SIMetrix Self Training

Issue 1.1

SIMetrix Technologies Ltd. Terence House, 24 London Road, Thatcham, Berkshire, England RG18 4LQ
www.SIMetrix.co.uk t: 0(+44)1635 866395

SIMetrix TECHNOLOGIES

Overview Table of Contents

Intermediate Table of Contents

SIMetrix TECHNOLOGIES

Detailed Table of Contents

SIMetrix TECHNOLOGIES

Table of Figures

1 Introduction

This self-training course has been designed to allow users to learn how to use SIMetrix without requiring the expense and time needed for an instructor based course.

The material presented covers a range of experience levels; section 2 for example is designed for complete novices while sections 13 to 16 on the script language present more advanced material designed for experienced users.

1.1 System Requirements

This training course assumes you are using SIMetrix version 5.6 on Windows. Later version are likely to be OK, but be aware that some images of SIMetrix windows and dialogs may be inaccurate.

Although in many case earlier versions will work OK, the initialisation procedure described in 1.2 below will not function. If you currently use an earlier version, we strongly recommend that you install version 5.6 to complete this training course.

1.2 Installation of Training Material

You should already have installed the training course using the MSI installer. The install program installs this PDF file along with a number of support files that are referred to throughout the course.

The support files are installed to **My Documents\SIMetrix\Training** and are divided into sub-directories relating to the section number. So for example, material for Section 2.x can be found in subdirectory **Section-2**.

IMPORTANT NOTE 1: Ensure that the training material is freshly installed and isn't what was left over by another user. If it is not freshly installed, we suggest that you uninstall it, then delete or move the contents of **My Documents\SIMetrix\Training** then reinstall. Be sure to close this file first before uninstalling otherwise you may be asked to reboot your computer by the uninstaller.

IMPORTANT NOTE 2: As you work through the training you will, as a consequence, make changes to to the SIMetrix configuration. This will render some of the software unsuitable for a subsequent training session which should be able to rely on a known configuration.

As the very first step, therefore, go to the Command Shell, type **setup_training** and press enter. This requires version 5.6 of SIMetrix. This will (run a script to) carry out all the reconfiguration necessary to put your installation of SIMetrix into a known state for a new session. When you first execute this script, you will see a warning message advising that some of your preference settings will be lost. If accepted you will see the display shown in Figure 1-1 as final confirmation:

Figure 1-1. Training Setup Confirmation

1.3 About this Training

A broad outline of the training is:

- The Practical Overview guides you through building, examining and improving a circuit.
- Individual sections keep description to a minimum and provide practical illustrations about the topic in hand.
- Documentation conventions include:
  - Where the word 'press' appears it refers only to keys on the keyboard; the word 'click' is used to refer to clicking on an object on the screen.
  - **Bold** text is generally used to denote SIMetrix filenames and menu items. In the Glossary of Terms, a word in bold indicates that the word is, itself, in the glossary.

- Italics is generally used to highlight a point which is important but which might be overlooked.
- Monospaced font is used to denote scripting code or output on the Command Shell.

There are several appendices as follows:

- Appendix A. is a list of acronyms used in the documentation.
- Appendix B. is a glossary of terms. It is more than a glossary and provides terminology-related information that is not provided elsewhere in this training.
- Appendix C. provides notes about text editors with particular reference to Notepad++. You will need a syntax highlighted editor for editing scripts.
- Appendix D. provides notes about Netlists.
- Appendix E. provides notes about graph symbolic values with a table showing the available variables, descriptions and contexts.
- Appendix F. provides some titles for further reading

1.4 Feedback

We would like this documentation to be interactive. If you have comments about how it could be improved, additional material that could be covered, alternative means of presentation, additional remedies for a simulation or plot that will not run, or whatever, please contact us.

1.5 Conventions Used in the Software

Normal Windows conventions generally apply and the exceptions include:

- On the Schematic Editor, a component that is 'selected' in Windows terms (or 'highlighted') is coloured blue. (All other components are coloured red.)
- The technique of selecting and moving a curve in the Waveform Viewer.
- If you click on minimise in the Schematic Editor window it does not minimise to the taskbar; to minimise the application, do so on the Command Shell.
- When you close a window containing a graph without having saved the contents, you do not get prompted to save. (Such graphs can be regenerated very easily, though.)
- The undo stack is accessible with the green 'back' arrow (or Ctl+Z) but the redo stack is available only from the Edit menu.

There is very little in SIMetrix that is case sensitive.

1.6 General Operational Notes

1.6.1 Error Message Display

Most SIMetrix errors are displayed in the Command Shell without any alert. If something doesn't work, look in the **Command Shell** to see whether there is any message explaining the problem. If the command shell is buried under other windows, press the space bar to bring it to the surface.

1.6.2 Why Doesn't It Work

If you get different results from those shown in this training (or no result at all), check the following:

- Have you got a ground symbol and a DC path to it?
- Are all the component values correct?
- Are all the components the right way round (where relevant)?
- Have you set up the simulation correctly? Go to Choose Analysis. Have you got the right boxes ticked?

1.6.3 Defining Shortcut Keys

You can define your own shortcut keys. In the Command Shell, go to **File|Options|Edit menus**. On the Edit Menu System dialog, expand Schematic and, within that, expand Simulator. Click on Choose Analysis and click on the Accelerator (short cut key) button and press the key or combination that you

want to assign. (Key 5 is not assigned to a menu item by default.) If you choose a key that is already assigned, your new assignment will over-write the default one at a global level in the user's data area.

2 Practical Overview of the Software

The approach that this practical overview adopts is to build a working circuit (a hybrid PSU). We will build the circuit in the Schematic Editor, run some analyses on it, take some measurements and modify the circuit to produce improved performance.

When you run the software, the opening screen that appears is called the **Command Shell** and the other parts of the program can be launched from it.

First, we will open the Schematic Editor, place components on it, wire them up and then assign values to them.

2.1.1 Placing Components

On the Schematic Editor, we will build the circuit shown as Figure 2-1.

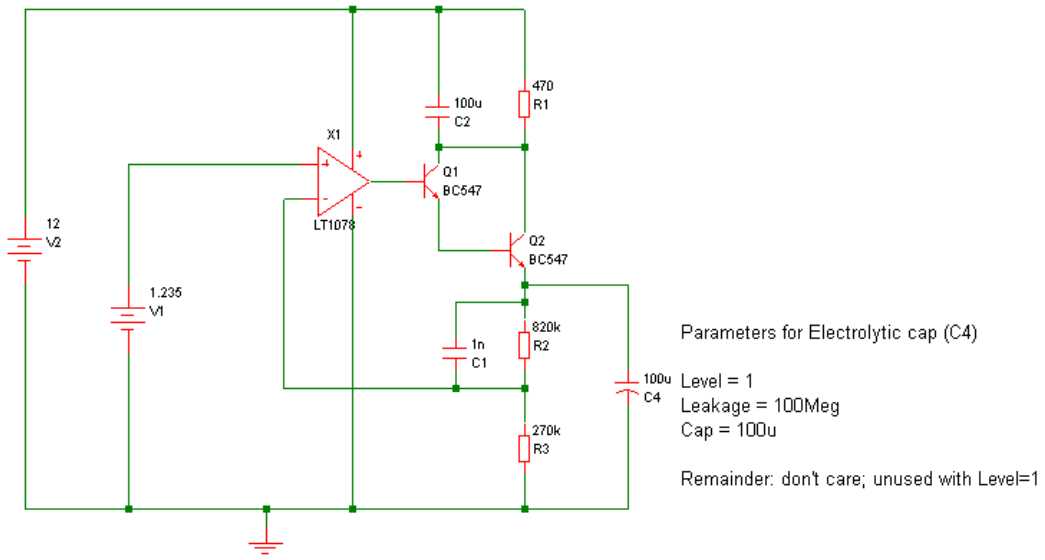

Figure 2-1. First part of the circuit

The procedure is:

1. In the Command Shell, go to **File|New Schematic Window** . This opens a blank Schematic Editor.
2. In the Schematic Editor, go to **Place|From Model Library...** (Ctl+G) which, unsurprisingly, opens the model library.
3. From the left pane, select Op-amps (you can also filter the selection using wildcards at the bottom left of the dialog, for example 'lt*' finds all display names beginning with 'lt' or 'LT').
4. In the right pane, a (possibly filtered) list of op-amps appears. Select LT1078 and the symbol for it appears in the bottom left pane.
5. Click the Place button. The model library dialog disappears and a shadow of the symbol appears at the cursor position. Click left to place it; it always snaps to grid. There is more information about placing items in Inserting Fixed Probes on page 21 and Random Probing on page 21.
6. To place an NPN transistor of a specific type, go to **Place|From Model Library**. Select NPN from left pane, select BC547 from right pane. Click Place button.
7. To place the battery, go to **Place|Voltage Sources|Power Supply** (or press V). For the electrolytic capacitor, go to **Place|Passives|Electrolytic Capacitor (Simple)**. There is more information about parameterised parts under Practical Development of Parameterised Parts on page 35.
8. Repeat steps 2 through to 5 for all the remaining components. You also need to be aware of the following:

2.1.2 Things you will Need to Know

- An easy way to repeat a component placement is to select it and do Ctl+D.
- The Home key will resize the schematic to fit the current window.
- If you are in a sequence-mode (such repeat placing), right click to get out of it.
- The Schematic Editor has an undo stack (with hotkey) and a redo facility (via the menu).
- The ground connection is essential; nothing works without it.
- Bipolar Junction Transistors are not listed as such; select NPN or PNP.
- There is a highlighting facility to assist in locating a component in a big schematic. For example on the Schematic Editor, go to **Edit|Highlight Component by Reference**. The Enter text dialog dialog opens, type in the component reference and it will turn a bold, bright pink in the schematic. There is a variety of highlighting options in the Schematic Editor.
- To delete a component in the Schematic Editor, select it and press the Delete key.
- To cancel selection of a component from the menu, press the Esc key.

2.1.3 Component References

On the Schematic Editor the component references (the numbered labelling or IDs) are numbered sequentially in the order in which they are placed. They can be changed (but no duplicates are allowed). Default values are assigned to a component but we will change some of these later.

Note: Although it makes no difference to the simulation it will be helpful for the training if you ensure that the component references are the same on your Schematic Editor as in the training material. This is so that if we make reference to a component there is no ambiguity about which component we are talking about. To edit a component reference, select the component and go to **Edit|Change Reference** or press F8.

2.1.4 Wiring

Wire up the circuit. Move the cursor near to any pin and it changes to a pen. Click while it is a pen, drag to the destination, release and click again. The mode persists in case you need it again. If not press Esc, or right click anywhere, to exit wiring mode. (This is temporary Smart Wiring mode. Other wiring modes are described in the User Manual starting on page 77.) Some other points about wiring are:

- The route that the wiring chooses is not always a good one but you can add your own corner to a route with one mouse click (in Smart Wiring mode) and then move on.
- When part of the cursor is over an item it changes colour to pink (for wiring) or cyan (for components).
- If you disable or enable Smart Wiring, the change does not take effect on any open schematic. To implement the change, close and re-open. It reads the option settings for that schematic when you open it. To change mode, in the Command Shell go to **File|Options|General|Schematic tab|Wiring group**.
- There is a detach feature on the  icon. If you just drag a component it will take its wiring with it. Detach allows you to move it away from its position cleanly.

2.2 Setting Parameters and Markers

2.2.1 Setting Component Parameters

Double click the component and an appropriate dialog box appears in which you can change values.

2.2.2 Setting Bias Annotation Markers

These are markers at which you can measure DC voltage and current (the so called Bias Point).

1. Do Ctl+M (voltage marker) and left click to place it at the emitter of Q2.
2. Do Shf+M (current marker) and left click to place it on one of the pins of V2.

Note: A current marker must be attached to the pin of a component, not just anywhere on the 'wire'. The pin is coloured red by default and blue (with blue crosses) when the component is selected.

Note: You can also use **Place|Bias Annotation|Auto Place Voltage Markers** though this can sometimes clutter up the schematic making both markers and the rest of the schematic difficult to read. If you wish to delete all the markers, go to **Place|Bias Annotation|Delete Markers**.

Note: Bias Annotation Markers address only the DC Bias Point. Information about probing can be found under Inserting Fixed Probes on page 21 and Random Probing on page 21.

2.2.3 Running a DC Operating Point Simulation (DCOP)

Go to **Simulator|Choose Analysis** and on the Choose Analysis dialog ensure than DCOP is the only Analysis Mode that is checked as shown as Figure 2-2.

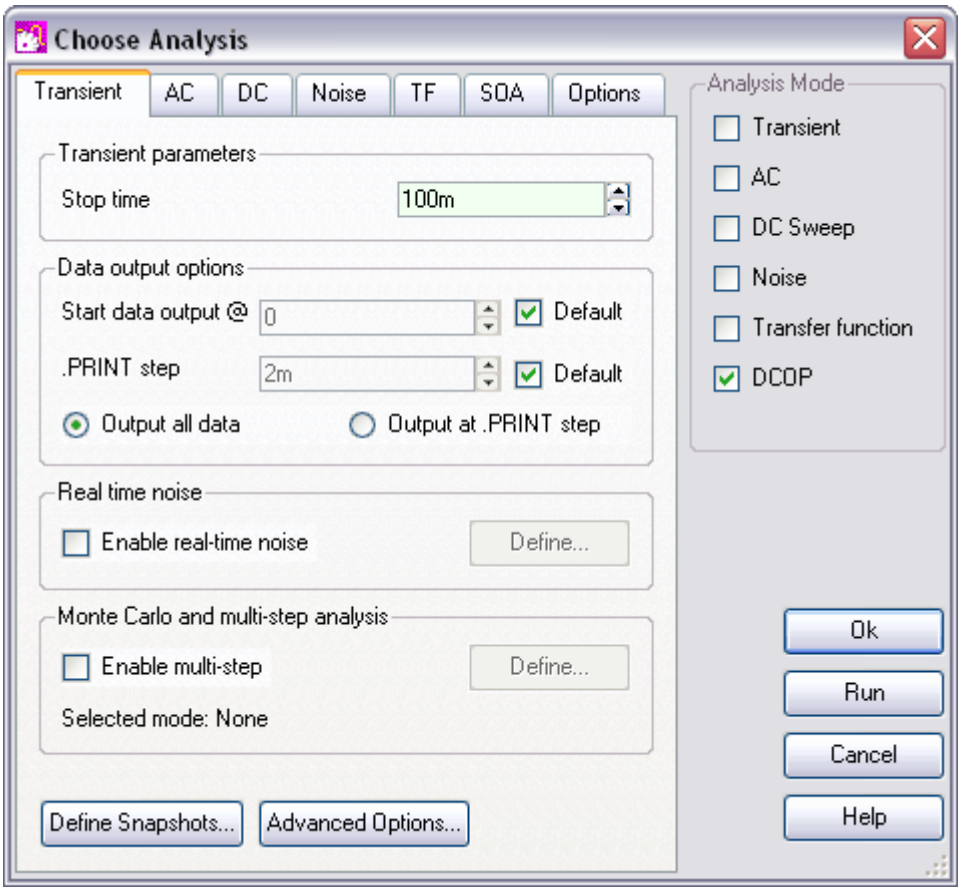

Figure 2-2. Choose Analysis - DCOP

Click Run, the dialog closes and the DC values appear on the Schematic Editor. They should be 4.98063v and 49.6191uA (u is used instead of µ). If the values you obtain are different, you may wish to check the following:

- Is there a ground connection? The output voltage will be around 3.2-3.3v if ground is missing.
- Are all the resistor values correct?
- Are the R1 and R2 voltages correct?
- Are the collectors of Q1 and Q2 connected? V2 will be slightly low if they are not.
- Are the C4 parameters correct? A default leakage of 1MΩ leads to V2 current being high by 5uA.

2.2.4 Completing the Circuit

We have now added enough components to get the feel of it.

1. Open **output.sxsch** as follows.

   Select menu from schematic editor **File | Open …**
   Navigate to **My Documents\SIMetrix\Training\Section-2**
   Select file **output.sxsch**

   The schematic opens in a new tab on the Schematic Editor.

2. Select all (Ctl+A) and copy it (Ctl+C).
3. Change tab to your own circuit and paste. (You may have to make room for it)
4. Move it to align the ground line and make the three connections by selecting each line and dragging it to the required location. The completed circuit is Figure 2-3.

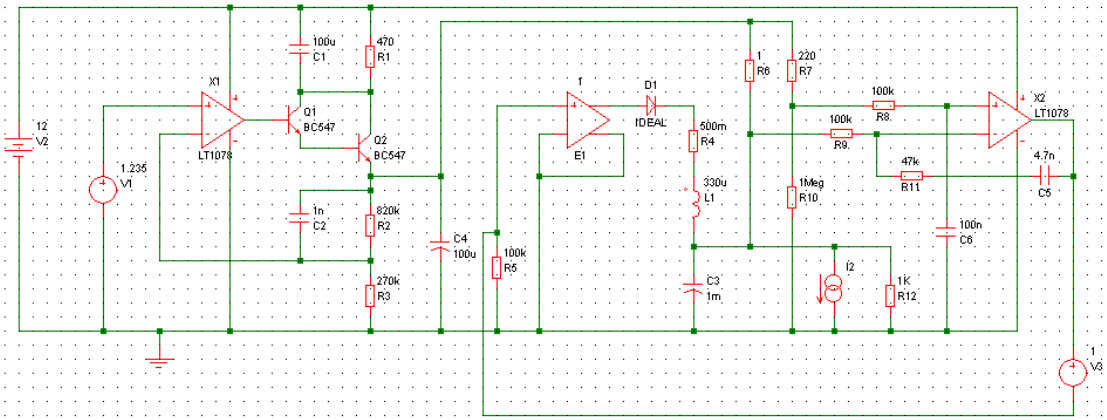

Figure 2-3.The Completed Circuit

2.3 Examining the Design

2.3.1 Inserting Fixed Probes

We now set up a fixed probe to create a curve of voltage against time.

1. Go to **Probe|Place Fixed Voltage Probe** (or Press 'B') and a shadow of the symbol appears at the cursor.
2. Left click to place it on the output which is on the connection between L1 and C3.
3. Double click the probe and this opens the 'Edit Probe' dialog. The only thing you need to change is the Curve Label. Replace the default entry with 'Output Voltage'.
4. Click OK. Notice that the Component Reference changes to 'Output Voltage'.

2.3.2 Random Probing

You do not need to have a fixed probe on the Schematic Editor to plot a curve. When the simulation is complete you can deploy a single-use probe of many different kinds. They are to be found on the Schematic Editor in the **Probe** and **Probe AC/Noise** menus. An example is given under Random Probing on page 26.

2.3.3 Running a Transient Analysis

Now we are going to set up a transient analysis.

1. Go to **Simulator|Choose Analysis** and ensure you are looking at the Transient tab.
2. Uncheck DCOP and set Analysis Mode to 'Transient'.
3. Enter 100m for stop time. The circuit will then run for 100mS and plots what it does in that time.
4. Click Run. This opens another window showing the curve of Output Voltage v Time. The scale of the y-axis is selected automatically to show the range of the dip. Notice that there is only one y-axis. The curve is shown as Figure 2-4.

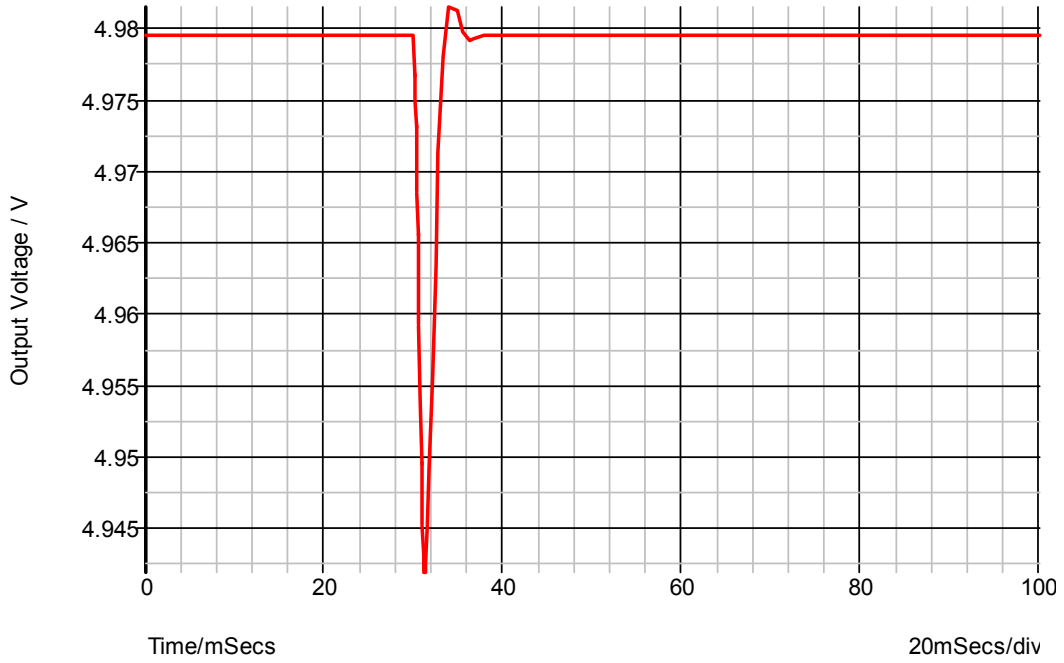

Figure 2-4. Transient Analysis - no probe

5. On the Schematic Editor, right click on the load (I2) and, from the context menu that appears, select Probe Current. This puts a shadow of the probe at the cursor position. Left click to place it exactly on a pin attached to the load. Result should be as Figure 2-5.

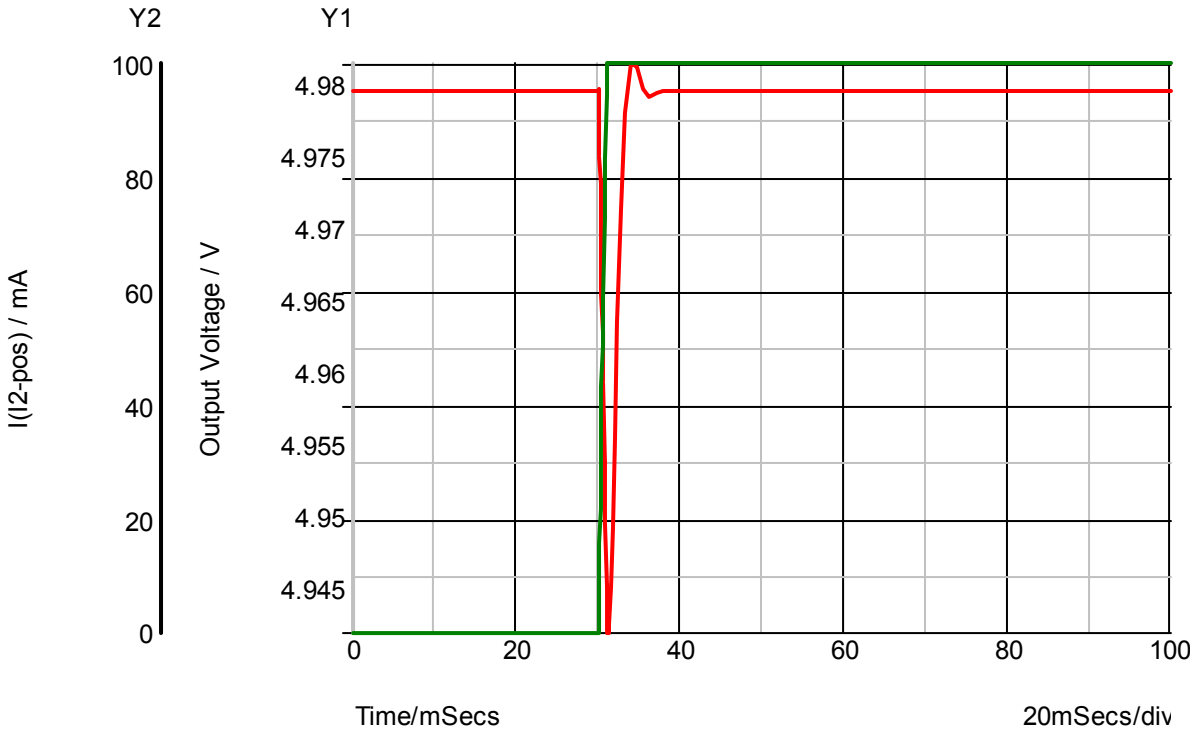

Figure 2-5. Transient Analysis - with current probe

- A second curve (in green) on the same grid which shows the step change in current from 0 to 100mA which caused the voltage dip.

- A second y-axis (they are now labelled Y1 and Y2) showing load current (also automatically scaled)

With Automatic Colouring enabled, SIMetrix will colour the first curve in red, second in green, third in blue and then khaki, aqua, purple, brown, navy, amber, green.

2.3.4 Cursors and Basic Measurement

In the Waveform Viewer, go to **Cursors|Toggle On/Off** to show cursors. Values are always displayed in the nearest engineering unit.

Two pairs of crosshairs, which both always follow the curve, appear on the curve. Both pairs are shown in dashed lines. The one with the shorter lines is the main cursor and the one with the longer lines is the reference cursor. The software puts them at the 10% and 90% points by default as shown as Figure 2-6.

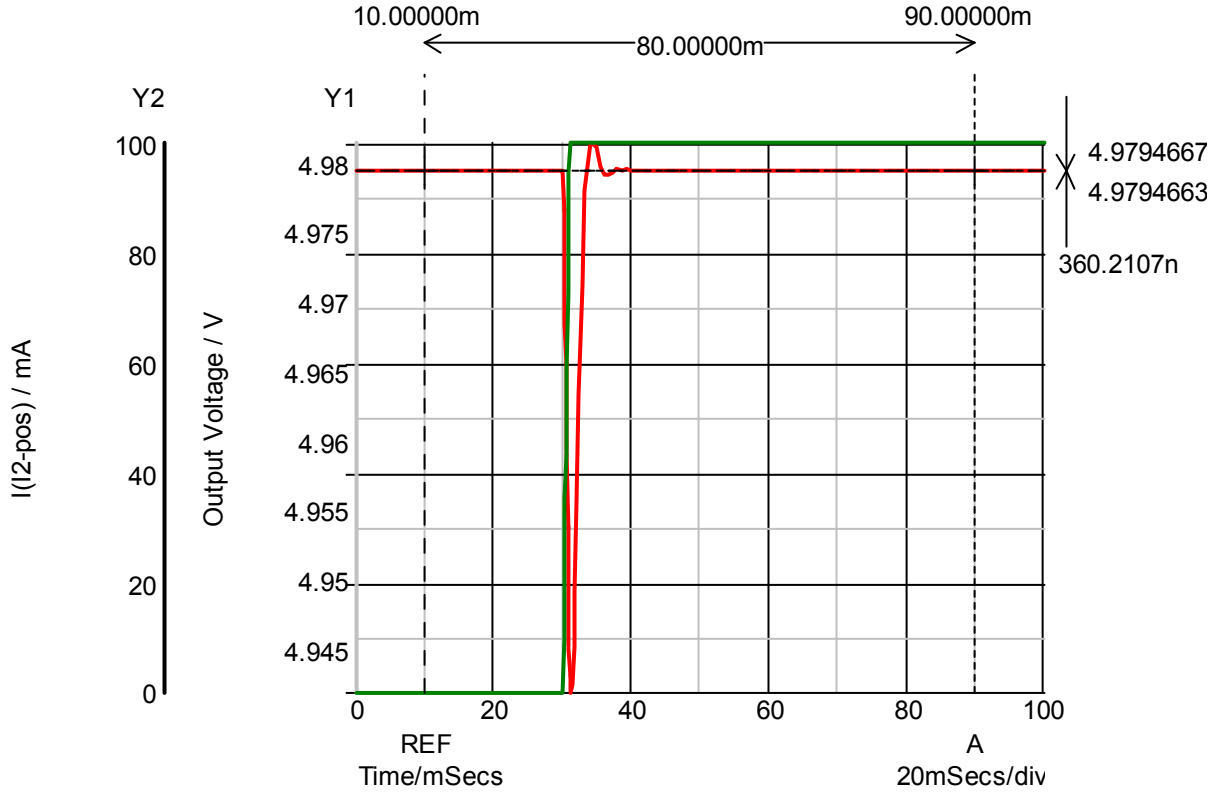

Figure 2-6. Cursors

You can move any of the four lines. Move the cursor near to a vertical crosshair and the cursor changes to indicate horizontal movement and vice versa.

1. Ensure that the main cursor is at the stable level of the output voltage
2. Move the reference cursor to the bottom of the droop
3. Read the height of the droop. It should be about 37.59mV.

2.3.5 AC Analysis / Predicting Stability

We wish to improve the load transient performance of the circuit, that is, the droop we measure in the previous step. To do this we will first perform some measurements using AC analysis. This carries out a frequency sweep.

From this you can determine whether there is sufficient gain margin to increase the loop gain without the circuit going into oscillation.

1. Double click on V3 and enable AC. Click OK.
2. Go to the menu **Probe AC/Noise** and select **Bode Plot Probe**.

3. Place it in parallel with V3 as shown as Figure 2-7.

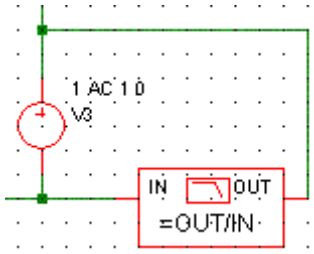

Figure 2-7. Bode Plot Probe

4. Double click on the Output Voltage probe and uncheck AC sweep in the Edit Probe dialog. Click OK. (This is so that we don't get a new trace for the output voltage probe.)
5. Go to **Simulator|Choose Analysis** and select the AC tab and set the values and selections shown as Figure 2-8. Be sure to uncheck the **Transient** box and check the **AC** box. Click Run.

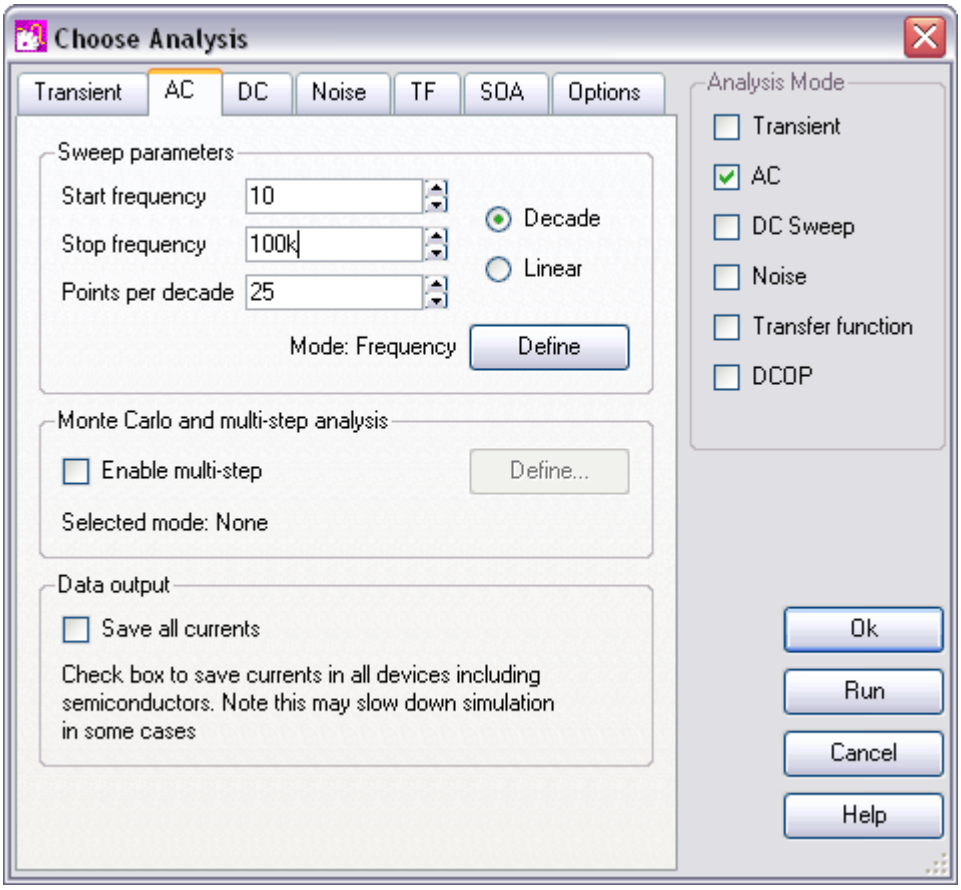

Figure 2-8. Choose Analysis - AC

This produces curves such as shown as Figure 2-9.

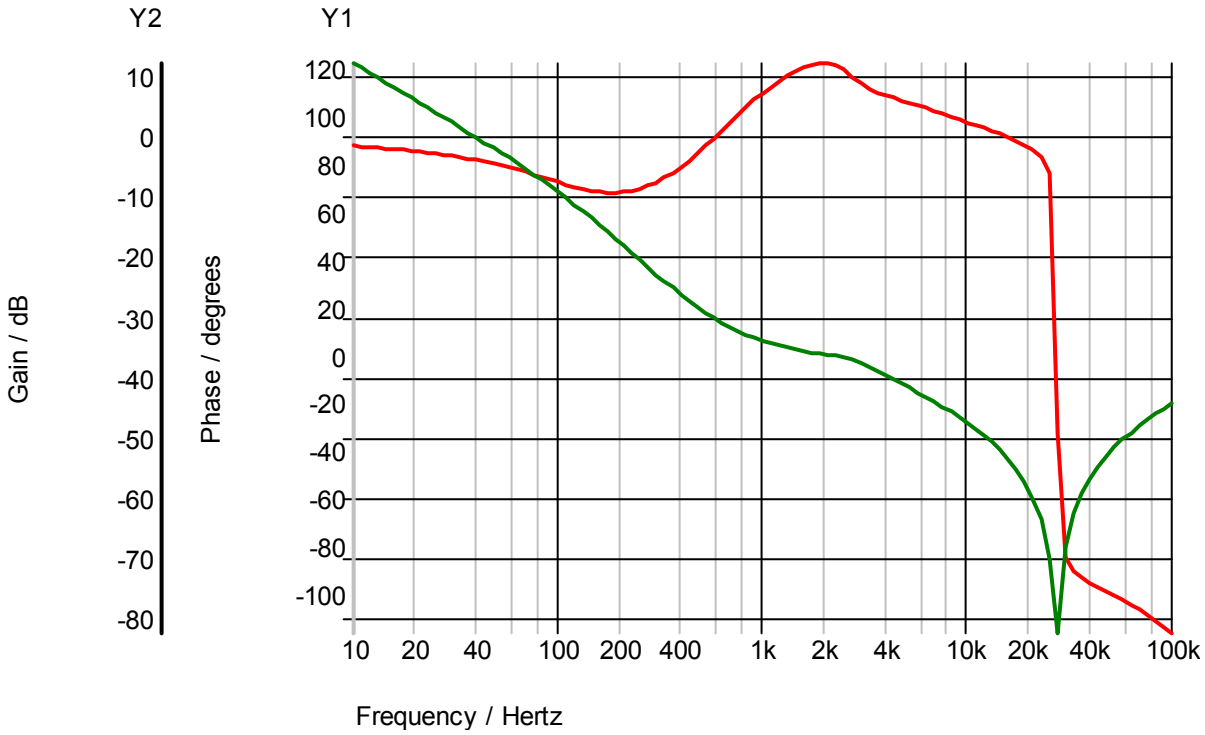

Figure 2-9. AC Analysis Result

Note: A phase shift of 0/360° indicates instability if the gain is above 0dB. The phase shift at 0dB gain is 90° so the loop is currently very stable. We therefore have room to increase the gain. We shall try increasing it by a factor or 10.

2.3.6 Increase Loop Gain and Set up a Transient Analysis

1. On the Schematic Editor, change R9 to 10K.
2. Change back to Transient Analysis using **Simulator|Choose Analysis**. Clear the AC box and check the Transient box.
3. Select the tab in the graph window with the original transient response. Delete the current curve (should be green) by checking the box labelled I(I2-pos) then select menu **Curves|Delete SelectedCurves**.
4. Run the simulation; the result is shown as Figure 2-10.
5. Measure the droop as before. You can move the cursors to the new curve by placing the mouse cursor at the centre of the crosshair then left click and drag to the new curve.
6. Close all the curves as preliminary to the next step.

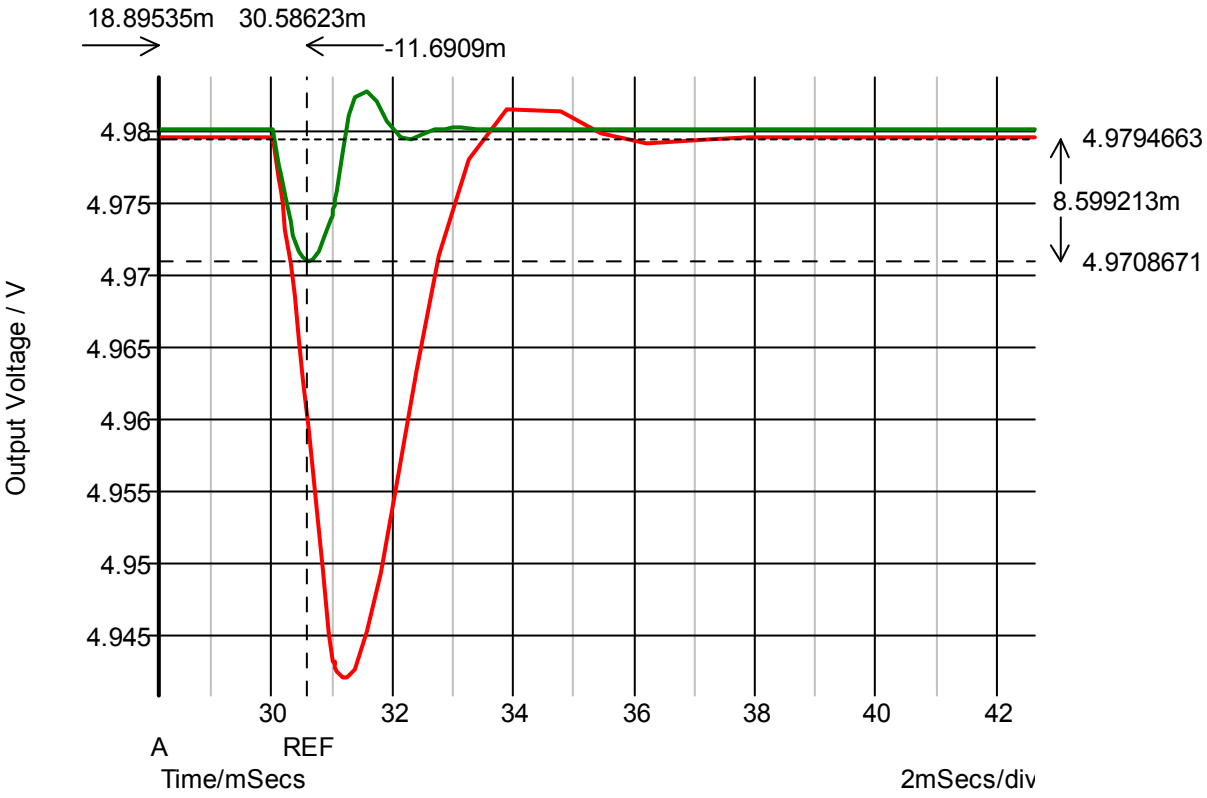

Figure 2-10. AC Analysis - Improved Result

2.3.7 Random Probing

1. Close the graph window
2. To probe the circuit after the simulation is complete, right click on blank space in the Schematic Editor and select Probe Current from the context menu.
3. Click on L1 and the curve shown in as Figure 2-11.

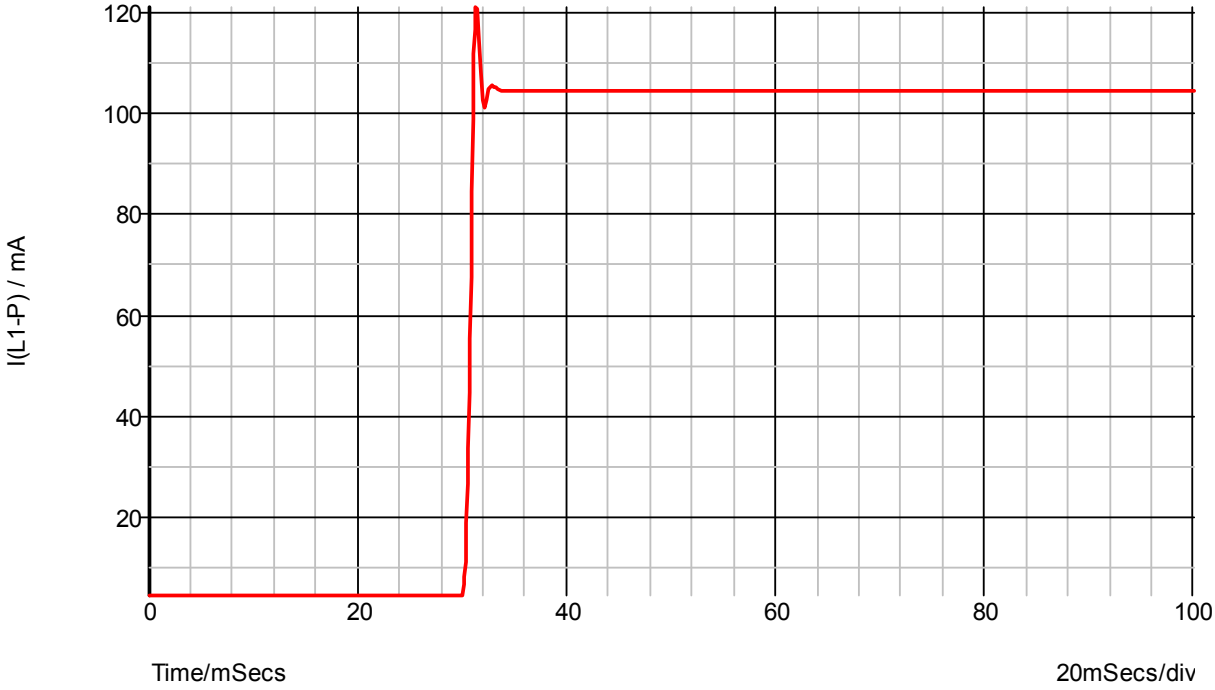

Figure 2-11. Post-probe current in L1

2.4 Configuration Options

To change a configuration, go to **Command Shell|File|Options**. Here you can edit menus, fonts and colours. Two points are worth mentioning:

- Font style and font colour are separate because they need to be for use on a Linux platform
- Black background automatically changes the foreground to a contrasting colour.

Command Shell|File|Options|General gives you a dialog with six tabs which contain a wide range of options, most of which are self explanatory.

2.4.1 Editing Options

You can edit any item on the Schematic Editor by double clicking it, by selecting it and F7 or by right clicking and selecting **Edit Part** from context menu. No editing is possible on an Annotation Marker.

All the editing modes are described in the User Manual on page 79.

2.4.2 Wiring Options

The dialog offers three wiring modes: Classic, Grow Wire and Orthogonal. In Orthogonal, when you move a component in the Schematic Editor, its wiring moves with it but maintains horizontal and vertical directions .

The only difference between Classic and Grow Wire is that when you separate components joined by their pins, Grow Wire creates a wire to keep them connected whereas Classic does not.

The default mode is Classic but it is recommended that you change this to orthogonal.

2.4.3 Temporary and Permanent sub-modes

You may find it helpful to be aware of the similarities and differences between 'temporary' and 'permanent' wiring sub-modes. You don't need to go to the **File|Options|General|Schematic tab|Wiring group** dialog to change between them; you just manipulate the Schematic Editor differently for each. The similarities are:

- Both produce the same result
- Both change the shape of the cursor to a pen

- Both invoke smart wiring mode if enabled in the dialog
- The left hand end of the status bar indicates the mode at all times.

The operational differences are:

- **Temporary**. Move the select cursor to the pin of any component or the end of a wire and it changes to the pen. Click, drag and click again. The pen remains to draw another wire; if you don't want to do that, right click or Esc to revert to select mode. You can also use the middle mouse button, scroll wheel or F3 to get into temporary mode.
- **Permanent**. Toggle the pen icon to open or close permanent wiring mode. The pen icon turns a lighter shade of grey. Right click cancels the wire but the pen remains. To clear the pen, toggle the icon or press Esc.

If you change a mode in the **File|Options|General|Schematic tab|Wiring group** dialog, the change takes effect only when you open a new schematic. Any schematic still open when you change the mode keeps the mode it had before the change.

The factory default is for Smart wiring to be ON. Further detail on wiring is in the User Manual, page 77.

2.5 Background Information

2.5.1 The Schematic Editor Grid

The Schematic Editor Grid is present all the time. It may be hidden using **View|Toggle Grid**, but it remains active even if not visible.

To prevent the grid being too intrusive, at certain zoom settings not all lines are shown. This gives the appearance of some wiring snapping to a grid line that is not visible.

2.5.2 Use of the Symbol Library for Placing Parts

The majority of parts useable for simulation are available either from dedicated menus (under **Place**) or from the **Place | From Model Library…** menu.

Access to the complete symbol library can be made via the menu **Place|From Symbol Library**. Be aware that the parts obtained directly from the symbol library will not necessarily work in a simulation. We supply a number of symbol that are designed to be used with models that are known to exist but which for legal, commercial or technical reasons are not supplied with SIMetrix itself.

3 Model Installation

This section of the training addresses a problem which you will eventually face when you need to use a device that is not available in the SIMetrix Model Library. SIMetrix Technologies Ltd supply many models with the simulator but there are many more available from component manufacturers usually available from their web sites. Having obtained a model you will want to install it.

Installing a new model is usually very easy as long as it is compatible with SIMetrix and the vast majority are compatible. This training provides for most, if not all, the problems you will encounter and cites additional resources you can use if necessary.

This section provides essential reading to clarify terminology and provide some explanations. This is followed by a set of hands-on examples of how to carry out associations of increasing complexity.

3.1 Preliminaries

3.1.1 What do we Mean Here by Model?

There are different meanings that attach to the word 'Model' in SIMetrix and they are explained in Appendix B. Our usage throughout this section is that it is a representation of a device.

SIMetrix is compatible with the vast majority of 'SPICE' models supplied by manufacturers. These are supplied as plain text files.

Models come in two types, Primitive and Subcircuit and, apart from any text that is commented out, they identify themselves as follows:

- In Primitive the first line of the text begins with the string ".MODEL" (not case sensitive).
- In Subcircuit the first non-comment line of the text begins with the string ".SUBCKT and the last line is ".ENDS" (not case sensitive).

In this learning experience, these terms will be explained in the context of examples.

3.1.2 Important Terminology

- **Model**. A model is a block of pure ASCII 8-bit text (**not** UNICODE or anything containing tags) which **does** contain specific identifiers. A text file may contain one or many models. Model files typically have the extension .mod, .cir, .lib or .spi but these are by no means the only extensions in use. SIMetrix does not care what the model file extension is.
- **Symbol**. This is the graphical representation of a model. SIMetrix supplied with a library of symbols and many may be used with models that you may download from a manufacturer.
- **Association**. This answers the question, "I have a model; what symbol should I use to represent it on a schematic?" The same symbol may be used for a wide range of models. For example, there are many different types of NPN transistor (for example, 2N2222, BC547) but all have the same symbol.

  The link between model and symbol is known as an association. SIMetrix stores these in two files with the extension .cat. The following describe the various methods of defining associations.

3.1.3 Association Strategies

SIMetrix employs a number of 'association' strategies, each one being deployed if the preceding one failed to make an automatic association. The full picture is, starting with the simplest:

- **Built-in Association**. SIMetrix looks up the part number in an internal database that is supplied with SIMetrix (This database is the 'All.cat' file in the support folder). The 'All.cat' database contains a list of known part numbers with an appropriate SIMetrix symbol name for each. If the part is found in that database then association is complete.
- **Implicit Association**. Always and only used with primitive models declared by a standard SPICE statement in standard syntax. The term **.MODEL** (not necessarily in caps) is the first token on a line. The second token is the name of the device and the third is a code identifying its type ('d' for diode etc.).

- **Embedded Association**. Association information is supplied within model itself using a special syntax which is specific to SIMetrix. Its unique characteristic is that it contains the string ***#ASSOC.**
- **Simulated Association**. SIMetrix performs a sequence of simulations to match the type of device and pinout. This is analogous to being told, "Here is a device with three pins but the part number has been rubbed off. Go and find out what type of device it is".
- **Manual Association** when all else fails. In this process you must tell SIMetrix what symbol to use

3.2 Procedures for Installation

In the following sections we will work through the various scenarios that you may encounter when installing a model. The examples are intended to give you an understanding of what is happening behind the scenes when you install a model. The actual procedure is often to simply drag and drop the model file to the command shell then select the model from the **Place | From Model Library…** menu. But, it is possible for problems to occur as some models available from third parties may not be correctly designed. By understanding the inner workings you will be better equipped to resolve those problems

3.2.1 Installation - the First Step to All Examples

1. Ensure that you have SIMetrix running and that the whole of the Command Shell is visible.
2. Locate the relevant model file in My Computer or Windows Explorer.
3. Drag and Drop it into the body of the Command Shell.
4. Click OK to confirm. Observe that a 'Completed' message appears in the Command Shell.
5. On the Schematic Editor go to **Place|From Model Library...** and click on * Recently Added Models * at the top of the left pane
6. Click on the device you have just installed. Notice whether a symbol appears in the bottom left hand pane. If so you are in for an easy ride. If not, a message appears saying, "SIMetrix does not know what symbol to use for this model. Click Place to resolve". Click Place in either situation.

3.2.2 Example of Built-in Association (AD624.mod)

1. Carry out the procedure at Installation - the First Step to All Examples immediately above using the AD624.mod file that you will find in **My Documents\SIMetrix\Training\Section-3**.
2. In **Place|From Model Library...** (step 5 above) select the Device and click Place. On the Schematic Editor a shadow of the symbol appears at the cursor point.
3. Left click and the job is done.

The AD624 model is already listed in the ‘All.cat’ database supplied with SIMetrix and so the installation of this model is straightforward.

3.2.3 Example of Implicit Association (STL_Diode.mod)

This is an example of a primitive model which is associated using implicit association. It is shown as Figure 3-1.

1. Carry out the procedure at Installation - the First Step to All Examples on page 30, using the STL_Diode.mod file that you will find in **My Documents\SIMetrix\Training\Section-3**.
2. In **Place|From Model Library...** (step 5) select the Device and click Place. SIMetrix defaults to a standard junction diode and a shadow of the symbol appears at the cursor point.
3. Left click and the job is done.

The STL_Diode model is a fictitious part and is not in the ‘All.cat’ database. But SIMetrix is able to figure out what type of device it is because the .MODEL statement refers to diode parameters as identified by the ‘d’ after the model name. But it is not able to determine what type of diode it is so defaults to a junction diode symbol. If it is actually a schottky diode for example, you can change this by editing the association using the command shell menu **File | Model Library | Associate Models and Symbols**

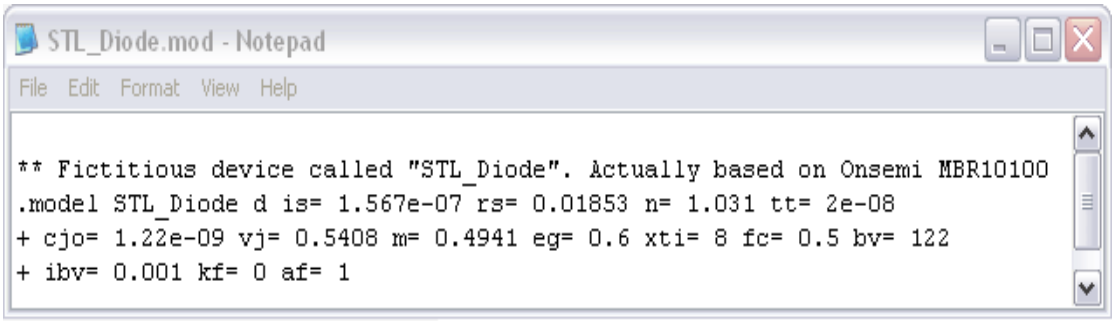

Figure 3-1. A complete primitive model

The only non-commented line starts with the line "**.model**". There are also two line-continuation characters "+" at the beginning of subsequent 'lines'. The values are picked up by the program and plugged into the relevant equations.

3.2.4 Example of Embedded Association (STL_OPAMP_EMB.cir)

This is an example of embedded association. The *#ASSOC line in the model file defines this.

1. Carry out the procedure at Installation - the First Step to All Examples, using the file STL_OPAMP_EMB.cir file that you will find in **My Documents\SIMetrix\Training\Section-3**.
2. Select the Device and click Place. SIMetrix defaults to a standard op-amp symbol. A shadow of the symbol appears at the cursor point.
3. Left click and the job is done.

