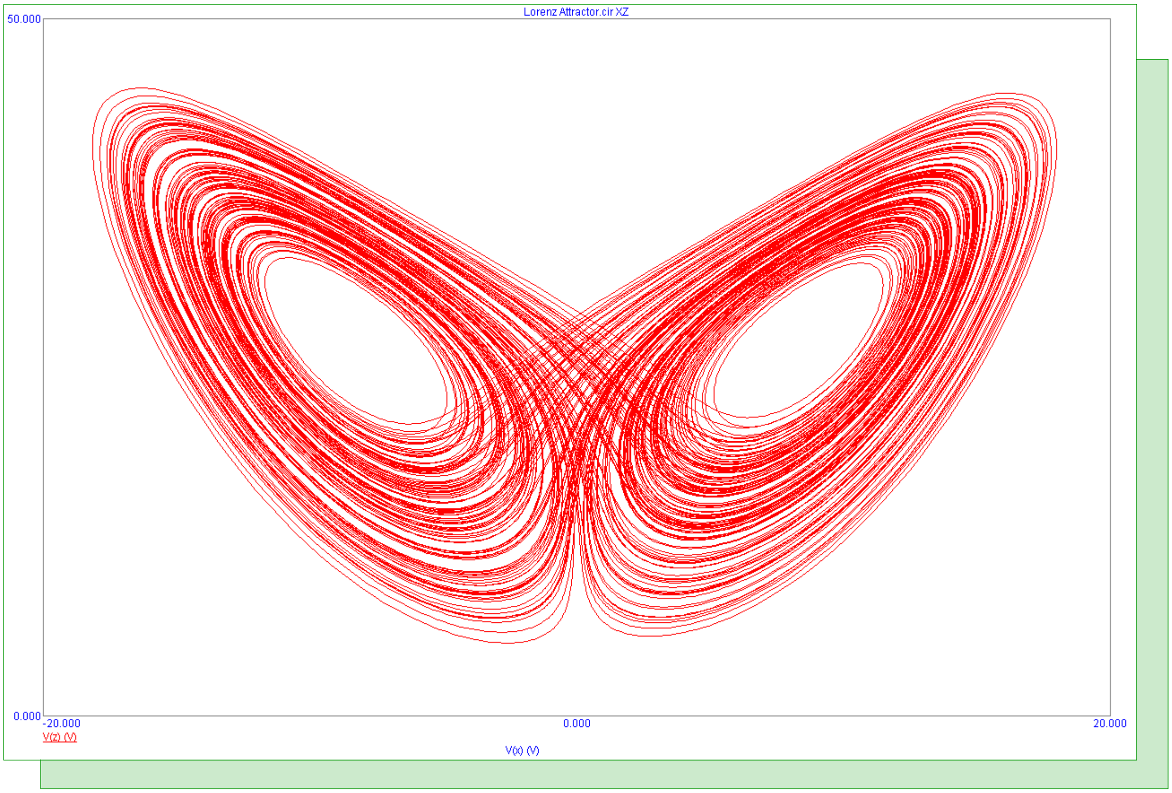
spectrum news

Applications for Micro-Cap™ Users

Spring 2007

News



Simulating the Lorenz Attractor

Featuring:
- Using the Waveform Buffer
- Simulating the Lorenz Attractor
- Lead Acid Battery Macro

News In Preview

This newsletter's Q and A section describes how to enter common engineering and mathematical symbols as text into the schematic. The Easily Overlooked Feature section describes the use of the thumb nail plot capability in an analysis to provide a small guide plot and aid in scaling and panning a plot.

The first article describes how to use the Waveform Buffer to store and retrieve waveforms so that they can be imported into other simulations.

The second article describes how to use behavioral modeling to simulate the Lorenz Attractor.

The third article describes a macro that models the discharge process of a lead acid battery.

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 - Czech

- *Resime Elektronicke Obvody,* Dalibor Biolek, BEN, First Edition, 2004. ISBN# 80-7300-125-X

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 Circuits High 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 am placing text documentation on areas of my schematic. It would be useful if I had characters such as the ohm symbol or the PI symbol. Is there any way to enter these into a Micro-Cap schematic?

Answer: There is a method to use these symbols in a schematic. You must have the Symbol font installed on your system. The Symbol font actually has quite a few symbols that can be useful for engineering or mathematical documentation. The following list displays some of the symbols available in this font along with their corresponding keyboard entry. Note that all of the characters in a text string in a Micro-Cap schematic must share the same font so these symbols would have to be entered separately and used in conjunction with a standard text string.

Ω - W
α - a
β - b
μ - m
π - p
θ - q
∴ - \
Δ - D
Σ - S

The following symbols have a different entry method but are also available in the Symbol font. For these you want to hold the ALT key down and then type in the four digits on the numeric keypad, not the standard numbers at the top of the keyboard. When you release the ALT key after typing these numbers, the corresponding symbol will appear when the font has been set to Symbol.

° - ALT + 0176
∞ - ALT + 0165
± - ALT + 0177
⇒ - ALT + 0222
≈ - ALT + 0187
∠ - ALT + 0208
≤ - ALT + 0163
≥ - ALT + 0179
÷ - ALT + 0184
≠ - ALT + 0185
≡ - ALT + 0186
√ - ALT + 0214
∫ - ALT + 0242

Easily Overlooked Features

This section is designed to highlight one or two features per issue that may be overlooked among all the capabilities of Micro-Cap.

