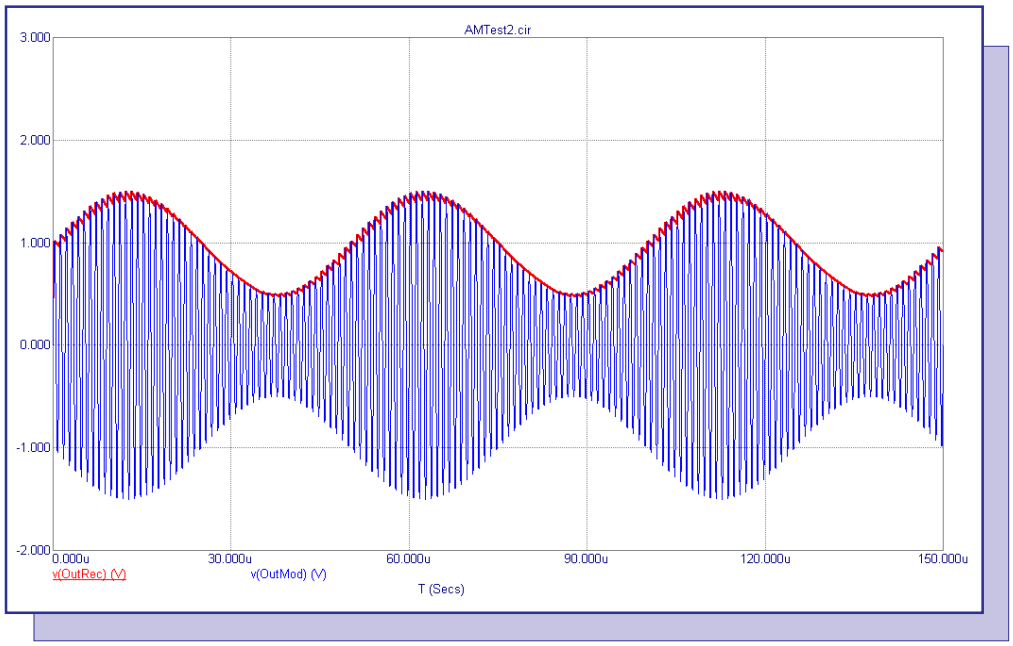
spectrum news

Applications for Micro-Cap™ Users

Winter 2005

News



Amplitude Modulator Macro

Featuring:

- Using Global Nodes
- Amplitude Modulator Macro
- Optimizing a Core Model from a Data Sheet

News In Preview

This newsletter's Q and A section describes how to plot circuit variables that are within a subcircuit or macro. The Easily Overlooked Features section describes how the cursor tracker in an analysis will display the values of stepped parameters when stepping has been enabled.

The first article describes the use of global nodes in a schematic with an example in how to easily change a digital library component's power supply.

The second article describes the creation and operation of an amplitude modulator macro which can be used in numerous communications applications.

The third article describes the process of optimizing model parameters of a core model from a data sheet B-H curve through the Model feature available in Micro-Cap.

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 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: Is it possible to plot the voltage of a node that lies within a macro during the simulation of the calling circuit?

Answer: All of the standard circuit variables within a macro or a subcircuit are available for plotting during a simulation. To reference a node name or a part name of an object within a macro or subcircuit, dot notation needs to be used as in the following format:

Subcircuit\Macro Part Name + "." + Node\Part Name

For example, to reference the voltage of node 6 in the macro X3, the following expression would be used:

V(X3.6)

Similarly, the expression:

I(X6.R1)

would plot the current through the resistor R1 that is within the macro or subcircuit X6.

The dot notation may be used to plot circuit variables that are nested within multiple layers of macros or subcircuits by concatenating the macro or subcircuit names. The expression:

V(X1.X2.X3.10)

would plot the voltage of node 10 in the X3 subcircuit, within the X2 subcircuit, within the X1 subcircuit.

The Variables list which is invoked by right clicking in a Y Expression field of the Analysis limits displays a menu which shows all of the circuit variables available for plotting. Subcircuit and macro variables will also be displayed in this list if they are present in the circuit.
.

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.

Stepping Information in the Cursor Tracker
When a simulation uses stepping, multiple branches of each waveform will appear in the plot. Sometimes it can be difficult to determine which branch belongs to which value of the stepped parameter. A new feature has been introduced to the cursor trackers in Micro-Cap 8.0.7 to enable an easier reading of the stepped values for each branch of the waveform through the cursors.

For a stepped waveform, when Cursor mode is enabled, the cursors can be transferred from branch to branch through the Up and Down Arrow keys. If the Title of the plot is set to Auto, then the stepped values of the branch that the most recently moved cursor is on will be displayed in the plot title. This branch information is now also available within the cursor trackers.

The cursor trackers have the ability to display both the X, Y coordinates at its present point and the stepped values of the current branch. These two options can be enabled under the Trackers option in the Scope menu. The two options are:

Cursor
Cursor: Branch Info

If Cursor is enabled, the X, Y coordinates will be displayed, and if Cursor: Branch Info is enabled, then the stepped values of the current branch will be displayed. Both, either, or neither of these options may be enabled. The figure below displays the use of the cursor trackers when only the Cursor: Branch Info option has been enabled.

