TRAN2524_A.DWG, Page 1 of 1]

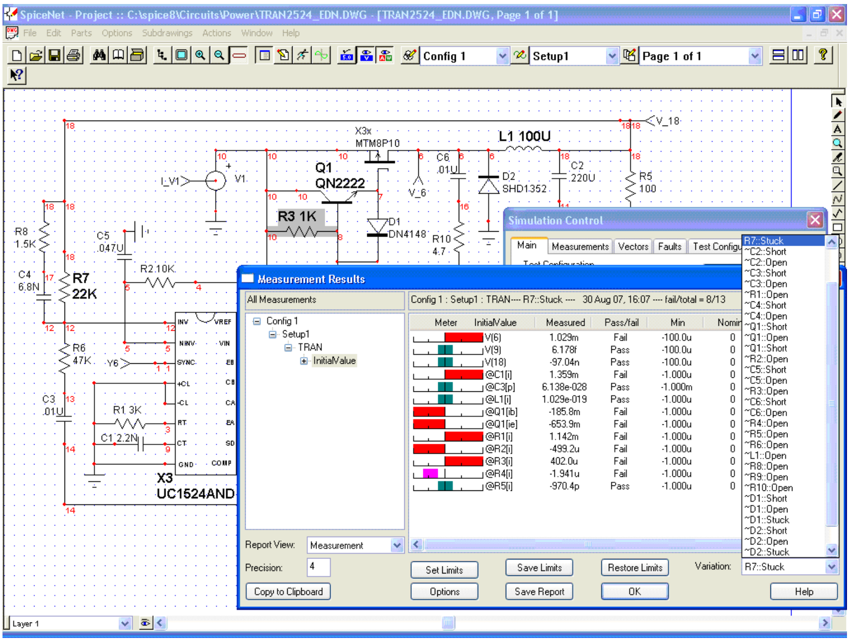
Simulation Control — Faults

Measurements: Config 1 / Setup1

Failure Modes — Switch Selected State:

- ✗ L1::Open
- ✓ L1::Short
- ✓ Q1::Open
- ✗ Q1::Open
- ✗ Q1::Short
- ✗ Q1::Short
- ✗ R1::Open
- ✗ R10::Open
- ✗ R2::Open
- ✗ R3::Open
- ✗ R3::Short
- ✗ R4::Open
- ✗ R5::Open
- ✗ R6::Open
- ✓ R7::Stuck
- ✗ R8::Open
- ✗ R9::Open
- ✓ V1::V1 stuck

**Figure 11:** Three new faults were added to the list and simulations for the four faults will run automatically



SpiceNet - Project :: C:\spice8\Circuits\Power\TRAN2524_EDN.DWG - [TRAN2524_EDN.DWG, Page 1 of 1]

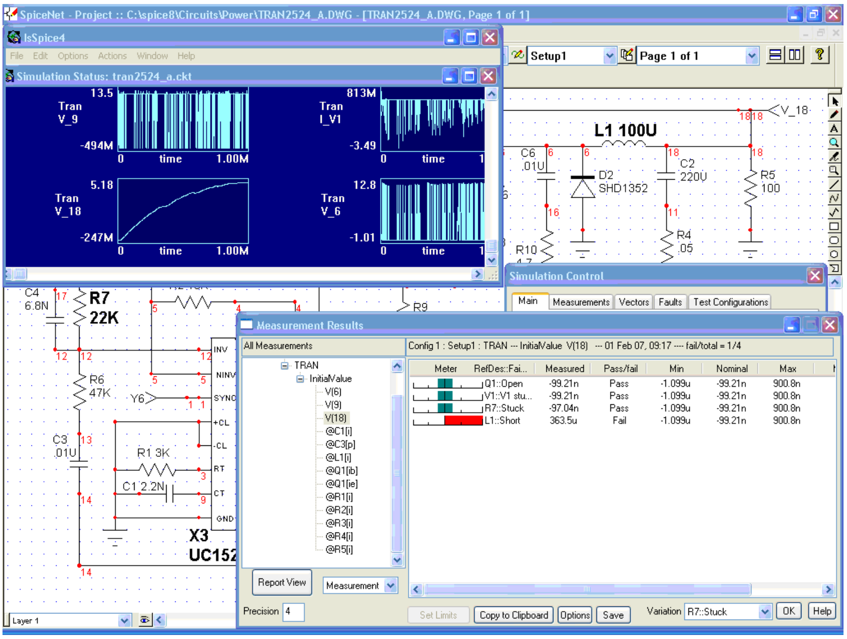
Measurement Results

Config 1 : Setup1 : TRAN---- R7::Stuck ---- 30 Aug 07, 16:07 ---- fail/total = 8/13