Thumb Nail Plot

Micro-Cap has the capability to produce a thumbnail plot of the selected plot group. This feature provides a small guide plot which gives a global view of where the current plot displays are on the whole curve. You can select different display views by dragging a box in the plot with the left mouse button. Dragging the box with the right mouse button pans the selected plot group. Transient, AC, and DC analysis all have this capability in both the standard analysis and the Probe analysis. The thumbnail plot is created by selecting the Thumb Nail Plot command under the Scope menu.

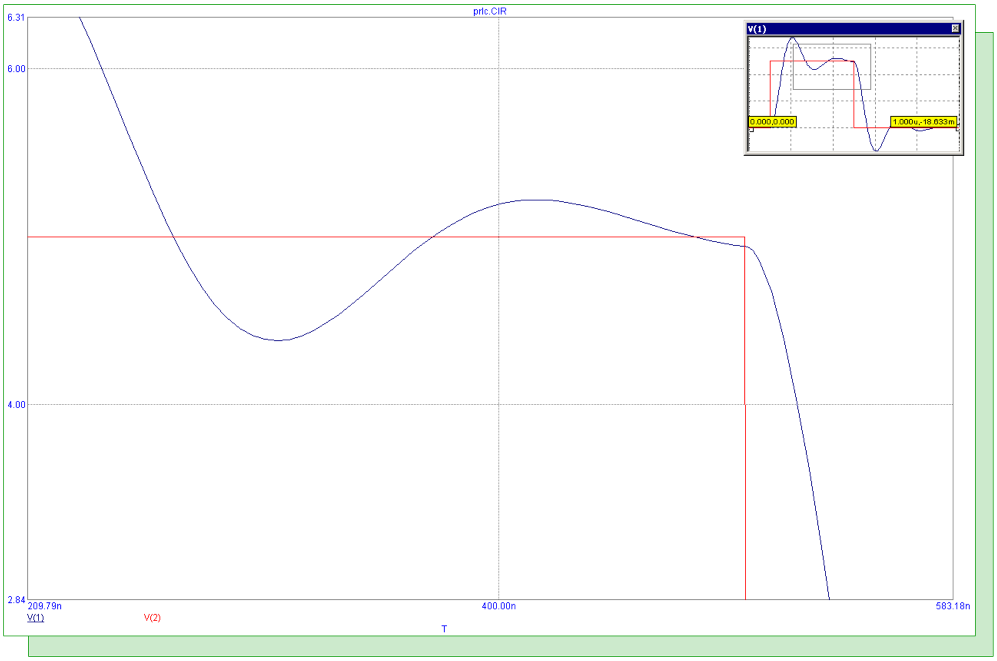


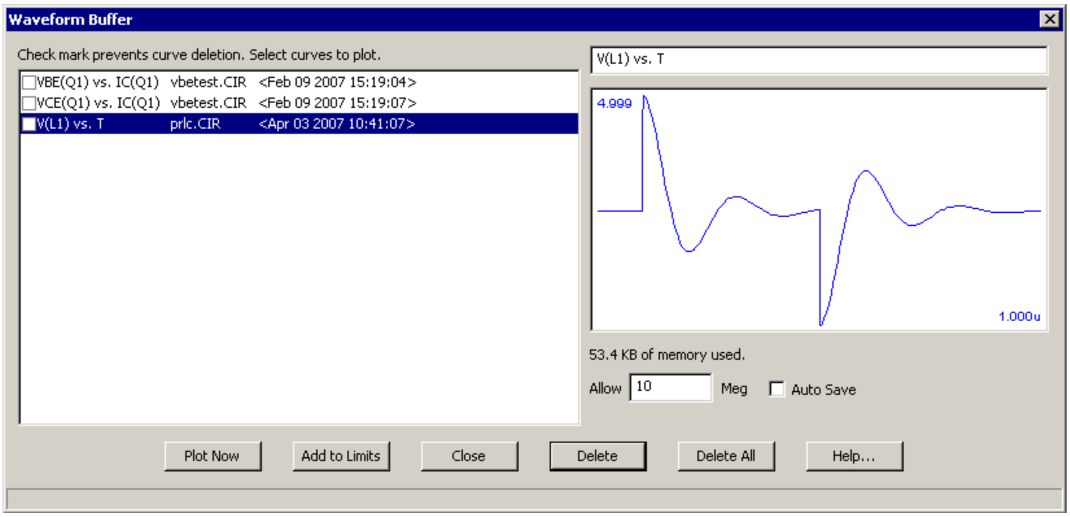
Fig. 1 - Thumb nail plot

The thumbnail plot shows only a single plot group at a time. To view a different plot group in the thumbnail, click in the plot group or an a plot expression in that group in the main analysis window. The rectangular region present in the thumbnail plot shows the area of the curve that is currently being displayed within the main plot. The thumbnail plot will also display the location of the cursors from the Cursor mode.

Using the Waveform Buffer

Micro-Cap 9 provides a feature that can easily display waveforms within an analysis plot that have been saved from previous runs or even from simulations of other circuits. The Waveform Buffer provides the capability to store and retrieve waveforms. The buffer can be used with waveforms created in transient, AC, DC, or distortion analysis.

