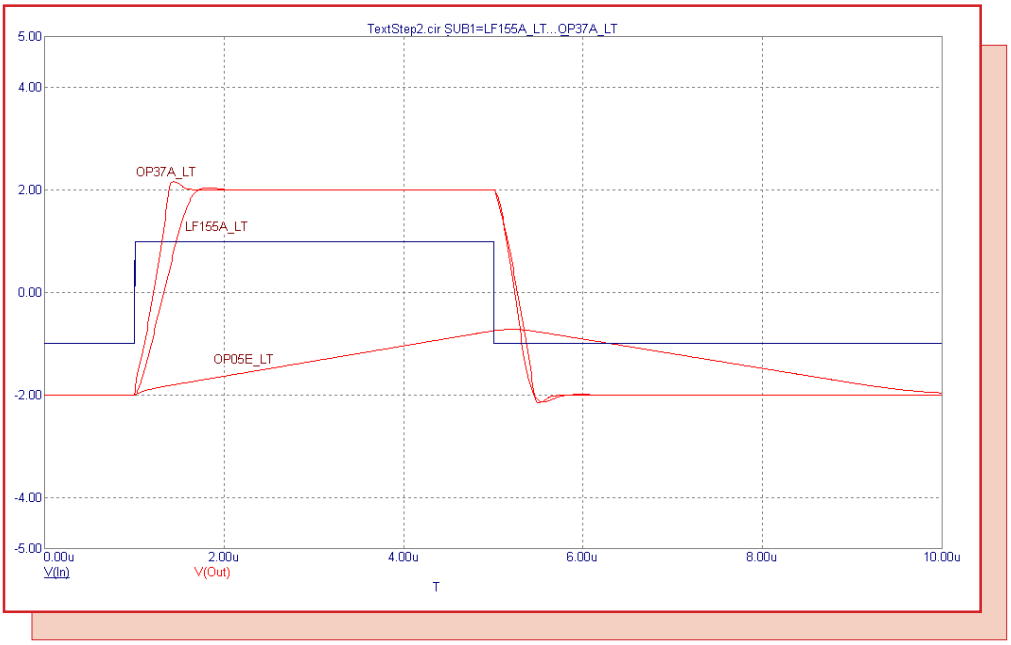
spectrum news

Applications for Micro-Cap™ Users

Fall 2003

News



Text Stepping Models and Subcircuits

Featuring:

- Micro-Cap and Hyperthread Technology
- Text Stepping of Model Statements and Subcircuits
- Digital Clock Macro

News In Preview

This newsletter's Q and A section describes how to determine which display settings are associated with which components, and the use of the Convergence Assist option in the Preferences. The Easily Overlooked Features section describes the simultaneous multi-row analysis limits edits.

The first article describes how Micro-Cap is affected by the new hyperthread technology from Intel.

The second article describes how to use text stepping to step model and subcircuit names.

The third article describes the creation of a simple digital clock macro.

Contents

Book Recommendations

General SPICE

• *Computer-Aided Circuit Analysis Using SPICE*, Walter Banzhaf, Prentice Hall 1989. ISBN# 0-13-162579-9

• *Macromodeling with SPICE*, Connelly and Choi, Prentice Hall 1992. ISBN# 0-13-544941-3

• *Inside SPICE-Overcoming the Obstacles of Circuit Simulation*, Ron Kielkowski, McGraw-Hill, First Edition, 1993. ISBN# 0-07-911525-X

• *The SPICE Book,* Andrei Vladimirescu, John Wiley & Sons, Inc., First Edition, 1994. ISBN# 0-471-60926-9

MOSFET Modeling

• *MOSFET Models for SPICE Simulation, William Liu, Including BSIM3v3 and BSIM4*, Wiley-Interscience, First Edition, ISBN# 0-471-39697-4

VLSI Design

• *Introduction to VLSI Circuits and Systems,* John P. Uyemura, John Wiley & Sons Inc, First Edition, 2002 ISBN# 0-471-12704-3

Micro-Cap - German

• *Schaltungen erfolgreich simulieren mit Micro-Cap V,* Walter Gunther, Franzis', First Edition, 1997. ISBN# 3-7723-4662-6

Micro-Cap - Finnish

• *Elektroniikkasimulaattori,* Timo Haiko, Werner Soderstrom Osakeyhtio, 2002. ISBN# ISBN 951-0-25672-2

Design

• *Microelectronic CircuitsHigh Performance Audio Power Amplifiers,* Ben Duncan, Newnes, First Edition, 1996. ISBN# 0-7506-2629-1

• *Microelectronic Circuits.,* Adel Sedra, Kenneth Smith, Fourth Edition, Oxford, 1998

High Power Electronics

• *Power Electronics,* Mohan, Undeland, Robbins, Second Edition, 1995. ISBN# 0-471-58408-8

• *Modern Power Electronics,* Trzynadlowski, 1998. ISBN# 0-471-15303-6

Switched-Mode Power Supply Simulation

• *SMPS Simulation with SPICE 3,* Steven M. Sandler, McGraw Hill, First Edition, 1997. ISBN# 0-07-913227-8

• *Switch-Mode Power Supply SPICE Simulation Cookbook*, Christophe Basso, McGraw-Hill 2001. This book describes many of the SMPS models supplied with Micro-Cap.