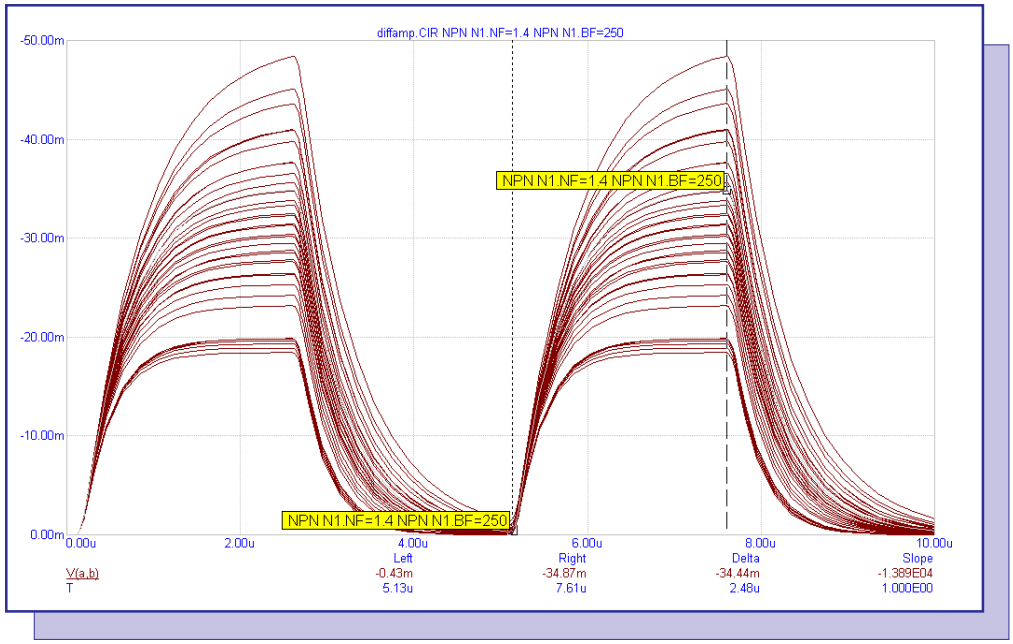


Fig. 1 - Cursor Tracker with Branch Information

Using Global Nodes

Global nodes are nodes whose names are globally available to all levels of the circuit, from the top level through to any macros or subcircuits used in the circuit. Using a global node name will tie all nodes together that share the same name whether they are in the main circuit or in a subcircuit three levels down. This feature provides a powerful method to assign common characteristics such as power supplies throughout a multilevel schematic. Global nodes are designated by prefacing the node name with $G_ in the following format:

$G_Name

In the Micro-Cap libraries, global nodes are used extensively in the modelling of the digital components to define the circuitry at the power supply pins of each component. In the digital interface library (DIGIO.LIB), the main power supplies for the digital families are defined within subcircuits such as:

```
.subckt DIGIFPWR AGND
+  optional: DPWR=$G_DPWR DGND=$G_DGND
+   params: VOLTAGE=5 REFERENCE=0
V1 DPWR AGND {VOLTAGE}
R1 DPWR AGND 1E9
V2 DGND AGND {REFERENCE}
R2 DGND AGND 1E9
R3 AGND 0 1m
.ends
```

This subcircuit is used as the power supply for the 74/TTL family of components that defines the 5 volt power supply and the ground reference for the family. The DPWR and DGND nodes used within the subcircuit have their default external connections assigned through the OPTIONAL: statement in the subcircuit header which connects these nodes to global nodes with the names $G_DPWR and $G_DGND. Each component whose interface model specifies the DIGIFPWR subcircuit above will also use the $G_DPWR and $G_DGND nodes as their default power supplies. Consider the 7400 2-input Nand gate model shown below:

```
.SUBCKT 7400  1A 1B 1Y
+    optional: DPWR=$G_DPWR DGND=$G_DGND
+     params: MNTYMXDLY=0 IO_LEVEL=0

U1 nand(2) DPWR DGND
+     1A 1B 1Y
+     DLY_00 IO_STD MNTYMXDLY={MNTYMXDLY} IO_LEVEL={IO_LEVEL}

.model DLY_00 ugate (tplhTY=11ns tplhMX=22ns tphlTY=7ns tphlMX=15ns)

.ENDS  7400
```

The OPTIONAL: statement within the subcircuit header of the 7400 assigns the DPWR and DGND nodes used within the subcircuit to the same global nodes defined in the power supply circuit. Note that the U1 nand gate specified in the subcircuit uses the DPWR and DGND nodes as its power supply references.

Using an On Schematic Power Supply with a Digital Library Part

A Winter 2000 newsletter article described a process by which a digital library component could access a power supply that exists on the schematic. Using the global node capability provides a simpler technique for accomplishing the same task.

