Waveform Calculator User Guide

Product Version 5.1.41
December 2005

Cadence Design Systems, Inc., 555 River Oaks Parkway, San Jose, CA 95134, USA

Contents

1

Overview

The Waveform Calculator is a scientific calculator with both algebraic and Reverse Polish Notation (RPN) modes. You can use it to

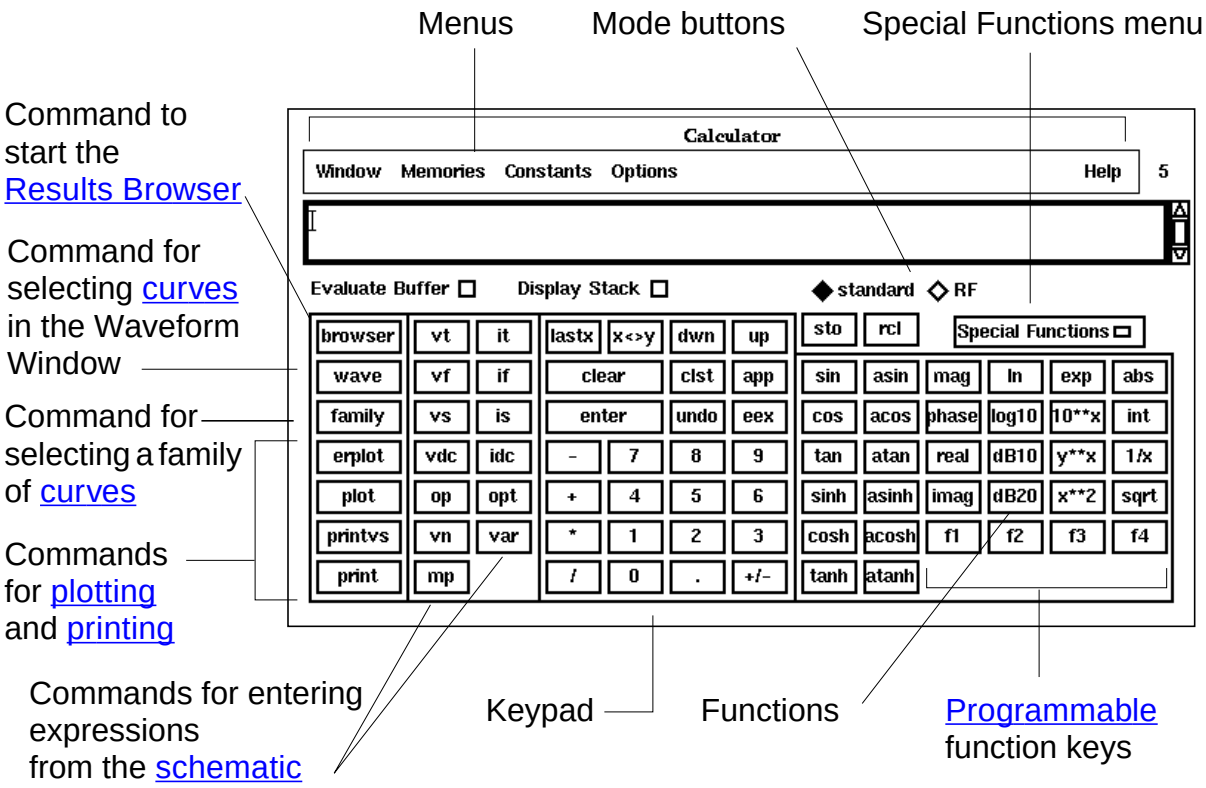
- Build, print, and plot expressions containing your simulation output data
- Enter expressions, which can contain node voltages, port currents, operating points, model parameters, noise parameters, design variables, mathematical functions, or arithmetic operators, into a buffer
- Store the buffer contents into a memory and then recall the memory contents back into the buffer
- Save calculator memories to a file and load those memories back into the calculator

The help screens for the calculator assume that you are familiar with the syntax for either algebraic or RPN scientific calculators. If you have trouble entering expressions, you might want to review the manual for your own handheld scientific calculator.

Each of the calculator functions has a corresponding SKILL command. For more information about SKILL calculator functions, refer to the *OCEAN Reference*.

About the Calculator

The calculator has several kinds of buttons. For help on any button or area, click the labels below.



About the Algebraic and RPN Modes

The calculator has both algebraic and Reverse Polish Notation (RPN) modes. The calculator uses the syntax rules you would expect from any scientific calculator.

To change modes

➤ Choose *Options – Set Algebraic*.

or

➤ Choose *Options – Set RPN*.

Using Keys

Most keys and functions are available in both modes. The following keys are unique to either the algebraic or RPN mode:

Key	Mode	Key	Mode	Key	Mode
(,), and ()	Algebraic	lastx	RPN	app	RPN
space	Algebraic	x<>y	RPN	enter	RPN
=	Algebraic	dwn and up	RPN	clst	RPN

Click here for help using the buffer and stack in RPN mode.

Entering Constants

To enter a constant into the buffer

➤ Select the constant from the Constants menu.

In Algebraic mode, the constant is added to the right side of the buffer.

In RPN mode, the current buffer expression is pushed onto the stack, and the specified constant is placed in the buffer.

The constants are pi, twoPi, sqrt2, degPerRad, charge, boltzmann, and epp0. These constants are internally defined in the Analog Expression Language (AEL) for expression evaluation. Refer to the *Analog Expression Language Reference*.

Note: Calculator expressions should be entered with the right syntax. In Calculator expressions, 2k, 2p etc. are interpreted as 2 kilo (2000) and 2 pico(2e-12) respectively. Expressions like ’2 multiplied by p’ should be entered as ’2*p’ and not as ‘2p’. Also, any string of characters (and not integers) following say ’2p’, e.g. ’2phfyhff’ will be treated as ’2p’ only without any syntax errors. However, expressions like ’2pghg45jk’ will throw syntax errors. The same caution also applies, while using any constant like ’pi’. If the user wants to use any expression like ’4 multiplied by pi’, they should use ’4*pi’ and not ’4pi’ (which will be evaluated as 4p = 4e-12).

About Standard and RF Modes

The calculator has standard and RF modes. In RF mode, the calculator keypad provides mathematical functions commonly used in RF circuit design. This document describes both the standard and RF modes.

Window Size and Location

You can customize the window size and location of the calculator.

➤ To set the default size of the calculator window, add the following command to your ~/.cdsinit file:

```
armSetCalc('defaultSize width:height)
```

The width and height are in pixels (integers) and default to 490:275.

➤ To set the default location of the calculator window, add the following command to your ~/.cdsinit file:

```
armSetCalc('defaultLocation x:y)
```

X and Y are integer screen coordinates with a default location is 0:0.

Calculator Bindkey

You can define a bindkey for the calculator.

➤ Type the following in the CIW:

```
hiSetBindKey( "encap" "<Key>x" "calCalculatorFormCB()" )
```

Substitute the key you want to use for the x.

Starting the Calculator

There are several ways to start the calculator:

- From the Waveform Window or Simulation window, choose *Tools – Calculator*.
- From the Schematic Window choose *Sim-Tools – Calculator*.
- From the CIW, choose *Tools – Mixed Signal Environment – Calculator*.
- For Analog Environment,

- From the CIW, choose *Tools - Analog Environment - Calculator.*
- From Analog Design Environment Window, choose *Tools - Calculator.*

Closing and Quitting the Calculator

To close the Calculator window, but preserve the contents of the buffer and stack

➤ Choose *Window – Close*.

To close the Calculator window and clear the buffer and stack

➤ Choose *Window – Quit*.

Use this command if you need to reinitialize the calculator.

2

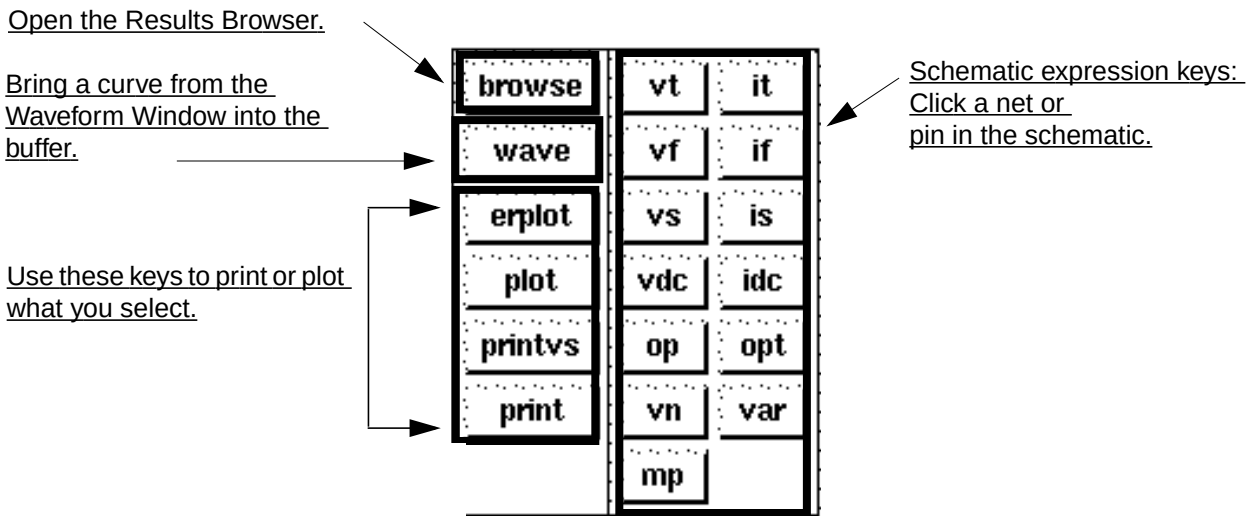
Using the Calculator

Selecting Data

There are three ways to bring simulation results into the calculator. You can

- Use the schematic expression keys to click nets and pins in the schematic and select their results
- Use the Results Browser to select results out of the UNIX file system hierarchy
- Use the *wave* command to select a curve in the Waveform Window

For more help, click the highlighted text in the figure below.



Selecting Data in a Schematic Window

The schematic expression keys let you enter data into the calculator buffer by selecting objects in the Schematic window.

Note: To use the *vn*, *var*, *op*, *opt*, or *mp* functions, you must either select results or have just run a simulation.

vt	transient voltage	*it*	transient current
vf	frequency voltage	*if*	frequency current
vs	source sweep voltage	*is*	source sweep current (I vs V curves)
vdc	DC voltage	*op*	DC operating point
vn	noise voltage	*opt*	transient operating point
var	design variable	*mp*	model parameter

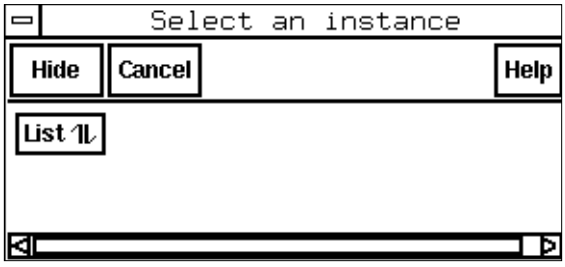
1. Click a schematic expression key.
2. Click the appropriate object in the schematic.

   If more than one parameter is available for the expression and instance you picked, a form appears. Select the parameter you want from the List field and click *OK*.
3. When you have finished selecting objects, press the *Esc* key while the cursor is in the Schematic window.

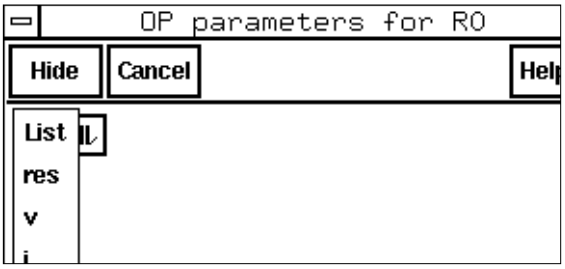
Choosing Parameters from Schematic Data

To select a parameter in the schematic with a schematic expression key

1. Click an instance in the schematic.



2. Choose the parameter you want from the List field.



Note: When you use the *op*, *opt*, *mp*, *vn*, or *var* functions, you must have just run a simulation, or you must choose *select results* from the *Results* menu in the Simulation window. Otherwise, the system does not know what to display.

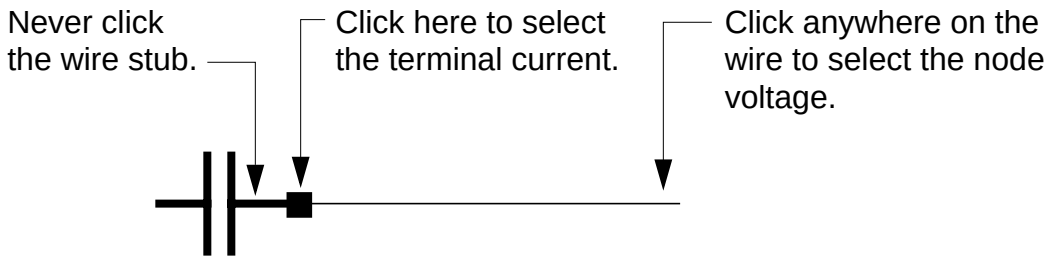
Choosing Voltages or Currents

To select voltages in the schematic

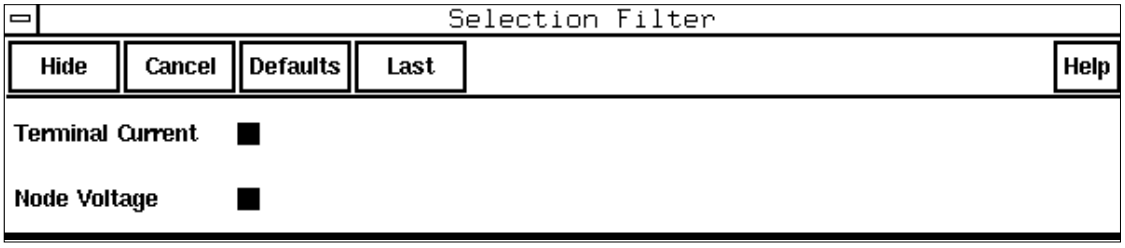
➤ Click wires.

To select currents

➤ Click square pin symbols, not wires.



You can use the Selection Filter form to restrict selection to either pins or wires. Press *F3* if the Selection Filter form did not appear.



Selecting Curves in the Waveform Window

Use the wave key to create an expression from a curve in the Waveform Window and place the expression into the calculator buffer.

1. Click *wave* in the calculator.
2. Click a curve in a Waveform Window.

   The waveform expression that the system enters in the calculator is the expression on the Waveform Window status banner at the tracking cursor location.

   If the banner expression cannot be evaluated to a waveform (because it is only a descriptive title), the system automatically creates a Cadence® SKILL language function to represent the waveform you selected.

Selecting a Parametric Set of Curves

To select a parametric set of curves in the Waveform window

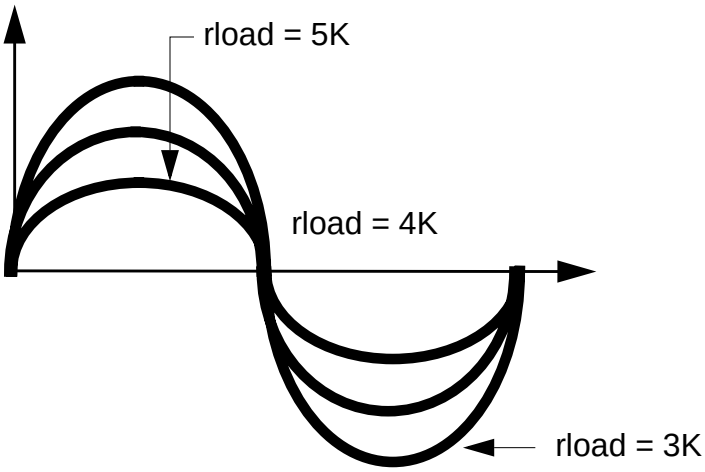
1. Click *family* in the calculator.
2. In the Waveform window, click a curved part of the parametric wave.

Isolating One Curve from a Waveform Group

You can select a single curve from a parametric curve group in the Waveform Window. This lets you perform calculations on the single curve or display it separately on another set of axes.

1. In the Calculator, click *wave*.

2. Click a parametric curve in the Waveform Window.



An expression representing the individual curve appears in the calculator buffer.

3. In the Calculator, click *erplot*.

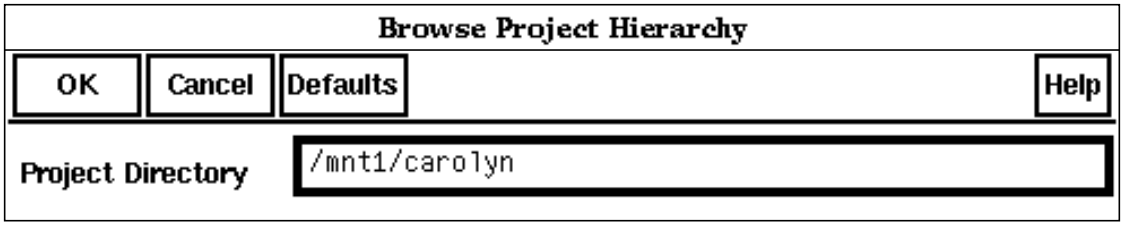
The Waveform Window is erased, and the single curve appears.

Selecting Simulation Results with the Results Browser

To select waveform expressions from simulation output data in the Results Browser

1. Click *browse* in the calculator.

The Browse Project Hierarchy form appears.



2. Enter the path to the project directory in the form and click *OK*.

The Results Browser appears.

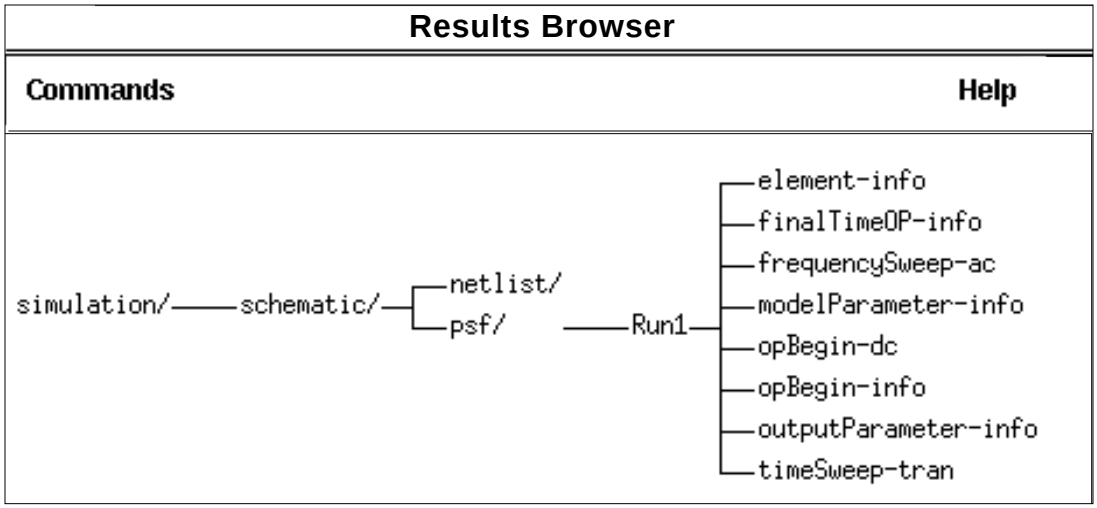
3. Click left in the Results Browser to expand the data hierarchy until you get to the scalar or waveform data you want to enter in the calculator buffer.

4. Click the scalar or waveform data to enter the expression for that data in the calculator buffer.

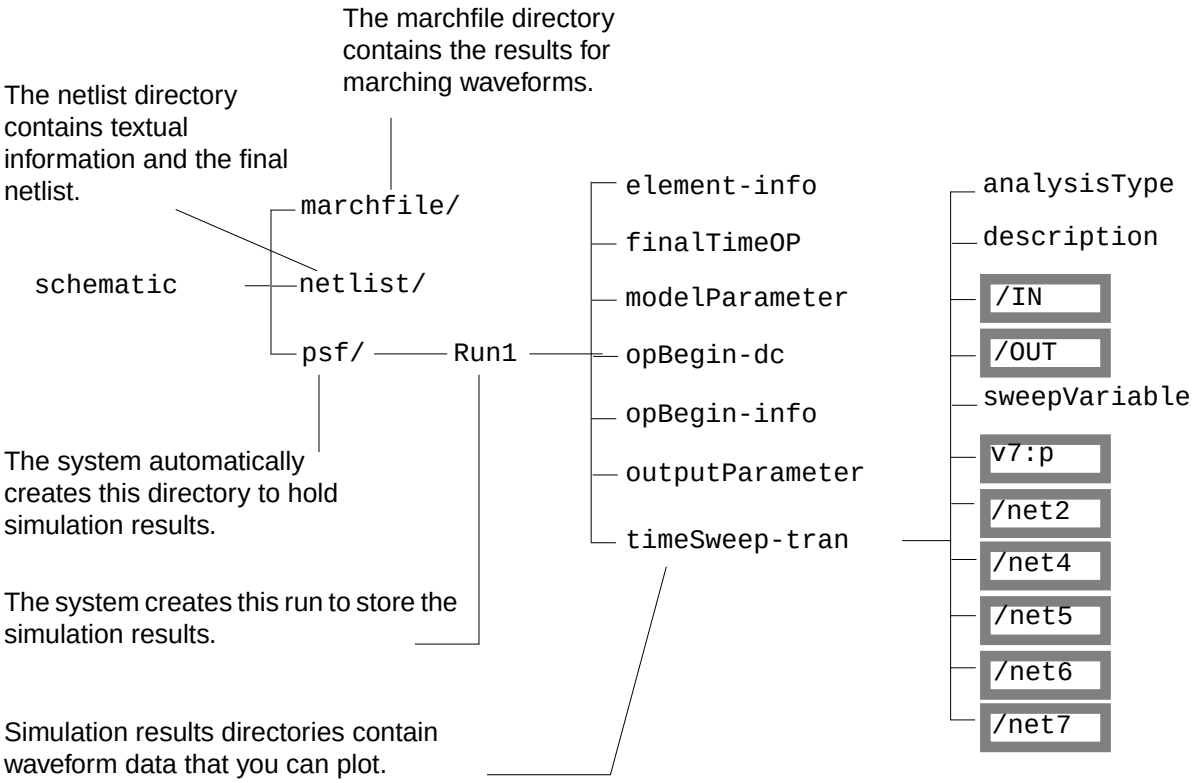
Using the Results Browser

The Results (Data Results library) Browser lets you

- View simulation waveforms and text results
- Plot waveforms
- Copy the waveform expression for simulation results directly into the Waveform Calculator



The Results Browser stores data hierarchically in objects called nodes. You can expand a particular node down to the lowest level of the hierarchy. The following figure shows the file structure for simulation results in the Results Browser.