Open STL_OPAMP_EMB.cir in a text editor (using a monospaced font) and you will see a file which begins as shown in Figure 3-2:

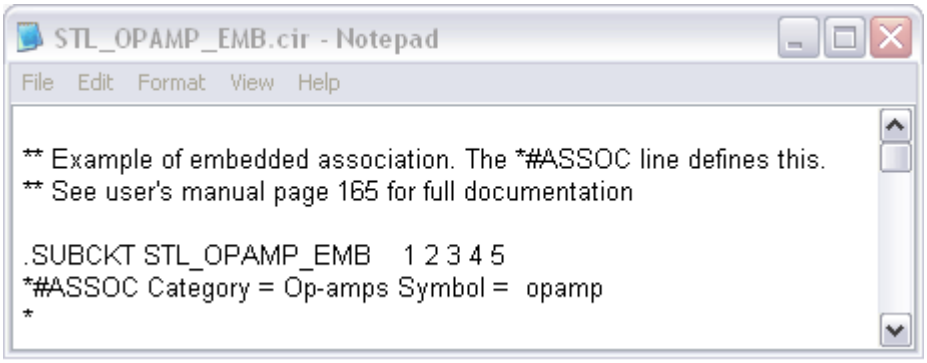

Figure 3-2. The opening lines of STL_OPAMP_EMB.cir

SIMetrix does not have a built-in op amp. Instead this is a subcircuit of other components which make the model behave like an op amp. Such a circuit is known as a 'macro model'. Op amp manufacturers do not usually supply full transistor level models for op amps to protect their intellectual property.

A 'macro model' will typically simulate the published behaviour of the device but will not usually correctly model the device when operated outside of its expected operating region.

3.2.5 Example of Simulated Association (STL_MOSFET.spi)

Carry out the procedure at Installation - the First Step to All Examples using the STL_MOSFET.spi file that you will find in **My Documents\SIMetrix\Training\Section-3**. A notice appears saying, "SIMetrix does not know what symbol to use for this model. Click Place to resolve."

When you click Place there is a short delay (a few seconds) while SIMetrix carries out a series of simulations to find out what the device is.

Like the STL_Diode part, STL_MOSFET is a fictitious part and is not listed in the 'All.cat' database. Unlike STL_Diode, it is not defined using a primitive .MODEL statement but instead is a sub-circuit

definition. Sub-circuits do not convey any information about the type of device that it represents. So, in this situation, SIMetrix uses a 'Simulated Association' to determine what type of part it is using a series of simulations. Simulated Association is not guaranteed to work but SIMetrix errs on the side of caution. If it is not completely certain then it will say that it cannot identify the device.

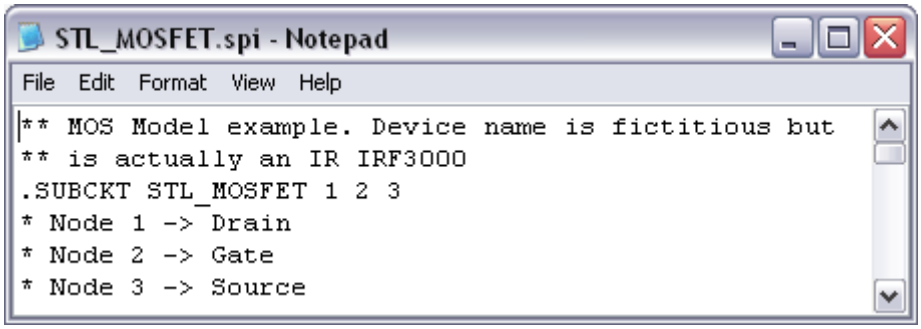

Figure 3-3. The symbol data in the STL_MOSFET model

3.2.6 Manual Association using Auto-generated Symbol (STL_INSTAMP.mod)

1. Carry out the procedure at Installation - the First Step to All Examples using the STL_INSTAMP.spi file that you will find in **My Documents\SIMetrix\Training\Section-3**. The 'Associate Symbol with Model ...' dialog opens as Figure 3-4.

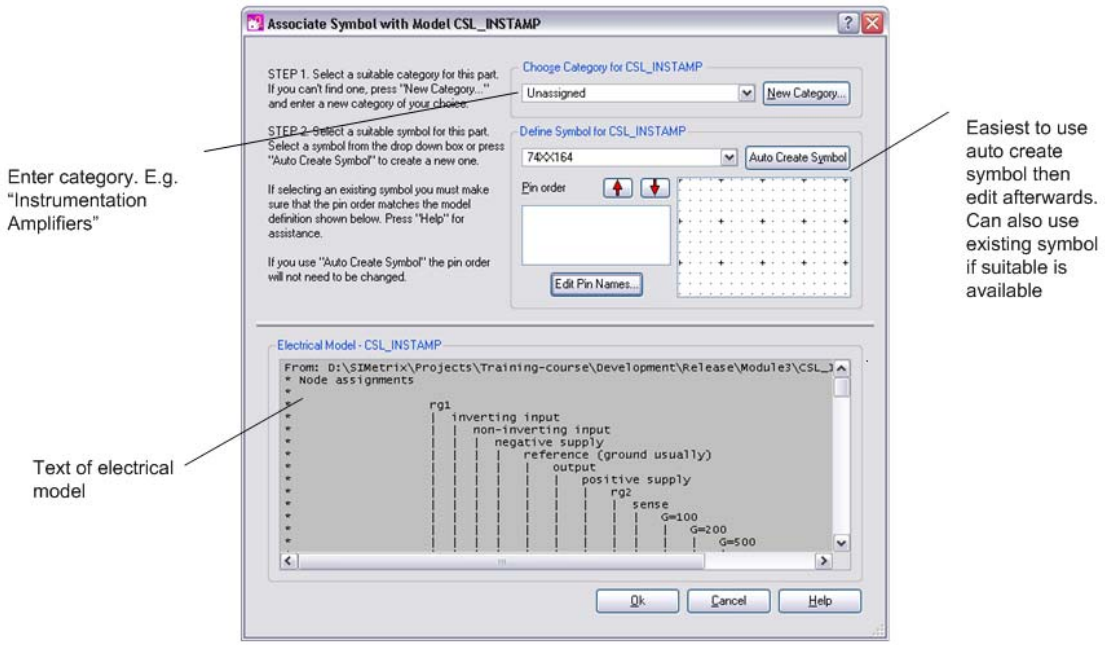

Figure 3-4. The Auto Create facility on the Associate Symbol dialog

2. We know that in this case we are installing an instrumentation amplifier. Change the category to 'Instrumentation Amplifiers'. This category is the one which will appear in the left pane of **Place|From Model Library**. If there is no suitable category, click 'New... Category..." and create one.
3. In this example there is no suitable symbol so click the 'Auto Create Symbol' button. This reads the model and does two things:

- It creates a box with the correct number of connections (in this case 12).
- It labels each connection with either the literal or a cryptic abbreviation of the connection name as given in the comment-text from the model file shown in the pane below it. Where it is unable to parse the name sufficiently to make a sensible abbreviation, such as for 'reference (ground usually)', it removes prohibited characters and replaces spaces with

underscores. Comment text, if present, is always above .SUBCKT.

Notice that the Pin order pane does not scroll vertically but is multi-column.

If the model file does not contain any pin names (called 'node assignments' in this model) or if it doesn't recognise the format of the commented text, it will use the numbers which form the arguments to the .SUBCKT STL_INSTAMP identifier.

4. Click on 'Edit Pin Names'. (This symbol cannot already have been used so it is OK to Edit Pin Names). Change 'reference_ground_usually' to something shorter.
5. Click OK and a shadow of the new symbol appear at the cursor position on the Schematic Editor.

The STL_INSTAMP model, like the STL_MOSFET model is implemented as a sub-circuit and is also a fictitious part number unavailable in the 'All.cat' database. Unlike STL_MOSFET, SIMetrix is not able to use simulated association as this algorithm works only with transistors and diodes. With all the automatic strategies exhausted, the only method remaining is for the symbol association to be performed manually.

In this example of manual association, we used the Auto Create Symbol feature. When using this method, you should not change the pin order (this can be done using the up and down arrow buttons). The ability to edit the pin order is provided so that you can use an existing symbol. This is explained in the next section.

3.2.7 Manual Association using Existing Symbol (STL_OPAMP.mod)

This example illustrates how to use and edit an existing symbol which is appropriate apart from the connections not being in the same order as the model.

1. Carry out the procedure at Installation - the First Step to All Examples using the STL_OPAMP.mod file that you will find in **My Documents\SIMetrix\Training\Section-3**. The 'Associate Symbol with Model ...' dialog opens.

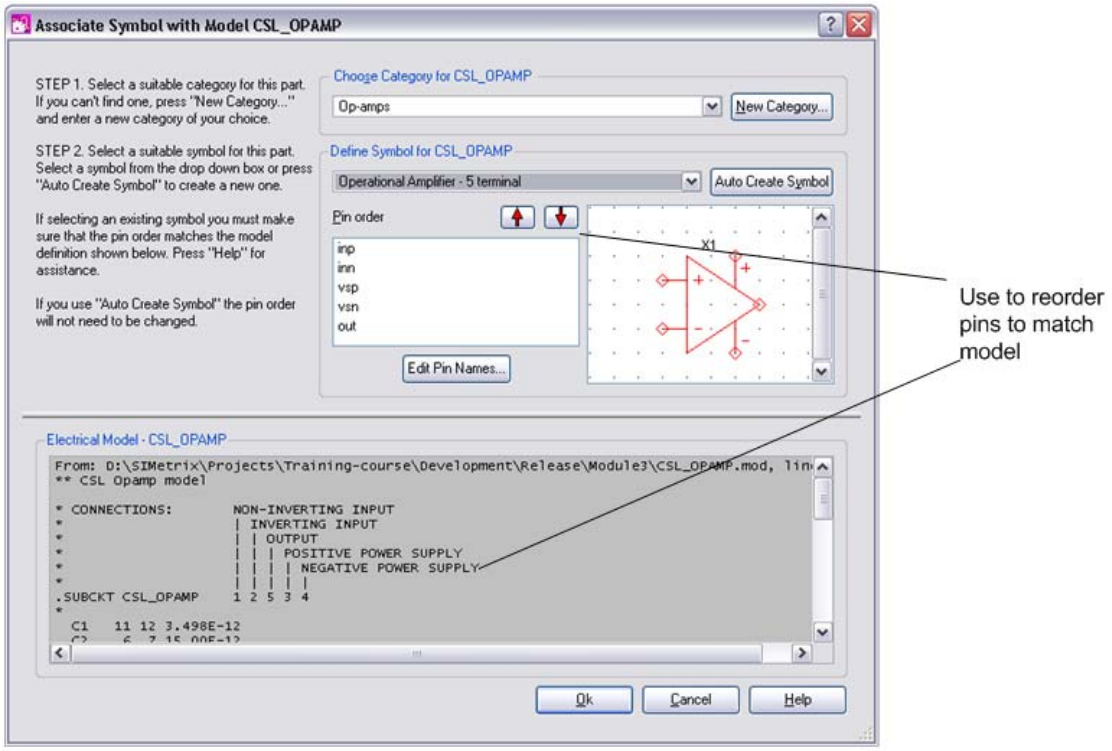

Figure 3-5. The Associate Symbol dialog showing re-order pins

2. In the 'Define Symbol for ... ' group, click the drop-down list.
3. Scroll down and select 'Operational Amplifier - 5 terminal'. This offers a standard op amp symbol.

4. Compare the cryptic pin names in the Pin Order pane with the connection names shown in the pane below it.

```
inp = input positive (non-inverting)
inn = input negative (inverting)
vsp = voltage supply positive
vsn = voltage supply negative
out = output
```

You will note that the pin order in the symbol does not match the terminal order in the model. It is essential that they do otherwise the model will not simulate correctly. The procedure below changes the pin order, but note that this only affects the model being associated (STL_OPAMP); it does not globally change the symbol pin order and so will affect other models that use this symbol.

5. Select 'out' and move it upward two places using the appropriate red arrow.
6. Change the category to 'op-amps'. This category is the one in which will appear in the left pane of **Place|From Model Library**. If you leave it unassigned it will go into the 'unassigned' category, but will otherwise be functional.
7. Click OK.

Note: The Edit Pin Names button is present on this dialog but it is not appropriate to use it here because this edits the symbol definition and editing this may affect existing schematics or/and other models associated with this symbol. If you try to change a pin name having chosen an existing symbol you will get a warning message.

Note that the device name is always the first word after .MODEL or .SUBCKT

3.3 Troubleshooting

Most problems are caused by errors in the model files or the way they were downloaded.

- Ensure that the identifiers .MODEL (Primitive) or .SUBCKT (Subcircuit) are present or the file is not in a valid format. A Subcircuit file must also have a matching .ENDS identifier at the end of the model.
- If you have downloaded the model, ensure that it contains only plain ASCII. If you simply left clicked on the link and saved what the internet browser opened, it will be an HTML file and this will not work. It is more reliable to right click the link and 'Save target as...' plain text (which is what it needs to be).
- Visit:
  www.simetrix.co.uk/site/support/kb/InstallingModels-53.htm
  or
  www.SIMetrix.co.uk/site/support/kb/InstallingModels-53.htm#Troubleshooting
  for more help with installing models or troubleshooting

3.4 Further Information and References

- To prepare a model file for distribution to colleagues or customers, see the Users' Manual Chapter 6 "Embedded Association".
- For installing and associating many models at once, see the Users' Manual Chapter 6 "Associating Multiple Models and Symbols".
- To change an association or category, in the Command Shell go to **File|Model Library|Associate Models and Symbols**.

4 Using Built-in Parts

There are two types of part you can use on a schematic. They are:

- **Named Part**. This is the straightforward situation where you place a device with a known part name. The parameters are known to the system and there is probably a model and associated symbol readily available.
- **Parameterised Part**. A parameterised part is one which is defined only by a number of parameters. An example would be transformers which are defined by turns ratio, primary inductance and such like. Some parts may also have a name but it is the parameters that you are more likely to be interested in when designing a circuit.

This section of the training addresses parameterised parts. After a brief discussion about signal sources, we will develop instances of:

- A waveform generator
- An electrolytic capacitor
- An ideal transformer
- An ideal DC transformer
- A parameterised op amp

4.1 Signal Sources

Almost every simulation needs a signal source at some point. SIMetrix distinguishes, more for clarity than anything else, between specific device types and a 'Universal Source' (which as its name implies can generate anything).

Most of the work in simulation will, however, require only three types of source - DC power supply, AC source and waveform generator and each has its own symbol in SIMetrix. The DC source looks like a battery and that is all it is, the AC source is used only for AC analysis and the waveform generator generates all the forms shown on the Edit Waveform dialog. In this section of the training we will experiment with the facilities that the waveform generator provides.

4.2 Practical Development of Parameterised Parts

4.2.1 Two Important Dialogs

Working with the Waveform Generator will involve extensive use of both the 'Choose Analysis' and the 'Edit Waveform' dialogs. Some characteristics of the two are as follows:

- **Choose Analysis**
  - To open it, go to **Simulator|Choose Analysis**; there is no default shortcut key but you can define your own. See command shell menu **File|Options|Edit Menus…**.
  - This is where you select one or more Analysis Mode(s). The following exercises all require Transient Analysis.
  - The parameters for each Analysis Mode are on the associated tab. For Transient analysis, this is where you set the stop time.
  - This dialog contains a Run button which both accepts the input and runs the simulation.
- **Edit Waveform**
  - To open it, select the Waveform Generator on the Schematic Editor, right click for context menu and select Edit Part. The shortcut key is F7 (but the Waveform Generator must be selected).
  - This is where you select the wave shape.
  - You set the parameters for the wave in the 'Time Frequency' and 'Vertical' groups on this dialog.
  - There is no Run button. You have to click OK, focus returns to the Schematic Editor and you run the simulation manually (**Simulation|Run** or F9).

Both dialogs are modal; you cannot leave either of them open and return focus to the Schematic Editor.

4.2.2 Waveform Generator

4.2.2.1 Preliminary Steps

The first steps for all the experiments in this section are:

1. In the Command Shell', go to **File|New Schematic** to open a new schematic sheet.
2. In the Schematic Editor go to **Place|Voltage Sources|Waveform Generator** (or press W).
3. Add a ground symbol to the bottom (press G) and a Fixed Voltage Probe (press B) to the top.
4. Go to **Simulator|Choose Analysis:**
   - Select the Transient check box and associated tab.
   - Set the Stop time to 100uS
   - Click OK

You can leave this schematic in place for all the experiments in this section.

4.2.2.2 Square Wave

1. Select the waveform generator, right click and select Edit Part to open the Edit Waveform dialog. (You can also simply double click the waveform generator)
2. Set values to:
   - For Wave shape click Square
   - The 50% duty cycle is fixed, vertical dimensions - baseline (zero) labelled 'Initial' on the dialog and top of the waveform is the top (1) labelled Pulse on the dialog.
   - Set the frequency to 20k. Note that the period automatically changes to 50u. In the dialog the period or frequency are two representations of the same thing. If you change one, the other one changes to match. Click OK.
3. Click Run. This produces a curve like the red curve shownas Figure 4-1.
4. Leave the Waveform Viewer window open, double click or select +F7 to open Edit Waveform and set the Offset to zero. Note that the Initial and Pulse values change to -500m and 500m respectively. Initial/Pulse and Offset/Amplitude are different ways of defining the same quantity. Click OK.
5. Click Run. This adds the green curve which swings around zero by 500mV as shown as Figure 4-1.

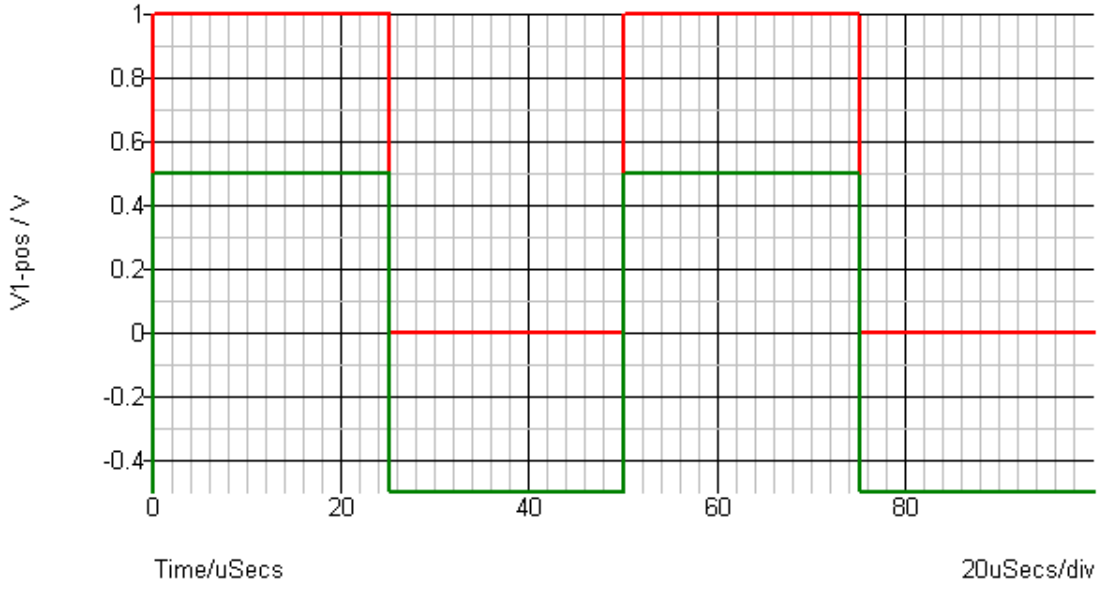

Figure 4-1. Square Wave

4.2.2.3 Triangle Wave

1. Close the waveform viewer window.
2. Select the generator, right click and select Edit Part to open the Edit Waveform dialog.

3. Set values to:
   - For Wave shape click Triangle
   - Leave the previous settings and click OK
4. Click Run. This produces a curve like the red curve in Figure 4-2.
5. Leave the Waveform Viewer window open, open Edit Waveform and edit the Duty Cycle to 80%. The Duty cycle is the ratio of the rise time to the total period. Click OK.
6. Click Run. This adds the green curve as shown in Figure 4-2.
7. Leave the Waveform Viewer window open, open Edit Waveform and edit the Duty Cycle to 100%. Click OK.
8. Click Run. This adds the blue curve (effectively a sawtooth) as shown in Figure 4-2. Be aware that this is not exactly 100% as it is not possible to have a zero fall time.

Note: In the Vertical group on the Edit Waveform, Initial is the low point and Pulse is the high point. In this particular context these terms are misleading. They are present here to be consistent with the square wave where the terms are applicable. The same applies to Offset and Amplitude. Offset is the centre line and Amplitude is the peak-to-peak voltage swing.

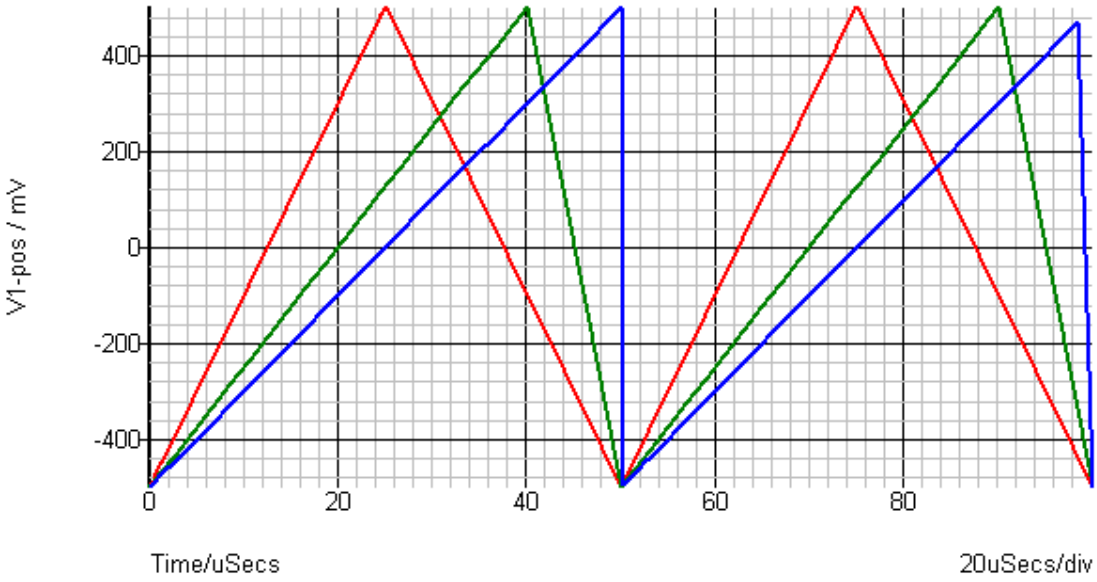

Figure 4-2. Triangle Wave

4.2.2.4 Sawtooth

Edit Waveform has a sawtooth option. Sawtooth sets the duty cycle to 99.9%. 100% is not possible as this would imply a zero fall time.

4.2.2.5 Sine Wave - illustrating delay (phase shift) and interpolation options

1. Close the graph window
2. Open a new tab in the Schematic Editor and do **Place|Voltage Sources|Waveform Generator** (or press W).
3. Place a Ground symbol (or press G) to its lower pin and a Voltage Probe on its upper pin. (**Place|Probe|Voltage Probe** or press B.)
4. On the Schematic Editor go to **Simulator|Choose Analysis** and select Transient. Set Stop time to 400uS.
5. Click the Waveform Generator and press F7 to call up Edit Waveform. Select Sine. Accept default Period and Frequency. Ensure that Delay is set to zero and that 'Off until delay' is unchecked. Click OK.
6. Run the simulation (**Simulator|Run** or F9.) and leave the curve on screen. It should look like Figure 4-3.

Note: In the Vertical group on the Edit Waveform, Initial is the low point and Pulse is the high point. In this particular context these terms are misleading. They are present here to be consistent with the

square wave where the terms are applicable. The same applies to Offset and Amplitude. Offset is the centre line and Amplitude is the peak-to-peak voltage swing.

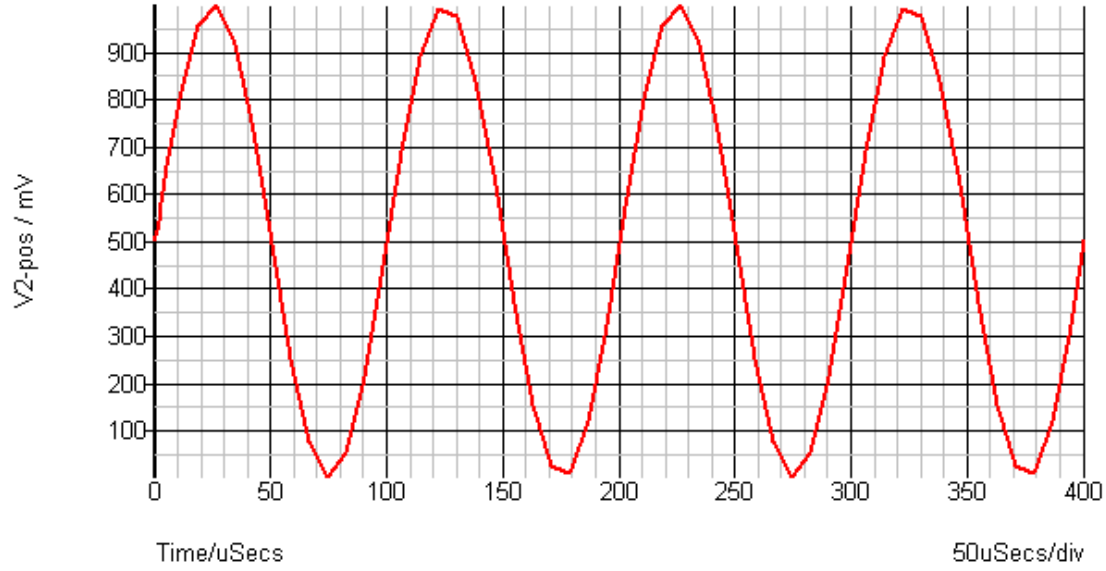

Figure 4-3. Sine wave using default maximum time step (8us)

You will notice that the peaks and troughs are not smooth. This is because of two things, the way SIMetrix joins up points and the setting of maximum time step. Points are joined using straight lines (linear interpolation) and the time step is set to be as large as possible. There are various ways of coercing the simulator to use smaller time steps and one such method is discussed below.

We will now address the time step question and produce a smoother curve.

1. On the Schematic Editor, open the Choose Analysis dialog (menu **Simulator|Choose Analysis…**) and click the Advanced button. You will see the default Max time step is 8uS and the default box is checked. Uncheck the Default box and set Max time step to 1uS. Click the Close button.
2. Run the simulation again and this plots curves such as Figure 4-4.

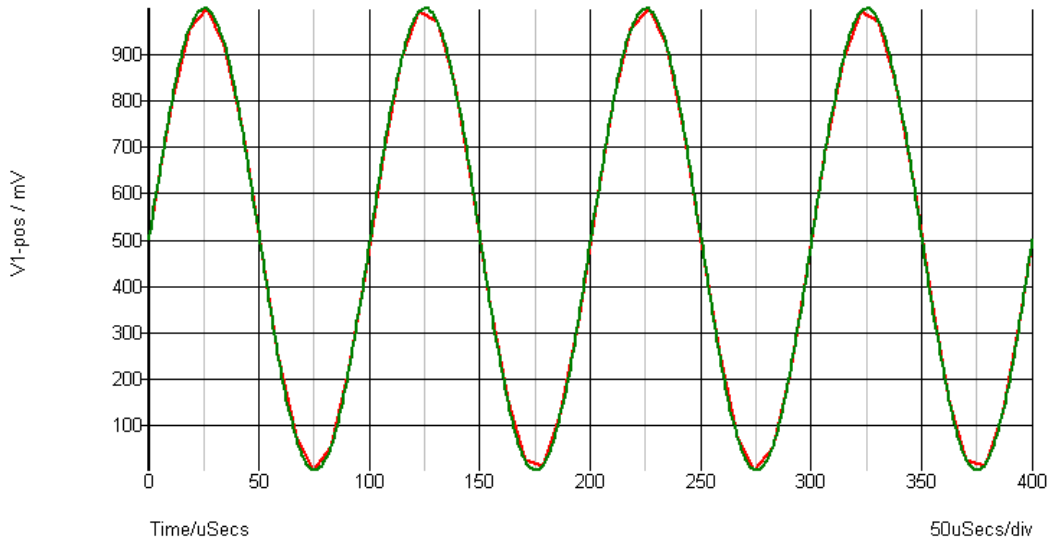

Figure 4-4. Sine wave using smaller time step (1us)

The new green curve, with 1uS time step, is much smoother at the peaks and troughs. But the simulator is using shorter time steps and as a result the simulation will take longer. Note that we set the *maximum* time step here and that the actual time step that the simulator uses is automatically adjusted may be smaller.

4.2.3 The Delay Feature

We will now look at the 'delay' feature.

1. On the Schematic Editor click the Waveform Generator.
2. Click the Use delay and Use Phase radio buttons and observe that the field label changes in accordance with which radio button is selected. These are two different ways of specifying the same thing. With the existing settings, a 50uS delay amounts to the same thing as a phase shift of 180 degrees and a delay of 25uS could be called a cosine.
3. Ensure that the Use delay radio button is checked and enter 50uS in the delay field. Check the Off until delay box.
4. Run the simulation again. It should look like the Figure 4-5 where the blue curve represents the delay. The delay (if enabled) determines when the first zero-crossing takes place.

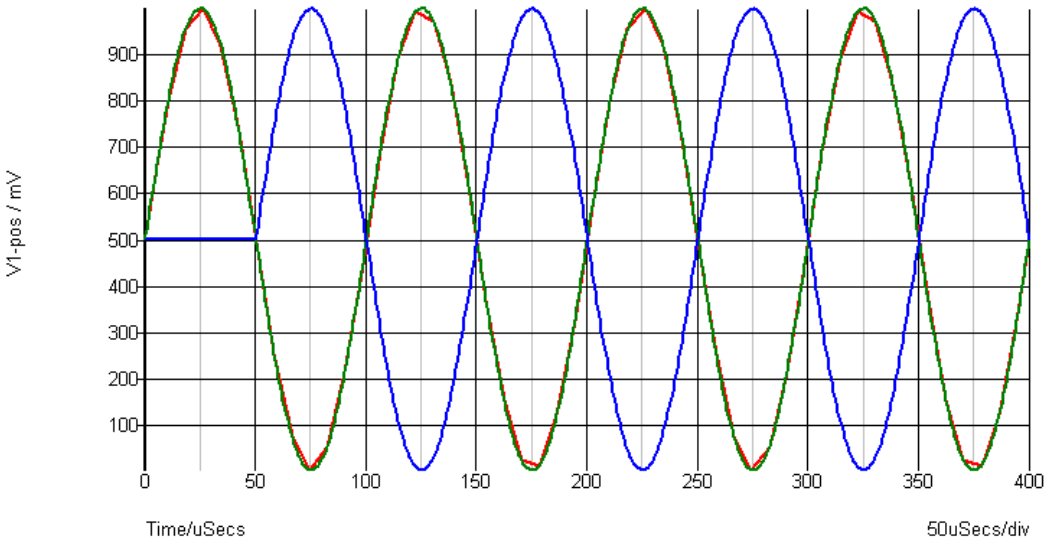

Figure 4-5. Sine wave with 50uS delay

4.2.3.1 Pulse and One Pulse

This topic illustrates rise and fall time limitations

1. Close the graph window
2. Open the Choose Analysis dialog and set the Stop time to 500uS.
3. Open the Edit Waveform dialog and set the parameters as shown in Figure 4-6 but do them in the following order (if you don't follow this order it makes some decisions itself):
   - In the Wave shape group, select Pulse
   - Ensure that Equal rise and fall is checked
   - Check the Default rise and fall box
   - Set the Duty (cycle) to 40%
   - Set the Delay to 75u
   - Set Offset to zero

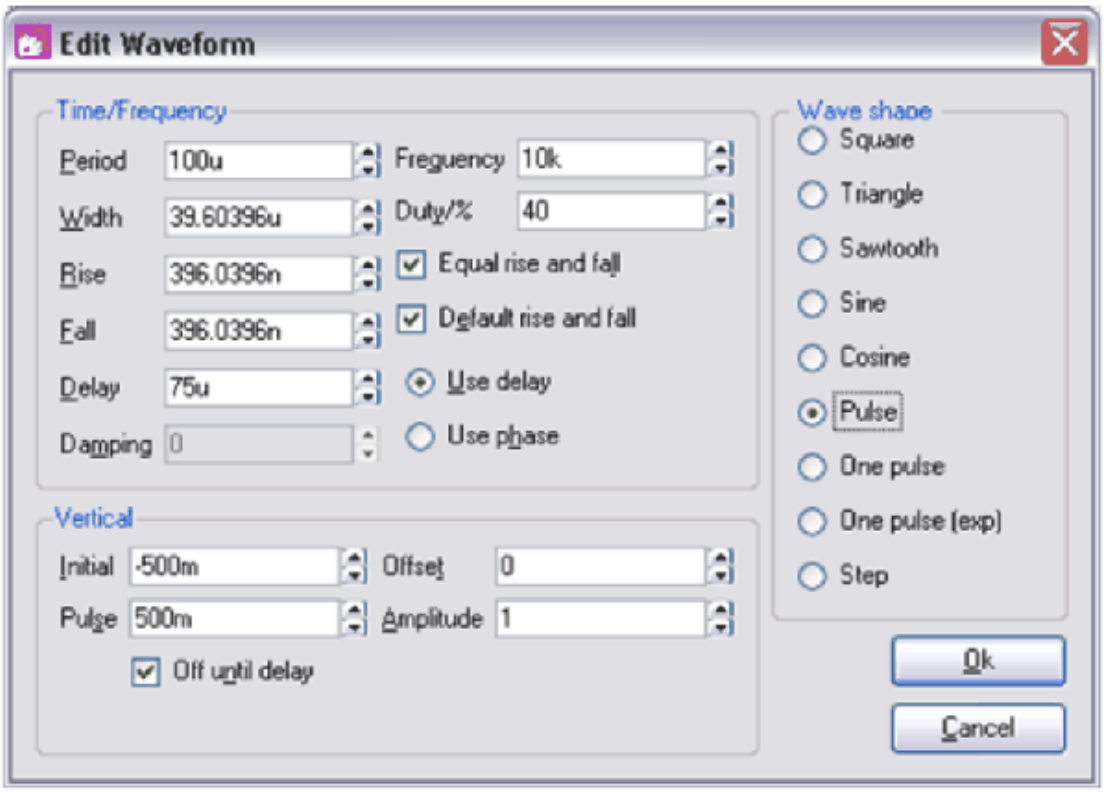

Figure 4-6. Edit Waveform Dialog for Pulse

4. Click Run and a curve similar to Figure 4-7 appears.

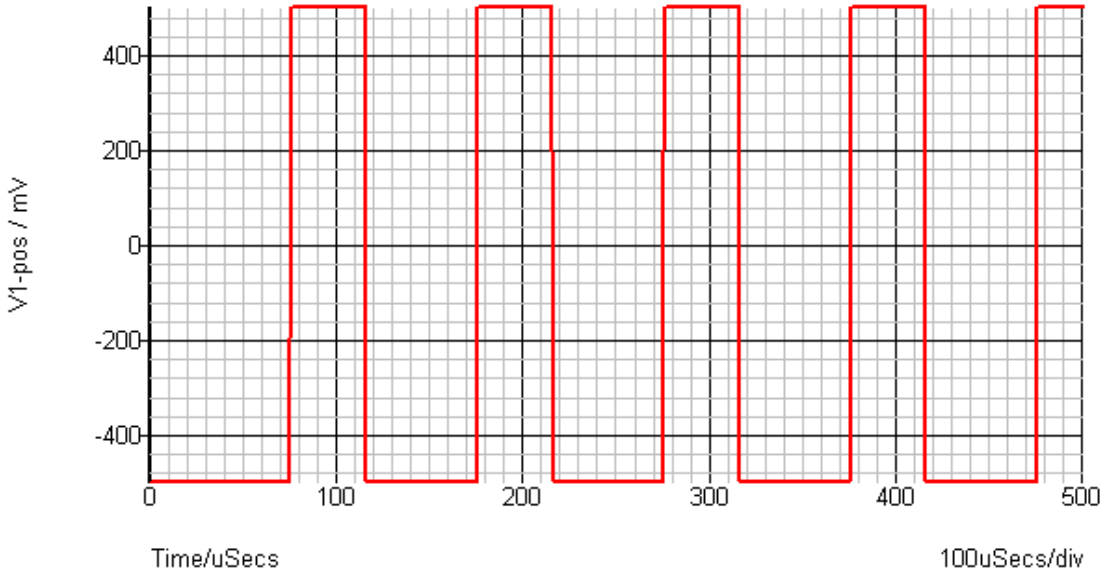

Figure 4-7. Pulse

On the screen curve, rubber-band select a rectangle as narrow as possible around a rise time and the scaling will expand to something like Figure 4-8.

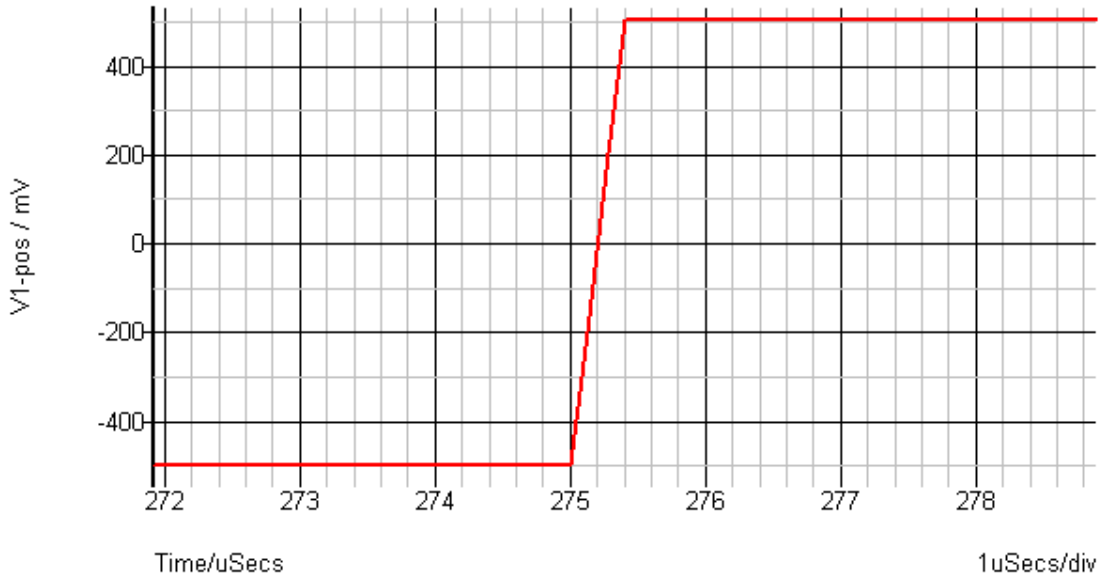

Figure 4-8. Pulse showing finite rise time

If you check the Default rise and fall button on the Edit Waveform dialog it will make the width 1%. The duty cycle starts and ends at the mid-voltage point of the rise and fall times. In this dialog various fields are interacting with one another (Width, Duty cycle, Rise and Fall) but the parameter that you set stays as you set it. It calculates the others to suit.

You may like to try the One pulse option. As the name implies, this produces a single pulse instead of a repeating pulse waveform.

4.2.3.2 One Pulse (exponential)

'One Pulse (exponential)' uses the Time Constant (TC).

1. Close the graph window.
2. Open the Edit Waveform dialog. Set the Pulse Width to 200uS, the Rise TC to 10uS and the Wave Shape to One Pulse (exp).
3. Run the simulation and see a curve like Figure 4-9.

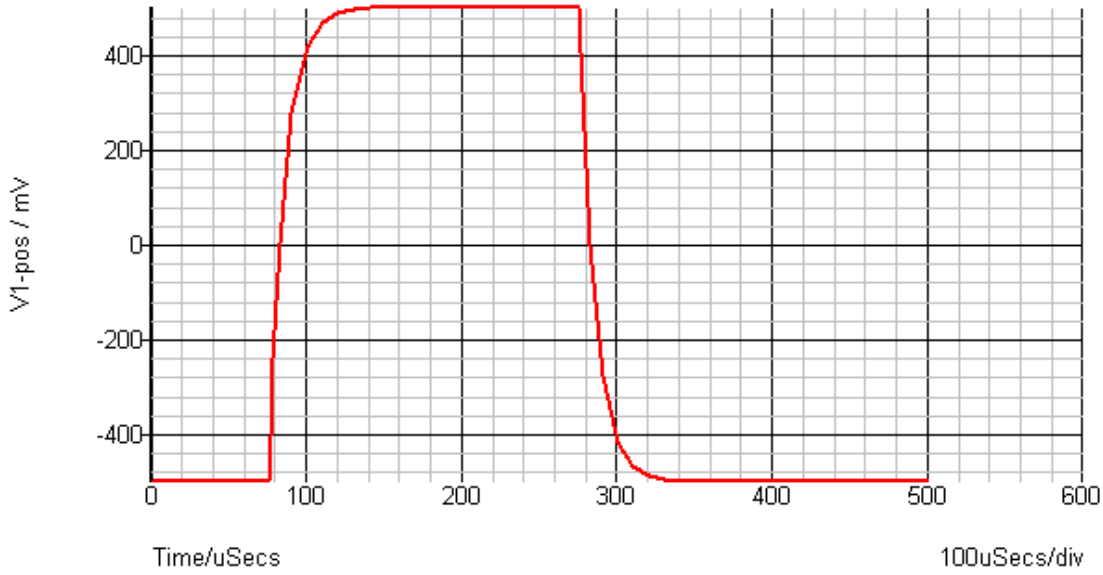

Figure 4-9. One Pulse (exponential)

You may like to try the Step option. This gives a single rising edge. To get a falling edge, reverse the initial and Pulse values.

4.3 Electrolytic Capacitor

The standard capacitor uses an ideal model and does not include imperfections such as resistive losses and terminal inductance. For many non-polarised devices such as polyester and polypropylene devices, this idealised model is satisfactory in many applications.

However, for electrolytic capacitors this ideal model is not satisfactory. To overcome this problem, SIMetrix is supplied with an electrolytic capacitor model and there are two versions of this: Simple and one Detailed. Here we will use the simple model. (Currently the electrolytic capacitor uses the US symbol consisting of a curved negative plate and a straight positive plate) .

Although this model was designed for use with electrolytic capacitors, it can also be used to model other types of capacitor in applications where series or shunt losses are important.

1. Close the graph window and open a new schematic sheet
2. On the Schematic Editor go to **Place|From Symbol Library|Passives|Electrolytic Capacitor Level 1-3**. This, the simple model, is shown as Figure 4-10.

There are three 'levels' of sophistication of the simple model. Level 1 models capacitance and Leakage Resistance but not ESR or ESL, level 2 brings in ESR and level 3 brings in ESL. These levels are indicated in Edit Device Parameters (select and F7). The benefit of using a lower level is that there are fewer internal components and the simulation is quicker. Initial condition if enabled with the Use Initial Condition check box, sets the initial voltage of the capacitor. If Initial condition is disabled, the capacitor voltage will settle according to the DC conditions.

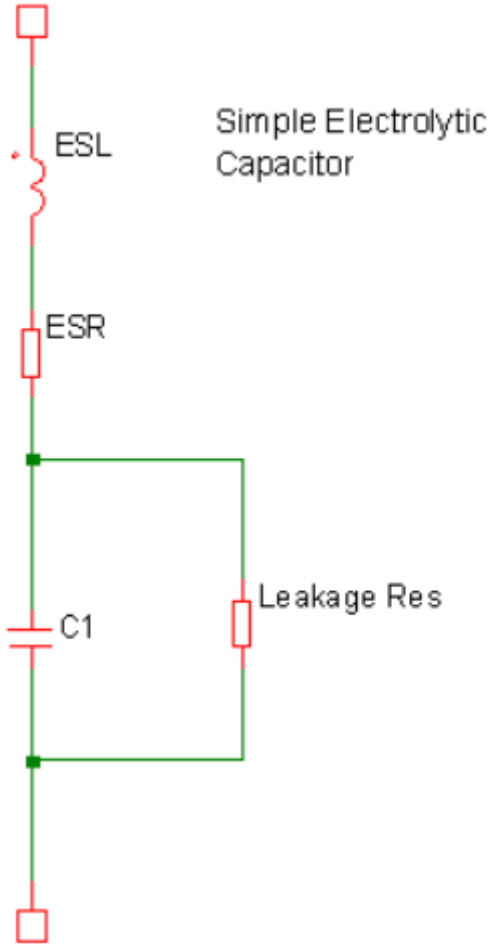

Figure 4-10. Simple Model of an Electrolytic Capacitor

The Detailed model brings into play the fact that ESR and Leakage Resistance both have different values at low and high frequencies.

Level 4 brings in ESR LF and ESR HF, level 5 brings in everything. ESR is quite an important design parameter. If you are using the device in a circuit where it might see both high and low frequencies, you may need to include both of them. The Detailed model is shown as Figure 4-11.

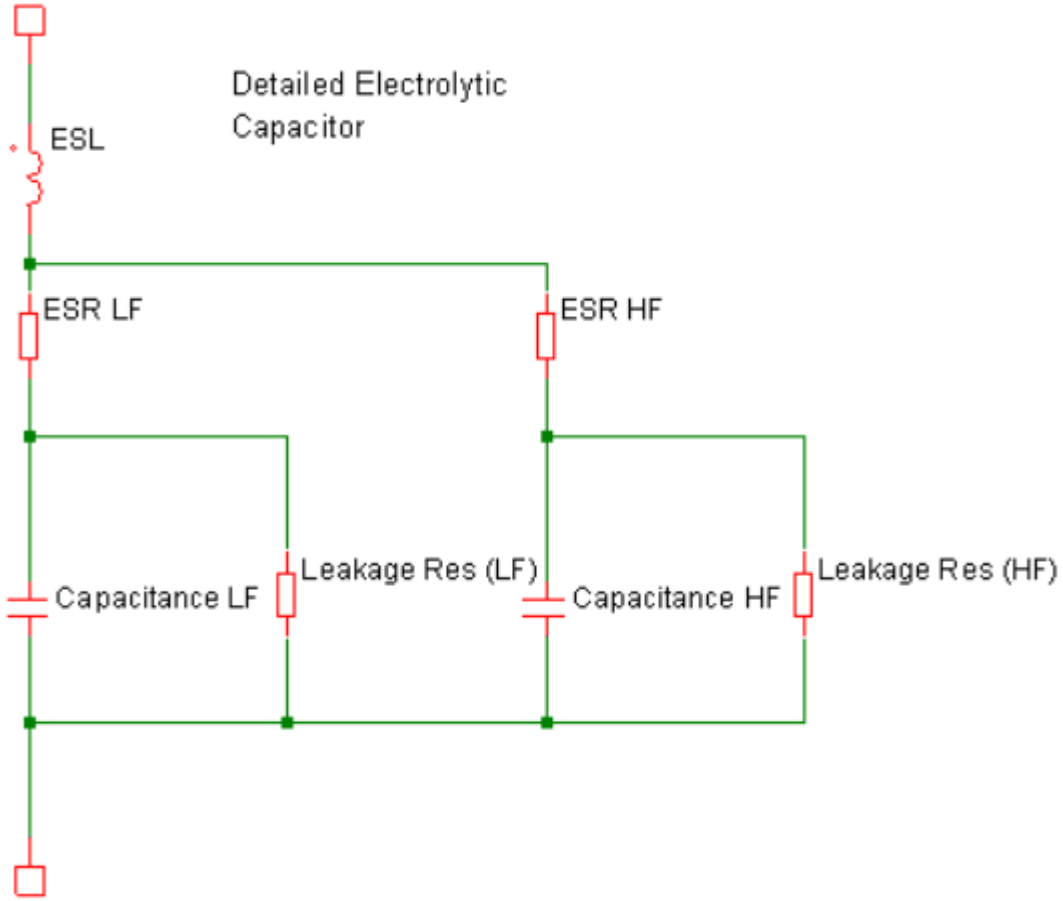

Figure 4-11. Detailed Model of an Electrolytic Capacitor

4.4 Ideal Transformer

The Ideal transformer models the voltage-current transfer of a transformer and the inductive behaviour. All other characteristics are ideal. That is, there are no series and shunt losses, no saturation and no hysteresis. The inductance is ideal and linear.

There is also a saturable transformer model that includes saturation and hysteresis effects.

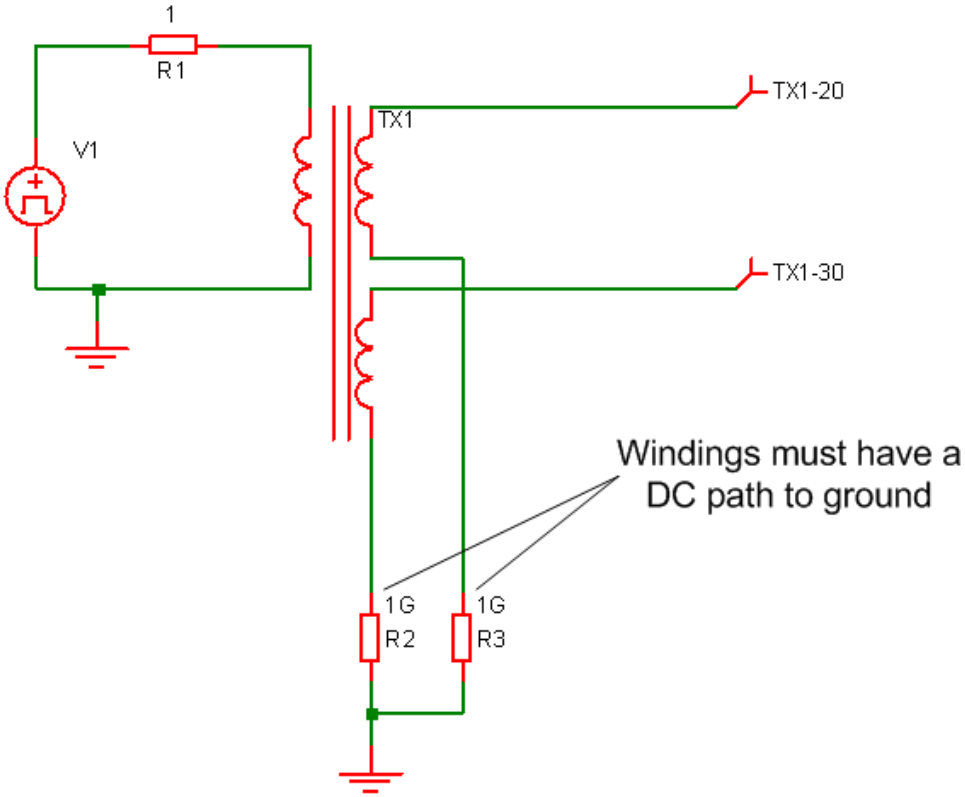

Figure 4-12. The Ideal Transformer Circuit

The circuit is for a one-primary, two-secondary ideal transformer shown as Figure 4-12. Note that because these windings have no resistance a resistor must be added to the primary circuit for the simulation to run. Also the secondary requires a DC path to ground provided by R2 and R3. We will enter the above circuit and work through how to set up the transformer.

1. In a new schematic sheet, open **tx.sxsch**. You will find this file in **My Documents\SIMetrix\Traning\Section-4.** You should see the above circuit but missing the transformer.
2. Go to **Place|Magnetics|Ideal Transformer**. This immediately calls up the dialog asking you to define it. This differs from the placement of many devices where you would place the device on the schematic then subsequently edit it.
3. In the Configuration group, specify one primary and two secondaries. You can specify up to 20 primaries and secondaries.
4. In the Define Windings group, click the drop-down box entitled Select winding. Notice that there are two entries

   Sec. 1: 1
   Sec. 2: 1

   These two entries define the turns ratio for secondary 1 and secondary 2 relative to the primary. Currently they are both set to 1.
5. Set the number of primaries to 2, then click on the drop down box again. Note that there are now three entries:

   Prim. 2: 1
   Sec. 1: 1
   Sec. 2: 1

   The first entry is the turns ratio for primary 2 with respect to primary 1.