The circuit in Figure 2 simulates a basic 4-bit digital adder. Each of the digital gates in the adder is taken from the CD4000 family which is capable of handling a wide range of power supply voltages. In the digital library files, the CD4000 series of parts will by default access the $G_CD4000_VDD and $G_CD4000_VSS global nodes for their power supplies as seen in the OPTIONAL: statement within the subcircuit header for the CD4002B component below:

```
.SUBCKT CD4002B IN1A IN2A IN3A IN4A OUTA
+  optional: VDD=$G_CD4000_VDD VSS=$G_CD4000_VSS
+  params: MNTYMXDLY=0 IO_LEVEL=0
```

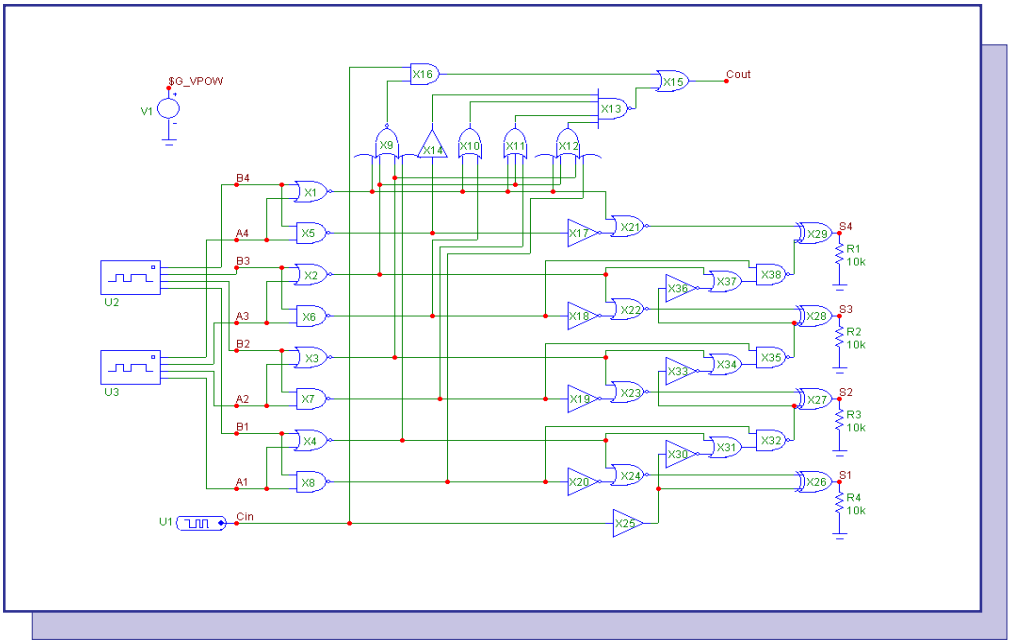


Fig. 2 - Four Bit Adder Circuit

To modify these models to use an on schematic power supply, the actual subcircuit header for the models used in the schematic will need to be edited. Rather than editing the library file which would affect any other schematic that references the library, the models can be localized in the schematic file by selecting the Refresh Models command under the Edit menu. In the Refresh Models dialog box, enable the Subcircuits checkbox for the Type field as that is the only model type of interest in this circuit. Enable Add for the Action to use, and then click OK. All of the subcircuits used will now have their models stored in the Models text page of the schematic. For each of the CD4000 subcircuits, the VDD node is the power supply node. The VDD node assignment in the OPTIONAL: section of the header will then be assigned to a new global node name such as:

```
.SUBCKT CD4002B IN1A IN2A IN3A IN4A OUTA
+  optional: VDD=$G_VPOW VSS=$G_CD4000_VSS
+  params: MNTYMXDLY=0 IO_LEVEL=0
```

With the above, the VDD node will now be connected to a global node called $G_VPOW. In the schematic, the power supply for these gates is defined by the V1 pulse source in the upper left. The pulse source is specified to start at 3V and then rise up to 15V after 2ms at which level it will stay for the rest of the simulation. The node that the pulse source is connected to has been assigned the global node name, $G_VPOW. The pulse source will now be connected to the power node of the subcircuit. The VSS node within the subcircuit will continue to use the ground reference defined within the DIGIO.LIB file.

The three digital stimulus sources in the schematic have all had their POWER NODE attribute assigned to the $G_VPOW global node so that they will also use the pulse source as the power supply. However, since no analog components are connected to the stimulus sources, the power and ground nodes are never accessed for this simulation by these sources. The Stim1 source at the Cin input produces a constant 0 output. The two Stim4 sources at the A and B inputs have had their COMMAND attributes specified to produce random strings of data. They both use define statements similar to:

```
.define AIN
+0ns Rnd
+label=start
+.15m ?
+.3m goto start -1 times
```

where a random Hex state is calculated every 150us. Both the Rnd expression and the ? character can be used to declare a random variable in a digital stimulus statement.

The 10k resistors at the S outputs of the adder were placed in the schematic so that the output digital states of the adder would be converted into their equivalent voltage in order to view the effect that the schematic power supply has on the digital components. The digital power supplies are only used when an AtoD or DtoA interface is needed within the schematic.

The resulting transient analysis of the adder is displayed in Figure 3. The top plot displays the digital states at the inputs and outputs of the adder. The D(Cin) and D(Cout) waveforms show the binary response for the carry in and carry out nodes of the adder. The top three waveforms in the plot display the hex results for the A and B input and the S output states. Note that the hex results for the S outputs reference the internal digital node names at the outputs with the expression:

```
Hex(41$DTOA,39$DTOA,37$DTOA,35$DTOA)
```

The internal digital node names need to be used because with the presence of analog components at these nodes, any reference to S1, S2, S3, or S4 will assume an analog voltage which is not applicable with the Hex operator.

