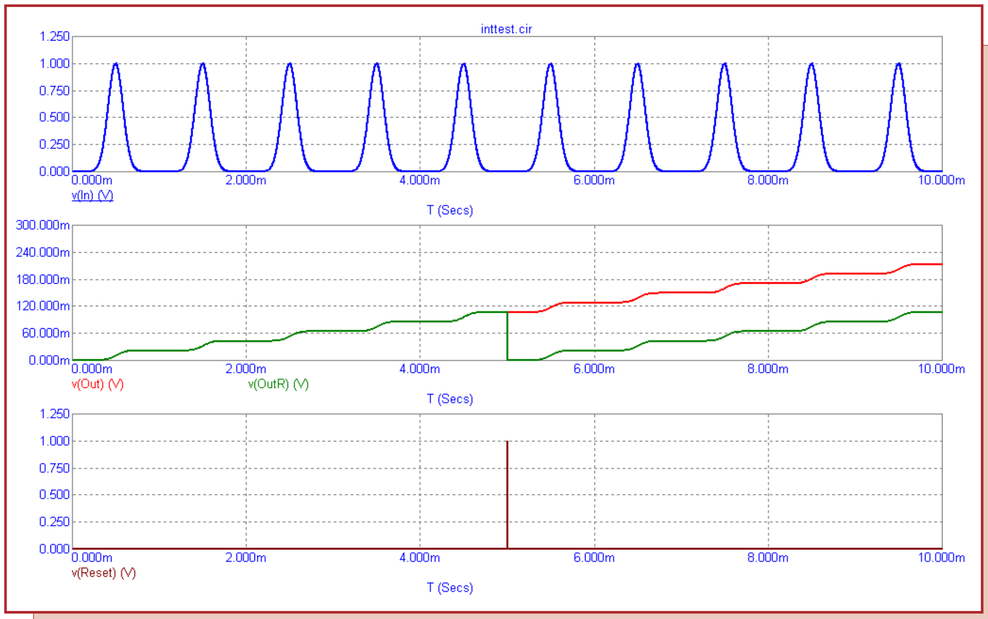
Spectrum News

Applications for Micro-Cap™ Users

Summer 2005

News



Resettable Integrator Macro

Featuring:

- Resettable Integrator Macro
- Centralizing Component, Shape, and Model Files
- Passing Parameters Through Batch Files

News In Preview

This newsletter's Q and A section describes the reason why noise and voltage/current waveforms cannot be plotted together in the same Probe AC analysis. It also describes how to specify the noise input and output for an AC probe run. The Easily Overlooked Features section describes how to toggle the input units used to optimize a core model from CGS to SI.

The first article describes a modification to the integrator macro so that it will have the capability to be resettable during a simulation.

The second article describes how multiple users can share component, shape, model library, and macro files among themselves in a shared directory.

The third article describes a method on how to pass parameters through to a circuit while running Micro-Cap in batch mode.

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 running a Probe AC Analysis on my schematic. If I enable the Voltage option under the Vertical menu, I can click on any of the nodes in the schematic, and the appropriate waveform will be displayed. If I then enable the Onoise option under the Vertical menu and click in the schematic, nothing will happen. My previous voltage waveforms stay on the screen, but no noise waveform is displayed.

Answer: In AC analysis, the network equations are structured differently when computing noise waveforms, so it is not possible to simultaneously plot noise variables and other types of variables such as voltage and current. You will need to delete the displayed voltage waveforms. Once this is done, the noise waveform can be plotted.

Starting with Micro-Cap 8 version 8.0.9, when the Many Curves option is selected under the Probe menu, the waveform types that are not available to be plotted with the displayed waveform will have their options disabled in the Vertical menu. For example, if a voltage waveform is currently displayed, the Inoise and Onoise options will be disabled. To access these options again, all of the displayed plots must be cleared.

Question: When I plot Onoise in a Probe AC Analysis, how do I know which node it is referring to?

Answer: The Onoise waveform will always refer to the Noise Output node specified in the AC analysis limits dialog box. To access the limits dialog box in probe, go to the Probe menu and select Limits or just hit the F9 hotkey. The Noise Output field should be specified with either the node name or the node number of the desired output node. The Noise Input is also specified in this same dialog box and should be defined with the Part attribute name of a source.

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.

Change Core Units

The Model program embedded within Micro-Cap can optimize model parameters from data sheet information. For the core, the information that needs to be input to optimize a model is the B-H curve of the core material. The B-H curve in a data sheet is typically plotted in one of the following unit types:

CGS: B=Gauss and H=Oersteds
SI: B=Teslas and H=Amps/Meter

The default settings for a core in the Model program is to have the input in CGS units. However, if the B-H curve information is in SI units, rather than having to manually convert the data over to CGS units, a single command will change the input unit type that the core will accept. The Change Core Units command available under the Model menu will toggle the input unit type between CGS and SI, so that it can easily handle either of the two as an input.