Micro-Cap Questions and Answers

Question: I have been running Dynamic DC analysis. When I display the currents, powers, or conditions, it can sometimes be difficult to determine which component the display belongs to. Is there a way to view which component the display is linked to?

Answer: There is a simple way to view which displays belong to a specific component. While in Dynamic DC analysis, enable the Select mode. In Select mode, click on one of the components in the schematic. Any displays associated with this component will be redrawn in the select color. This will work for any current, power, or condition display. Voltage displays are associated with a node and not a specific component so they will not be affected by this method.

Question: I am running a 100ms transient analysis on my schematic. The simulation gets to approximately the 30ms mark, and it then starts the simulation over again. It will then run through the entire simulation and any subsequent simulations. If I revert my circuit, I get the restart again during the first run. Why does the restart occur?

Answer: In the Preferences dialog box, there is an option called Convergence Assist which will be enabled. Convergence Assist is invoked if the simulation runs into convergence problems. It may modify Global Settings parameters such as RELTOL, ABSTOL, VNTOL, ITL2, ITL4, METHOD, GMIN, and others to achieve convergence. Since a change in the Global Settings can affect the SPICE matrix, the simulation needs to be restarted. If the assist succeeds, it adds a .OPTIONS statement to the text area of the circuit with the modified parameters so that subsequent runs converge more readily.

Easily Overlooked Features

This section is designed to highlight one or two features per issue that may be overlooked because they are not made visually obvious with a toolbar button.

Simultaneous Multi-Row Analysis Limits Edits
The analysis limits dialog box contains five text fields for each expression that is entered into Micro-Cap. The text fields are: P, X Expression, Y Expression, X Range, and Y Range. These five fields describe the waveform to be plotted, the range to plot it in, and which plot group the waveform is to be placed in. For the P, X Expression, X Range, and Y Range fields, all of the waveforms often share the same value such as T for the X Expression in a transient analysis. Performing a left click on the column header invokes a menu that lets you rapidly change the value of all of the rows for that column simultaneously. One of the most common uses for this technique is to quickly set all of the Y Range fields to Auto for the next run.

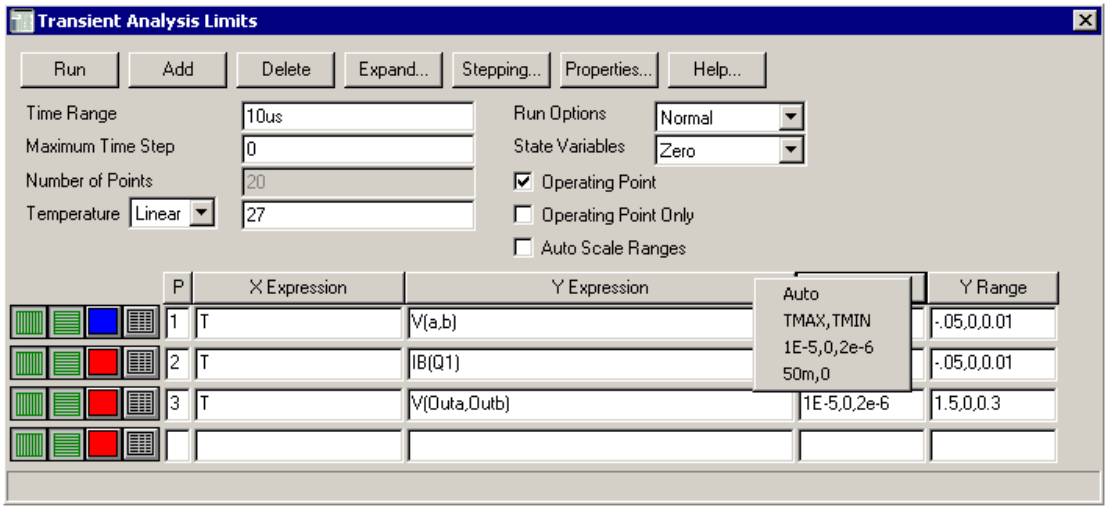


Fig. 1 - Transient Analysis Limits dialog box

Figure 1 displays an example of the menu that was created when the X Range header was clicked. The menu contains entries that are either predefined in Micro-Cap or have been specified by the user in one of the rows of the column. In the above figure, the 'Auto' and 'TMAX,TMIN' entries will always be in the menu for the X Range. The '1E-5,0,2e-6' and '50m,0' entries were taken from the values that are currently defined for the X Range fields. Selecting any of these will set all rows to the selected value.