The bottom plot shows the analog voltages at the S outputs along with the voltage of the power supply. The power supply voltage is delineated by the thick black plot. The CD4000 series of components produce output voltages very close to the power supply rail. Note how the S outputs rise to whatever the current voltage of the pulse source power supply is.

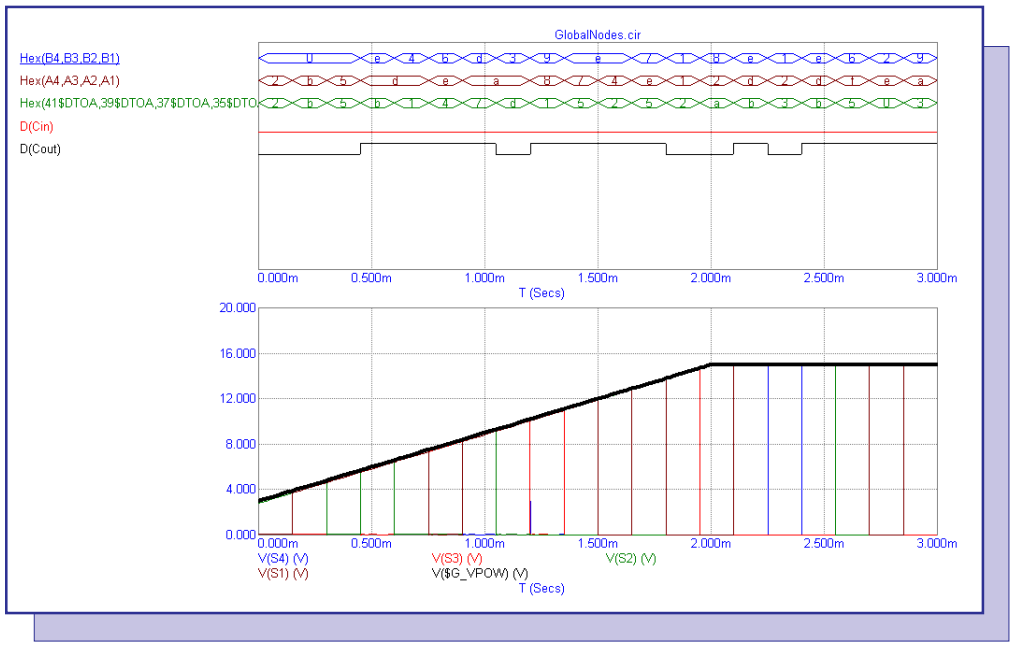


Fig. 3 - 4-Bit Adder Simulation

One limitation of the digital/analog interface models in the DIGIO.LIB file is that they are only specified to work over limited operating voltages. For example, the CD4000 series can operate with power supplies from 3V to 15V. Others, such as the standard TTL models, are only designed to work at 5V. Operating these components outside of their range of acceptable power supplies would require the creation of a new I/O model for the family to produce valid results.

While the example in this article referenced digital circuitry, the use of global nodes is valid for analog components employing the same technique. This method can be used as a way to define the power supply of opamps or the common substrate connection of BJTs through multiple levels of circuitry. Any user who needs a common connection across multiple levels will find the global nodes a useful tool.

Amplitude Modulator Macro

Amplitude modulation plays a key role in many types of communication systems, the most obvious one being AM radio. In amplitude modulation, a message signal is shifted to a high frequency carrier signal whose frequency coincides with the necessary transmission requirements. The amplitude of this carrier signal is modulated depending on the strength of the message signal.

The usual amplitude modulation is double sideband modulation. In this type of modulation, a DC signal is added to the message signal prior to using it to modulate the carrier signal. Double sideband modulation typically carries the form of:

$$Vs(t) = Vo*(1 + m*Vm(t))$$

where Vo is the DC offset, m is the modulation index, and Vm(t) is the message signal. The modulation index should be set to a value in the range from zero to one where m=0 represents no modulation and m=1 represents a maximally modulated signal. This modified message signal is then multiplied by the carrier signal to get the final modulated signal such as:

$$Vmod(t) = Vs(t) * Vc(t)$$

where Vc(t) is the carrier signal which is typically a sinusoidal waveform.

The macro model for the amplitude modulator is shown in Figure 4. The macro circuit consists of only a nonlinear function voltage source, E1, which performs the mathematical computations of the modulation. There are five input parameters for this macro: FS, VPeak, ModIndex, Offset, and Type. The FS parameter defines the frequency of the sinusoidal carrier signal. The VPeak parameter sets the peak amplitude of the carrier signal. The ModIndex parameter defines the modulation index, and the Offset parameter sets the offset value that is added to the input signal. These parameters correspond to m and Vo, respectively, in the Vs(t) equation above. The Type parameter determines whether the carrier signal will be a sine or a cosine function.