The Waveform Buffer is available under the Scope menu when in an analysis. The dialog box appears as below:



Waveform Buffer

Check mark prevents curve deletion. Select curves to plot.

```
□VBE(Q1) vs. IC(Q1)  vbetest.CIR  <Feb 09 2007 15:19:04>
□VCE(Q1) vs. IC(Q1)  vbetest.CIR  <Feb 09 2007 15:19:07>
□V(L1) vs. T         prlc.CIR     <Apr 03 2007 10:41:07>
```

V(L1) vs. T

53.4 KB of memory used.

Allow 10 Meg □ Auto Save

Fig. 2 - Waveform Buffer dialog box

The list displays the waveform labels, the circuit name from which they came, and the date and time they were stored in the Waveform Buffer. A check box adjacent to the label protects the waveform from deletion. The plot on the right provides a view of the selected waveform. Above the plot is a field to edit the waveform label. The other interesting functions of the dialog box are as follows:

Plot Now: This immediately adds the selected waveforms to the analysis plot and places entries into the analysis limits so the waveforms are displayed in future runs also.

Add to Limits: This places entries in the analysis limits for all selected waveforms so that they will be displayed the next time the simulation is run.

Delete: This deletes the selected waveforms whose check boxes adjacent to the waveform name have not been enabled.

Delete All: This deletes all waveforms whose check boxes adjacent to the waveform name have not been enabled.

Auto Save: When enabled, the auto save will save all waveforms from every analysis run into the buffer. As you can imagine, this will use up space after a while so the Allow Meg field specifies how much memory is available to the Waveform Buffer. Once this limit is reached, waveforms will be deleted starting with the oldest to make room for the newer waveforms. Again, the check box adjacent to the curve name will protect the waveform from being deleted if it is enabled.

Waveform Buffer Examples

The waveform buffer is simple to use. For the following example, the UA709 amplifier has been set up to operate with a gain of 100. A transient simulation was then run. Once the simulation is finished, right clicking on a waveform name invokes the popup menu shown below.

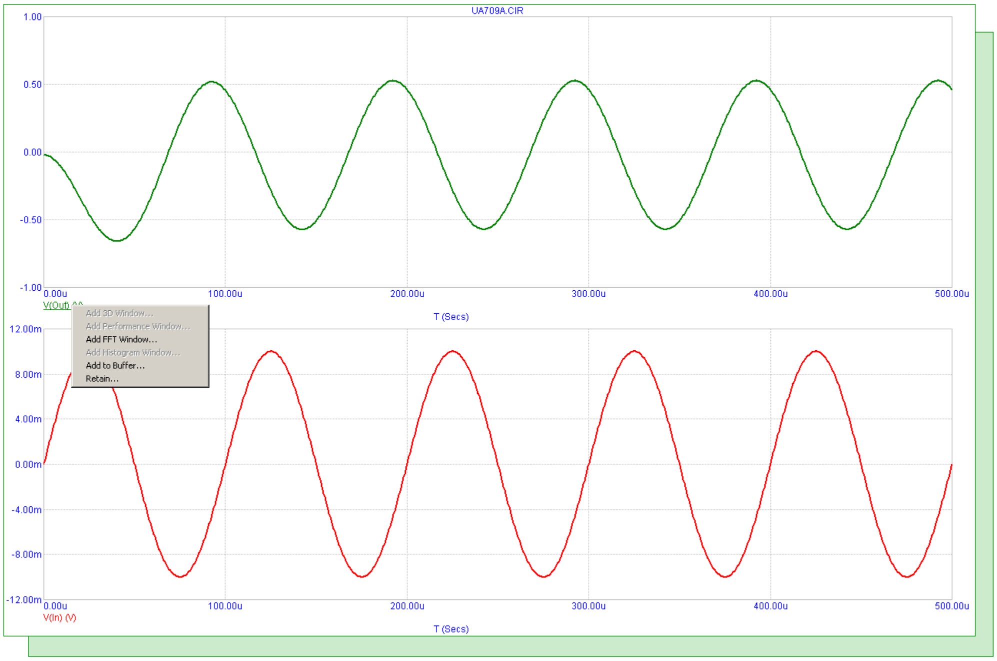
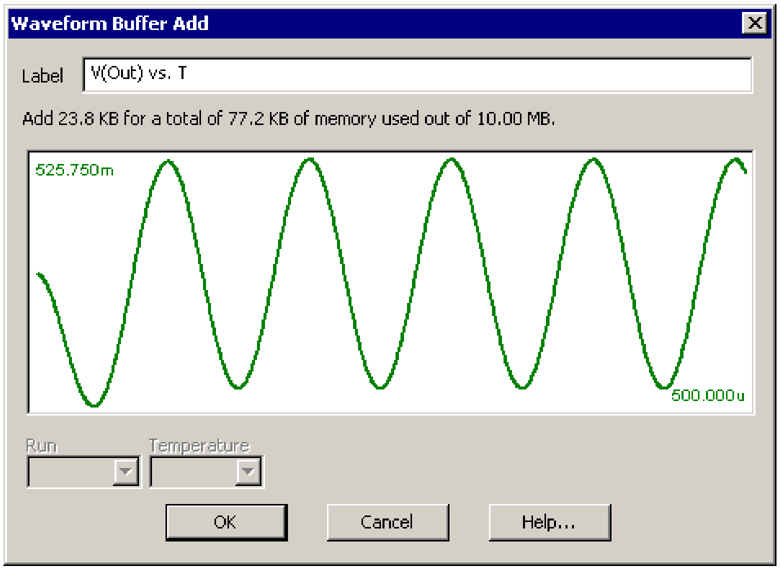


Fig. 3 - Waveform name right click menu

Two of the entries in the popup menu are applicable to the Waveform Buffer. Add to Buffer will store the selected waveform in the Waveform Buffer. Retain stores the selected waveform in the Waveform Buffer and also adds an entry in the analysis limits so that the saved waveform will be plotted in future runs. The Add to Buffer and Retain capabilities are also available under the Scope / Waveform Buffer menu. For this example, the Retain command was selected from the menu for the V(Out) waveform. The following dialog box is then invoked.