Clicking on the Y Expression header will invoke the Variables List which lets you choose circuit variables to place in the Y Exression fields. Obviously, there aren't many cases where one would want to set the Y Expression field to the same value for all waveforms, but there are the rare instances where it may be useful as a starting point in defining waveforms.

Micro-Cap and Hyperthread Technology

Intel has recently incorporated a new technology called hyperthreading into their latest CPU offerings. It was initially introduced for the 3.1GHz Pentium 4 processor and will probably become fairly standard for business computers in the future.

Hyperthreading technology acts much like a multiple processor system. In fact, as far as the Windows operating system is concerned, there are two processors in the computer. Each application loaded is assigned a thread through the processor. The advantage to this technique is that multiple programs can be running at the same time without as much system degradation since one application isn't dominating the CPU's processing time. This can be quite useful if you have programs running in the background such as a virus scanner or when Micro-Cap is performing a long simulation.

While this technology will improve the stability and performance of the system when more than one program is running, only applications that have been written to take advantage of hyperthreading will see their individual performance improve when compared to a single processor computer.

The main disadvantage of the hyperthreading technology is that it still has only a single floating point unit. Applications that use highly intensive floating point operations such as Micro-Cap would see little to no benefit from optimizing for hyperthreading due to the floating point unit bottleneck. If two applications are running that both use the floating point unit extensively then much of the benefit of hyperthreading is lost. The optimum situation is to have an application such as Micro-Cap running along with an application such as Microsoft Word so that the floating point unit can be devoted entirely to Micro-Cap. The hyperthreading benefit will still be available when running Word at the same time.

A question that arises from all this is if Micro-Cap is running by itself, does hyperthreading help or hinder the speed of a simulation? With hyperthreading enabled, when you go to the Windows Task Manager and view the Processes that are running, the MC7.EXE process will appear to be taking up only 50% of the CPU time even though the other 50% is only being used by the System Idle Process. On a standard single processor, when Micro-Cap is running an analysis, it will typically take up to 99% of the CPU time when running by itself. At first instance, this would seem to be a large detriment to running Micro-Cap with hyperthreading as it appears to use only half of the resources of the CPU then it would if hyperthreading was disabled.

In order to test this, we ran a section of our trusty benchmarks on a new Dell Dimension 8300 which contains the hyperthreading technology. The benchmarks were run twice, once with hyperthreading enabled and a second time when hyperthreading was disabled. Hyperthreading can be enabled/disabled in the BIOS screen of the system. This is the screen that can be accessed usually by hitting either the F2 or Delete key when the system first boots up. The following table shows a comparison of a sample of the circuits in the benchmarks along with the time it took for this entire section of the benchmarks to run. Note that the circuits shown in this sampling have been modified so that they are not a direct time comparison with the circuits of the same name that are distributed with Micro-Cap.

Circuit	Disabled	Enabled
UC1845_BOOST.CIR	10.657s	10.719s
S_BOOST_VM.CIR	1.234s	1.234s
CORE2.CIR	3.203s	3.187s
S_BUCK_VM.CIR	11.938s	11.469s
S_BUCK_SYN.CIR	1.141s	1.156s
S_BUCK_SYN2.CIR	5.391s	5.453s
APC.CIR	4.015s	4.156s
S_BUCKBOOST_.CIR	1.656s	1.593s
S_FLYBACK_CM.CIR	2.844s	2.937s
S_FLYBACK_VM.CIR	2.531s	2.532s
THERMAL1.CIR	5.437s	5.406s
S_FORWARD_VM.CIR	3.719s	3.735s
S_FULL_CM.CIR	8.484s	8.532s
S_FULL_VM.CIR	2.813s	2.750s
S_FULL_XFMR.CIR	1.906s	1.891s
S_HALF_VM.CIR	2.344s	2.391s
S_PUSH_VM.CIR	3.172s	3.079s
S_PUSH_CM.CIR	1.360s	1.343s
S_2FOR_CM.CIR	2.906s	2.937s
S_2FLY_CM.CIR	1.329s	1.328s
ALL	227.966s	231.429s

As can be seen in the above table, having hyperthreading enabled or disabled makes very little difference in the time of the runs. The Disabled column displays the run time of the circuit when hyperthreading is off, and the Enabled column displays the run time when hyperthreading is on. The difference between the total benchmark run times of the two states is about 1.5%. This falls within the error tolerance when running the benchmark on the same system from one day to the next.

Our recommendation is to leave the hyperthreading enabled. There is very little, if any, penalty in having it enabled, and it is a useful feature when running multiple applications.

Text Stepping of Model Statements and Subcircuits

A useful feature of stepping in Micro-Cap is the ability to step through specified text strings. This feature can be used to step the model name or subcircuit name of a component. This method provides a simple route to compare the output waveforms of a schematic when different models are being used for the simulation.