If there is currently data in the B and H columns, the Change Core Units command will also scale that data to the new units. Therefore, prior to entering data, check to make sure that the core units match the data units that are to be input.

Plotting the B-H curve in a transient analysis can also be done in both CGS and SI units. The B and H operators will plot in Gauss and Oersteds, and the BSI and HSI operators will plot in Teslas and Amps/Meter. Figure 1 displays the B-H curve of the same material using both unit types.

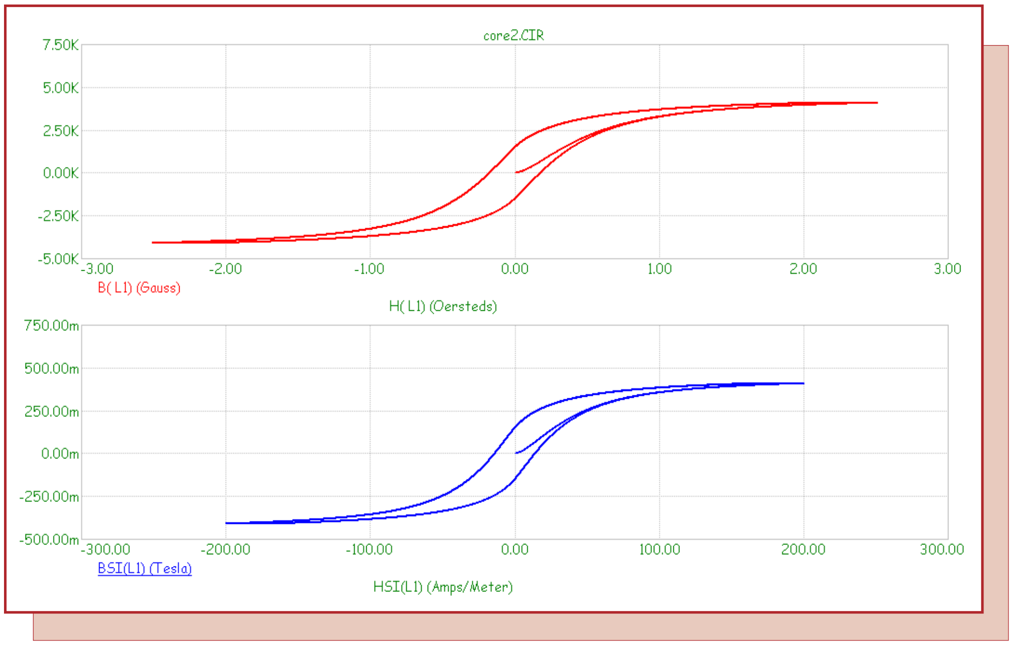


Fig. 1 - CGS and SI B-H curve plots

Resettable Integrator Macro

The Int macro within Micro-Cap is one of the most useful macros for system modeling. It produces an output signal which is the integral of the input signal to the macro. With the current Int macro, the output is the integral of the waveform over the entire period of the simulation. There is no means of resetting the macro. This article will describe a revised macro circuit that adds a resetting capability to the integrator macro.

The standard integrator macro that is distributed with Micro-Cap is shown in Figure 2. Two parameters are passed to this macro: SCALE and VINIT. The SCALE parameter multiplies the integral by its defined value, and the VINIT parameter defines an initial value for the integration.