Waveform Buffer Add

Label: V(Out) vs. T

Add 23.8 KB for a total of 77.2 KB of memory used out of 10.00 MB.

Fig. 4 - Waveform Buffer Add dialog box

This dialog box lets you edit the label for how the waveform will be listed in the Waveform Buffer. If the simulation has produced multiple runs such as with Stepping, you can select which branch will be stored. In this instance, the waveform label is defined as V(Out) vs. T. Clicking OK saves the waveform in the buffer. Since Retain was selected, the following entry was also made in the analysis limits as a new Y Expression to be plotted:

Buffer("V(Out) vs. T")

This expression has been assigned to the same Plot Group as the original V(Out) waveform. Now when the transient analysis is run again, the V(Out) vs. T waveform will be retrieved from the Waveform Buffer and plotted so it can be compared to subsequent plots of V(Out).

In the schematic, the feedback resistor has been modified from 100kohm to 50kohm so that the opamp now produces a gain of 50. The resultant transient simulation in Figure 5 displays the modified V(Out) waveform versus the retained V(Out) waveform. Note that when a waveform is retrieved from the Waveform Buffer, its waveform expression in the plot is enclosed with quotation marks. It will now be easy to compare any future output waveform versus the original output waveform.

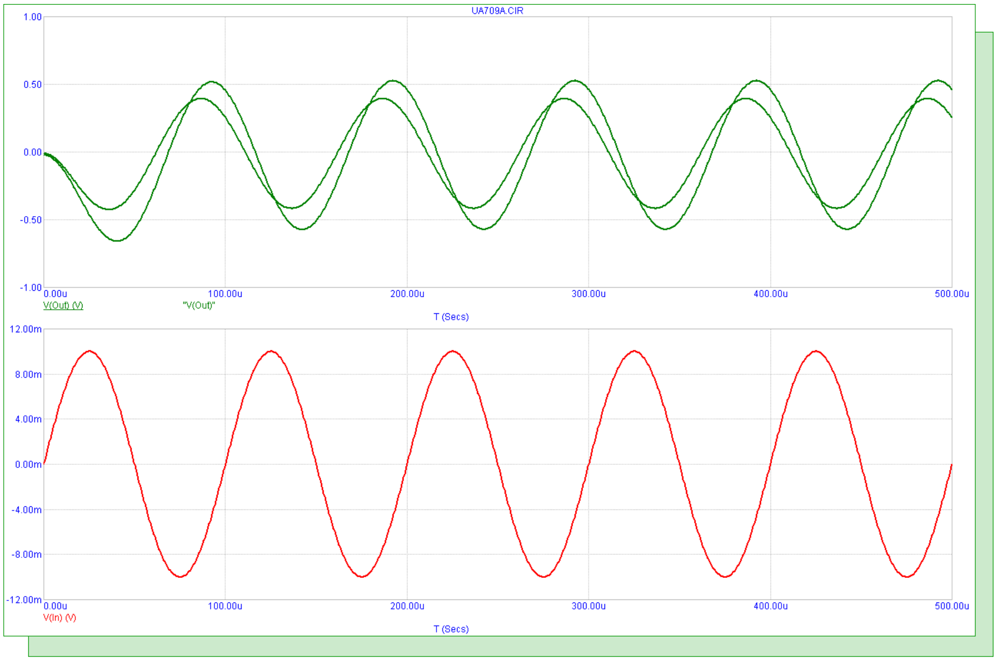
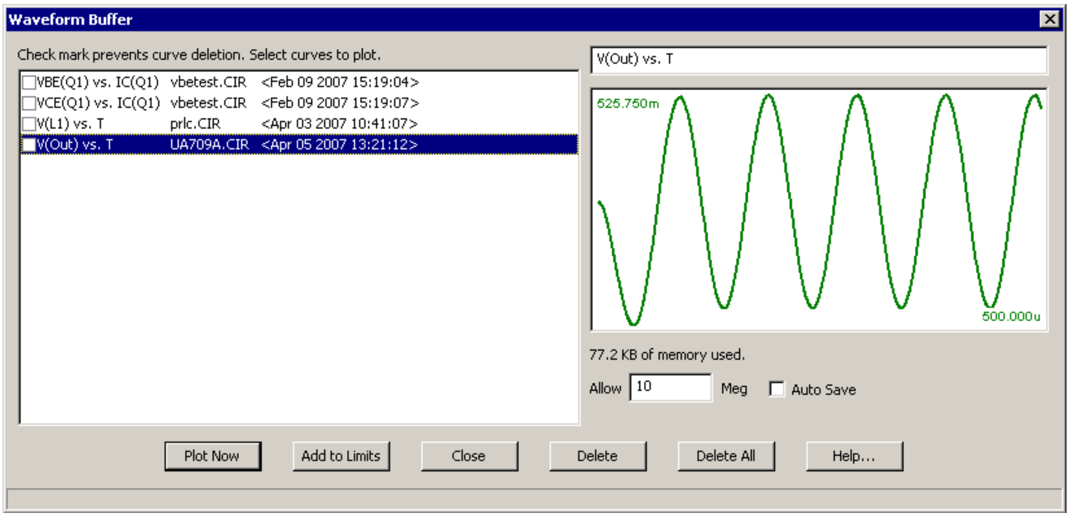


Fig. 5 - The retained V(Out) waveform

Since the V(Out) vs. T waveform is stored in the Waveform Buffer, it can be imported into any transient, AC, DC, or distortion analysis for any circuit.