Text stepping requires the use of a .define statement to declare the initial value of the define variable. This initial value will be overwritten when the stepping is performed, but it must be a model or subcircuit name that Micro-Cap will find and that is applicable to the component. If the initial value doesn't adhere to these standards, then Micro-Cap will return an error message. The define variable name must then be used to specify the MODEL attribute, if you are stepping a model statement, or a NAME attribute, if you are stepping a subcircuit.

To step this define variable, in the Stepping dialog box, the Symbolic Parameter Type needs to be selected which provides a list of all symbolic parameters available to be stepped. For text stepping, the List Method must be used since the Log and Linear methods will obviously not be applicable when stepping a text string.

Stepping a Model Statement
The schematic in Figure 2 displays a basic differential amplifier. There are six NPN transistors within this amplifier configuration. Each of these transistors has had its MODEL attribute defined as NPN1. A define statement has been added to the schematic that states:

.DEFINE NPN1 2N2222A

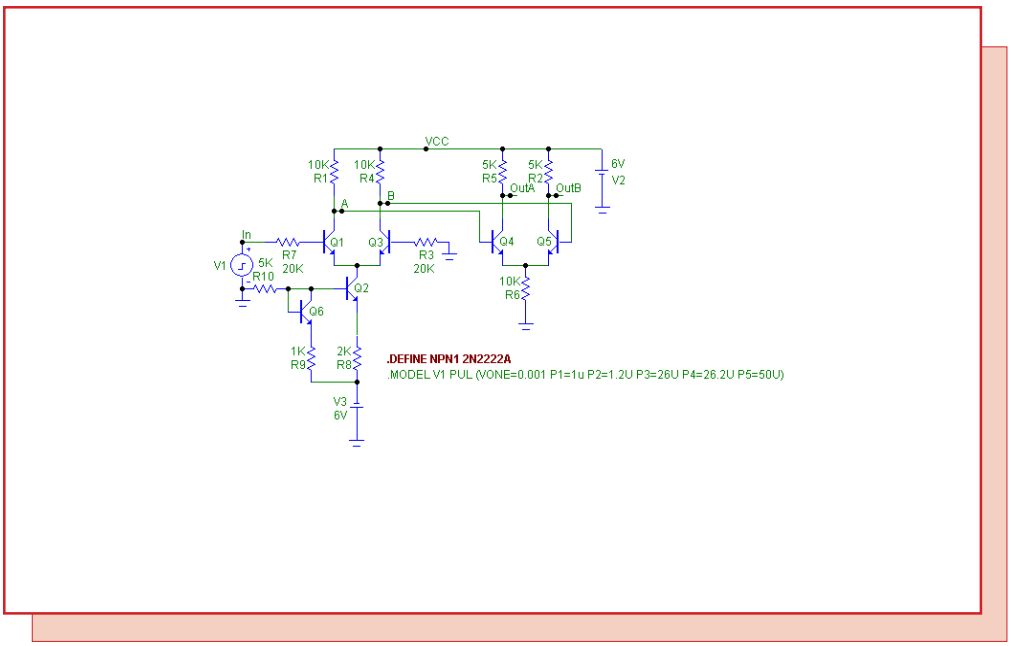


Fig. 2 - Model Statement Stepping Example Circuit

Through this define statement, the MODEL attribute for all six transistors is now set at 2N2222A which is a valid NPN model in the Micro-Cap libraries. To step the model statement, an analysis then needs to be entered. For this example, AC analysis will be used. In the Stepping dialog box for the AC analysis, the Parameter Type option has been set to Symbolic, the Method option to List, and the Step It option to Yes. The Step What and List fields are defined as follows:

Step What: NPN1
List: 2N1711, 2N2222A, 2N2218, ZTX869

The Step What field should be set with the name of the variable in the define statement which was NPN1 in this case. The List field is a comma delimited list of values. For text stepping, these will be the names of the models that are to be simulated. For this circuit, each of these model names must be the name of a valid NPN model statement that resides locally in the circuit file or in the libraries. These settings will produce four analysis runs where the NPN transistors in each run are set to one of the model names specified in the list field.

The resulting AC analysis is displayed in Figure 3. The magnitude in decibels of the voltage at node OutA is the waveform shown. Each branch of the plot has been labelled with the name of the model that produced the corresponding run. Note that the 2N1711, 2N2218, and 2N2222A are all basic general purpose transistors and produce similar outputs. The ZTX869 transistor is a medium power, high current transistor that has a smaller bandwidth and higher gain compared to the other three transistor as is evident in the plot.