The most interesting aspect of the macro is the use of the Type parameter. The VALUE attribute of the E1 nonlinear function voltage source has been defined as Type. In the circuit calling the macro, the Type attribute must be defined as either SM for sine modulation or CM for cosine modulation. When an analysis is invoked, either SM or CM will then be substituted into the VALUE attribute of E1 in place of the Type declaration. Define statements within the macro circuit set the equations that are assigned to SM and CM as follows:

```
.define SM Offset*(1+ModIndex*V(In))*VPeak*Sin(2*PI*FS*t)
.define CM Offset*(1+ModIndex*V(In))*VPeak*Cos(2*PI*FS*t)
```

These equations determine the actual modulation expression that will be used. The only difference between the two expressions is that the SM variable uses a sine wave as the carrier waveform, and the CM variable uses a cosine wave. Setting the Type variable to any value besides SM or CM in the calling circuit will produce an error. However, this method is easily expandable if the function source is capable of handling the new expression. For example, if a basic amplitude modulation was needed, a new define statement could be added to the macro such as:

```
.define BM V(In)*VPeak*Sin(2*PI*FS*t)
```

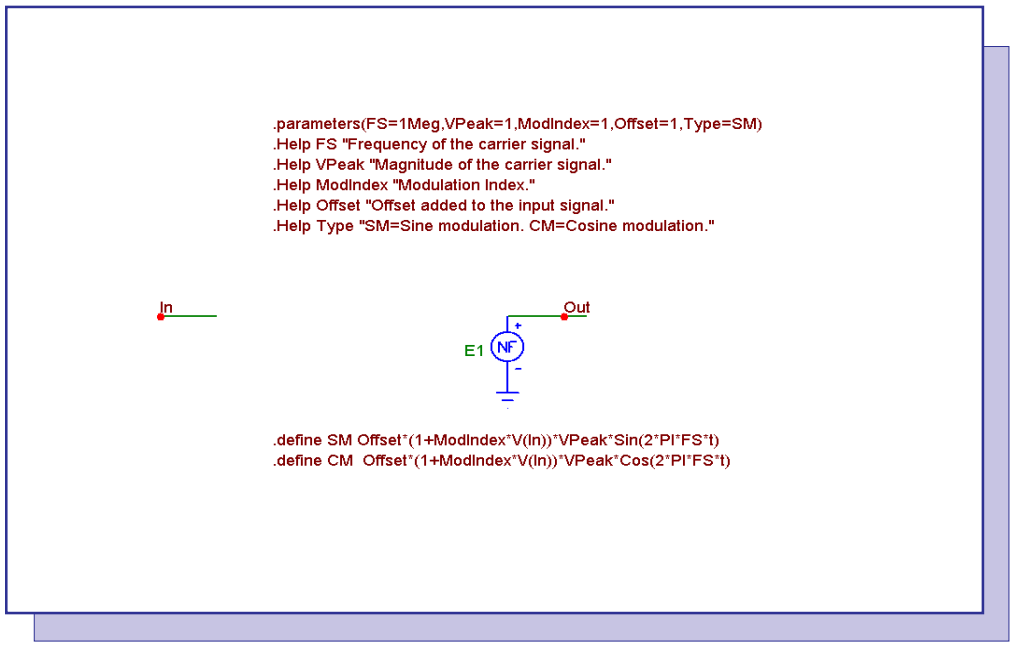


Fig. 4 - Amplitude Modulator Macro

Then BM would become a valid setting for the Type parameter and would invoke the specified expression.

An example circuit using the amplitude modulator macro is displayed in Figure 5. The input to the macro is a 1V, 20kHz sine wave. The macro has its parameters defined as:

FS = 1Meg
VPeak = 1
ModIndex = .5
Offset = 1
Type = SM

The carrier signal is defined as a 1V, 1MHz sine wave. The input signal will be offset by 1V and will use a modulation index of .5. The output of the modulator is then sent to a simplified AM receiver which is essentially acting as a peak detector.

The resulting transient analysis is shown in Figure 6. The waveform V(OutMod) is the signal at the output of the modulator. The information in the modulated signal is carried in the envelope of the waveform. The waveform V(OutRec) is the signal at the output of the receiver. With the simple AM receiver, there is some distortion with the carrier ripple on the rising edge of the waveform, but it is a good overall representation of the adjusted sine wave that was input to the modulator which shows why the peak detector method of retrieval is widely used. More elaborate receivers would of course have tuners, filters, level shifters, and gain stages to more fully reproduce the input signal.