For a second example of using the Waveform Buffer, the same circuit with the UA709 amplifier will be used. The feedback resistor has been restored to its original 100kohm value. In this case, an AC simulation is run, and the expression dB(V(Out)) is plotted. Once the AC analysis is finished, the Waveform Buffer is invoked. As shown in Figure 6, the V(Out) vs. T waveform previously stored from the transient analysis is now available. This waveform is selected and the Plot Now button clicked. The transient output voltage will then be plotted alongside the AC results as shown in Figure 7.



Waveform Buffer

Check mark prevents curve deletion. Select curves to plot.

□VBE(Q1) vs. IC(Q1)	vbetest.CIR	<Feb 09 2007 15:19:04>
□VCE(Q1) vs. IC(Q1)	vbetest.CIR	<Feb 09 2007 15:19:07>
□V(L1) vs. T	prlc.CIR	<Apr 03 2007 10:41:07>
□V(Out) vs. T	UA709A.CIR	<Apr 05 2007 13:21:12>

V(Out) vs. T

77.2 KB of memory used.

Allow 10 Meg □ Auto Save

Fig. 6 - Waveform Buffer dialog box with V(Out) vs T

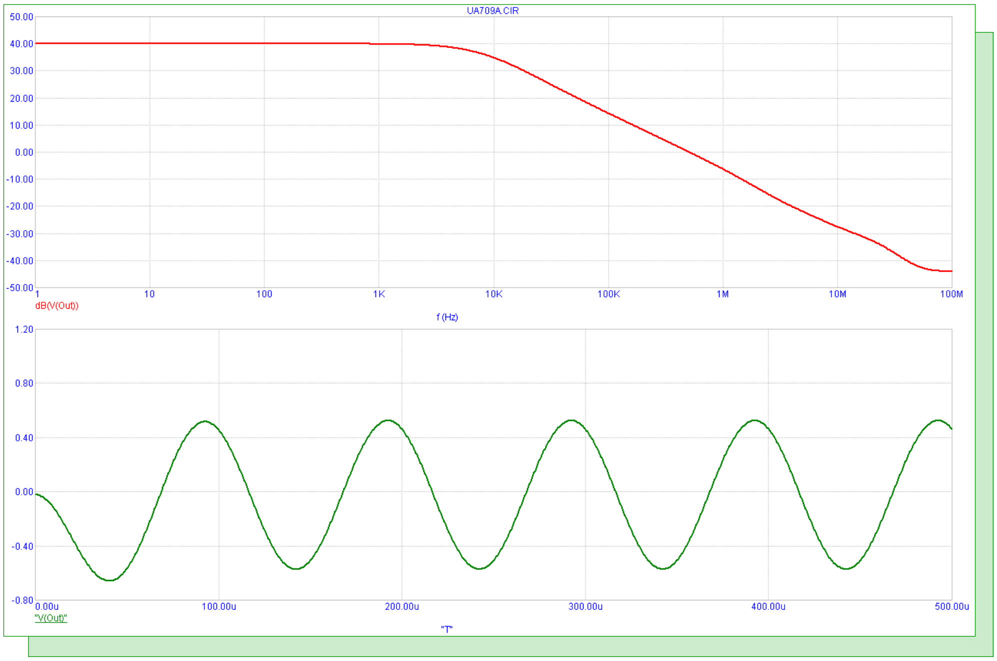


Fig. 7 - Plotting V(Out) vs T in AC analysis

When plotting a waveform from the buffer, an entry is always made in the analysis limits so that the waveform will automatically be displayed in future runs. To delete this waveform for any subsequent runs, simply delete that entry within the analysis limits.

When a waveform is stored in the buffer, the color, width, pattern, and style of that waveform are also stored. These will be the default entries for the waveform when it is initially retrieved from the buffer. Once retrieved, the waveform properties can be modified through the analysis limits or the Properties dialog box like any other waveform in the plot.

Simulating the Lorenz Attractor

The Lorenz attractor describes how the state of a nonlinear, three dimensional, dynamic system changes over time in a chaotic fashion. The attractor was originally discovered by Ed Lorenz who derived it from a simplified model of convection rolls in the earth's atmosphere. However, it also arises within lasers and dynamos.

The three differential equations that define the Lorenz attractor are:

dx/dt = Sigma*(y-x)
dy/dt = (Rho-z)*x - y
dz/dt = x*y - Beta*z

where Sigma is often referred to as the Prandtl number, Rho is often referred to as the Rayleigh number, and Beta is a geometric factor.

In Micro-Cap, differential equations such as these can be simulated by using the behavioral models available in the Macro section of the Analog Primitives. The schematic for the Lorenz attractor appears in Figure 8.