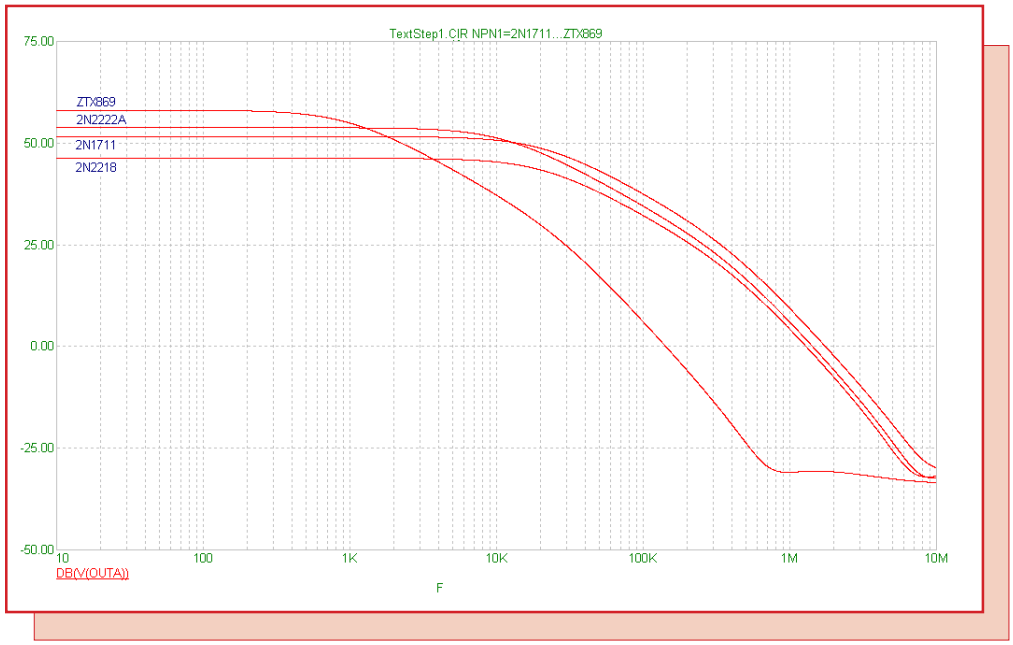


Fig. 3 - Model Statement Stepping Analysis

Stepping a Subcircuit Name

Stepping a subcircuit name is very similar to stepping a model statement. The schematic in Figure 4 displays an opamp in a noninverting configuration with a gain of two that has a pulse source as its input. The opamp in the schematic is a subcircuit component. Its NAME attribute has been defined with the name SUB1, and there is a corresponding define statement in the schematic that states:

.define SUB1 LF155A_LT

This defines the opamp as using the LF155A_LT subcircuit model. For this example, transient analysis will be run. Again the Stepping dialog box settings have been set so that the Parameter Type option is Symbolic, the Method option is List, and the Step It option is set to Yes. The Step What and List fields are defined as follows:

Step What: SUB1
List: LF155A_LT, OP05E_LT, OP37A_LT

The Step What field has been defined with the name of the define variable referenced in the subcircuit. The List field sets up the analysis so that it will produce three runs with each run using one of three different opamp subcircuit models.

One item to be careful of when stepping subcircuit names is that any model referenced in the List field must have the same pin layout as the subcircuit component that is present in the schematic. If the Pin Names are displayed for the X2 component in the schematic, it can be seen that they are defined as:

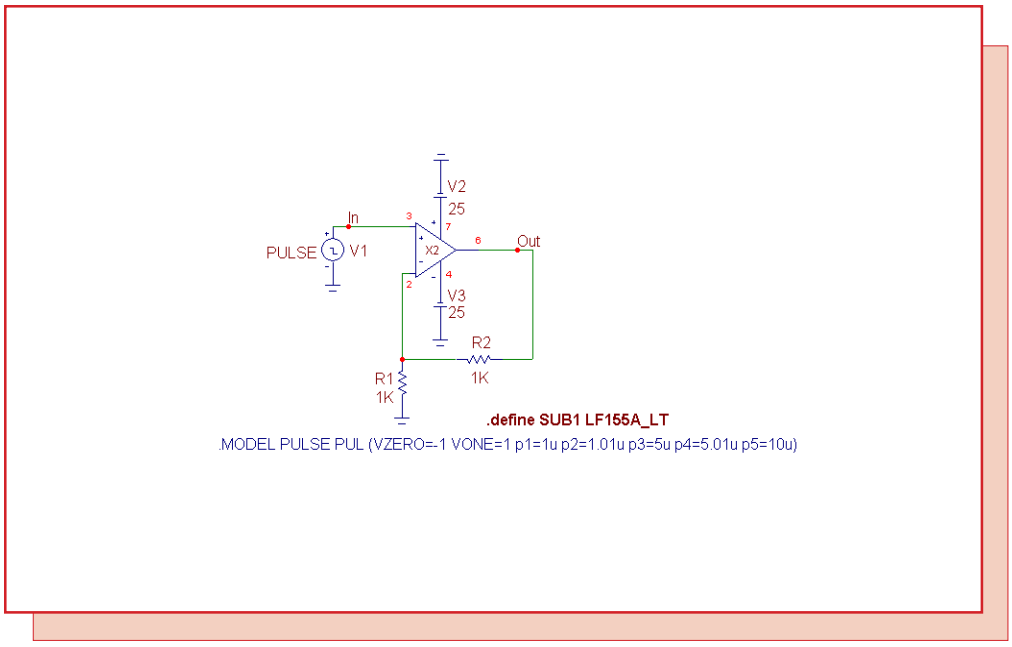


Fig. 4 - Subcircuit Name Stepping Example Circuit

3 - Noninverting input
2 - Inverting input
7 - Positive power supply
4 - Negative power supply
6 - Output

All three subcircuit models referenced in the List field have their pin layout defined such as:

.SUBCKT LF155A_LT 3 2 7 4 6

where the pins in the subcircuit model match the functions above. A subcircuit model that uses different pin numbers or has a different amount of pins will return an error. Note that if in the subcircuit model, pin 2 referenced the noninverting input and pin 3 referenced the inverting input, no error would be given because the pin names are the same as in the X2 component, but the resulting analysis would not produce the expected results since the inputs would be swapped when the simulation is run.

The transient analysis of this circuit is displayed in Figure 5. The plot displays the opamp output response to a 4us pulse at its input for each of the three subcircuit models stepped. Each branch of the simulation has been labelled with the subcircuit model name that produced the corresponding run. As can be seen in the analysis, the OP05E opamp is too slow to pass the pulse through, but the LT155A and the OP37A opamps both do a good job of reproducing the pulse with the specified gain at the output.

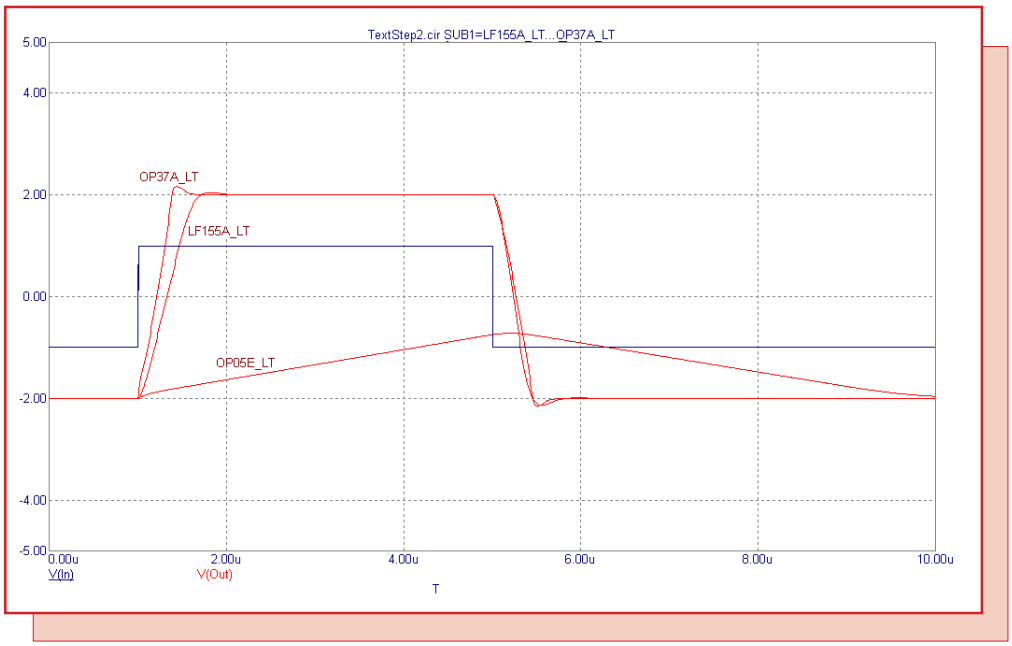


Fig. 5 - Subcircuit Name Stepping Analysis

Digital Clock Macro

The digital clock macro provides a simple method for implementing a digital clock stimulus in a schematic. It eliminates the complexity some people run into when trying to define a clock stimulus in the traditional way.

The traditional way of defining a digital stimulus with a clock output is to use the Stim1 component. This component is placed in the schematic and has its COMMAND attribute defined with a symbolic name. Any valid text string may be used for the name. In this case, we will use the name Clock. The symbolic name then needs to be specified using a define statement such as the following:

```
.define Clock
+ Label = Start
+ 0ns 0
+ +50ns 1
+ +50ns Goto Start -1 Times
```

The define statement can be specified in the text area or directly within the Attribute dialog box of the Stim1 component. In this case, the digital stimulus has been defined as a clock with a 100ns period and a 50% duty cycle. At 0ns, the output of the clock is set to a zero state. 50ns later, the clock output is set to a one state. 50ns after that, the Start label is returned to and the sequence begins again. The 'Goto Start -1 Times' specifies that this loop will repeat forever.