6. Set the number of primaries back to 1. (Step 5 above was simply to demonstrate the meaning of the entries in the windings definition drop down box. Many users misunderstand this)
7. Select Sec. 1 and set the turns ratio to 10. Note that the drop down box entry changes to 'Sec. 1: 10'.
8. Click the drop down box and select the Sec. 2 entry. Set the turns ratio to 50 as illustrated in Figure 4-13.
9. Set the primary inductance as 1mH. The inductance for the secondaries is determined by the turns ratio. Leave the other (coupling) fields at their default values.
10. Close box then place transformer on the sheet. Your schematic should match Figure 4-12 above
11. Run simulation in the usual way. You should see two sine waves with amplitudes 5V and 25V

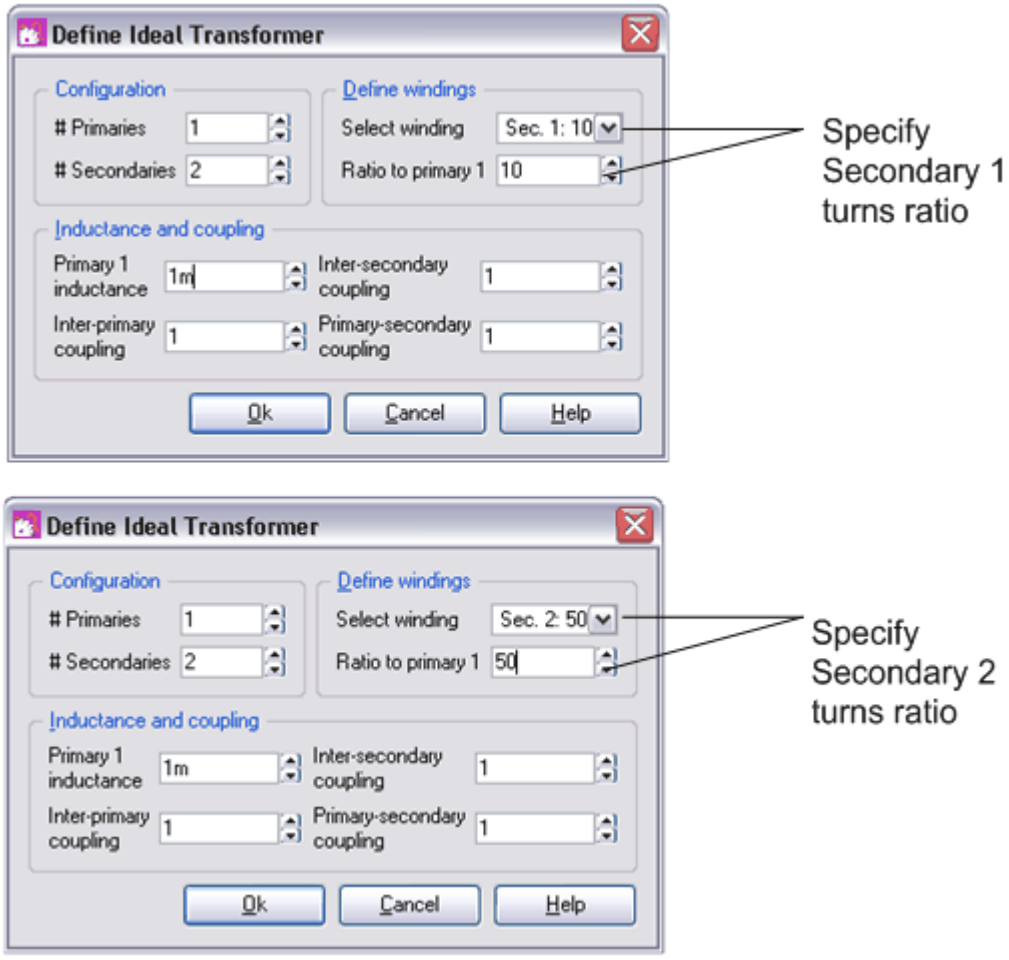

Figure 4-13. The Ideal Transformer dialogs

4.5 Ideal DC Transformer

The ideal DC transformer is the same as the ideal transformer except that it does not have any inductance – in effect its inductance is infinite.

This can be used to simulate an idealised DC transformer or may sometimes be useful to model a transformer that may ultimately be implemented using a real magnetic device, but at this stage in the design process, you do not wish to have to consider the effects of the transformer's magnetics.

4.6 Parameterised Op Amp

To work with an op amp you would typically choose one by part number and model from the Model Library . The Parameterised Op amp caters for two situations where you may not be able to do this:

- You might not have a model for the op amp but you have some manufacturer's data sheets from which you can find parameter values.
- You perhaps don't even have manufacturer's data; you are designing the circuit. You know some of the important parameters. This approach is often more useful than manufacturer's model as you know exactly what is and is not included in the model. Once the circuit is working you can find a manufacturer's part that meets your specification.

The Edit Device parameters dialog for an op amp is shown as Figure 4-14. You can access it by going to **Place|Analogue Functions|Parameterised Op Amp**, selecting it and pressing F7.

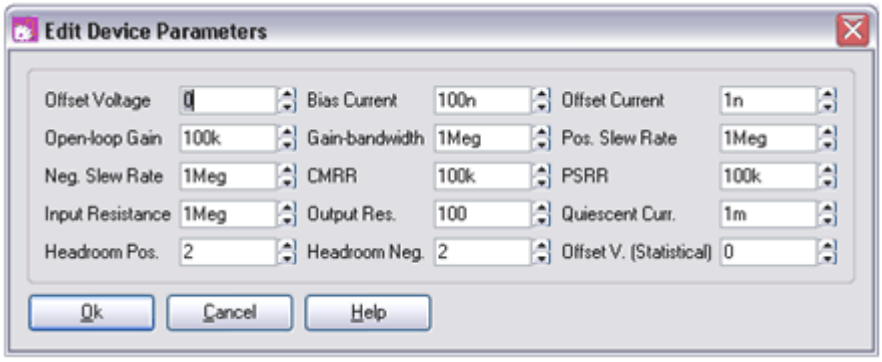

Figure 4-14. Edit Device Parameters for a Parameterised op amp

4.7 Other Components

While the simulator is intended primarily for analogue circuits, it can also model digital circuits. They include:

- ADC and DAC
- Counters
- Shift registers
- Gates
- Bus registers

5 Symbol Editor

In this section we will amend an existing symbol and also create one from scratch to specified requirements.

A standard symbol library with around 640 symbols is shipped with SIMetrix but there are a number of situations where you need to adapt an existing one or create a new one. For example

- For use as a hierarchical block which is covered under Hierarchical Schematics, Busses and Digital on page 105.
- For use with a model where there is no suitable standard symbol.
- You wish to change the graphics to suit personal taste or corporate standards.

5.1 Symbol Library

Symbols in the library are grouped into files located in support\symbollibs under the SIMetrix root. (In Linux versions they are under share/symbol-libs) Symbols have the extension .sxslb (SIMetrix symbol library). There is more information about symbols under Additional Information about Symbols, Selection and the Library Manager on page 53.

You can create your own symbol library files and place these in any location you wish. However, we advise against placing your own symbol library files in the same folder as the system symbol libraries.

5.1.1 Symbol Library Manager

From the Command Shell go to **File|Symbol Editor|Symbol Manager**. The LH pane lists all the symbol library files that are currently installed. This facility enables you to:

- Select a symbol library file and drill down to see what symbols it contains.
- Move symbols from one file to another, copy, rename and re-categorise them
- Add a symbol library file from an external source.
- Remove a file (there is no request for confirmation and no undo facility here).
- Create a symbol library file
- Move a file up or down the list (using the red buttons). You would need to do this to ensure that, if there are duplicated symbol names in other library files, the one that is placed on the Schematic Editor is the one you want. It searches from the top.

Right click on symbol in the right hand pane to see a context menu listing the features available (Rename, Copy Symbol, Delete).

5.1.2 Opening the Symbol Editor

There are various ways of invoking the Symbol Editor; here is one method:

1. On the Schematic Editor go to **Place|From Model Library** and select the AD624/AD. Place it on the Schematic Editor.
2. Select it, right click it and select **Edit Symbol** from the context menu that appears. The Symbol Editor opens with the selected symbol on it.

5.2 Gridlines

In the Symbol Editor there is a major grid and a minor grid. The major grid, represented by the small crosses, corresponds 1-to-1 with the Schematic Editor grid. On the Schematic Editor all pins and symbols must lie on the grid so on the Symbol Editor all pins will snap to a major grid point. Segments, on the other hand, will snap to the minor grid. The density with which the minor grid is displayed depends on the zoom level. At the extreme zoom-in there are 120 minor grid lines to one major grid line.

5.3 Altering an Existing Symbol

We are going alter the AD624 symbol to look more like the data sheet where it is depicted as a triangle as shown in Figure 5-1.

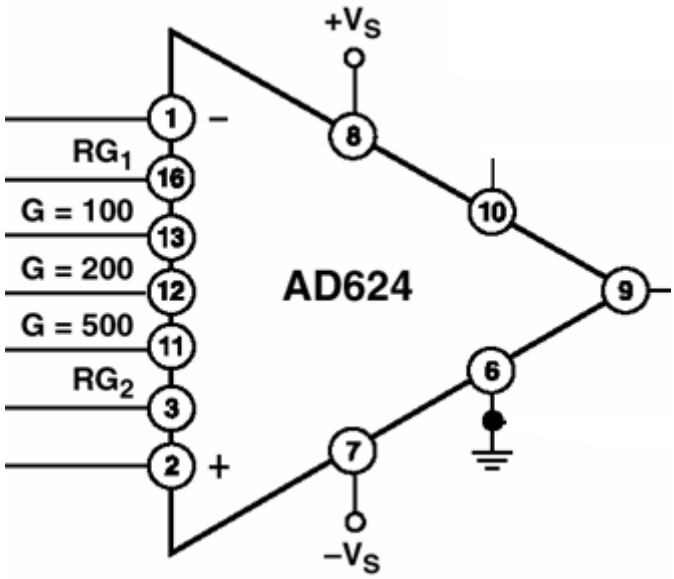

Figure 5-1. The AD624 Datasheet Illustration

1. Place the existing AD624 symbol on the Schematic Editor. You can do this using the **Place|From Symbol Library…**then navigate to Vendor → Analog → Amplifiers/Buffers/Comparators → AD624
2. Select the symbol, select **Edit Symbol** from the right-click context menu.
3. Select the single vertical segment on the right hand side of the box and press Delete.
4. Drag the ends of the top and bottom sides to form a triangle as shown in Figure 5-2.

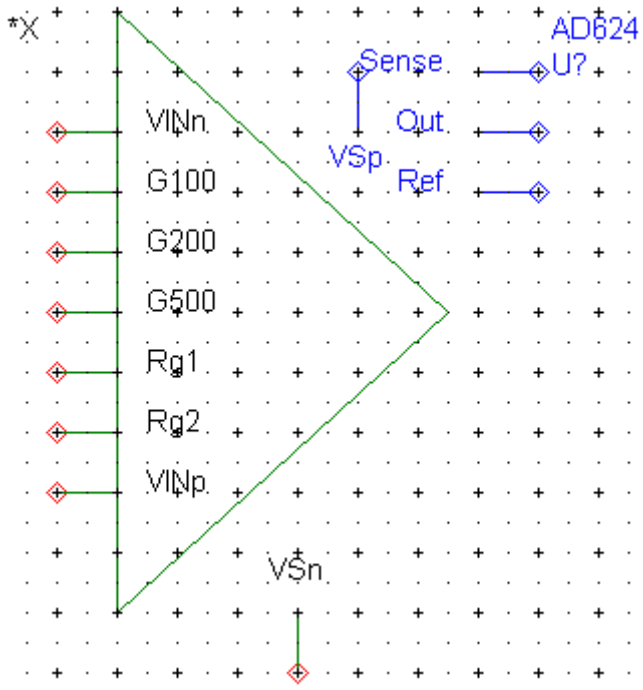

Figure 5-2. The Emerging New Shape

5. Now we move the pins. There are two elements to a pin: the point and the label. If you select the point (the diamond shape), it will move along with the label. If you select the label alone it will move independently. It is conventional to add a 'finger' to the point to join it to the symbol but such a line is just a plain segment the same as any other. To rotate an item in 90 deg steps, select first, then click the Rotate button on the toolbar (or press F5)

6. Click on the output point of the op amp and move it horizontally to get the shape you want. You should be able to find a position where the upper and lower lines which join at the output pass though major grid points.
7. We don't want the label 'Out' to appear on the Schematic Editor so we will hide it. Select the text and right click it (or F7) to call up the Edit Property Pin dialog. Check 'Hidden' and click on OK. This adds a star on the Symbol Editor but does not show at all on the Schematic Editor.
8. On the original symbol design, some of the labels were right aligned; now it makes more sense if they are aligned to the centreline of the pin. Select a label and press F7. Choose a 'Centre' justification (Top or Baseline) and click OK. You can also make minor adjustments to the position on the Symbol Editor. The final shape is shown as Figure 5-3. The text justification (or alignment) defines the position of the reference point on the text and is similar to text alignment in a word processor (e.g. left, right, centre)

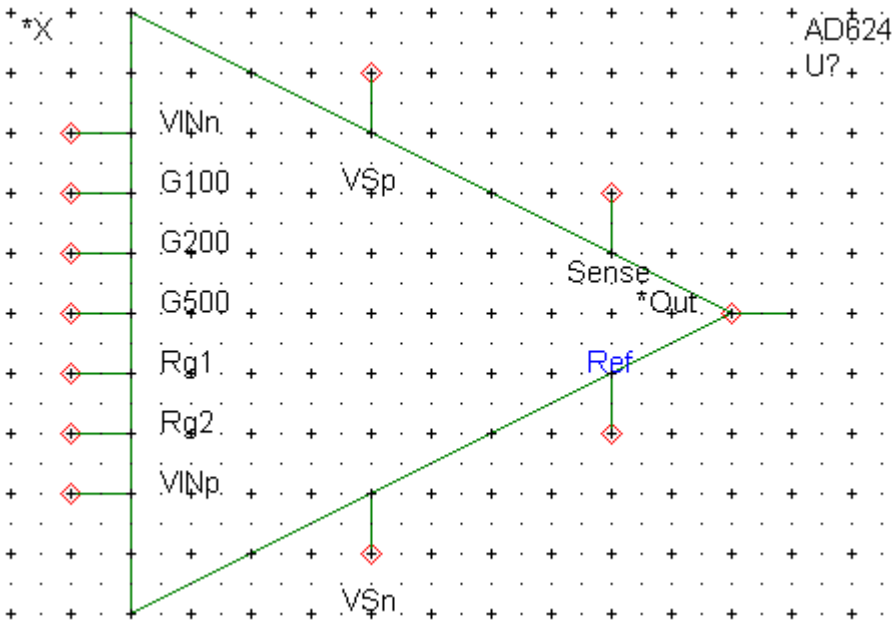

Figure 5-3. The Final New Shape

5.3.1 Saving the Symbol

If you are editing an exiting symbol it is convenient to save it to the system library overwriting the existing symbol. This is because the symbol will have all its associations in place and they will stay in place with the amended version of the symbol.

If you wish you can save the symbol as a completely new symbol retaining the original. To do this you must select a new *internal name* when you save the symbol. (The *internal name* is one of the fields in the symbol save GUI – you will see this below)

You may save the symbol to a new library file of your own. If you do this you must change the internal name otherwise there will be duplicate internal names in the library. Duplicate internal names cause complications that must be avoided.

As stated above, saving the symbol with the same internal name to a system library which contains the original will overwrite it. Actually it doesn't physically overwrite the symbol in the original library but is instead saved to a file in the application data area. The saved changes take precedence over the original so the final behaviour seems like it has been overwritten This is explained more fully in the User Manual, chapter 4, Symbol Library Manager → Editing System Symbol Libraries.

In the following procedure, we will overwrite the system symbol:

1. Go to **File|Save…**. (There is no 'Save As...' facility listed but 'Save' is effectively a 'Save As…'.)
2. Leave all the fields at their defaults. This will ensure the original symbol is overwritten.
3. Click OK.

4. Change focus back to the Schematic Editor. If you have just edited the symbol, notice the last message in the Command Shell and press Q on the keyboard. A shadow of the new symbol appears at the cursor. (This does not represent any permanent use of the Q key; it is only a shortcut to place the symbol you have just edited.)
5. If you have not just edited the symbol, go to **Place|From Symbol Library...** and select from within the categories of devices.

5.4 Create a New Symbol from Scratch

When creating a new symbol you need to think about four things:

- The symbol's shape
- Pins
- Pin order
- Properties

5.4.1 The Process

We will assume that you have a MAX4073 current sensor chip and have decided to create a symbol from scratch at the start. Let's assume that this is a device for which no suitable symbol exists. The Data Sheet is given as shown as Figure 5-4.

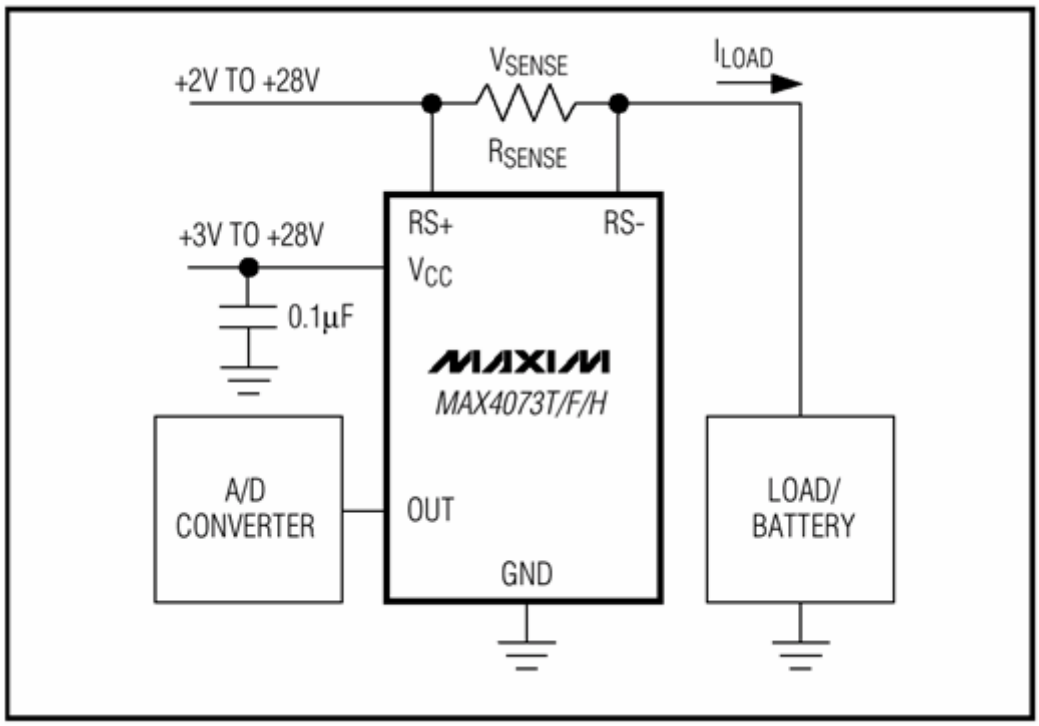

Figure 5-4. The MAX4073 Data Sheet

Work towards the symbol shown as shown as Figure 5-5. (There are two ground connections on the MAX4073 model although the data sheet shows only one.)

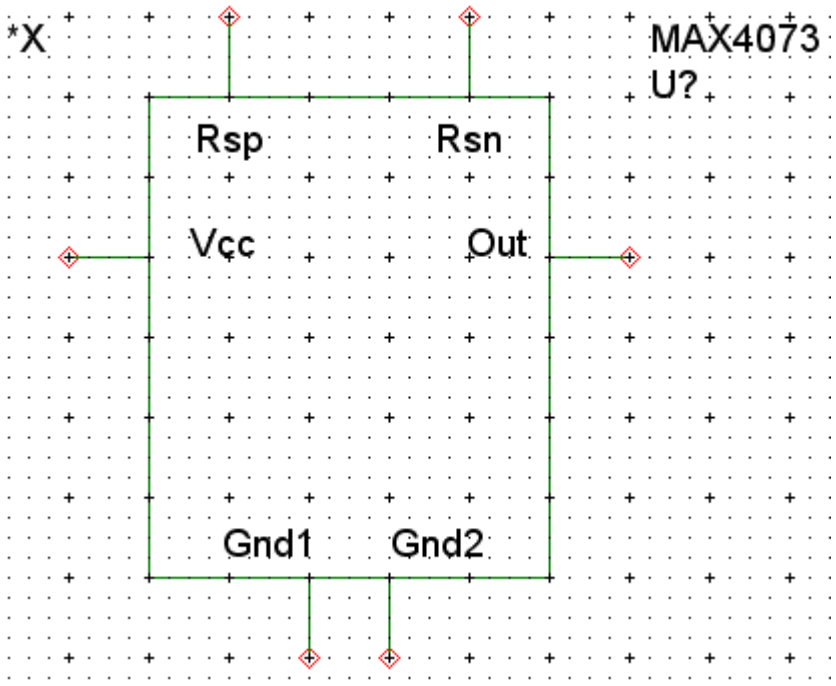

Figure 5-5. Symbol for the MAX4073

1. In the Command Shell go to **Symbol Editor|New Symbol**. A new tab opens in the Symbol Editor.
2. To draw the rectangle, click the pen icon, draw, right click to stop continuous elements, click icon again to exit the pen mode.
3. Go to **Property/Pin|Place Pin** and place the first pin in a suitable position.
4. Go to **Property/Pin|Place Pin (repeated)**, click Yes to accept the default name and place the other five pins. The Symbol Editor assigns numbers to the pins starting from 0. Having placed all the pins, you can now edit all their properties.
5. For each pin:
   - Click on the default name and press F7.
   - Edit the pin name. The pin name will accept the '+' and '-' characters but they can cause problems in some situations. They are better avoided; use 'p' for positive and 'n' for negative. Avoid starting the pin name with a number. The pin name must not contain spaces although the software will appear to allow them.
   - Choose an appropriate justification.
   - Ensure that Hidden is not checked. Accept the default font. Click OK.
   - Position the text appropriately.
   - Add the 'fingers'.
6. On the Symbol Editor, go to **Property/Pin|Edit Pin Order**. On the Select Pin Order dialog that appears, select a pin and use the Up/Down arrows to move it and arrange them as:

   Gnd1 Gnd2 Vcc Rsp Rsn Out

   This is the order of the pins in the model file and so affects how the data is presented to the simulator when you run the simulation. It avoids the need to adjust pin order later (although it is possible to re-sequence them later if necessary).
7. On the Symbol Editor, go to **Property/Pin|Add Standard Properties**. This calls up the Default Properties dialog which is illustrated (as Figure 5-6). Symbol properties are an important SIMetrix concept.

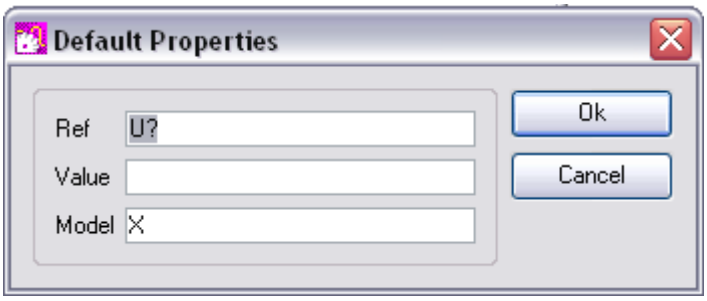

Figure 5-6. The Default Properties dialog

8. Enter MAX4073 as the Value and click OK. While the symbol is still in the Symbol Editor it should closely resemble the Symbol for the MAX4073. The question mark following the 'U' will be replaced by a sequential number when the symbol is placed on the Schematic Editor because only then will the system know what number to give it.
9. On the Symbol Editor select the Value property (the MAX4073), right click and select **Edit Property/Pin/Arc...** (or press F7). This calls up the Edit Property dialog of the property 'Value'. Leave Text Location as Auto and ensure that 'Protected' in the Property Attributes group is NOT checked. If this is checked, the value of this property cannot be edited on the schematic and you need to be able to do this. Protected means it is defined in the symbol and you can't change it in the Schematic Editor (although you still can in the Symbol Editor). Click OK.

5.4.2 Saving the New Symbol

We recommend that you save new symbols to a new symbol library stored in standard location for user symbols. (**My Documents\SIMetrix\Symbols**)

Some of the fields on this dialog may already be populated.

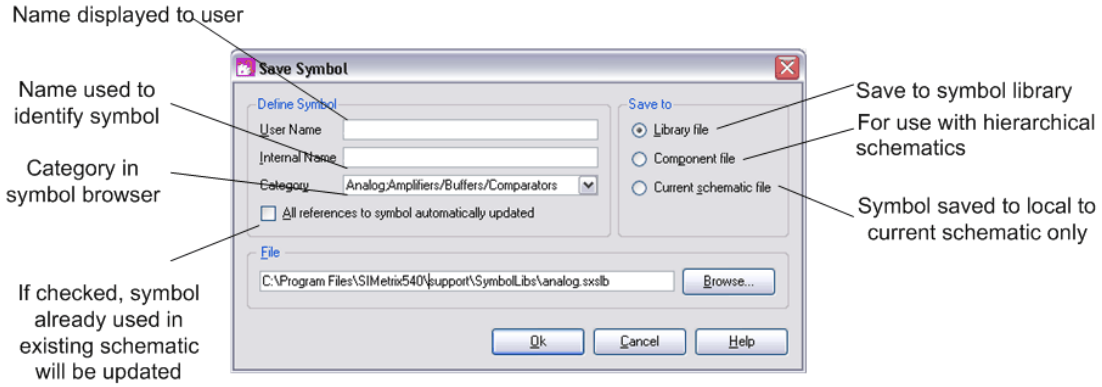

Figure 5-7. The Save Symbol dialog

1. On the Symbol Editor go to **File|Save** and first select 'Library File' in the 'Save to' group. (If you complete any other fields and then select 'Library File' the fields you have already entered are deleted.) The Save Symbol dialog is shown as Figure 5-7.
2. Ensure that 'All references to symbol automatically updated' is unchecked. If this is checked, someone else who is using this symbol will find that, when they re-open a schematic, the symbol will have changed. The Schematic Editor stores copies of the symbols it has embedded but when you open a schematic, it will look first in the Symbol Library. If a symbol with the right internal name is present, it will use that global symbol rather than the one that is stored in the schematic.

3. In User Name enter MAX4073. Notice that Internal Name mimics what you are entering. If you are creating a symbol for a numbered part, it is worth building the part number into the internal name. If you are creating a part from a generic name it's worth adding something to the internal name to make it unique (put your initials in it, for example). The category is the grouping in which it appears in the Symbol Library. The internal name is not displayed on the Schematic Editor and should be restricted to alphanumeric characters and the underscore.
4. Click browse and create a new symbol library. Click the 'User' button on this dialog and it will take you to the default directory for user symbols which is under the **My Documents\SIMetrix\Symbols**. Call it Maxim Symbols. Avoid saving to the system symbol library. We are creating a new symbol library so click OK to confirm that you want to do this.

When saved, the symbol should be available from **Place|From Symbol Library…** then Vendor → Analog → Amplifiers|Buffers|Comparators

5.5 Additional Information about Symbols, Selection and the Library Manager

The names listed in both the Symbol Library Manager and the Select Symbol dialog are not the internal names; they are the display names that will appear when you display the symbol on the Schematic Editor. When you select a symbol its internal name appears at the bottom left hand corner of the Symbol Library Manager.

There are some situations in which you need to change the search order for symbols. When you select a display name in the right hand panel of the Select Symbol dialog, SIMetrix begins searching for the corresponding internal name in the file at the top of the list in the left hand panel of the Symbol Library Manager, and works its way downwards. The system will not prevent you from having the same internal name in more than one symbol file; internal names are unique within a file. To ensure that the Schematic Editor picks up the symbol you want, in the Symbol Library Manager, move the file containing it up above the file(s) thought to contain a duplicate name of a symbol that you do not want to display.

There are no duplicate names in the system libraries. When building your own symbols, the rule is "don't make duplicate internal names".

6 Analysis Modes

6.1 Transient Analysis

We met an example of running a Transient Analysis earlier in the training in Running a Transient Analysis on page 21.

Transient analysis is the analysis mode closest to real life. In transient analysis, the behaviour of the circuit is computed from t=0 to the specified stop time. Usually the stop time is all you need to specify.

The Transient analysis parameters are:

- Stop Time (this page).
- Maximum and Minimum time step (this page).
- Simulation Tolerances (page 55).
- Integration Method (page 56).
- Sensitivity and Data Output Options (page 58).

6.1.1 Stop Time

Transient Analysis computes circuit behaviours from t=0 to a specified stop time. Stop Time is the most important of the Transient Analysis parameters.

The default values for some parameters are calculated from the stop time. For example, the default maximum time step equals the stop time divided by 50. This means that the simulator will not choose a timestep that is larger than 1/50th of the stop time.

To set a Stop Time, select menu **Simulator|Choose Analysis…** and enter a value under Stop time in the Transient parameters group. When you run the simulation it will stop after having run for the defined time. If you find that you needed to run it for longer, there is no need to start it again with a different Stop Time. You can instead go to **Simulation|Restart Transient...** and enter the increased stop time. Click OK and the simulation will not restart from t=0, it will pick up from where it left off and continue to the new stop time.

6.1.2 Pause Simulation

Whenever a simulation is running a 'Simulation Running' dialog is present on screen. This dialog carries a Pause button and Resume button.

6.1.3 Maximum and Minimum Time Steps

6.1.3.1 Maximum Time Step

At each time point the simulator makes an assessment of when it will do its next time point. To get the shortest possible run time you want time steps to be large as possible. To get the most accurate result the smaller the better. We usually need a compromise.

The simulator has some quite sophisticated algorithms internally which cause the time step to vary possibly over multiple orders of magnitude. Small time steps are used when there is much activity and detail is required while longer time steps are used when the circuit is settled.

The simulator will always keep the time step below the configured maximum time step. So, one way of improving simulation precision is to reduce the maximum time step. The default value is the Stop time divided by 50.

Tip – In many applications it is usually better to tighten the simulation tolerance rather than reduce the maximum time step. However, in very simple circuits, especially circuits without any reactive components, the tolerance settings are sometimes ineffective and in these situations, reducing the maximum time step is an appropriate method of improving simulation accuracy.

To adjust the maximum and minimum time steps, select menu **Simulator|Choose Analysis…**then click the Advanced Options… button. Enter value under Max. time step in Time step group.

6.1.3.2 Minimum Time Step

Occasionally the simulation will abort if the time step required falls below the configured minimum time step. The minimum time step defaults to 1e-9 x maximum time step. A situation can arise when the time step gets smaller and smaller and has to stop somewhere.

We will demonstrate a trivial example of this

1. Open the schematic **My Documents\SIMetrix\Training\Section-6\mintimestep.sxsch**. See Figure 6-1
2. Run simulation

Notice that it aborts with the error (displayed in the command shell) "*** ERROR *** Timestep too small". This can be fixed by reducing the minimum time step as follows:

1. Select menu **Simulator|Choose Analysis…**
2. Click the Advanced Options… button. Under Min. time step in Time step group, enter 100p

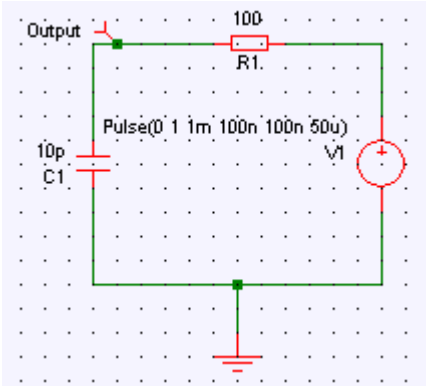

Figure 6-1. The Minimum Time Step Demonstration

6.1.4 Simulation Tolerances

Note: Do not confuse this with component tolerances. This is about simulation tolerances and is quite different. Simulation tolerances give you some measure of control over the simulation accuracy/speed trade off. If it is tighter the simulation will run more slowly but the result will be more accurate and vice-versa.

The full range of tolerance parameters are described in the Simulator Reference Manual Chapter 8. At this stage we will consider three of them:

- **Relative Tolerance** (RELTOL) so called because it is related to the calculated voltage/current.
- **Current Tolerance**. Absolute value set on the Options tab in Choose Analysis dialog but the default value is usually adequate.
- **Voltage Tolerance**. Similar comments to Current.

To edit Tolerances, go to **Choose Analysis|Options** tab.

The default value of RELTOL gives moderate accuracy which is satisfactory for most applications. Tighten RELTOL to improve accuracy but the simulation will run slower. Tighten the tolerance if you are simulating resonant or oscillatory circuits or if you are performing spectral analysis. In oscillatory and resonant circuits, errors can build up over time and this makes it more important to use tighter tolerances.

Important Note: Don't interpret RELTOL literally. A value of 0.1% (1m) does not mean that the results are 0.1% accurate.

The tolerance values are used primarily to control the accuracy of the non-linear iteration algorithms and the integration algorithms used to simulate reactive components such as capacitors.

The following is a simplified version of the algorithm used to control time step based on the tolerance parameters.

At each time point in the simulation two values are generated and compared. We will call them the **Estimated Error** (generated by the simulator) and the **Acceptable Error** (generated by the estimation algorithms).

Here is an example to explain what takes place. The scenario is that the simulator calculates that a particular node voltage is 1.136V and estimates the error on it to be 15mV. The Tolerance engine then takes the 1.136V, multiplies it by the RELTOL (default 1m or 0.001) and gets 1.136mV. To this it will add the voltage tolerance (1uV) to get 1.137mV.

The entire convergent process at each time point is:

1. The simulator carries out the calculation appropriate for that time point and arrives at the value 1.136V
2. The simulator runs its estimation algorithm to arrive at the **Estimated Error** of 15mV
3. Tolerance engine takes the result from step 1, multiplies it by RELTOL then adds the Voltage Tolerance to the product to obtain **Acceptable Error** (1.136V x 0.001) + 1mV = 1.137mV
4. The simulator compares the **Estimated Error** with the **Acceptable Error** and, in this example, finds it unacceptable (15mV > 1.137mV). If it had been acceptable the sequence would have continued from step 6.
5. The simulator estimates where the last step should have been to achieve the required accuracy, goes back to that point and repeats the calculation. When the result is accurate enough it proceeds to step 6.
6. The simulator assesses where the next time step needs to be (within the maximum limit) and repeats the whole sequence until the stop time is reached.

Note: A reasonably reliable way to determine an appropriate RELTOL value is run two simulations with different RELTOL values. For example run one with the default value then another with RELTOL reduced by a factor of ten. If the two simulations agree with each other within the accuracy you require, then continue with the coarser of the two tolerances. If not reduce the tolerance further and repeat.

6.1.5 Integration Method

Integration Method is the mathematical algorithm used to model reactive devices - inductors and capacitors. The method used to solve differential equations is **numerical**. All the Integration Methods have advantages and disadvantages and the choice of which to use depends on the circumstances.

The smaller the time interval, the smaller the error in approximating to the voltage.

Three Integration Methods are used:

- Backward Euler
- Trapezoidal rule
- Gear difference formula

The general rule is that if the circuit that has resonance that is important (for example, in an oscillator) then use Trapezoidal. Otherwise use Gear.

Trapezoidal and Gear do not work so well when there is an abrupt change in slope and it is in these kind of circumstances that the simulator will fall back, internally, to using Backward Euler. You don't have any option to select Backward Euler; the simulator selects it automatically when it needs to for an individual time point.

The only option you have regarding Integration Method are Trapezoidal or Gear. You can access them in **Choose Analysis|Transient tab|Advanced Options**.

6.1.5.1 Trapezoidal Rule

In SIMetrix Trapezoidal is the default method in common with many SPICE and SPICE-like simulators. However, in many situations Gear gives better results. Below we demonstrate the problems that each method has.

1. In the Schematic Editor, open **My Documents\SIMetrix\Training\Section-6\bridge.sxsch**. This a ready-to-run demo to illustrate the benefit of gear integration.

2. In the Schematic Editor, select menu **Simulator|Choose Analysis…** then Advanced Options button. In the Integration Method group note that the integration method is Trapezoidal.
3. Run the simulation and observe the ringing illustrated in Figure 6-2.
4. Change the integration method to Gear (see step 2 above).
5. Run the simulation again and observe that you now get a clean curve.

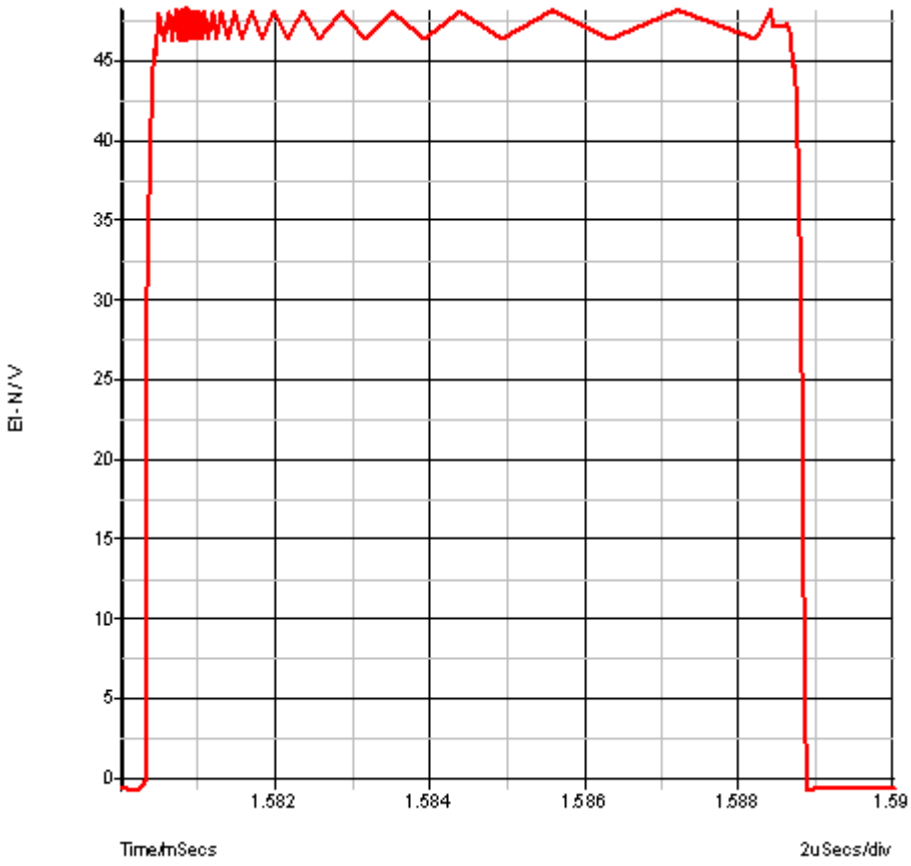

Figure 6-2. Trapezoidal Ringing

6.1.5.2 Gear Difference Formula

There are situations when Gear is not the best option because it introduces a kind of damping into the circuit which is not really there.

1. In the Schematic Editor, open **My Documents\SIMetrix\Training\Section-6\resonant.sxsch**.
2. Select menu **Simulator|Choose Analysis…** and click the Advanced Options button.
3. In Integration method select Gear. Click Close
4. Run the simulation and observe that its amplitude is decaying.
5. Run the simulation again using Trapezoidal integration.
6. The two curves will look like those in Figure 6-3.

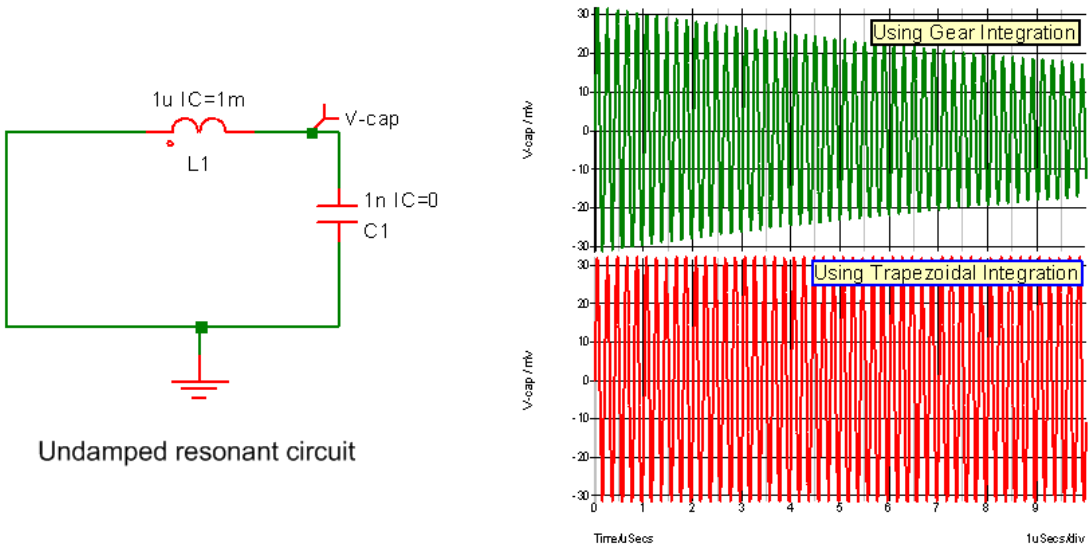

Figure 6-3. When Not to Use Gear Integration

The green waveform is decaying and it should decay only if there is resistance in the circuit (which there isn't). When you change the integration method to Trapezoidal the amplitude does not decay. The decay with Gear would be less if the simulation tolerances were tightened.

6.1.6 Sensitivity Analysis

There is no GUI user interface to this; it does not appear in the Choose Analysis dialog.

1. Open **My Documents\SIMetrix\Training\Section-6\sens.sxsch** in the Schematic Editor.
2. (Optionally) on the Schematic Editor, press F11 and this splits the Schematic Editor window horizontally with the schematic in the upper pane. Notice the **.SENS** line.
3. Run the simulation in the normal way. The output is a text file named **design.out** in the same folder as the one from which **sens.sxsch** was opened.
4. To open the file, in the Command Shell go to **Graphs and Data|View List File** or **Edit List File** as required or just open it from the folder with Notepad or similar. (The report is designed to be viewed in a mono-spaced font.)
5. Scroll down to the heading 'Sensitivity Analysis' and then further to the parameter name of R3. Below is an extract from two lines of the data.

Extract from Sensitivity Analysis data

Parameter name	Param value	Param sensitivity	Param normalised sensitivity
R3	1k	1.2319971625258m	1.23199716252577
R3 temp	27	0	0

The important number is in the last column. In this example, if the value of R3 were to be changed by 1%, the output would change by 1.23199716252577%.

R3 temp is zero because this is the parameter on R3 which allows you to set its temperature. It does not have any temperature coefficients and so a change in temperature will not affect the output.

6.1.6.1 Data Output Options

Data output options provide some control over disc space used to store simulation results. With large circuits and long runs the simulator is capable of generating vast quantities of data that can fill a hard disc. This topic is not dealt with by this training but there is article about it on the SIMetrix Technologies website at http://www.simetrix.co.uk/site/support/support.htm . Click on Knowledge Base

at the left of the screen. There are two articles on Handling Simulation Data. The second article listed concerns reading saved data back in.

6.2 AC Analysis

AC analysis is a *small-signal* analysis. In small-signal analyses, the signal is treated as if it were insignificantly small. You may specify a 1 Volt signal at the input (most do, it's a convenient zero dB), but the analysis will process it without disturbing the bias levels. If you are studying a 60dB amplifier, for example, the output will show 1000V and it won't even blush!

Small-signal analyses operate on a linearised circuit. A trivial example of a linearised circuit is shown in Figure 6-4 and Figure 6-5. The circuit can also be found at

My Documents\SIMetrix\Training\Section-6\ diode-linearisation.sxsch

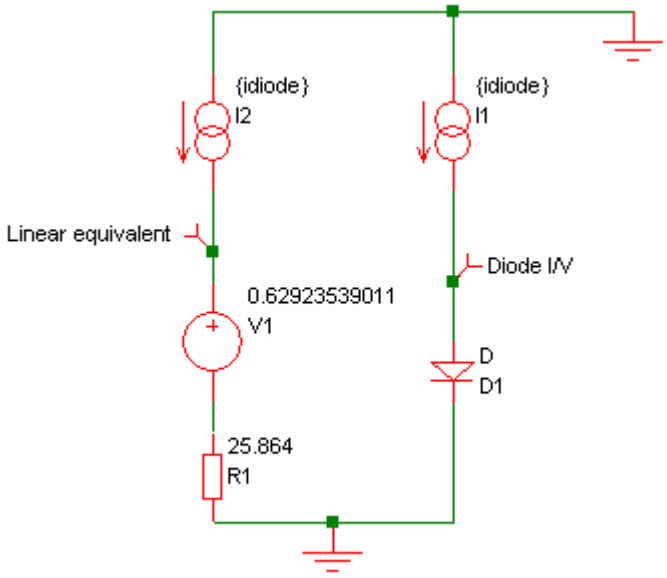

Figure 6-4. Circuit to Demonstrate Linearisation of a Diode

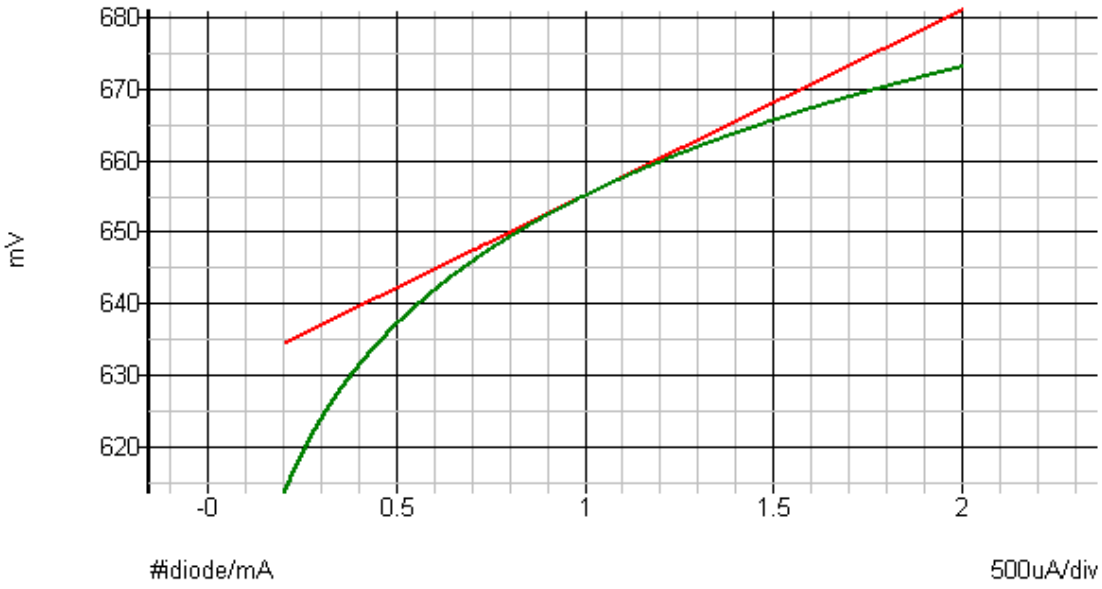

Figure 6-5. Linearisation of a Diode curve

The graph shows the linearised version (V1 and R1, red curve) and the non-linear large signal version (D1, green curve). Over the small signal range of 0.9mA to 1.1mA, the linearised version is a reasonable approximation to the non-linear version. This is the principle used in small-signal analyses.

Small-signal analysis cannot be used in all situations. It is not valid if the operating point of the circuit has to change in normal operation. In a sample-hold circuit for example, the sampling switch must change from fully on to fully off to behave correctly. The operating point in these two states is profoundly different while the small-signal approximation can only be valid in one of those states.

6.2.1 AC Analysis Modes

The most common AC analysis mode is AC sweep. This performs a frequency sweep with a single input. AC sweep allows the analysis of open-loop and closed loop frequency response.

In addition to frequency sweep, the following are also available:

- A device value
- A model parameter
- Temperature
- A parameter used in an expression

6.2.2 Typical Application of AC Sweep (768Hz Bandpass Filter)

1. In the Schematic Editor, open **My Documents\SIMetrix\Training\Section-6\768Hz bandpass.sxsch** .
2. Select menu **Place|Voltage Sources|AC Source** and place an AC Voltage source between the left end of R1 and ground.
3. Open Choose Analysis Dialog and check AC box and Click AC Tab
4. Set Start frequency to 100 and Stop frequency to 10k. Click OK.
5. Run the simulation and a curve such as Figure 6-6 appears.

Figure 6-6. Curve from 768Hz Bandpass Filter

6.3 DC Sweep

6.3.1 Description

DC Sweep performs a repeated dc operating point simulation while sweeping some device or parameter. Because this is a DC simulation, reactive components are ignored. So, capacitors (with no initial conditions) are treated as open-circuits and inductors (with no initial conditions) are treated as

short-circuits. Capacitors and inductors that have initial conditions specified are treated as voltage and current sources respectively, with a magnitude equal to the initial condition value.

Time varying sources are given fixed DC values in a DC sweep. If an explicit DC value is supplied then that value will be used. If not the t=0 value will be used.

6.3.2 Practical Illustration

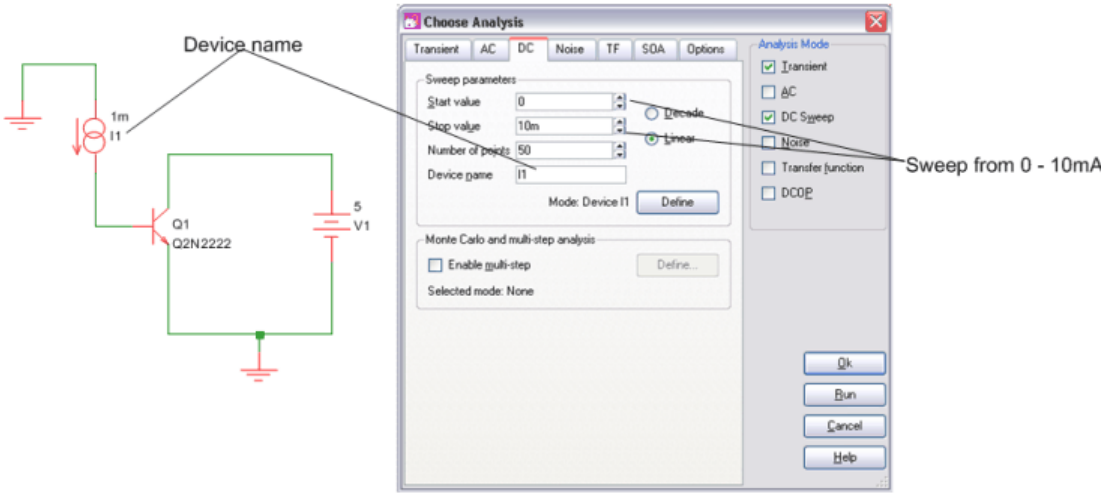

Figure 6-7. Setting up a DC device sweep

1. Open the file **My Documents\SIMetrix\Training\Section-6\dc.sxsch** in the Schematic Editor. There are no probes on the circuit and so there will be no curve produced.
2. Select **Probe|Place Fixed Current Probe** (shortcut U) and place it on the collector of Q1.
3. To set up go to **Simulator|Choose Analysis…**. The illustration above shows the circuit, the dialog and the values which you should enter. The presence of 1mA indication beside the current source has no meaning in this analysis; this, after all, is what we are sweeping. The 1mA is the default value it would have in anything other than a DC sweep. You can do a decade sweep or a linear sweep. In this case it will be a linear sweep.
4. Check the DC Sweep box and the DC tab.
5. Enter the start and stop values, the number of points and the device name, I1 in this instance.
6. Run the simulation (key F9) and Figure 6-8 shows the result.

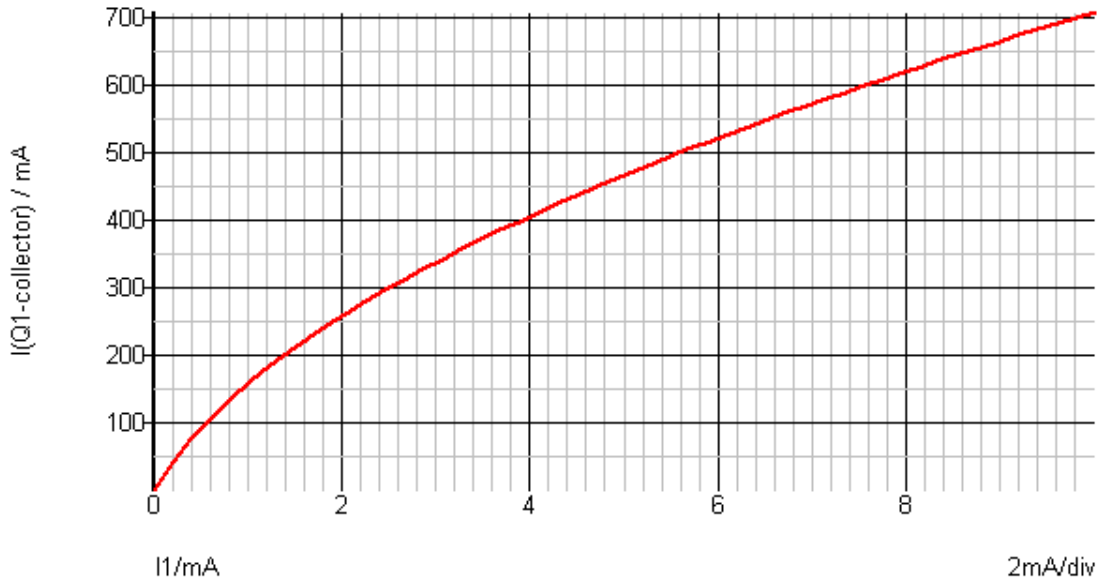

Figure 6-8. The DC sweep curve

6.4 Noise Analysis

This is another small-signal analysis mode as explained above for AC analysis.

All electronic components generate noise and many types of noise are inherent and can be predicted using fundamental laws.

Although all real devices generate at least some noise, energy storage components such as capacitors and inductors are usually considered noise less.

6.4.1 Types of Noise

6.4.1.1 Thermal Noise (or Johnson Noise)

This arises as a consequence of the thermal agitation in the structure of the resistor.

The equation for thermal RMS noise (Vn) in a resistor is $Vn = \sqrt{4KBTR}$ where K is Boltzmann's constant which is approximately 1.38e-23 , B is the bandwidth over which the noise is being measured, T is temperature (Kelvin) and R is the resistance.

6.4.1.2 Shot Noise

This noise is found in semiconductor junctions and is related to the current flow. This equation is $Vn = \sqrt{2qBi}$ where q is the charge on an electron which is approximately 1.6e-19 , B is the bandwidth over which the noise is being calculated and i is the current.

6.4.1.3 Flicker Noise (or 1/f Noise)

Flicker noise, also known as 1/f noise, is less well understood than thermal and shot noise but is found almost universally in nature. As the 1/f nomenclature implies, it has an inverse frequency relationship; that is the noise power reduces with increasing frequency.