| Meter | InitialValue | Measured | Pass/fail | Min | Nomin |
|---|---|---|---|---|---|
| 🟥 | V(6) | 1.029m | Fail | -100.0u | 0 |
| 🟩 | V(9) | 6.178f | Pass | -100.0u | 0 |
| 🟩 | V(18) | -97.04n | Pass | -100.0u | 0 |
| 🟥 | @C1[i] | 1.359m | Fail | -1.000u | 0 |
| 🟩 | @C3[p] | 6.138e-028 | Pass | -1.000m | 0 |
| 🟩 | @L1[i] | 1.029e-019 | Pass | -1.000u | 0 |
| 🟥 | @Q1[ib] | -185.8m | Fail | -1.000u | 0 |
| 🟥 | @Q1[ie] | -653.9m | Fail | -1.000u | 0 |
| 🟥 | @R1[i] | 1.142m | Fail | -1.000u | 0 |
| 🟥 | @R2[i] | -499.2u | Fail | -1.000u | 0 |
| 🟥 | @R3[i] | 402.0u | Fail | -1.000u | 0 |
| 🟪 | @R4[i] | -1.941u | Fail | -1.000u | 0 |
| 🟩 | @R5[i] | -970.4p | Pass | -1.000u | 0 |

Report View: Measurement
Precision: 4
Variation: R7::Stuck

**Figure 12:** Multiple faults simulated with “R7 stuck” selected, which affects the “measured” readings on left



SpiceNet - Project :: C:\spice8\Circuits\Power\TRAN2524_A.DWG - [TRAN2524_A.DWG, Page 1 of 1]

Simulation Status: tran2524_a.ckt

Measurement Results

Config 1 : Setup1 : TRAN --- InitialValue V(18) --- 01 Feb 07, 09:17 ---- fail/total = 1/4

| Meter | RefDes:Fai... | Measured | Pass/fail | Min | Nominal | Max |
|---|---|---|---|---|---|---|
| 🟩 | Q1::Open | -99.21n | Pass | -1.099u | -99.21n | 900.8n |
| 🟩 | V1::V1 stu... | -99.21n | Pass | -1.099u | -99.21n | 900.8n |
| 🟩 | R7::Stuck | -97.04n | Pass | -1.099u | -99.21n | 900.8n |
| 🟥 | L1::Short | 363.5u | Fail | -1.099u | -99.21n | 900.8n |

Precision 4
Variation R7::Stuck

**Figure 13:** Effects of all enabled faults on signal line V_18

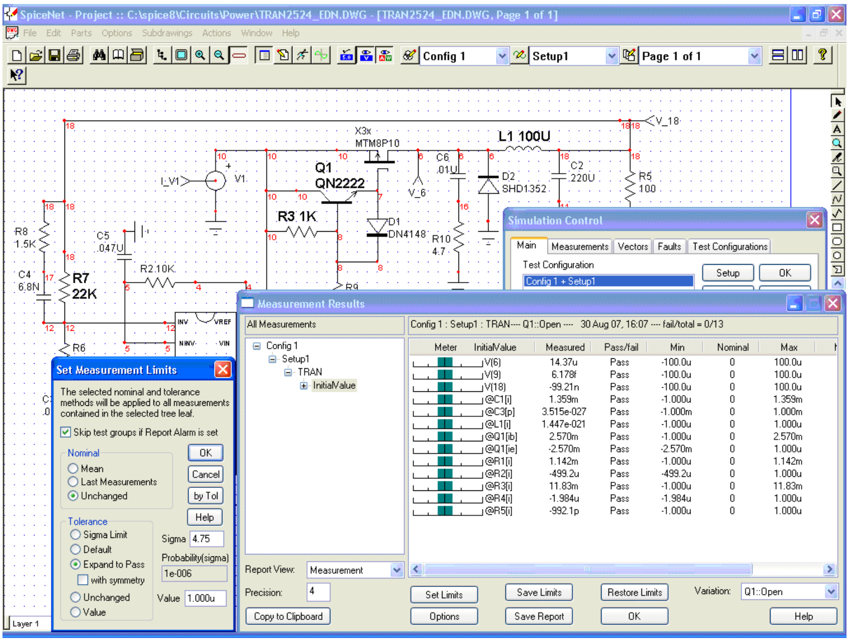
The top five fault selections in Figure 12's list are enabled (corresponding labels put in bold text on the schematic). After simulation, these faults can be randomly selected from the "Variation" menu as shown, to quickly view corresponding changes in the measured readouts on the left. Figure 13 provides a breakout of the "initial value" devices and signals on the left, showing the measured data (right) that the four enabled faults produced for the selected signal V(18). All of these features provide a fast "what-if" analysis of how faults affect critical devices and signal lines. Also shown are thumbnail simulation waveforms that display in real time during fault runs, with V(18) in the low left corner.