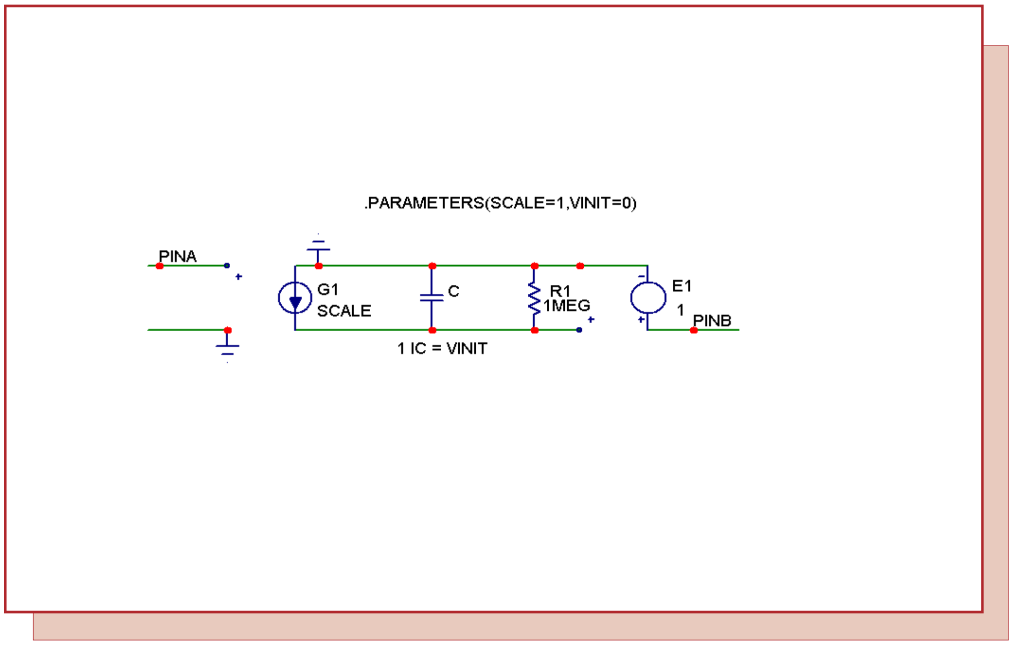


Fig. 2 - Standard Integrator macro circuit

The operation of the macro is fairly simple and takes advantage of the integrated nature of a capacitive voltage. The input voltage to the macro is converted by the G1 source into an equivalent current multiplied by the SCALE parameter. This current is then used to charge a capacitor. Since the value of the capacitor is 1 Farad, the voltage across it will be the integral of the current though the capacitor. The VINIT parameter is used to set the initial condition of the capacitor in order to provide an initial value that the integrated voltage is then built upon. The parallel resistor keeps the voltage finite when there is a DC voltage input. The E1 source then provides a buffered output which reproduces the integrated voltage across the capacitor at the output pin.

The standard integrator macro will be used as the basis for the resettable integrator macro. The resettable integrator macro circuit is shown in Figure 3. It passes the same two parameters, SCALE and VINIT, as the standard integrator macro. The basic operation of each of the macros is the same.

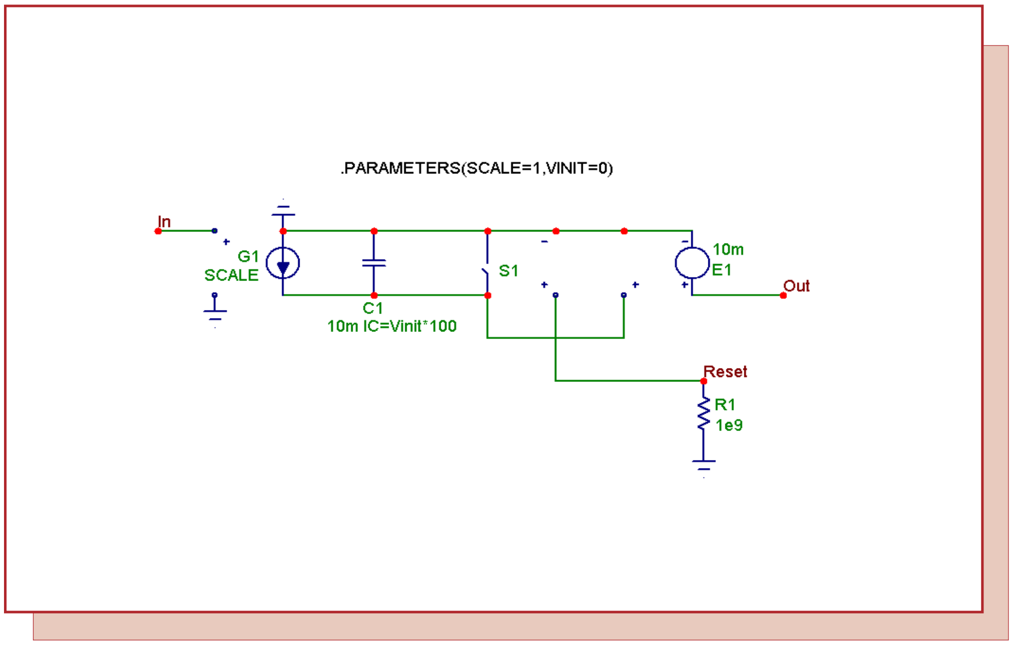


Fig. 3 - Resettable Integrator macro circuit

The main modification is that the parallel resistor has been replaced by an S (V-Switch) component. The input of the switch samples the voltage at the Reset pin, and the output of the switch models the parallel resistance of the integrator. The model statement for the switch is defined as:

.MODEL RPAR VSWITCH (RON=1u ROFF=1Meg VON=.75 VOFF=.25)

When the voltage at the Reset pin is below .25 volts, the switch will be off and the resistance of the switch will be 1Meg. The macro will then operate in the same manner as the standard integrator macro. When the voltage at the Reset pin is greater than .75 volts, the switch will be on and the resistance of the switch will be 1u. Note that the default value of RMIN, which sets the minimum absolute value of a resistance, in the Global Settings is 1u so this is the smallest the resistance can be without modifying that setting. The small parallel resistance will cause the capacitor to quickly discharge bringing the output voltage of the integrator down to zero volts which essentially resets the macro. The VINIT parameter will only set the value of the integrator at the beginning of the simulation. Any reset operation will use zero volts as the baseline. The Von and Voff parameters of the switch were assigned these specific values so that a boolean voltage waveform could be used on the Reset pin.