The contents of the `psf` directory varies depending on the simulator you use. This directory might contain nodes such as the following:

Node	Contents
`element`	Component parameters for design circuit elements
`finalTimeOP`	Operating point of the component parameters at the end of the simulation
`modelParameter`	Simulation model parameters
`opBegin-dc`	Voltage of the node at T=0

Node	Contents
`opBegin-info`	Simulator parameters at T=0
`outputParameter`	Information about the output parameters
`srcSweep`	Waveform data for the DC analysis
`timeSweep`	Waveform data for the transient analysis
`frequencySweep`	Waveform data for the AC analysis

Simulation results also include the following files:

Node	Contents
`analysisType` and `description`	Information about the type of analysis
`sweepVariable`	Information about the swept variables

The simulation results, which are stored in directories such as `timeSweep-tran`, might be scalar data (numeric data) or waveform data, which is highlighted.

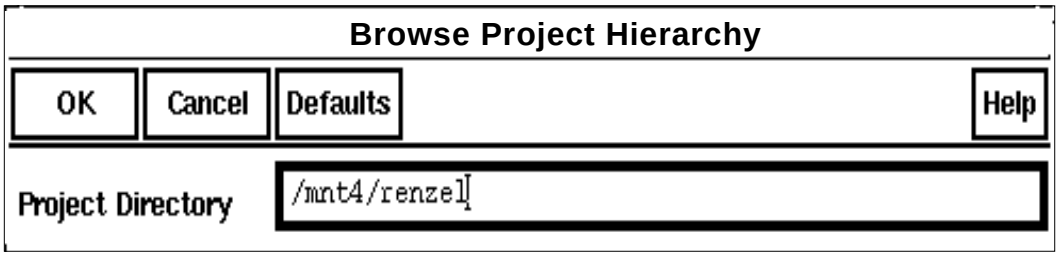
If you press the middle mouse button over a node, a menu of commands pops up. You can use these commands to perform actions on that node. The Results Browser also has a Commands menu in the banner that helps you manipulate the display of data and lets you close the Results Browser.

Starting the Browser

1. To start the Results Browser, do one of the following:
   - Click *Browse* on the Waveform Calculator form
   - Choose *Tools – Results Browser* in the Waveform Window
   - Choose *Tools – Mixed Signal Environment – Results Browser* in the CIW.

- For Analog Environment, from the CIW, choose *Tools - Analog Environment - Results Browser.*
- For Analog Environment, from Analog Design Environment Window, choose *Tools - Results Browser.*

The Browse Project Hierarchy form appears.



2. Type in the path of a project directory containing simulation output files.

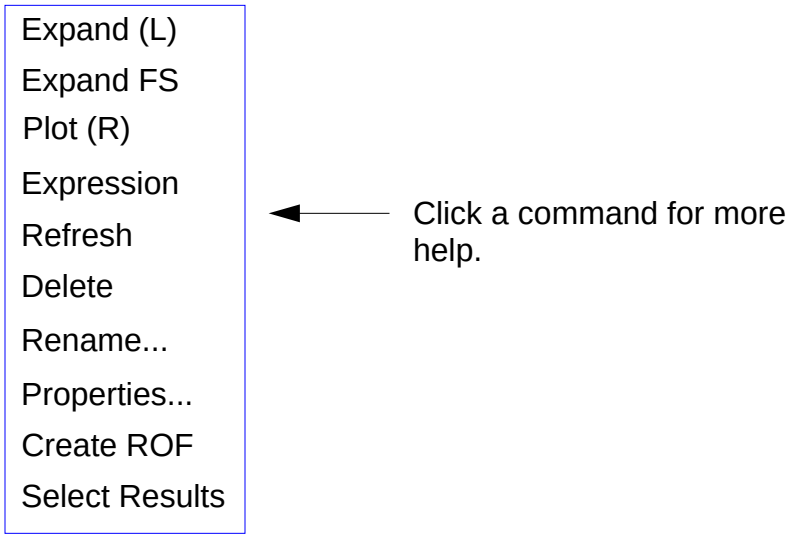
If you just finished a simulation, the current data directory is shown. Otherwise, the form defaults to your home directory.

3. Click *OK*.

The Results Browser appears.

The Results Browser has a pop-up menu that lets you move through the file system and data hierarchy, perform commands on the file system and data, and display various properties of the data hierarchy.

➤ To display the Results Browser pop-up menu, press the middle mouse button over a node.



Managing the Display

The commands on the Results Browser menu help you manage the display of data.

To place the root (the vertical list of nodes on the left) of the Results Browser hierarchy in the middle of the window

➤ Choose *Commands – Root*.

After you use the scroll bar to pan through the Results Browser nodes, you can use this command to return to the default configuration of the Results Browser.

To move up a level in the Results Browser hierarchy

➤ Choose *Commands – Unexpand*.

To redisplay the complete Results Browser hierarchy after you add new files to the file system (with the browser open)

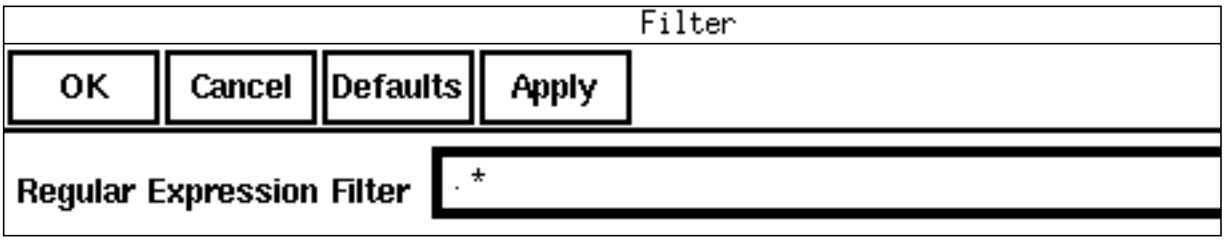
➤ Choose *Commands – Refresh*.

You can also use this command to redraw the current Results Browser display.

To limit the number of net, instance, and terminal elements displayed

1. Choose *Commands – Filter*.

The Filter form appears.



2. Enter a regular expression filter.

   This command is useful for simplifying the Results Browser display while browsing simulation results for large circuits.

To keep the most recently expanded node in the center of the window

➤ Choose *Commands – Centering On*.

If you do not want the most recently expanded mode to move automatically to the center of the window, you can turn off the centering mode.

➤ Choose *Commands – Centering Off*.

You can also use an environment variable, `browserCenterMode`, in your `.cdsenv` file to turn centering on or off. To turn centering on, add the following line to the `.cdsenv` file.

```
asimenv.misc browserCenterMode boolean t
```

To turn centering off (which is the default) add the following line to the `.cdsenv` file.

```
asimenv.misc browserCenterMode boolean nil
```

To close the Results Browser window

➤ Choose *Commands – Close Browse*r.

Redrawing the Browser

To redraw the Results Browser display for the current level in the Results Browser hierarchy

1. Place the pointer over the current level in the Results Browser.
2. Press the middle mouse button and choose *Refresh* from the pop-up menu.

Note: To refresh the entire Results Browser hierarchy, choose *Commands – Refresh* on the Results Browser menu.

Configuring the Browser

You can customize the size and location of the Results Browser window.

➤ To set the default size of the Results Browser window, add the following line to your *~/.cdsinit* file:

```
armSetBrowser('defaultSize width:height)
```

The width and height are in pixels (integers), and they default to 800:500.

➤ To set the default location of the Results Browser window, add the following line to your *~/.cdsinit* file:

```
armSetBrowser('defaultLocation x:y)
```

x and y are the screen coordinates (integers) of the location you want, and they default to 0:0.

Note: To customize the Results Browser for the current session only, enter these commands on the input line of the CIW.

Expanding Nodes

In the Results Browser, a node represents an object that contains information. Scalar data (numeric data) and waveform objects are highlighted.

➤ To expand a node, click left on it.

- ❑ If you click a file node, the file contents appear in a file viewing window. You can use the window functions to manipulate the file.
- ❑ If you click a directory node, the directory contents appear in the next level of the Results Browser hierarchy. If the directory contains simulation data, the data files open and the simulation data hierarchy expands.
- ❑ If you click a highlighted object, the scalar data or the expression for the waveform object is entered into the calculator buffer for mathematical processing.

Note: You can also expand a node by choosing the *Expand (L)* command from the Results Browser pop-up menu.

To see an example of an expanded hierarchy of nodes containing simulation results, click here.

You can use the *Expand FS* command from the Results Browser pop-up menu to expand only the UNIX file system—not the psf hierarchy that the system uses to display simulation results.

To expand the UNIX file system for a node

➤ Press the middle mouse button over the node and choose *Expand FS*.

The UNIX file system expands to show any directories or files under the node.

Plotting Waveforms

You can plot several waveforms from the same or different simulations in the Waveform Window.

To plot a waveform object in the Waveform Window

1. Expand the Results Browser nodes until you see highlighted waveform data.

   Waveform data is in the psf directory, which is under the simulation results for the simulator you chose.

2. Click right over the waveform data.

   The waveform data is plotted in the Waveform Window. When you plot a waveform this way, it is added to the existing waveforms in the Waveform Window.

   To erase the existing waveforms and plot only the selected waveform, use the *Window – Reset* command in the Waveform Window before you click right on the node to plot.

Note: You can also use the *Plot (R)* command from the Results Browser pop-up menu to plot waveform objects in the Waveform Window.

Copying Expressions into the Calculator

To copy the expression for a Results Browser object to the calculator buffer

➤ Click left over a highlighted object in the Results Browser.

The expression for the object appears in the calculator buffer.

Note: You can also use the *Expression* command from the Results Browser pop-up menu to enter expressions in the calculator buffer.

You can use this command to enter expressions for

- Operating points
- Model parameters
- Noise parameters
- Waveform data objects
- Scalar data

Deleting UNIX Files

To delete a file or directory from the UNIX file system

1. Place the pointer over the file or directory you want to delete.
2. Press the middle mouse button and choose the *Delete* command from the pop-up menu.

   A dialog box asks if you want to delete the file or directory.

   A warning message appears in the CIW if the node you select is not part of the file system.
3. Click *OK* on the dialog box.

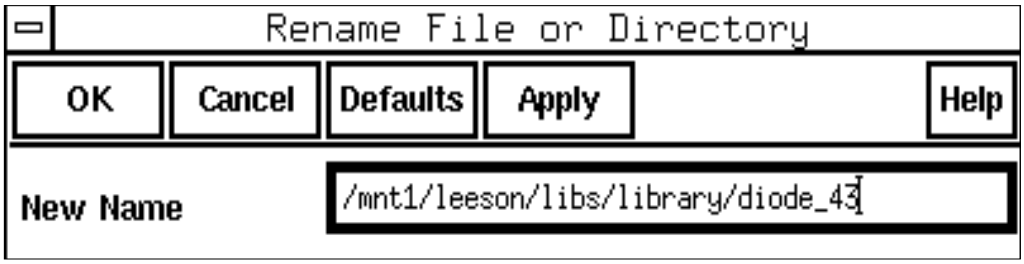
   The directory or file is deleted from the file system.

Renaming UNIX Files

To rename a file or directory in the UNIX file system

1. Place the pointer over the file or directory you want to rename.
2. Press the middle mouse button and choose the *Rename* command from the pop-up menu.

   The Rename File or Directory form appears.



3. Enter the new file or directory name and click *OK*.

Displaying Properties

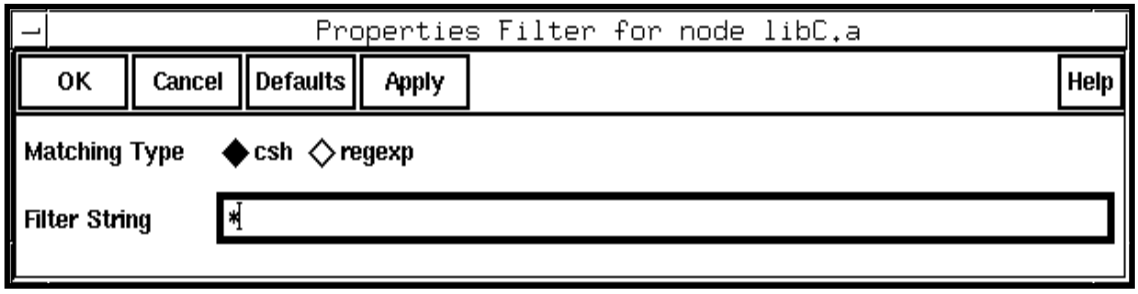
You can view the properties associated with the simulation output data hierarchy and filter the properties so that only the simulation properties for a particular type of component appear. (A property is an associated value, such as a DC operating point.)

1. Place the pointer over a simulation output data object, which is any node under the *psf* directory.

   Simulation data is in the psf directory, which is under the simulation results for the simulator you chose.

2. Press the middle mouse button and choose the *Properties* command from the pop-up menu.

   The Properties Filter form appears.



3. Choose a Matching Type.

   Matching Type lets you choose the type of pattern matching.

   - **csh** uses the same pattern-matching tools used by the UNIX C-shell to match file names. This is the default.
   - **regexp** uses the standard UNIX regular expression pattern- matching syntax.

4. Change the filter string.

   The default filter string, which is an asterisk (*), matches everything. You can change the default to match any string you want. For example, to match all the resistors in a schematic with the csh matching mechanism, set the filter string to R*.

5. Click *OK* on the form.

   The results appear in a text window.

Using External Data

You can use the *Create ROF* command to create a run object file, which lets you access external data generated by a standalone simulator. (When you use the Analog Artist simulation environment to run simulations, this file is generated automatically for you.)

To create a run object file

1. Place the pointer over a directory that contains the external simulation data.
2. Press the middle mouse button and choose the *Create ROF* command from the pop-up menu.

   A run object file called runObjFile is created in the next level of the hierarchy.
3. Click left on the node containing the simulation results to expand it.

Note: You cannot see the run object file unless you use the *Expand FS* command to expand the file system.

Selecting Results

You can link data from a previous simulation to a schematic window. Then you can probe the schematic to compare the results of the previous simulation to current results.

1. Place the pointer over a node containing simulation results.
2. Press the middle mouse button and choose *Select Results* from the pop-up menu.

   The schematic window is redrawn and the CIW tells you that the simulation results for the node are selected.

Note: You can also use the *Results – Select Results command* from the Simulation window.

Plotting or Printing Results

You can plot or print the value of the calculator buffer expression against an independent variable.

You can plot or print only the expressions that are in the buffer, not the memories. You must recall memory expressions into the buffer to plot or print them.

You can print or plot multiple expressions by separating each expression by a space using the *app* (append) function in RPN mode or the space key in algebraic mode. For example,

```
expr1 expr2 expr3
```

Plotting Expressions

To erase the Waveform Window and plot the buffer expression

➤ Click *erplot* in the calculator.

To plot the buffer expression without first erasing the Waveform Window

➤ Click *plot* in the calculator.

For example, to plot the I vs. V curve after a DC source-sweep analysis

1. In the calculator, click *IS*.
2. In the schematic, click the output terminal of the device.

   Terminals are the square symbols at the end of the wire stub.

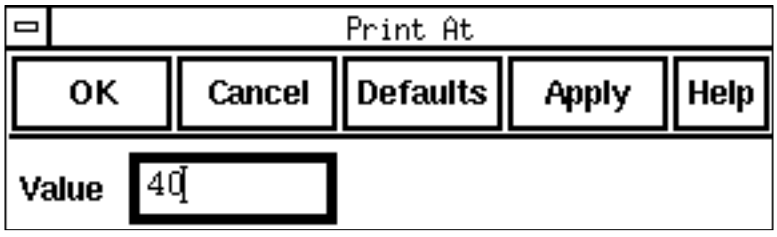
   Now you have an expression in the buffer for the IV curve.
3. Click *erplot* in the calculator.

   The system opens a Waveform Window (unless one is already open) and draws the curve.

Printing One Expression Value

The *print* command prints the value of an expression at a single value of the independent variable.

1. Put the expression into the calculator buffer.
2. Click *print*.
3. If the expression is a waveform, enter the value of the independent variable.



The Results Display Window appears, displaying the results in numerical format.

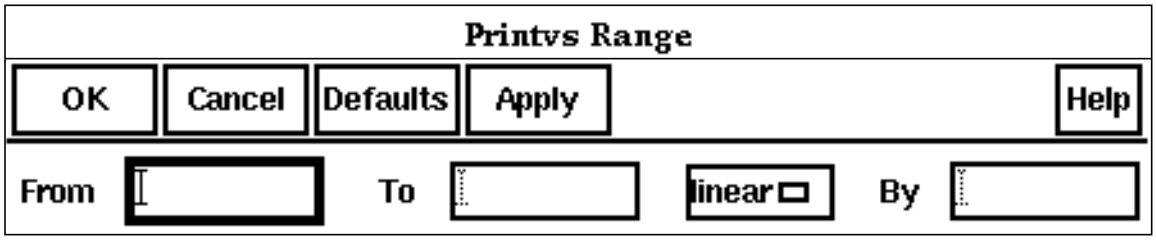
Note: If the expression in the calculator buffer represents a parametric waveform, then the *print* button prints the value of the expression at the specified point for different sets of sweep parameters.

Printing a Range of Expression Values

The *printvs* command prints a table showing the value of the buffer expression over a range of independent variables.

1. Put the expression into the calculator buffer.
2. Click *printvs*.

   The Printvs Range form appears.



3. Type in the starting and ending values.

   Leave all of the fields blank to print the raw simulation data.
4. (Optional) Choose a linear or logarithmic range.
5. Enter the increment for the calculation in the By field.

Note: When results related to signals represent multi-bit buses, the *printvs* button prints the data corresponding to each bit. For example, if you use Calculator to print `VDC("/x<2:4>")`, where `x<2:4>` is a sub-bus of the bus `x<0:5>` and `x<2>=3`, `x<3>=3.5`, and `x<4>=4`, you would get:

```
bit     VDC("x<2:4>")
```

```
0       3
1       3.5
2       4
```

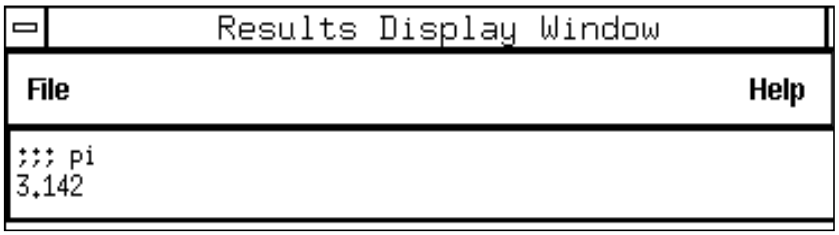
For the bus represented as x<2:4>, the swept parameter *bit* varies from 0 to the length of the bus/sub-bus, which is 2 in this example. So, bit0 is x<2>, bit1 is x<3>, and bit4 is x<4>.

Setting the Range Calculation Increment

You can set the range calculation increment in the Printvs Range form in several ways:

- If the range is linear, enter the increment.
- If the range is logarithmic, enter the number of points per decade.
- To print the raw simulation data, leave all three fields blank.

  The result of the evaluation is displayed in the Results Display Window.

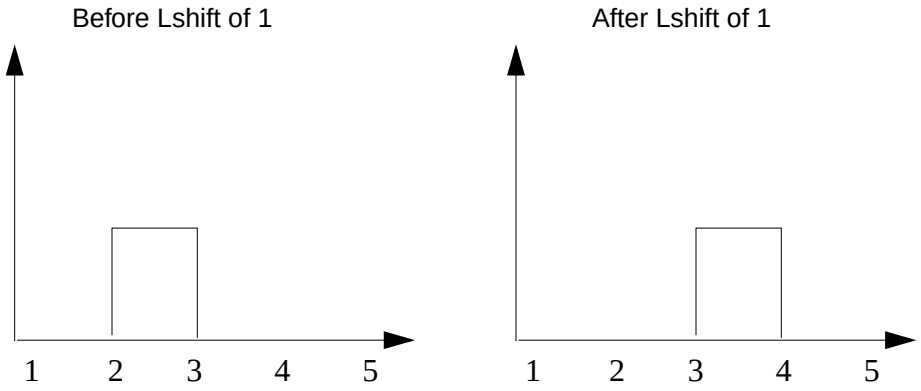


Note: If you specify range values outside the range of the original simulation data, the calculator extends the simulation endpoint values to the requested values using linear extrapolation, which can give misleading results.

Applying Waveform Transformations

Shifting the X Axis (L shift)

The *lshift* function shifts the X axis in the Waveform Window to the left by a specified amount. Use negative values to shift the X axis to the right.

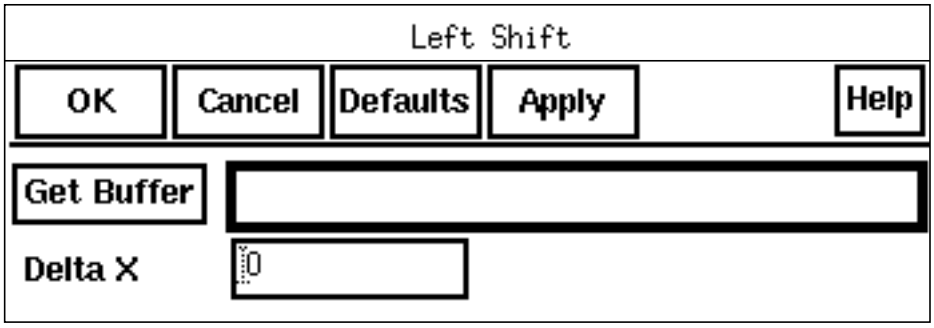


In Algebraic Mode

In algebraic mode you set up the waveform expression after selecting the *lshift* function.

1. From the Special Functions menu, choose *lshift*.

   The Left Shift form appears.



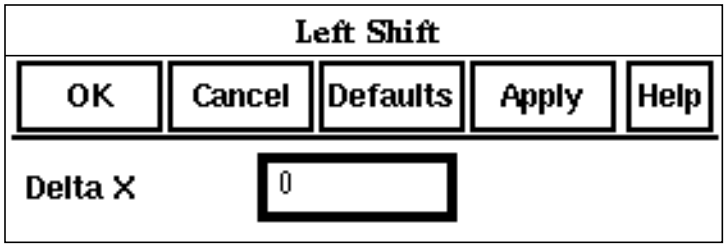
2. Enter a value in X axis units in the Delta X field, and click *OK*.

In RPN Mode

In RPN mode the *lshift* function acts on the expression already in the buffer:

1. From the Special Functions menu, choose *lshift*.

   The Left Shift form appears.