6.4.2 Noise Measurement

The noise at the output is a combination of all types of noise added together. SIMetrix has, however, a facility to program each individual device to measure its contribution to the total noise and what proportions of it is Shot Noise and Thermal Noise. It is not possible to find out the components of each type of noise on the output, only at component level.

Plotting of noise in SIMetrix requires a different technique from most other plotting. You need to designate a single output Terminal, run the simulation and only then select from the menu which graph or individual component you want to plot. You cannot cross-probe noise. On the Probe AC/Noise menu three of the entries are:

- Plot Output Noise
- Plot Input Noise
- Probe Device Noise

6.5 Practical Illustration of Noise Measurement

1. Open the file **My Documents\SIMetrix\Training\Section-6\jfet_amp.sxsch** in the Schematic Editor.
2. You will see that the circuit has no output terminal. Add a Terminal Symbol to the output of X1 using **Place|Connectors|Terminal** (or Y). Accept the default name of VOUT.
3. Open the Choose Analysis dialog, check the Noise box and open Noise tab.
4. Set start and Stop frequencies to 100 and 100k respectively.
5. In the Noise parameters group enter 'VOUT' as the Output node and 'V3' as the Source Name. (In this case is V3 is the name of a device, not the name of a node. The output is the name of a node.) Leave the **Ref** node empty.
6. Click Run. A dialog appears on screen for a few seconds but no graph is plotted.
7. On the Schematic Editor, go to **Probe AC/Noise|Plot Output Noise** and the output noise curve appears.

8. On the Schematic Editor, go to **Probe AC/Noise|Plot Input Noise** and the input noise curve is added to the graph. Then click **Probe AC/Noise|Probe Device Noise** and the cursor changes to a pen. Click on Q4. Repeat the process (you have to go to 'Probe Device Noise' again) for Q3. Your result should look like Figure 6-9.

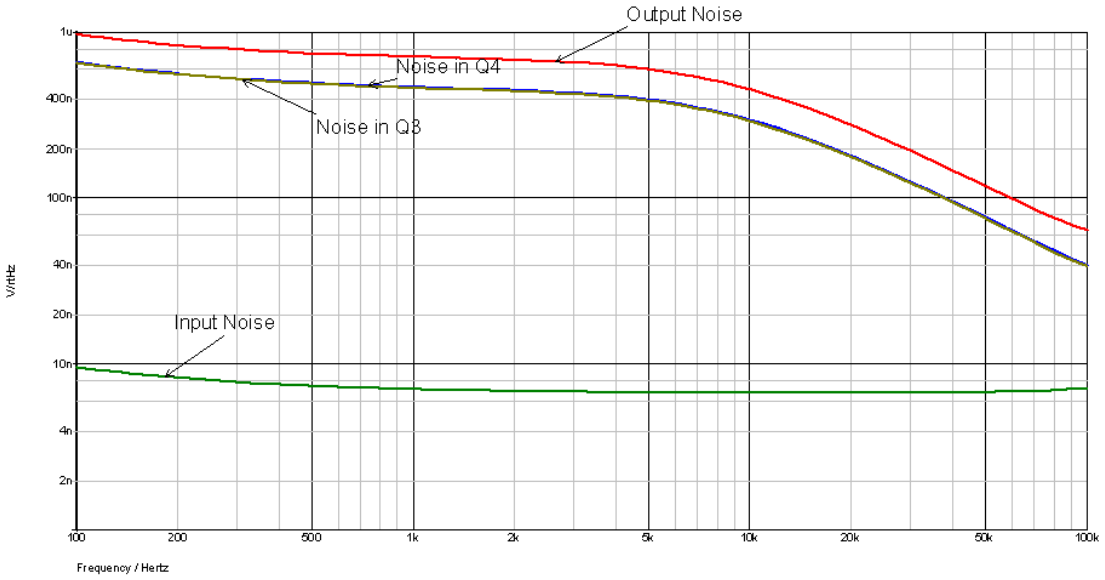

Figure 6-9. Noise curves

6.6 Interpreting the Illustration of Noise Measurement

6.6.1 Noise Units

The units on the y-axis are those in which noise is always measured, namely "Volts per root Hertz" (V/rtHz). Looking back to the equation we can see that the noise voltage is proportional to the square root of the bandwidth. This is saying that at 1k the noise is 750nV for every root Hertz bandwidth. Let's suppose that this is a straight line. Because noise is always dependent on the bandwidth over which you measure it, we are saying that at 1k the noise is 750nV/rtHz. So if we consider a narrow range from 400Hz to 2kHz, that is 1.6kHz bandwidth, we could take that 750nV, multiply by the square root of 1,600 and that will give the actual RMS noise we will see in volts. To get the total noise over a wider bandwidth with a curve that isn't flat we need to integrate the equation for the curve between specified limits.

This feature is available using graph measurements. Here is an example.

1. Select the 'Output Noise' curve by selecting the appropriate check box on the Legend Panel.
2. Select menu **Measure|More Functions**.
3. Go to **Measure|Other|Total Noise**. (In version 5.5 go to **Total Noise** in the list on the right hand side.)
4. Notice the measurement appear below the label on the Legend Panel. (You may need to resize the panel using the splitter bar.)

6.6.2 Output Noise

This is represented by the red curve. This is the total noise at the node specified as the output in the Choose Analysis dialog box. In the above example this is the VOUT node where the terminal was connected

6.6.3 Input Noise

This is the 'Input Referred Noise'.

Suppose you have an amplifier with a gain of 10, a signal input and signal plus noise at the output. The question is how much noise would you have to put on the input of an equivalent but noise-less

amplifier to match the amount of noise that is being generated by amplifier alone. In this case, because the amplifier has a gain of 10, the input referred noise will be the noise at the output divided by 10. This amplifier will not have a gain of 10 at all frequencies; it will roll off. The noise part way down the dip referred to the input will be bigger because the gain is less.

The green curve on Figure 6-9 is the Input Referred Noise.

6.6.4 Device Noise

Devices Q3 and Q4 (the jfets) were probed and the green and yellow curves (almost one on top of the other) represent the noise which each is contributing to the total.

6.7 Transfer Function

Transfer Function (TF) is really a variation on AC Sweep but does things in a different way that is sometimes useful.

TF can do in a single run something that may take several AC runs. The essential difference between them is:

- AC Analysis calculates the response at many nodes of circuit to a stimulus at just one input.
- TF is the reverse of AC analysis. It simultaneously calculates the response from many inputs to a single output. TF also calculates input and output impedance.
- Everything that TF can do, AC is also able to do but may take more than one AC run to achieve it.

1. On the Schematic Editor, open **My Documents\SIMetrix\Training\Section-6\TF_Freq.sxsch**.
2. Run the simulation. No graph is plotted.
3. One the Schematic Editor, go to **Probe|Add Curve**. This opens the Define Curve dialog. Select, from the Available Vectors dropdown, **V1#Vgain, V2#Vgain, V3#Vgain and V4#Vgain**, separating them with semi colons. The 'Y Expression' box should look as shown in Figure 6-10.

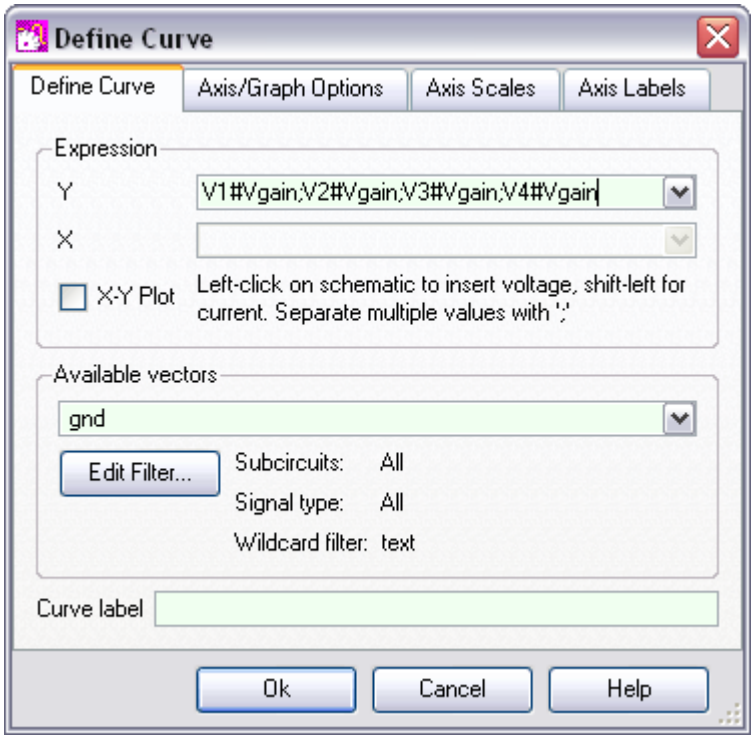

Figure 6-10. The Define Curve Dialog

4. On the Define Curve Dialog, click OK and the curves such as Figure 6-11 appear. A Legend Box has been added to give the curve meaningful names.

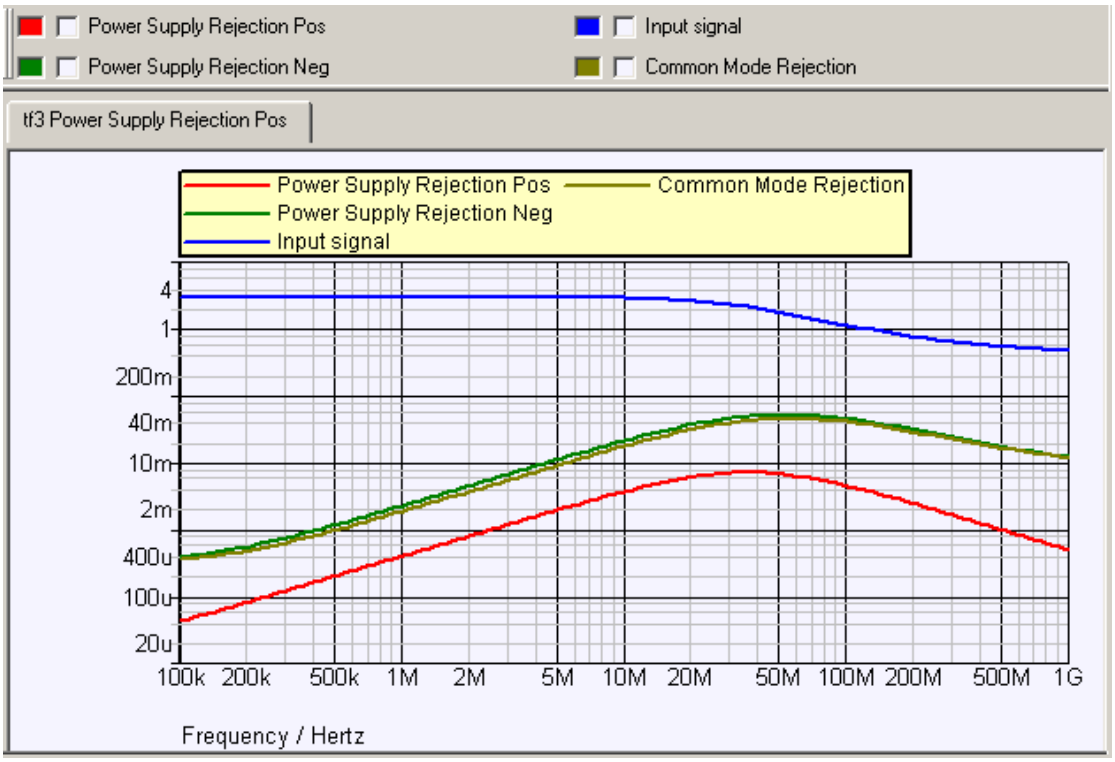

Figure 6-11. One Selection of TF Function Curves

6.8 Further Detail

6.8.1 How SIMetrix assigns net names

All connections on a schematic must have a name and this is known as the 'net name'. SIMetrix will choose a name based on something that is connected to it but you cannot specify which it chooses. What you can do is to add a Terminal Symbol which will force your choice of net name. It is convenient to use this method to define an output for noise or transfer function analysis.(Some other Schematic Editors refer to attached labels. SIMetrix does not have a concept of labels; everything is done with symbols.) The Terminal Symbol is a special symbol with the default name VOUT.

7 Waveform Viewer and Data Analysis

We have already seen the Waveform Viewer in the practical illustrations we have worked on so far. This section says more about random probing, multiple y-axes, multiple graphs and annotation. Collectively we call this Data Analysis.

7.1 Probe Types

SIMetrix maintains two different types of probe:

- **Fixed probe**. This is placed on the schematic before the simulation run and generates its curve either during or after completion of the run.
- **Random probe**. Random probe does not have a lifetime. It's just click and plot. Click on the schematic after the run is completed and the appropriate curve is produced, but not subsequently updated.

7.1.1 Fixed Probe Types

There are several types of fixed probe, all of which plot the specified signal until the analysis is complete.

- **Voltage**. Self explanatory.
- **Current**. This is a single pin device that must be put on to the pin of the component.
- **Inline current**. This probe has two pins and you wire it in to the circuit.
- **Differential Voltage**. This probe measures the voltage between two points.
- **Phase**. Used only in AC Analysis.
- **dB**. Used only in AC Analysis.
- **Bode Plot**. This probe plots the phase and gain of a loop.
- **Bus**. Covered in Hierarchical Schematics, Busses and Digitalon page 105.

7.1.2 Extended Practical Illustration of Fixed Voltage Probes and Data Analysis

7.1.2.1 Multiple Probes and Multiple Curves

This illustration introduces a number of features of Fixed Probes and Data Analysis that we have not seen before in this training.

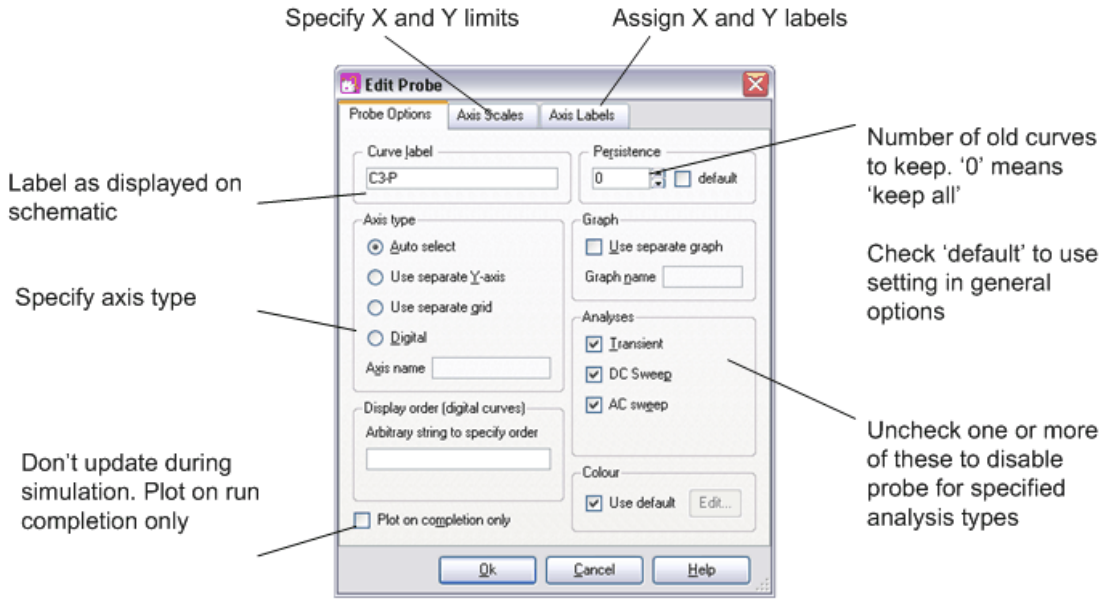

Figure 7-1. The Edit Probe Dialog

1. Open **My Documents\SIMetrix\Training\Section-7\hybrid-psu.sxsch** in the Schematic Editor.
2. Using menu **Probe|Place Fixed Voltage Probe…,** place a fixed voltage probe on the output (the node between L1 and C3).
3. Open the Edit Probe dialog (select the probe and F7 or double click it). When you open it there is a default label name related to the position of the probe on the schematic. This is the name that appears above the graph and you can edit this. The dialog is shown as Figure 7-1.
4. Change the Curve Label to VOUT and click OK.
5. Run the simulation and the (by now, familiar) curve appears. Notice also that in the 'Legend Panel' the only curve shown (VOUT) is in red.
6. Change the value of R9 to 100k.
7. Run the simulation again and notice the new curve is green and that both red and green curves are shown in the Legend Panel.

7.1.2.2 Persistence

1. Open the Edit Probe dialog by selecting the probe and pressing F7.
2. Set Persistence to 1 and click OK.
3. Run the simulation and the new blue curve is the only one shown. The setting of Persistence determines how many of the most recent curves are displayed.

Note: Don't change Persistence settings and run the simulation again while curves are being displayed. Close the curves or you can get into a situation where curves are proliferating and overlaying each other.

7.1.2.3 Auto Naming

1. Close the Waveform Viewer and run the simulation again to obtain a single red curve from VOUT.
2. Add a new voltage probe on any of the wires or nodes surrounding the emitter of Q2. Notice that the auto-naming feature has supplied the name Q2-emitter.
   Note: Auto naming is context sensitive only if you place the probe on a wire. If you place it in free space, it still assigns a name but not one which relates to its position in the circuit.
3. Run the simulation again and notice that red and green curves are now sharing the y-axis.

7.1.2.4 Separate Grids

1. Edit the new Q2-emitter probe and check 'Use separate grid' in the Axis type group. Click OK.
2. Run the simulation again and notice that the new curve is on a new grid which has a suitably expanded y-scale. Your graphs should like Figure 7-2.

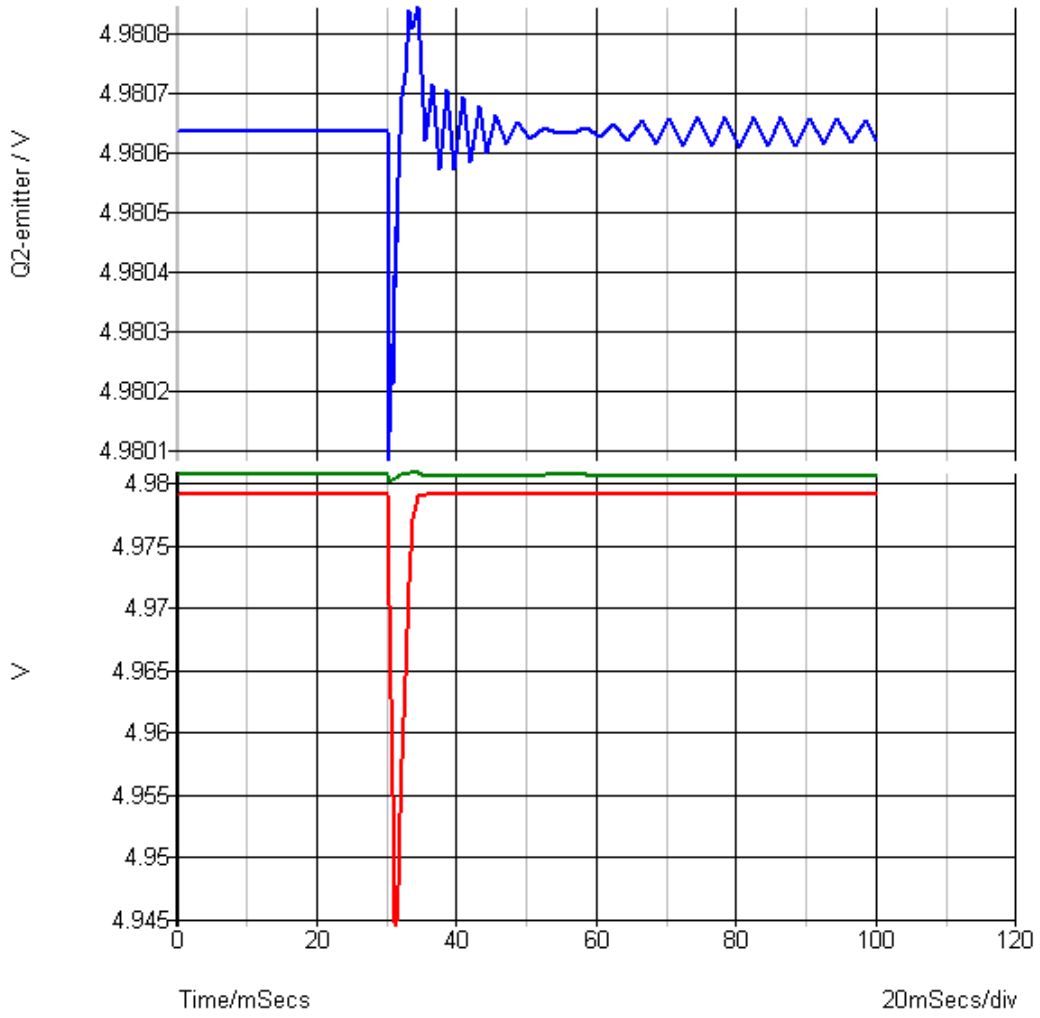

Figure 7-2. Two Grids

3. Now add a third voltage probe on any of the wires or nodes surrounding the Q2 collector. (When you have done so, you will notice that SIMetrix obviously thought that Q1 was more interesting than Q2 so it based the name on that but you can change it).
4. Run the simulation again and notice that the blue curve and the green curve are on the same grid whereas the red curve is on the main grid as shown in Figure 7-3. Keep the curves open.

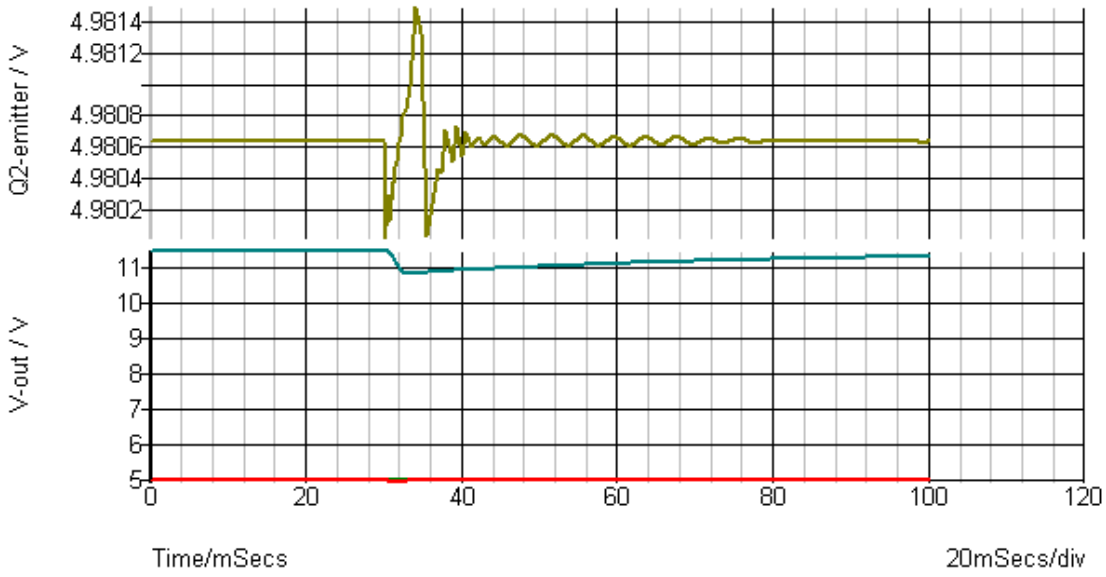

Figure 7-3. Two Grids and Three Curves

5. Open the Edit Probe Dialog for Q1-collector and:
   - Check Use separate grid
   - Set Persistence to 1
   - In the Axis type group enter **grid3** into the Axis name field. (It can be anything but for this training call it grid3.) Click OK.
   - Edit Q2 emitter probe and set Persistence to 1.
   - Close Waveform Viewer.
6. Run the simulation again and notice that there are three separate grids. You can specify grids for any probe as long as they all have a unique axis name. Your curves should look similar to Figure 7-4.
7. We now have three graphs, one for each probe. There is one on the main grid, one on the 'Use separate grid' but which has blank axis name, and the third which has 'Use separate grid' and a name of grid3. Axis names must be unique.
8. The way to control where every probe goes is to give its axis a name in the Edit Probe dialog. This is only an arbitrary identifier; all that it does is to give a name to each axis.

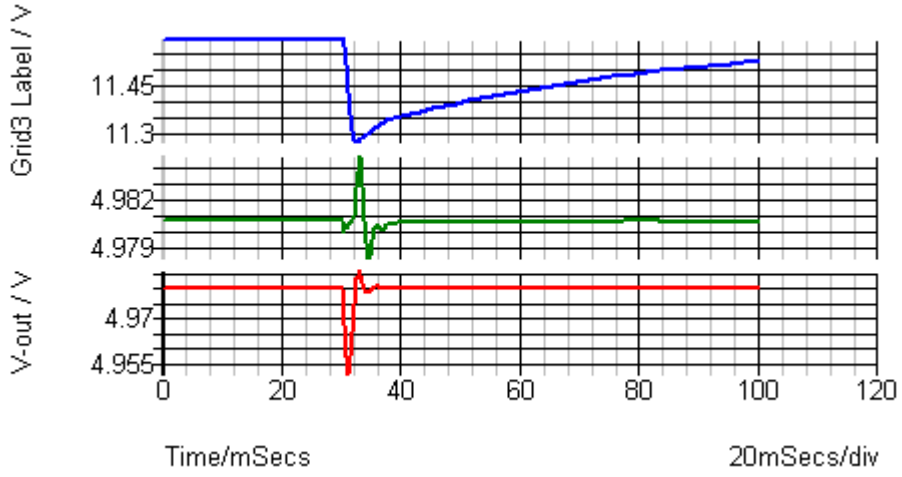

Figure 7-4. Three Grids

7.1.2.5 Separate Y-axes

1. On the two probes that have 'Use separate grids', change the Axis type group selection to 'Use separate Y-axis'.

2. Close the waveform viewer and run the simulation again and a curve like Figure 7-5 should appear. Multiple y-axes work the same way as multiple grids.

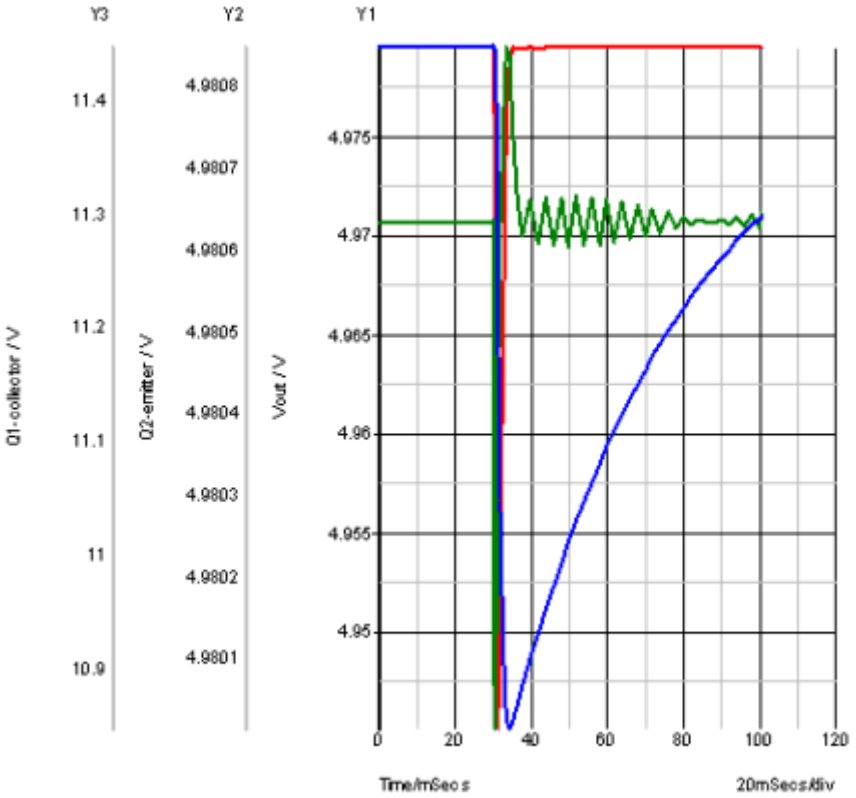

Figure 7-5. Multiple Y-axes

7.1.2.6 Digital

Digital axes are simply axes with a small vertical dimension and would typically be used for digital signals. But you can plot analog signals on digital axes as well and that is what we will do here.

1. On the two probes that have 'Use separate Y-axis', change the Axis type group selection to 'Digital'.
2. Close the Waveform Viewer and run the simulation again. A curve like Figure 7-6 should appear. Digital axes are designed for digital waveforms but can be used for analogue. Each probe goes on its own axis so you can always see each waveform separately. They are physically small.

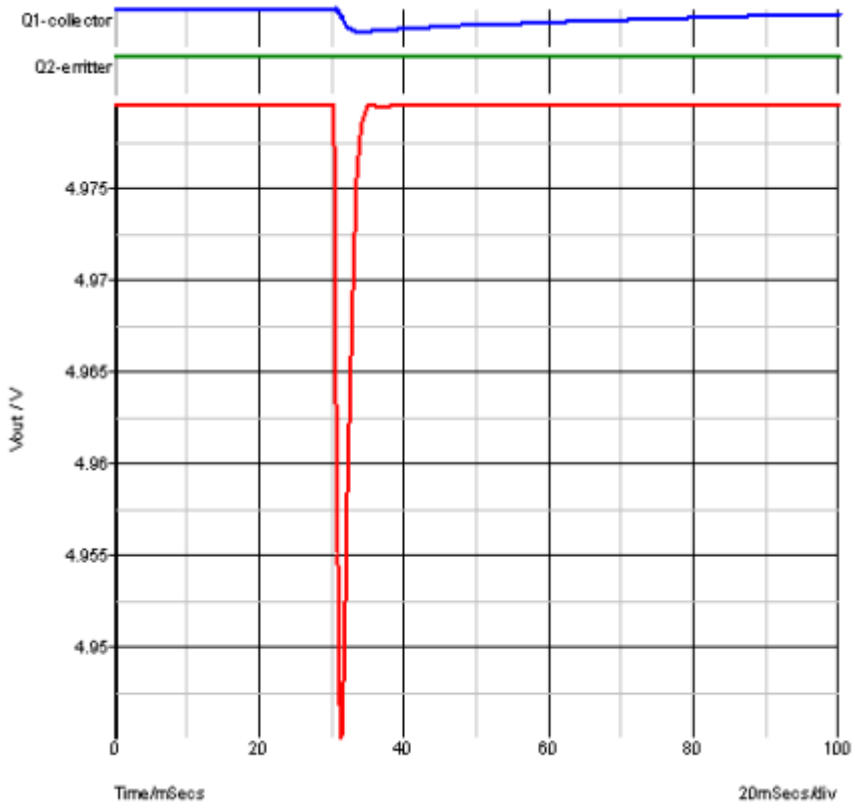

Figure 7-6. Digital Curves

7.1.3 Fixed Current Probes

Fixed current probes must be placed on a component pin. They will not work if they are placed in the middle of a wire. It is recommended, when wiring, that a length of 'wire' (albeit a short one) is introduced between devices rather than connecting the pin of one to the pin of another. This is so that, when probing, the probe knows the identity of the device to which it is connected.

Figure 7-7 shows the placement of a fixed current probe correctly and incorrectly.

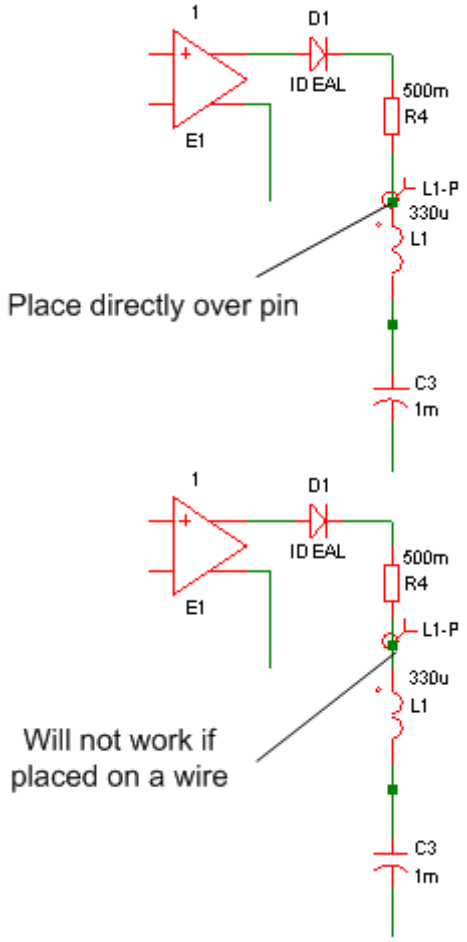

Figure 7-7. Fixed Current Probes - Right and Wrong Placement

1. Delete the probes on Q2 emitter and collector from the previous example.
2. Using menu **Probe|Place Fixed Current Probe…** add a current probe to the top end of L1 top pin.
3. Close the Waveform Viewer and run the simulation. A curve like Figure 7-8 appears. The important thing to note here is that SIMetrix has automatically provided a separate axis for the two probes because it has recognised that one is in mA and the other is in volts. It automatically splits them because it is unlikely that they would want to share an axis. It also numbers each Y-axis.

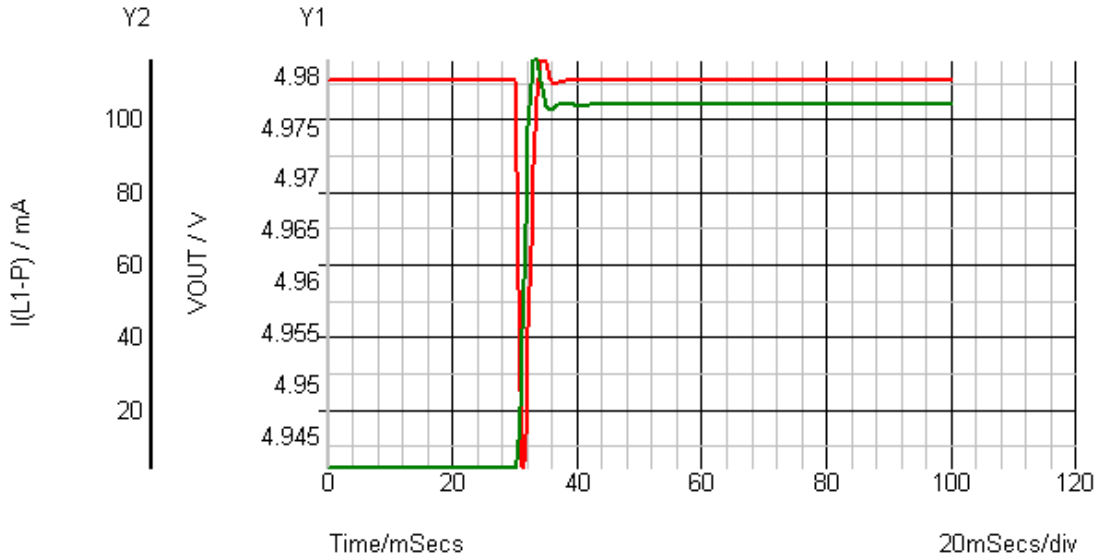

Figure 7-8. Voltage and Current curves

7.1.4 Other Fixed Probes

The fixed voltage and current probes on which this training has been concentrating cover the majority of probe usage. The other probes listed under Fixed Probe Types on page 66 (and others) are described fully in the manuals and specific training does not cover them.

7.1.5 Random Probes

Random probing is designed to be used after a run is complete. A summary of random probing would be:

- **Voltage...** This, single ended voltage, draws a curve.
- **Voltage (New graph sheet)...** This is exactly the same as Voltage but always creates a new tab sheet in the window.
- **Voltage - Differential...** Click on each point and the curve represents the first click minus the second click.
- **Voltage - Digital...** This is the same as voltage but curves it on a digital axis, just a different style of axis. It would not normally be used for analogue signals.
- **Voltage - AC Coupled...** This is the equivalent of switching on the AC Coupled button on an oscilloscope. It takes the DC component out of the signal.
- **Voltage - Bus**. Covered in this training.
- **Current in Device Pin...** Click within half a grid width of the end of the pin.
- **Current in Wire...** measures the current in the wire and identifies the component in a rather cryptic way. It gives the display name of the component followed by -n or -p. The convention in SIMetrix is that -p (positive) means the upper end of a vertical component and -n is the lower. (If the wiring contains a loop, this facility does not work.)
- **Current in Device Pin (New graph sheet)...** Equivalent of voltage but creates a new tab.
- **More Probe Functions...** The functionality that can be accessed here is little used and will not be described in this training.
- **Power In Device...** Click on the device rather than a node or a pin and it will plot the power being dissipated by the device.

7.1.5.1 A First Attempt

This practical illustrates random probing voltage, current in device pin, power in device, new grid and voltage differential.

1. Open **hybrid-psu.sxsch** and ensure that all the fixed probes are removed.

2. Close the Waveform Viewer and run the simulation. The Waveform Viewer is not displayed.
3. In the Schematic Editor use the right-click context (popup) menu to select **Probe Voltage**, or select **Probe|Voltage** from the main menu. Then click on the output (C3-P). A red curve appears on a newly created grid along with its identifier in the Legend Panel.
4. In the Schematic Editor, use the context (popup) menu **Probe Current** and click on the top pin of L1. The following things change:
   - A green curve appears on the same grid.
   - The y-axis voltage on C3-P is now labelled Y1.
   - A new y-axis labelled Y2 appears scaled in mA.
   - A new identifier appears in Legend Panel linking the green curve to Y2
5. In the Schematic Editor, go to **Probe|Power in Device** and click on Q2. The following things change:
   - A blue curve appears.
   - A new y-axis labelled Y3 appears.
   - A new identifier appears in Legend Panel linking the blue curve to Y3.
6. The graph is a bit cluttered so click on the New Grid icon on the Waveform Viewer toolbar. In this instance we are creating a grid before we put a curve on to it.
   General Note: You can add grids upwards and add y-axes to the original (bottom) one. What you cannot do is to add another y-axis to any grid other than the original.
7. On the Schematic Editor, go to **Probe|Voltage - Differential**. Now it is expecting you to click twice and it will subtract the voltage at the second click from the voltage at the first. Click on the top of L1 and then on the bottom. It has now adapted the y-axis to suit, labelled the curve with precisely what it has done and added an identifier in the Legend Panel. The result should look like Figure 7-9.

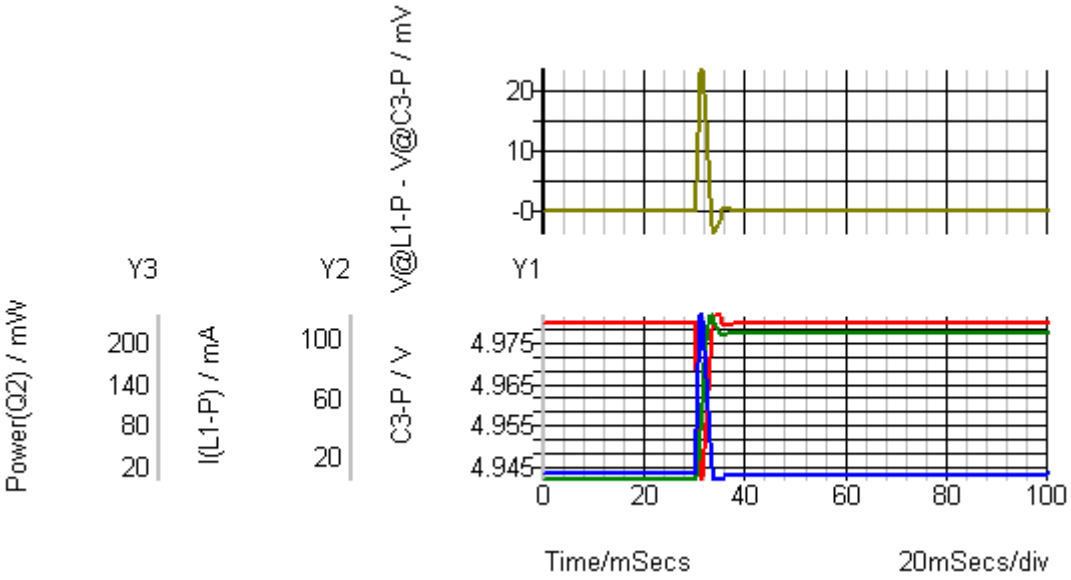

Figure 7-9. The Result of the First Attempt

8. To round off the demo, go to **Curves|Stack all Curves**. You can also move selected curves (selection is by checking boxes on the Legend Panel).
9. Keep the waveform viewer open for the next section.

7.1.5.2 Managing Curves

To select a curve, tick its box in the Legend Panel.

In SIMetrix there is a difference between highlighting a curve and selecting it. Highlighting also applies to the Schematic Editor. You can highlight one or more curves by selecting them and going to **Curves|Highlight Selected Curves** (or pressing H). They are all then coloured to a bold, bright pink. Press U to unhighlight selected curves.

There are different ways of moving curves. Moving a curve from one grid to another on the same sheet can be carried out using the built-in facility. Moving to another sheet requires the use of a (non-Windows) clipboard.

7.1.5.2.1 Moving Curves Using the Built-in Facility

1. The following assumes that you kept the waveform viewer open from the previous section.
2. On the Waveform Viewer, check the C3-P box 

on the Legend Panel.
3. Click around the grids, each time to the left of the y-axis and see that when selected the axis line goes heavier. Select any grid except the one with the C3-P curve.
4. Click the Move Curve icon  on the toolbar. Note that the curve has moved to the selected grid. General: It will move any selected curve to the selected axis.
5. Uncheck C3-P on the Legend Panel.

7.1.5.2.2 Moving Curves to a New Sheet

This is a cut and paste action. The cut, copy and paste functions do not use the Windows clipboard. This is mainly because the amounts of data can be very large. It uses a clipboard which is not only specific to SIMetrix, it is specific to this instance of SIMetrix. You are, therefore, not able to copy a curve from one session of SIMetrix to another.

1. Select one or more curves on the sheet and copy them as though you were using the Windows clipboard. (e.g. using menu **Edit|Copy**)
2. Go to **File|New Sheet** (press F10) on the Waveform Viewer and the new sheet appears as a new tab.
3. Paste. (e.g. using menu **Edit|Paste**)

7.1.5.2.3 Deleting a curve

1. Select a curve.
2. Click the delete icon  on the Waveform Viewer.

Note: There is no undo facility.

7.1.5.2.4 Identifying Curves

If you are trying to identify one or more curves out of many on the screen, some of them the same colour, you can do so by first selecting it on the Legend Panel and pressing H. This facility is only for identifying curves; you need to select before moving or deleting.

7.2 Using the Cursors

7.2.1 Preparing the Measurement

This facility will move a cursor to an interesting point on a graph automatically.

1. On the Schematic Editor ensure R9 is set to its original value of 47K.
2. Close the Waveform Viewer and run the simulation.
3. Do a voltage probe on the output.
4. Zoom in. Click on the Fit to Width magnifying glass  and rubber band select the part of the curve showing activity.
5. On the Waveform Viewer, go to **Cursors|Toggle On|Off** .
6. Place the reference cursor (labelled REF) by dragging it to some point before the load response.
7. Place the Main cursor (short dashes) somewhere on the falling edge.
8. On the Waveform Viewer, go to **Cursors|Move|Main Cursor to Next Trough** (or press F6). The main cursor behaves as instructed.

7.3 Adding Labels

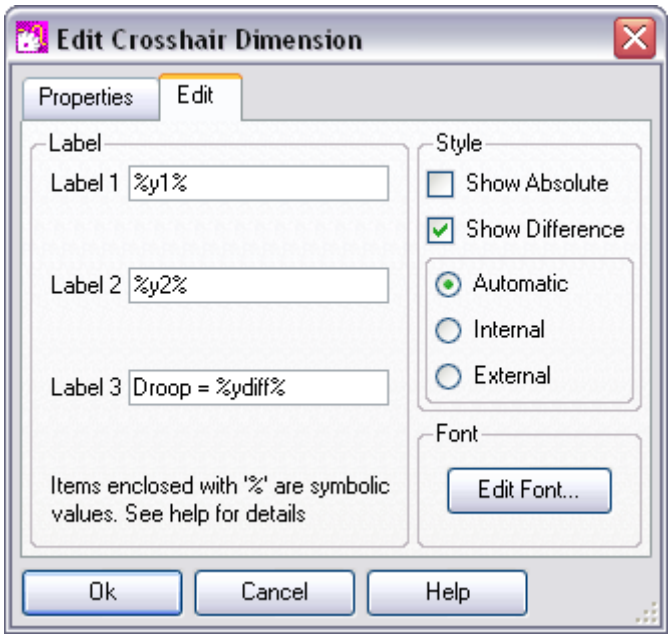

Figure 7-10. Edit Crosshair Dimension dialog - Edit Tab

9. Double click on the measurement of vertical droop on the right hand side. This opens the Edit Crosshair Dimension dialog shown as Figure 7-10.
10. As you drag the main cursor, the Droop value updates accordingly (the difference between the voltage values at the reference and main cursors). Prefix the default entry in Label 3 with 'Droop = ' so that the complete entry looks like:
    Droop = %ydiff% . (If you want to use a percent sign in the free text type a double percent.)
    Internal means that the arrows will always be displayed between the cursors. External means they will always be displayed outside the cursors.
11. Uncheck Show Absolute. Click OK and the result will look something like Figure 7-11.

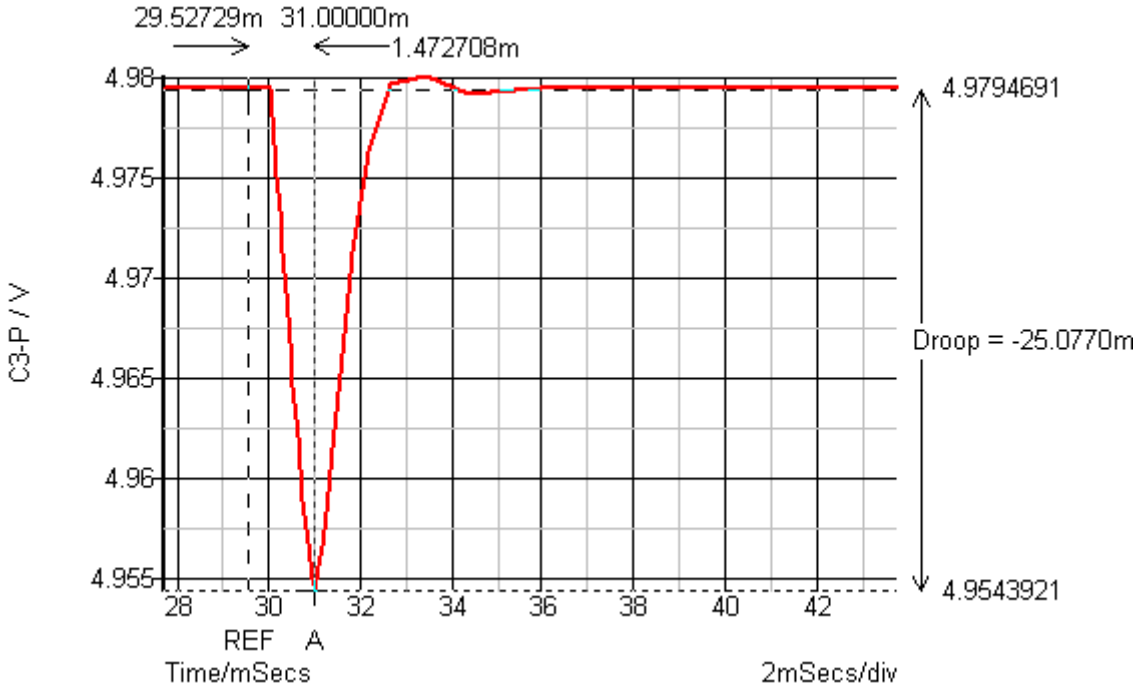

Figure 7-11. Vertical Droop

You will see that the absolute values are no longer present and the free text (“Droop =”) precedes the value. In the Style group of the dialog, Automatic is selected by default. This refers to the orientation of the arrows (internal or external). By convention, internal arrows point outwards and external arrows point inwards.

There are some notes about this facility as follows:

- A feature of this automatic cursor movement facility is that the search that it carries out is very sensitive. For example, if it finds a trough of a few picovolts, it will go to that.
- You can put expressions in the Label fields on the dialog. They must be enclosed within curly braces. For example {1/%xdiff%} will evaluate a frequency from a period measurement. To display a curly brace type two (as with %).
- There is an Edit tab and a Properties tab on the Edit Crosshair Dimension dialog. Most of the properties are internal and you are unlikely to need to know about them.

There is more information about labels in Appendix E. as well as full documentation in the Script Reference Manual, Chapter 7, Graph Objects.

7.4 Annotating the Graph

Annotating a graph may be useful when it is to be used in a report. There are five types of annotation:

- **Curve Marke**r. This is text which locks to the curve.
- **Legend Box**. This lists all the curves that are on the sheet irrespective of whether they are on multiple grids. It displays a sample of each curve in its colour.
- **Free Text**. Plain text, no background, can be placed anywhere.
- **Text Box**. Similar to Free Text but is in a box with a background colour.
- **Caption**. Similar to Free Text but is in a large font designed for a heading.

An example of each type is shown on Figure 7-12.

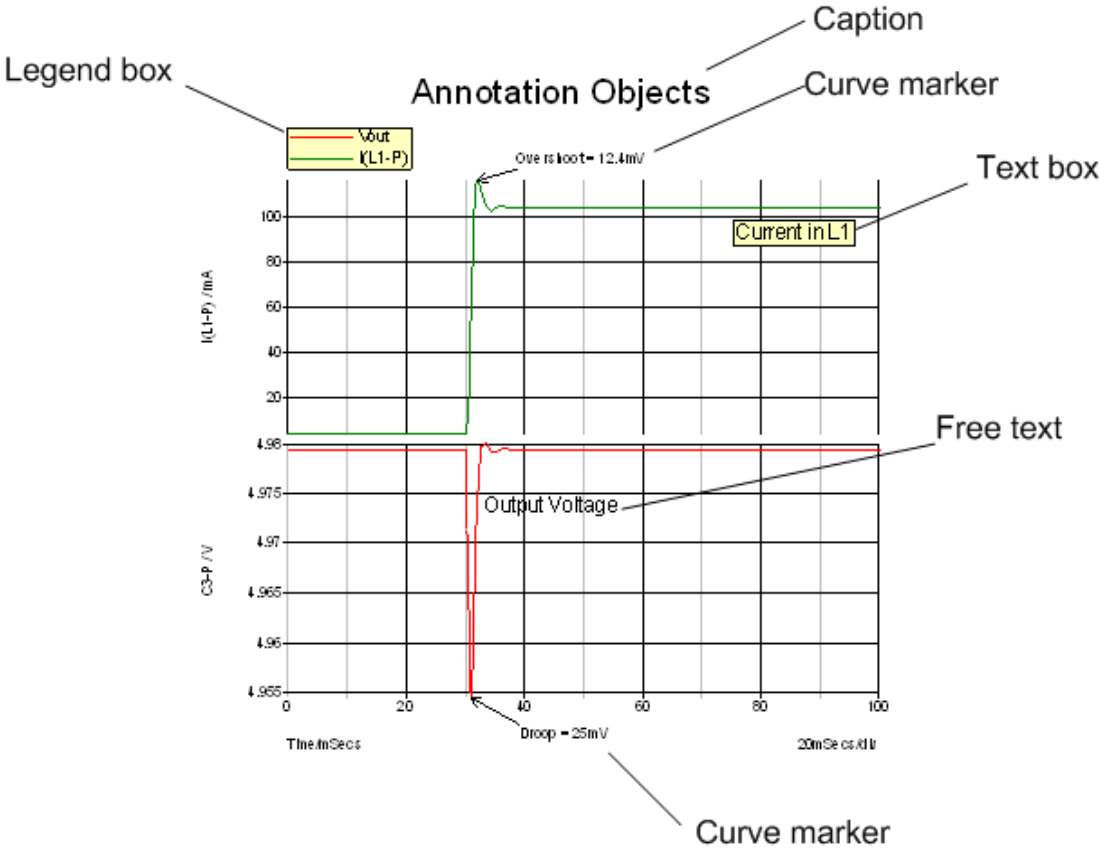

Figure 7-12. Annotation Markers

7.4.1 Curve Marker

When you place a Curve Marker it attaches itself to what it deems to be the first curve that was plotted. The next one goes on to the next curve and so on. It will not put more than one marker on any curve until all curves have at least one marker attached.

1. On any curve in the Waveform Viewer, go to **Annotate|Add Curve Marker**. It appears immediately attached to the right hand end of the curve. You can click on the label and move it but the arrow remains at that point. You can move the arrow; move the cursor near it and observe that it changes form. In that form it will pick up the arrow and allow it to be moved elsewhere on the curve.
2. Double click the Curve Marker to open the Edit Curve Marker dialog which offers only one label in this case. Experiment with what you can achieve in composing a single label. One possibility and its curve are shown as Figure 7-13.