A 1e9 resistor on the Reset pin provides a DC path to ground for the node. With the addition of this resistor in the macro, the Reset pin can be left floating in a circuit that calls the macro. When floating, the Reset pin will be disabled, and the integrator will operate like the standard model.

The value of the capacitor in the macro has been set to 10mF in order to reduce the RC time constant when discharging the capacitor during the reset operation. Since the value of the capacitor does scale the voltage across the capacitor by the factor 1/C, the VofV dependent source (E1) at the output of the macro must also be scaled accordingly to compensate for this factor. Setting the value of E1 to 10m cancels the effect that the capacitor value has on the integrated waveform.

Finally, the initial condition of the capacitor must also be scaled so that the VINIT parameter will be treated appropriately. The IC keyword sets the initial voltage across the capacitor at the beginning of the simulation. However, the E1 source multiplies this initial voltage by 10m when propagating the voltage to the output. Therefore, the VINIT parameter must be multiplied by 100 when setting the IC value of the capacitor.

Should the time constant for the reset operation be too slow for an application, changing the capacitor value in the macro will also necessitate changing the E1 value along with the multiplier for the capacitor IC statement.

An example circuit has been assembled which contains both the standard integrator macro and the resettable integrator macro. The two integrators share a common input which is a Voltage Source defined as:

DC 0 AC 1 0 Gaussian 1 500u 200u 1m

which will produce a periodic Gaussian pulse waveform every 1ms. The Reset pin of the resettable integrator has a pulse source connected to it which will provide a short reset pulse at 5ms.

The resultant transient analysis is displayed in Figure 4. The top plot is the input waveform. The bottom plot is the reset waveform. The middle plot contains the output waveforms of both integrators with V(Out) being the standard integrator output and V(OutR) being the resettable integrator output. The two outputs match until 5ms when the reset pulse occurs. At this point, the resettable integrator output returns to zero volts and starts the integration over again.

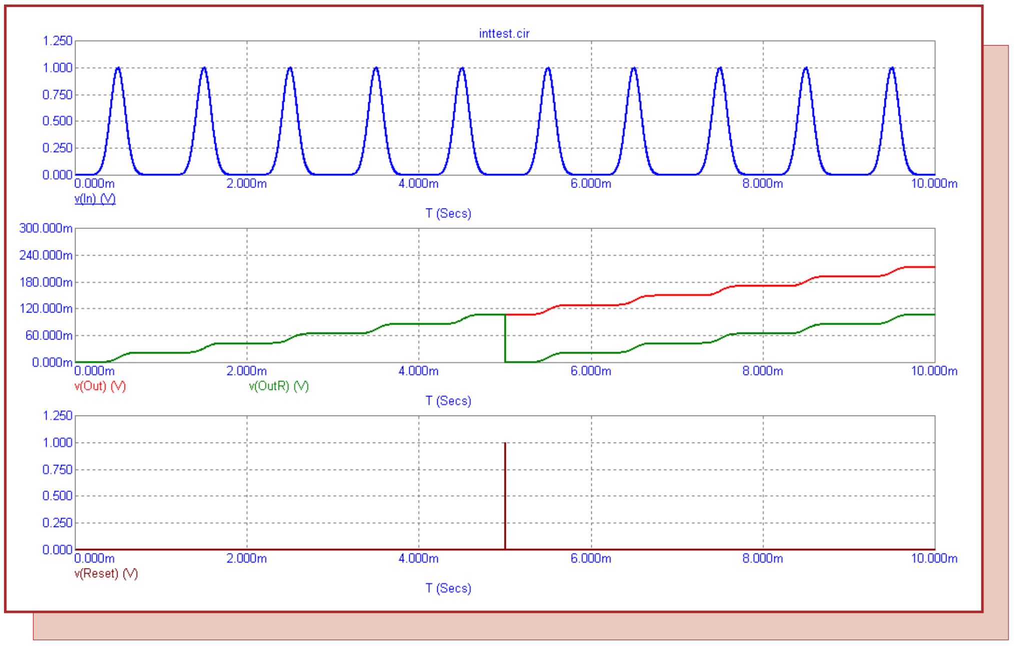


Fig. 4 - Integrator output waveforms

Centralizing Component, Shape, and Model Files

When multiple Micro-Cap users are working together, a common need is to share models among all of the users. If the Micro-Cap installations are on the local drives of each of the users, there are two main methods for exchanging models. The first method requires that the library or macro file be passed to each user manually, and then the component and shape information would need to be imported by the individual user. This method can be a bit tedious depending on the number of users who need access to the model. The second method, which will be the procedure covered in this article, is to create a shared directory where the users can centralize all of their component, shape, and model files in.