The shortcut for this technique is to use the digital clock macro whose circuit is displayed in Figure 6.

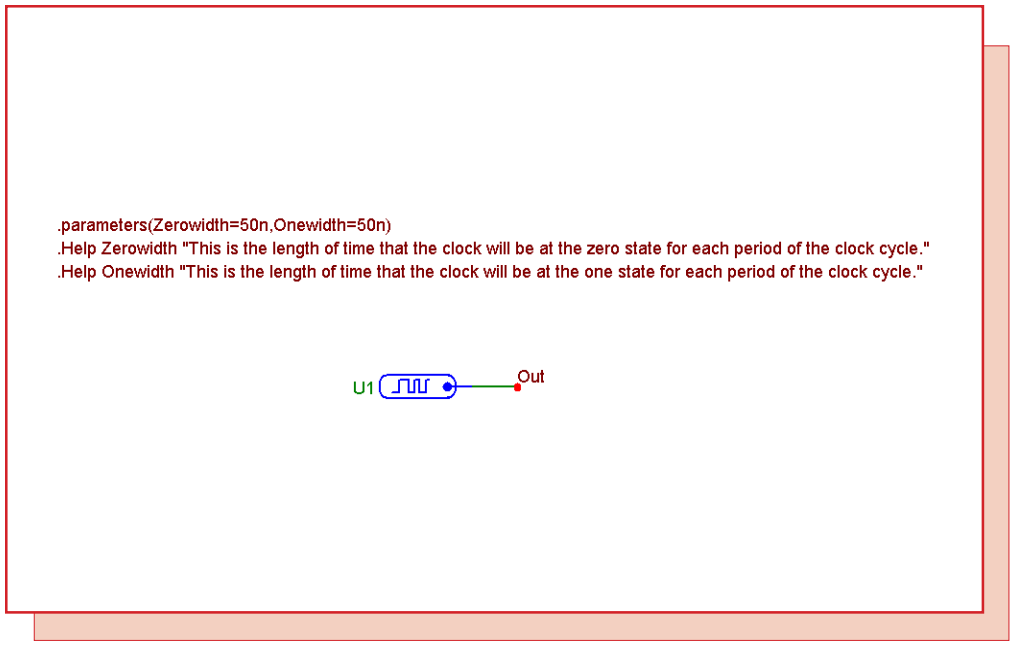


Fig. 6 - Digital Clock Macro Circuit

The digital macro circuit consists of just one Stim1 digital stimulus source. There are two parameters defined for this macro: Zerowidth and Onewidth. The Zerowidth parameter defines the time in seconds that the clock will be at the zero state for each period of the clock cycle. The Onewidth parameter defines the time in seconds that the clock will be at the one state for each period of the clock cycle. The period of the clock cycle is the sum of the Zerowidth and Onewidth parameters. The COMMAND attribute for the Stim1 source has been defined as CLOCK. In the text area of the macro circuit, there is a corresponding define statement as follows:

```
.define CLOCK
+ Label = Start
+ 0 0
+ +Zerowidth 1
+ +Onewidth Goto Start -1 times
```

The two macro parameters are used here to define the transition times of the stimulus output. In this case, the stimulus output starts at the zero state at time=0. After Zerowidth seconds, the output transitions to the one state. In another Onewidth seconds, the Goto statement is triggered and sets the stimulus output back to the zero state. The Goto loop is set to repeat indefinitely.

That is the entire macro. While simple, it clearly demonstrates one of the advantages of using macros in being able to simplify operations. Now, only two parameters need to be specified to create a digital clock of any period with any duty cycle in a schematic.

The digital clock macro is used in the circuit displayed in Figure 7. The macro, X1, is used as a clock input to the first JK flip-flop of a 4-bit binary counter. Standard Stim1 sources are used to