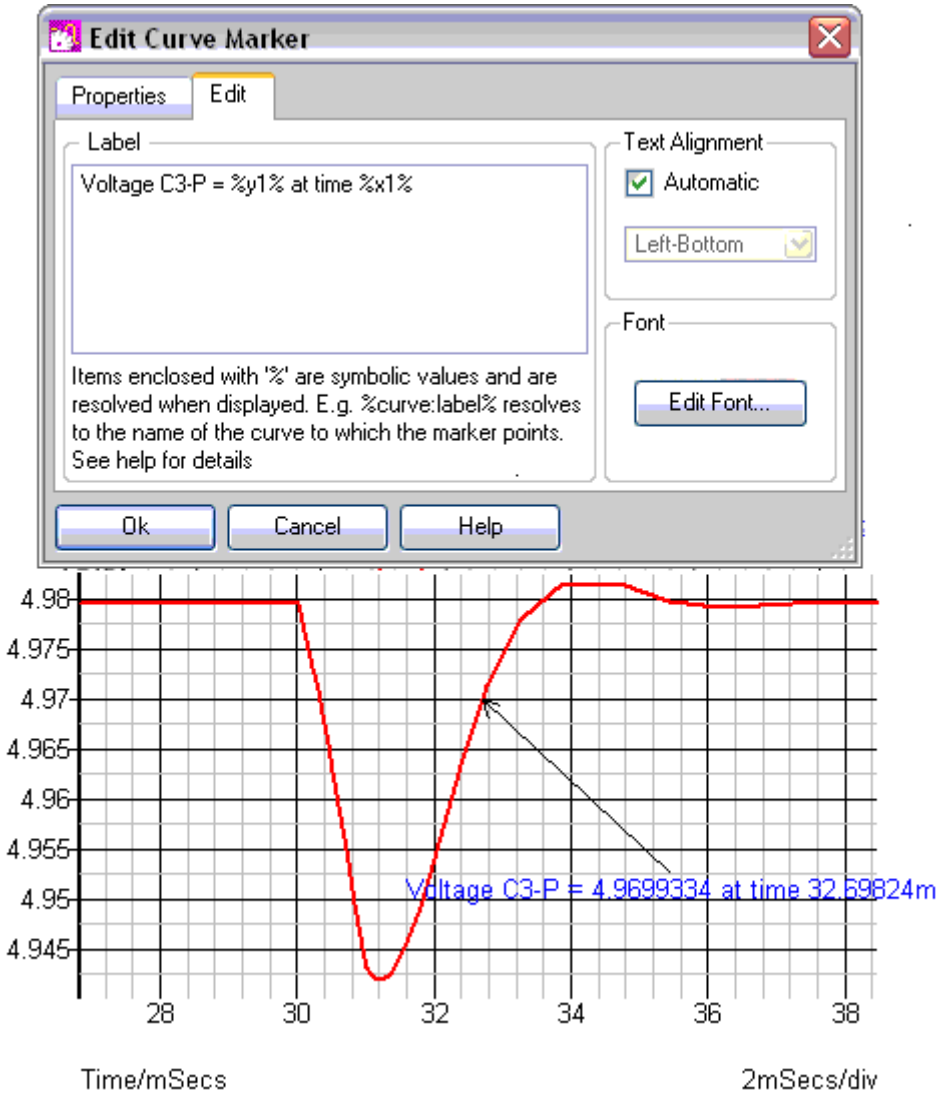

Figure 7-13. One Possibility for Curve Marker Composition

7.4.2 Notes:

- If you move the arrow along the curve, the marker updates the co-ordinates that it displays.
- If you change the font it is effective for this marker only.

7.4.3 Legend Box

A Legend Box is illustrated showing two curves. You can pick it up and move it to different location. The background colour is configurable.

If you attempt to print a sheet without a Legend Box present, you will be asked whether you want one added automatically.

7.4.4 Other Annotation Types

The remaining annotation types are straightforward and need no explanation. All have similar characteristics such as double click to edit and move to any location.

7.5 Exporting and Importing Data

SIMetrix provides the facility to import and export data and to export graphics. The data formats available are as follows:

- Export resizeable graphics in vector format, specifically:
  - Windows Metafile (.wmf)
  - Enhanced Windows Metafile (.emf)
  - XML Scalable Vector Graphics (.svg).
- Export bitmap graphics:
  - Portable Network Graphics (.png) is the default output format.
  - JPEG
  - BMP.
- Import and export graph data as tabulated ASCII values.

7.5.1 Data Export

1. In the Waveform Viewer, select the curve or curves whose data is to be exported.
2. Go to **Edit|Copy ASCII Data**. If you did not select a curve, a further dialog appears asking you to do so. The data associated with that curve is now on the Windows clipboard.
3. Paste it into Microsoft Excel and you will see that the first row contains the vector names of the axes. If multiple curves from the same simulation run are pasted, the Time column will appear only in column A and the y-axes values stretching out to the right. If plots from multiple simulation runs (which will have different time values) are pasted, they will give separate data going from left to right in Excel.

7.5.2 Data Import

For data import to be successful, the data must be in exactly the same format as ASCII export data, specifically:

- Must be tabulated ASCII values with columns separated by tabs.
- The first column must contain x-values.
- Other columns must contain y-values
- The first row must contain vector names. It is recommended to use the V() and I() notation to indicate voltage and current. For example, use V(OUT) instead of VOUT. SIMetrix will then know what units to assign to the axes.

To import data from the Windows clipboard into the Waveform Viewer, go to **Edit|Paste ASCII Data**.

7.5.3 Graphics Export

We can use two methods:

7.5.3.1 Copy and Paste

1. In the Schematic Editor, go to **File|Save Picture...**. Add .emf to the file name to ensure that this is the format used

   OR

   In the Waveform Viewer, go to **Edit|Copy Graphics|Colour**.
2. Paste into the target application. If available, use **Paste Special...** and select 'Enhanced Metafile' format. If you save as monochrome (and optionally if saving in colour) the output file will carry markers enabling plots (which may all be black) to be identified. This may be useful if the copy is eventually photocopied in monochrome.

7.5.3.2 Save As...

The only Save As... options in the Schematic Editor are Schematic Files, Components and earlier versions of SIMetrix.

1. In the Waveform Viewer, go to **File|Save Picture...**
2. In **Save as/ Type**: select 'Windows Meta Files (.emf, .wmf)'
3. Enter filename without extension and click Save
4. Import to the target application as file.

7.5.4 Graphics Import

There is no facility to import graphics.

7.6 Background Information relating to Waveform Viewer

7.6.1 Persistence and Default settings

By default it will plot one graph on top of the last one, a different colour by default. If you set the Persistence option some of the last few curves will be deleted. As supplied, zero is the default and means all will be kept. Persistence is the number (of the most recent curves) that it will keep. The default box and zero do not mean the same thing. Zero keeps all, default causes the viewer to go in search of any global setting and give that priority. Global Persistence can be set in **Command Shell|File|Options|General**… Graph/Probe/Data Analysis tab, Fixed probe global options.

7.6.2 Identification of Curves in the Legend Panel

Each time you run a simulation it creates a data group which is assigned a unique name relating to the analysis type, e.g. tran1, ac4.

In the Command Shell, go to **Graphs and Data|Change Data Group** . This gives you the opportunity to select which Data Group you want to use. When you run a simulation it automatically makes the one you have just run the current one.

You can open the data from earlier runs and random probe but you cannot change the names. If it is a transient, they will be called tran1, tran2 etc.

7.7 Plotting Arbitrary Expressions

The various fixed and random probes provide cross probing for most applications. You are not, however, limited to the facilities provided on the menus. You can plot any arbitrary expression of any combination of circuit voltages and currents. You do this after the simulation is complete using the **Probe|Add Curve...** menu.

You can also specify an expression to be automatically plotted using the **.GRAPH** control. See The .GRAPH Facility on page 82 and Fourier Automatic Update on page 93.

In this illustration we are going to look at the flux in the transformer which is proportional to the sum of I x N or each winding.

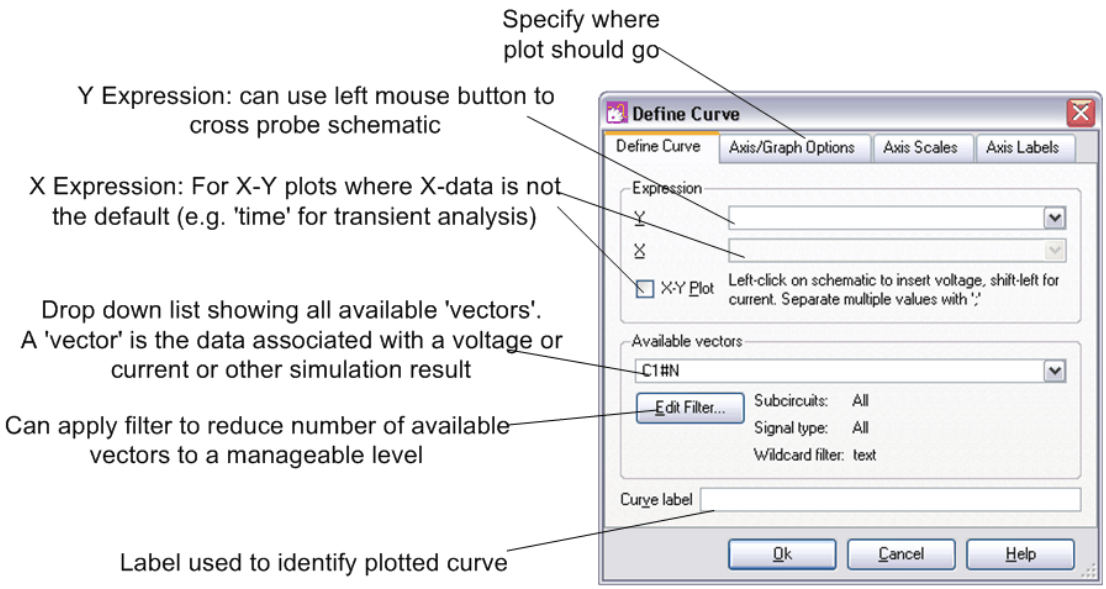

Figure 7-14. The Add Curve Dialog

1. Open **My Documents\SIMetrix\Training\Section-7\rectifier.sxsch**.
2. Go to **Probe|Add Curve...** or Command Shell menu **Graphs and Data|Add Curve...** (the Define Curve dialog opens with either). Figure 7-14 gives an explanation of all the fields on this dialog and there is also additional explanation provided under The Define Curve dialog on page 82.
3. In the dialog, on the Define Curve tab, Expression Group, Y - Shift-click the top of the transformer primary. This enters the name of the current TX1#10.
4. Add *5+ to the expression
5. Shift-click the top of the secondary. The result should be: TX1#10*5+ TX1#20.
6. Click OK.
7. Run the simulation and the curve will look something like Figure 7-15.

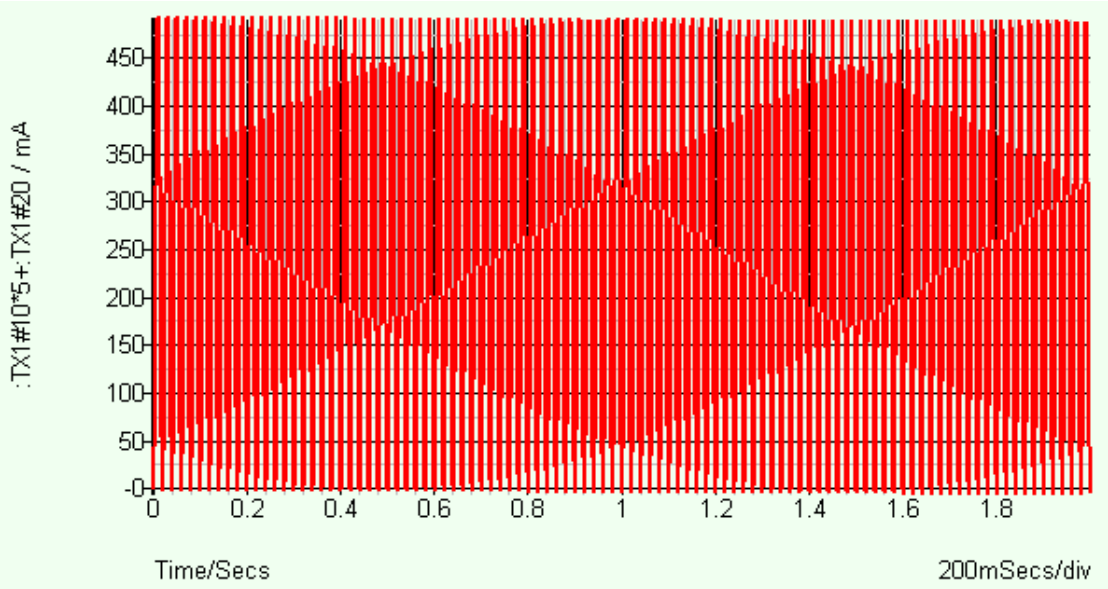

Figure 7-15. The Flux in the Transformer

7.8 Working with Curves

7.8.1 The Define Curve dialog

The main purpose of this dialog is to plot expressions of any arbitrary combination of resistance, currents etc. The main point for this training is to enable you to enter and evaluate expressions which you cannot do from the menus.

It is a semi-modeless dialog in that, while open, it will allow you to click on a node on the schematic and it inserts the voltage at that point in the Expression field. Holding the shift key gives the current at that point; it is essential that you click on the pin so that you get the current flowing into the device.

The drop-down box on both X and Y expression fields give the most recently used vectors. There is also a list of 'Available vectors' in a group of that name and a filter to limit the list if that is necessary.

If you leave the Curve label field blank the curve will be labelled with the expression.

On this dialog the Axis/Graph Options tab, Axis type Auto select will plot the graph on the most appropriate axis or create a new one if necessary. You can force it to use the selected axis regardless of whether it is appropriate. You can also choose a new axis or a new grid. It defaults to Auto.

7.8.2 Update Curves

The Update Curves facility on the Waveform Viewer rebuilds an existing plot with the most recent simulation data.

1. Open **My Documents\SIMetrix\Training\Section-7\rectifier.sxsch** and run the simulation
2. Plot the current in D1 using the context menu **Probe Current...**
3. Change R2 to 500m and run the simulation again.
4. Use **Probe Current...** again and the curve is re-plotted using the latest simulation data.

7.8.3 Update Curve Settings

On the Waveform Viewer, the **Plot|Update Curves Settings...** provides some options for the Update Curves facility:

- Specify that old curves are deleted (the default).
- Specify that any selected curves are not updated (the default).

7.8.4 The .GRAPH Facility

.GRAPH is a simulator command that may be placed in the netlist. (Netlist is explained at Appendix D.)

You can manually place **.GRAPH** commands in the schematic to plot circuit signals not available through the standard fixed probes.

To place in the schematic use the F11 window. This is an edit box that is opened when you press F11 in the schematic editor. Everything in the F11 window is passed to the simulator via the netlist.

If you want to enter an expression in **.GRAPH** you need to know the name of the voltage or current to be used in the Expression.

On the Schematic Editor, as you move the cursor round, you see the status line block 5, says NET= and that gives you the Netname and the Netname is also the voltage name for the voltage at that point. This indication always displays the voltage name.

To get the current name put the cursor on the pin and type Ctl+P (don't worry that it changes to a pen symbol), the name of the pin, and hence the name of the current vector appears in the Command Shell. (If Ctl+P calls up the printer dialog it may be because SIMetrix does not have focus.)

There is a good example of the use of .GRAPH at Fourier Automatic Update on page 93.

7.9 Plot Journal

A plot journal is a SIMetrix script that will build a graph and its curves. A plot journal does not store any data; it is simply a sequence of instructions to create a graph using current simulation data.

Plot journals are created directly from an existing graph. They are useful in situations where you may have spent a little time constructing a number of curves displayed in a particular layout and wish to reproduce this at some later time.

In summary:

- The plot journal does not save the graph itself. Instead it saves a set of commands to recreate the graph from scratch
- A plot journal is a SIMetrix script and uses the same script language as covered in the Scripting section of this training.

A practical illustration is as follows:

1. On the schematic editor, open **My Documents\SIMetrix\Training\Section-7\rectifier.sxsch** and run the simulation
2. Go to **Probe|Voltage Differential...**. Click on the anode of D1 then the anode of D2
3. Go to **Probe|Current...** and click on the anode pin of D1.
4. Select waveform viewer menu **Curve|Stack All Curves**
5. To record the Plot Journal script, go to **Plot|Create Plot Journal**
6. Enter a filename such as **journal**. This will be saved with the **.sxscr** extension which is the standard SIMetrix script extension.
7. Close the waveform viewer.
8. Run the plot journal using the Command Shell menu **File|Graph|Run Plot Journal**. You should see the same graph as in step 4.

Note: Plot journal is not guaranteed to work in all cases as it relies on the expression used to create a curve being stored in the curve itself. This will usually be the case if the curve is created using the standard probe menus or fixed probes. But it won't work for performance analysis or histogram curves for example and may not work for curves created using user scripts.

7.10 Save and Restore a Graph

1. To save a graph, select **File|Save As...**
2. To restore use Command Shell menu **File|Graph|Open...** . You can also double click the **.sxgph** file in Windows Explorer if you allowed the association between this extension and SIMetrix to be made during installation.

Note: The process of saving graphs and restoring them can sometimes change the colours of curves. When you restore a graph, Restore takes the next colour in the sequence and uses that which might, of course, be different from the last time you saw it. You can change colour but if you change a curve colour using the **Curves|Edit Curve Colour** menu, it will not subsequently take a sequence-colour; it will keep the colour you gave it.

7.11 Data Handling

All simulation runs create data which is stored on disk. In some cases the quantity of data generated can be very large and consideration needs to be given to how SIMetrix handles this data.

If you only ever run short simulations you probably don't need to worry about data handling.

7.11.1 Locating SIMetrix Data Files

All simulation data is stored in a disk file located in temporary folder. To find where this is located:

- Select **File|Options|General...**
- Select File Locations tab
- See the 'Temp Data' item in the list - this is the full path where the data items are stored.

If you look in the specified folder you will notice that files with the extension **.sxdat** are created when you run a simulation. These are the simulation data files. This folder will be cleared on all **.sxdat** files at certain times: exactly when is controlled by an option setting.

7.11.2 Managing Large Files

If your data files are filling up your disk, you can:

1. Choose a location with more disk space.
2. On the Command Shell, go to **File|Options|General...**
3. Select the **File Locations** tab
4. Double click the 'Temp Data' entry in the list
5. Locate a suitable path and click OK. **Always** choose a local drive. **Never** locate temp files on a network drive. If you put the temporary data on a network drive it will both slow down the simulation quite seriously and lose the data if the network connection fails while the simulation is running.

If you have a problem with file size, visit:
www.SIMetrix.co.uk/site/support/kb/HandlingSimulationData.htm

Note: File compression does not work well with data files.

7.11.3 Controlling When SIMetrix Releases Files

For a long simulation, it is a good idea to monitor the files in the temporary folder. If they get large, say greater than 1GB, you should change the option setting that controls when these files are deleted.

- On the Command Shell, go to **File|Options|General...**
- Select the **Graph/Probe/Data Analysis** tab
- Note the options in **Temporary data file delete options**. The default setting is **When program starts**.

The explanations of the options are:

- **Never**. If you select **Never**, although it doesn't delete them it will over-write them. During a session, the first transient analysis you do will be called tran1, the second tran2 and so on. If you then shut down SIMetrix and restart it, the first transient analysis will be called tran1 and over-write the tran1 that is already there. There is no warning that this over-writing is about to take place.

  This is intended as a temporary folder; if you wish to keep anything that appears in it you must copy it out to a different and more permanent folder before closing down SIMetrix. Working in the temporary folder, the assumption is that the files are owned by the program. If you want to take ownership of some data you must copy it somewhere other than the temporary folder.
- **When program starts**. This is the default. Clears the folder every time you start SIMetrix.
- **When program closes**. A little bit more aggressive than when program starts.
- **When data is no longer needed**. When this option is not set, SIMetrix keeps links to the data for the three most recent runs. At the fourth oldest run SIMetrix 'releases' the data from the program but the data file still exists. With this option set, the data file is also deleted at this point.

To work with the results of a previous simulation, in the Command Shell go to **Graphs and Data|Change Data Group**. This lists the data groups to which SIMetrix has maintained its links.

A further possibility is, in the Command Shell, to go to **File|Data|Load...** and SIMetrix will load the files it finds in the current directory. If you go to **File|Data|Load Temporary Data** SIMetrix will load the files it finds in the temporary directory. After you have opened such a data group it is marked as permanent and will not be over-written by subsequent simulations.

8 Fourier Analysis

8.1 Introduction

Fourier analysis applies a Fourier transform to time domain data to obtain the data in the frequency domain. In SIMetrix, Fourier analysis is not an analysis mode but a post-processing operation performed on simulation data after a run is complete.

A great deal of care must be taken when carrying out Fourier analysis; there are numerous pitfalls and it is possible to obtain results that are quite erroneous without realising it. This part of the training explains the issues that affect the quality of the result and provides a benchmark against which you can judge whether your result is likely to be a good one.

It is important to distinguish between two distinct issues which are:

- **Accuracy of the Source Data** - This concerns the quality of input to the Fourier Analysis and is unconnected with the Fourier analysis itself. We will see how to achieve this in Accuracy of the Source Data next.
- **Accuracy of Analysis Processing** - This concerns what the whole process (interpolation and analysis) does with the input and what can go wrong. We address this in Accuracy of Analysis Processing on page 88.

This training illustrates Fourier analysis of a known sine wave because:

- The correct result is known in advance
- It illustrates the problems that might be encountered
- It is quite a reliable predictor of the accuracy that might be achieved with other inputs.

The abbreviations used in this section are DFT (Discrete Fourier Transform) and FFT (Fast Fourier Transform).

8.2 Accuracy of the Source Data

8.2.1 First Attempt at Fourier Analysis

1. Open **My Documents\SIMetrix\Training\Section-8\amp.sxsch** in the Schematic Editor.
2. Run the simulation. **Amp.sxsch** does not have any probes attached so no plot will appear.
3. Select menu **Probe|Fourier|Probe Voltage Custom...**. A probe appears on the schematic.
4. Probe the input (top end of V1) and the Define Fourier Plot dialog appears.
5. Accept the defaults and click OK. This plots the waveform shown as Figure 8-1.

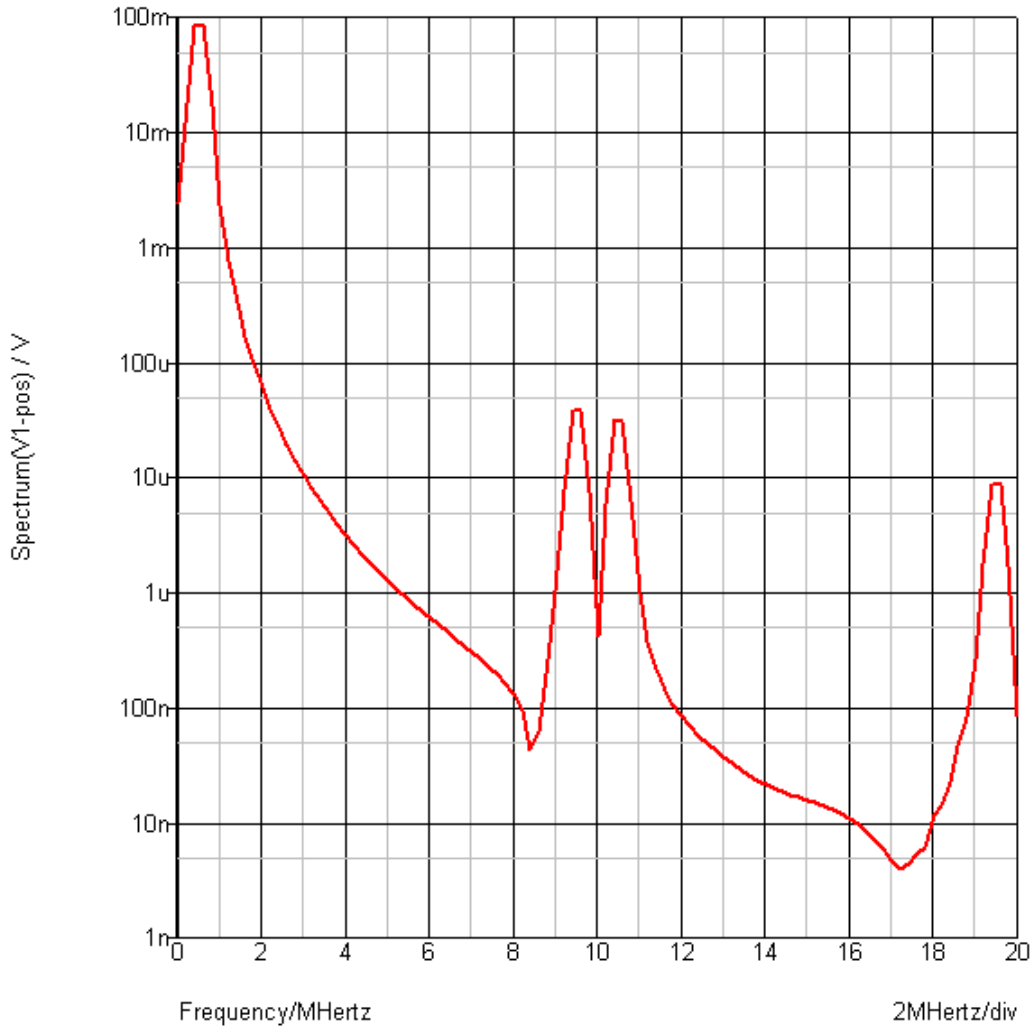

Figure 8-1. Fourier Analysis of Sine Wave - First Attempt

On the schematic, V1 is generating a 500KHz sine wave. We know what a Fourier analysis of this should look like; it should be a sharp spike at 500KHz and nothing much else. As you can see, the result here deviates from what we expect; there are two components 500KHz either side of 10M and an additional component at 19.5MHz. There are two things we can do to improve this. The first is to increase the number of calculation points (reduce the maximum time step). We will do this now.

8.2.2 Increase the Calculation Points

1. Go to **Simulator|Choose Analysis** then click the Advanced Options button
2. Set the maximum time step to 50nS.
3. Close and run the simulation again.
4. Probe as before and click OK. This adds the green curve shown as Figure 8-2. We have got rid of two peaks around 10MHz but not the one at 19.5MHz.

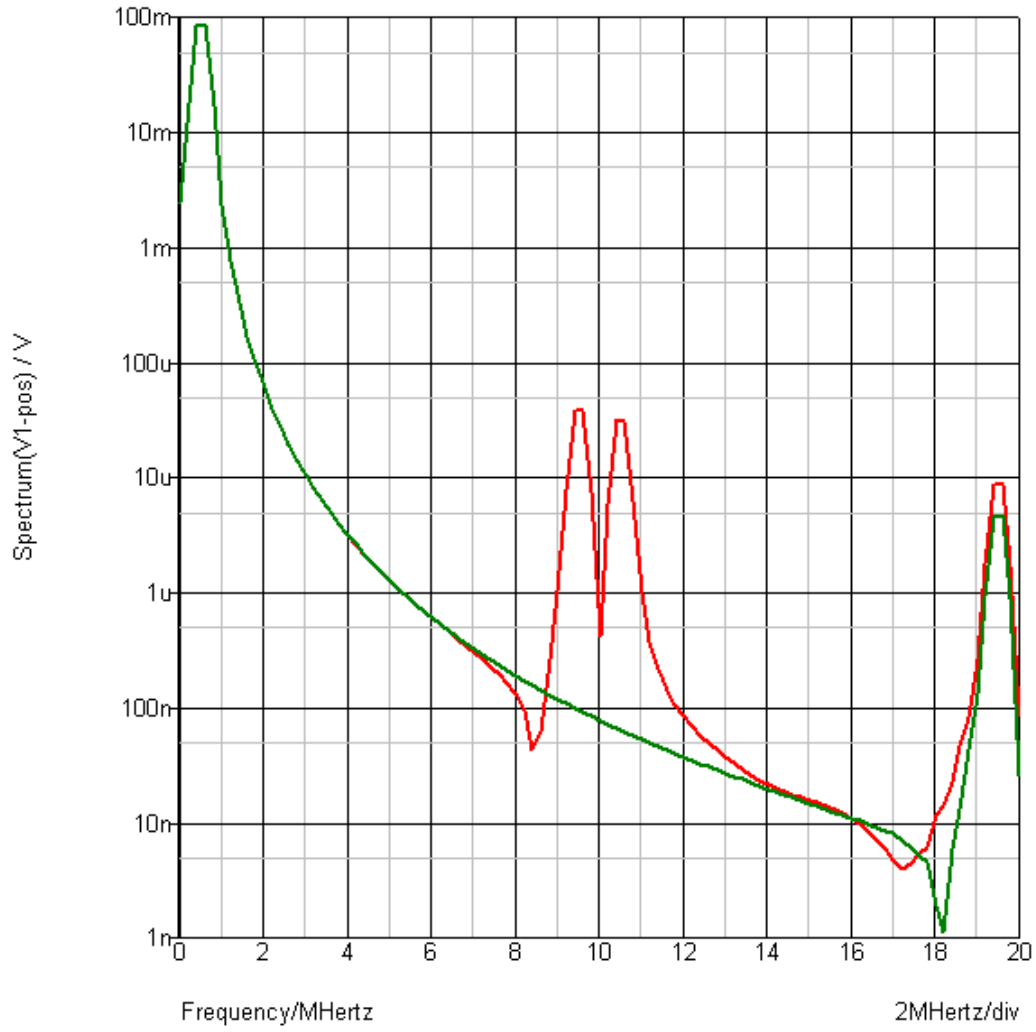

Figure 8-2. Fourier of Sine Wave - 50ns Max Time Step

8.2.3 Increase the Calculation Points Further

1. Go to **Simulator|Choose Analysis** then click the Advanced Options button
2. Set the maximum time step to 20nS.
3. Close and run the simulation again.
4. Probe as before and click OK. This adds the blue curve shown as Figure 8-3.

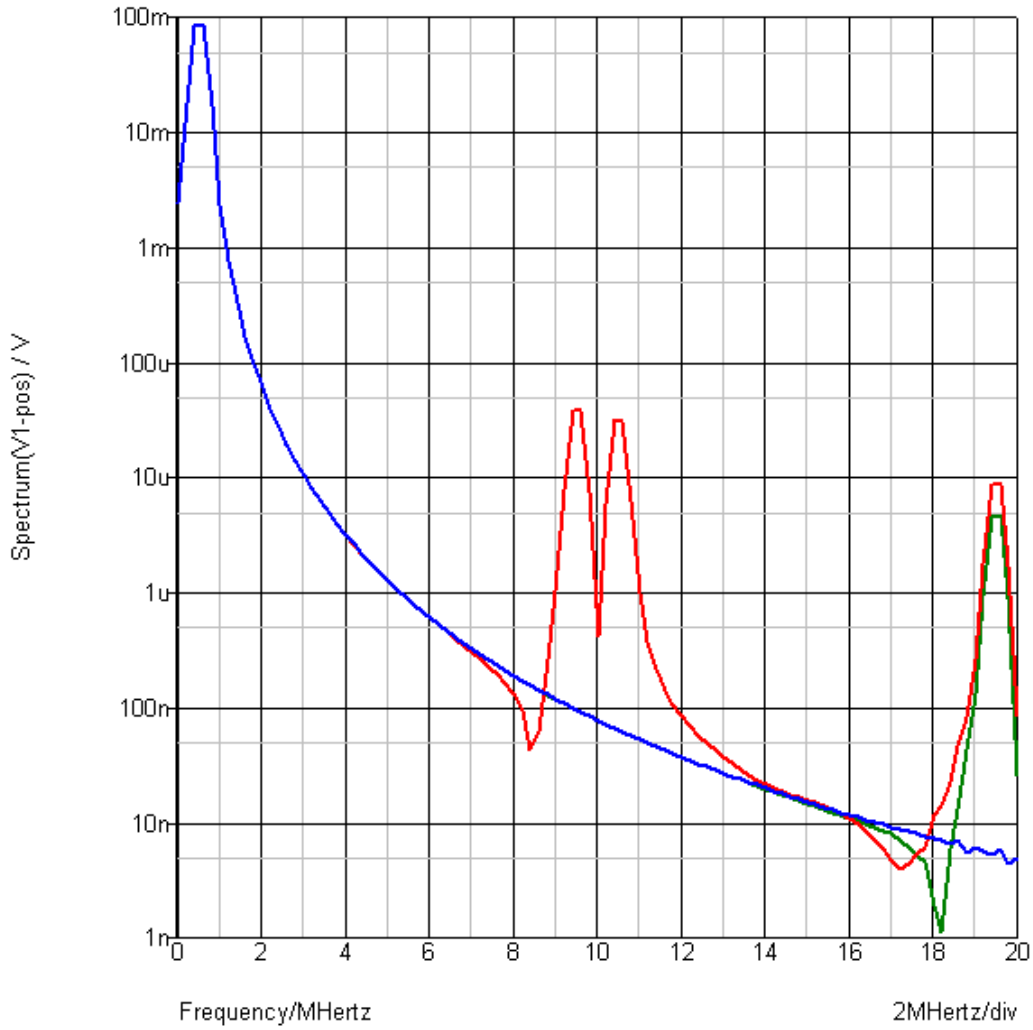

Figure 8-3. Fourier of Sine Wave - 20ns Max Time Step

This is as much improvement as we need (or can get) by reducing the maximum time step. The analysis is very sensitive to time step initially; you just need to make sure it is low enough.

Tip - If you haven't got a sine wave in your schematic, put one there temporarily, plot its Fourier spectrum then use that as a benchmark. This will give you more confidence that the real analysis is valid.

Note: The blue curve on Figure 8-3 still shows some error. Notice that the curve decays rather slowly away from the fundamental at 500kHz. There should be a narrow peak at 500kHz falling rapidly on each side. This is caused by 'spectral leakage' and is a result of the fact that we performed our analysis on an incomplete number of cycles. Fourier analysis gives best results when performed on a whole number of cycles. We will see how to do this in the next section. For more information on spectral leakage see Spectral Leakage and Windowing on page 93.

8.3 Accuracy of Analysis Processing

Fourier analysis places some requirements on the input it will reliably process:

1. The Discrete Fourier Transform (DFT) requires that sampling points are equally spaced.

2. The Fast Fourier Transform (FFT), which is simply an efficient algorithm for calculating the DFT, requires equally spaced sampling points AND that the number of sampling points equals 2^n where n is an integer.
3. To get the best results, the data presented to a Fourier Transform should be a whole number of cycles.

Simulators generally do not produce data with equal sampling steps and so do not comply with 1. and 2. above. In order to resolve this, the data is interpolated. Interpolation is a method of constructing **new data points** from a discrete set of known data points. Interpolation, however, is vulnerable to aliasing (see Interpolation and Aliasing on page 93). We will see the effects of aliasing in a later section and explore ways of reducing it.

SIMetrix offers an alternative algorithm, Continuous Fourier, which is not subject to conditions 1. and 2. above and therefore does not suffer from aliasing. We will also demonstrate this in a later section.

Firstly, we will look at how to perform the analysis on a complete number of cycles to reduce the spectral leakage we see in the curve in the blue curve in Figure 8-3.

8.3.1 Truncate the Data

SIMetrix allows you to specify the exact fundamental frequency. If the algorithm **knows the fundamental frequency**, it can work out how to truncate the data so that the Fourier analysis algorithm is presented with an exact number of cycles. This will dramatically reduce spectral leakage. We demonstrate this now.

1. Go to **Probe|Fourier|Probe Voltage Custom...**
2. Probe the V1P and the Define Fourier Plot dialog appears.
3. On the Fourier tab, Signal Info group, check **Know fundamental frequency** and set the Frequency field to 500KHz. (500kHz is the frequency of the sine source V1)
4. Click OK. This plots the waveform shown as Figure 8-4.

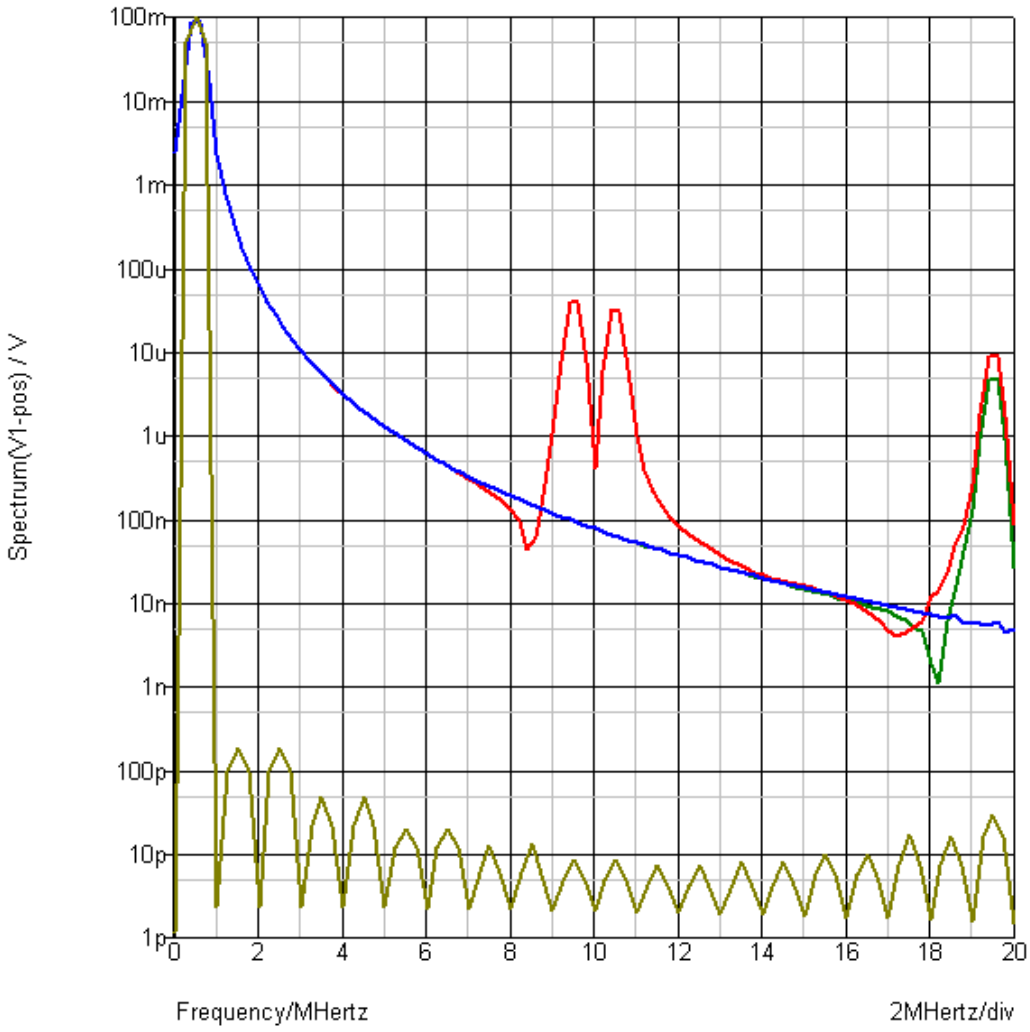

Figure 8-4. Fourier - Know fundamental frequency

As you can see this is a dramatic improvement. Above 4MHz we are seeing what amounts to noise at around 100pV, nearly -180dB from the fundamental.

8.3.2 A Demonstration of aliasing

1. Open **My Documents\SIMetrix\Training\Section-8\rectifier.sxsch** and run the simulation.
2. Go to **Probe|Fourier|Arbitrary...** then select the Define Curve tab.
3. Press and hold Shift and click on the anode of D1. D1#P (the signal name of the current in D1) appears in the Expression field.
4. Click the Fourier tab on this dialog and ensure that the values are as shown in Figure 8-5. (Resolution/Hz is greyed out). In particular:

   Check **Know fundamental frequency** and set to 50Hz
   Set Stop freq./Hz to 200

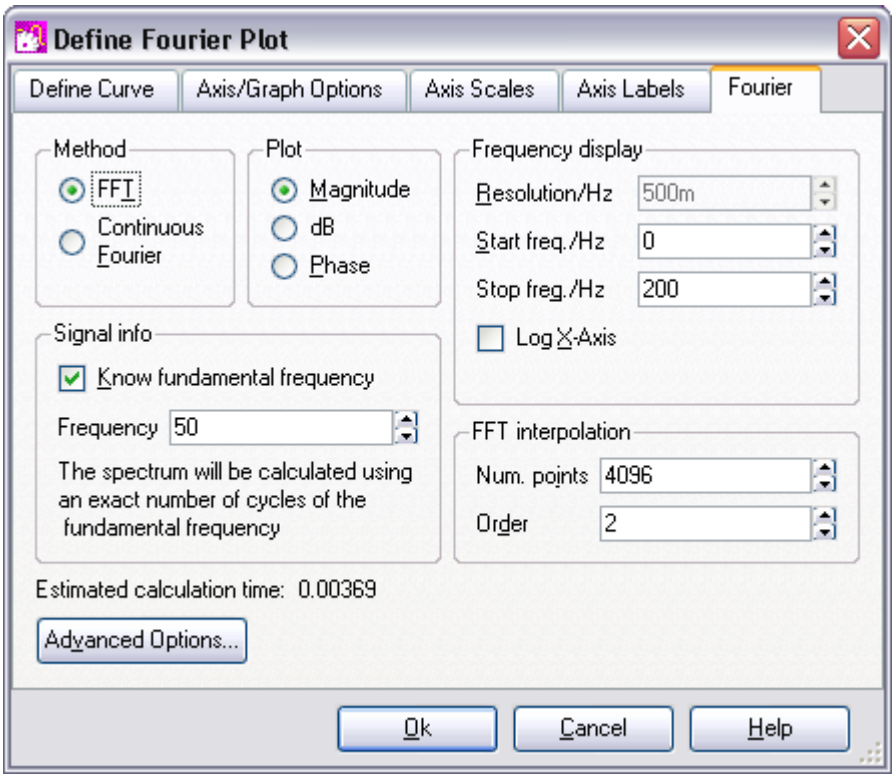

Figure 8-5. The Define Fourier Plot dialog

5. Click OK and a curve as shown as Figure 8-6 will be plotted. If it is not, check that you have used every detail in the preceding steps.

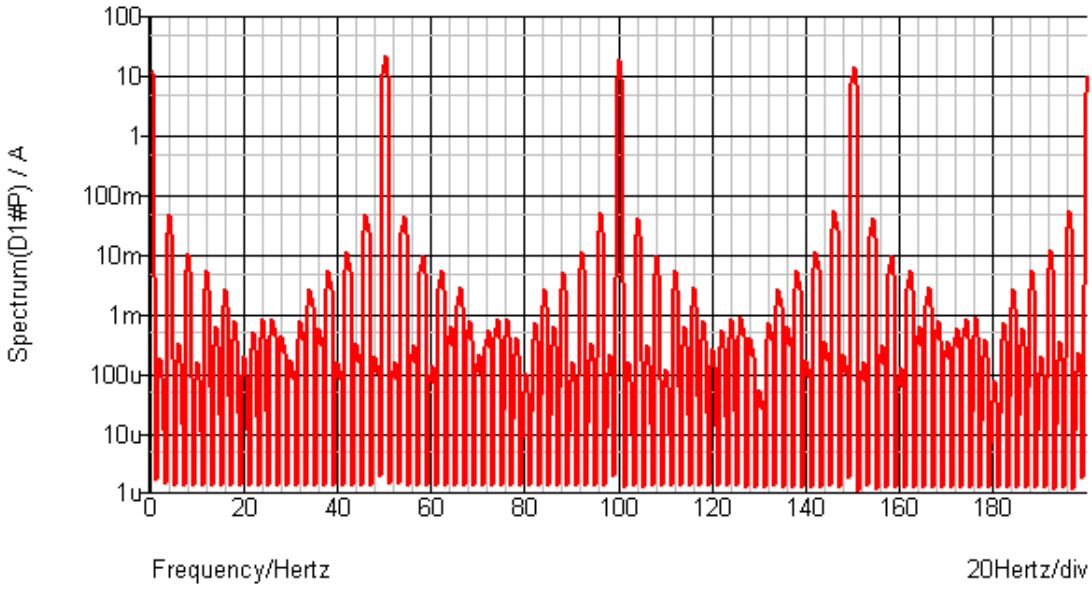

Figure 8-6. Fourier of Rectifier 4096 Interpolation Points

6. Identify the peaks at the fundamental and major harmonics at 50, 100, 150Hz etc. All the other signals are aliases (see Interpolation and Aliasing on page 93) which have appeared because we have an insufficient number of interpolation points. We can improve on that.
7. Repeat the plot (using the above steps) but use 131072 FFT interpolation points. Click OK and a curve like Figure 8-7 will be plotted.

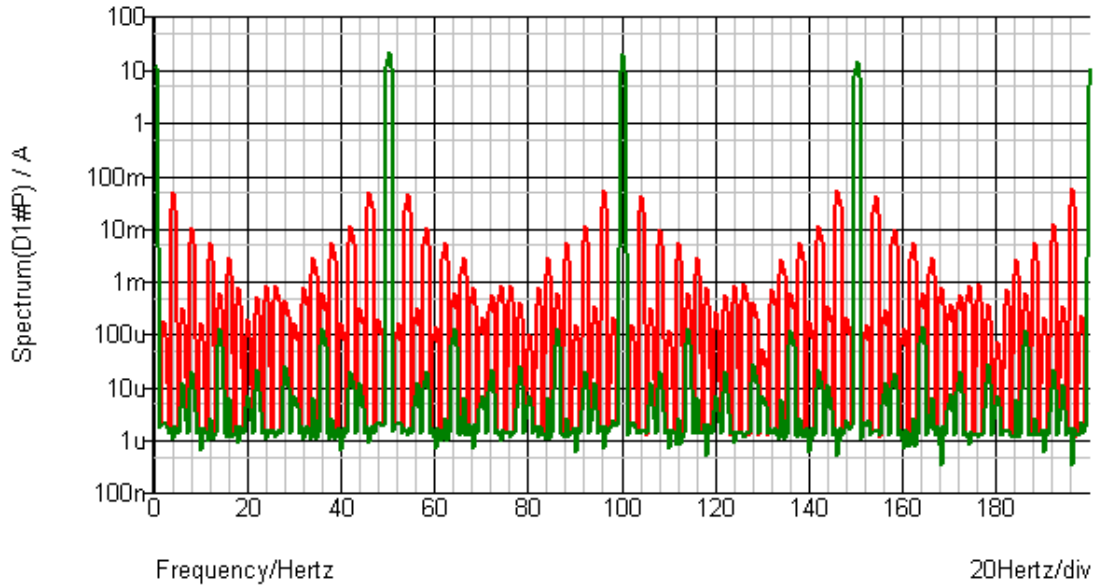

Figure 8-7. Fourier of Rectifier 131072 Interpolation Points

This is a big improvement but aliases still remain.

8.3.3 Continuous Fourier

There is an explanation of Continuous Fourier at Continuous Fourier - Explanation on page 94; this is a practical illustration. With **rectifier.sxsch** open in the Schematic Editor:

1. Close the waveform viewer window.
2. Go to **Probe|Fourier|Arbitrary...** then select the Define Curve tab.
3. Press and hold Shift and click on the anode of D1 or type D1#P in the Expression field.
4. Click the Fourier tab on this dialog and, in the Method group, select Continuous Fourier. The FFT interpolation group greys out and the Estimated calculation time increases significantly. With continuous Fourier, always check the calculation time as it can be high. The operation cannot be aborted once started.
5. Click OK and a curve such as Figure 8-8 will be plotted.

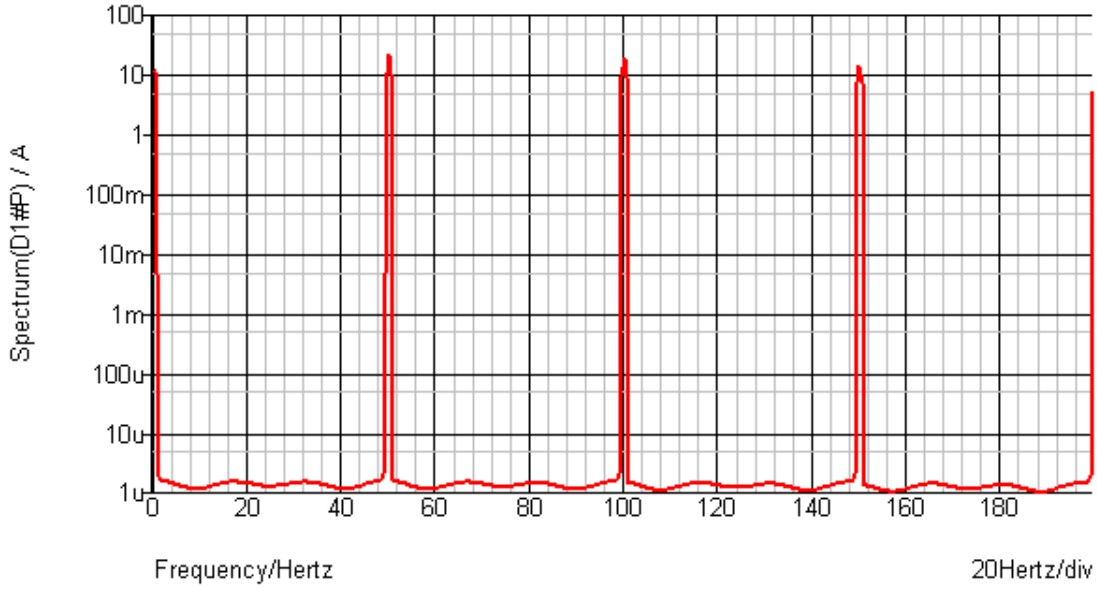

Figure 8-8. Continuous Fourier on Rectifier

Notice that, with Continuous Fourier, aliases have been eliminated.

Tip. Although the demonstration here shows Continuous Fourier delivering a superior result, the general advice is to use the FFT in most cases. It is only when a signal has a very high frequency content that Continuous Fourier produces a significantly better result. If in doubt, try both.

Tip. With any kind of spectral analysis, it is good practice to put some form of control in place to ensure that it gives the expected results from known input. If you don't get a sensible result with a sine wave you are not going to get a sensible result with any other signal.

8.3.4 Fourier Automatic Update

Up to now we have been randomly probing and in some cases this can be somewhat repetitive. There is no fixed probe for FFT but it is possible to set up the analysis so that a Fourier plot is created automatically on each run.

On the Schematic Editor press F11 to split the window. In the lower pane, add a new line (anywhere within the code) and enter:

.graph "spectrum(d1#p, 32768)" ylog=log xmax=200 xmin=0

The lines in this pane are appended to the NetList. There is no need to save before running the simulation again. A 'Modified' flag appears on the status line when you make any change and it will prompt for saving before you close the schematic.

.GRAPH is the underlying command used to implement fixed probes.

For documentation on **.GRAPH**, see The .GRAPH Facility on page 82 and the Simulator Reference Manual Chapter 6.

8.4 Fourier Background Material

8.4.1 Spectral Leakage and Windowing

When presented with incomplete cycles FFT produces a phenomenon known as spectral leakage. Spectral leakage can be reduced by applying a Window function to the input data and indeed SIMetrix uses a window function by default. However, it is always best to use complete cycles if possible and when simulating a system with a fixed frequency input this is straightforward as we have seen. It is important to note that using an exact number of cycles does not remove the need for a windowing function as the cycles are unlikely to be exactly repetitive.

By default, SIMetrix uses the Hanning window. This has the effect of producing two additional spectral lines, 3dB down and one either side of the spectrum point of the true frequency.

If you wish to change the window function, click the **Advanced Options...** button in the Fourier tab. Select the Rectangular window option to switch off windowing altogether.

8.4.2 Interpolation and Aliasing

Even with the maximum time step set as low as necessary, it cannot be guaranteed that all the time steps are equal (as FFT requires). The waveform must therefore be re-sampled with data points that are equally spaced and this process is known as **interpolation**.

According to Nyquist's Theorem, if you sample data at a frequency of twice the highest frequency present, you can exactly reconstruct the original data from the sample. If these conditions are not met, then spurious data will appear in the sample and this is known as **aliasing**. If, for example, you sample at 2MHz and signal is present at 1.1MHz, then the sampled data will indicate the presence of a beat-frequency of 900KHz which was never really there (sometimes known as 'wrap around').

In an unknown signal the highest frequency present could be very high although the high frequency components may be at a low level. Determining whether or not you are seeing aliasing is a matter of trial and error. If increasing the number of interpolation points changes the result significantly then this is likely to be due to aliasing.

We used the example of a rectifier to demonstrate this effect as it is common for the current in rectifier diodes to show a high-frequency content. This can be an important factor in the analysis of EMC.

8.4.3 Continuous Fourier - Explanation

In addition to FFT, SIMetrix has a quite different Fourier algorithm called Continuous Fourier. Continuous Fourier works by numerically integrating the Fourier integral. Because it is not a discrete algorithm it does not require sampled data and so does not require interpolation. It follows therefore that it does not suffer from aliasing. But, it is very much slower than the FFT and is more sensitive to large time steps. As well as immunity from aliasing, it also has the additional benefits, that you can specify any range of frequencies for the result. The FFT algorithm by contrast must always calculate the whole spectrum from DC to 1/2p where p is the FFT interpolation period. This benefit can mitigate the increased calculation time if you are interested only in a narrow range of frequencies.

9 Multi-step Analysis

9.1 Multi-step Analysis

Multi-step Analysis is the process of repeating a step a specified number of times while changing some parameter. With one exception you can specify exactly what you are going to change between each step. The only exception is Monte Carlo Analysis described on page 97.

In this illustration we are sweeping a resistor which is acting as a load to a power supply. In this case you could do a run, change the resistor manually, do another run and so on. This illustration shows how all that can be carried out automatically.