The most obvious requirement for centralizing files is that there is a shared location which all of the users have access to. For this article, the shared directory will be:

Z:\MCAP\

The second requirement is that all the data to be shared needs to be in new files not just appended to the files distributed with Micro-Cap. This is typically the case with library files that contain the model information. However, many users who add data in the Component or Shape editors input the data into the Standard.cmp (component file) and the Standard.shp (shape file). For centralizing the component and shape information, the users would need to create new CMP and SHP files to store the component and shape data.

Shape files
The shape (.SHP) files contain all of the shape information that Micro-Cap uses and are created and maintained through the Shape Editor. To create a new shape file, access the Shape Editor under the Windows menu. Click on the New File icon, and a New File dialog box will appear. Change the Save In field so that it points at the shared directory the files will be centralized in. In this case, the path would be Z:\MCAP\. Give the file the desired name, and then click the Save button. A new SHP file will be created in the shared directory. Any new shapes should be created in this file.

If the user has already created shapes prior to sharing the files, the technique to centralize them will depend on which file the shapes are located in. One possibility is that the shapes were created in a separate shape file from the Standard.shp file that is distributed with Micro-Cap. In this case, the first step is to select the shape file from the drop down list box right below the Shape Editor toolbar. Click the Remove File icon. This command removes the shape file from memory. Now the shape file will need to be moved over to the shared directory. Any of the normal Windows operating system file movement operations can be used to do this. Once the file has been moved to the new directory, click the Open File icon in the Shape Editor, and load the shape file from the new directory.

The other possibility is that the shapes were created in the Standard.shp file. First, create a new shape file in the shared directory as described above. Then select the Standard.shp file in the drop down list box below the toolbar. Select a shape that needs to be shared. Click on the Select All icon, and then click the Copy icon. In the drop down list box, select the new shape file. Click the Add button. Make sure that the name assigned to the shape is the same name as defined in the Standard.shp file. Click the Paste icon. Drag the entire shape so that it matches the grid alignment of the copy in the Standard.shp file. The shape's location should visually line up when switching between the two files. If the grid alignment is off, the shape will appear offset in the schematics. Return to the Standard.shp file and delete the instance of the shape in that file. Micro-Cap will use the first instance of a shape it finds so deleting the shape ensures that it will find it in the new shape file. Repeat the copy and paste process for all the user created shapes that need to be shared.

All other users who want access to this shape file will need to use the Open File icon in the Shape Editor to load the shape file into their copy of Micro-Cap. When a shape file is either created or opened by a user, the file will be loaded into their copy of Micro-Cap whenever the program loads until the file is manually removed by the user through the Remove File icon in the Shape Editor. Clicking on the drop down list box right below the Shape Editor toolbar lets you switch between the shape files that are currently loaded.

When loading both a shape and a component file from another user, the shape file must be the first one loaded since the component file will have references to its corresponding shape file. If this is done out of order, no error message will appear. However, any components that reference a shape that is not loaded will have its shape set to a default shape which is typically the battery.

Component files

The component (.CMP) files contain all of the component information that Micro-Cap uses and are created and maintained through the Component Editor. To create a new component file, access the Component Editor under the Windows menu. Click on the New File icon, and a new file will be added to the component tree on the right side of the editor. The file will initially be located in the same directory as the MC8.EXE file and called New.cmp. To change the path and name of the file, highlight the file reference in the component tree. The path and file name will now be available for editing. In our case, we would change the reference to something such as:

Z:\MCAP\Shared.cmp

Any new components should be created in this file. When the Component Editor is closed and saved, the Shared.cmp file will then be created in the Z:\MCAP\ directory.

As with shapes, components that the user has already created may be present in a separate component file or in the Standard.cmp file that is distributed with Micro-Cap. If the components were created in a separate component file, the path reference for the file will need to be changed in the Component Editor. Highlight the file reference in the tree on the right. Simply overwrite the old path with the path to the shared directory. When the Component Editor is closed and saved, a new instance of the component file will be created at the path specified. A copy of the file will still exist in the folder it was previously assigned to, but any future changes will only modify the file in the directory pointed to within the Component Editor.