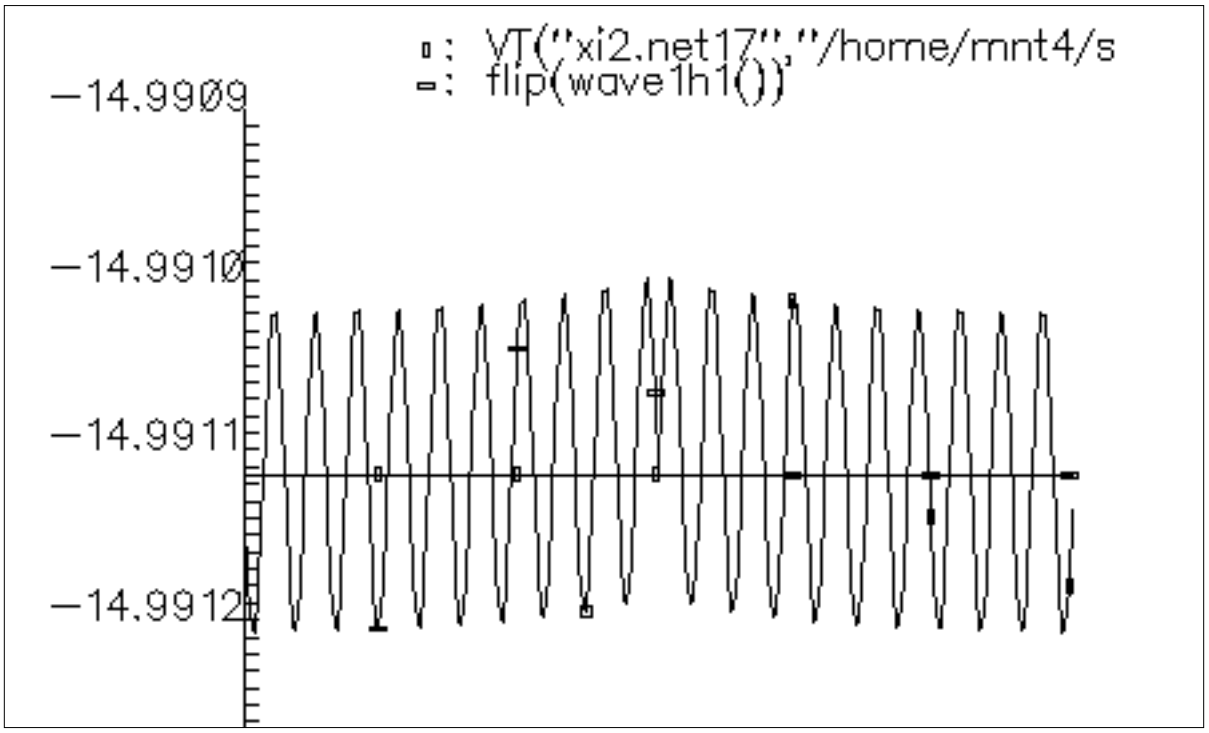


2. Enter a value in X axis units in the Delta X field and click *OK*.

Y Axis Flip

To flip a waveform around the y axis

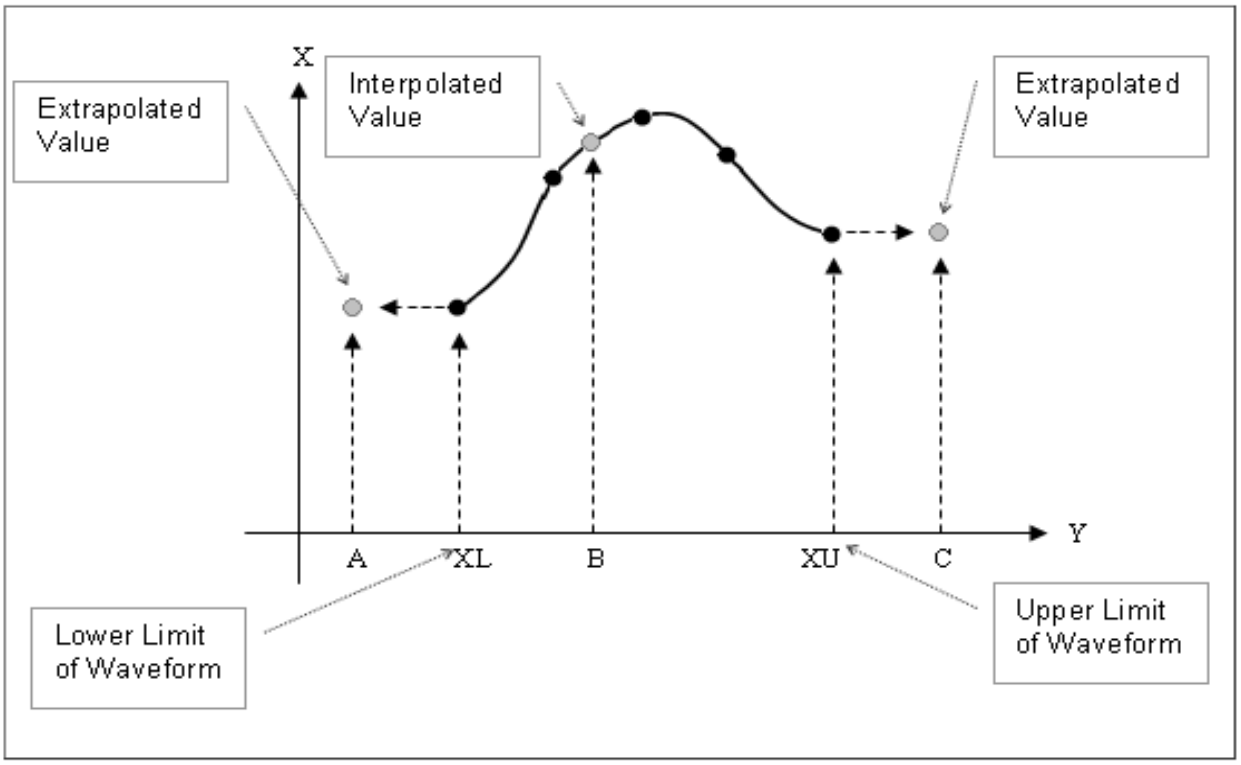
➤ From the Special Functions menu, choose *flip*.



Extrapolating and Interpolating Values

When a point on the X-axis is within the range of the waveform, its Y value is interpolated using first order linear interpolation. When a point is beyond the range of a waveform, constant extrapolation is applied to determine its Y-value. In this case, the Y-value is taken to be the same as that of the closest point on the waveform (the starting point or the end point).

In the diagram below, the existing points in the waveform are marked with black dots, with the lines marked XL and XU pointing to the lower and upper limits, respectively, of the waveform. The vertical dotted lines A and C show examples of values being extrapolated. The vertical dotted line B shows an example of a value being interpolated. The extrapolated and interpolated points are marked by gray dots.



Using Memories

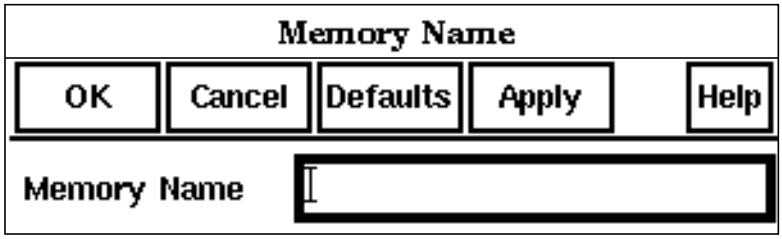
You can store the buffer in a memory and recall it later. You can also save the calculator memories to a file. Use the Memories menu to work with memories.

Storing Memories

To store the current buffer expression in a memory

1. In the calculator, choose *Memories – Store* or click *sto*.

   The Memory Name form appears.



2. Type a name for the memory and click *OK*.

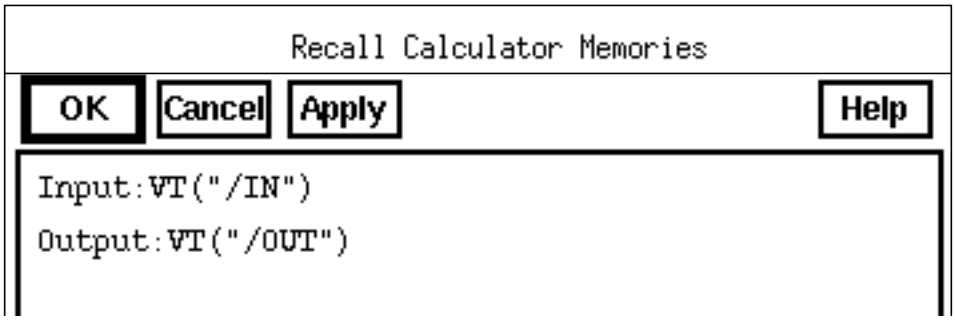
There is no limit on the number of memories you can use.

Recalling Memories

To recall a memory into the buffer

1. Choose *Memories – Recall* or click *rcl*.

   The Recall Calculator Memories form appears.



2. Double click to select an expression, or highlight the expression and click *OK*.

   The recalled expression stays in the calculator memory pool.

In Algebraic mode, the recalled expression is appended to the end of the buffer. The memory *asfmem* stores the buffer temporarily while you enter a special function expression.

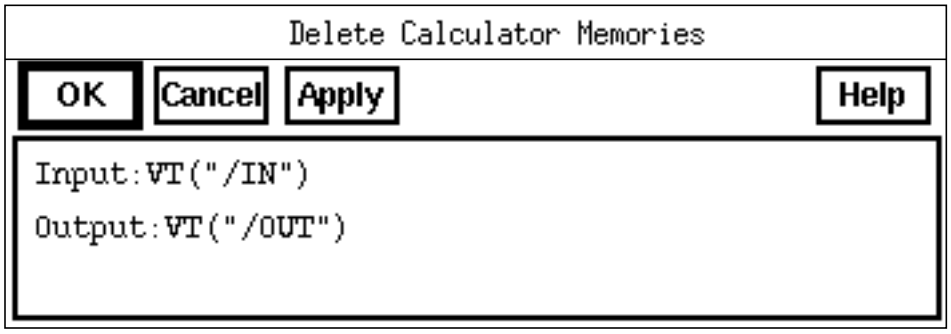
In RPN mode when you recall a memory, any expression currently in the buffer is pushed onto the calculator stack.

Deleting Memories

To delete an expression from the memory

1. Click *Memories – Delete*.

   The Delete Calculator Memories form appears.



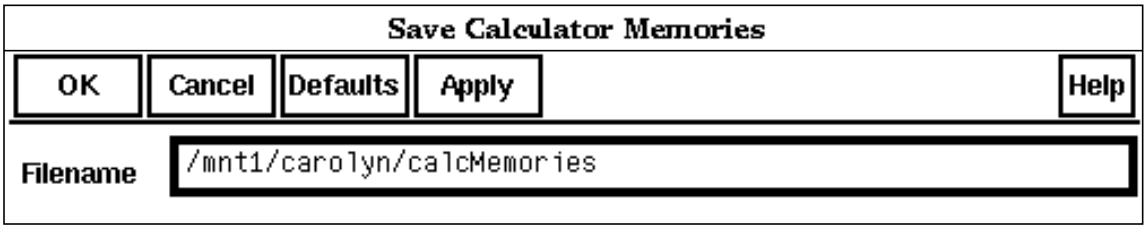
2. Double click to delete an expression, or highlight the expression and click *OK*.

Saving and Loading Memories

Use the *Save* command to write memories to a file. Use the Load command read memories from a saved file.

1. Choose *Memories – Save* or *Memories – Load*.

   The Save Calculator Memories form or the Load Calculator Memories form appears.



2. Enter the path and name for the memory file.
3. Click *OK*.

Defining Functions and Function Keys

Defining New Functions

You can define a function and add it to the Special Functions menu with the following steps.

1. Define the form that prompts for user-defined arguments to the special function.
2. Define the syntax of the special function in the callback procedure.
3. Register the special function.

Defining a Form

The following example shows how to define an input form for a function that takes three arguments. The first argument is the buffer expression. The other two arguments are the boundaries of the range of the expression on which you want to operate.

```
procedure( CreateMyForm()
   let( ( fieldList a b )
    a = ahiCreateStringField(
        ?name 'from
        ?prompt "From"
        ?value ""
    )
    b = ahiCreateStringField(
        ?name 'to
        ?prompt "To"
        ?value ""
    )
    fieldList = list(
        list( a 5:0 120:25 40 )
        list( b 160:0 110:25 30 )
    )
    calCreateSpecialFunctionsForm( 'MyForm
        fieldList )))
```

In this example, the From and To fields are string fields created in a two-dimensional form specification for fieldList. The form is created by the call to calCreateSpecialFunctionsForm. This function creates and registers the form with the specified form symbol, MyForm.

Defining a Callback Procedure

You define a callback procedure that is called from the entry on the Calculator Special Functions menu. Since this example uses a form to prompt for additional information required by the special function, the callback procedure is

```
procedure( MySpecialFunctionCB()
  calCreateSpecialFunction(
        ?formSym 'MyForm
        ?formInitProc 'CreateMyForm
        ?formTitle "Test"
        ?formCallback "calSpecialFunctionInput( 'test
        '(from to) )"
  )
)
```

In this procedure, a call is made to calCreateSpecialFunction, which creates and displays the form and then builds the expression in the buffer with the specified form fields.

Using Stack Registers in the Procedure

You can use the special symbol 'STACK in the list of form fields to get expressions from the stack.

For example, if you want to insert a stack element between the From and To arguments in the special function expression, you could specify the callback line as follows:

```
?formCallback "calSpecialFunctionInput('test '(from STACK to))"
```

If your special function does not require a form to prompt for additional arguments, you can define your callback as follows:

```
procedure( MySpecialFunctionCB()
    calSpecialFunctionInput( 'test nil )
)
```

Registering the Function

You register the function and callback with calRegisterSpecialFunction:

```
calRegisterSpecialFunction(
    list( "test" 'MySpecialFunctionCB )
)
```

The next time you select *Calculator* from the Waveform or Simulation window menu, your new special function appears in the Special Functions menu.

SKILL User Interface Functions for the Calculator

For SKILL Functions of Waveform Calculator, please refer to *chapter 22* of *Virtuoso Analog Design Environment SKILL Language Reference*.

Assigning Function Keys

You can assign buffer and stack manipulation procedures to the four function keys *f1*, *f2*, *f3*, and *f4*. To do this, use SKILL commands that you type in the CIW or add to your .cdsinit file.

For example, you can use the *f1* key to create the expression for the magnitude and phase of an AC waveform in the buffer by defining the following RPN mode procedure:

```
procedure(f1( )
    calCalcInput('(enter phase xchxy mag append))
)
```

This *calCalcInput* function manipulates the buffer containing the expression

```
VF("/net1")
```

to create the expression

```
mag(VF("/net1")) phase(VF("/net1"))
```

3

RPN Mode

This section describes how you use the calculator in RPN mode.

About the Buffer

The buffer area stores data and expressions that you enter for the calculator to manipulate.

You can enter expressions into the buffer in several ways:

- Use the keypad and function keys.
- Read calculator memories.
- Type at the keyboard.

You can edit the buffer contents by backspacing to delete unwanted characters.

Note: If you enter multiple expressions, separate them with one or more spaces.

Using the Keypad, Functions, and Memories

You can select numbers, functions, operators, and constants from the keypad. The calculator automatically adds the selection to the buffer.

The functions on the keypad (such as log) make the entire buffer's contents the argument of the functions. For example, if the buffer contains expr1 and you select log, the following appears in the buffer:

`log(`*`expr1`*`)`

Select a function when there is only one expression in the buffer. if the buffer contains multiple expressions, the space between expressions causes an error, as in the following example:

`log(`*`expr1 expr2`*`)`

To read a calculator memory into the buffer, choose Memories – Recall.

Entering Variables

To enter a variable expression into the buffer

1. Select the *var* button from the Waveform Calculator form.

   The Select an Instance form appears.

2. On the schematic, choose an instance that uses this variable as a parameter value.
3. Choose the variable from the *List* cyclic button on the Select an Instance form.
4. Select *OK*.

   The expression appears in the buffer.

Note: To use the *var* function, you must either select results or have just run a simulation.

Entering Constants

To enter a constant into the buffer

➤ Select the constant from the Constants menu.

The current buffer expression is pushed onto the stack, and the specified constant is placed in the buffer.

The constants are pi, twoPi, sqrt2, degPerRad, charge, boltzmann, and epp0. These constants are internally defined in the Analog Expression Language (AEL) for expression evaluation. Refer to the *Analog Expression Language Reference*.

Entering Multiple Expressions

You can enter more than one expression into the buffer with the app (append) key. Each expression must be separated by a space, which you can enter from the keyboard.

Many functions cannot operate on multiple expressions. Be careful not to enter one of these functions while you have more than one expression in the buffer.

You can build the expressions separately in the stack and then combine them in the buffer with the app (append) key.

Evaluating the Buffer

Evaluating the buffer is useful only for expressions that contain scalar functions or variables. The Waveform Calculator evaluates expressions that contain waveforms or undefined variables as NaN (not a number).

To evaluate an expression in the buffer after the arithmetic operations are performed on it

➤ Select the Evaluate Buffer option.

About the Stack

Like a Reverse Polish Notation (RPN) scientific calculator, the Waveform Calculator uses a buffer and a stack to build and manipulate expressions. One-expression functions operate on the buffer only and leave the contents of the stack unaffected. Multiple-expression functions operate on both the buffer and the stack elements.

An RPN calculator performs arithmetic operations by positioning the expressions in the stack the same way you would on paper. For example, to add *a* and *b*, you write down the two numbers and then perform the addition. Similarly, you always position the expressions for an operation in the calculator in the natural order first, then enter the arithmetic operator.

The stack performs many movements automatically to help you perform long chain calculations.

The stack “lifts” every expression in the stack when a new expression is entered and “drops” expressions into position when you perform multiple-expression operations.

Because operations are performed when the function is executed, the length of the expression being built is limited only by the Cadence® SKILL language string length (maximum 8191 characters). In addition, the stack drops during calculations involving the buffer and the first stack register. This lifting and dropping of the stack lets you retain intermediate expressions in long calculations without having to store the expressions into other memory locations.

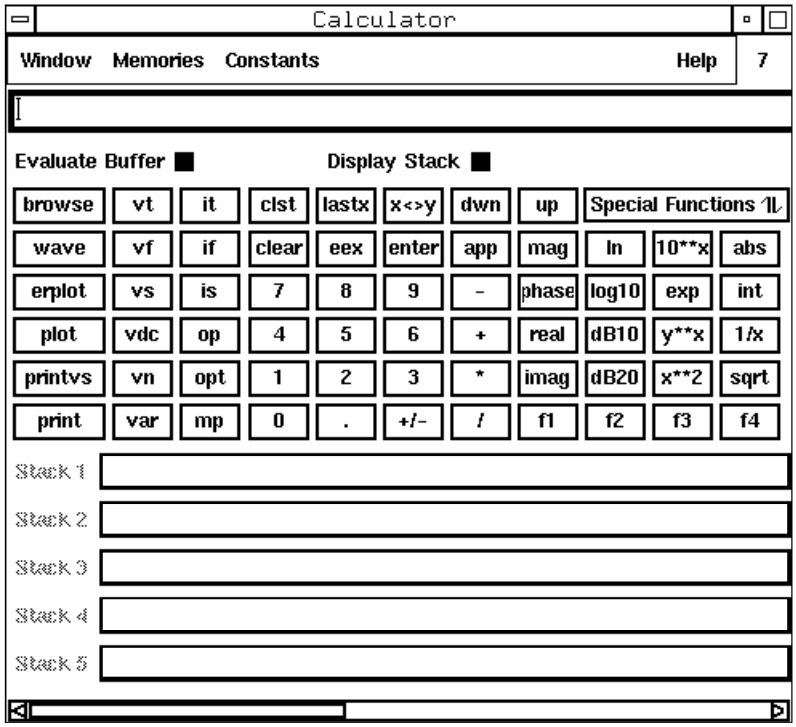
By starting every problem at its innermost set of parentheses and working outward (just as you would with pencil and paper) you maximize the efficiency and power of the RPN calculator logic.

Displaying the Stack

To display the top five elements in the stack

➤ Select the Display Stack option.

You can use this option to help you understand how the Waveform Calculator performs mathematical operations. It lets you look at the first five registers of the stack simultaneously so you can see the effect of a given keypad sequence.



Moving the Buffer and Stack Register

You can move expressions within the stack, and between the stack and buffer, with the *enter*, *dwn*, *up*, and *x< >y* functions.

To recall the previous expression into the buffer, use the *lastx* key.

Pushing Expressions on the Stack

When you make an entry into the calculator buffer, you must tell the calculator that you have finished and are ready to make the next entry.

To push the current expression onto the stack

➤ Click *enter*.

Unless the buffer is empty, *enter* duplicates the contents of the buffer and pushes it onto the stack. All the existing expressions in the stack are pushed down one register. For efficiency, many calculator functions perform an implicit *enter* operation.

Exchanging the Buffer and Stack

Use the up and dwn commands to exchange the contents of the buffer and the stack registers.

To move the contents of the first stack register into the buffer and move the contents of the buffer to the bottom of the stack

1. Select *dwn*.
2. Keep selecting *dwn* to cycle the buffer and stack to their original states.

To scroll the stack in the opposite direction

➤ Select *up*.

To exchange the contents of the buffer and the first stack register without changing the contents of the remaining stack registers

➤ Select *x*< >*y* (x exchange y)

Recalling the Last Expression

The lastx register contains the last expression in the buffer upon which a mathematical operation was performed.

To place the contents of the lastx register into the buffer

➤ Select *lastx*.

The calculator pushes the current buffer expression onto the stack.

The lastx register is most useful in calculations where an expression is used more than once. By recovering an expression using lastx, you avoid manually reentering that expression into the calculator.

For example, to enter the expression (1+x)/x, use the following key sequence (with the evaluate buffer option turned off):

```
1
enter
```

```
clear
x
+
lastx
/
```

Clearing the Buffer and Stack

There are several ways to clear the calculator buffer and stack.

To remove a single character from the buffer

➤ Press the backspace or delete key.

To clear the buffer without affecting the stack

➤ Click *clear* on the calculator.

To clear the buffer and stack

➤ Click *clst* on the calculator.

Operators and Functions

The calculator has both algebraic and Reverse Polish Notation (RPN) modes.

Note: Each of the calculator functions has a corresponding SKILL command. For more information about SKILL calculator functions, refer to the *OCEAN Reference*.

In RPN mode the calculator uses the syntax rules you would expect from any handheld scientific calculator.

For example, to enter the function (1+x)/x in RPN mode, you would use this key sequence

```
1 enter clear x + lastx /
```

Help is also available for

- The buffer and stack
- Algebraic mode

Single-Expression Functions

These functions operate on only a single expression in the buffer.

Key	Function	Key	Function
mag	magnitude	*exp*	e^x
phase	phase	*10**x*	10^x
real	real component	*y**x*	y^x
imag	imaginary component	*x**2*	x^2
ln	base-e (natural) logarithm	*abs*	|x| (absolute value)
log10	base-10 logarithm	*int*	integer value
dB10	dB magnitude for a power expression	*1/x*	inverse
dB20	dB magnitude for a voltage or current	*sqrt*	$\sqrt{x}$

Note: Selecting these functions while the buffer contains multiple expressions is an error in RPN mode because there is a space between expressions. For example, `ln(expr1 expr2)` is invalid because the logarithm function takes only one argument, not two.