9.1.1 Multi-step Example

1. Open **My Documents\SIMetrix\Training\Section-9\hybrid-psu.sxsch** in the Schematic Editor.
2. Place a Fixed Voltage Probe on the output (L1/C3 node).
3. Open **Edit Probe dialog** and set Curve Label to VOUT.
4. Open the **Choose Analysis** dialog.
5. Ensure that in the Analysis Mode group only **Transient** is checked.
6. Check **Enable multi-step** and click on **Define...**
7. Select **Device**
8. In the **Device name** field enter 'R12'
9. In the Step parameters group:
   - Click on **Decade**.
   - Set Start value to 10.
   - Set Stop Value to 1k.
   - Set Steps per decade to 2. This will sweep the resistor from 10Ω to 1000Ω and do so logarithmically.
10. Run the simulation and zoom in to produce a curve such as Figure 9-1. Notice that each of the multiple curves has its own colour and identifier in the Legend Panel.

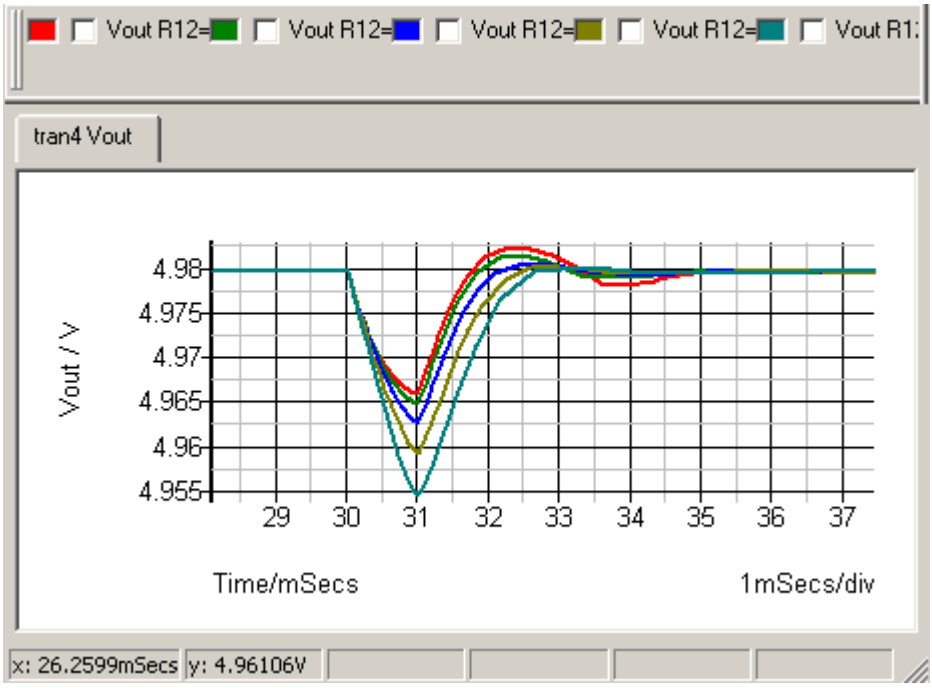

Figure 9-1. Multi-step Analysis on Load Resistor

Several Sweep Modes are available: Device, Parameter (a variable or an expression in curly braces), Model parameter, Temperature (this sweeps the global circuit temperature) and Monte Carlo.

9.1.2 Performance Analysis

Performance Analysis takes the family of curves, extracts information from each one and builds a new curve from that information. For this purpose we are going to use a 'Goal Function' which does just this. We are going to use the Goal Function 'Minimum' which returns the y-axis value of the lowest point on each curve.

9.1.3 Practical Illustration of Multi-step Analysis

1. Using the same hybrid-psu schematic, open the **Choose Analysis** dialog.
2. Ensure that **Enable Multi-step** is checked.
3. Click **Define** and change the **Steps per decade** to 5.
4. Check the **Group curves** box. This can make the initial curve display look less cluttered.
5. Run the simulation and zoom in to see the busy part. Leave the Waveform Viewer open.
6. On the Schematic Editor, go to **Probe|Performance Analysis**.
7. In the Expression field enter **Minimum(**
8. On the Schematic Editor click on the output. This appends 'R6 N' to the expression.
9. Type the closing bracket. You should see “minimum(R6_N)” in the Expression box.
10. Click OK.
11. Don't assume at this point that you have made a mistake. SIMetrix used logarithmic axes because **step** was specified as being logarithmic. On the Waveform Viewer, right click to get the context menu and click on **Axes|Edit Axis**.
12. Select **Lin** for the y-axis and click OK. The curve of minimum output voltage against load should look like Figure 9-2.

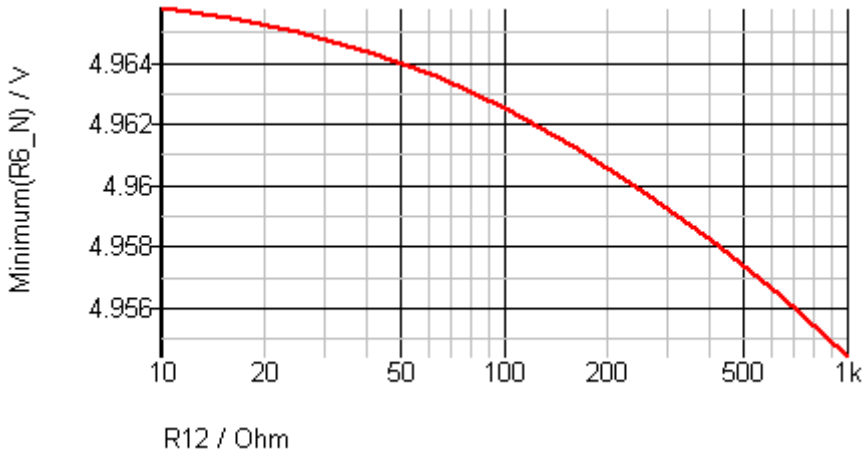

Figure 9-2. Performance Analysis - Minimum Output Voltage against Load

10 Monte Carlo Analysis

10.1 Overview

Monte Carlo analysis is a procedure to assess manufacturing yields by repeating simulation runs with varying applied random variations to component parameters. The technique is very powerful and usually gives a more realistic result than worst-case analysis which varies component values to their extremes in a manner which produces the worst possible result.

Note: SIMetrix components and models are not generally defined with tolerances; you need to specify these as part of the process of setting up Monte Carlo analysis.

10.2 Setting Component Tolerances

For the practical illustration we will use the bandpass filter. Filters provide quite a good example of a Monte Carlo analysis because filters usually require precisely defined component values.

1. Open **My Documents\SIMetrix\Training\Section-10\768Hz bandpass.sxsch**
2. On the Schematic Editor go to **Monte-Carlo|Set All Resistor Tolerances...** This sets all resistors to the same tolerance.
3. Enter 1%
4. Select capacitor C2. Go to **Monte-Carlo|Set Selected Component Tolerances...** and enter 10%.
5. Select capacitor C1. Go to **Monte-Carlo|Set Selected Component Tolerances...** and enter 5%. (You can hold the Ctl key to select multiple items).

10.3 Setup and Run Monte Carlo

1. Open the **Choose Analysis** dialog and set Start frequency to 600, the Stop frequency to 900 and the number of points to 500. Set Sweep to Linear.
2. Click **Enable multi-step**.
3. In the Monte Carlo and multi-step analysis group, click the **Define...** button and set the **Number of steps** to 100 (direct it to do 100 simulation runs).
4. Click OK and this returns you to **Choose Analysis**
5. Click **Run** and, after you have zoomed in appropriately you should get a set of curves similar to Figure 10-1.

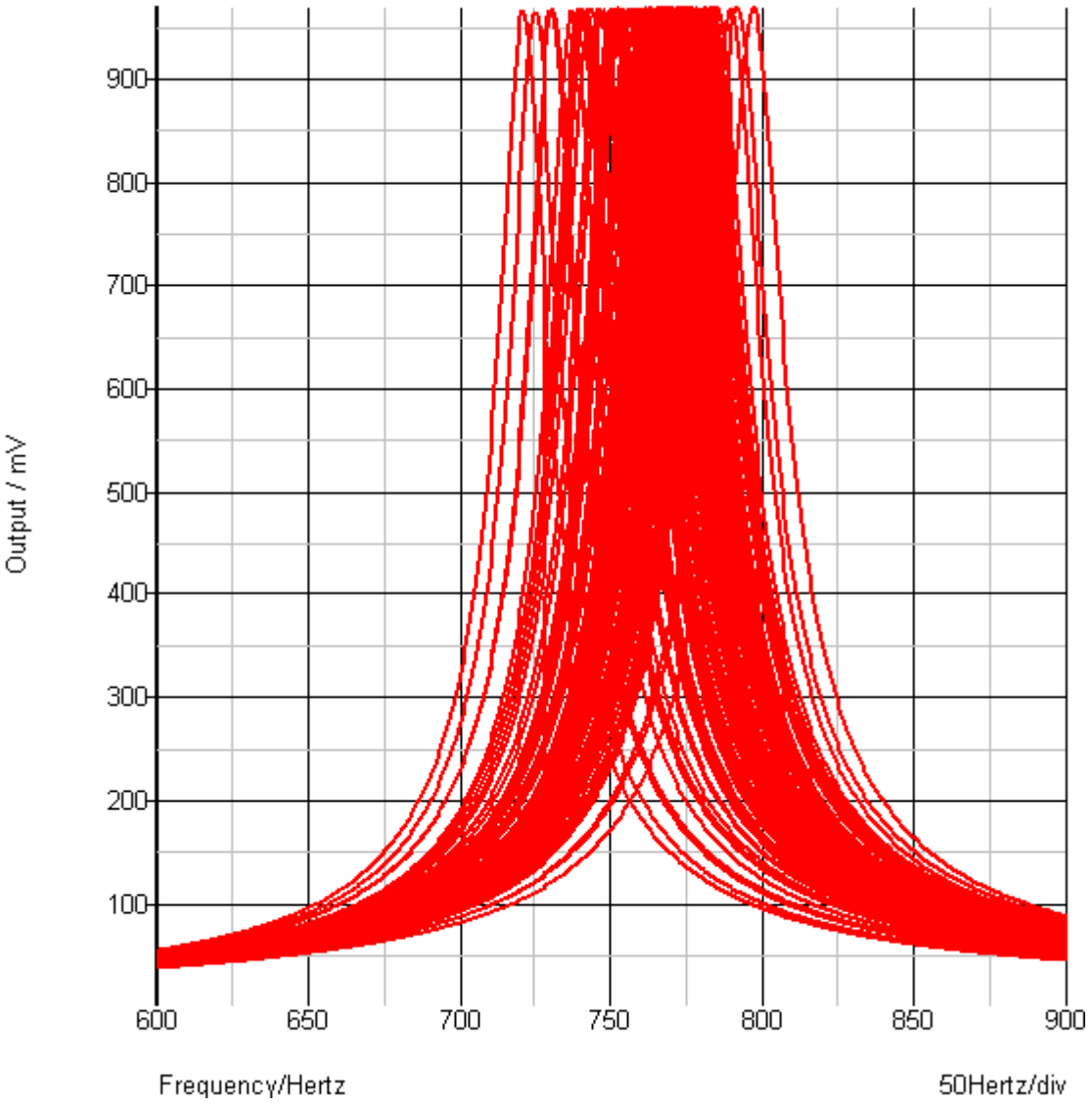

Figure 10-1. Monte Carlo Curves

Note: All these curves are the same colour and have only one identifier in the Legend Panel. It is not possible (using Monte Carlo) to split them out to examine individual curves. Monte Carlo automatically Groups curves such as we did explicitly in Practical Illustration of Multi-step Analysis on page 96.

10.4 Monte Carlo Histogram

Monte Carlo Histogram analysis displays a probability density curve and calculates mean, standard deviation, minimum and maximum values. This allows an assessment of manufacturing yield to be made.

To prepare the framework (or bins) for the histogram SIMetrix takes the specified start and stop frequencies and divides them into a number of bins with evenly spaced frequency.

To populate the framework we are going to use the Goal Function called 'CentreFreq' (it accepts the North American spelling - CenterFreq) to find the centre frequency of all 100 curves. It then puts each centre frequency in the appropriate bin according to the value that the Goal Function returned. The resulting bar graph is usually a normal distribution.

1. In the Schematic Editor go to **Monte-Carlo|Plot Histogram...** . This calls up the Define Performance Analysis dialog. You can click on the Functions button to get a list of all available functions (from the Help system).
2. Type **CentreFreq(** or **CenterFreq(** if you prefer

3. On the Schematic Editor click on the output and notice that R1_P (this is the 'vector' name) has been appended to the Expression field.
4. This function requires two arguments separated by a comma. Append ,3) to the expression (the 3dB point - see Goal Function in Help). You should see “CentreFreq(R1_P,3)” in the expression box.
5. Click OK and notice how long it took to process (only) 100 curves. You should see a histogram similar to Figure 10-2. It is a bit coarse because it needs a reasonable number of values in each bin and with 100 runs there are only 5 bins.

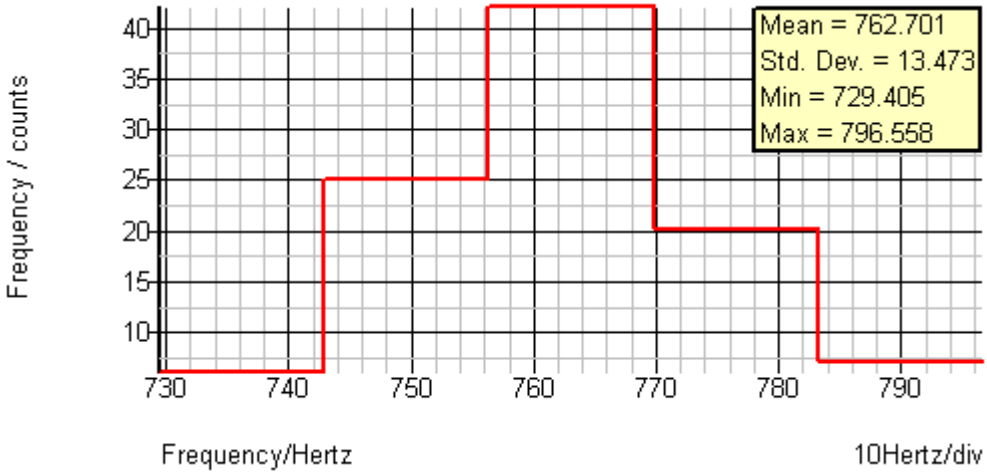

Figure 10-2. Monte Carlo CentreFreq Histogram

The interesting figure here is the Standard Deviation. Manufacturing is often carried out according to so many multiples of the standard deviation (typically between 3σ and 6σ). 97% of units manufactured will lie within 3σ. In the example shown, therefore, 97% of units manufactured will lie within 762.701Hz ±40.419Hz (13.473 x 3 = 40.419). At 6σ the percentage changes to 99.99%.

10.5 Single Step Monte Carlo

Now we are interested in finding out how much the signal was attenuated at the bandpass frequency (768Hz) and for this we will use the single-step feature.

What this does is not a complete sweep; it just measures one point. It then changes all the component values and does another point. So instead of getting a collection of curves you get a collection of single points.

This plot is unusual in that it is not a normal (or Gaussian) distribution. You should be aware that the mean and standard deviation values have limited value in this curve.

10.5.1 Setup and Run Single Step Monte Carlo

1. On the Schematic Editor **delete the probe** because all that would be plotted would be essentially a set of random values.
2. Open the **Choose Analysis** dialog
3. In the Monte Carlo and multi-step analysis group, uncheck **Enable multi-step**.
4. In the Sweep parameters group, click the **Define** button. This calls up the Define Sweep Mode dialog.
5. In the Define Sweep Mode dialog: Set **Sweep mode** to Monte Carlo. Set **Frequency** to 768. Set **Number of points** to 1000.
6. Click OK
7. Run the simulation and notice how quickly it completes processing of 1000 points. Now you need to probe it.

10.5.2 Histogram for Single Step Monte Carlo

1. On the Schematic Editor go to **Monte-Carlo|Plot histogram...**. This opens the Define Performance Analysis dialog with the Expression field blank.
2. On the Schematic Editor, click anywhere on the **output** (where the probe used to be before you deleted it). This will enter R1 P into the Expression field.
3. Click OK and the Waveform Viewer displays the Single step Monte Carlo Histogram which should look like Figure 10-3. You can drag the data box to a more convenient location (if necessary).

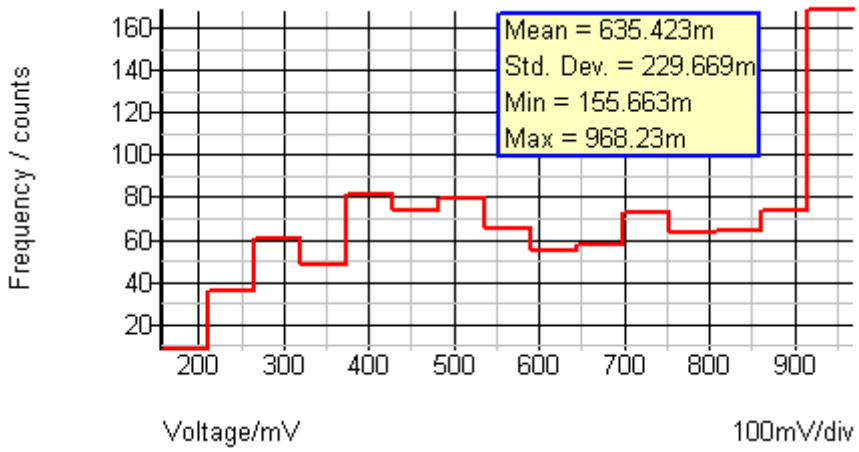

Figure 10-3. Histogram of Single Step Monte Carlo

In this example the statistical data (mean and standard deviation) are not useful as the distribution is clearly not Gaussian. But you can learn something from the plot and the minimum value is useful. If for example, you needed the circuit to have a minimum output amplitude at 768Hz of 500mV, then clearly many production units are likely to fail and higher spec components will be needed.

You may wish to repeat the above with tighter tolerances for the capacitors to get an idea of what would be needed to achieve a 500mV specification.

10.6 Statistical Distributions

In the examples we have used so far the distribution has been Gaussian (also known as Normal Distribution) which is the default distribution when a component's tolerance is specified using the menu-based method described in this training.

SIMetrix, however, offer a choice of three types of distribution:

- Gaussian (the default)
- Uniform (also known as Rectangular)
- Worst Case (WC)

10.6.1 Gaussian

If you specify a 5% tolerance, SIMetrix will plot a Gaussian distribution making 5% the value at which 3 x standard deviation - 3σ - occurs. 3σ covers 99.7% of occurrences.

Be aware that although the tolerance is specified as 5%, there is a 0.3% chance of the value lying outside that 5% range. SIMetrix will not exclude these from the sample. Manufacturers of discrete components are likely to reject devices that fall outside the specified tolerance, so this behaviour may not always be an exact representation of ‘real-life’. However, the effect on the final result is likely to be unmeasurable.

10.6.2 Uniform

Uniform distribution is a horizontal 'line'. It has an equal possibility of getting any value within each tolerance range.

10.6.3 Worst Case

Worst case distribution chooses values that are at one or other extreme and none else.

10.6.4 Using an Expression to Specify a Component Tolerance

If you need to plot different types of distribution on different components, the tolerance must be specified in a different way. Instead of specifying a tolerance parameter, the component's value must be defined using an expression that calls a distribution function. For example, to set a resistor of, say 100Ω, to use a Gaussian distribution:

1. Double click the resistor (or select and F7)
2. Instead of entering a plain resistance value, enter the expression:
   { 100 * GAUSS(0.01) }
   The function GAUSS(0.01) will return a random value with a mean of 1.0 and a 3σ Gaussian distribution of +/- 10%
3. Click OK.

Tolerances can be applied to model parameters. See Simulator Reference Manual Chapter 7.

10.6.5 Which Distribution to Choose

It is sometimes difficult to decide whether to use uniform or Gaussian. It will depend on what the component is. If it is something like a resistor where 1% is a very ordinary sort of tolerance from which manufacturers would expect a very high yield then the Gaussian distribution is probably the most appropriate. There are situations, particularly in semiconductor devices such as high-precision operational amplifiers, where manufacturers cream off the best and grade them. Each one that reaches a certain standard is graded with a particular suffix (and sold at a higher price). The distribution of these looks uniform and the distribution of the rest looks Gaussian with a hole in the middle.

10.6.6 Using a Non-Default Distribution

To change the default distribution to UNIFORM with an option setting:

1. On the Schematic Editor, open **My Documents\SIMetrix\Training\Section-10\resistor.sxsch**
2. Run the simulation
3. Go to **Monte-Carlo|Plot Histogram...** and click on the connection between I1 and R1. This enters the vector I1_N in the Expression field on the Define Performance Analysis dialog
4. Click OK and the red curve below is plotted.
5. On the Schematic Editor, press F11 and add the line **.options mc_absolute_rect** anywhere in the list.
6. Run the simulation again
7. Go to **Monte-Carlo|Plot Histogram...** and click on the connection between I1 and R1 (as above).
8. On the Define Curve dialog select the **Axis/Graph Options** tab and, in the Graph options group, click **Add to selected**.
9. Click Ok and combined Gaussian (red) and Uniform (green) graphs appear on the same grid as illustrated in Figure 10-4.

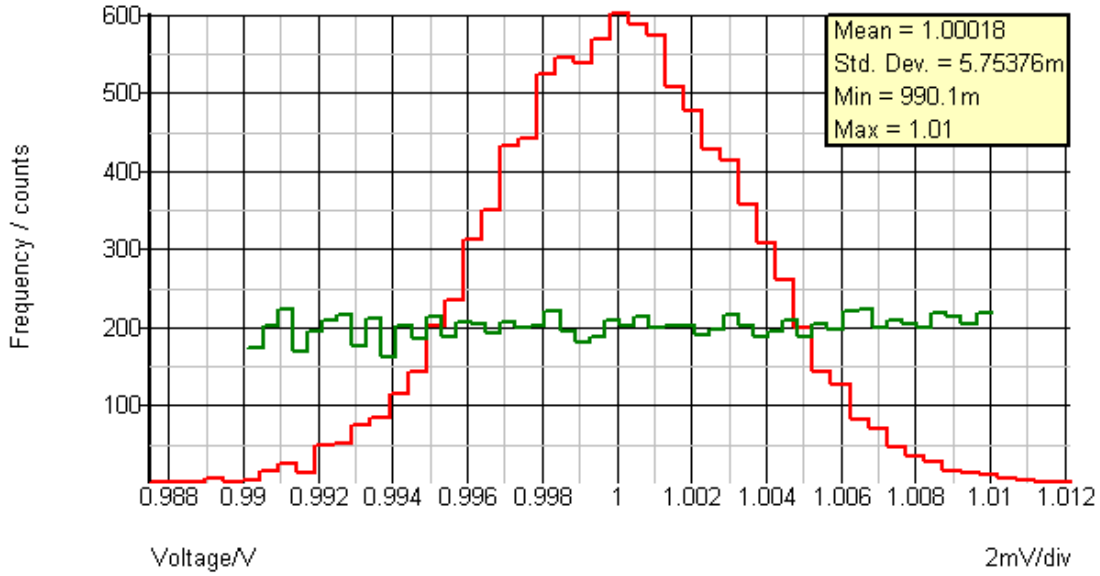

Figure 10-4. Gaussian and Uniform Distributions on the same Grid

The Uniform (green) distribution might bear some explanation. Notice that the green curve (uniform or rectangular distribution) is a relatively flat line bounded by the limits 0.99 and 1.01 representing the ±1% tolerance of the resistor. The red curve extends beyond ±1% because the Gaussian tolerance specification defines the 3σ points but is not actually bounded.

There are two other mc options, **MATCH RECT** and **LOGFILE** (see the Simulator Reference Manual, chapter 6, Command Reference, **.options** statement.

You can also use the **WC()** function (worst case) which returns only values at the tolerance extremes (see the Simulator Reference Manual, chapter 7, Monte Carlo Analysis).

10.7 Matched Components

It is possible to specify **matching characteristics**, for example in a resistor network, where the matching tolerance is usually much tighter than the absolute tolerance.

10.7.1 Matching Components and Tolerances

1. In a new Schematic Editor, build the circuit shown as Figure 10-5.
   Note that R1 is 100Ω, R2 and R3 are 1kΩ and that all resistors have a 1% tolerance.
   So that your circuit is consistent with the instructions that follow, ensure that the component references (R1, R2 and R3) are the same as shown in the picture below. You can use F8 to change the reference if needed.

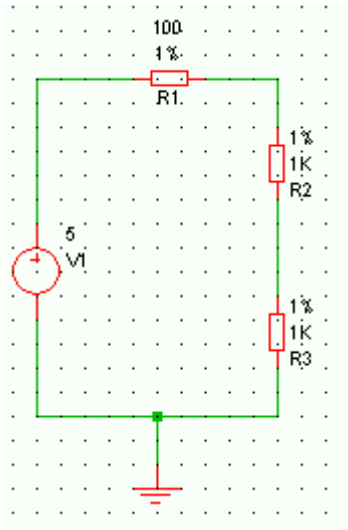

Figure 10-5. Resistor-Divider Circuit

2. Select R2 and R3.
3. Go to **Monte Carlo|Match Selected Components** and enter 'net1' as the lot name in the dialog which opens. Matching tolerance will not work without a lot name. The name is arbitrary but must apply to all the components in that particular lot.
4. Ensure that R2 and R3 are still selected, go to **Monte Carlo|Set Match Tolerances** and enter '0.1%' in the dialog which opens.
5. On the Schematic Editor go to **Choose Analysis**, check the DC Sweep mode box and open the DC tab.
6. In the Sweep Parameters group, click the Define button:
   - Check the Monte Carlo radio button in the Sweep Mode group.
   - Set the Number of points to 10000.
   - Click OK.
7. Run the simulation.
8. On the Schematic Editor go to **Monte Carlo|Plot histogram...** and click on the connection between R2 and R3.
9. Click OK and this plots the histogram for the matched components.
10. Select R2 and R3 if they aren't already..
11. On the Schematic Editor go to **Monte-Carlo|Unmatch selected components.**
12. Run the simulation again.
13. On the Schematic Editor go to **Monte Carlo|Plot histogram...** and click on the connection between R2 and R3.
14. On the Define Performance Analysis dialog, select the **Axis/Graph Options** tab click on Add to selected.
15. Click OK and this plots the unmatched histogram on the same grid (with a legend box) such as the Figure 10-6.

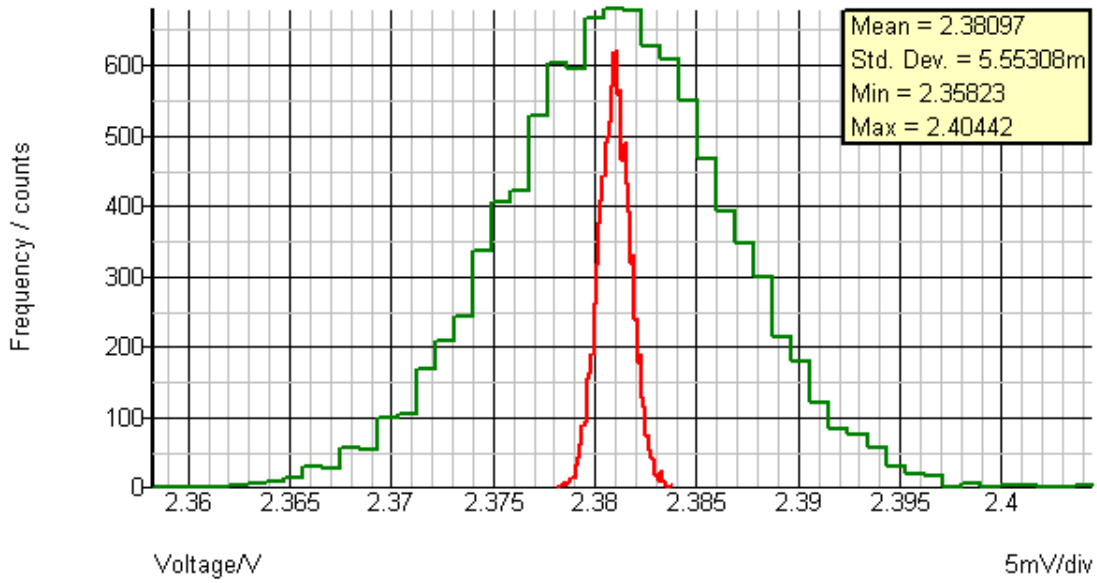

Figure 10-6. Matched and Unmatched Resistors Histogram

In the above diagram the red curve is the matched distribution and the green curve is the unmatched distribution.

11 Hierarchical Schematics, Busses and Digital

A Hierarchical Schematic is one which has several layers. In a hierarchy, individual schematics may be encapsulated into a symbol which then becomes known as a 'component'. Components may themselves be placed on schematics which may also be encapsulated into further components. The end result is a hierarchy of schematics with a single top level, or root.

The top level would include at least one component but could have any number. As you descend into one of them you will find a new schematic which could consist of more components. The underlying schematic of a component is known as a 'child' schematic and would include one or more 'module ports' which represent connections to the symbol. Conversely the schematic that contains the component is known as a ‘parent’ schematic.

To demonstrate the hierarchical system, we will enter a design for an 8-channel data acquisition system (based on a real but much bigger system) which comprises:

- 8 channel reed relay multiplexer
- Low noise amplifier (LNA)
- ADC
- Digital control

While the main focus of the module is working with a Hierarchical Schematic, several other areas are introduced, including:

- Use of a script
- Bus connections
- Digital simulation
- Analog to Digital Conversion (SIMetrix has one as a built-in component)

We will be building a hierarchical design from the bottom up. To save time a few shortcuts have been developed. We will now build the first schematic and use it as a 'component' in a hierarchy. The word 'component' is the specific word for a block which sits within a hierarchy.

11.1 Stages in Building the Hierarchical Schematic

11.1.1 Run the Script

1. In the Schematic Editor, open an empty schematic sheet
2. In the Command Shell, go to **File|Change Directory** and change directory to **My Documents\SIMetrix\Training\Section-11\build.sxscr**. (The first place that SIMetrix will look for a script is in the current directory.)
3. In the Command Shell, on the command line, type:

   build

   then press Enter. Watch most of the circuit being built as a result of the execution of a script. (This script has been slowed down very considerably to make it visible.) It will look like Figure 11-1. If nothing happens and you see the message 'Error : Unknown command "build"' then it is likely that you are not in the correct directory. Go back to the previous step and check. Make sure that you have an empty schematic open.

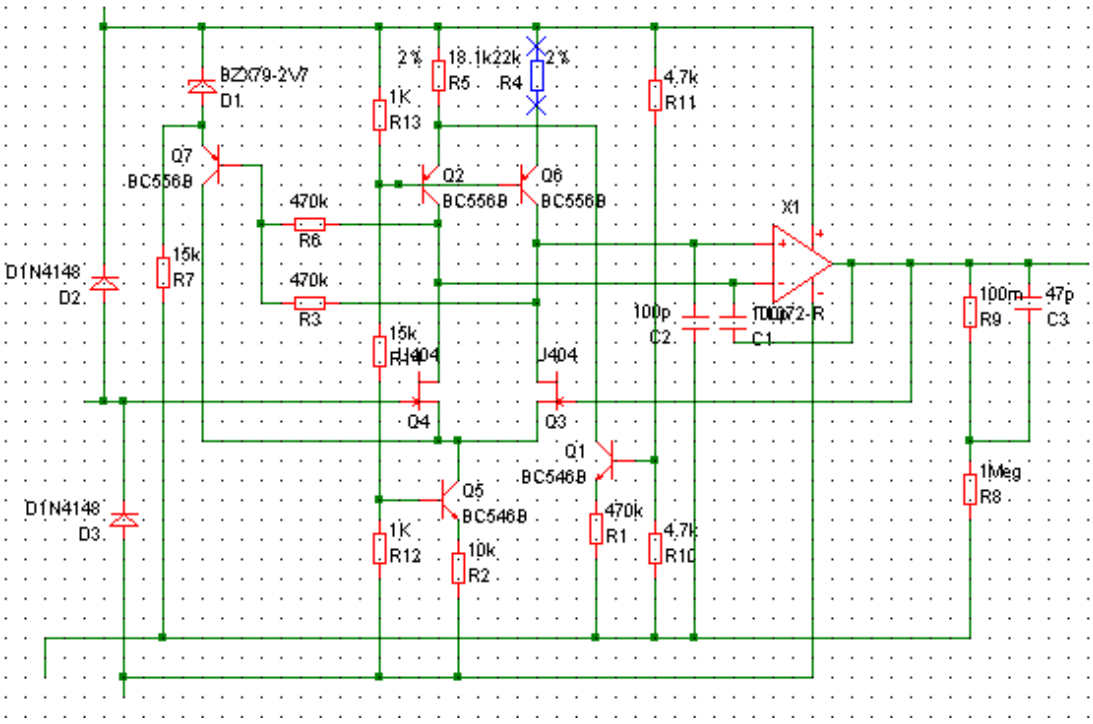

Figure 11-1. The script-built part of jfet amp

11.2 Add Module Ports

11.2.1 Essential Background

Each input and output of the jfet amp must have Module Port attached to it which the automatic symbol creator will implement as a component pin. Each module port must correspond to a single pin on the symbol. There are two aspects of them that must be correct:

- Their names must match the pins on the symbol (box) that we will create later to host the component. The (display) names are VSP, INP, GND, VSN and VOUT.
- It is recommended that they are facing in the correct direction because the automatic symbol creator will position the symbol pins on the basis of the module port's orientation (for example, a module port with its pin facing left will appear on the right hand side of the symbol. You'll see it in a minute.)

11.2.2 Practical Placing and Orientating

1. On the Schematic Editor, go to **Hierarchy|Place Module Port** (or press H) and, when you move the cursor it changes to a shadow of a module port with its pin facing left. Place it on the output (the default name of VOUT is correct for this position).
2. Repeat the last step for the remaining module ports (VSP, VSN, INP, GND) as shown in Figure 11-3. The names may be assigned by double-clicking or select then F7. Place and orientate them correctly (to rotate a selected item, press F5).

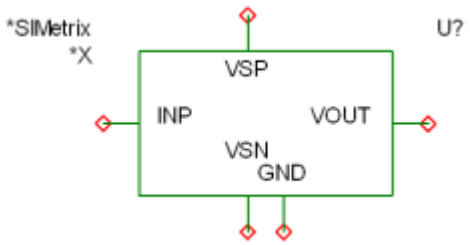

Figure 11-2. The jfet amp symbol

3. Ensure that your schematic is the same as Figure 11-3.

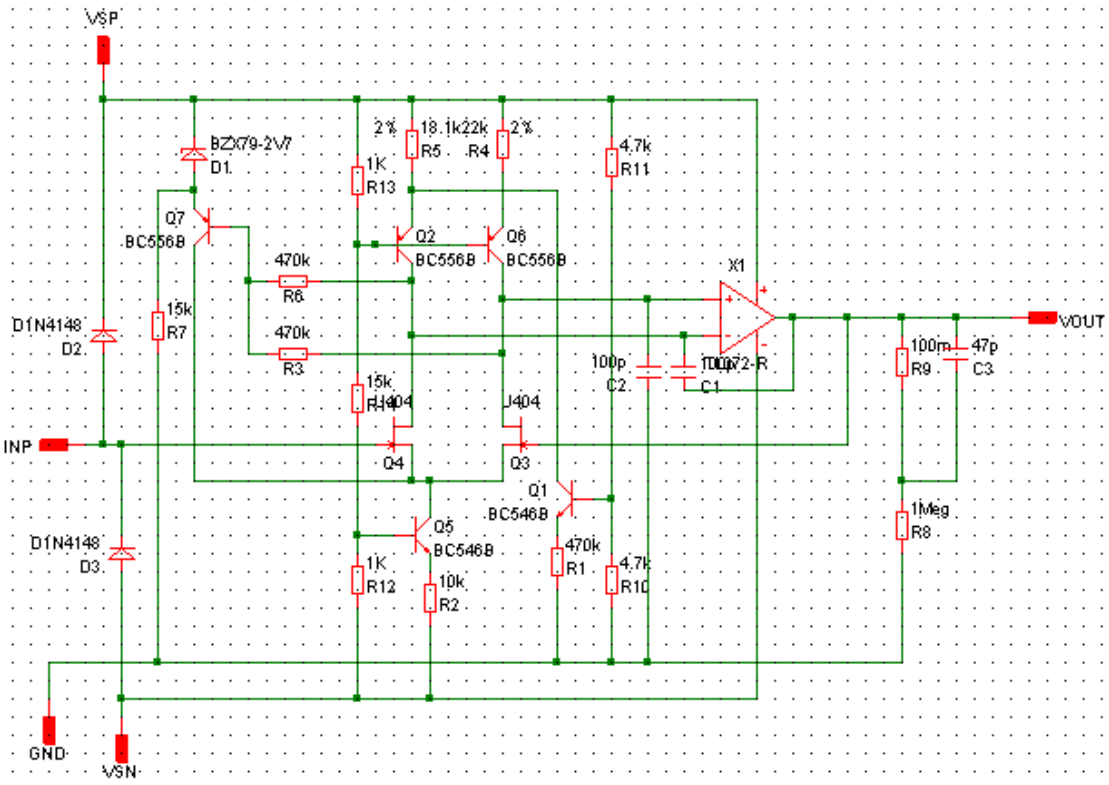

Figure 11-3. The jfet amp with Module Ports

11.2.3 Defining a Symbol for the Component

1. On the Schematic Editor, go to **File|Save As...** . (The default location in which it will be saved is the current directory.)
2. In **Save as type**: select 'Components'
3. Enter file name 'jfet_amp'

Note: When we were manually associating a symbol with a model it was important to ensure that pin order was correct. In this process, pin order is not important. What is important when working with hierarchical components is that Module Port names exactly match the Symbol Pin Names. An error message appears if there is a mismatch between names.

Note: The Save as type drop down list offers the choice of Schematic Files or Components. Exactly the same material in the same format is saved in the file whichever selection is chosen. The only difference is the filename extension.

11.2.4 Auto-create and Place a Symbol

1. On the Schematic Editor, go to **Hierarchy|Open/Create Symbol for Schematic**. This will automatically create a symbol for this schematic. The symbol will work fully as it stands and does not require any further editing. The symbol that was auto-created for the jfet amp is shown as Figure 11-4. Note: If there is already a symbol connected with that schematic, SIMetrix open that symbol rather than create a new one.
2. Save the symbol using the default settings. By default it will save to the same filename as the schematic. Don't be alarmed, you are not about to overwrite the schematic. The symbol is stored in the same file as the schematic so that the schematic and associated symbol are always inextricably linked. Also, if you save a schematic to a file that already has already been used to save a symbol, the symbol will not be overwritten.
3. Close the symbol.

Note: Component symbols are referenced by their file system path name. This can be either a full path or a path relative to the location of the schematic. Relative paths are usually the best choice.

4. On the Command Shell, go to **File|New Schematic Window**.
5. Save the unnamed sheet to **test_amp.sxsch**. (If you omit this step, placing by relative path with not work.)
6. On the Schematic Editor, go to **Hierarchy|Place Component (Relative Path)...**
7. Locate **jfet_amp.sxcmp** and place the symbol in the usual manner.

You can also place a component by dragging and dropping from Windows Explorer into the Schematic Editor. This facility uses full path or (if the control key is held down) relative path.

11.3 Complete and Run the JFET amplifier

1. Complete a test rig around the jfet amplifier on **test_amp.sxsch** as shown in Figure 11-4. V3 is a single pulse source defined using the Universal source from menu **Place|Voltage Sources (Universal Source)**. Specification is:

   Initial 0V
   Pulse 10mV
   Delay 1ms
   Rise and fall time 10us
   Width 2ms

   See Figure 11-5
2. Go to Choose Analysis. Set Analysis Mode to Transient and Stop Time to 10mS.
3. Run the simulation and obtain a curve such as Figure 11-6. You can see that it amplifies a single 2ms pulse with 5v offset cleanly.

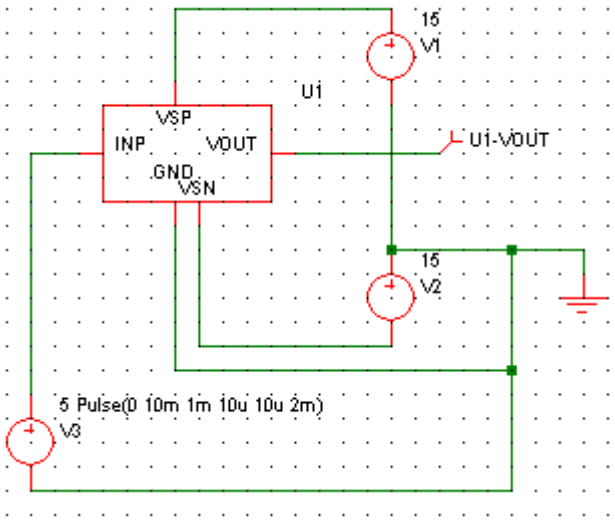

Figure 11-4. Test Rig for the Low Noise Amplifier

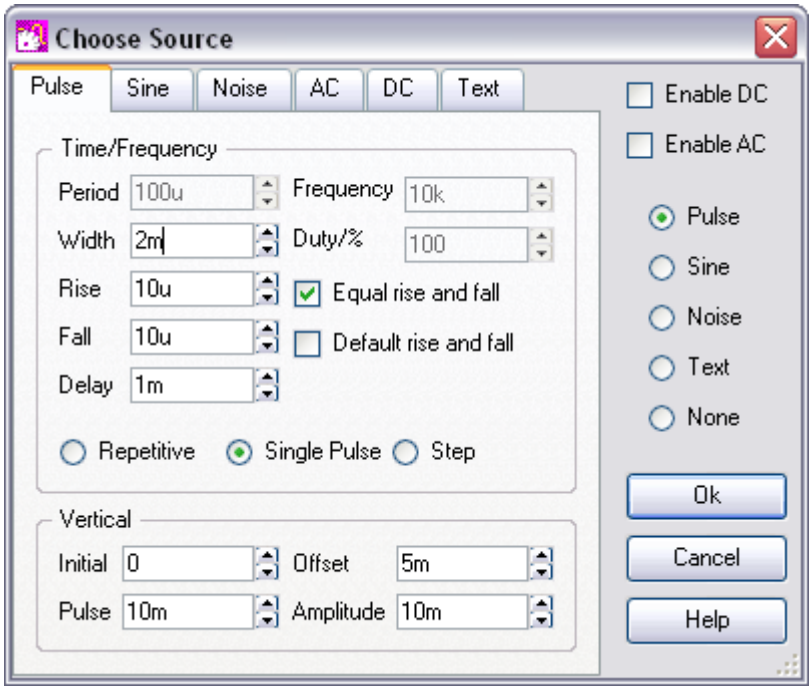

Figure 11-5. The Choose Source (Single Pulse) Dialog

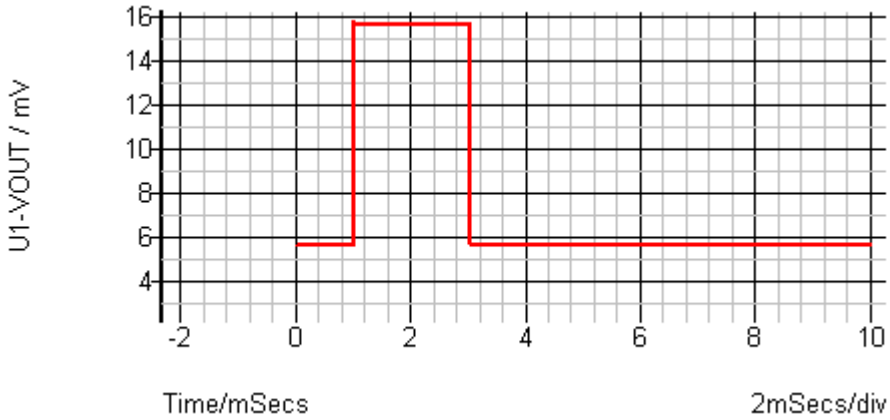

Figure 11-6. Simulation curve for LNA single pulse

11.4 Navigating Hierarchies

1. Select U1 on the test_amp schematic.
2. Right click for the context menu and select **Descend Hierarchy** (or press Ctl+E). Notice that the fourth block in the status bar is indicating 'U1' which is the parent from which you descended into this child.
3. On the Schematic Editor, go to **Hierarchy|Ascend** (or press Ctl+U).

There are several important points to note about hierarchies:

- The fourth block on the status bar always indicates the genealogy of the schematic that is being displayed.
- A child can have only one parent but a parent can have many children.
- When you have several schematic tabs open in the Schematic Editor, some of them may be parents and children of each other. Be aware that selecting the tabs to navigate to a schematic in the hierarchy may not do what you expect. The only way that SIMetrix can keep track of where you are in a genealogy is by following your use of the **Descend**

Hierarchy and **Hierarchy|Ascend** functions, and these are what you should always use to navigate a hierarchy.

11.5 The Multiplexer and Bus Rippers

On the Schematic Editor, open **My Documents\SIMetrix\Training\Section-11\mux.sxcmp**. The design isn't quite finished and we will complete it.

11.5.1 Bus Rippers

1. On the Schematic Editor, go to **Place|Connectors|Bus Ripper...** . This calls up the **Define Bus Ripper** dialog.
2. Enter **IN1** as the Bus Name, leave the Start index as 0, enter End index as 7 (it is an 8-bit bus with wires numbered 0-7) and leave Style as the default. If a Bus Rippers is named then other connections to the same bus must have the same name.
3. Click OK. When you next move the cursor a shadow appears on it.
4. Place the symbol but, for the moment, not up against the wiring.
5. Select the Bus Ripper and click the 'Mirror' icon  (or press F6)
6. Move the Bus Ripper to connect with the IN1 wiring at the top left of the schematic. Don't worry about the 8 spurious blue diagonal lines that appear while the Bus Ripper is selected; they disappear when it is not selected.
7. Connect up the missing wire on line 3.
8. Place another Bus Ripper named **IN2** on the IN2 wiring.
9. Place another Bus Ripper named Control on the CP wiring but this time, instead of Mirror, you will need to rotate it three times and flip it.
10. Use a wiring method to add bus connections from IN1, IN2 and Control Bus Rippers out to the left as shown in Figure 11-7. Notice that SIMetrix knows that the connection is to a bus and uses a heavier line to signify that.

11.5.2 Module Bus Ports

1. Go to **Hierarchy|Place Module Bus Port** on to the bus wiring for IN1, IN2 and Control. In each case:
   - Double click (or select+F7)
   - Edit the Size to 8
   - Edit the name to match the bus name.
   - Click OK
   - Rotate twice.
   - Move into position.

11.5.3 The Complete Schematic

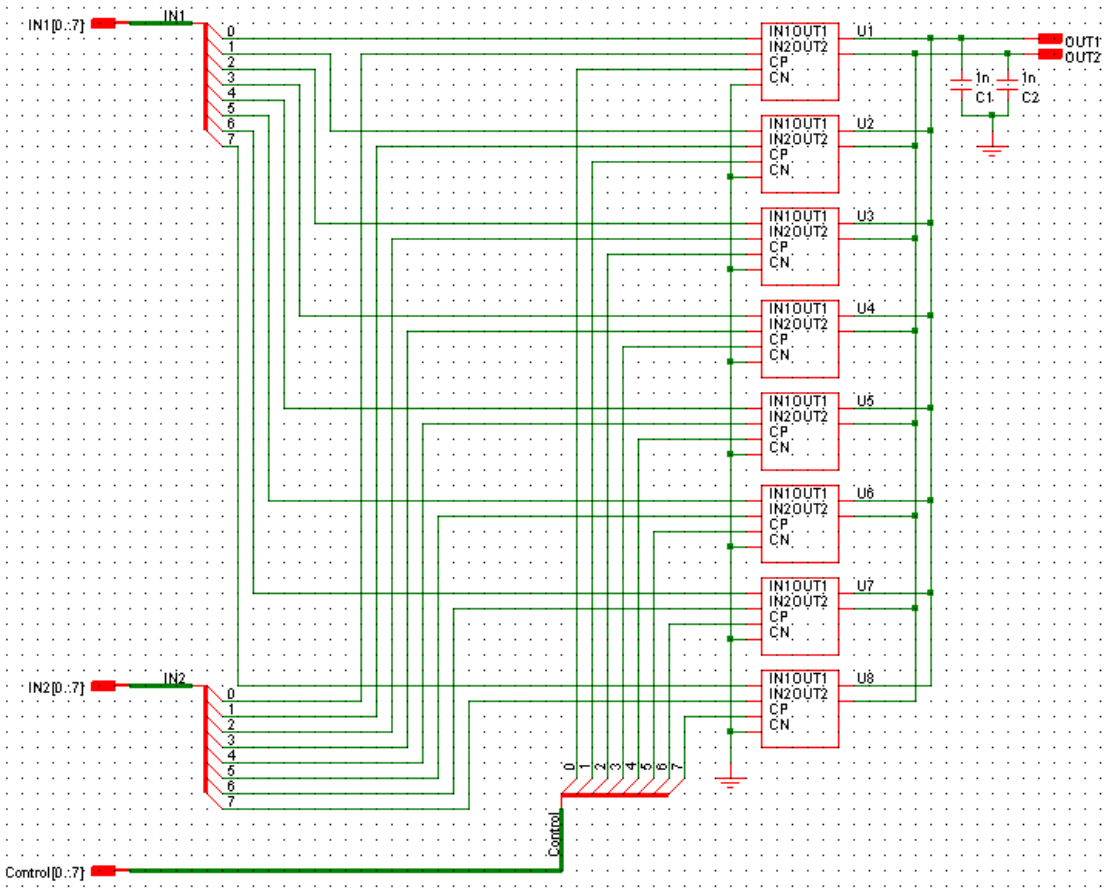

Figure 11-7. The Completed MUX Schematic

11.5.4 The MUX Symbol

This is the same process that we carried out in Auto-create and Place a Symbol on page 107.

1. On the Schematic Editor, go to **Hierarchy|Open/Create Symbol for Schematic**. A (by now fairly predictable) symbol appears, Figure 11-8, in a new window of the Symbol Editor.
2. On the Symbol Editor go to **File|Save**. Ensure that the pathname is correct and that you are saving as a Component.

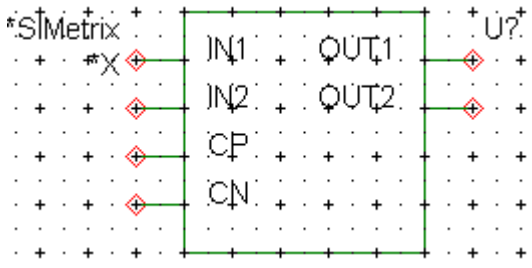

Figure 11-8. The MUX Symbol

11.6 The Dual Relay Model

SIMetrix does not have a model for the relay so we will make one up from a built-in device called a 'delayed switch'.

1. Open **My Documents\SIMetrix\Training\Section-11\dual-relay.sxcmp** in the Schematic Editor.
2. Go to **Place|Analog Functions|Delayed Switch**.

3. Copy and paste it to the position shown for S1 in Figure 11-9, ensuring that the '+' sign is on the left. You will need to rotate it to get the correct orientation – use the F5 key.
4. Place the second to the position shown for S2 in Figure 11-9. Again, make sure it is connected correctly with the '+' symbol on the left. You will need to mirror the symbol to get this right. Use F6 to mirror.
5. Select both so that you can edit them both at the same time. Use the parameters shown in Figure 11-9.

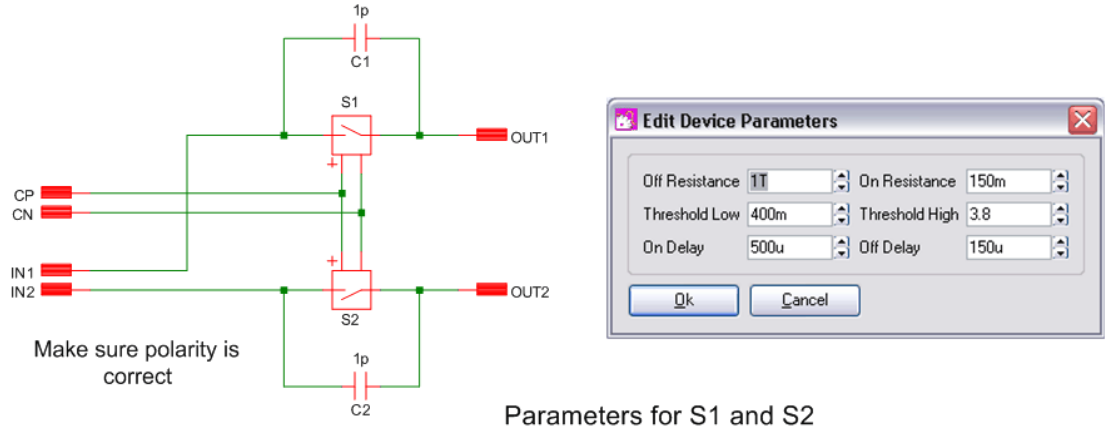

Figure 11-9. Definition of the Delayed Switch

6. Save the schematic.

11.7 Complete the Top Level

1. Open **My Documents\SIMetrix\Training\Section-11\top.sxsch** in the Schematic Editor.
2. Go to **Hierarchy|Place Component (Relative Path)** and add the MUX **mux.sxcmp** and the JFET LNA **jfet_amp.sxcmp**.
3. Wire up the remaining bus connections. The final circuit is shown as Figure 11-10.