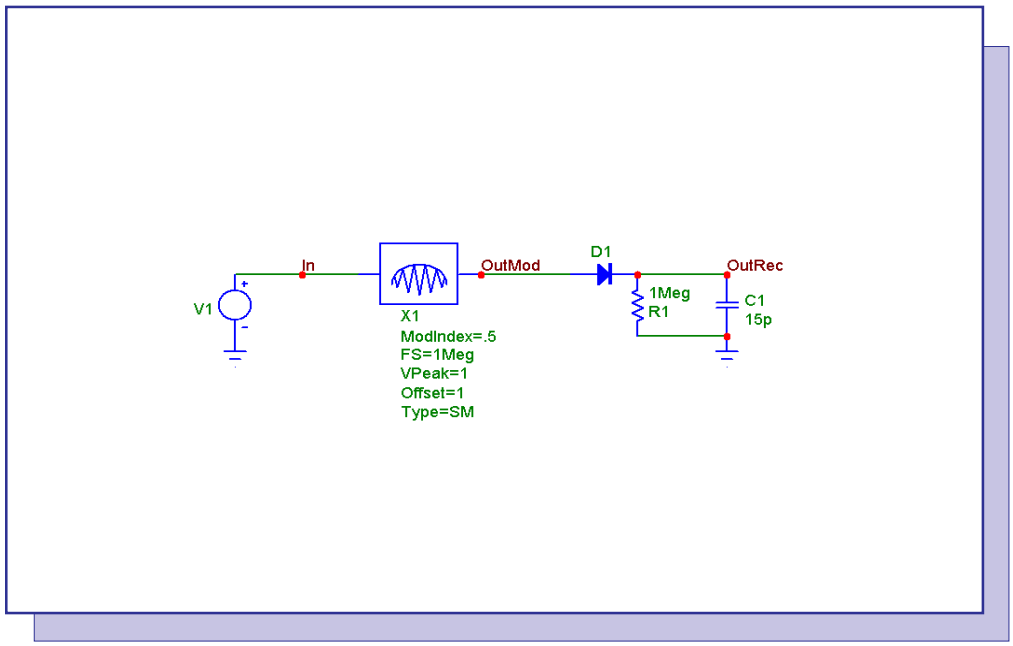


Fig. 5 - Amplitude Modulator Example Circuit

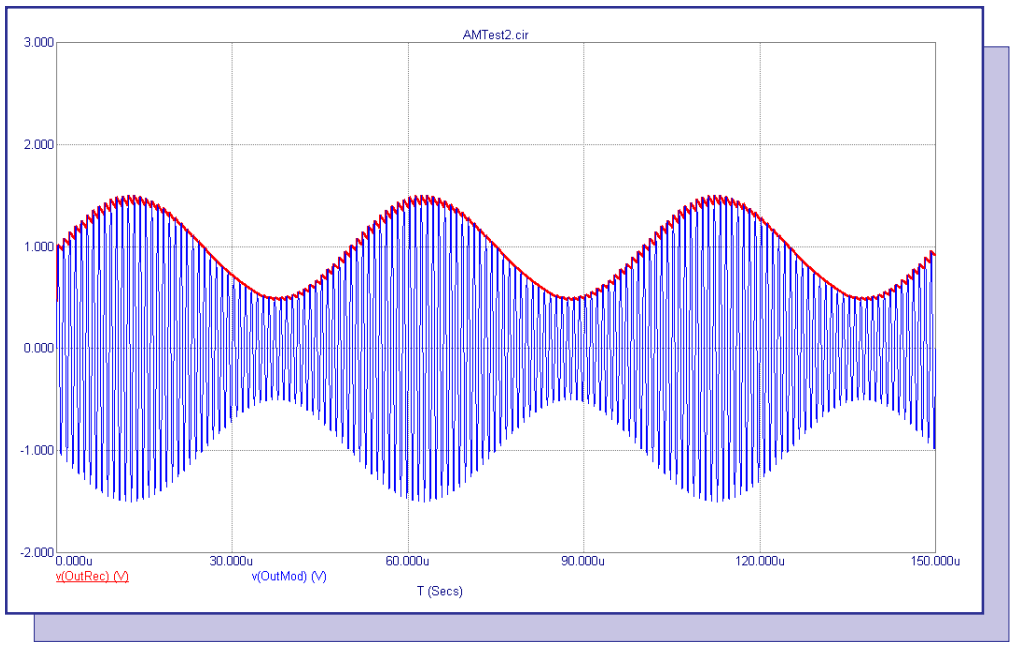


Fig. 6 - Amplitude Modulator Simulation

Optimizing a Core Model from a Data Sheet

The Jiles-Atherton core model used within Micro-Cap is a model based upon contemporary theories of domain wall bending and translation. There are seven parameters that make up this model. Four of them determine the shape of the B-H curve, and three are based on physical measurements of the core. The four curve fitting parameters need to be optimized in order to produce a reasonable B-H curve for a specific core material. While a manual process of iterating the core model parameters is possible, this can be quite tedious. The simplest method to produce a core model is to use the Model program capabilities that come with Micro-Cap.

The Model feature is designed as an interactive, optimizing curve fitter that can take numbers from data sheets and output model parameters for simulation. The Model programming capability in Micro-Cap is invoked by either creating a new MDL file or by opening an existing MDL file. This article is going to go through the process of creating a core model of the T35 MnZn core material available from Epcos. The B-H curve at an operating frequency of 10kHz and a temperature of 25C taken from the T35 data sheet is shown in Figure 7.