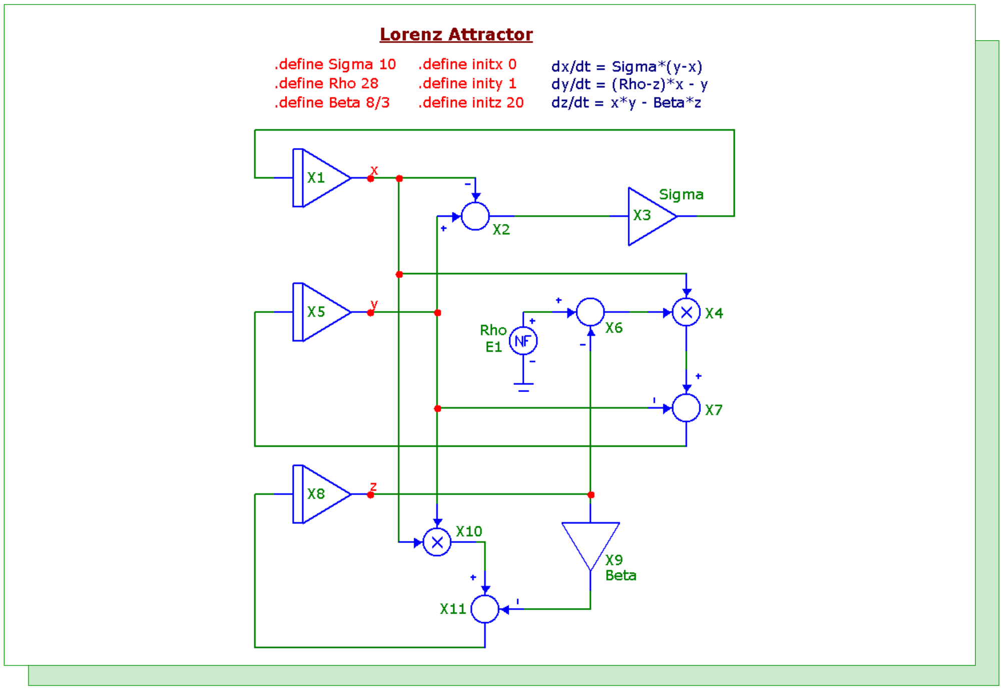


Fig. 8 - Lorenz attractor schematic

The nodes x, y, and z calculate voltages equivalent to their corresponding variables in the above equations. The x variable is calculated using an Int macro, a Sub macro, and an Amp macro. The Sub macro (X2) subtracts the voltage at node y by the voltage at node x. The difference is then fed into an Amp macro (X3) whose Gain parameter is set to Sigma. The output of the Amp macro is then input into an Int macro (X1) which calculates the integral to produce x.

The y variable is calculated using an Int macro, two Sub macros, a Mul macro, and an NFV function source. The NFV function source (E1) produces a voltage equivalent to Rho. The Sub macro (X6) subtracts the Rho value by the voltage at node z. The difference is then fed into a Mul macro (X4) where it is multiplied by the voltage at node x. This product is then fed into another Sub macro (X7) where it is subtracted by the voltage at node y. This difference is then input into an Int macro (X5) which calculates the integral to produce y.

The z variable is calculated using an Int macro, a Mul macro, an Amp macro, and a Sub macro. The voltages at nodes x and y are multiplied together using the Mul macro (X10). The voltage at node z is scaled by the factor Beta using the Amp macro (X9). These two products are input into a Sub macro (X11). The difference is then input into the Int macro (X8) which calculates the integral to produce z.

The values of Sigma, Rho, and Beta are set through define statements. The following three define statements are also present in the schematic:

```
.define InitX 0
.define InitY 1
.define InitZ 20
```

These three define statements set the initial values for the x, y, and z variables. The InitX, InitY, and InitZ are used to define the VINIT parameter for the X1, X5, and X8 Int macros respectively. The VINIT parameter of the Int macro sets an initial voltage at the output of the integrator. As with other chaotic systems, the Lorenz attractor is very sensitive to these initial conditions. Even a small change in these will produce an entirely different plot.

The common settings to show chaotic behavior with the Lorenz attractor is to set Sigma to 10, Rho to 28, and Beta to 8/3. A 200 second transient analysis is run using these values which produces the classic butterfly plot when V(Z) is plotted versus V(X) as shown in Figure 9.