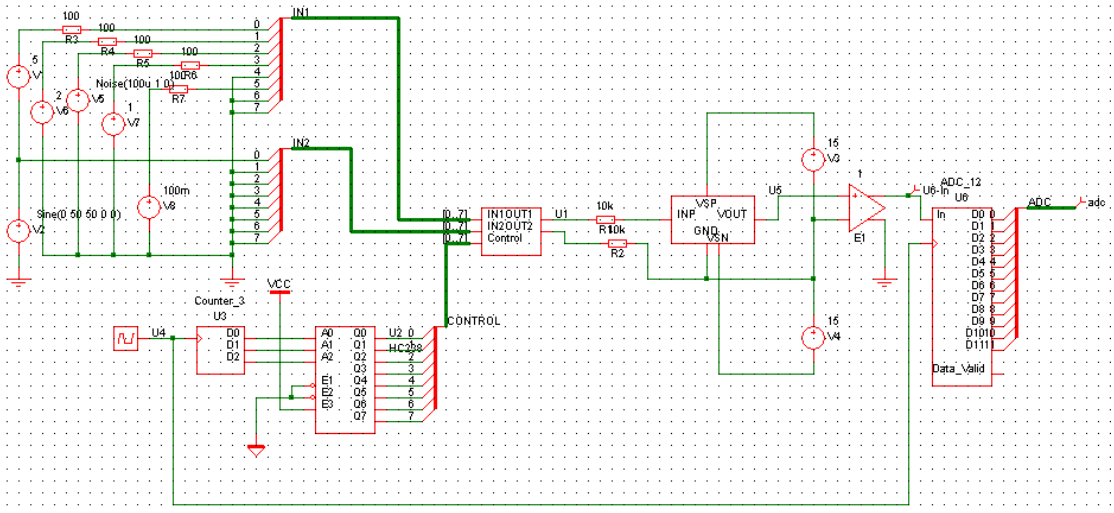

Figure 11-10. The Final Circuit

11.7.1 Brief Description of the System

The inputs to the system are a set of sources to represent, in reality, a great many sensors scattered all around the plant/site/vehicle, whatever. The control consists of Time Division Multiplexing (TDM) so that the one signal reaching the ADC to be measured cycles around all the inputs.

Open the ADC Converter dialog and notice that in the Timings group, a value of 1u might suggest that only 1M conversion per second might be possible whereas the quoted Maximum conversion rate is

2Meg. This isn't a mistake; a pipelined architecture, for example, would typically have this type of characteristic.

The clock drives the counter. It is a 1 of 8 so any one of 8 lines will be active at any one time. These are going to be controlling the relays. The real thing would have had a microprocessor controlling it but in the simulation we have an input signal on all eight channels which gets sampled repeatedly.

- A 50mS transient has already been set up in **top.sxsch**.
- A fixed voltage probe has been added to the input of the ADC.
- A bus probe has been added to the ADC output.

11.7.2 Run Simulation

- Run the Simulation.
- The output of the TDM sampling with ADC amplitudes is shown as Figure 11-11.

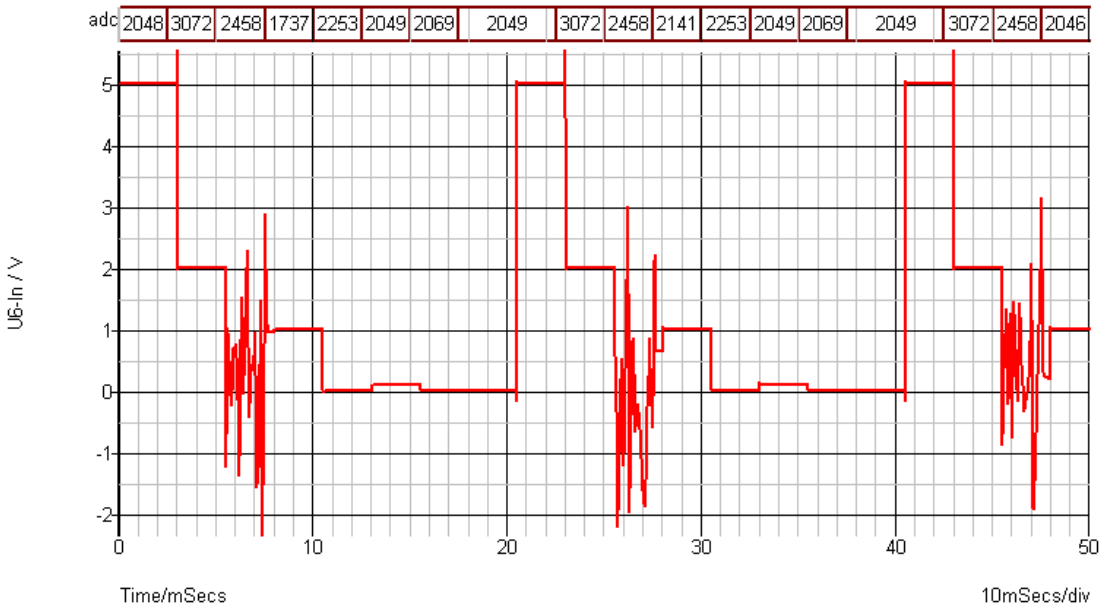

Figure 11-11. The TDM Output Curve with the ADC Amplitudes

The figures along the top are the ADC measurements. ADC does not makes its measurement until towards the end of the timeslot to give it time to stabilise.

A 12-bit converter will give 4096 (2^12) values, and with the 5v offset described earlier, 0v will correspond to a reading of 2048. The cycle completes every 20mS. What we are seeing is the different timeslots in the 8 channel multiplexer. On channel:

- 0 is a 5v supply which is originating in V1
- 1 is a 2v supply which is originating in V6
- 2 is fed from a noise generator V5
- 3 is a 1v supply which is originating in V7
- 4 is grounded
- 5 is 100mV from V8
- 6 is grounded
- 7 is grounded

11.8 Digital Processing

SIMetrix offers a digital simulator and components. Many of the components can be found under **Place|Digital Generic** and they comprise:

- Logic Gates

- Counter
- Shift Register
- Bus Register
- ADC
- DAC

All digital inputs and outputs assume 5v logic levels when interfacing with analog parts. These are all completely standard digital design elements and are not specific to SIMetrix. They will not be explained further in this training.

11.8.1 Digital and Analog Simulators

The digital devices are simulated using an event driven digital simulator that is closely integrated with the analog simulator. One thing to note is that the digital simulator only understands logic levels and doesn't know anything about voltages and currents. This leads to some unusual behaviour that we will demonstrate in the next section.

11.8.2 Digital Signals

1. On the Schematic Editor, probe the D0 output of U3 using the fixed menu **Probe|Voltage...** (don't use the context menu **Probe Voltage...**). This gives a curve such as Figure 11-12. The difference between fixed menu **Probe|Voltage...** and the context (popup) menu **Probe Voltage...** is that the latter is more intelligent. Menu **Probe|Voltage...** always plots on analog scales whereas the context (popup) menu **Probe Voltage...** recognises that a digital signal is present and plots it on a digital axis. (Fixed menu **Probe|Voltage Digital...** always gives a digital graph regardless.)

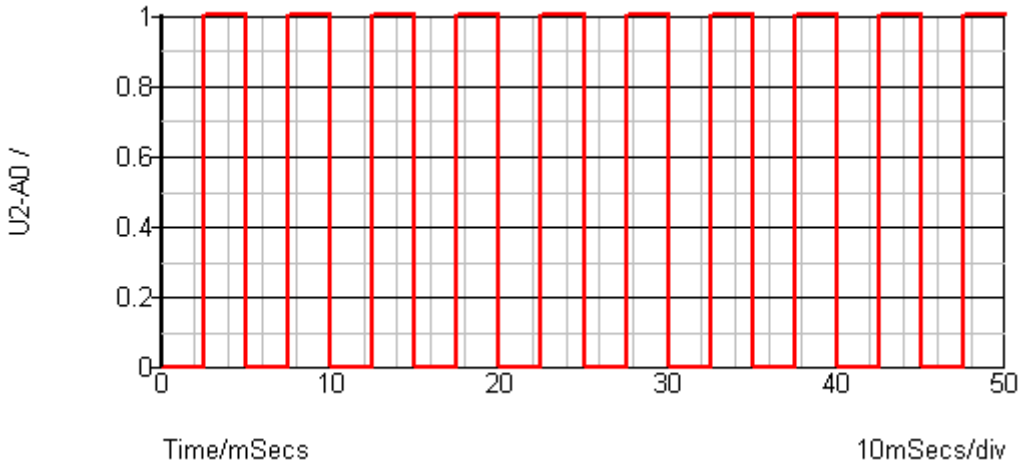

Figure 11-12. The Digital Output of the Counter

The point to note about this curve is that, despite having been plotted on analog scales, it represents a digital signal. The peaks represent logic 1 not 1v.

2. Close the graph and probe U2 Q2 pin using the fixed menu **Probe|Voltage...** . This gives a curve such as Figure 11-13.

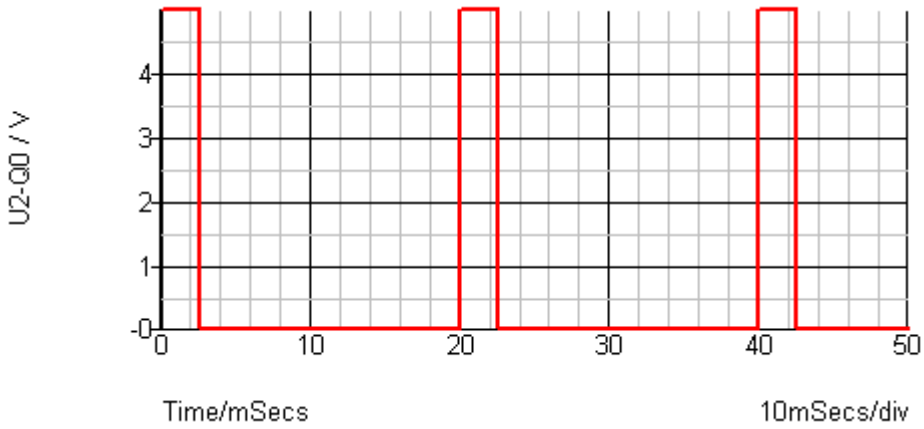

Figure 11-13. The *Analog* Output of Q2

This time you will note that the signal is showing voltage levels of 5V. Although U2-Q0 is digital, it is connected to an analog component inside U1 (the mux). SIMetrix inserts a special component called a bridge to convert from digital to analog domain. The bridge is connected implicitly by the simulator and does not show on the schematic. The Plot is of the analog output of this bridge. Internally there is also a digital input to the bridge which would be a true digital signal with levels 0 and 1, but this signal does not get plotted in response to the cross-probing action.

This interconnection between analog and digital simulators is an important concept. For more information please see the Simulator Reference Manual, Chapter 9, heading Analog to Digital Interfaces.

11.8.3 Digital Bus Probes

The ADC output has a fixed bus probe attached. Bus probes plot the connective signal on a bus but, to work, all the signals must be digital.

Bus probe parameters can be edited in the usual way. Bus probes can produce plots in one of 4 formats:

- Decimal values
- Hexadecimal values
- Binary values
- Analog waveform

11.8.4 Random Probing Hierarchies

Hierarchies may be random probed to any depth, that is, any child schematic may be probed. The genealogy (hierarchical path) is shown in the Waveform Viewer Legend Box.

Here is an example to demonstrate how we can probe the same point on the same schematic and get different results depending on how we navigated to the schematic.

1. In the Schematic Editor with **My Documents\SIMetrix\Training\Section-11\top.sxsch** loaded, go to **Probe|Delete All Fixed Probes**.
2. Run the simulation.
3. Select U1and Descend into it. This takes you down one level into **mux.sxcmp**.
4. On **mux.sxcmp** select U1 and Descend into it. This takes you down one further level into the U1 instance of **dual-relay.sxcmp** .
5. Go to **Probe|Voltage...** and random probe IN1. This plots the red sine wave shown as Figure 11-14. Notice that the Legend Box shows that the genealogy of the red curve is **U1.U1.IN1**.
6. On the Schematic Editor go to **Hierarchy|Ascend** which takes you back to **mux.sxcmp**.
7. On **mux.sxcmp** select U2 and Descend into it. This takes you down one further level into the U2 instance of **dual-relay.sxcmp** .

8. Go to **Probe|Voltage...** and random probe IN1 again. This plots the green pulse shown in Figure 11-14. Notice that the Legend Box shows that the genealogy of the green curve is **U1.U2.IN1**.

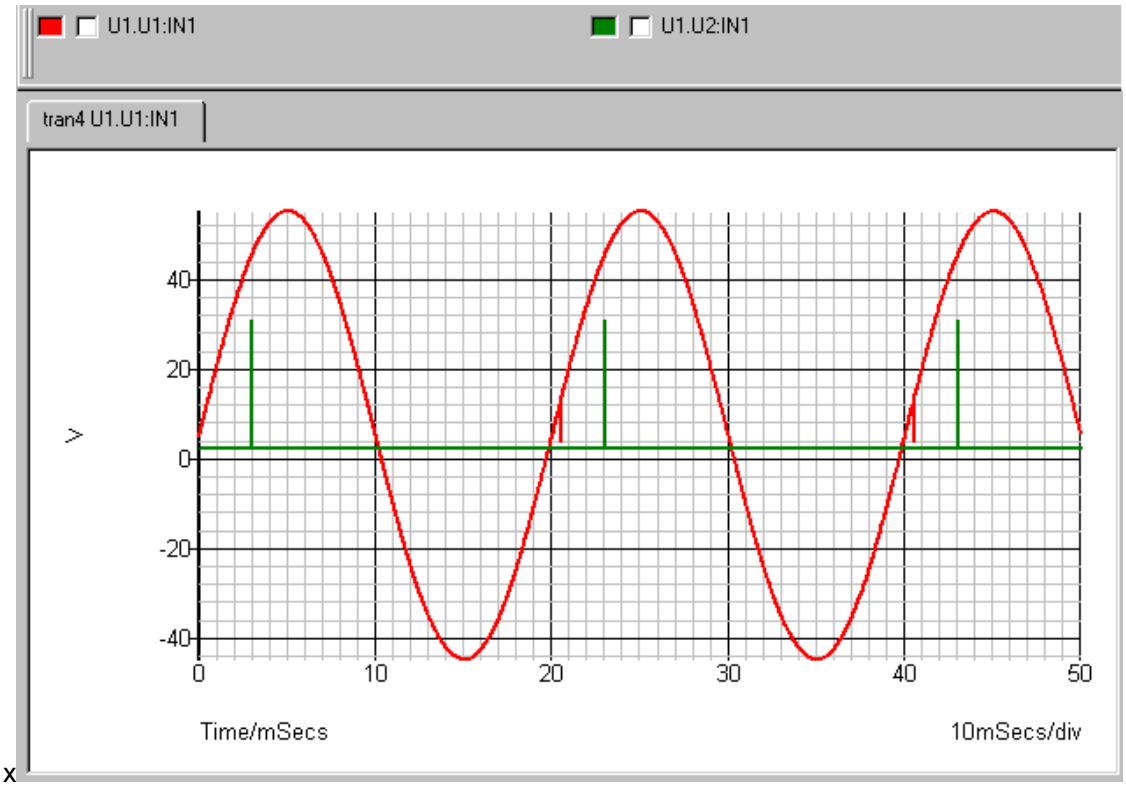

Figure 11-14. Cross Probing Hierarchy showing Legend Panel

11.8.5 Fixed Probes in Hierarchies

There is one further point to be drawn from this example and that is the difference in beahaviour of Fixed and Random probing when used in hierarchies. A fixed probe applies to all the units in the circuit, in this case eight of them. A random probe placed in one of the units, probes that unit only.

1. Add a fixed Voltage Probe to IN1 then rerun simulation. Note there are 5 curves. There would be 8, one for each relay, but three are missing as the signals are connected to ground at the top level. The graph is shown as Figure 11-15.

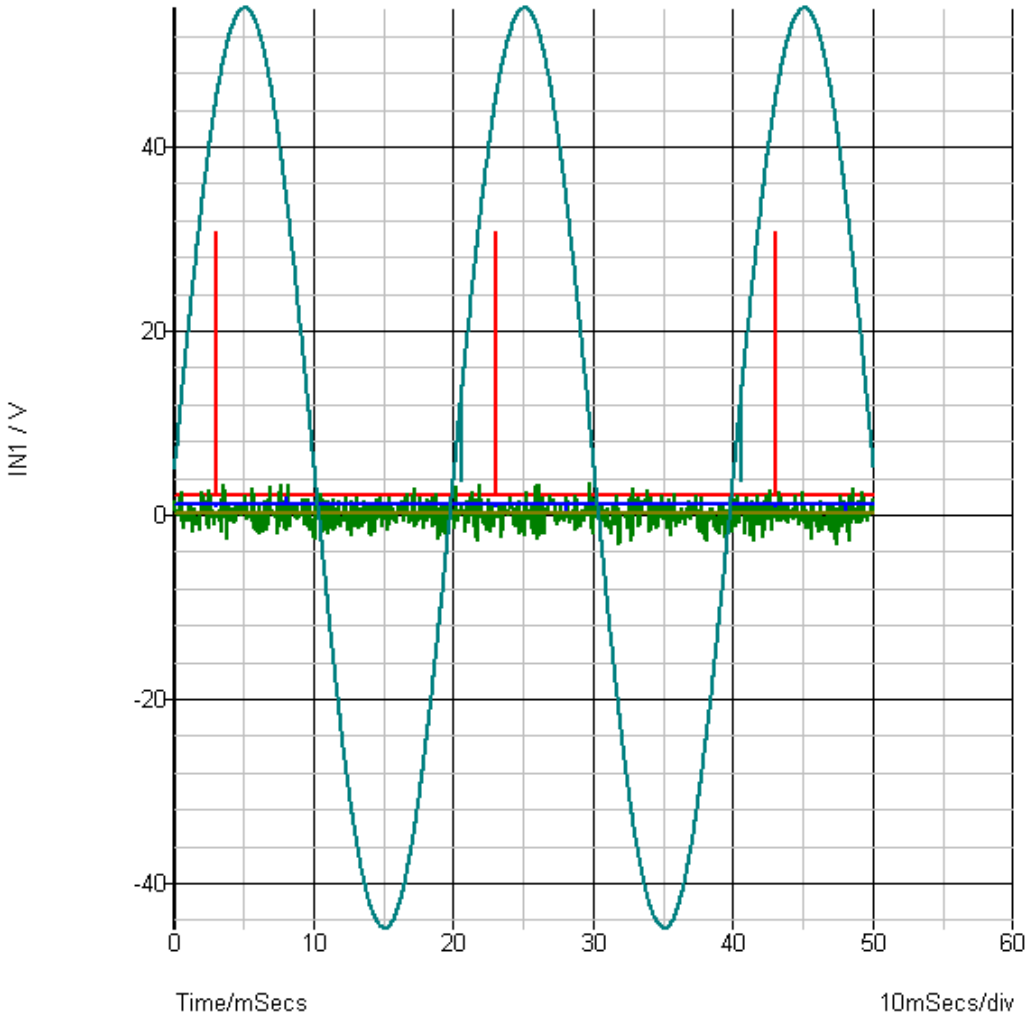

Figure 11-15. Fixed Probing Hierarchy showing 5 Curves

12 Advanced Modules, Parts and Behavioural Devices

This section covers:

- Non-linear transfer functions
- Laplace transfer functions. You can use these in a system level design to implement whatever filter characteristics you need without actually having to design the filter itself. At this stage you don't need to think about how to implement the filter itself.
- Saturable transformer.
- Soft recovery diode.

12.1 Non-linear Transfer Function

A non-linear transfer function simply produces an output, voltage or current that is mathematically related to any number of voltages and currents that you supply. In its simplest form it could be just an adder for example - an output voltage that is the sum of two input voltages.

12.1.1 Power in a Transistor

We will demonstrate an example where we will measure the power in a transistor and use this to drive a thermal network. We will use this to derive the chip temperature of the transistor.

1. Open **My Documents\SIMetrix\Training\Section-12\Mospower.sxsch**. This is a simple DC-DC converter. We are going to incorporate a power-measuring device into this circuit. We are putting measurement only on the drain and the source and assuming that power in the gate is insignificant. The total power is the sum of the IxV products for each pin.
2. On the Schematic Editor, go to **Place|Analog Behavioural|Non-linear Transfer Function...** . Unusually in SIMetrix this opens the Define Arbitrary Source dialog before you place anything. SIMetrix needs to know certain things before it can create an appropriate symbol.
3. Enter the Expression 'I(V1)*(V(n1)-V(n2))'. Voltages are in the form V(n1), V(n2) etc and currents I(V1,), I(V2) etc. where (n1) means first, (n2) means second and so on. The expression is not case sensitive.
4. Enter the fields in Inputs group to indicate two voltage inputs and one current input.
5. Leave Single ended voltage connection selected under Outputs.
6. Click OK and a shadow appears at the cursor. Place it and wire up the circuit as shown in Figure 12-1.

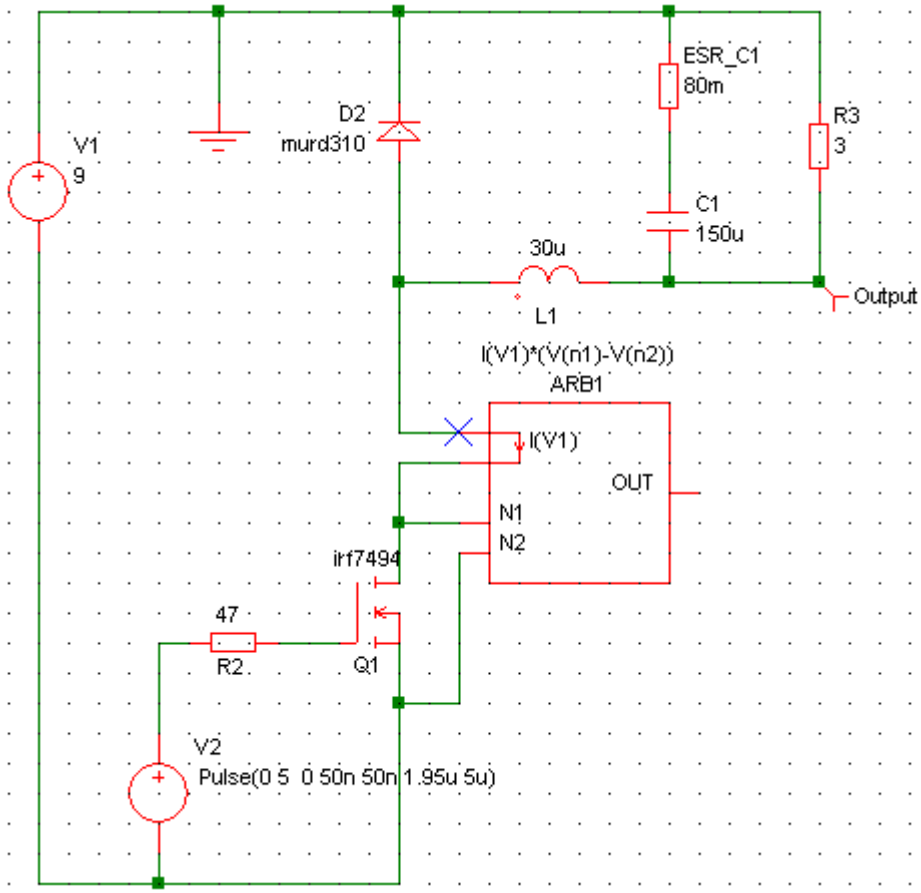

Figure 12-1. Schematic of Power Monitor example

7. Run the simulation and this gives the voltage at the fixed probe on the output as shown in Figure 12-2.

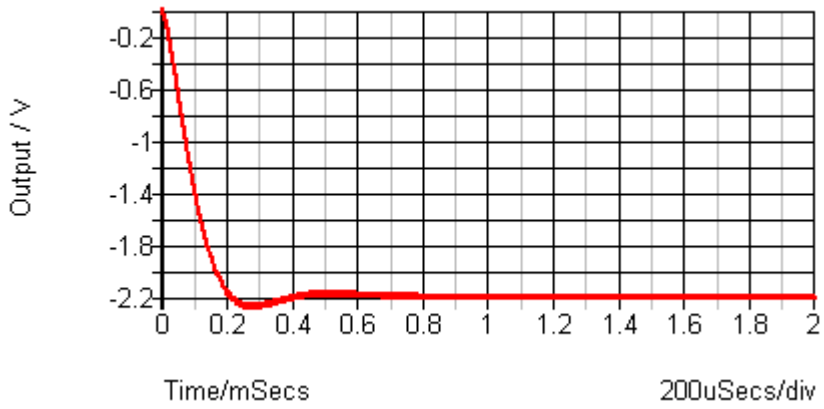

Figure 12-2. Output Voltage Curve only

8. Add a fixed voltage probe to the output of ARB1 and run the simulation again. This adds in the power output (in green) as shown in Figure 12-3 which has been shown with Waveform Viewer **Curves/Stack All Curves** selected. The green curve has been plotted as a voltage because the internal calculation is representing power as voltage (1v represents 1W).

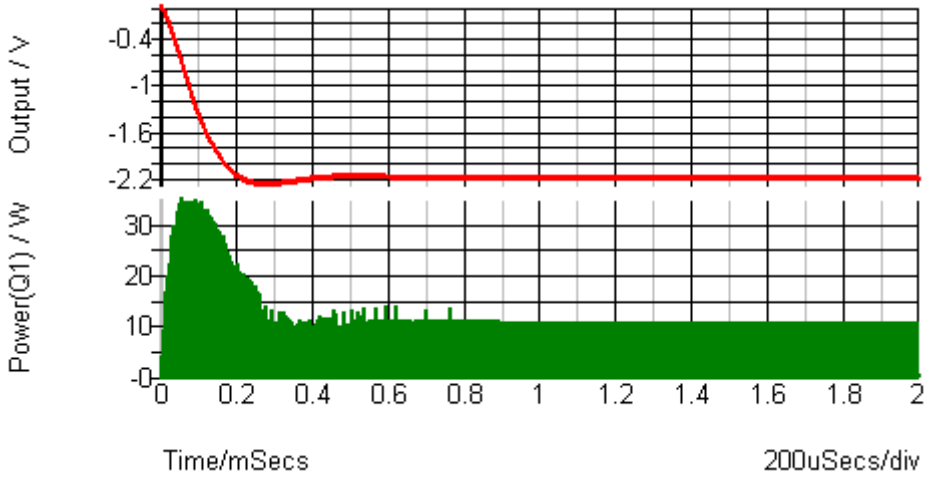

Figure 12-3. Output Voltage and Power in Q1

9. (Optionally) expand the green graph horizontally by a very considerable amount to see the power bursts at switching transitions.

12.1.2 Temperature in a Transistor

1. On the Schematic Editor go to **Place|Hierarchy|Component (Relative Path)**, select **thermal.sxcmp** and place it on existing schematic.
2. Edit ARB1 to give differential output current. Double click device to bring up editor then select Differential current in the outputs group.
3. Wire up the circuit shown as Figure 12-4.

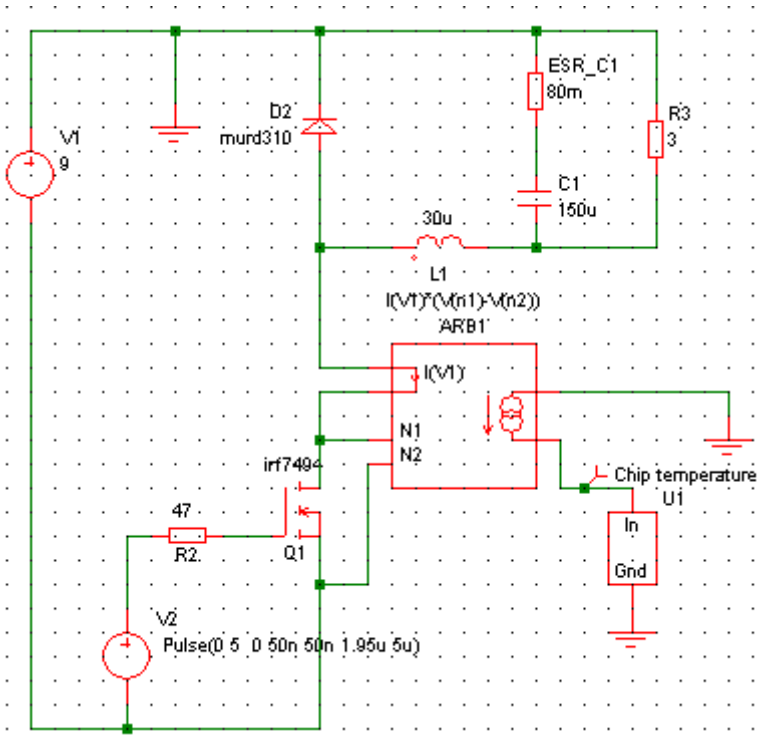

Figure 12-4. Power Monitor with Thermal Network

4. Connect a fixed probe to the Thermal Node as shown. (Labelled ‘Chip temperature’ in Figure 12-4 below)
5. Edit the probe label to Chip Temperature.

6. Close the waveform viewer then run the simulation and observe the chip temperature rise. The curves shown as Figure 12-5 (temperature is green).

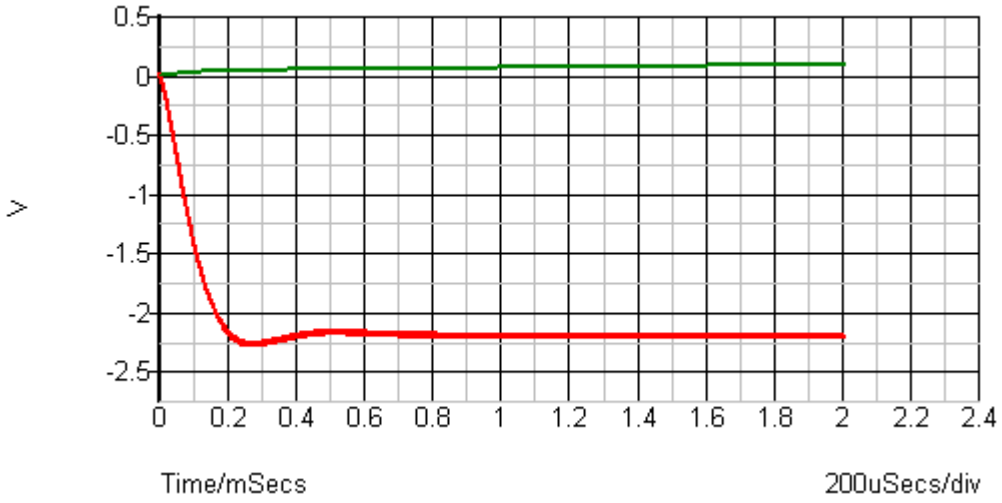

Figure 12-5. Temperature Rise Curve

You can use the chip temperature that you take here to feed back into a voltage controlled resistor which you could place in the source of Q1 to simulate the self-heating effects of the FET, particularly its on-resistance. The FET model itself will always have an on-resistance already in it so you would need to edit the model of the FET to remove the RDS(ON) component from it.

12.1.3 About Arbitrary Source Expressions

It is possible to put a wide range of expressions into a SIMetrix Expression field such as the one in the Define Arbitrary Source dialog. In the Simulator Reference Manual you should look up the coverage of the underlying device (in this case the Arbitrary Source) which is the first item in Chapter 4 of that manual.

The full statement on Expressions is in the Simulator Reference Manual in Chapter 3 under heading “Using Expressions”. This section also provides a full list of functions that can be used.

12.2 Laplace and Filter Functions

SIMetrix provides a means of defining a function in terms of a Laplace expression in the ‘s’ variable. This includes some built-in functions that define popular filter characteristics namely Butterworth, Bessel and Chebyshev.

12.2.1 Laplace Transfer Function

The Laplace Transfer Function is defined using the 'S' variable. The expression in which it appears must be a quotient of polynomials in 'S' or an expression that reduces to a quotient of polynomials.

Non-linear functions such as square root and exponential are not supported. Examples of expressions that are supported are:

- 1/(s*1u+1). We will take this low pass filter as an example.
- s/(s^2+0.1*s+0.01). This is a second order function such as would occur if both capacitor and inductor were present.
- 1/(1+1/(s+1)).

The following simple example demonstrates how you can implement a simple first order low pass filter using a Laplace block.

1. Open an empty schematic.
2. Go to **Place|Analog Behavioural|Laplace Transfer Function**. This opens the Define Laplace Transfer Function dialog.
3. In the 'Define output using S variable’ enter 1/(s*1u+1) Leave the remaining selections at their defaults. Click OK.

4. Place the shadow and build the remainder of the circuit shown as Figure 12-6.
5. Open the editor for the probe ‘Laplace block’ then select ‘Use separate grid’ in the ‘Axis type’ group

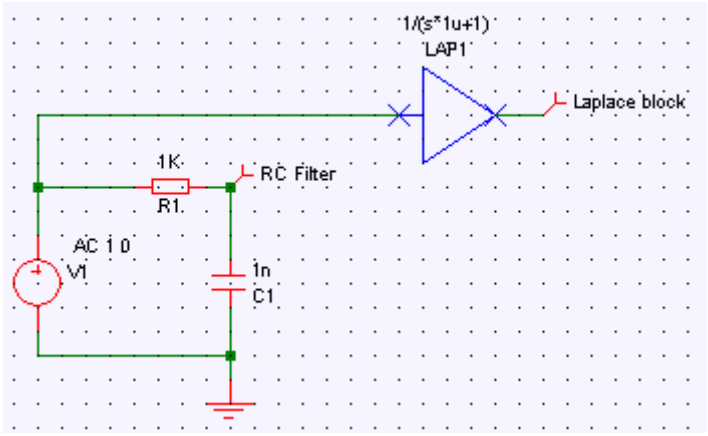

Figure 12-6. Simple Filter Using RC network and Laplace

6. Open the Choose Analysis dialog, set Analysis Mode to AC, Start frequency to 1k and Stop frequency to 10M.
7. Run the simulation. This is what you should see. If you see the two curves in the same grid, you may have missed step 5 above.

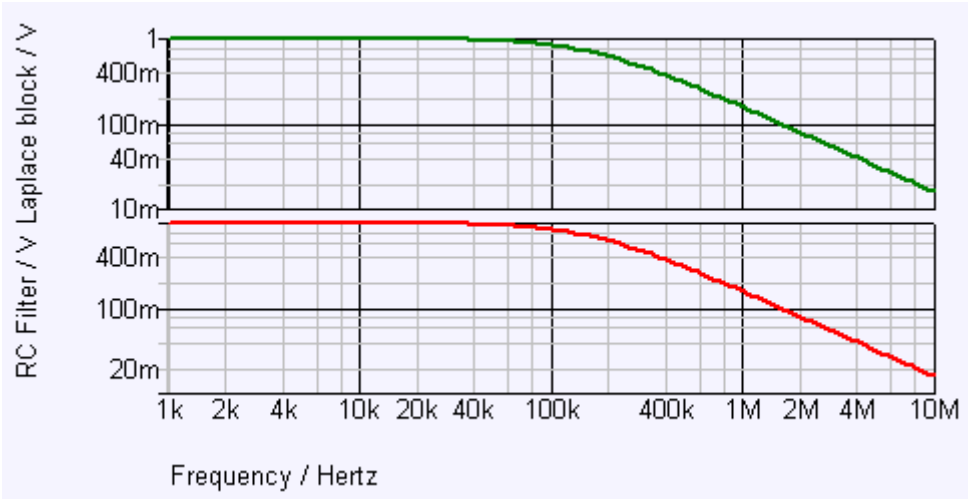

Figure 12-7. RC and Laplace Curves

Component values of 1k and 1nF give a 1uS time constant.

12.2.2 Butterworth Filter

We will now edit our Laplace block to demonstrate a Butterworth filter.

1. Double click the Laplace device LAP1
2. In ‘Define output using s variable’ replace the expression with

   ButterworthLP(3,159k)
3. Close box
4. Run simulation. You should see a steeper roll-off for the Laplace output

You may wish to experiment with the other filter characteristics available. Open the Laplace block editor then click the Help button for details of the other filter functions available.

12.3 Magnetic Components

The following magnetic components are supported:

- Ideal transformer (page 43).
- Ideal DC transformer (page 45).
- Ideal inductor (not covered in this training).
- Lossy inductor (not covered in this training).
- Saturable Transformer – see next section.

12.3.1 Saturable Transformer

Other transformer types are a subset of the saturable transformer so we will concentrate on this device.

1. Open **My Documents\SIMetrix\Training\Section-12\saturable.sxsch**.
2. Go to **Place|Magnetics|Saturable Transformer/Inductor...** . This opens the Define Saturable Transformer/Inductor dialog.
3. Ensure that the following have been set.
   - 1 primary
   - 1 secondary
   - 1:1 turns ratio
   - 50 Primary turns
   - In the Define Core group, click 'Select core type' and, from the drop-down list which is sorted strictly alphabetically, select 'EP17-3C94-E63'.
   - Click OK and place the transformer in the circuit.
4. Place a fixed voltage probe and a fixed current probe on the transformer secondary
5. Go to Choose Analysis, set Analysis mode to Transient and Stop time to 500us.
6. Run the simulation and observe the current and voltage in the transformer to confirm that saturation takes place at about 5A.

12.4 The Soft Recovery Diode

There is no satisfactory industry-standard model for a soft recovery diode. Manufacturer's models tend to be based on the standard SPICE diode model. Unfortunately this model cannot be easily adapted to show a soft recovery characteristic and many manufacturer's models give unsatisfactory results.

SIMetrix has a flexible built-in model for a soft recovery diode. However, because this model is not in widespread use, you will need to develop the model's parameters yourself from datasheet values. Here we will learn how to do this.

12.4.1 Specifying Soft-recovery Diode

To specify a soft recovery diode, we need to specify parameters shown in Figure 12-8 below. These are

- dIf/dt
- Irm
- If
- Trr

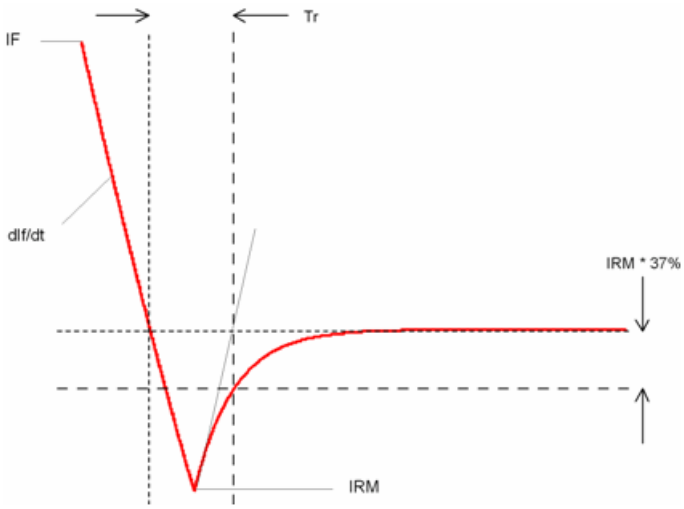

Figure 12-8 Soft Recovery Characteristic

MUR1520, RURP1520

Electrical Specifications $T_C = 25^oC$, Unless Otherwise Specified

SYMBOL	TEST CONDITION	MIN	TYP	MAX	UNITS
V_F	I_F = 15A	-	-	1.05	V
	I_F = 15A, T_C = 150°C	-	-	0.85	V
I_R	V_R = 200V	-	-	100	µA
	V_R = 200V, T_C = 150°C	-	-	500	µA
t_{rr}	I_F = 1A, dI_F/dt = 100A/µs	-	-	30	ns
	I_F = 15A, dI_F/dt = 100A/µs	-	-	35	ns
t_a	I_F = 15A, dI_F/dt = 100A/µs	-	20	-	ns
t_b	I_F = 15A, dI_F/dt = 100A/µs	-	10	-	ns
$R_{\theta JC}$		-	-	1.5	°C/W

DEFINITIONS

V_F = Instantaneous forward voltage (pw = 300µs, D = 2%).

I_R = Instantaneous reverse current.

t_{rr} = Reverse recovery time at dI_F/dt = 100A/µs (See Figure 6), summation of $t_a + t_b$.

t_a = Time to reach peak reverse current at dI_F/dt = 100A/µs (See Figure 6).

t_b = Time from peak I_{RM} to projected zero crossing of I_{RM} based on a straight line from peak I_{RM} through 25% of I_{RM} (See Figure 6).

$R_{\theta JC}$ = Thermal resistance junction to case.

pw = pulse width.

D = duty cycle.

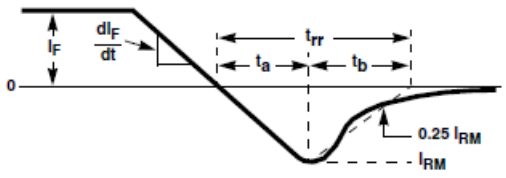

FIGURE 6. t_{rr} WAVEFORMS AND DEFINITIONS

Figure 12-9 Soft Recovery Specification

We will take the example of a MUR1520 rectifier diode. The data sheet for this device is shown in Figure 12-9. From this:

If = 15
dIf/dt = 100e6

Calculate Irm from ta and dIf/dt:
Irm = dIf/dt x ta

Irm = 100e6 x 20e-9 = 2A
Trr = 30n

Now we will use these values to create a diode model.

1. Open **My Documents\SIMetrix\Training\Section-12\soft_recovery.sxsch**
2. Select **Place|Create Model|Soft Recovery Diode…**
3. Enter values as shown in Figure 12-10
4. Click OK then place diode with the cathode point upwards as shown in Figure 12-11
5. Place a fixed current probe at the D1 anode
6. Go to **Choose Analysis**, set Analysis mode to Transient and Stop time to 500ns.
7. Click OK and run the simulation. Once the curve has been created, use the Edit Axis dialog to set the y-axis max to 1 to obtain the curve similar to that shown as Figure 12-12.

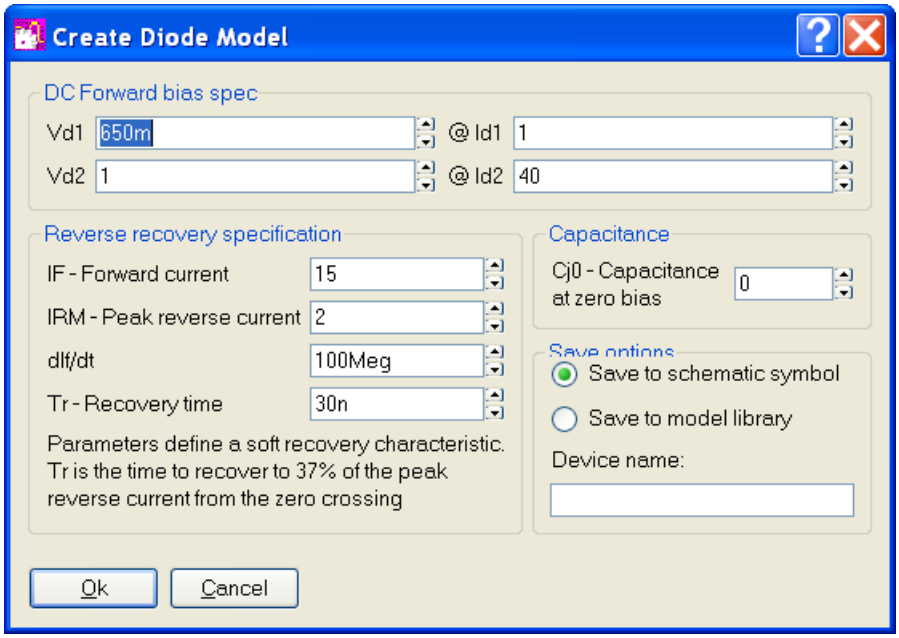

Figure 12-10 Soft Recovery Diode Parameters

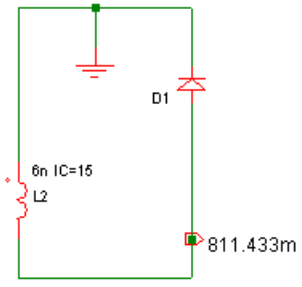

Figure 12-11 Soft Recovery Diode Test Schematic

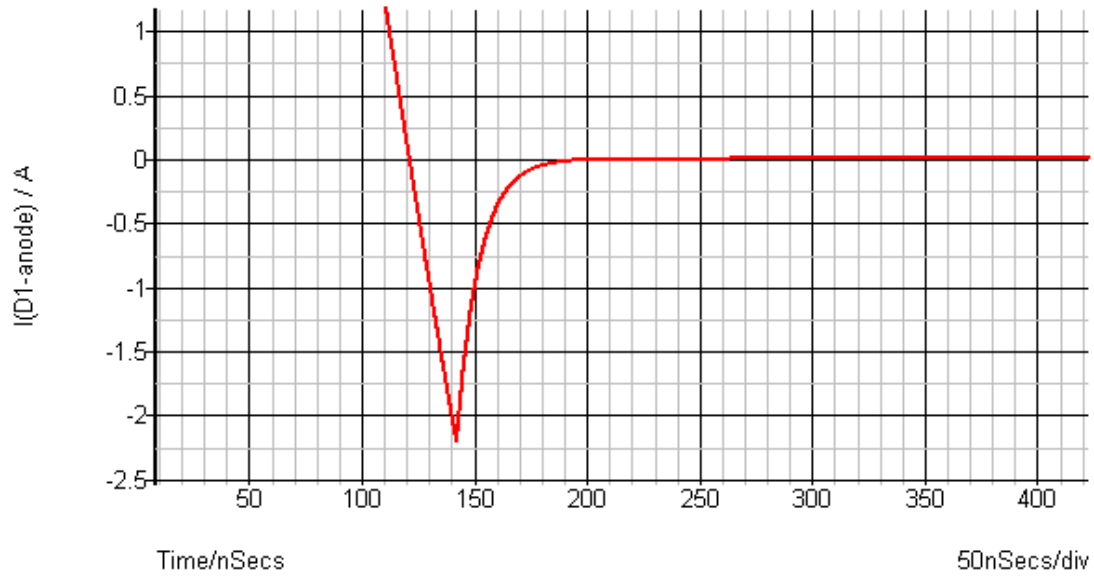

Figure 12-12. Soft Recovery Diode Curve

12.4.2 General Information About the Soft Recovery Diode

Referring back to the Create Diode Model Dialog, data required by the 'DC Forward bias spec' group will usually be supplied clearly on the data sheet.

The data required by the SIMetrix 'Reverse recovery specification group' is as follows:

- **IF** This is the forward current (specified as a test condition)
- **dIF/dt** This is the rate of change of current (specified as a test condition)
- **IRM** This is the peak reverse current
- **Tr** This is the time taken to recover to 37% of IRM (time constant)

These are the measurements that SIMetrix needs but manufacturers give variations on them. Values given in datasheets will be subject to some variation. For example the data sheet from which the above example was taken gave:

- **IF** explicitly as 15A
- A separate note on the data sheet to say that **dIF/dt** was 100e6 (100Meg).
- An equation to calculate **IRM** by multiplying the gradient by the time interval between crossing the x axis and reaching the minimum point 20nS).
- **Tr** was clear enough from adding together two time intervals of 10nS and 20nS.

This is typical of the teasing out of data to obtain it in the form that SIMetrix requires it.

Note: Be careful which information you use. The data sheet might be quoting figures for an extreme test condition and you may be trying to model performance in typical conditions.

Manufacturers don't often quote capacitance for rectifier diodes. This can be responsible for ringing in some circuits so it often useful to have a value that is at least the right order of magnitude. The datasheet shown above (from Fairchild) did not quote capacitance, but another (from International Rectifier) quoted a typical value of 55pf at 200V and even provided a graph showing capacitance vs reverse bias. The value we are interested in is the zero bias capacitance and from IR's graph we estimate this to be about 350pF. The SIMetrix diode model does include the effect of capacitance roll-off with the reverse voltage.

13 Introduction to Scripting

13.1 Overview

SIMetrix features a simple interpreted script language, loosely based on BASIC, in which most of the user interface is written.

The script language has access to every part of the application: schematic entry, waveform viewer and simulator. It can be used to automate repetitive actions, perform custom measurements and to define custom GUI elements.

In this section we will introduce the script language with three demonstrations:

1. Automated run that reads data from spreadsheet values
2. Distortion measurement
3. Customised toolbar

This training can do no more than offer a small sample of what you can achieve with scripts. For full reference information, please refer to the **Script Reference Manual**. This provides details of the language along with comprehensive documentation for the many hundreds of functions and commands that support the language. In addition the script manual provides details about customising the standard GUI by modifying the internal scripts.

To write scripts you will need a text editor and to this end we have supplied the free open-source editor Notepad++ with this training course. For details on installation and setting up Notepad++, please see Appendix C. If you have your own favourite editor you would like to use, you may also wish to see Appendix C. for information about setting up syntax highlighting for your own editor.

Although all the scripts we will use are supplied with this training course, we recommend that you type them manually because you will probably learn more from doing so than simply pasting from the supplied media. None of them are very long.

Note: The default temporary directory which SIMetrix uses is: My Documents\SIMetrix\Scripts. SIMetrix changes directory automatically at run time to the directory which holds anything you direct it to run. You may find it easier to copy all the files you are going to use to this directory and do all your work in it.

13.2 General Information

13.2.1 What is the Script Language for?

Scripting is useful for:

- Automated runs
  - Long runs to be done overnight
  - Custom multiple runs
  - Optimisation
- Measurements
  - Specialised analysis of simulation results
- Customise GUI
  - Edit/add/delete menus
  - Edit/add/delete toolbar buttons
  - Component Interfaces

13.2.2 Where to Put Scripts – The Script Directory

SIMetrix searches for the script you type in at the command line in the current directory, then the default user script directory. This is initially set up to be at SIMetrix\Scripts under your My Documents folder. Scripts are expected to have the extension **.sxscr**.

You can also enter a full path name with any extension.

If you are running a script from the current working directory, be aware that SIMetrix changes this directory according to whatever schematic is being displayed. This makes the current working

directory somewhat volatile. If the script you are developing is intended to operate only on schematics in the same directory as the script, this will not be a problem.

If, however, you are developing a script for more general use, you should always place it in the script directory.

In this training we are assuming that all scripts are placed in the script directory.

13.2.3 More Information

The definitive source of information on the script language is the Script Reference Manual. See menu **Help|Online Manuals|Script Reference Manual**.

13.3 Hello World – A Trivial Example

Anyone who has learnt the 'C' programming language will be familiar with the now celebrated "Hello World" program - possibly the simplest program that can be written. Here we will write and execute a SIMetrix "Hello World" script.

1. In the Command Shell, go to **File|Scripts|New**. This opens a text editor with an empty script.
2. Enter:

   Echo "Hello World!"

3. Save to the file "**Hello.sxscr**" then exit the text editor
4. At the command line in the Command Shell, run the script by typing: Hello

The celebrated text appears in the Command Shell.

13.4 Automated Runs – A More Complex Example

As a first example we will build a script that will perform multiple runs using a set of data obtained from a spreadsheet. The idea is to read the values of a number of components from a spreadsheet table and use these to set the values in a schematic.

There are various ways you can perform multiple runs with different data. That is, change component values, model parameters, circuit topology or analysis specification. These are:

- Run separate netlists. OK for running completely different designs
- Build include file in script. This is the most flexible and powerful method but a little tricky
- Pass data using global variables. Simple but limited to numeric value changes only
- Programmatically edit schematic. Useful if you don't want to get involved with netlists
- Pass data to Run command. OK for simple cases.

In this example we will be programmatically editing the schematic

13.4.1 Run a Script Using External Data

The task is to run a repeat simulation using component values taken from a spreadsheet. There are four problems that we will address:

- How do we get external data into the simulator? (next)
- How do we edit component values by running a script? (page 130)
- How do we create an Undo facility? (page 131)
- How do we run a simulation from a script? (page 131)

13.4.2 Importing Data

There are two methods of importing data using a script. The methods import the data:

- From the clipboard (next)
- From the file system (page 129)

13.4.3 Importing Data from the Clipboard

Now we will start to write some code.

1. Create a new script. The remainder of this course will assume that you have set up your preferred text editor as the standard SIMetrix editor. For details see Appendix C. Once set up the menus in **File|Scripts** will call up your editor.

   To create a new script select menu **File|Scripts|New**
2. Enter the following lines of code. (There is no space between the two single quotes in line 3. Make the * to be the first symbol on a line for that line to be a comment.

```
Let lines = ReadClipboard()
Let numLines = Length(lines)
Let refs = ''
for idx=0 to numLines-1
    if lines[idx]<>'' then
        Let firstDataLine = idx+1
        Let refs = Parse(lines[idx])
        exit for
    endif
next idx
Show firstDataLine
Show refs
```

3. Save the script with filename **show_refs.sxscr** to the script folder **My Documents\SIMetrix\Scripts**. (Scripts must have filename extension .sxscr).
4. Open the small file **My Documents\SIMetrix\Training\Section-13\data.xls** in a spreadsheet and copy all the data to the clipboard.
5. In the Command Shell, go to **File|Scripts|Run...** and in the Open dialog, select **show_refs.sxscr** and then click Open.

The output, which appears in the Command Shell, should be:

```
firstDataLine = 1
Index refs
0 'R1'
1 'C1'
2 'L1'
```

If you get an error message first check that you have entered the script correctly. If the error is:

```
Cannot find vector of name 'firstDataLine'
```

then you haven't correctly copied any data to the clipboard from the spreadsheet.