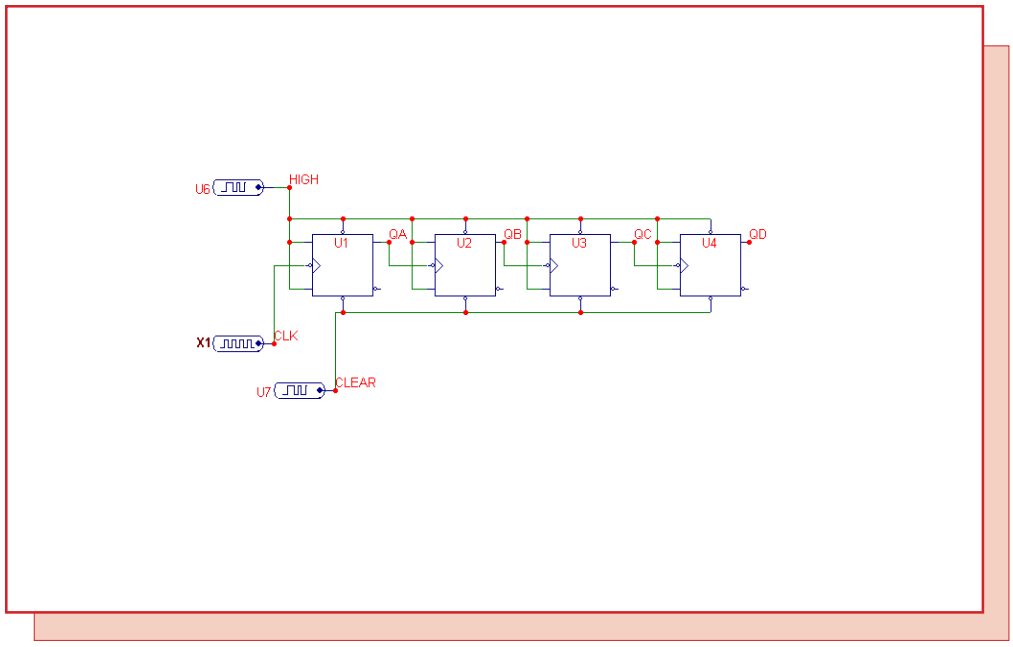


Fig. 7 - Digital Clock Macro Example Circuit

define the digital waveform inputs for the Clear and Preset pins of all four JK flip-flops. The digital clock macro has its parameters defined as:

Zerowidth = 40n
Onewidth = 60n

These parameters define a digital clock with a period of 100ns whose one state has a 60% duty cycle.

The resulting analysis is displayed in Figure 8. The d(CLK) expression plots the output of the digital clock macro. Two horizontal tags have been placed to show the period of the clock and the width of the one state during that period. The width of the one state is shown as 60ns, and the period is shown as 100ns which both match the specifications defined for the macro.

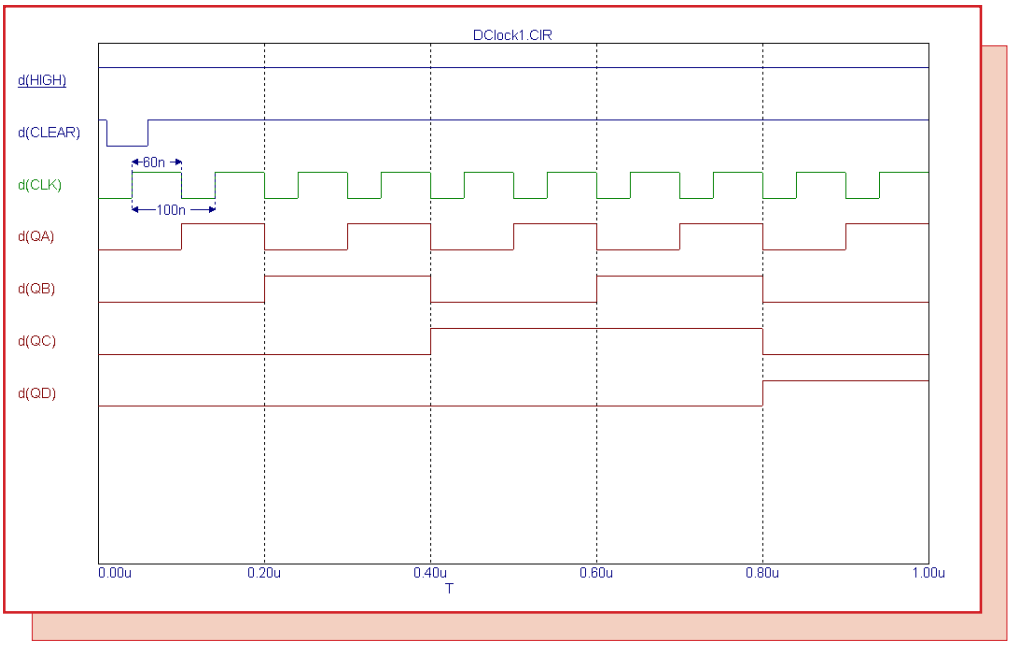


Fig. 8 - Digital Clock Macro Analysis

Product Sheet

Latest Version numbers

Micro-Cap 7 ..... Version 7.2.4
Micro-Cap 6 ..... Version 6.3.3
Micro-Cap V ..... Version 2.1.2

Spectrum's numbers

Sales ..... (408) 738-4387
Technical Support ..... (408) 738-4389
FAX ..... (408) 738-4702
Email sales ..... sales@spectrum-soft.com
Email support ..... support@spectrum-soft.com
Web Site ..... http://www.spectrum-soft.com
User Group ..... micro-cap-subscribe@yahoogroups.com