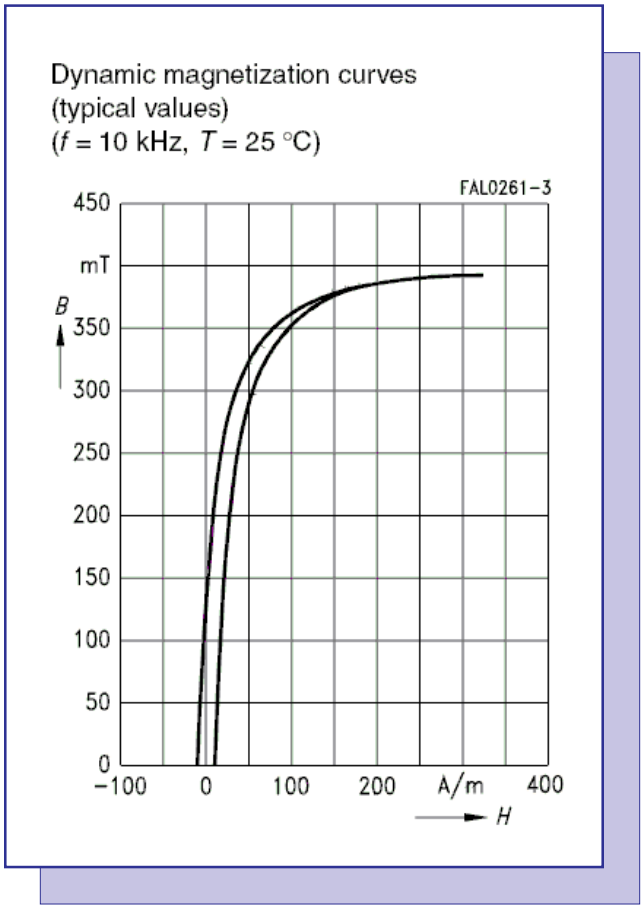


Fig. 7 - B-H Curve for the T35 Material

Once the B-H curve information is available, the first step is to launch the Model feature within Micro-Cap. Under the File menu, select the New option. Change the type to MDL and click OK. The model may also be added to an existing MDL file. In this case, use the Open command to open the desired MDL file. Next click on the Model / Add Part / Core menu sequence. A new core entry will be created in the file that will be used to input the T35 information.

Four text fields in the upper left of the screen, T1 through T4, are available for each part. The T1 field holds the part name. The T3 field holds the memo description of the part. The T2 and T4 fields are comment fields used only within the MDL file. For the T35 model, the fields have been defined as follows:

T1: T35
T2: Source - EPCOS SIFERRIT PDF Datasheet
T3: Core MnZn Material
T4: Spectrum Software 1/12/2005

In the Epcos data sheet, the B-H curve has its units defined in terms of Teslas versus A/m. In the Model file, the core initially has its units defined in terms of Gauss versus Oersted. To toggle between these core unit specifications, select the Change Core Units option under the Model menu. Once the core units match the units used in the data sheet, it is time to input data from the B-H curve.

The core takes data triplets as the input. The first two fields are the H value and its corresponding B value. The third field in the triplet is one called Region. The Region will take a value of 1, 2, or 3. A value of 1 designates the B-H data as being part of the initial permeability curve. A value of 2 designates the data as part of the top B-H curve, and a value of 3 designates the data as part of the bottom B-H curve.

Typically, when estimating data points from a B-H curve, data should be taken from the top half (B>0) or the right half (H>0) of the curve. Model will create the missing half as a mirror image. Since the Epcos data sheet only displays the top half, the choice in this case is easy. The following data triplets have been estimated from the data sheet and entered into the Model file:

H	B	Region
0	0	1
200	.385	2
35	.3	2
8	.2	2
-10	0	2
10	0	3
28	.2	3
55	.3	3
200	.385	3

Since the initial permeability curve was missing in the Epcos data sheet, only a single triplet was entered for Region 1 where both the H and B values are at 0. Both Region 2 and Region 3 have been represented with four data points each. Typically, each region should contain around four to six data points. With too few data points, the Model optimizer may not have a good enough sample. With too many data points, the Model optimizer may not optimize as closely to the data sheet B-H curve. Although seemingly counterintuitive, this last point is due to the fact that each data point is weighted equally in importance. The Model optimizer tries to find the curve that

produces the least amount of error in relation to all of the data so it may create a greater error margin in a crucial area of the curve in order to tighten up the error margin in a less important area of the curve.

Once all of the data triplets have been entered, the Area, Path, and Gap parameters can be manually set in the Model Parameters section of the screen. Since the T35 is a core material rather than an actual core component with a geometry, these parameters will be left at their default values in this case. The four curve fitting parameters can now be calculated. Under the Model menu, the Optimize command initiates the optimization. The final results of the Model optimization for the T35 model appear in Figure 8.