Among other things, this example introduces the **Parse ()** function. When you copy from a spreadsheet, each line is made up of tab-delimited strings.

13.4.4 Importing Data from a File

The previous example read data via the clipboard which may be convenient in some applications. But it is also possible to read the data from a file. Here we will show how to read the tabulated data from the file data.txt which you will find in **My Documents\SIMetrix\Training\Section-13**. This has the same values as the spreadsheet.

1. Change directory to **My Documents\SIMetrix\Training\Section-13** if not already. (You can use menu **File|Change Directory…**

2. Make the following two changes to the code, save it and run the script again. Change the first line to:

```
Let lines = ReadFile('data.txt')
```

3. Delete the last two lines (starting 'Show') then add the following line to the end of the code:

```
Show lines
```

4. Save then run the script and all the data in the file is now displayed in the Command Shell:

```
Index lines
    0 'R1   C1    L1'
    1 '10   2.000000E-09      1.000000E-06'
    2 '11   2.400000E-09      1.200000E-06'
    3 '12.1 2.880000E-09      1.440000E-06'
    4 '13.31      3.456000E-09      1.728000E-06'
    5 '14.641     4.147200E-09      2.073600E-06'
    6 '16.1051    4.976640E-09      2.488320E-06'
    7 '17.71561   5.971968E-09      2.985984E-06'
    8 '19.487171  7.166362E-09      3.583181E-06'
```

13.4.5 Editing Values using a Script

We will use schematic commands to edit schematic instance values programmatically. We will use the **Prop** command to edit the VALUE property.

1. Open **My Documents\SIMetrix\Training\Section-13\LCR.sxsch** in the Schematic Editor and note the values of the components
2. Open show_refs.sxscr from the previous example (you may have it still open in your text editor)
3. Delete the final line (the show command) and replace them with the following additional code:

```
Let numRefs = Length(refs)
for idx=firstDataLine to numLines-1
    if lines[idx]<>'' then
        Let values = Parse(lines[idx])
        for refidx=0 to numRefs-1
            Unselect
            Select /prop REF {refs[refidx]}
            Prop VALUE {values[refidx]}
        next refidx
    endif
next idx
```

4. Save the script with filename **edit_values.sxscr**.
5. Run the script and notice the values change on the schematic.
6. Using windows explorer or suitable alternative, make a copy of **data.txt** to a different filename.
7. Open **data.txt** in a text editor and change one or more of the values in the **last** line. Save it to the existing filename.
8. Run the script again and notice the value change on the schematic. The script sequentially changes the component values through all the values in the data.txt file, but the values that remain are the last ones in the table.
9. Delete the altered **data.txt** and rename the copy to **data.txt** so as to restore its original data. You will need it again later.

13.4.6 Creating an Undo Facility

One of the problems with the above procedure is that the original values are lost. This can be fixed using the Noundo and Undo commands. The Noundo command takes a snapshot of the schematic. Undo restores the snapshot.

1. Add the following code to **edit_values.sxscr**.

```
Noundo (as the first command in the script)
Undo (as the last command in the script)
```

2. Save the script as **undo.sxscr**.
3. Close schematic **without saving** and reopen
4. Rerun script. You will see the values flicker then return to their original values.

13.4.7 Run Simulator from a Script

We now have everything set up to run the simulation. Simulations are initiated by the Run command but, note, this command starts a simulation on a netlist not on a schematic. To run a schematic from a script, you must first create a netlist for it (see appendix D.).

1. Add these two lines immediately after the line 'next refidx', that is, as the last item in the if-endif statement:

```
Netlist LCR.net
Run LCR.net
```

2. Save the script as **run_simulation.sxscr**.
3. Run the script. You should see a graph such as Figure 13-1.

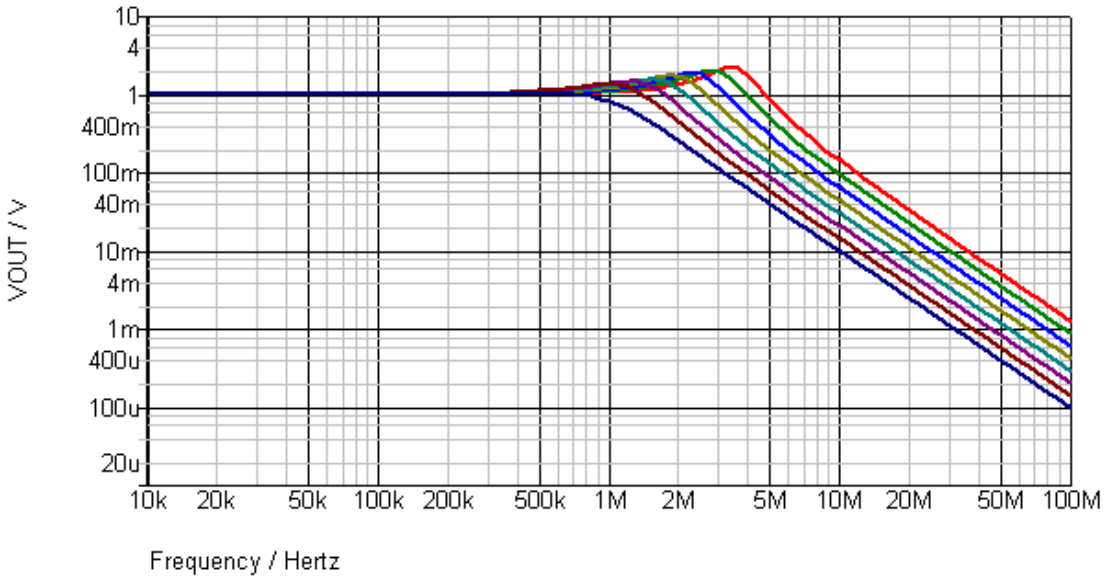

Figure 13-1. LCR Simulation Run from a Script

13.4.8 The Script Line-by-Line

Let's have a look at the script so far and see what each line is doing. The first line:

`Noundo`

We have already explained. This takes a snapshot of the state of the schematic. Later execution of the command Undo will restore it. The next line:

`Let lines = ReadFile('data.txt')`

reads the lines from the file 'data.txt'. The lines are stored in the string array 'lines'. Each element of the array is a single line from the file.

Let numLines = Length(lines)
assigns the number of elements in the array ‘lines’ to the variable numLines.

Let refs = ''
initialises the variable refs to an empty string.

for idx=0 to numLines-1
starts a for loop. This is iterating through each line in the ‘lines’ array.

if lines[idx]<>'' then
is an ‘if’ statement. This is testing to see if the line we are looking at is empty or not. If it is empty we skip it. This allows the data file to contain blank lines without causing an error in the script execution

Let firstDataLine = idx+1
this is storing the index of the next line containing data in the variable firstDataLine. This is so we know where to start looking for the data in the next part of the script

Let refs = Parse(lines[idx])
this parses the line containing the names of the components whose values we wish to change. The Parse() function breaks the text up into ‘tokens’ with each token in a separate element in an array. So in this case, Parse() will return an array size 3 holding the values ‘R1’, ‘C1’ and ‘L1’. This was demonstrated at the beginning – see 13.4.3.

exit for
exits the for loop. The purpose of this loop was to find the component names. We have now found them so we can exit.

endif
ends the if statement `if lines[idx]<>'' then` see above

next idx
ends the for-loop.

Let numRefs = Length(refs)
assigns the size of the refs string array to numRefs. This is the number of component references we found in the line ‘R1 C1 L1’ in the ‘data.txt’ file.

for idx=firstDataLine to numLines-1
starts a new for-loop. This iterates through the rest of the lines starting at firstDataLine.

if lines[idx]<>'' then
we test again for an empty line

Let values = Parse(lines[idx])
get the values from the line

for refidx=0 to numRefs-1
start a new for-loop – this one iterates through the 3 components we are altering

Unselect

The unselect command unselects everything on the schematic. We must do this because the next command is going to select something very specific so that we can edit it.

```
Select /prop REF {refs[refidx]}
```

Here we use a braced substitution for the first time. This is a very important concept in SIMetrix scripts. **Select** is a command that will select an object on the schematic according to a specification supplied consisting of a property name (**REF**) and a property value. In this case we want the property value that is stored in the array element **refs[refidx]**. If we omitted the curly braces ('{' and '}') the command would use the literal string '`refs[refidx]`' not the value stored in the array element. So this line selects a component with the REF property equal to the value stored in the array. This will be one of the values 'R1', 'C1' or 'L1'.

```
Prop VALUE {values[refidx]}
```

The **Prop** command edits a property on a select instance or instances. In this case it will edit the **VALUE** property to the value held in **values[refidx]**. Notice we again use a braced substitution.

```
next refidx
```

Terminates the refidx for-loop

```
Netlist LCR.net
```

This creates a netlist of the schematic and saves it in the file LCR.net. When we run a simulation, we run it on a netlist (see Appendix D.) but first we must create the netlist. This is done by the Netlist command.

```
Run LCR.net
```

Finally we run the simulation.

```
endif
```

Terminate the if-statement

```
next idx
```

Terminate the for-loop

```
Undo
```

This will restore the state of the schematic that was saved by the Noundo command at the top of the script.

13.4.9 Handling Simulation Data

When running multiple simulations, consideration must be given as to how the simulation data is handled. By default, SIMetrix keeps the data for the three most recent runs. So if you are running four or more runs from a script only the data for the last three will be available on completion. You can handle data in a number of ways:

- Do a measurement and display the result at the end of each run. The data can be then be safely discarded
- Save the data for later recovery. Use **SaveGroup** command which is fully described in the Script Reference Manual.
- Plot the results as needed at the end of each run using either fixed probes or **Plot/Curve** commands in the script.

13.4.10 Simple Measurement

1. Add these lines to the script after the Run command.

```
Let minus3dbPoint = XFromY( db(VOUT), -3)
```

```
if length(minus3dbPoint)<1 then
    Echo "Data does not cross -3db"
else
    Echo {'Minus 3dB point is at: ' & minus3dbPoint}
endif
```

2. Save the script as **measurement.sxscr**.
3. Delete the fixed probe from the schematic
4. Run the Script
5. This outputs a simple measurement that locates the -3dB point of the response. The output appears in the Command Shell and you can copy it and paste into a spreadsheet. It looks like the following:

```
Minus 3dB point is at: 5.33017Meg
Minus 3dB point is at: 4.40728Meg
Minus 3dB point is at: 3.63769Meg
Minus 3dB point is at: 2.99616Meg
Minus 3dB point is at: 2.46116Meg
Minus 3dB point is at: 2.01473Meg
Minus 3dB point is at: 1.64201Meg
Minus 3dB point is at: 1.33078Meg
```

The final item may have the wrong value if you did not restore the original values in **data.txt** after Editing Values using a Script on page 130.

The above made use of the analysis function XFromY(). This function returns the x value of a curve given its y-value. See the Function reference chapter of the script reference manual for full details.

13.5 Experiments with Simulation Data

You will need to access simulation data in a script if you wish to perform any kind of measurement or analysis. Here we do some simple experiments to show how data is organised.

13.5.1 Data Groups

1. Shut down SIMetrix, then restart it
2. In the Command Shell, go to **Graphs and Data|Change Data Group...**
3. Notice that there is just a single entry: 'global (Global vars)'. This is the global group and is always available
4. Now open the **lcr.sxsch** schematic and run it
5. Select **Graphs and Data | Change Data Group...** menu again. Notice there are three groups: 'op1', 'ac1' and 'global'. The 'ac1' group should be highlighted. This is the current group and this is from where the data for any plotting operation will be obtained. 'op1' contains the data generated during the DC operating point analysis that always precedes an AC analysis
6. Close the dialog
7. In the Command Shell, type:

   Display

   This lists all the vectors in the current group - i.e. group 'ac1'. The term ‘vector’ is widely used in SIMetrix documentation. A vector is the data for a single simulation signal, usually a voltage or current. So the vectors in the current group are the voltages and currents available for plotting.

13.5.2 Accessing Simulation Data in a Script

Any vector in any group can be accessed using a name in the form:

```
groupname:vectorname
```

An example would be:

```
ac1:VOUT
```

This name can be used in an expression for plotting or analysis.

The groupname may be omitted if it is in the current group. For example if ac1 is the current group, you can use :VOUT instead of ac1:VOUT.

When a successful simulation is run, at least one new group is created. This group becomes the current group so you can always access the data in the latest run using the form :vectorname.

Voltage and current vectors follow consistent naming rules. Details can be found in the Simulator Reference Manual, Chapter 2.

13.5.3 Experiments with Vector Names

1. In the Command Shell, go to **Graphs and Data|Delete Data Group...** click Select All and then OK to delete all the data groups. (This won't actually delete the global group)
2. On the Schematic Editor, open **lcr.sxsch**, delete the probe if it is still present and run the simulation.
3. In the Command Shell, type:

   ```
   Plot :VOUT
   ```

   a new graph is plotted. This has plotted the vector VOUT in the current group. You will note that the graph uses linear scales whereas if you plot using the menus you would get logarithmic scales.
4. Close the graph.
5. In the Command Shell, go to **Graphs and Data|Change Data Group...**, note the name of the ac group (you will need it in step 7 below) and change the current group to 'global'.
6. Again type at the command line:

   ```
   Plot :VOUT
   ```

   Notice the error message in the Command Shell:

   ```
   Error : Cannot find vector of name 'VOUT'
   ```

   This is because we are trying to plot a vector called VOUT in the current group, but the current group is 'global' and there is no vector called VOUT in that group.
7. Now type at the command line:

   Plot *groupname*:VOUT
   where *groupname* is the group noted in step 5 above

   Notice that the graph plots OK because we are accessing data using a fully qualified name.

14 Background Information on Scripts

The following describes how scripts are processed by SIMetrix. An understanding of the inner workings will help with script development and diagnosing problems when they arise.

14.1 Script System Structure

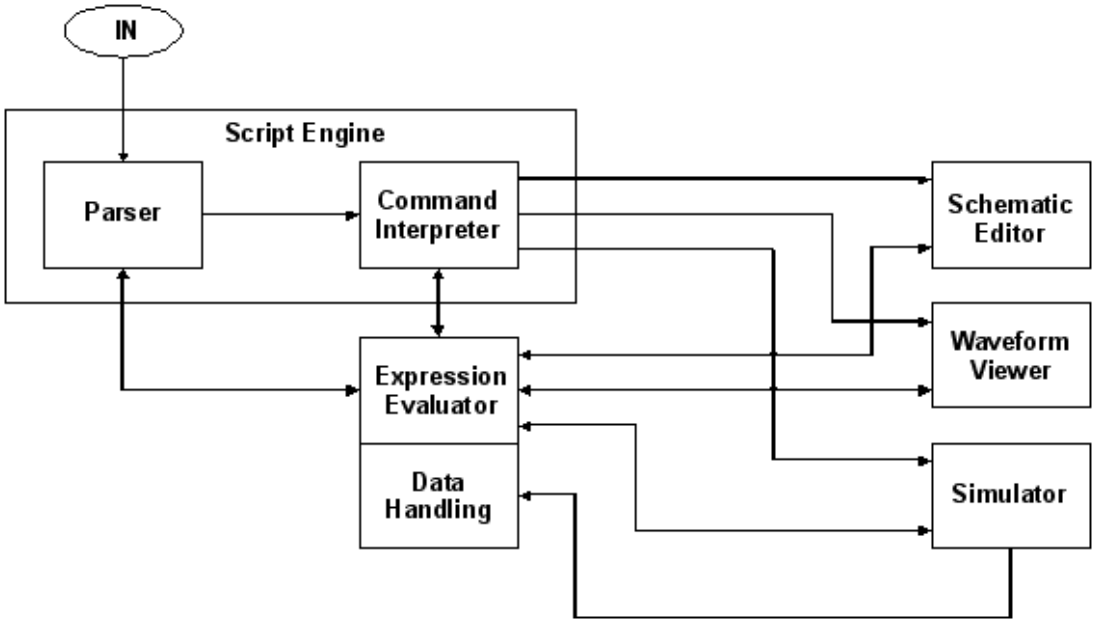

Figure 14-1. Script System Structure

14.1.1 How a Script is Processed

Script file is presented to the Parser.

- The Parser analyses the script's grammar then executes it line by line
- The Parser recognises keywords such as **if, else** and **for**.
- Any line that does not start with a keyword is assumed to be a command

The Parser executes the script by sending commands to the Command Interpreter which scans the command line and breaks it up into words or phrases known as tokens. A token is either a single unquoted word or a group of words enclosed in quotation marks: "". Braced substitutions are evaluated and presented as a single token. (Multiple tokens if expression evaluates to a non-scalar value)

The scanned command is then sent to the part of the program responsible for executing that command. This could be (but not limited to) the schematic editor, waveform viewer, simulator or command shell.

Both the Parser and the Command Interpreter may encounter expressions. The Parser reads them in test expressions for loops and conditional statements. The Command Interpreter encounters expressions in braced substitutions and as arguments to the **Let**, **Plot**, **Curve** and **Show** commands. These four commands are the only commands that can accept an expression directly.

All expressions are evaluated by the Expression Evaluator. Expressions may make calls to functions.

14.1.2 Data Handling

The basic "unit" of data is a vector. The voltage and current data created by a simulation run is a vector. Vectors used for simulation data and variables used in scripts are essentially the same thing.

The simulator evaluates the voltage on every node in the circuit and (virtually) every device pin current at every simulation point. This generates large amounts of data which can easily be take up more space than available RAM, so data is stored to disk located in the temporary data directory.

Individual vectors are recovered to RAM on demand for plotting or analysis.

Vectors are organised into data groups. There is usually one group stored in each data file. Each analysis creates a single group with a name related to the analysis type. E.g. 'tran1', 'op2', 'ac3' etc.. These are stored in files of the same name with the extension **.sxdat**.

15 Analysis Functions

SIMetrix has a range of mathematical functions allowing detailed analysis of simulation results. To demonstrate, we will use the **FFT()** function to analyse distortion in a signal. This exercise will also illustrate the **Truncate(), Range(), Interp(), XFromY()** and **Locate()** functions.

The approach we will adopt is to separate out the Fundamental and measure its amplitude using the **Mag()** function. We then measure the residue after removal of the fundamental. From these two value we can calculate the distortion.

We will build the code in chunks and save each chunk with a meaningful filename. Before we proceed with next example, we shall look at some particular features of the language that we will be using.

15.1 More about Vectors

15.1.1 The 'time' Vector

The vector called **time** is always generated by a Transient analysis and carries the values of the time points. Usually the first point would be zero, but be aware that there is a transient analysis option that causes data at the start of the run to be skipped. In this case the first point would time of the first data point output. The last value of the time vector is always the last time point simulated.

15.1.2 The 'length' Function

In SIMetrix the function **length()** returns the number of elements in a vector. Consequently the function **Length(time)** carries the number of time points.

15.1.3 Simulation Vectors and References

Most vectors generated by the simulator actually comprise two arrays of values. One array contains the y-values and the other the x-values. The x-values are sometimes referred to as the 'Reference'. You can use the **Ref()** function to access the x-values explicitly.

15.1.4 Complex Vectors

Simulation vectors returned from an AC analysis are complex. Also some functions, e.g. the **fft()** function, also return complex data. However, there are differences between these two examples of complex vectors. The AC analysis return data has complex y-values but real x-values whereas the return value from **fft()** has complex x-values, even though they represent a real quantity. In the latter case the imaginary terms are always zero.

Some functions do not accept complex arguments and for these we need to convert the complex data to real data in a meaningful way. Functions that convert complex data to real values include, **real()** which returns the real part, **imag()** which returns the imaginary part, **mag()** which returns the magnitude ($=\sqrt{(real^2+imag^2)}$), **phase()** which returns the phase in degrees and **phase_rad()** which returns the phase in radians. The phase functions operate a state machine which keeps track of 180/-180 flips and returns a continuous phase output. In some situations this behaviour is undesirable and in such cases you can use the **arg()** function which returns an output bounded between 180 and -180.

In the following example you will see the **mag()** function used to pass data to **Locate()**. Locate accepts only real arguments.

15.2 FFT and the Need for Pre-processing

The **FFT()** function performs a spectral analysis of the supplied vector using the Fast Fourier Transform algorithm. This algorithm imposes the following requirements on the input time domain data:

- The data points must be evenly spaced
- The number of points must be a binary power - e.g. 1024, 32768 etc.

Simulation data does not usually comply with the above requirements, so we therefore use the **Interp()** function to interpolate the data before presenting it to the **FFT()** function.

There is a further problem. To give an exact answer, Fourier analysis assumes that the input signal is exactly periodic and that the data presented is a whole number of cycles. When these conditions are not met, the result is 'spectral leakage' whereby some unintended frequency terms show up as sidebands to the expected spectral lines. Spectral leakage can be minimised by two methods:

1. Truncate incomplete cycles from the beginning and end of the data. We can do this if we know in advance the exact fundamental frequency. With a simulation this is often the case as it would have been specified in a signal source. For circuits driven by a free running oscillator, this would not be the case.
2. Apply a windowing function. This is typically a bell shaped curve that tapers to zero at each end. This does itself distort the result, but does so in a predictable and usually non-harmful way.

The **FFT()** function applies the 'Hanning' window by default and this is usually adequate for most applications. The data truncation operation, however, is not included in the **FFT()** function, so we must do this separately.

We will demonstrate how to do this in the example below.

15.3 FFT In Action

The schematic **My Documents\SIMetrix\Training\Section-15\test_fft.sxsch** is the source which generates the raw data. It is a sine wave with a small amount of 3rd harmonic added to it.

15.3.1 Pre-processing

Enter the following code in a text editor.

```
Let fundamental = 20k
Let data = :VOUT
Let stopTime = :time[length(:time)-1]
Let numCycles = floor(stopTime*fundamental)
Let zeroCrossings = XFromY(data, 0)
Let endTime = zeroCrossings[length(zeroCrossings)-1]
Let startTime = endTime - numCycles/fundamental
Let truncatedData = Truncate(data, startTime, endTime)
Let interpolatedData = Interp(truncatedData, 32768)
Plot /ylog fft(interpolatedData)
```

Now:

1. Save the code as **Pre_process.sxscr**
2. Run the schematic **test_fft.sxsch** to generate the raw data.
3. In the Command Shell go to **File|Scripts|Run** and run the script pre_process.sxscr
4. On the Waveform Viewer, zoom in on about half a grid width from the start and you will see the Fourier clearly showing the fundamental and the 3rd harmonic shown as Figure 15-1.

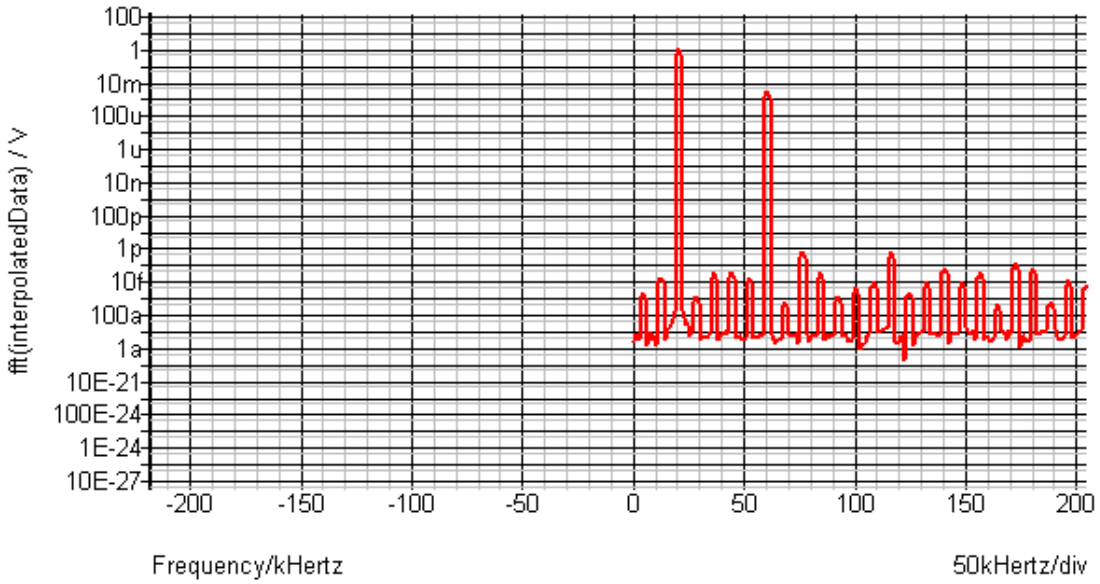

Figure 15-1. Script-generated Fourier Analysis

Let's go through this script line-by-line:

```
Let fundamental = 20k
```

Set the fundamental to 20k. (This is exactly equivalent to setting the frequency on the Schematic Editor in **Probe|Fourier|Arbitrary|Fourier tab**. In Signal Info group, you could have checked **Know fundamental frequency** and enter it in the Frequency field.)

```
Let data = :VOUT
```

Expression - (This is equivalent to setting the expression on the Define Curve tab of the same dialog. The colon signifies that the data is a vector in the current data group.)

```
Let stopTime = :time[length(:time)-1]
```

Assigns the value of **stopTime** which we will need to know. You can get a direct readout of this by typing **Show :time[length(:time)-1]** into the Command Shell (the answer is in seconds).The **time** vector returns the simulation time points, so therefore **length(time)** returns the number of time points. **time[index]** returns a single value, so for example **time[0]** returns the first time point **time[1]** the second etc. As we start from index zero, the last point is at index **length(time)-1**. So the final time point is at **:time[length(:time)-1**].

```
Let numCycles = floor(stopTime*fundamental)
```

This is multiplying stopTime by the fundamental frequency (the product is 20.2 in this case) and rounding down using the **floor()** function to obtain the number of complete cycles.

```
Let zeroCrossings = XFromY(data, 0)
```

Best results are obtained if the complete cycles start and end at zero. The **XfromY** function is effectively a lookup function which returns all the values of x for a single specified value of y (if there are any). Specifying zero as the y value, this populates the zeroCrossings array with the set of x values at which the curve crosses the x-axis.

```
Let endTime = zeroCrossings[length(zeroCrossings)-1]
```

This is finding the last point at which zero crossing takes place. This is assigned to the vector **endTime**.

```
Let startTime = endTime - numCycles/fundamental
```

This is counting back from the end. It first turns **numCycles** into a time by dividing by a frequency, subtracting this from **endTime** and assigning it to **startTime**.

```
Let truncatedData = Truncate(data, startTime, endTime)
```

We can now identify a number of complete cycles which start and end at x=0. We now assign this to **truncatedData**. This completes the cleanup but the time steps are still not evenly spaced so interpolation is now necessary.

```
Let interpolatedData = Interp(truncatedData, 32768)
```

Here we interpolate the data with 32768 points which, by their nature, are evenly spaced. The data now complies with FFT requirements.

```
Plot /ylog fft(interpolatedData)
```

We call the **fft** function to analyse the waveform and the plot command to plot it. The '/ylog' in the line of code determines that the y axis is to be logarithmic.

15.3.2 Identify the Index of the Fundamental

Replace the last line (Plot) of **Pre_process.sxscr** with the following code:

```
Let spectrum = FFT(interpolatedData)
Let freqVec = Ref(spectrum)
Let fundIndex = Locate(mag(freqVec),fundamental)
```

Line 1: Create the 'spectrum' array and assign the FFT output to it.

Line 2: Create the 'freqVec' array and use the Ref() function to extract the frequencies (the x values) from spectrum.

Line 3: Create the 'fundIndex' vector and use the **Locate()** function to find the index of the fundamental frequency in freqVec.

There is no need to save the script at this point because we do not run it.

15.3.3 Calculate the Amplitude of the Fundamental

Append the following code to the script you entered in the previous topic. This illustrates the **Range()** function to select the required range. The range we will select is between 0.5 and 1.5 times the fundamental.

```
Let fundamentalRange = Range(spectrum, fundIndex/2,(fundIndex*3)/2)
Let numElems = Length(fundamentalRange)
Let fundamentalMag = sqrt(mag(mean(fundamentalRange^2) * numElems))
```

Line 1: Assign the sub-range of 'spectrum' to the new array 'fundamentalRange'.

Line 2: Assign the number of elements in the sub-range to the new vector 'numElems'.

Line 3: This is capturing the magnitude of the fundamental; we will need this later. The calculation is a root mean square of all the magnitudes in the sub-range.

1. Save the script as **FundamentalMagnitude.sxscr**.
2. Now, as a check on progress, add the line
   ```
   Show fundamentalMag
   ```
3. Run the script to display the value which should be 1.23393045.

15.3.4 Eliminating the Fundamental Range

Having measured the magnitude of the fundamental we must now remove it (and the DC term) from the original data 'spectrum'. This is so that we can measure everything else that was in the raw data. We do this by assigning zero to all the elements in fundamentalRange. This array is complex so we must assign a complex value.

Append the following code to **FundamentalMagnitude.sxscr**:

```
for idx=fundIndex/2 to (fundIndex*3)/2
    Let spectrum[idx] = (0,0)
next idx
Let spectrum[0] = (0,0)
Let spectrum[1] = (0,0)
```

Line 1: Start the **for** loop.

Line 2: Set the complex elements to zero. (0,0) means zero as a complex number.

Line 3: Next

Line 4: Set the DC term to zero.

Line 5: This is necessary because the default Hanning window 'leaks' the DC component to the second bin (spectrum[1]). If the **fft** were specified with no window (labelled as 'Rectangular' window in the options) this line of code would not be necessary. Also be aware that the 'Blackman' window requires even wider suppression of the DC component.

15.3.5 Calculate the Magnitude of the Residue

This is exactly the same use of the **mag()** function but this time on 'spectrum'. Remove the “Show fundamentalMag” line then append the following code :

```
Let residue = sqrt(mag(mean(spectrum^2) * Length(spectrum)))
Show residue
```

Now:

1. Save the script as **ResidueMagnitude.sxscr**.
2. Run the script

The correct result is 0.0025 and you will see this is the peak value of the third harmonic in the graph obtained in Figure 15-1. But the Command Shell is indicating a result of 0.00306186. What's wrong?

The problem is caused by the Hanning window which is used by the FFT function. This is scaled to make peaks (i.e. by measuring a single bin) correct, but we have measured not just the peak but also two sidebands that are a result of the Hanning window. If these sidebands are included the overall power is overestimated by 1.5. This means the result is high by the square root of 1.5. This is one of many FFT gotchas, but is not important in the present process because we will be calculating the residue in the same way and subject to the same scale factor. As the end result is a ratio of these two calculations the overestimates cancel. However, if you used the above method to calculate the absolute value of a spectral component, you should be aware of this scale factor.

15.3.6 Final THD Calculation

We now add the final calculation for THD as the ratio of Residue/fundamentalMag. We will use the **FormatNumber()** function to format the ratio as a percentage. Remove the Show command then append the following two lines of code to **ResidueMagnitude.sxscr**:

```
Let thdMsg = 'THD = ' & FormatNumber(residue/fundamentalMag,4,'%')&'%'
Echo {'THD=' & thdMsg}
```

Save the script as **THDCalculation.sxscr**.

15.3.7 Plot Result and Annotate Graph

As a final flourish, we will plot a graph of the Fourier spectrum and annotate it with the distortion result using a Text Box.

Add the following line immediately after the 'Let spectrum = FFT(interpolatedData)' line (in Identify the Index of the Fundamental on page 141.

```
Let origSpectrum = spectrum
```

Add the following lines at the end of the script:

```
Plot /ylog origSpectrum
AddTextBox {thdMsg} 0.95 0.95
```

Save the script as **MyMeasureofDistortion.sxscr**.

Now run (and debug!) the script. If all else fails, crib the one we prepared earlier which is **measure_dist.sxscr**. The final output is shown as Figure 15-2.

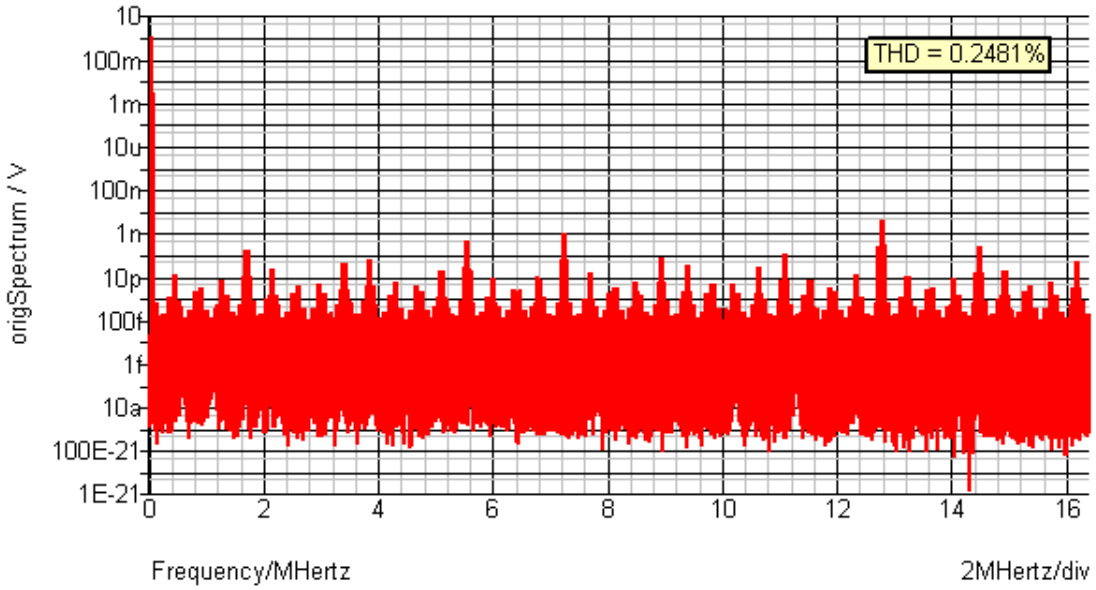

Figure 15-2. Output from FFT Script - annotated

It has plotted the original spectrum (before elimination of the fundamental range) and annotated it with the THD ratio.

16 Scripting to Customise the User Interface

16.1 Scope

All menus, key definitions and toolbars call a script or a script command. The majority call a built-in script which is a script built in to the SIMetrix binary executable files.

- Simple menu and accelerator key definitions maybe edited using **File|Options|Edit Menus...**
- Menus can also be edited, added or deleted using the **DefMenu** and **DelMenu** commands executed from the startup script

Toolbars and tool buttons can be created/edited using the commands **DefButton, CreateToolButton, DefineToolBar** and **CreateToolbar** . As with the menu commands these should be executed from the startup script.

16.2 Built-in Scripts

The Graphical User Interface (GUI) is implemented in the script language using built-in scripts. Points to remember include:

- The source for all built-in scripts can be found on the install CD. You can also obtain the source for the built-in scripts from the download page for the version you are using.
- Built-in scripts can if necessary be overridden, but it is usually better to redefine the GUI elements that call them (menus, toolbuttons etc).
- The entire menu system is constructed when SIMetrix starts by running the built-in script **menu.sxscr**.

16.3 Defining Menus with DefMenu

1. In the Command Shell, type the following. Note that there are no spaces before or after the '|' symbol.

```
DefMenu "Shell|&File|Hello World" Hello
```

2. Now look in the Command Shell File menu. Notice the new menu item at the bottom. Selecting this menu item runs the script **Hello.sxscr** created earlier. You can also put the menu item at the top by entering the following line. But this won't override the menu definition created in 1. above; to see this work, restart SIMetrix first.

```
DefMenu /pos 0 "Shell|&File|Hello World" Hello
```

Notes:

- To make new menu definitions permanent, use the commands in a script and make a call to this in the startup script. To edit the startup script, use:

```
File|Scripts|Edit Startup
```

- The pipe symbol '|' is the submenu separator
- First name is the identifier for the top level menu. 'Shell' signifies the command shell main menu. See DefMenu documentation for others
- 'File' is prefixed with '&'. '&' always prefixes the Alt key accelerator, in this case the 'F' in 'File'. This is shown underlined when the Alt key is pressed.

16.4 Defining New Tool Buttons

You can define new tool buttons and also new tool bars. First we look at how to add a new button to one of the predefined toolbars. There are three stages:

- Create the new button using the CreateToolButton command. This defines the button's graphic and name but not the command it calls
- Add the new button to the desired toolbar using the Set command
- Define the command or script that the button will call.

16.4.1 Creating the Button

We will define a button to run our trivial "Hello World" script. First we must define a graphical image for the button. This can be one of:

- The predefined built-in images as listed in the script manual (see **DefineToolBar** command documentation)
- An external image that you create from scratch or obtain from a clipart library. We will use an external image.

1. Look in the **images** folder and locate the file called **globe.bmp**
2. Copy this file to the folder support\Images under the SIMetrix root (typically this would be C:\Program Files\SIMetrix560). All images used for toolbuttons must be in this folder. Note that this image is 32 x 32 pixels. All standard buttons are 16 x 16. SIMetrix will adjust the tool bar size to fit
3. In the Command Shell, go to **File|Scripts|New** to create a new script called add button.sxscr
4. Add these lines (the line start 'Set' should be all one line):

```
CreateToolButton hello_button globe.bmp "Hello World"
Unset Commandshellmainbuttons
Set Commandshellmainbuttons= "SchemNew;SchemOpen;-;SymbolNew;-
;Print;Options;hello button"
DefButton hello_button Hello
```

5. Run the script in the usual way
6. To make button definition permanent, make call to script in the startup script. (**File|Scripts|Edit Startup**)

See the Script Reference Manual Chapter 7 "Creating and Modifying Toolbars" for details of predefined toolbars.

16.4.2 Adding the Button to a New Toolbar

This introduces the **CreateToolBar** and the **DefineToolBar** commands. To add buttons to a new tool bar use the DefineToolBar command

Create a script called **add_button_to_new_toolbar.sxscr** and add these lines:

```
CreateToolButton hello_button globe.bmp "Hello World"
CreateToolButton help_button quest.bmp "Help"
CreateToolBar CommandShell ShellTop
DefineToolBar ShellTop "hello_button;help_button"
DefButton hello_button Hello
DefButton help_button Help
```

Copy the file images\quest.bmp to the folder support\Images under the SIMetrix root as described in step 2 of 16.4.1 above. Run the script.

The above lines create a new tool bar in the command shell with two buttons.

Appendix A. - List of Acronyms

List of Acronyms

ADC	Analogue-to-Digital Converter
BJT	Bipolar Junction Transistor (NPN and PNP to distinguish them from other types)
DCOP	DC Operating Point
DFT	Discrete Fourier Transform
ESL	Effective Series Inductance
ESR	Effective Series Resistance
FFT	Fast Fourier Transform
GUI	Graphical User Interface
IGBT	Insulated Gate Bipolar Transistor (SIMetrix has a built-in model)
LNA	Low Noise Amplifier
MC	Monte Carlo (analysis)
PWL	Piece-wise Linear (pieces of linear section)
RDS	Drain Source Resistance (FET)
RMS	Root Mean Square
SLM	Symbol Library Manager
SMPS	Switch Mode Power Supply
TC	Time Constant
THD	Total Harmonic Distortion
WC	Worst Case

Appendix B. - Glossary

Terms in bold are, themselves, present in the glossary.

Association	SIMetrix maintains libraries of **models** and **symbols** separately. Prior to placing a symbol on the Schematic Editor it must be associated with a **model** and it is at this point that the symbol acquires some of its **properties** that give it the required functionality in the circuit. There are various forms of Association and they are all described in this training.
Component	A block which sits within a **hierarchy**. A component has two parts, a **symbol** and/or a schematic or other components
Curly Braces {}	SIMetrix will evaluate any expression that it finds in curly braces.
Curve	Line on a graph sometimes known as a **Trace**. See also **Plot**.
Data Group	A collection of **vectors**. In the Waveform Viewer the tabs are identified by labels such as tran1, tran2 etc. These are data group names. (see also **group**)
Directory	SIMetrix maintains the concept of a current directory. It can be changed in the Command Shell using File|Change Directory... or in the Schematic Editor by switching to the tab of a schematic saved in a different directory.
Display Name (Schematic/symbol editor)	Name displayed to the user for a symbol. The display name may be descriptive and contain spaces and other non-alpha numeric characters. By contrast the i**nternal name** is used to identify a symbol and may only contain alphanumeric characters
Focus	There is a distinction between having focus and being 'Selected'. If you have more than one window open, Command Shell|New Schematic opens a new tab in the window that is marked as 'Selected'.
Genealogy	Relationships in a Hierarchical Schematic.
Graph	This term is used to denote the complete package containing the title, axes, units, curve(s) and grid.
Hierarchy	A tree structure made up of **components** and connections between them.
Ideal Component	A component whose only characteristic is its primary function. For example, an ideal (perfect, unrealisable) transformer is one which has no losses, resistance, saturation, capacitance, hysteresis etc.
Internal Name (Schematic/symbol editor)	This is the symbol name by which the Schematic Editor identifies a symbol. This is distinct from the **display name** which is the descriptive name displayed to the user in various user interface elements.
Macromodel	A Macromodel is a model defined as a sub-circuit of primitive devices such as transistors, fixed sources and controlled sources. Operational amplifiers are typically modelled using macromodels.

Model	The term ‘Model’ has two meanings in common use and they are often confused. They are as follows: • Device equations. A ‘model’ defines a set of equations that model the behaviour of a device such as a bipolar transistor or MOS transistor. ‘Gummel-Poon’ for example is the model used for the basic bipolar transistor. Other examples of models in this sense include ‘BSIM3’ and ‘VBIC’. • A set of parameters that plug in to the device equations to define a specific manufactured part number or IC process element. For example the model for a 2N2222 transistor would consist of a number of parameters such as IS and BF that are referenced within the Gummel-Poon device equations.
MODEL (property)	The MODEL property typically has a value that is a single letter. It defines the first letter to be used on the **netlist** line and so therefore defines the type of device. For example a MODEL property of ‘Q’ defines a bipolar transistor.
Model Library	The repository in which models are stored. Accessed from the Schematic Editor on the Parts menu. See also **Parts Browser**.
Module Port	A point in a hierarchical structure where a child interfaces with its outer world. The child's module port names must be identical to the pin names of the symbol that hosts the child.
Name (of a property)	All properties have a ‘name’ and a ‘value’. For more information, see the User’s Manual, Chapter 4 → Properties
Netlist	The text file that defines the circuit topology and also specifies the analyses to be performed by the simulator. See also Appendix D.
Parameterised Part	A part that is defined by user-supplied parameter values.
Parser	In computer science and linguistics a 'parser' is an algorithm that analyses a sequence of **tokens** to determine their grammatical structure.
Parts Browser	The tool used to view the **Model Library**. This term is used in some of the SIMetrix documentation.
Persistence	The number of "old" **curves** from previous runs that will be kept. For example, setting persistence to 1 will cause all but the most recent to be deleted. Setting to 0 means "keep all".
Point	A single value (time, frequency etc)
Properties	A property is an item of text attached to a schematic instance. Each property must have a name and a value. The most important are **Ref, Value** and **Model**. To view and edit properties, select the item, right click and select Edit Properties.
Random probing	Probing the circuit after a simulation has taken place.
REF (schematic property)	The REF property is the schematic property that is used to identify a schematic device. For example R12, Q4, C45. The REF property is sometimes called a ‘reference designator’.
Reference vector	The reference vector is another name for the x-values of a vector. Typically this is ‘time’ or ‘frequency’

Symbol	A graphical representation of a component which may, or may not, have an **association** with a **model**.
Token	One item in a text string such as a command line. Typically tokens are separated by spaces but words separated by spaces but enclosed in quotation marks are also considered to be a single token.
Trace	See **curve**.
VALUE (property)	The term 'value' has two meanings in SIMetrix which can be confusing. There is the 'value' of a property (see below) and there is also a property with the name VALUE. In this training, the latter is distinguished by using upper case letters. The VALUE property typically carries the value of a part (e.g. '1k' for a resistor) or its model name (e.g. Q2N2222)
Value (of a property)	All schematic properties have a 'name' and a 'value'. For more information, see the User's Manual, Chapter 4 → Properties
Vector	A single data item. Its name is called a Vector **Name**.

Appendix C. - About Text Editors

To use the script language you will need a text editor. Included with the media is the free (and open source) text editor "Notepad++". Despite its name, this has no connection or similarity to the standard Windows program 'Notepad'.

Also included with the media is the file userDefineLang.xml which defines syntax highlighting for the SIMetrix script language for use with "Notepad++". See below for instructions on how to install this file.

We have included this particular editor because it is free, redistributable and has adequate features. You may already have a favourite editor and will of course prefer to use that. If you wish to set up syntax highlighting for your chosen editor, see section below.

We strongly recommend that you **do not** use the standard windows editor Notepad. But if you do, you should be aware of one of its irritating quirks as follows:

It will append whatever filename you enter with .TXT even if you supply an extension. The only two ways to stop it doing this are to associate the extension you want to use with Notepad or set Save As to 'All Files).

Finally, whatever you do, avoid using a word processor as a text editor. Even if you save your file as plain text, there is a risk that it will use non-ASCII characters (e.g. Hex 96 for '-' instead of Hex 2D). You will get strange error messages if this happens.

Setting Up Syntax Highlighting for Notepad++

Supplied with the training material is the file userDefineLang.xml which defines SIMetrix script language syntax highlighting for Notepad++. We have not found an official method of loading this file. We simply established by trial and error where these details were stored. The following procedure works for the version supplied with the training material:

1. Shut down Notepad++ if it's open.
2. Delete all files in 'c:\documents and settings\<username> \Application Data\Notepad++'. This will delete any setting you already have. If this is unacceptable, you will probably need to setup syntax highlighting from scratch. See next section.
3. Copy the userDefineLang.xml file to c:\documents and settings\<username>\Application Data\Notepad++

Step 3 appears to be necessary. Just copying the userDefineLang file after Notepad++ has already created its configuration doesn't seem to work.

Setting Up Syntax Highlighting for any Text Editor

Obviously you need to refer to the documentation with your text editor to setup syntax highlighting. What you will need though is a list of keywords, function names and commands that SIMetrix uses. Here is a list of keywords:

all
do
else
elseif
end
endif
endwhile
exit
for
if
loop
next
script
step
then
to
while

AND
OR

To get a list of function names, type this at the command line:

```
ListFunctions /name functions.txt
```

This will write a list of function names to the file functions.txt (in current working directory). The list will actually include some functions that are not documented in the script manual. These are either for internal use only, for use by SIMetrix developers only for testing purposes, or in some cases functions that are under development and are not yet mature enough to be released.

To get a list of commands type at the command line:

```
ListCommands commands.txt
```

This will write a list of command names to commands.txt. As for functions, this also includes some undocumented commands.

Appendix D. - Netlists

Throughout the training, reference is made to 'Netlists'.

A netlist is a text file that is a description of the circuit. It includes details of the interconnections between devices. In the strictest meaning of netlist, this is all that it would have, but SIMetrix uses a looser definition which also includes model definitions and simulator commands. In short it is a text file that contains enough information for a simulation to be run (although it may omit library models as the simulator will obtain these for itself). 'Netlist' in the sense that we use it is also traditionally known as a 'Simulation Deck'; this is terminology that dates back to a time when computer input was typically in the form of a deck of punched cards.

When a simulation is run on a schematic, SIMetrix first creates a netlist and then presents it to the simulator. The simulator then runs whatever analyses are specified in the netlist and sends the data back to the front end for plotting or other processing.

In the schematic editor, there is a feature that allows you to explicitly add lines to the netlist. This is what has become known as the F11 window. Users that are familiar with the netlist format can add additional commands or model definitions that are not available through the GUI - or maybe are available in the GUI but are easier to enter directly.

For information on the netlist format, please refer to the Simulator Reference Manual Chapter 2 heading 'Netlist Format'.

Appendix E. - Graph Symbolic Values

Most graph objects have one or more label properties that can be used to display text on the graph. As well as literal text, these label properties may also use symbolic values enclosed with '%'. These symbolic values return values of other properties belonging to the object. For example curve marker objects have a property called ‘X1’ which is always set to the x-location of the curve to which it is attached. So %X1% in a curve marker label will return the x-location allowing it to be displayed on the graph. The X1 property is updated every time the curve marker is moved; the label value is re-evaluated every time the graph is repainted. (Sometimes it is necessary to force a repaint to get labels with symbolic values or/and expressions to update. You can do this by moving another window over the graph or adjusting the size of the window slightly)

Some properties return the ID of another graph object. For example the Curve property returns the ID of the curve to which it is attached. These can be used to access properties of the referenced object. This is done by appending with a ':' followed by the referenced object's property name. For example %curve:label% returns the label property of the curve attached to the curve marker.

This indirect access to graph object properties can be nested to any level although there is probably no good reason for any more than two levels. %curve:xaxis:label%, for example has two levels; %curve% returns the ID of a curve, then %curve:xaxis% returns the id of an axis then %curve:xaxis:label% returns the label property belonging to the axis.

Full documentation is available in the script reference manual, Chapter 7, Graph Objects. This lists the available objects and their property names. There is also a sub-heading titled Symbolic Values that explains the above.

However, deducing all the different possibilities for symbolic values, especially the indirect values, requires some effort. For this reason, the following table has been compiled which lists a range of complete symbolic values that are meaningful for use in labels for various objects. This list is not exhaustive, but probably has everything that is useable.

Note that the symbolic variable names, like everything in SIMetrix, are **not** case sensitive.

Table showing variables, descriptions and contexts

Variable	Description	Can use with
%curve:label%	Curve's label	Curve marker
%curve:shortlabel%	Curve's label without the groupname suffix that is sometimes displayed	Curve marker
%curve:xunit%	Curve's x-axis units	Curve marker
%curve:yunit%	Curve's y-axis units	Curve marker
%x1%	Curve's x-value at curve marker	Curve marker
%y1%	Curve's y-value at curve marker	Curve marker
%curve:xaxis:label%	Curve's x-axis label	Curve marker
%curve:yaxis:label%	Curve's y-axis label	Curve marker
%curve:measurements%	Any measurements assigned to the curve	Curve marker
%graph:maincursor:x1%	x-position of main cursor	Anything
%graph:maincursor:y1%	y-position of main cursor	Anything
%graph:refcursor:x1%	x-position of ref cursor	Anything

%graph:refcursor:y1%	y-position of ref cursor	Anything
%graph:grouptitle%	Title of initial data group. This is actually the netlist title (first line) and for schematic simulations will be the full path of the schematic.	Anything
%graph:sourcegroup%	Data group name that was current when first curve added to graph. E.g. 'tran1', 'dc5' etc	Anything
%curve1:label%	Label for curve attached to crosshair 1	Dimension
%curve2:label%	Label for curve attached to crosshair 2	Dimension
%curve1:shortlabel%	Curve1's label without the groupname suffix that is sometimes displayed	Dimension
%curve2:shortlabel%	Curve2's label without the groupname suffix that is sometimes displayed	Dimension
%curve1:xunit%	Curve1's x-axis units	Dimension
%curve2:xunit%	Curve2's x-axis units	Dimension
%curve1:yunit%	Curve1's y-axis units	Dimension
%curve2:yunit%	Curve2's y-axis units	Dimension
%date%	Date when object created	Textbox, free text, caption, legend box
%time%	Time when object created	Textbox, free text, caption, legend box
%version%	Product name and version	Textbox, free text, caption, legend box

Expressions

Graph object labels may contain expressions enclosed in curly braces. These will be evaluated and the result of the evaluation replaces the complete expression and curly braces. Any script function may be used although only a subset are applicable.

The function 'cv()' is particularly useful. (Needs version 5.5; for earlier versions use 'GetCurveVector()'). cv() returns the data for a curve and you can use this with functions that return a scalar from a vector to attach measurements to curve markers or cursors. Use %curve% as the argument for cv(), i.e. cv(%curve%).

For example, this will return the RMS value for the curve attached to a curve marker:

```
{RMS1(cv(%curve%))}
```

For crosshair dimension objects (the cursor dimensions) use %curve1% or %curve2%.

The Truncate() function is useful if you want to display a measurement applied to a range marked out by the cursors. So the following example will return the RMS value of the curve attached to a curve marker between the range marked out by the cursors.

```
{RMS1(Truncate(cv(%curve%), %graph:refcursor:x1%,
 %graph:maincursor:x1%))}
```

You can also use string functions. For example, %graph:title% usually returns the pathname of the schematic. (This is not guaranteed - but this will always be the case if the schematic has been saved and was run using the regular menus). You can use the SplitPath function to obtain just the file name. E.g.:

```
{(splitpath('%Graph:GroupTitle%'))[2]}
```

You can use the above in any object including free text, text boxes and captions. (Captions are identical to free text, they just have a different default position and font).

The numeric functions above will usually result in a display with more significant digits than desirable. To format the result with less accuracy, use the FormatNumber() function. For example:

```
{FormatNumber(RMS1(cv(%curve%)), 5)}
```

Will display the result to 5 digits.

Appendix F. - Further Reading

The following references provide further information on circuit simulation

1. The SPICE Book. Andrei Vladimirescu, ISBN 0-471-60926-9

   This book provides a good general introduction to the SPICE program on which the SIMetrix core algorithms are based.
2. Designing Analog Chips. Hans Camenzind. ISBN 978-1-58939-718-7

   Although the main subject is about designing integrated circuits, this book is nevertheless useful for general analog design using simulation. This book can be downloaded for free at http://www.designinganalogchips.com
3. The Designer's Guide to SPICE and Spectre. Kenneth S. Kundert. ISBN 0-7923-9571-9

   A more advanced reference on using circuit simulators and assumes familiarity with the operation of simulation programs.