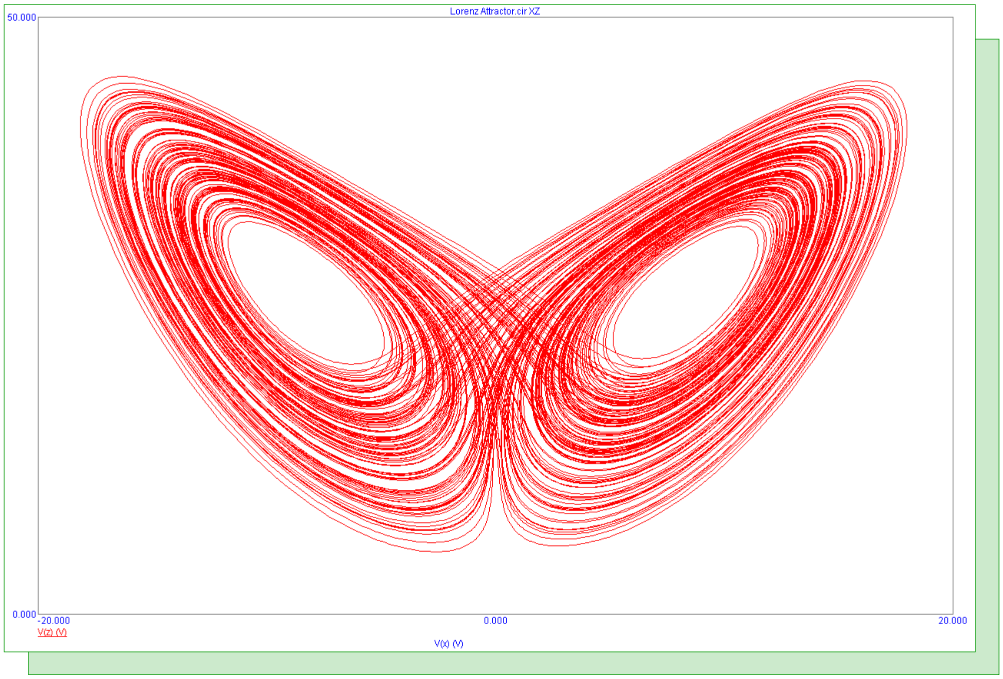
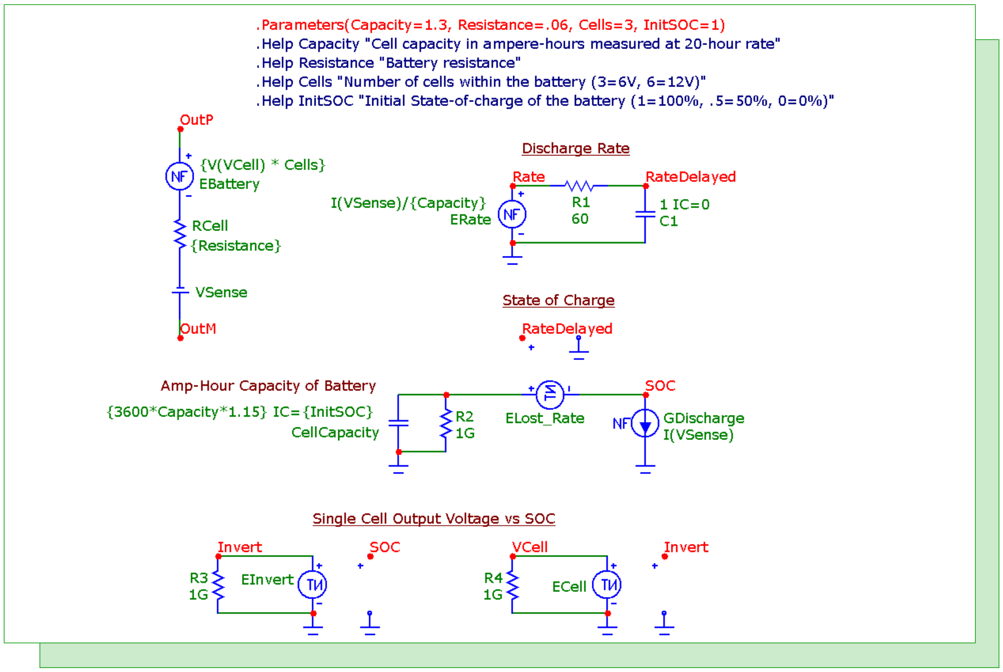


Fig. 9 - Lorenz attractor butterfly plot

Lead Acid Battery Macro

The standard battery component available in Micro-Cap is an ideal battery that provides a constant voltage over any length of time and with any current. For some simulations, a battery model may be needed that discharges over the course of the analysis. The lead acid battery macro that appears in the figure below provides a model that simulates the discharge characteristics of a battery. This model was derived from the articles "Simple PSpice models let you simulate common battery types" in the October 28, 1993 issue of EDN and "PSpice models nickel-metal-hydride cells" in the February 15, 1996 EDN Design Ideas Supplement. Both articles were written by Steven Hageman.



```
.Parameters(Capacity=1.3, Resistance=.06, Cells=3, InitSOC=1)
.Help Capacity "Cell capacity in ampere-hours measured at 20-hour rate"
.Help Resistance "Battery resistance"
.Help Cells "Number of cells within the battery (3=6V, 6=12V)"
.Help InitSOC "Initial State-of-charge of the battery (1=100%, .5=50%, 0=0%)"
```

Fig. 10 - Lead acid battery macro

The macro circuit has four parameters that are passed through to it: Capacity, Resistance, Cells, and InitSOC. The Capacity parameter defines the nominal capacity of the battery measured at the 20 hour rate. It is defined in units of Amp-hour. The Resistance parameter defines the internal resistance of the battery. The Cells parameter defines the number of cells that comprise the battery. 3 cells model a 6V battery and 6 cells model a 12V battery. Finally, the InitSOC parameter defines the initial state of charge that the battery will start the simulation with. A value of 1 means the battery is 100% charged and a value of 0 means that no capacity remains.

The discharge rate of the battery is calculated by the ERate nonlinear function voltage source (NFV). It is defined with the following equation:

{I(VSense)/Capacity}

where I(VSense) is a measurement of the current through the battery which is divided by the nominal capacity of the battery to produce the discharge rate. The voltage at node Rate is equivalent to the instantaneous discharge rate of the battery in C units. The R1 and C1 combination provides a 60 second RC time constant. This delayed voltage is then fed into the input of the table source ELost_Rate which lowers the cell capacity at high rates of discharge. The TABLE attribute of the ELost_Rate source has been defined as:

(0.05,0.0) (0.089,0.11) (0.16,0.20) (0.62,0.39) (0.8,0.47) (1.6,0.44)

At greater discharge rates, the state of charge will be lowered by the amount specified in the table. Values between data points are linearly interpolated. The CellCapacity capacitor is used in conjunction with the GDischarge current source and the ELost_Rate table source to calculate the state of charge of the battery. The CellCapacity capacitor has its CAPACITANCE attribute defined as:

```
{3600*Capacity*1.15} IC={InitSOC}
```

The capacitance is determined by the Capacity parameter muliplied by 3600 to convert the Amp-hour value to Amp-second. The 1.15 multiplier is an adjustment multiplier that helps account for the manufacturer's specified capacity versus the 0 volts this model produces at no capacity. The IC statement defines the initial state of charge through the InitSOC parameter. The GDischarge current source reproduces the current through the battery, I(VSense), in order to discharge the capacitor. Essentially, the voltage stored in the capacitor minus the voltage across the ELost_Rate source produces a voltage at node SOC that is equivalent to the state of charge of the battery.