## Establishing Design Test Limits

Establishing electrical test limits for devices and signal lines can be achieved in a number of ways. One method of course is to assign them per the designer's discretion. Other ways make use of acceptance criteria from product specifications and component datasheet ratings (i.e., device maximum power). For limits on signal lines, a designer can also run a Monte Carlo statistical analysis, which randomly varies component parameters through their tolerance range, then examines variation on signal lines. Once the circuitry is tweaked enough where variations are acceptable, the range can be used to establish pass/fail signal limits for fault simulation. Such limits are prescribed within the "Set Limits" dialog box for convenience. For example, the (left dialog) in Figure 14 provides a number of ways to open the min/max test limits to accept all measured readings under the present fault at hand (i.e., Q1 open). Sigma limits could have been selected if a previous Monte Carlo run was administered. The "Expand to Pass" option was chosen. Corresponding colored histograms to the right reflect the new "passed" measurements (green). This is useful for quickly studying test limit compliance with successively selected faults, that is, seeing how other faults' measured results comply with the relaxed test limits. For example, when the "R7 stuck" fault was next selected (Figure 15), some of the histograms changed color (failed) despite the new "expand to pass" test limits from Q1 open. Corresponding measured data is also shown.

## Conclusion

Knowing how a design will operate under faulty conditions can be examined early in a product's design-simulation phase. Historically SPICE tools have provided powerful ways of varying component tolerances and temperature to study signal variation for manufacturing compliance. This method and other means can also be used to establish pass/fail criteria for acceptable operation under fault conditions. In the end, design safeguards can be incorporated early in the design cycle to help eliminate possible damage from faults encountered in production and in the field.



Set Measurement Limits

The selected nominal and tolerance methods will be applied to all measurements contained in the selected tree leaf.

☑ Skip test groups if Report Alarm is set

Nominal: ○ Mean ○ Last Measurements [✓ Unchanged]

Tolerance: ○ Sigma Limit ○ Default [✓ Expand to Pass] ☐ with symmetry ○ Unchanged ○ Value

Sigma 4.75

Probability(sigma) 1e-006

Value 1.000u

OK | Cancel | by Tol | Help

Measurement Results

All Measurements: Config 1 / Setup1 / TRAN / InitialValue

Config 1 : Setup1 : TRAN---- Q1::Open ---- 30 Aug 07, 16:07 ---- fail/total = 0/13

| Meter | InitialValue | Measured | Pass/fail | Min | Nominal | Max |
|---|---|---|---|---|---|---|
| 🟩 | V(6) | 14.37u | Pass | -100.0u | 0 | 100.0u |
| 🟩 | V(9) | 6.178f | Pass | -100.0u | 0 | 100.0u |
| 🟩 | V(18) | -99.21n | Pass | -100.0u | 0 | 100.0u |
| 🟩 | @C1[i] | 1.359m | Pass | -1.000u | 0 | 1.359m |
| 🟩 | @C3[p] | 3.515e-027 | Pass | -1.000m | 0 | 1.000m |
| 🟩 | @L1[i] | 1.447e-021 | Pass | -1.000u | 0 | 1.000u |
| 🟩 | @Q1[ib] | 2.570m | Pass | -1.000u | 0 | 2.570m |
| 🟩 | @Q1[ie] | -2.570m | Pass | -2.570m | 0 | 1.000u |
| 🟩 | @R1[i] | 1.142m | Pass | -1.000u | 0 | 1.142m |
| 🟩 | @R2[i] | -499.2u | Pass | -499.2u | 0 | 1.000u |
| 🟩 | @R3[i] | 11.83m | Pass | -1.000u | 0 | 11.83m |
| 🟩 | @R4[i] | -1.984u | Pass | -1.984u | 0 | 1.000u |
| 🟩 | @R5[i] | -992.1p | Pass | -1.000u | 0 | 1.000u |