Example: Plotting the Magnitude of a Signal

To plot the dB magnitude of a signal after an AC analysis in RPN mode

1. Click *vf* on the calculator.
2. On the schematic, click the net you want to plot.
3. With the cursor in the Schematic window, press the *Esc* key.

   This cancels the *vf* function. Otherwise, the command stays active.
4. Click *dB20* on the calculator.

   The calculator buffer now contains the expression you want to plot.
5. Click *plot* to show the curve.

Two-Expression Functions and Operators

In RPN mode, two-expression functions operate on both the buffer and the first stack element.

Key	Function in RPN Mode
*y**x*	y^x (in RPN mode, evaluated as $stack_1^{buffer}$)
app	Appends the first stack element to the end of the buffer expression, separated by a space. This operation lets you display multiple waveform expressions in RPN mode. **Note:** This key is not in algebraic mode.
+	Adds the buffer expression to the first stack register.
-	Subtracts the buffer expression from the first stack register.
*	Multiplies the buffer expression by the first stack register.
/	Divides the first stack register by the buffer expression.

Example: Instantaneous Power Dissipation

This example computes the instantaneous power dissipated by a resistor.

1. Click *vt*.
2. On the schematic, click the net connected to the appropriate pin of the resistor.
3. With the cursor in the Schematic window, press the *Esc* key.

   This cancels the *vt* function. Otherwise, the command stays active.
4. Click *it*.
5. Click the appropriate pin of the resistor.

   To select currents, click the square pin symbol. Do not click the wire stub.
6. Click * on the calculator.
7. Click *plot*.

Trigonometric Functions

The trigonometric functions work like the other single-expression functions. Selecting these functions while the buffer contains multiple expressions creates an error in RPN mode because there is a space between expressions.

Special Functions

Special functions help you analyze waveform data generated with calculator expressions. Some functions pop up a form where you enter the data required for the calculation. Other special functions act directly on the data currently in the buffer (RPN mode).

xmax	*fourEval*	*psdbb*
xmin	*frequency*	*riseTime*
ymax	*gainBwProd*	*rms*
ymin	*gainMargin*	*rmsNoise*
average	*groupDelay*	*root*
bandwidth	*harmonic*	*sample*
clip	*harmonicFreq*	*settlingTime*
compression	*iinteg*	*slewRate*
compressionVRI	*integ*	*spectralPower*
convolve	*ipn*	*stddev*
cross	*ipnVRI*	*table*

dBm	*lshift*	*tangent*
delay	*overshoot*	*thd*
deriv	*phaseMargin*	*value*
dft	*phaseNoise*	*xval*
flip	*psd*	*intersect*
evmQpsk		

Example: Average Value of a Current

This example shows how to compute the average value of a current during the simulation period.

1. Click *it* on the calculator.
2. On the schematic, click the terminal whose current you want to average.

   To select currents, click the square pin symbol. Do not click the wire stub.
3. With the cursor in the Schematic window, press the *Esc* key.

   This cancels the *it* function. Otherwise, the command stays active.
4. Choose *Special Functions – Average*.
5. Click *print* on the calculator.

   The system displays the average value in a text window.

RPN Mode Special Functions

Average Function

The *average* function computes the average of a waveform over its entire range. Average is defined as the integral of the expression `f(x)` over the range of `x`, divided by the range of `x`. For example, if `y=f(x)`, `average(y)=`

$$\frac{\int_{from}^{to} f(x)dx}{to - from}$$

where `to` and `from` are the range of `x`.

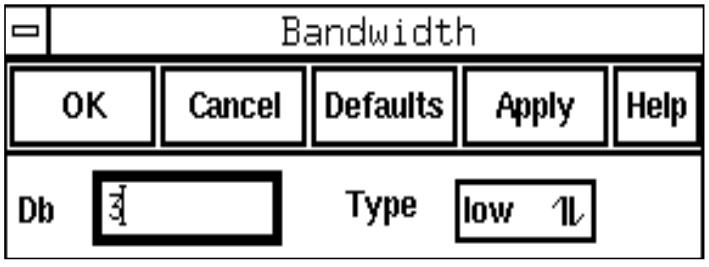
If you want a different range, use the clip function to clip the waveform to the range you want.

Bandwidth Function

The *bandwidth* function calculates the bandwidth of the waveform in the calculator buffer. Please note that the input waveform must represent a true voltage, NOT modified by a dB.

1. Select *bandwidth*.

   The Bandwidth form appears.



2. In the *Db* field, enter how far below the peak value you want to see data.
3. Choose a bandwidth response from the *Type* field.
   - low computes the bandwidth of a low-pass response.
   - high computes the bandwidth of a high-pass response.

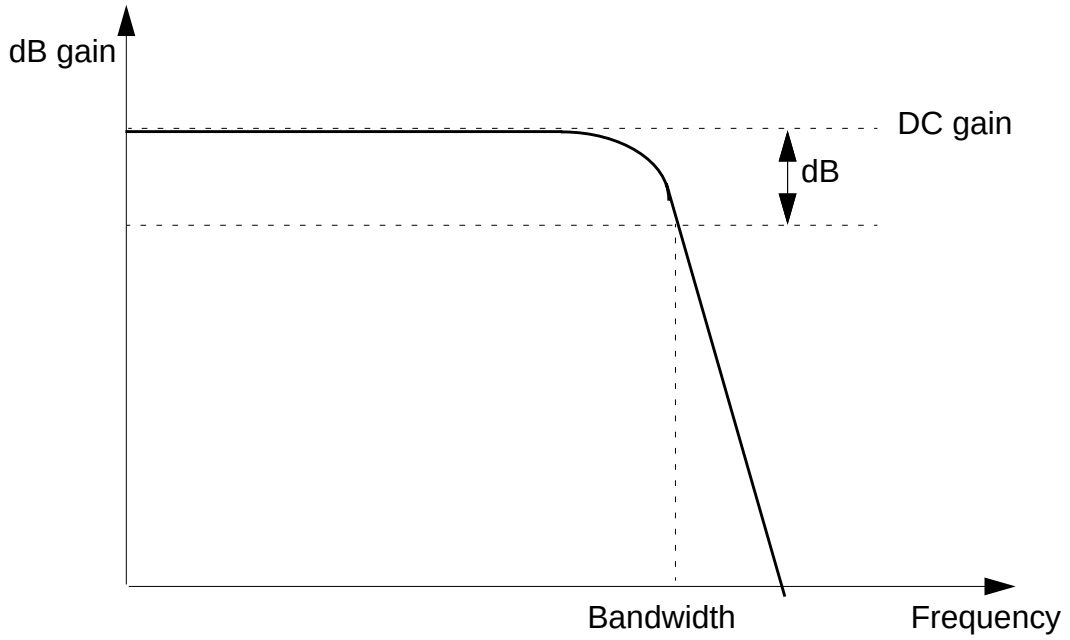
- band computes the bandwidth of a band-pass response.

4. Click *OK*.

Computing Low-Pass Bandwidth

The calculator computes the low-pass bandwidth by determining the smallest frequency at which the magnitude of the input waveform drops *n* decibels below the DC gain. (DC gain is obtained by zero-order extrapolation from the lowest or highest computed frequency, if necessary.) The *dB* field specifies *n*. An error occurs if the magnitude of the input waveform does not drop *n* decibels below the DC gain.

Low-Pass Bandwidth Function

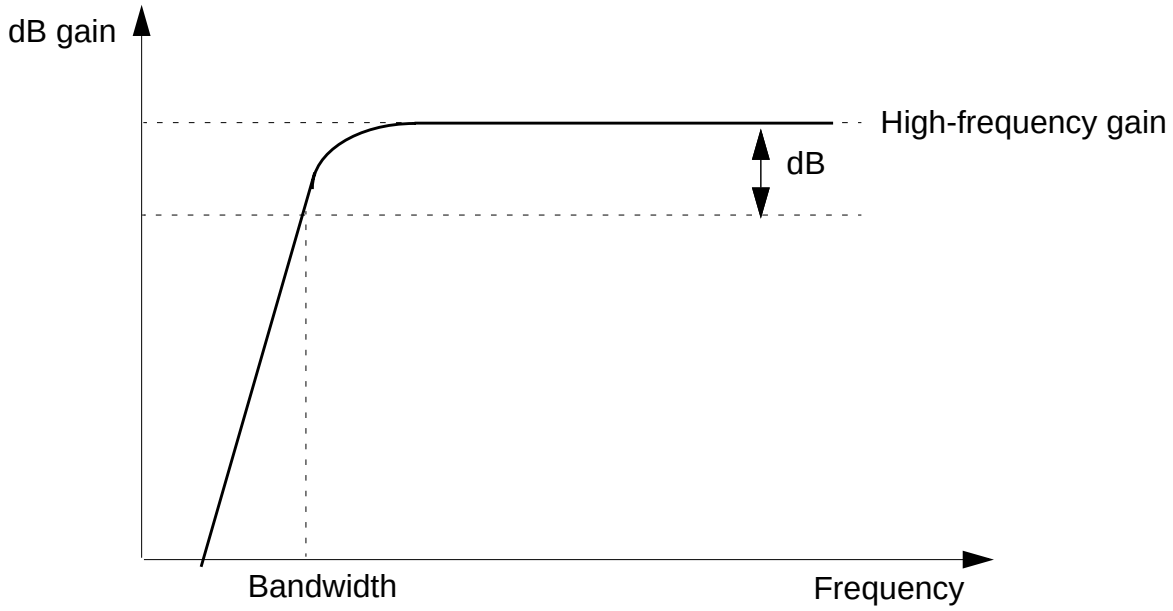


Computing High-Pass Bandwidth

The calculator computes the high-pass bandwidth by determining the largest frequency at which the magnitude of the input waveform drops *n* decibels below the gain at the highest

frequency in the response waveform. The *dB* field specifies *n*. An error occurs if the magnitude of the input waveform does not drop *n* decibels below the gain at high frequency.

High-Pass Bandwidth Function

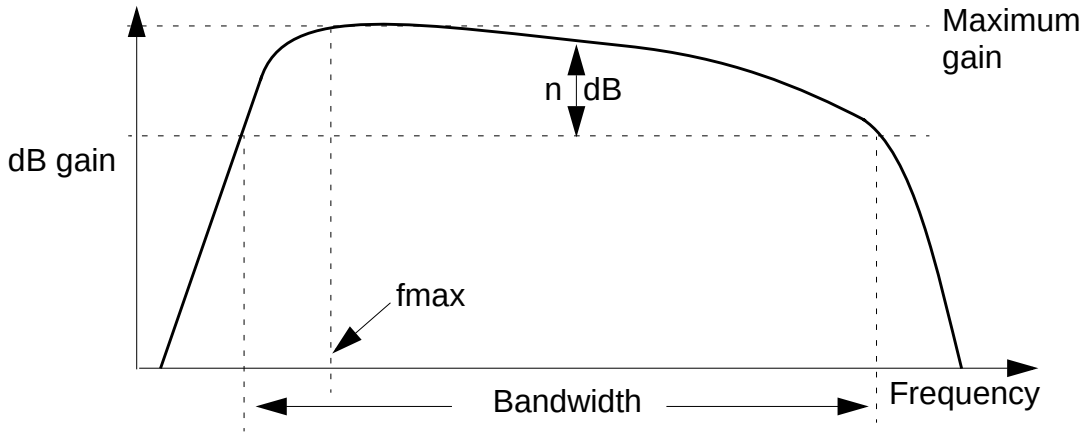


Computing Band-Pass Bandwidth

The calculator computes the band-pass bandwidth by

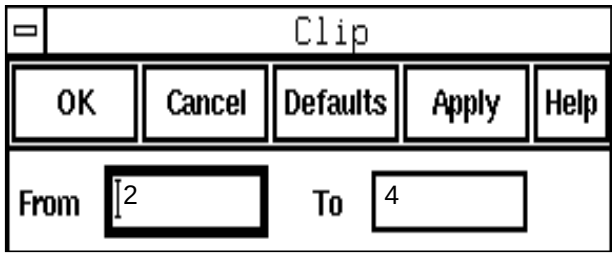
1. Determining the lowest frequency (f_{max}) at which the magnitude of the input waveform is maximized
2. Determining the highest frequency less than f_{max} at which the input waveform magnitude drops *n* decibels below the maximum (*n* is the number you enter in the *dB* field)
3. Determining the lowest frequency greater than f_{max} at which the input waveform magnitude drops *n* decibels below the maximum

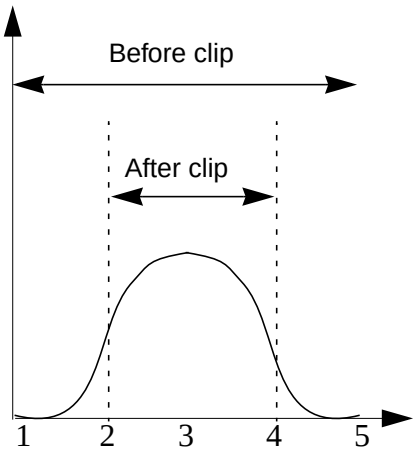
4. Subtracting the value returned by step 2 from the value returned by step 3. The value returned by step 2 or step 3 must exist.



Clip Function

The *clip* function restricts the waveform defined by the buffer expression to the range entered in the *From* and *To* fields. You can use the *clip* function to restrict the range of action of other special functions of the calculator such as integ, rms, and frequency.





Note: The *clip* function does not support multi-valued functions, that is, functions that have multiple y values corresponding to a single x value.

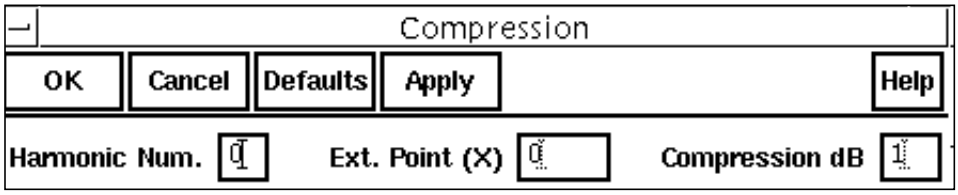
Compression Function

This function returns the *Nth* compression point value of a waveform at the extrapolation point that you specify. To use this function:

1. Set up the ne600p mixer cell from `dfII/samples/artist/rfExamples` library.The design variable frf should be set to 920MHz.
2. Ensure that the sourcetype on the rf port is 'sine' and the Amplitude (dBm) is set to prf.
3. Set up a PSS analysis. The beat frequency should be set to 40MHz. Set the number of harmonics to 2 (only two harmonics are required to determine the 1 dB compression point). Sweep the prf parameter from -30 to 10 in 10 linear steps.
4. Set the Model Library path to include the `dfII/samples/artist/models/spectre/rfModels.scs` file.
5. After running the simulation, call up the Waveform Calculator and the Results Browser. Click on *schematic->psf->Run1->sweeppss_pss_fd-sweep->sweepVariable->prf->10->sweeppss-004_pss-fd.pss->Pif* with the left mouse button. The following will appear in the calculator buffer:

   ```
   v( "/Pif" ?result "sweeppss_pss_fd-sweep" ?resultsDir "~/simulation/ne600p/spectre/schematic" ).
   ```

6. Click on *Special Functions -> compression*. The *Compression* form will be displayed.



7. Enter *Harmonic number=2.* This is the second harmonic of the 40 MHz fundamental frequency, which is the IF frequency (80MHz).
8. Enter *Ext. Point (X)* = `-25` field to specify the extrapolation point of the waveform. The extrapolation point is the X axis value.
9. Enter *Compression dB* = `1` to specify the compression coefficient (N).
10. Click on the *OK* button.
11. Click the *Evaluate Buffer* button in the Calculator. The result appears in the Calculator display.

To use this function, you must type the line below in the CIW

```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

CompressionVRI Function

This function performs an Nth compression point measurement on a power waveform.

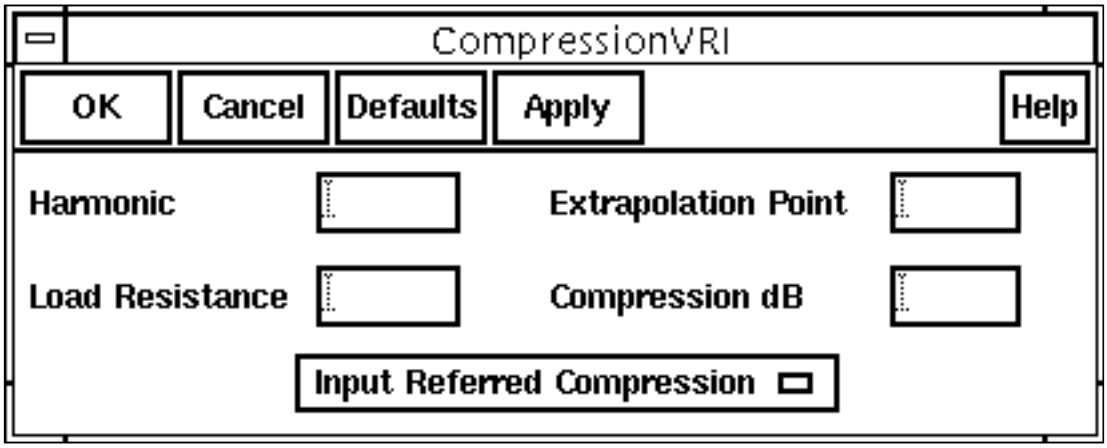
Use this function to simplify the declaration of a compression measurement. This function extracts the specified harmonic from the input waveform(s), and uses `dBm(spectralPower((i or v/r),v))` to calculate a power waveform. The function then passes this power curve and the remaining arguments to the *compression* function to complete the measurement.

The *compression* function uses the power waveform to extrapolate a line of constant slope (dB/dB) according to a specified input or output power level. This line represents constant small-signal power gain (ideal gain). The function then finds the point where the power waveform drops N dB from the constant slope line and returns either the x coordinate (input referred) or y coordinate (output referred) value.

To use this function:

1. Define the voltage waveform in the buffer.
2. Choose *compressionVRI* in the *Special Functions* menu.

   The CompressionVRI form opens.



3. Type a value in the *Harmonic* field to specify the harmonic index of the waveform.

4. Type a value in the *Extrapolation Point* field to specify the extrapolation point for the waveform. The default value is the minimum x value of the input voltage waveform.

   The extrapolation point is the coordinate value in dBm that indicates the point on the output power waveform where the constant-slope power line begins. This point should be in the linear region of operation.

5. Type a numerical value in the *Load Resistance* field. The default value is 50.

6. In the *Compression dB* field, type the delta (in dB) between the power waveform and the ideal gain line that marks the compression point. The default value is 1.

7. Choose whether the measurement is for *Input Referred Compression* or *Output Referred Compression*.

8. Click *OK*.

Convolution (convolve) Function

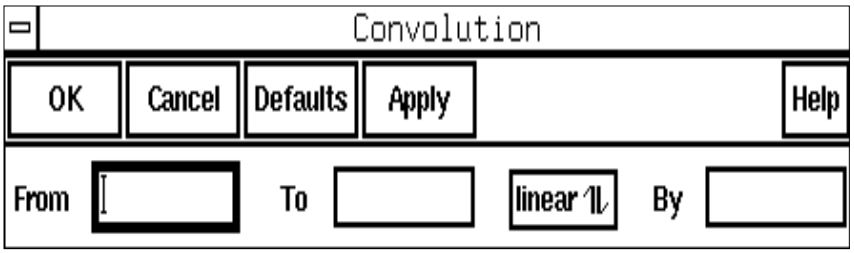
The *convolve* function computes the convolution of two waveforms.

1. Define the first waveform in the buffer.

2. Define the second waveform as the first stack element.

3. Choose convolve from the *Special Functions* menu.

   The Convolution form appears.



Convolution is defined as

$$\int_{from}^{to} f1(s)\,f2(t-s)\,ds$$

f1 and f2 are the functions defined by the first and second waveforms.

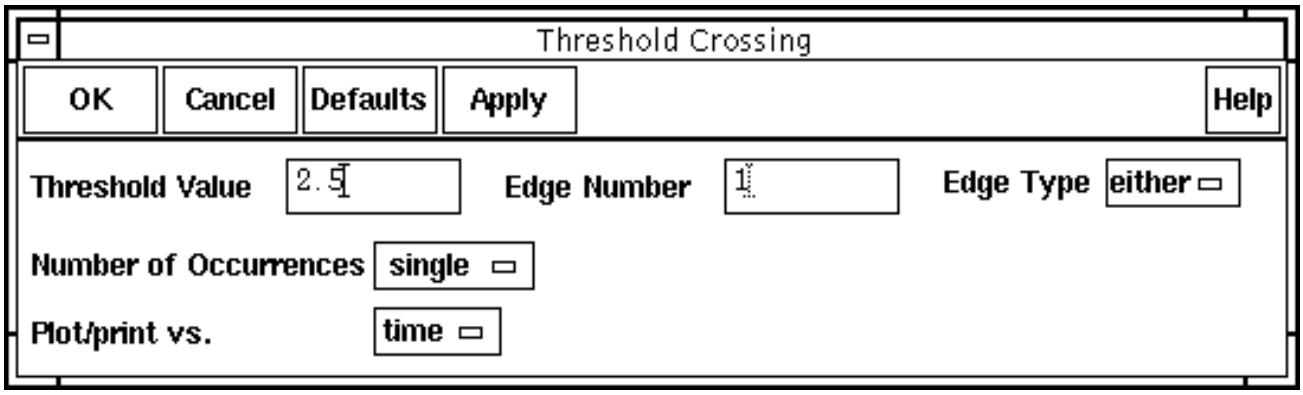
Note: The *convolve* function is numerically intensive and might take longer than the other functions to compute.

Threshold Crossing (cross) Function

The *cross* function computes the x-axis value *xcross* at which the nth crossing of the specified edge type of the threshold value occurs.

1. Choose *cross*.

   The Threshold Crossing form appears.

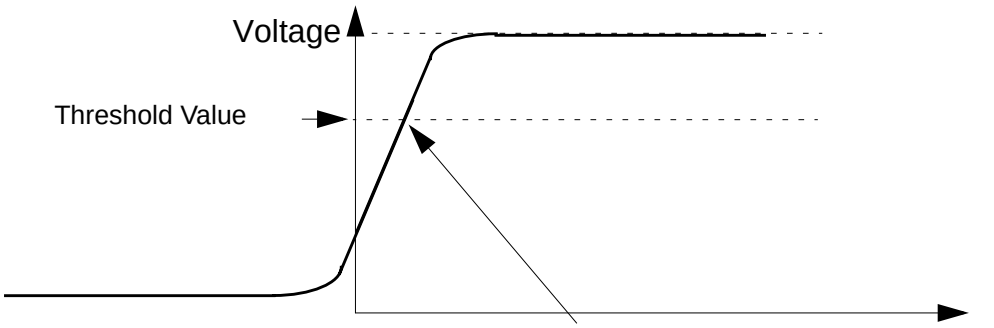


2. Enter the threshold value of the waveform at which to perform the calculation.
3. In *Edge Number*, enter the number of the crossing at which to perform the calculation.

   The integer you enter specifies which crossing is returned. (For example, 1 specifies the first crossing, and 2 specifies the second crossing.)

   If you specify a positive integer, the count starts at the smallest *x* value of the waveform, and the search is in the direction of increasing *x* values. If you specify a negative integer, the count starts at the largest *x* value of the waveform, and the search is in the direction of decreasing *x* values. If you enter 0, all the crossings found are returned in a list.

4. Select an *Edge Type* to determine the crossing as the rising edge, falling edge, or either edge.



5. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of a crossing event for a given waveform. If you select *multiple*, you can retrieve all occurrences of crossing for a given waveform, which you can later plot or print. The field *Edge Number* is dsabled if *Number of Occurrences* is specified as *multiple*.

6. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve crossing data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   cross(VT("/out") 2.5 0 0 t "time") (s)
--------   --------
27.87n     27.87n
823.3n     823.3n
2.027u     2.027u
2.806u     2.806u
```

Here, time values refer to each timepoint on the waveform where the crossing event occurs.