If the components were previously added into the Standard.cmp file, the first step is to create a new component file in the shared directory as described above. In the component tree, components may be moved between files by dragging on a component, and then dropping the component onto the + or - square next to the destination group name or onto a component that already exists within that group. Repeat the move process for all the user created components that need to be shared.

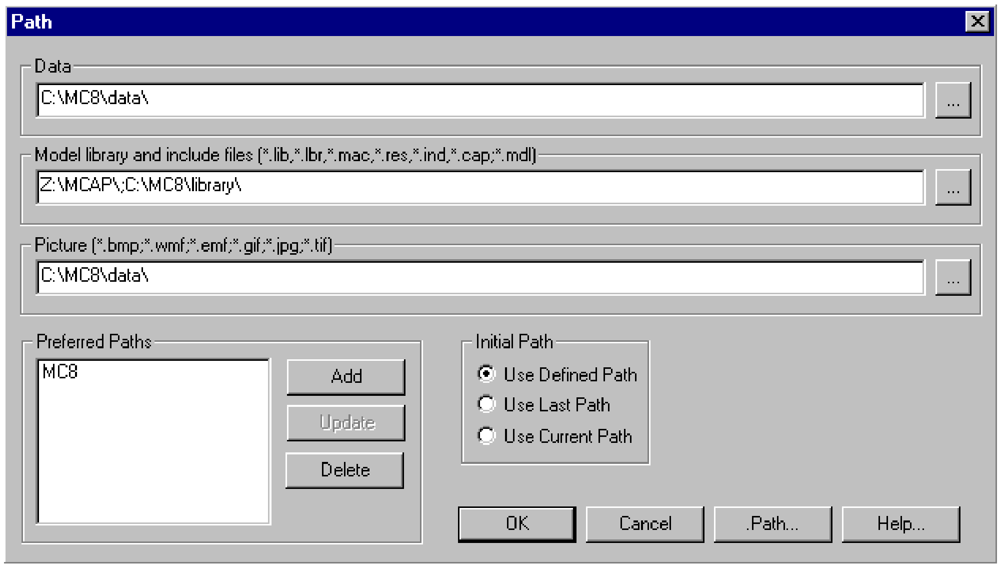
All other users who wish to access the component file will need to use the Open File icon in the Component Editor to load the component file into their instance of Micro-Cap. When a component file is created or opened by a user, the file will be loaded into Micro-Cap whenever the program loads. To prevent a file from loading with Micro-Cap, highlight the file reference in the component tree and then click the Delete icon. Clicking the Delete icon when a file reference is selected will only remove the file from the component library, it does not delete the file from the hard drive.

Library Path

Micro-Cap uses the library path to point to where the macro and library files are stored. The library path may be edited through the Paths option available under the File menu. Multiple paths may be specified by delimiting the paths with a semi-colon. For centralizing files, one path should point at the shared directory while another continues to point to the Library directory that is installed with Micro-Cap. The Model Library field within the Paths dialog box can be specified as:

Z:\MCAP\;C:\MC8\LIBRARY\

Placing Z:\MCAP\ first in the list will give it the highest priority. Whenever Micro-Cap needs a library file or a macro file, it will search all the directories listed here in order from left to right.


Path

Data: C:\MC8\data\

Model library and include files (*.lib,*.lbr,*.mac,*.res,*.ind,*.cap;*.mdl): Z:\MCAP\;C:\MC8\library\

Picture (*.bmp;*.wmf;*.emf;*.gif;*.jpg;*.tif): C:\MC8\data\

Preferred Paths: MC8

Initial Path: [✓ Use Defined Path] Use Last Path Use Current Path

Fig. 5 - Paths dialog box

Macro Files

Save all macro files into the shared directory. As long as the user's Library path above points to the shared directory, Micro-Cap will be able find them.

Library Files

As with the macro files, the library files to be centralized should be copied over into the shared directory. The main difference between library files and macro files is that the library files need to have a reference in the Nom.lib file in order for Micro-Cap to automatically search them when looking for models. A good method for modifying the Nom.lib file for shared files will be explained in the next section.

Nom.lib File

The Nom.lib file is located in the Library directory below the main Micro-Cap folder. It is simply a text file that contains a list of all of the libraries that Micro-Cap will automatically search through when a model needs to be located. Each library should be referenced in this list by using a .lib statement. For the user that creates the entries in the Component Editor, the libraries are automatically added into the Nom.lib when the Import Wizard or Add Part Wizard is used. All the other users who want Micro-Cap to have access to a library file would need to manually add the library reference to their Nom.lib file.