The EInvert table source inverts the voltage at node SOC. It does this through a simple table definition of:

(0,1) (1,0)

Since the input voltage should always be between 0 and 1, this table definition is equivalent to the equation (1-V(SOC)). When the state of charge is at 1 (100%), the output of this source will be 0V. This inverted voltage is then used as the input to the ECell table source. The ECell source sets the single cell voltage value versus the inverted state of charge value. The source has its TABLE attribute set through the following define statement:

```
.define CellV
+ (0.000E+00 2.171E+00) (5.222E-04 2.149E+00) (1.828E-03 2.128E+00)
+ (1.263E-01 2.101E+00) (4.908E-01 2.001E+00) (6.385E-01 1.949E+00)
+ (7.459E-01 1.900E+00) (7.834E-01 1.875E+00) (8.117E-01 1.850E+00)
+ (8.313E-01 1.826E+00) (8.436E-01 1.801E+00) (8.517E-01 1.773E+00)
+ (8.556E-01 1.750E+00) (8.591E-01 1.724E+00) (8.616E-01 1.702E+00)
+ (8.646E-01 1.676E+00) (8.677E-01 1.648E+00) (8.707E-01 1.623E+00)
+ (8.732E-01 1.600E+00) (8.850E-01 1.499E+00) (8.965E-01 1.401E+00)
+ (9.000E-01 1.333E+00) (1.000E+00 0.000E+00)
```

When the state of charge is 100%, the input voltage to the table is 0V so that the output voltage of the table source would then be 2.171V. Similarly, if the state of charge drops to 51%, the input voltage to the table would be .49V so that the output voltage of the table source would be approximately 2V.

The EBattery NFV source, RCell resistor, and VSense battery model the actual battery device. The VSense battery, previously referenced, is a 0V battery used to measure the current through the battery. The RCell resistor models the internal resistance of the battery and is defined by the Resistance parameter. Finally, the EBattery source models the output voltage of the battery. Its VALUE attribute is defined as:

```
{V(VCell) * Cells}
```

The equation multiplies the voltage at node VCell by the number of Cells that are passed through as a parameter to the macro to produce the battery voltage.

Obviously, the table values for ELost_Rate and ECell along with the multiplier in the CellCapacity capacitor may need to be modified when simulating specific batteries but this should provide a good general model of the lead acid battery.

A simple circuit was created to test the macro. The battery macro is discharged by a constant current source of 50mA. The parameters of the macro have been defined as follows:

Capacity=1.3
Resistance=.12
Cells=6
InitSOC=1

The resultant transient analysis is shown in Figure 11. The simulation was run with a time range of 72000 seconds or 20 hours. The X axis uses the variable Hours which was created with the following define statement:

.Define Hours T/3600

The top plot displays the output voltage of the battery and shows the voltage decaying down to 11.6V by the end of the simulation. The bottom plot shows the state of charge of the battery. It starts at 1 which is the value set by InitSOC, and at the end of 20 hours, the state of charge is down to around .33V or 33%.

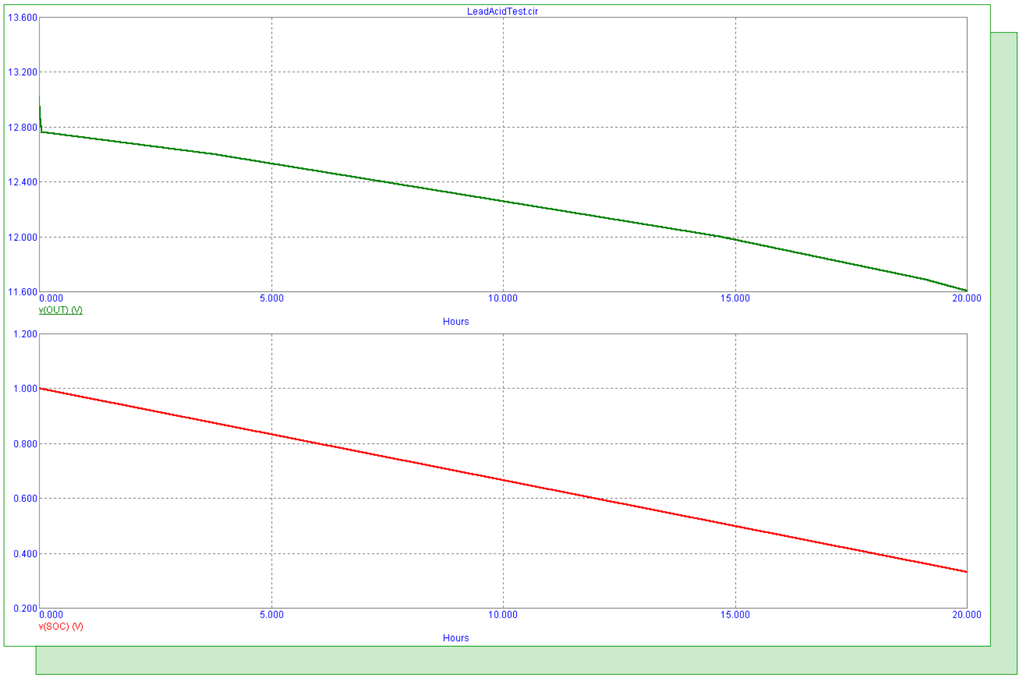


Fig. 11 - Battery macro test simulation

Product Sheet

Latest Version numbers

Micro-Cap 9	Version 9.0.1
Micro-Cap 8	Version 8.1.3
Micro-Cap 7	Version 7.2.4

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