The *cycle* option helps you retrieve crossing data against cycle numbers. For example:

```
cycle      cross(VT("/out") 2.5 0 0 t "cycle") (s)
--------   --------
1          27.87n
2          823.3n
3          2.027u
4          2.806u
```

Here, cycle numbers refer to the n’th occurence of the crossing event in the input waveform.

Note: The value for this field is ignored when *Number of Occurrences* is specified as *single*.

7. Click *OK*.

dBm Function

The *dBm* function performs the operation

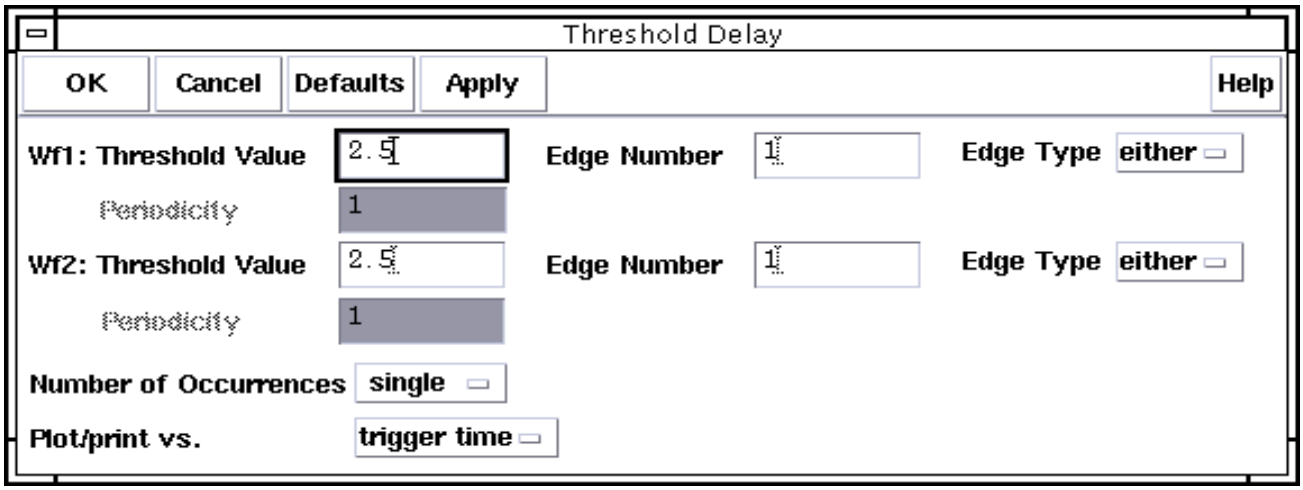
```
dB10(x) + 30
```

1. Enter the value in the buffer.
2. Select *dBm*.

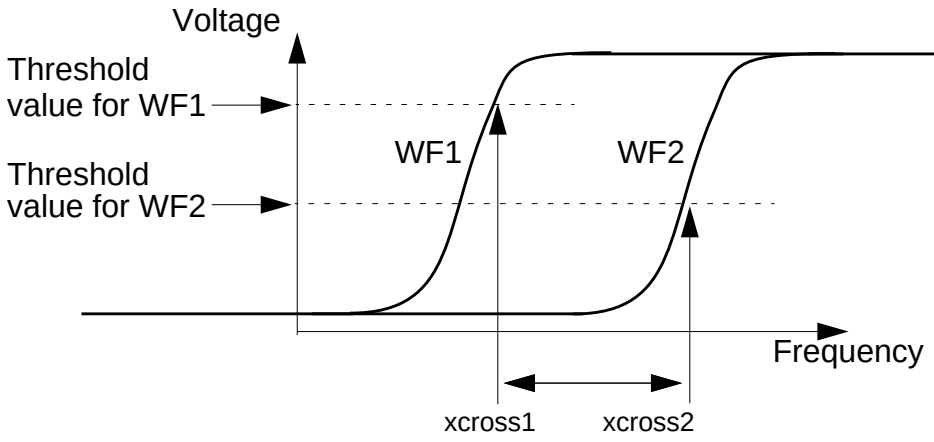
Delay Function

The *delay* function computes the delay between two points or multiple sets of points in a waveform by using the cross function.

1. Enter the second waveform into the first stack register.
2. Enter the first waveform into the buffer.
3. Select *delay* and fill in the Threshold Delay form.



4. For both waveforms, specify *Threshold Value*, *Edge Number*, *Edge Type* and *Periodicity* for the specified edge.



delay = xcross2 - xcross1

5. For *Number of Occurrences*, select *single* or *multiple* to indicate whether you want to calculate delay for one or more occurrences. If you select *multiple*, the delay is computed for the specified edges of the waveforms and also for the edges occuring at the periodic intervals (specified as *Periodicity*) for each waveform.

6. Specify *Plot/print vs.* as *trigger time*, *target time*, or *cycle*. The default setting is *trigger time*, which helps you retrieve delay data against trigger time. Trigger time is the time when the first waveform crosses the specified threshold value.

   For example:

```
cycle      delay(VT("/out") 1.5 1 "rising" VT("/out") 1.5 2 "rising" 1 1 t
           "trigger") (s)
--------   --------
18.19n     1.999u
2.018u     2.001u
4.018u     1.999u
6.018u     2.001u
```

   Here, time values refer to each timepoint on the waveform where the threshold delay occurs.

   Target time is the time when the second waveform crosses the specified threshold value. For example:

```
cycle      delay(VT("/out") 1.5 1 "rising" VT("/out") 1.5 2 "rising" 1 1 t
           "target") (s)
--------   --------
2.018u     1.999u
```

```
4.018u      2.001u
6.018u      1.999u
8.018u      2.001u
```

The *cycle* option helps you retrieve threshold delay data against cycle numbers. For example:

```
cycle       delay(VT("/out") 1.5 1 "rising" VT("/out") 1.5 2 "rising" 1 1 t
            "cycle") (s)
--------    --------
1           1.999u
2           2.001u
3           1.999u
4           2.001u
```

Here, cycle numbers refer to the n'th occurence of the delay event in the input waveform.

Note: The value for this field is ignored when *Number of Occurrences* is specified as *single*.

7. Click *OK*.

Derivative (deriv) Function

The *deriv* function computes the derivative of the buffer expression. You can plot the resulting waveform.

1. Select *deriv*.
2. Enter the expression and closing parenthesis into the calculator buffer.

Note: After the second derivative, the results become inaccurate because the derivative is obtained numerically.

Discrete Fourier Transform (dft) Function

The tool which converts a temporal (time domain) description of a signal (real or complex) into one, in terms of its frequency components is called the Fourier transform. DFT (Discrete Fourier Transform) is the discrete formulation of the Fourier transform, which takes such regularly spaced data values (samples in time domain), and returns the value of the Fourier transform for a set of values in frequency domain which are equally spaced. Most of the time, however, we work on real-valued signals only.

Consider a complex series (signal) *w(k)* with *N* samples of the form

w(0), w(1), w(2), …, w(k), …, w(N-1)

Further, assume that the series outside the range *0, N-1* is extended *N*-periodic, that is, *w(k) = w(k+N)* for all *k*. The DFT of this series will be denoted *W(n)*, will also have *N* samples and will be defined as:

$$W(n) = \frac{1}{N}\sum_{k=0}^{N-1} w(k)\left(e^{-2\pi i k\frac{n}{N}}\right) \qquad \text{where } n = 0, \ldots, N-1$$

- The first sample *W(0)* of the transformed series is the DC component, more commonly known as the average of the input series.
- The DFT of a real series results in a symmetric series about the Nyquist frequency (described below).
- The highest positive (or negative) frequency sample is called the Nyquist frequency. This is the highest frequency component that should exist in the input series for the DFT to receive 'unpredictable' results. More specifically, if there are no frequencies above Nyquist frequency, the original signal can be exactly reconstructed from the samples. The Nyquist Theorem (or Shannon's Sampling Theorem) exactly specifies this, that for a band limited signal, you must sample at a frequency over twice the maximum frequency of the signal, to reconstruct it from the samples.

While the DFT transform above can be applied to any complex valued series, in practice for large series it can take considerable time to compute, the time taken being proportional to the square of the number of points (samples) in the series. A much faster algorithm has been developed by Cooley and Tukey called the FFT (Fast Fourier Transform). The only requirement of the most popular implementation of this algorithm (Radix-2 Cooley-Tukey) is that the number of points in the series be a power of *2* i.e. $N=2^n$.

Given *N* input points, the FFT returns *N* frequency components, of which the first *(N/2 +1)* are valid. (The other components are mirror images and are considered invalid since the frequencies they represent do not satisfy the Nyquist Theorem above.) They start with the DC component, and are spaced apart by a frequency of *(1 / (n deltaT))*. The magnitude of the complex number returned is the frequency's relative strength.

The *dft* function computes the discrete Fourier transform of the buffer by FFT algorithm where *deltaT= (t2-t1) / N*. The waveform is sampled at the following *N* timepoints:

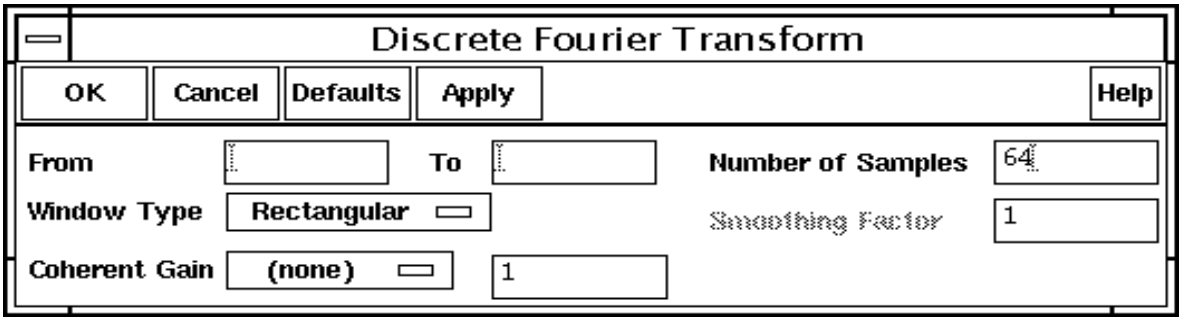
*t1, t1 + deltaT, t1 + 2 * deltaT, º, t1 + (N - 1) * deltaT*

The output of *dft()* is a frequency waveform, *W(f)*, which has (*N/2 + 1*) complex values: the dc term, the fundamental, and (*N/2 - 1*) harmonics.

Note: The last time point, (*t1 + (N - 1) * deltaT*), is (*t2 - deltaT*) rather than *t2*. The *dft* function assumes that *w(t1)* equals *w(t2)*. To use the *dft* function

1. Select *dft*.

   The Discrete Fourier Transform form is displayed.



2. Specify the range over which you want to compute the transform.

   Be sure to cover at least one complete period of your slowest frequency.

3. Enter the *Number of Samples* you want to take in expanding the Fourier transform.

   This number should be a power of *2*. If it is not, the system increases the value to the next higher power of *2*. Sample at a rate that is at least twice your highest frequency component (the Nyquist rate). Pick a sampling rate high enough that closely spaced frequency components can be resolved.

4. Select the *Window Type* option.

   For more information, see the table of window type option values later in this section.

5. Specify the *Smoothing Factor* (for the Kaiser window type only).

   The *Smoothing Factor* field accepts values from *0* to *15*. The value *0* implies no smoothing and is equivalent to a rectangular window. The default value for the *Smoothing Factor* field is *1*.

6. Click *OK*.

When you run the transient analysis, keep the maximum time step small enough to represent the highest frequency component accurately. The maximum time step should be smaller than the sampling period that you use for the discrete Fourier transform (DFT) of the time domain

waveform. The samples in the DFT will either hit a data point (calculated exactly by the simulator) or an interpolated point between two data points.

Choosing a maximum timestep during transient simulation that is smaller than the DFT sampling period ensures that sampling maintains a resolution at least equal to that of the transient time-domain waveform.

The start and stop times should not coincide with the boundaries of the time-domain waveform. The boundary solutions might be imprecise and generate incorrect results if used in other calculations.

One of the uses of fast Fourier transform (FFT) windowing is to reduce discontinuities at window edges caused by having a nonintegral number of periods of a signal in a window. This removes the abrupt edges, making them fall off smoothly to zero, and can improve the validity of the FFT components obtained. You can also use FFT windowing to ‘dig out’ the details of signal components that are very close Gin frequency or that consist of both large and small amplitudes.

The following table was obtained from the book *The FFT, Fundamentals and Concepts* by R. W. Ramirez, Prentice Hall, 1985. As explained in this reference, the values in the table were computed from software-generated windows and might vary slightly from theoretical values. In the third column, the peak magnitude of each window is compared with that of the rectangular window. In the fourth column, the amplitude of the highest side lobe is given in decibels referenced to the major lobe peak. The fifth column contains the 3dB bandwidth of the major lobe, normalized to one over the window’s width. The last column gives the theoretical rolloff of the side lobes.

Window name	**Shape equation**	**Major lobe height**	**Highest side lobe (dB)**	**Band-width (3 dB)**	**Theor. rolloff (dB/ octave)**
Cosine4	$A=(.5(1-\cos(2\pi t/T))^2$ for t=0 to T	0.36T	-46.9	1.79/T	30
ExtCosBell	A=0.5(1-cos(2π5t/T) for t=0 to T/10 and t=9T/10 to T A=1 for t=T/10 to 9T/10	0.9T	-13.5	0.95/T	18 (beyond 5/T)
HalfCycleSine	A=sin(2π0.5t/T) for t=0 to T	0.64T	-22.4	1.15/T	12
HalfCycleSine3	$A=\sin^3(2\pi 0.5t/T)$ for t=0 to T	0.42T	-39.5	1.61/T	24

Window name	Shape equation	Major lobe height	Highest side lobe (dB)	Band-width (3 dB)	Theor. rolloff (dB/ octave)
HalfCycleSine6					
Hamming	A=.08 + .46(1-cos(2πt/T)) for t = 0 to T	0.54T	-41.9	1.26/T	6 (beyond 5/T)
Kaiser					
Parzen	$A=1 - 6(2t/T-1)^2 + 6\|2t/T-1\|^3$ for t=T/4 to 3T/4 $A=2(1 - \|2t/T-1\|)^3$ for t=0 to T/4 and t=3T/4 to T	0.37T	-53.2	1.81/T	24
Rectangular	A=1 for t=0 to T	T	-13.2	0.86/T	6

Sources of Errors

- *dft()* performs interpolation to determine values of *w(t)* that are not directly available from the simulator output. This interpolation can cause an inaccurate spectrum.
- If *(t2 - t1)* is not the time period of *w(t)*, the output of *dft()* might be misleading.
- If the simulator generated values are inaccurate, *dft()* returns a frequency waveform with many 'insignificant' harmonics. You can minimize the amplitudes of these harmonics by increasing the accuracy of the simulator.

Discrete Fourier Transform Baseband (dftbb) Function

The *dftbb* function computes the discrete Fourier transform (fast Fourier transform) of a complex signal *z(t)= x(t) + j*y(t)*:

$$Z(n)= ReZ(n) +j*ImZ(n)=SUM[\ \sum_{k=0}^{N-1} z(k)*\exp(-j*theta*n*k)],$$

where *theta=2*Pi/N; n=0, 1, ..., N-1*.

Both waveforms are sampled at the following N timepoints:

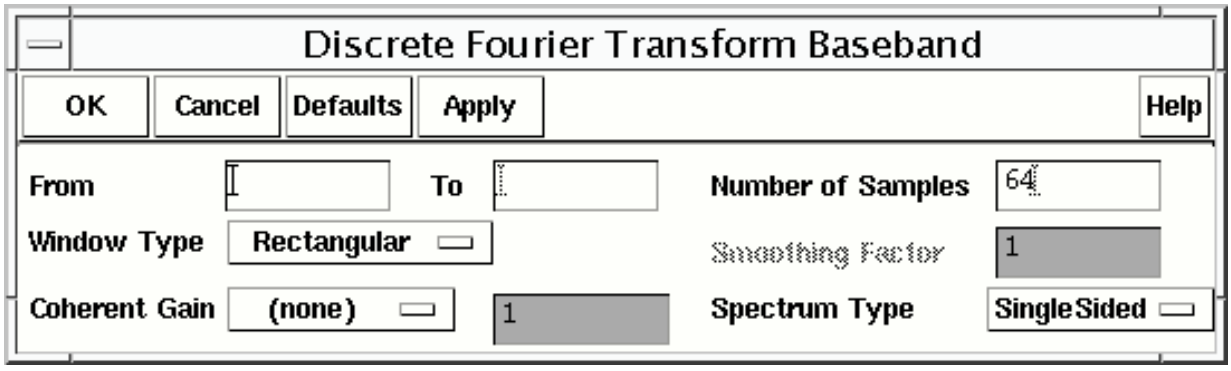
*t1, t1 + deltaT, t1 + 2 * deltaT, ..., t1 + (N - 1) * deltaT.*

The output of *dftbb(waveform1, waveform2)* are N complex values.

The above definition is for single-sided output waveforms. This holds true for double-sided output waveforms except that the previous output waveform is translated so that *n* varies from *-N/2 to (N/2)-1*.

1. Define the first waveform in the buffer.
2. Put the expression for the first waveform in the stack by clicking `enter`.
3. Clear the buffer and define the second waveform in the buffer.
4. Choose *dftbb* from the *Special Functions* menu in Calculator.

   The Discrete Fourier Transform Baseband form appears.



5. Fill in the values, as required. See the previous topic, on the *dft* function, to know more about these options.

   Note: For *Window Type*, only *Rectangular* is supported. AWD issues a warning if you create an expression by hand for a different window type and switches to the default *Rectangular* window type.

6. Click *OK*.

You get an expression for *dftbb* as *plotted*, *evaluated*, or *printed*.

evmQpsk Function

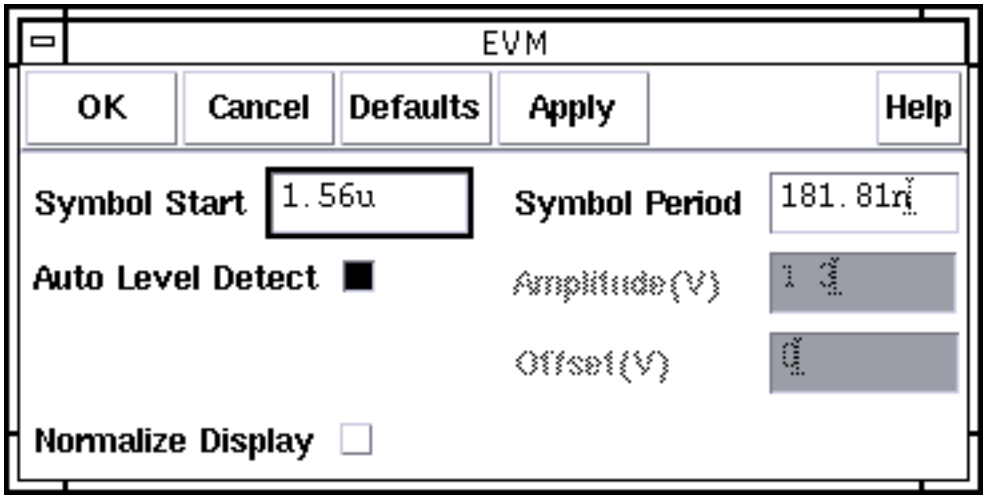
Error Vector Magnitude (EVM) is a useful measurement to describe the overall signal amplitude and phase modulated signal quality. It is based on a statistical error distribution normalized from an ideal digital modulation. Quadrature Phase Shift Keying (QPSK) is a

typical modulation scheme where EVM is useful. EVM is a signal path fidelity figure that could be affected by distortions such as background noise, amplitude modulated noise, compression effects, phase noise, and Inphase (I) & Quadrature-phase (Q) mismatches. Each of these effects have signature characteristics and can be graphically plotted in a constellation for inspection.

The *evmQpsk* function processes the I and Q waveform outputs from the transient simulation run to calculate the EVM and plot the I versus Q scatterplot. The EVM is calculated by detecting the I and Q signal levels corresponding to the four possible I and Q symbol combinations and calculating the difference between the actual signal level and the ideal signal level.

1. Define the waveform for the I signal in the buffer.
2. Put the expression for the first waveform in the stack by clicking `enter`.
3. Clear the buffer and define the waveform for the Q signal in the buffer.
4. Choose *evmQpsk* from the *Special Functions* menu in Calculator.

   The EVM form appears.



5. Specify the start time for the first valid symbol in the *Symbol Start* field. This can be obtained from the Waveform Viewer window by recording the time of the first minimum or first maximum (whichever is earlier) on the selected signal stream.
6. Specify a period for the symbol in the *Symbol Period* field. Each period is represented by a data rate. The data rate at the output is determined by the particular modulation scheme being used. For example, if the data rate selected is `5.5 Mbps`, it corresponds to a period of `181.8 ns`.

7. The *Auto Level Detect* option button is selected by default and the values in the *Amplitude* and the *Offset* fields are automatically calculated. *Amplitude* is calculated by averaging the rectified voltage level of the signal streams and *Offset* by averaging the sum of an equal number of positive and negative symbols in each signal stream. These values are used to determine the EVM value. If you would like to specify values in the *Amplitude* and the *Offset* fields, you deselect this option button.