When centralizing library files for multiple users, there is a nice technique available where the library list can be updated globally so that each user does not have to manually add each library to their Nom.lib. For this example, three libraries have been placed in the Z:\MCAP\ shared directory: Andy.lib, Bill.lib, and Tim.lib. A new text file should now be created in the Z:\MCAP\ directory. We'll call it Share.lib. In this file, place .lib statements that reference all of the libraries that are to be shared such as:

```
.lib "Andy.lib"
.lib "Bill.lib"
.lib "Tim.lib"
```

Each user would then need to edit their individual Nom.lib file by adding in a reference to the Share.lib such as:

```
.lib "Share.lib"
```

so that the Nom.lib will reference the Share.lib file which in turn will reference all of the libraries in the shared directory. Now if a new library is added into the Z:\MCAP\ library, the user just needs to add a .lib statement referencing the library into the Share.lib file. All users who have the Share.lib referenced in their Nom.lib file would then automatically have the new library available to them. Again, the Library Path in the Paths dialog box would need to point at the location the library files are stored at for this method to work. Otherwise, absolute paths would need to be defined for the .lib references.

Passing Parameters Through Batch Files

A little known feature in Micro-Cap is the ability to run simulations through a batch process from the Program Manager command line. Typically this feature is used when multiple circuits need to be run, and either the numeric output or a printout of the analysis can be used to analyze the finished simulations after the batch file is done running. The format for running the batch file from the command line is:

```
MC8 [[/S | /R] | [[/P] | [/PC | /PA]]] [@BATCH.BAT]
```

Using this format, circuits can be simulated in batch mode by including the circuit name and the analysis type, /T (transient), /A (AC), or /D (DC) on a line in the text file. The syntax of the circuit line inside the batch file is:

```
CIRCUITNAME [[/T /A /D] [/S | /R] [[/P] | [/PC | /PA]]]
```

The definition for each command line switch is as follows:

/T - Run transient analysis on the circuit.
/A - Run AC analysis on the circuit.
/D - Run DC analysis on the circuit.
/S - Save the analysis to disk for later recall.
/R - Retrieve the analysis run from disk for plotting.
/PC - Print the circuit.
/PA - Print the analysis plot.
/P - Print the circuit and analysis plot.

For example, if the text file TEST.BAT contains the following lines:

```
PRLC /A /T
DIFFAMP /D /PA
SUBCKT1 /A
```

and MC8 is invoked through the command line as MC8 @TEST.BAT. The following sequence will occur:

1) Load the circuit PRLC. Run AC analysis. Run transient analysis. Close the circuit file.
2) Load the circuit DIFFAMP. Run DC analysis. Print the resultant analysis plot. Close the circuit file.
3) Load the circuit SUBCKT1. Run AC analysis. Close the circuit file.
4) Exit Micro-Cap.

This works well for running multiple circuits, but consider the case where just a single circuit needs to be run where one of the parameters will need to be updated for each analysis run. The current batch process does not have the capability to pass parameters although this will be included in the next generation of Micro-Cap. For MC8, in order to pass a parameter through to the circuit file that is being simulated in a batch process, the batch capability of Micro-Cap will have to be combined with the general batch capability of the Windows operating system.

A schematic that has been setup to receive parameters through the combined batch process is displayed in Figure 6. The circuit uses the UA741 opamp model in a basic non-inverting amplifier configuration. The parameter that is to be modified through the batch process must be defined as a variable in the schematic. In this case, the feedback resistor has been defined with the parameter RFB. Typically, the RFB parameter would have a corresponding .define statement associated with it such as:

.Define RFB 10k

However, the define statement in the schematic has been commented out, and in its place, the following include statement has been inserted:

.include f:\mc8\data\batch.inc

The include statement includes text from an external text file in a schematic. For this statement, it will include a text file called Batch.inc that is located in the F:\MC8\DATA directory. The include file provides a means to store the define statement for the RFB variable in a file separate from the schematic which can then be modified through a Windows batch file.