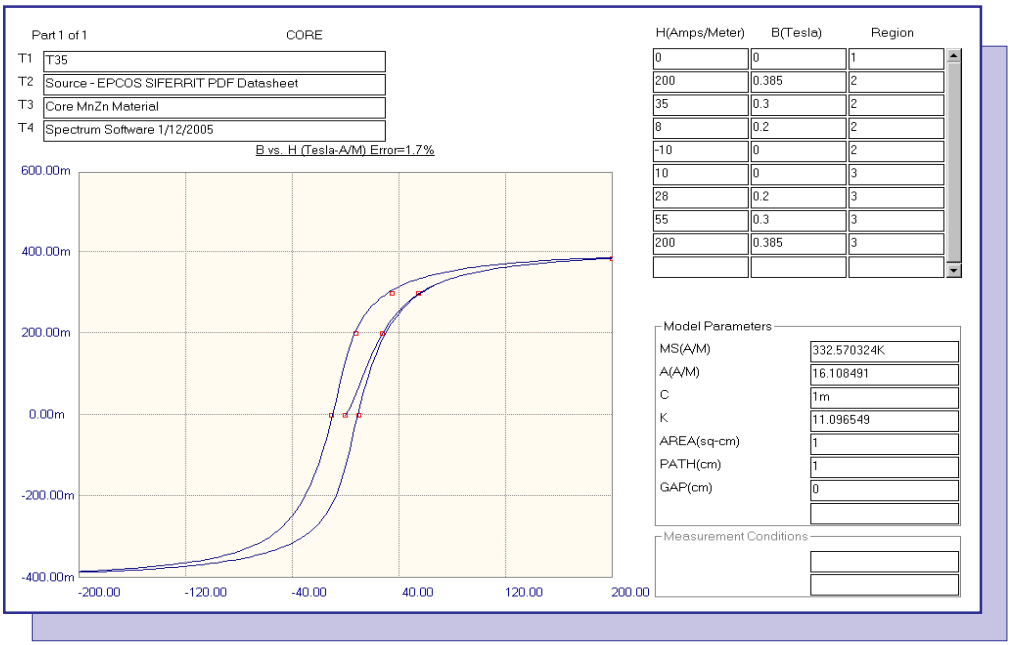


Fig. 8 - T35 Core Material Model Optimization Results

While the T35 optimization process is complete, this model is currently still available only in the MDL file. The next step is to make it accessible to the rest of Micro-Cap by creating a SPICE library file and adding the model to the Component library. This process is done by simply selecting the Add These Parts to the Component Library command under the Model menu. The dialog box that is invoked is shown in Figure 9.

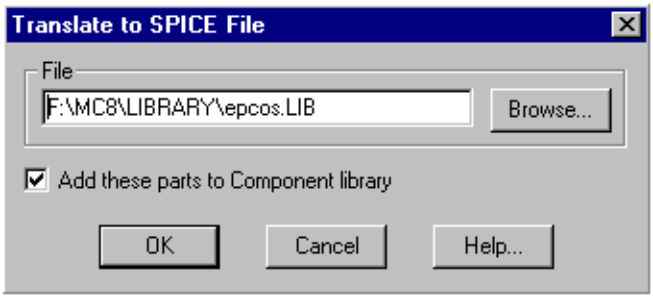


Fig. 9 - Add Parts to Component Library Dialog Box

The File field defines the name and location of the SPICE library to be created. The Add these Parts to the Component Library checkbox will add all of the parts in the MDL file to the Micro-Cap Component library. Generally, this checkbox should be enabled. If disabled, this command will only create the corresponding SPICE library file. Click OK to complete the process. The newly created SPICE library will now be displayed. Both this library and the MDL file may now be closed.

The T35 component is now available for placement in a schematic. Since the MDL file was called Epcos.mdl, the T35 component was placed in a group called Epcos on the main level of the Component menu.

The transient analysis displayed in Figure 10 shows the B-H curve of the T35 core model produced during a Micro-Cap simulation. Note that the B and H values of the curve have been plotted using the BSI and HSI operators. These operators produce the B and H values in terms of Teslas and A/m whereas the standard B and H operators will produce Gauss and Oersted.

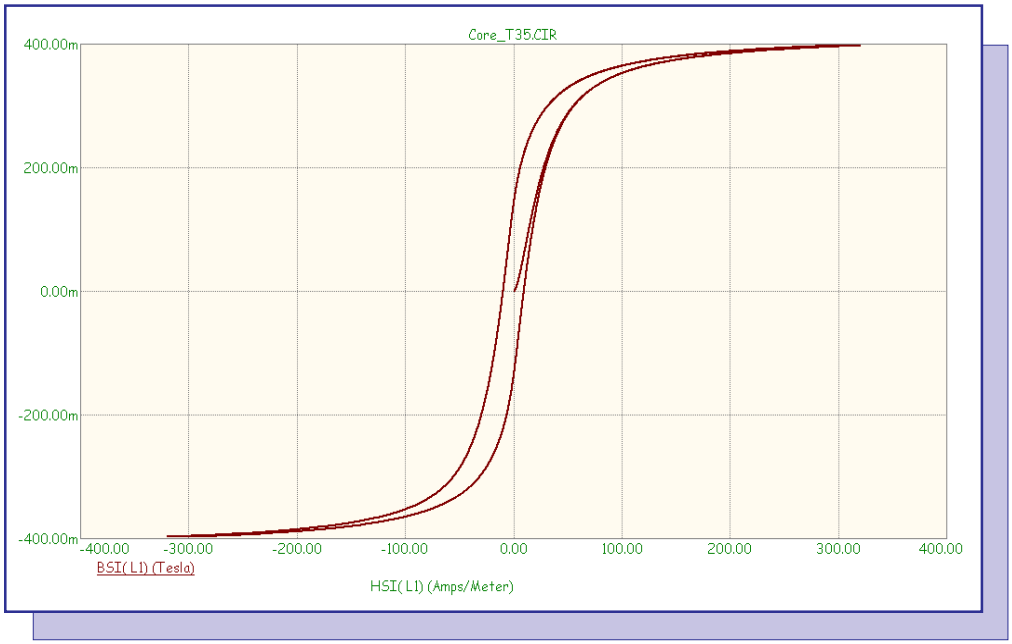


Fig. 10 - T35 B-H Curve Simulation

Product Sheet

Latest Version numbers

Micro-Cap 8 ..... Version 8.0.7
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