Caution

While this feature is convenient, it has to be used with caution as it tends to underestimate the EVM number. The automatic determination of levels means that for mismatching I and Q channels, the EVM would be underestimated because the I and Q channels are compared to their individual average signal level instead of a common ideal level. Also, undesired DC offset, if any, would not be included in the calculated EVM.

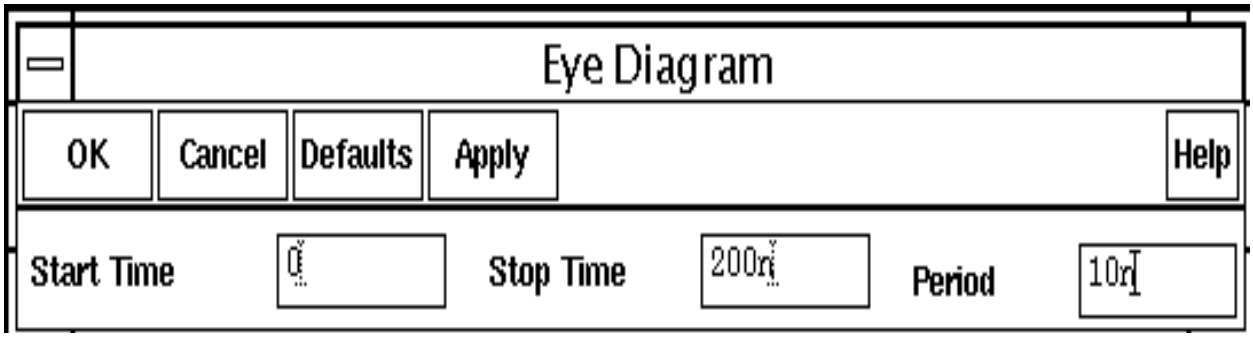
8. Select the *Normalize Display* option button when you want to see the scatter plot normalized to the ideal values `+1` and `-1` (for example, when superimposing scatter plots from different stages in the signal flow, where the levels may be quite different but the you want to see relative degradation or improvement in the scatter). This button does not affect the calculation of the EVM number.
9. Click *OK*.

eyeDiagram Function

The *eyeDiagram* function gives an eye-diagram plot in which the waveform signal is divided into fixed time periods, which are then superimposed on each other. The result is a plot that has many overlapping lines enclosing an empty space known as the "*eye*". The quality of the receiver circuit is characterized by the dimension of the eye. An open eye means that the detector will be able to distinguish between `1's` and `0's` in its input, while a closed eye means that a detector placed on `Vout` is likely to give errors for certain input bit sequences.

1. Define a waveform in the buffer.

2. Choose *eyeDiagram* from the *Special Functions* menu in *Calculato*r. The Eye Diagram form appears.



3. Enter the values for Start Time, Stop Time and Period.
4. Click *OK*.
5. Click on the *plot button* in the *Calculator* to plot the eyeDiagram.

Flip Function

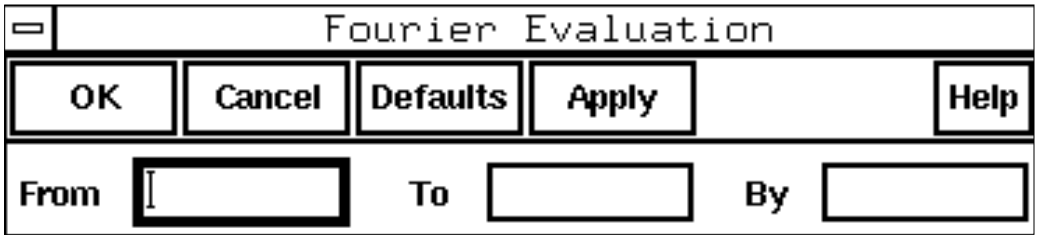
The *flip* function negates the X values. It has the effect of flipping the waveform horizontally in relation to the y axis.

Fourier Evaluation (fourEval) Function

The *fourEval* function evaluates the Fourier series represented by the buffer expression. This function is an inverse Fourier transformation and thus the inverse of the *dft* function described in Discrete Fourier Transform (dft) Function. It transforms the buffer expression from the frequency domain to the time domain.

1. Select *fourEval*.

   The Fourier Evaluation form appears.



2. Specify the time range over which you want to evaluate the series.

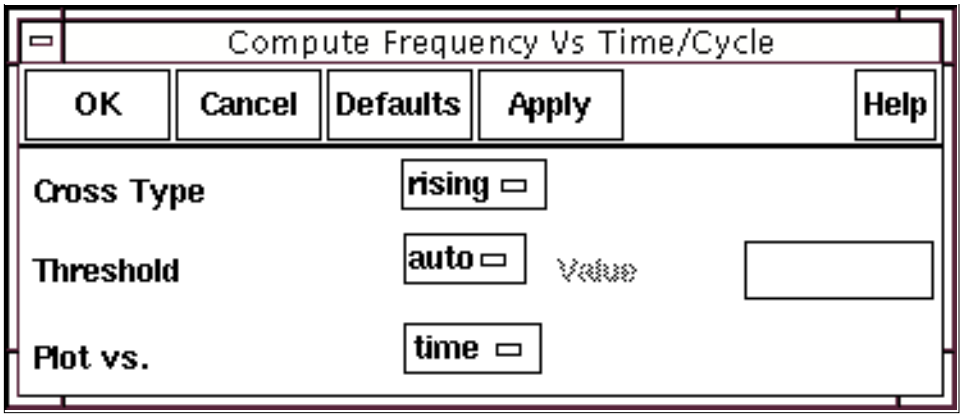
3. Enter the increment for evaluating the series.

Freq Function

The *freq* function estimates the frequency of the input waveform(s) as a function of time or cycle.

1. Select *freq*.

   The Compute Frequency Vs. Time/Cycle form appears.



2. Specify *Cross Type* as *rising* or *falling*.
3. Specify *Threshold* as *auto* or *user*. If you select *auto*, thethreshold value is calculated internally. If you select *user*, the *Value* field becomes editable and you need to also specify a value.
4. Specify what you want to plot the frequency against in the *Plot vs.* field. You may select *time* or *cycle*.
5. Click *OK*.
6. Click on the *plot button* in the *Calculator* to plot the frequency.

Frequency Function

The *frequency* function estimates the frequency of a periodic waveform. The system computes the reciprocal of the average time between two successive midpoint crossings of the rising waveform.

Gain (gainBwProd/gainMargin) Functions

The *gainBwProd* function calculates the gain-bandwidth product. This function requires one argument, the frequency response of interest over a sufficiently large frequency range.

$$gainBwProd\ (\ gain\ ) = A_o * f2$$

The gain-bandwidth product is calculated as the product of the DC gain A0 and the critical frequency f2.The critical frequency f2 is the smallest frequency for which the gain equals $1/\sqrt{2}$ times the DC gain A_o.

The *gainMargin* function computes the dB value of the buffer expression when its phase crosses 180 degrees. This value represents the gain margin in unity gain configuration.

Group Delay Function

The *groupDelay* function computes the group delay of the expression in the buffer.

Group delay is defined as the derivative of the phase with respect to frequency. Group delay is expressed in seconds. It is calculated using the *vp* function as shown below.

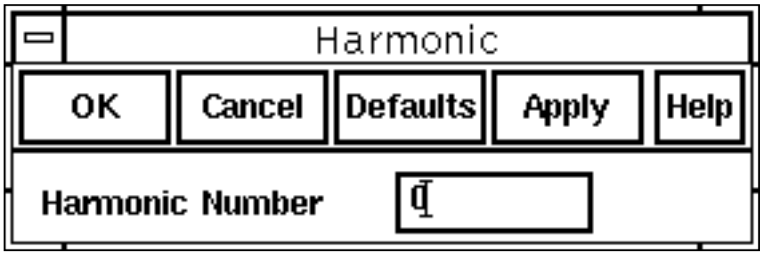
$$Group\ Delay = \frac{d\phi}{d\omega} = \frac{d}{df}\left[\frac{phase(/netX)}{360}\right]$$

Harmonic Function

This function returns the harmonic waveform of a waveform you specify.

1. Enter the expression for the waveform in the Calculator buffer.

2. Choose *harmonic* in the *Special Functions* menu of the Calculator to open the Harmonic form.



3. In the Harmonic form, specify the harmonic you want by typing the value in the *Harmonic Number* field.
4. Click *OK* or *Apply* in the Harmonic form.
5. Click *plot* in the Calculator to plot the waveform.

To use this function, you must type the line below in the CIW

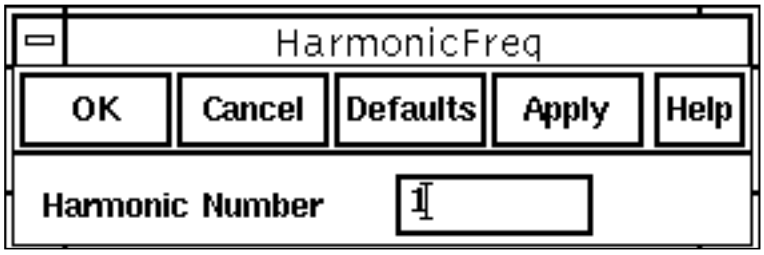
```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

Harmonic Frequency Function

This function returns the harmonic waveform of a waveform you specify.

1. Enter the expression for the waveform in the Calculator buffer.
2. Choose *harmonicFreq* in the *Special Functions* menu of the Calculator to open the HarmonicFreq form.



3. In the HarmonicFreq form, specify the harmonic you want by typing the value in the *Harmonic Number* field.
4. Click *OK* or *Apply*.

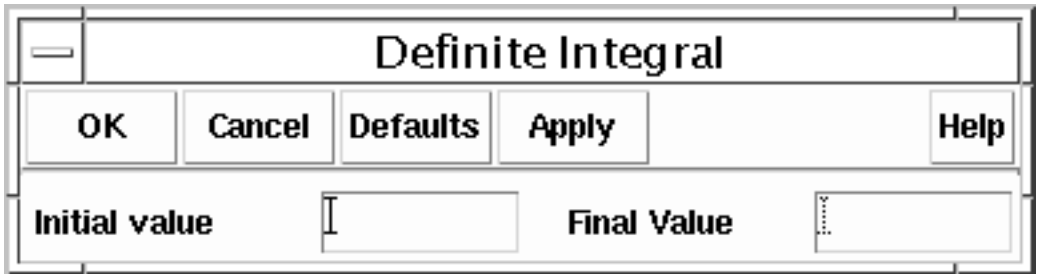
iinteg Function

The *iinteg* integral function computes the indefinite integral of the buffer expression with respect to the X-axis variable. The result is a waveform that can be plotted.

integ Function

The *integ* integral function computes the definite integral of the expression in the buffer. The result is the value of the area under the curve over a specified range on the X-axis of the expression.

1. Define a waveform in the buffer.
2. Choose *integ* from the *Special Functions* menu in Calculator. The Definite Integral form appears.



3. Enter the values for the limits of the definite integral in the *Initial Value* and *Final Value* fields.

   Note: You should specify either both the limits or neither. In case you do specify the limits, they become the end points of the range on the X-axis for definite integration. If you do not specify the limits, then the range for definite integration is the entire range of the sweep on the X-axis.

4. Click *OK*.

intersect Function

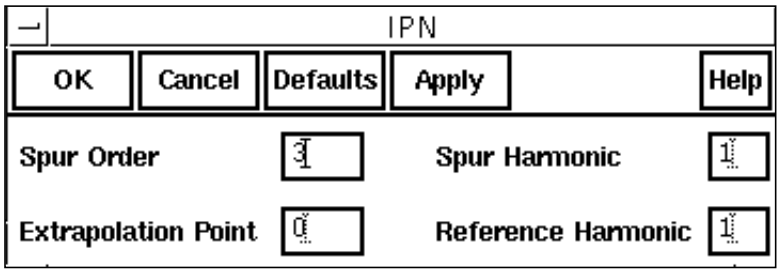
This function returns a waveform containing the points of intersection for the two waveforms passed as input.

1. Enter the expression for second waveform into the first stack register.
2. Enter the expression for first waveform into the buffer.
3. Choose *intersect* from the *Special Functions* menu of the Calculator.

ipn Function

This function plots the *Nth* order intercept between two harmonics of a waveform that you define.

1. Enter the expression for the waveform in the Calculator buffer.
2. Choose *ipn* in the *Special Functions* menu of the Calculator. The IPN form opens.



3. Type values for the following four quantities:

Spur Order	Spur Order determines what order of interference is calculated for the spurious and reference waves. The default value is 3; this corresponds to the *IP3* function. If you use a value other than 3, that order of interference is calculated between those two waves.
Spur Harmonic	Harmonic number for spurious waveform.
Reference Harmonic	Harmonic number for reference waveform.
Extrapolation Point	The extrapolation point for the *IPN* function. This is the X axis value.

4. Click *OK* or *Apply*.

   The expression is sent to the calculator buffer. To evaluate the expression, click the *Evaluate Buffer* button in the calculator.

To use this function, you must type the line below in the CIW

```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

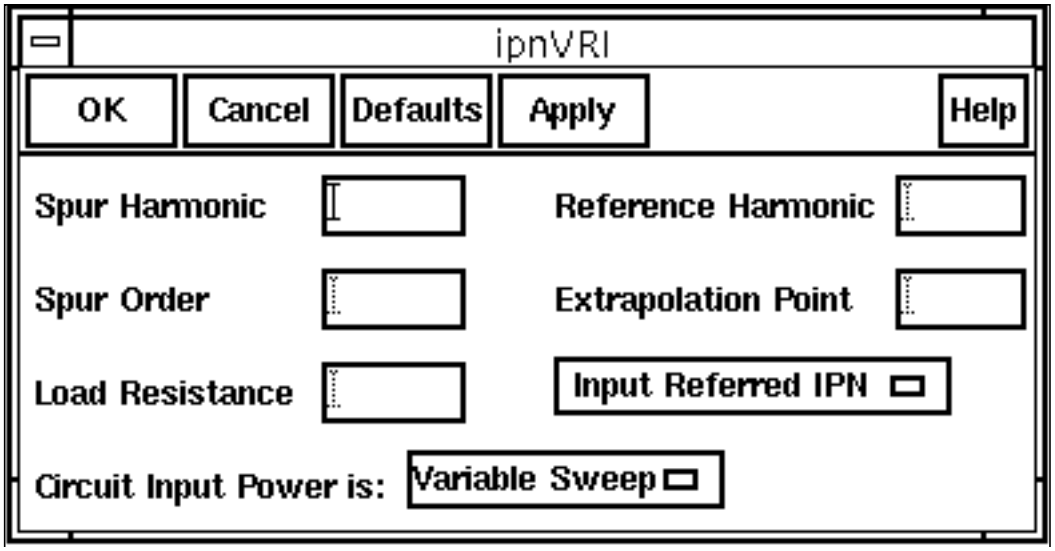
ipnVRI Function

This function performs an intermodulation Nth-order intercept point measurement.

Use this function to simplify the declaration of an ipn measurement. This function extracts the spurious and reference harmonics from the input waveform(s), and uses `dBm(spectralPower((i or v/r),v))` to calculate the respective powers. The function then passes these power curves or numbers and the remaining arguments to the *ipn* function to complete the measurement.

From each of the spurious and reference power waveforms (or points), the *ipn* function extrapolates a line of constant slope (dB/dB) according to the specified order and input power level. These lines represent constant small-signal power gain (ideal gain). The *ipn* function calculates the intersection of these two lines and returns the value of either the x coordinate (input referred) or y coordinate.

1. Enter the expression for the waveform in the Calculator buffer.
2. Choose *ipnVRI* in the *Special Functions* menu of the Calculator. The ipnVRI form opens.



3. Type values for the following quantities:

Spur Harmonic	Harmonic index for spurious waveform.
Reference Harmonic	Harmonic index for reference waveform.

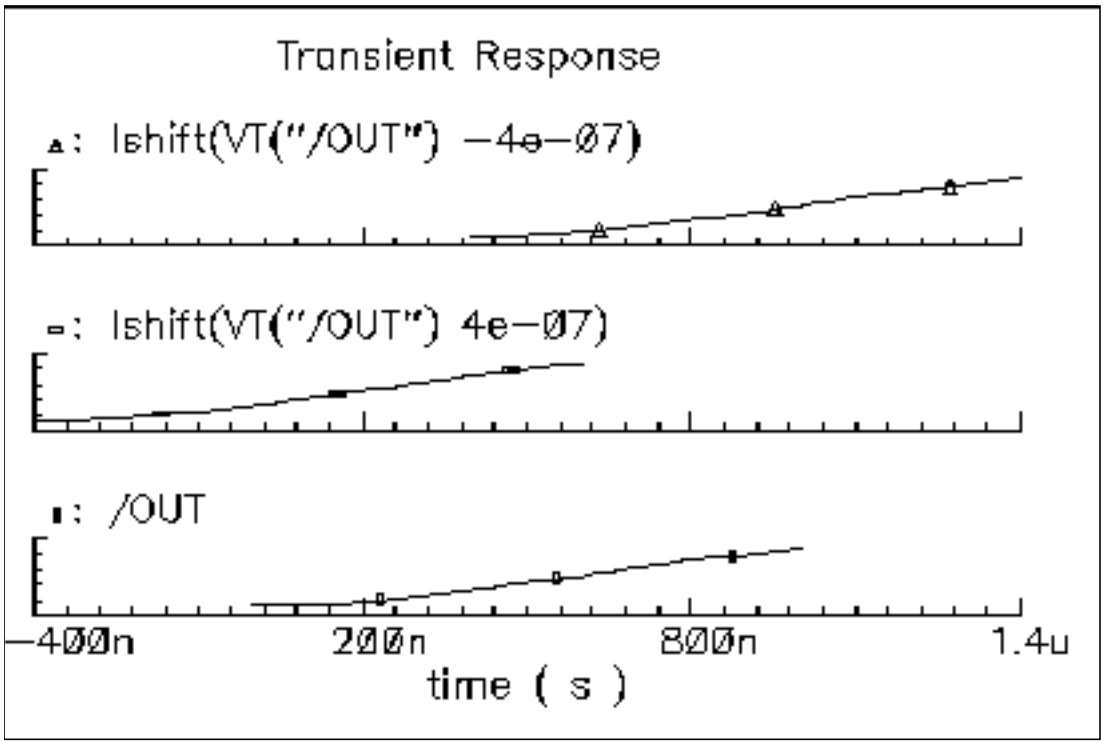
Spur Order	Spur Order determines what order of interference is calculated for the spurious and reference waves. The default value is 3; this corresponds to the *IP3* function. If you use a value other than 3, that order of interference is calculated between those two waves.
Extrapolation Point	The extrapolation point for the *ipn* function. This is the X axis value. The default is the minimum x value of the input voltage waveform.
Load Resistance	The resistance into the output port. The default value is 50.

4. To get the X-coordinate of the intercept, specify *Input Referred IPN*. To get the Y-coordinate of the intercept, specify *Output Referred IPN*.
5. Indicate whether the *Circuit Input Power* is a *Variable Sweep* or a *Single Point*.
6. Click *OK* or *Apply*.

Note: For an extended design example, using ipn and ipnVRI functions, see Appendix A, “Using the Calculator Special Functions with SpectreRF Simulation Results”.

Lshift Function

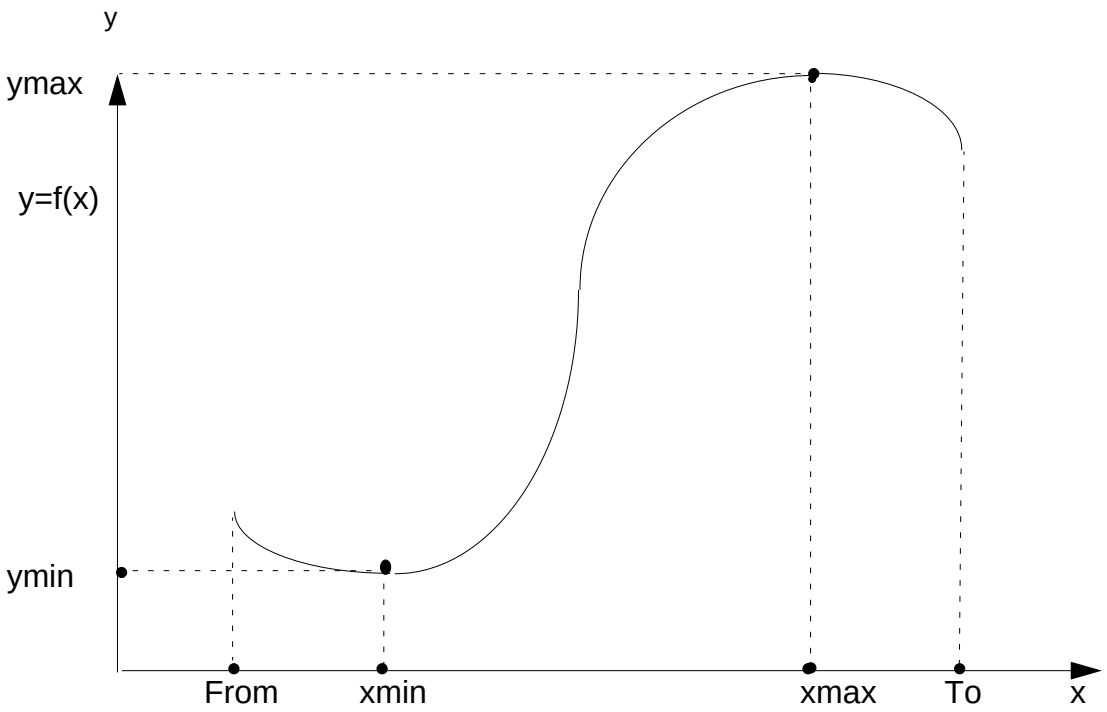
The *lshift* function shifts the data in the Waveform window to the left by a specified amount. A negative value shifts the data to the right.



Minimum and Maximum Functions

You can calculate minimum and maximum values of waveforms with the *xmax*, *xmin*, *ymax*, and *ymin* functions.

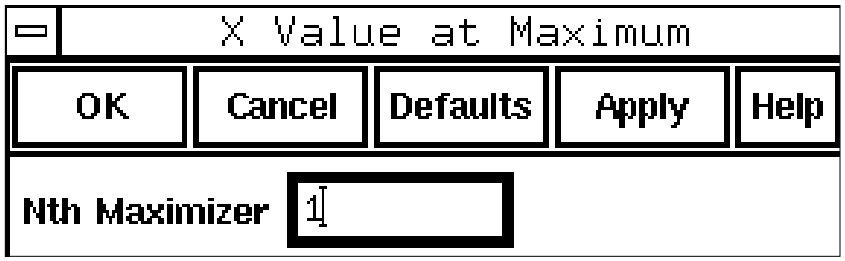
This figure shows the relationship of these functions.



xmax and ymax

The *xmax* function computes the value of the independent variable x at which the expression attains its maximum value, that is, the value of x that maximizes y=f(x).