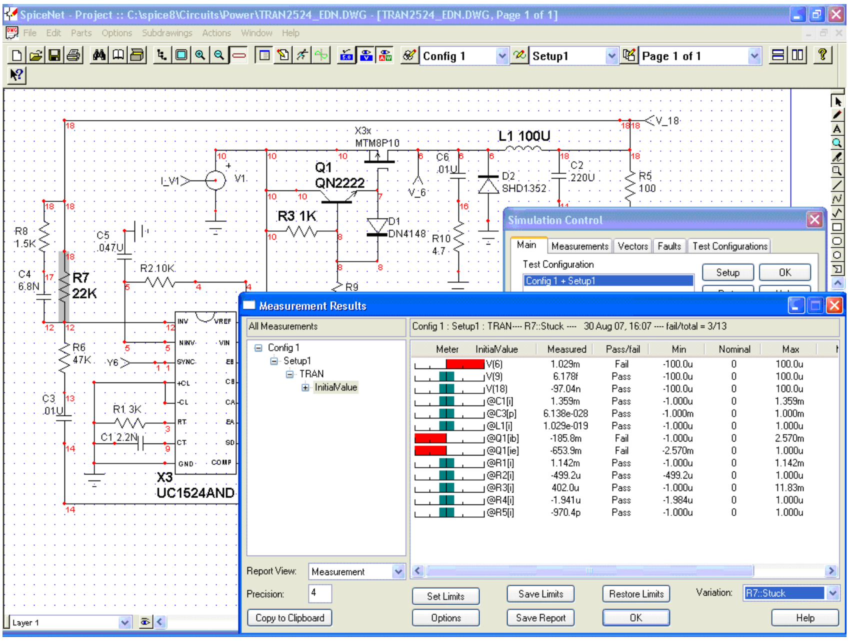
Report View: Measurement

Precision: 4

Variation: Q1::Open

Copy to Clipboard | Set Limits | Save Limits | Restore Limits | Options | Save Report | OK | Help

**Figure 14:** "Expand to Pass" option is set, which opens the pass/fail limits of "Q1 open" to pass all measurements



Measurement Results

All Measurements: Config 1 / Setup1 / TRAN / InitialValue

Config 1 : Setup1 : TRAN---- R7::Stuck ---- 30 Aug 07, 16:07 ---- fail/total = 3/13

| Meter | InitialValue | Measured | Pass/fail | Min | Nominal | Max |
|---|---|---|---|---|---|---|
| 🟥 | V(6) | 1.029m | Fail | -100.0u | 0 | 100.0u |
| 🟩 | V(9) | 6.178f | Pass | -100.0u | 0 | 100.0u |
| 🟩 | V(18) | -97.04n | Pass | -100.0u | 0 | 100.0u |
| 🟩 | @C1[i] | 1.359m | Pass | -1.000u | 0 | 1.359m |
| 🟩 | @C3[p] | 6.138e-028 | Pass | -1.000m | 0 | 1.000m |
| 🟩 | @L1[i] | 1.029e-019 | Pass | -1.000u | 0 | 1.000u |
| 🟥 | @Q1[ib] | -185.8m | Fail | -1.000u | 0 | 2.570m |
| 🟥 | @Q1[ie] | -653.9m | Fail | -2.570m | 0 | 1.000u |
| 🟩 | @R1[i] | 1.142m | Pass | -1.000u | 0 | 1.142m |
| 🟩 | @R2[i] | -499.2u | Pass | -499.2u | 0 | 1.000u |
| 🟩 | @R3[i] | 402.0u | Pass | -1.000u | 0 | 11.83m |
| 🟩 | @R4[i] | -1.941u | Pass | -1.984u | 0 | 1.000u |
| 🟩 | @R5[i] | -970.4p | Pass | -1.000u | 0 | 1.000u |

Report View: Measurement

Precision: 4

Variation: R7::Stuck

Copy to Clipboard | Set Limits | Save Limits | Restore Limits | Options | Save Report | OK | Help

**Figure 15:** R7 "Stuck at" still results in some failures, despite the relaxed test limits from Figure 14