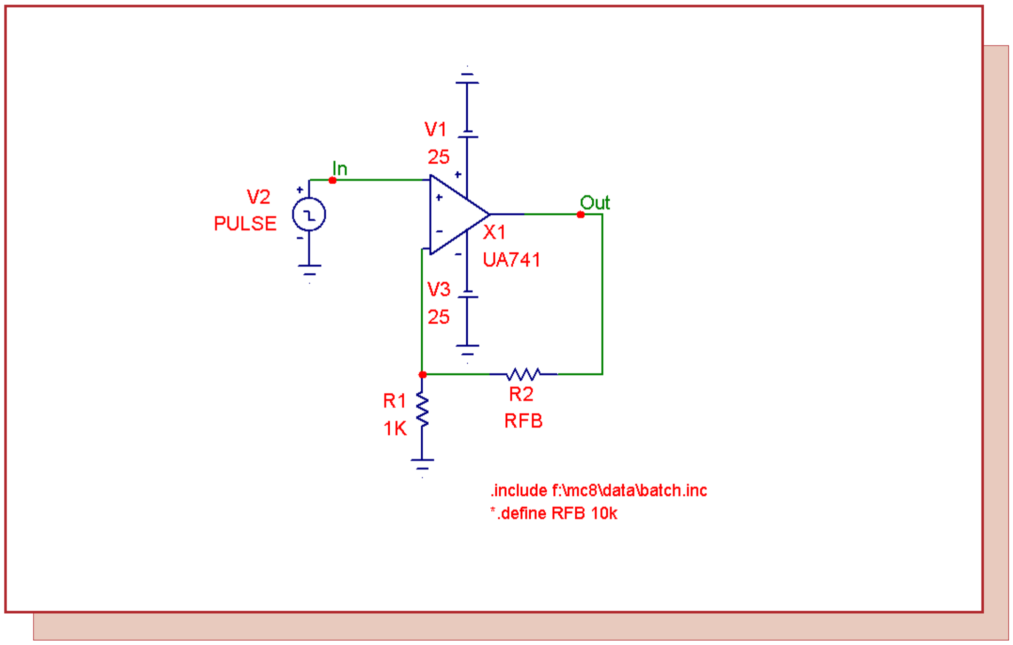


Fig. 6 - Batch file example circuit

The first text file created will be the batch file that Micro-Cap runs containing the circuit information. For this example, the file has been named LIST.BAT. Since only one circuit will be run, it just contains the following line:

f:\mc8\data\batch1 /a /pa

When the Micro-Cap batch file is run, the circuit Batch1.cir will be loaded, an AC simulation run, and then the results of the plot sent to the active printer. The second text file will be the batch file that will run under the Windows operating system. For this example, the file has been named TEST.BAT. It contains the following lines:

```
echo .define rfb 1k > f:\mc8\data\batch.inc
f:\mc8\mc8 @list.bat
copy f:\mc8\data\batch1.ano f:\mc8\data\batch1_1k.out
echo .define rfb 2k > f:\mc8\data\batch.inc
f:\mc8\mc8 @list.bat
copy f:\mc8\data\batch1.ano f:\mc8\data\batch1_2k.out
echo .define rfb 5k > f:\mc8\data\batch.inc
f:\mc8\mc8 @list.bat
copy f:\mc8\data\batch1.ano f:\mc8\data\batch1_5k.out
```

The first line uses the Echo command. This command repeats the string that follows it which in this case is the define statement for RFB. The redirect expression, >, is used to place the define statement into the text file Batch.inc. Basically, the include file that the schematic references is being created on the fly with this command.

The second line starts the Micro-Cap batch process. Micro-Cap will read the LIST.BAT file and run all of the specified simulations and associated commands. For this example, it runs an AC analysis on Batch1.cir and then prints the results.

The Batch1.cir has been setup so that the output waveform in the AC analysis is also sent to the numeric output file. The third line copies the AC numeric output file that was produced from the simulation and gives it a unique name. Since the Batch1.cir is going to be run a few times during this process, the numeric output must be saved to a new name after each run in order to be preserved. Otherwise, only the numeric output from the final run of Batch1.cir would be available.

These three lines are then repeated for the number of times the parameter needs to be changed. Two modifications need to be made for each set. The define statement value in the Echo line needs to represent the parameter's value for the next run, and the file name that the numeric output is being copied to needs to be unique. The TEST.BAT file can be executed by double clicking on the file icon for it in the Windows explorer.

In the above process, Batch1.cir is being run with the RFB parameter set at 1K, 2K, and 5K. Both the numeric output from the simulation and hardcopy printouts of the plot will be available to analyze after the batch process is complete.

In the Micro-Cap batch file, if the /P or /PA switches are being used to produce printouts of the analysis, there are two points to be aware of when passing a parameter with this method. The first one is that it is important to set the X and Y ranges in the analysis limits so that it will cover all the possible ranges of the plots being produced, or the plot may go offscreen and not appear in the printout. The Auto or AutoAlways options can also be used. The second point is to modify the Plot Title so the printout displays which parameter value the plot was created with. The Plot Title can be edited in the Plot page of the Properties dialog box. To edit the title, the Auto checkbox must be disabled in the Title field. This example used the following entry:

```
Batch1.cir  RFB=$RFB$
```

where the RFB syntax is replaced with the value of the RFB variable for the run. When RFB is set to 5K, this will produce the plot title:

```
Batch1.cir  RFB=5K
```

Product Sheet

Latest Version numbers

Micro-Cap 8	Version 8.1.0
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