The maximum might occur at more than one point on the x axis, so you must choose (in the *Nth Maximizer* field) which maximum value you want to see. The calculator returns the value of the Nth Maximizer counting from the left, that is, toward increasing X-axis values. If you enter a negative integer, the direction of search is reversed toward decreasing X-axis values (counting from the right).

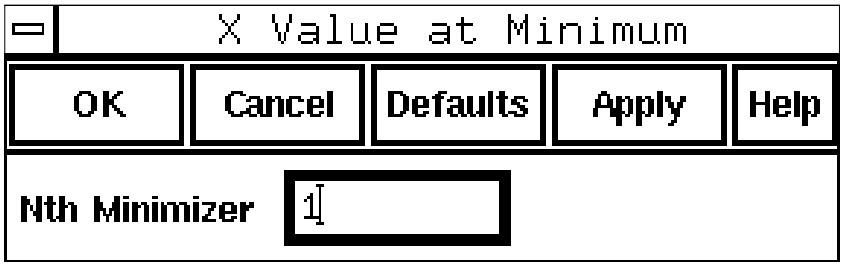


The *ymax* function computes the maximum y of the expression y=f(x).

xmin and ymin

The *xmin* function computes the value of the independent variable x at which the expression has its minimum value, that is, the value of x that minimizes y=f(x).

The minimum might occur at more than one point on the x axis, so you must choose (in the *Nth Minimizer* field) which minimum value you want to see. The calculator returns the value of the Nth Minimizer, counting from the left, that is, toward increasing X-axis values. If you enter a negative integer, the direction of search is reversed toward decreasing X-axis values (counting from the right).



The *ymin* function computes the minimum y of the expression y=f(x).

Overshoot Function

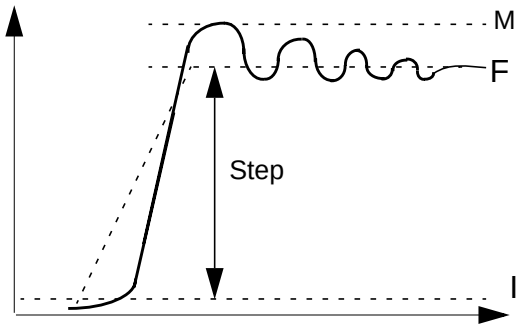
The *overshoot* function computes the percentage by which the buffer expression overshoots a step going from the *Initial Value* to the *Final Value* you enter.

To retrieve one or multiple occurrences of overshoot in a waveform:

1. Enter an expression in the Calculator buffer.

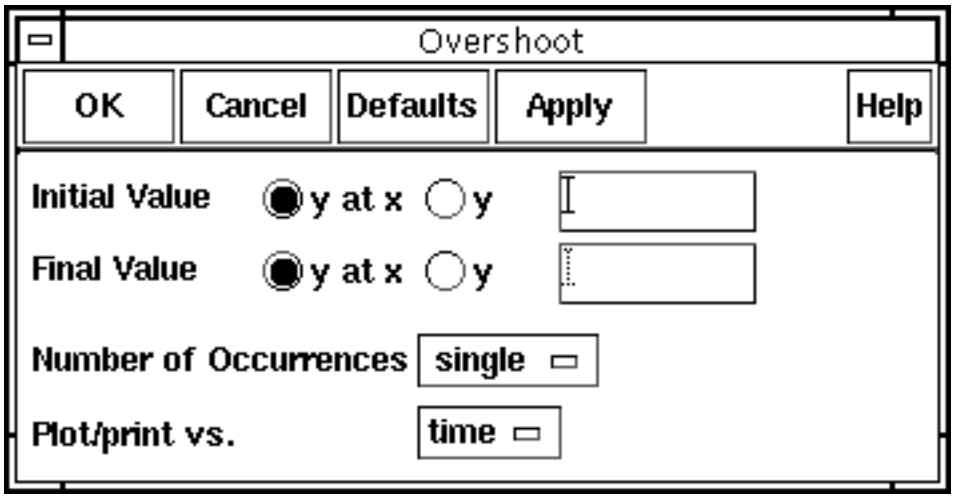
2. Select *overshoot* from the *Special Functions* menu.

$$Overshoot = \frac{(M-F)x100}{F-I}$$



3. There are two ways to set the *Initial* and *Final Value*:

- Use the *y* option and enter the y values directly (for example, 3V) or a horizontal marker name (for example, M2).
- Use the *y at x* option to enter an x value or vertical marker name (for example, M1). The system calculates the corresponding y value for the current waveform.



4. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of an overshoot event for a waveform. If you select *multiple*, you can retrieve all occurrences of overshoot for the waveform, which you can later plot or print.

5. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve overshoot data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   overshoot(VT("/out") 0.5 nil 4.95 nil 5 t "time")
--------   --------
7.862n     12.05
4.008u     12.05
8.008u     12.05
```

Here, time values refer to each timepoint on the waveform where the overshoot event occurs.

The *cycle* option helps you retrieve overshoot data against cycle numbers. For example:

```
cycle      overshoot(VT("/out") 0.5 nil 4.95 nil 5 t "cycle")
--------   --------
1          12.05
2          12.05
3          12.05
```

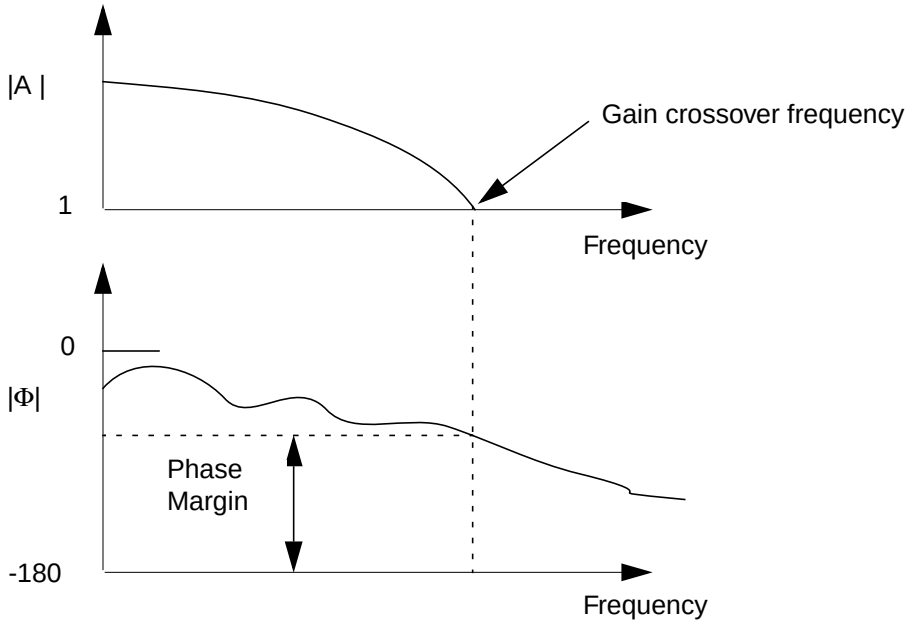
Here, cycle numbers refer to the n’th occurence of the overshoot event in the input waveform.

Note: The value for this field is ignored when *Number of Occurrences* is specified as *single*.

6. Click *OK*.

Phase Margin Function

The *phaseMargin* function computes the difference between the phase (in degrees) at 180 and the frequency at which the buffer expression magnitude equals 1. This function is similar to the phaseDegUnwrapped SKILL command.

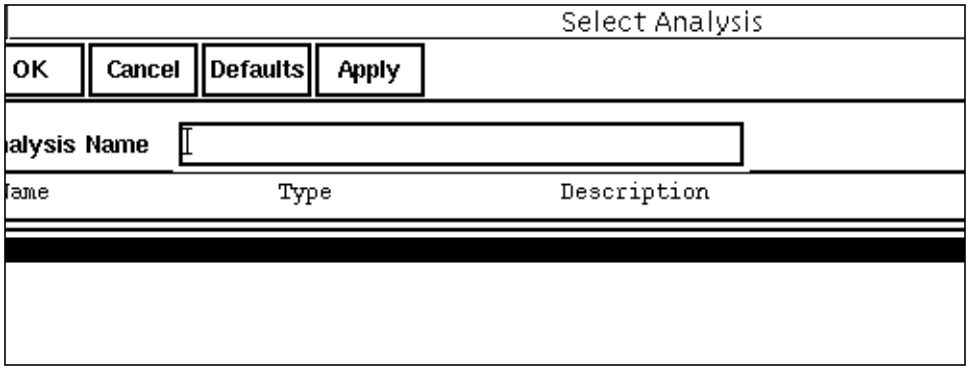


Phase Noise Function

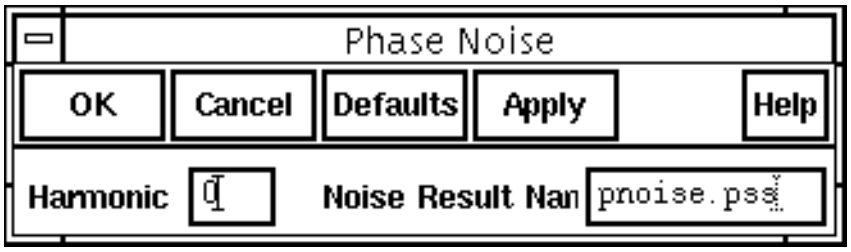
This function plots the phase noise waveform for noise analysis results. You need to follow the following steps to use this function.

1. Set up your PSS analysis.
2. Check the *oscillator button* in the *choosing analysis* form.
3. Set up a Pnoise analysis. Note the value you are entering for the relative harmonic in the *choosing analysis* form.
4. Run your simulation.

5. Invoke the Calculator . Choose *phaseNoise* in the *Special Functions* menu of the Calculator. The *Select Analysis* form appears.



6. Choose the pnoise-pnoise analysis name in the *Analysis Name* field of the *Select Analysis* form, or select the analysis name from the list box in the form. Click on the *OK* button in the form. The *Phase Noise* form appears.



7. In the Phase Noise form, enter *harmonic number=1* or the value you had entered for relative harmonic in the *pnoise choosing analysis* form.
8. Enter the *Noise Result Name* as pnoise-pnoise and click on *OK* in the form.
9. In your calculator buffer, you should see something like phaseNoise(1, "pss-fd.pss", “result "pnoise-pnoise").
10. Click on the *plot* button in the Calculator to plot the waveform.
11. The phase noise plot will appear.

To use this function, you must type the line below in the CIW

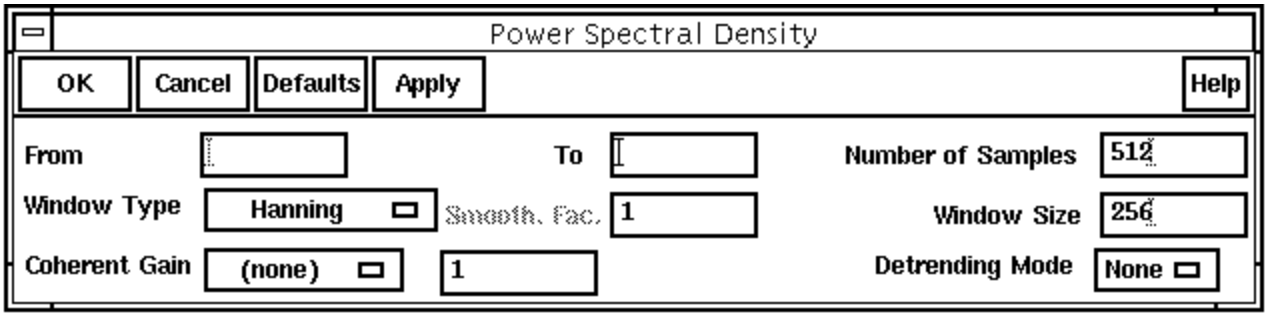
```
envSetVal("calculator" "oldexpr" ‘boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

Power Spectral Density (psd) Function

The power spectral density (*psd*) function describes how the power (or variance) of a time series (signal) is distributed with frequency. Mathematically, it is defined as the Fourier Transform of the auto correlation sequence of the time series (signal). The waveform is first interpolated, to generate evenly spaced data points in time. The spacing of the data points is the inverse of the *dft* sampling frequency. The *psd* is computed by first breaking up the time interval into overlapping segments. Each segment is multiplied, time point by time point, by the specified windowing function. The *dft* is performed on each windowed segment of the baseband waveform. At each frequency, the *dfts* from all segments are averaged together and the squared modulus of these averages, gives *psd*.

After you choose *psd* in the *Special Functions* menu of the Calculator, the Power Spectral Density form opens.



1. In the *From* field, type the starting time for the spectral analysis interval.
2. In the *To* field, type the ending time for the spectral analysis interval. You can use this parameter and the *From* parameter to exclude part of the interval. For example, you might set these values to discard initial transient data.
3. In the *Number of Samples* field, type the number of time domain points to use. The maximum frequency in the Fourier analysis is proportional to the *Number of Samples* parameter and inversely proportional to the difference between the starting time and the ending time.
4. Choose the *Window Type* that you want to use. If you select the *Kaiser* window type, then type in a value for the Kaiser smoothing factor. The smoothing factor must be in the range 0 <= factor <= 15, where 0 is the same as using a rectangular window.
5. In the *Window Size* field, type in the number of frequency domain points to use in the Fourier analysis.

   A larger window size results in an expectation operation over fewer samples, which leads to larger variations in the power spectral density. A small window size can smear out sharp steps in the power spectral density that might really be present.

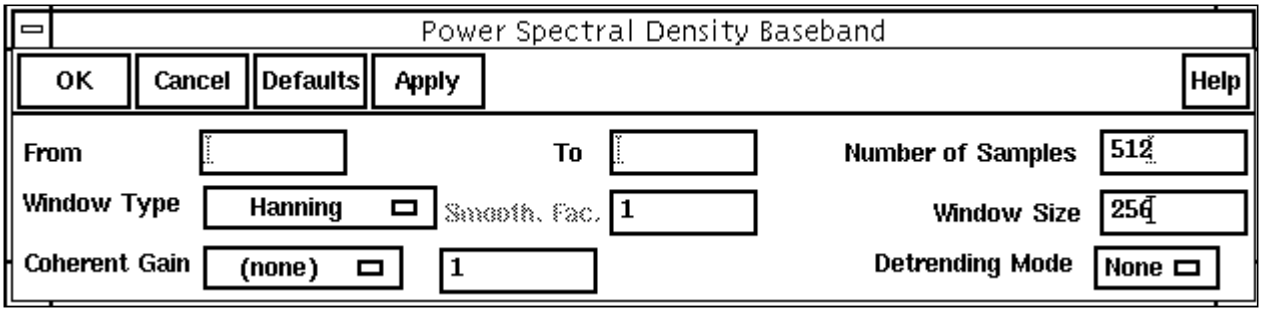
6. Choose a *Coherent Gain* factor. If you choose *magnitude*, *dB20*, or *dB10*, then enter a scaling factor. A non-zero factor scales the power spectral density by 1/(factor). Valid values for the factor are 0 < factor < 1. You can also use a value of 1 if you do not want the *Coherent Gain* factor to be used.
7. Choose which *Detrending Mode* to use.

   The *psd* function works by applying a moving windowed FFT to time-series data. If there is a deterministic trend to the underlying data, you might want to remove the trend before performing the spectral analysis. For example, consider analyzing phase noise in a VCO model. Without the noise the phase increases more or less linearly with time, so it is appropriate to set the detrending mode to *linear*. To subtract an average value, set the detrending mode to *mean*. Where the spectrum of raw data is desired, set the detrending mode to *none*.
8. Click *OK*.

Power Spectral Density Baseband (psdbb) Function

The power spectral density baseband (*psdbb*) function returns an estimate for the power spectral density of a waveform1+j * waveform2.

After you choose *psdbb* in the *Special Functions* menu of the Calculator, the Power Spectral Density Baseband form opens.



1. In the *From* field, type the starting time for the spectral analysis interval.
2. In the *To* field, type the ending time for the spectral analysis interval. You can use this parameter and the *From* parameter to exclude part of the interval. For example, you might set these values to discard initial transient data.
3. In the *Number of Samples* field, type the number of time domain points to use. The maximum frequency in the Fourier analysis is proportional to the *Number of Samples*

parameter and inversely proportional to the difference between the starting time and the ending time.

4. Choose the *Window Type* that you want to use. If you select the *Kaiser* window type, then type in a value for the Kaiser smoothing factor. The smoothing factor must be in the range 0 <= factor <= 15, where 0 is the same as using a rectangular window.

5. In the *Window Size* field, type in the number of frequency domain points to use in the Fourier analysis.

   A larger window size results in an expectation operation over fewer samples, which leads to larger variations in the power spectral density. A small window size can smear out sharp steps in the power spectral density that might really be present.

6. Choose a *Coherent Gain* factor. If you choose *magnitude*, *dB20*, or *dB10*, then enter a scaling factor. A non-zero factor scales the power spectral density by 1/(factor). Valid values for the factor are 0 < factor < 1. You can also use a value of 1 if you do not want the *Coherent Gain* factor to be used.

7. Choose which *Detrending Mode* to use.

   The *psdbb* function works by applying a moving windowed FFT to time-series data. If there is a deterministic trend to the underlying data, you might want to remove the trend before performing the spectral analysis. For example, consider analyzing phase noise in a VCO model. Without the noise the phase increases more or less linearly with time, so it is appropriate to set the detrending mode to *linear*. To subtract an average value, set the detrending mode to *mean*. Where the spectrum of raw data is desired, set the detrending mode to *none*.

8. Click *OK*.

Rise Time Function

The *riseTime* function computes the riseTime of the buffer expression, that is, the time required for the waveform to rise from `theta1` (*Percent Low*) to `theta2` (*Percent High*) of the difference between the specified *Initial Value* and the specified *Final Value*.

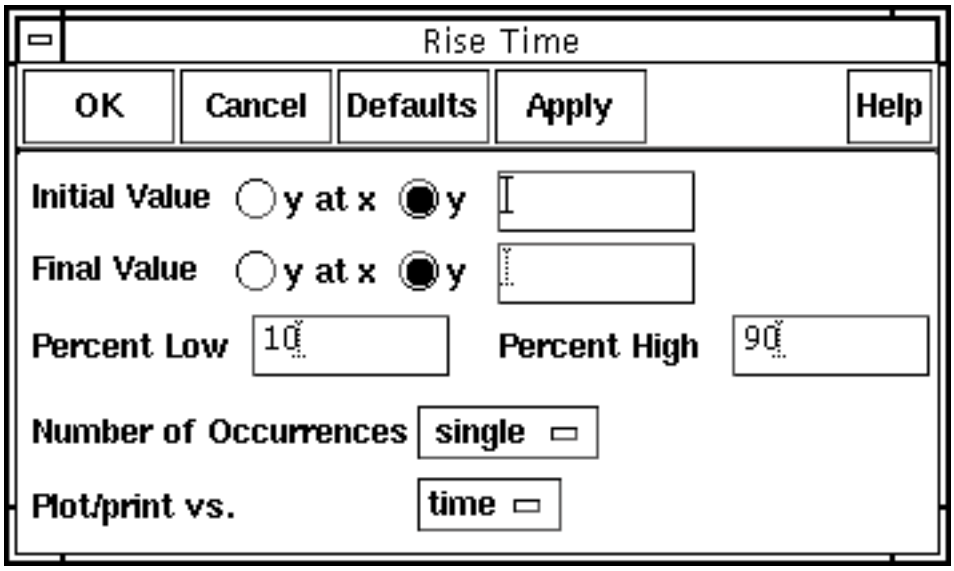
You can calculate the start and end of a riseTime event as follows:

```
Start = Initial Value + Low/100 (Final Value - Initial Value)
End = Initial Value + High/100 (Final Value - Initial Value)
```

To retrieve one or multiple occurrences of riseTime in a waveform:

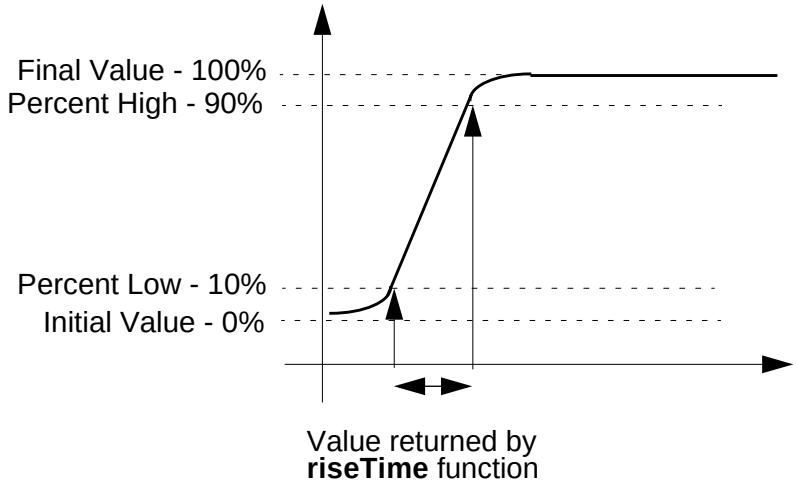
1. Enter an expression in the Calculator buffer.

2. Select *riseTime* from the *Special Functions* menu.



3. Set *Initial Value* and *Final Value* in the Rise Time form using either of the following ways:

- Use the *y* option for *Initial Value* and *Final Value* and enter the y values directly or a horizontal marker name.
- Use the *y at x* option for *Initial Value* and *Final Value* to enter an x value or vertical marker name.



For waveforms with multiple rise and fall edges, you should isolate edges of interest by using the Clip Function or enter values for y at x rather than entering y values directly in the Rise Time form.

4. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of a riseTime event for a given waveform. If you select *multiple*, you can retrieve all occurrences of riseTime for a given waveform, which you can later plot or print.

5. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve riseTime data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   riseTime(VT("/out") 0.5 nil 4.5 nil 10 90 t "time") (s)
--------   --------
12.22n     32.97n
4.012u     32.96n
8.012u     32.96n
```

   Here, time values refer to each timepoint on the waveform where the riseTime event occurs.

   The *cycle* option helps you retrieve riseTime data against cycle numbers. For example:

```
cycle      riseTime(VT("/out") 0.5 nil 4.5 nil 10 90 t "cycle") (s)
--------   --------
1          32.97n
2          32.96n
3          32.96n
```

   Here, cycle numbers refer to the n'th occurence of the riseTime event in the input waveform.

   Note: The value for *Plot/print vs.* is ignored when *Number of Occurrences* is specified as *single*.

6. Click *OK*.

Root-Mean-Square (rms) Function

The *rms* function computes the root-mean-square value of the expression f(x), over the specified range of x. This is the square root of the integral of the expression squared over the specified range, divided by the range.

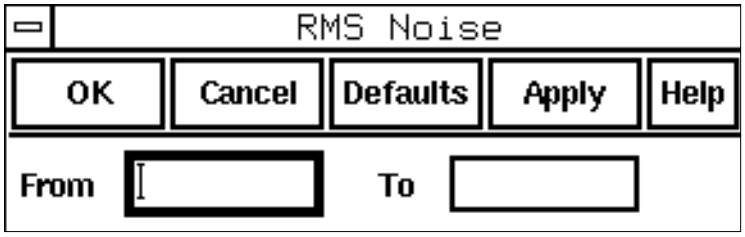
For example, if `y=f(x)`,

To compute the rms value of the expression over a smaller range, use the clip function inside the *rms* function.

$$rms(y) = \sqrt{\frac{\int_{from}^{to} f(x)^2 dx}{to - from}}$$

Root-Mean-Square (rms) Noise Function

The *rmsNoise* function computes the integrated root-mean-square of the total output noise over the bandwidth specified in hertz in the *From* and *To* fields.



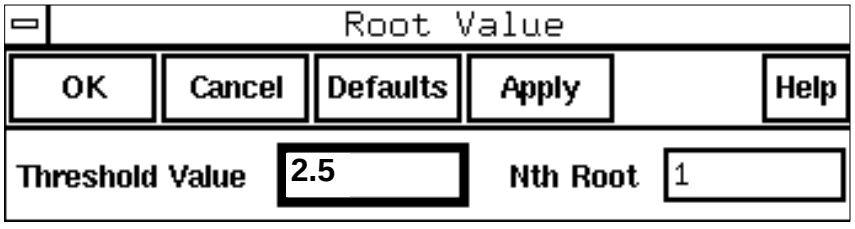
Note: To plot the squared noise voltage versus frequency instead, use the Results – Plot Noise command in the Simulation window.

Root Function

The *root* function computes the value of x at which f(x) equals the specified threshold.

1. Select *root*.

   The Root Value form appears.



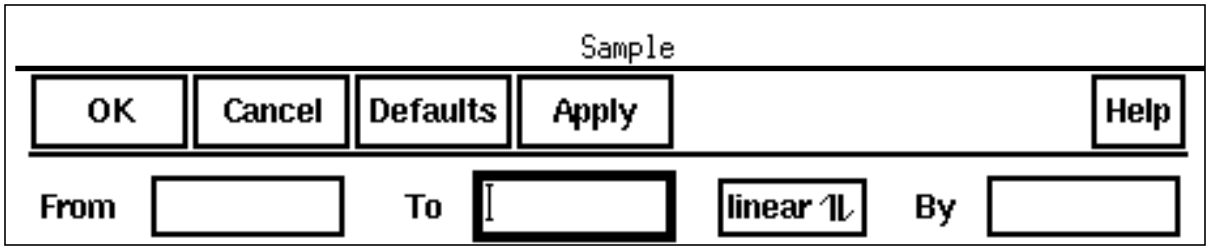
2. In *Threshold Value*, enter the waveform value at which to compute the root value.
3. Enter the *root* you want to see.

Sample Function

The *sample* function samples a waveform at the interval you specify. You can use this function to speed up plotting of waveforms that have many data points.

1. Define a waveform in the calculator buffer.
2. Select *sample*.

   The Sample form appears.



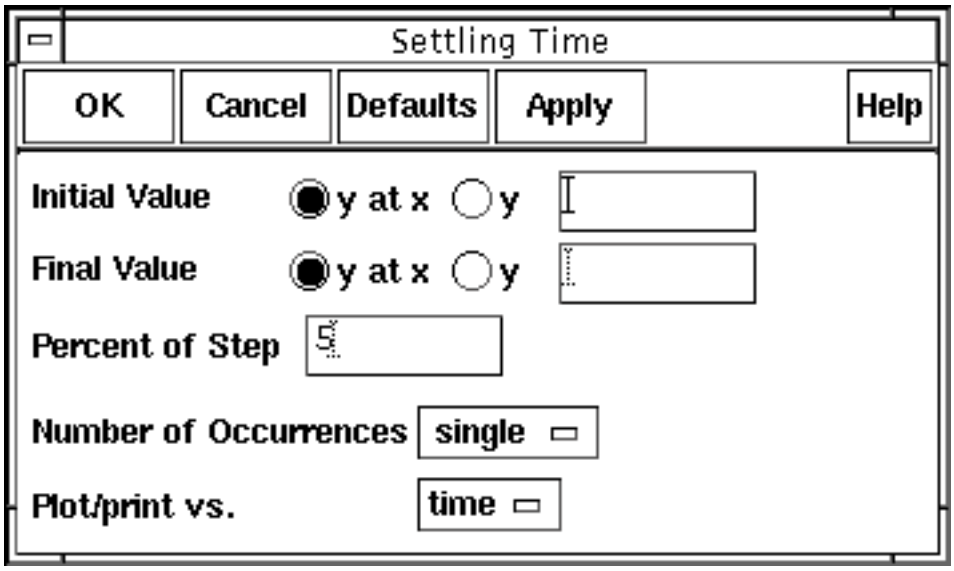
3. Specify the range and increment.

If you sample a waveform beyond its range, you get the final value of the waveform. You can use this function to demodulate a signal. Consider an AM modulated sine wave. Assume the carrier frequency is 1 GHz, and the modulation frequency is 1 MHz. If the waveform is sampled every 1 ns, the resulting signal will be cleanly demodulated (the 1 GHz carrier is completely eliminated by the sampling).

Settling Time Function

The *SettlingTime* function calculates one or multiple occurrences of the time by which the signal settles within the specified percent of step of the difference beween the *Final Value* and *Initial Value* from the *Final Value*.

1. Select *settlingTime*. The Settling Time form appears.



2. Set the *Initial* and *Final Value* in the Settling Time form as follows:

- Use the *y* option for *Initial Value* and *Final Value* and enter the y values directly or a horizontal marker name.
- Use the *y at x* option for *Initial Value* and *Final Value* to enter an x value or vertical marker name.

3. Enter the value for *Percent of Step*.
4. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of a settlingTime event for a given waveform. If you select *multiple*, you can retrieve all occurrences of settlingTime for a given waveform, which you can later plot or print.
5. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve settlingTime data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)    settlingTime(VT("/out") 0.5 nil 4.95 nil 5 t "time") (s)
--------    --------
7.862n      53.36n
4.008u      4.053u
8.008u      8.053u
```

Here, time values refer to each timepoint on the waveform where the settlingTime event occurs.

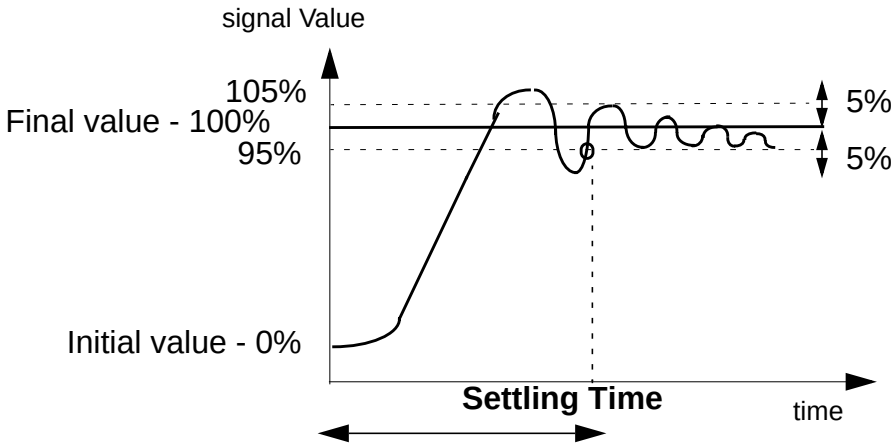
The *cycle* option helps you retrieve settlingTime data against cycle numbers. For example:

```
cycle      settlingTime(VT("/out") 0.5 nil 4.95 nil 5 t "cycle") (s)
-----      --------
1          53.36n
2          4.053u
3          8.053u
```

Here, cycle numbers refer to the n'th occurence of the settlingTime event in the input waveform.

Note: The value for *Plot/print vs.* is ignored when *Number of Occurrences* is specified as *single*.

6. Click *OK*.



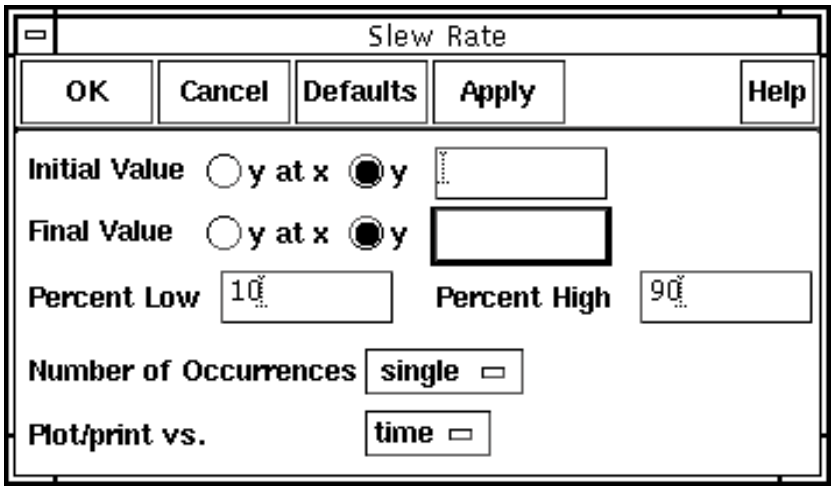
Slew Rate Function

The *slewRate* function computes the average rate at which the buffer expression changes from percent low to percent high of the difference between *Initial Value* and *Final Value*.

To retrieve one or multiple occurrences of slewRate in a waveform:

1. Enter an expression in the Calculator buffer.

2. Select *slewRate* from the *Special Functions* menu.



3. Set *Initial Value* and *Final Value* using either of the following ways:

   - Use the *y* option for *Initial Value* and *Final Value* and enter the y values directly or a horizontal marker name.
   - Use the *y at x* option for *Initial Value* and *Final Value* to enter an x value or vertical marker name.

   For waveforms with multiple rise and fall edges, you should isolate edges of interest by using the Clip Function or enter values for y at x rather than entering y values directly in the Slew Rate form.

4. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of a slewRate event for a given waveform. If you select *multiple*, you can retrieve all occurrences of slewRate for a given waveform, which you can later plot or print.

5. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve slewRate data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)    slewRate(VT("/out") 0.5 nil 4.5 nil 10 90 t "time")
--------    --------
12.22n      97.07M
4.012u      97.09M
8.012u      97.09M
```

   Here, time values refer to each timepoint on the waveform where the slewRate event occurs.

   The *cycle* option helps you retrieve slewRate data against cycle numbers. For example:

```
cycle      slewRate(VT("/out") 0.5 nil 4.5 nil 10 90 t "cycle")
-----      --------
1          97.07M
2          97.09M
3          97.09M
```

Here, cycle numbers refer to the n'th occurrence of slewRate computation for the input waveform.

Note: The value for *Plot/print vs.* is ignored when *Number of Occurrences* is specified as *single*.

6. Click *OK*.

Spectral Power Function

This function plots the spectral power for a current waveform and a voltage waveform that you define. To use this function:

1. Define the voltage waveform.
2. Define the current waveform.

   When you specify the second waveform, the first waveform is passed to the stack.
3. Choose *spectralPower* in the *Special Functions* menu of the Calculator.
4. Click *plot* in the Calculator to plot the waveform.

To use this function, you must type the line below in the CIW:

`envSetVal("calculator" "oldexpr" 'boolean nil)`

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

Standard Deviation (stddev) Function

The *stddev* function computes the standard deviation of a waveform (or a family of waveforms) over its entire range. Standard deviation (stddev) is defined as the square-root of the variance where variance is the integral of the square of the difference of the expression f(x) from average (f(x)), divided by the range of x.

For example, if `y=f(x)`

If you want a different range, use the clip function to clip the waveform to the range you want.

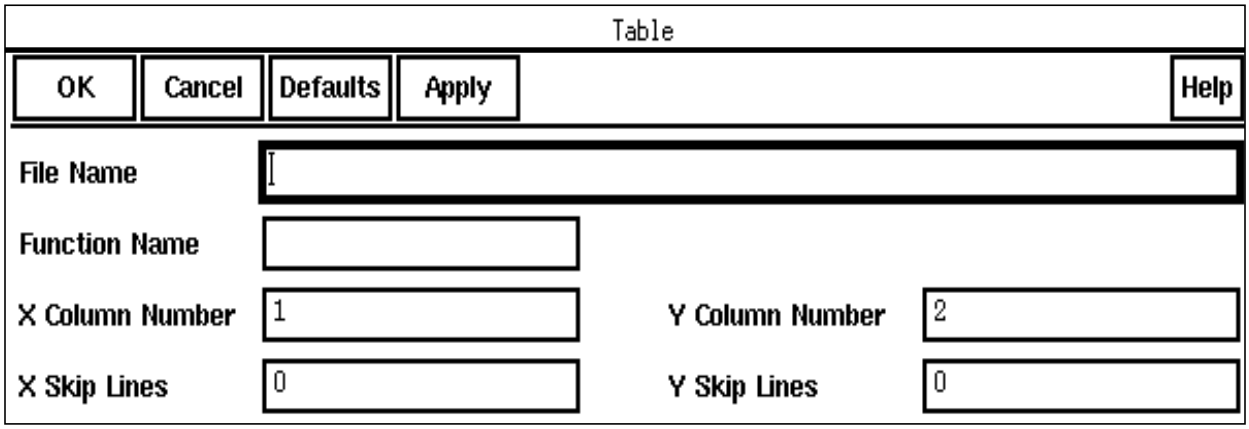
$$stddev(y) = \sqrt{\frac{\int_{from}^{to} (y - average(y))^2}{to - from}}$$

Table Function

The *table* function defines a piecewise linear function from a column of x and y values in a file. This function was previously named implicitX.

1. Enter the name of the data file and any name for the function.
2. (Optional) Enter the column numbers containing the X and Y axis data, if they are not in columns 1 and 2 respectively.
3. (Optional) Enter the number of lines to skip in each column from the top of the file before reading the data.

Note: Do not count comment lines beginning with a semicolon and blank lines in the number of lines to skip.



The X data must be real numbers increasing monotonically. The Y data can be real numbers, or complex numbers following this syntax:

```
(real_part imag_part)
```

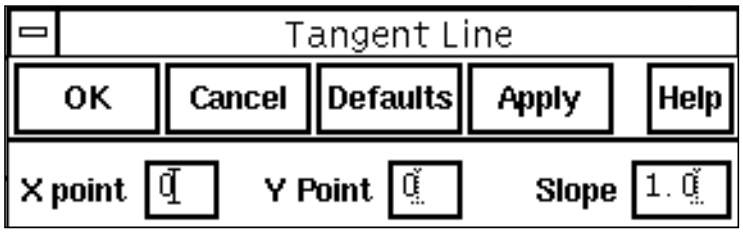
or

```
complex(real_part imag_part)
```

Tangent Function

This function plots a line that passes through x and y coordinates and the slope that you specify. To use this function:

1. Define a waveform in the buffer.
2. Choose *tangent* from the *Special Functions* menu in the Calculator.The Tangent Line form appears.



3. Type values for the *X point*, *Y Point*, and *Slope* fields.
4. Click *OK*.
5. Click *plot* in the Calculator to plot the tangent line.

To use this function, you must type the line below in the CIW

```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

Total Harmonic Distortion (thd) Function

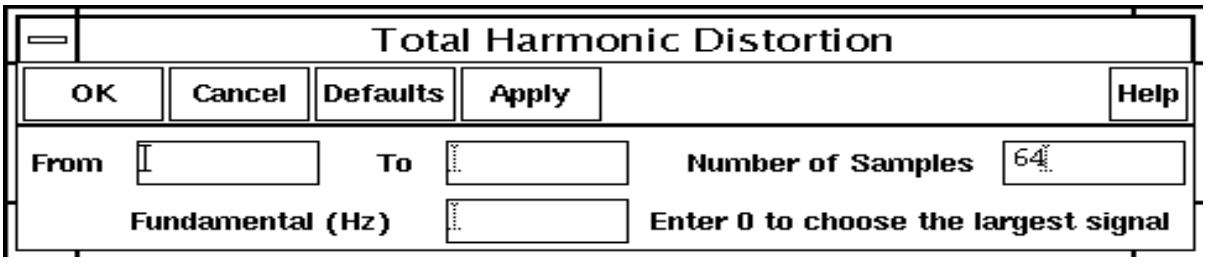
The *thd* function computes the percentage of total harmonic content of a signal with respect to the fundamental frequency. The computation uses the *dft* function (for information, see Discrete Fourier Transform (dft) Function). Assume that the *dft* function returns complex coefficients A_0, A_1..., A_f , ... Please note that fundamental frequency ***f*** is the frequency contributing to the largest power in the signal. A_0 is the complex coefficient for the DC component and A_i is the complex coefficient for the ***i***th harmonic where $i \neq 0, f$. Then, total harmonic distortion is computed as:

$$\frac{\sqrt{\sum_{i=1,\, i\neq 0,\, f} |A_i|^2}}{|A_f|} \times 100\%$$

To compute the *thd*, you need to perform the following steps:

1. Choose *thd* in the *Special Functions* menu of the calculator.

   The Total Harmonic Distortion form appears.



2. Specify the range and the number of samples.
3. Click *OK*.
4. Click Print to see the result.

The accuracy of the total harmonic distortion measurement depends on simulator options and the analysis parameters. For an accurate measurement set the following simulation options:

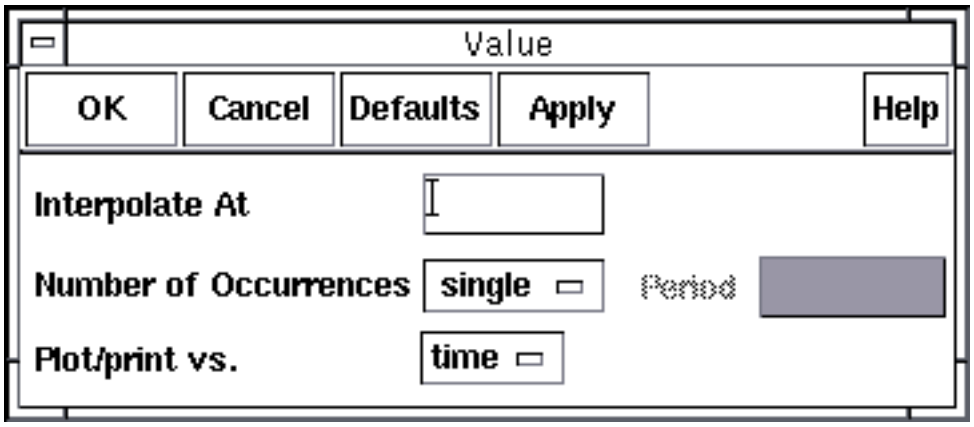
Option	Suggested Value
RELTOL	1e-5
ABSTOL	1e-13
VNTOL	3e-8
TRTOL	1
METHOD	gear
MAXORD	3

Set the simulation timestep to be 1/100th of a cycle, and simulate for ten cycles. End the simulation slightly beyond the tenth cycle. When you use the calculator, measure during the tenth cycle by specifying the beginning of the cycle as the From time and the end as the To time.

Value Function

The *value* function computes the value of the waveform at a point you specify.

1. Enter the point at which to compute the value in the *Intrapolate At* field.



2. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of an interpolated value for a given waveform. If you select *multiple*, you can retrieve all interpolated values for a given waveform, which you can later plot or print.

3. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve value data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   value(VT("/out") 2e-07 ?period 2e-07 ?xName "time") (V)
--------   --------
200n       4.965
400n       4.965
600n       4.965
800n       4.965
1u         -800.4u
```

Here, time values refer to each timepoint on the waveform where the interpolated values occur.

The *cycle* option helps you retrieve interpolated values against cycle numbers. For example:

```
cycle      value(VT("/out") 2e-07 ?period 2e-07 ?xName "cycle") (V)
--------   --------
1          4.965
2          4.965
3          4.965
4          4.965
5          -800.4u
```

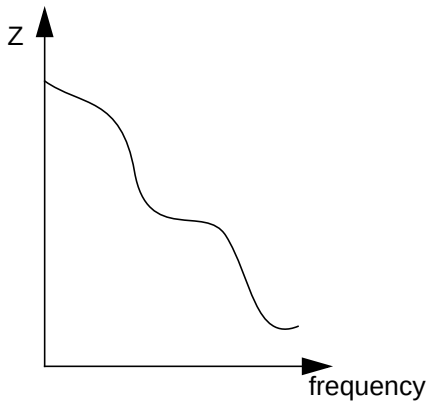
Here, cycle numbers refer to the n'th occurence of the interpolated value in the input waveform.

Note: The value for this field is ignored when *Number of Occurrences* is specified as *single*.

4. Click *OK*.

X Value (xval) Function

The *xval* function takes a single expression as an argument. It returns a waveform with the Y values equal to the X values: Y==X. This facilitates computations where the dependent variable (such as time or frequency) is needed in an expression. For example, you can use xval to compute the capacitance waveform for this curve:



$$\textit{Capacitance Waveform } C(f) = \frac{1}{2\pi Z \cdot xval(Z)}$$

Here are some syntax examples:

```
xval(VF("/IN"))
xval(VT("/net3"))
```

1. Set up the expression in the calculator buffer.
2. Select *xval*.

4

Algebraic Mode

This section describes how you use the calculator in algebraic mode.

Operators and Functions

The calculator has both algebraic and Reverse Polish Notation (RPN) modes.

Note: Each of the calculator functions has a corresponding SKILL command. For more information about SKILL calculator functions, refer to the *OCEAN Reference*.

In algebraic mode you build expressions from left to right. Whenever you click an operator or function key, the operator or function is added to the right side of the expression in the buffer. The system adds left parentheses automatically when necessary. You must enter the corresponding closing parentheses.

For example, to enter the function `(1+x)/x` in RPN mode, you use this key sequence:

```
1 enter clear x + lastx /
```

To enter `(1+x)/x` in algebraic mode, you use this key sequence:

```
1 + x () / x
```

Help is also available for

- The buffer and stack
- RPN mode

Single-Expression Functions

The following functions operate on only a single expression in the buffer.

Key	Function	Key	Function
mag	magnitude	*exp*	e^x

Key	Function	Key	Function
phase	phase (wrapped, in degrees)	*10**x*	10^x
real	real component	*y**x*	y^x
imag	imaginary component	*x**2*	x^2
ln	base-e (natural) logarithm	*abs*	|x| (absolute value)
log10	base-10 logarithm	*int*	integer value
dB10	dB magnitude for a power expression	*1/x*	inverse
dB20	dB magnitude for a voltage or current	*sqrt*	$\sqrt{x}$

Note: Selecting these functions while the buffer contains multiple expressions is an error in RPN mode because there is a space between expressions. For example, ln(expr1 expr2) is invalid because the logarithm function takes only one argument, not two.

Example: Plotting the Magnitude of a Signal

To plot the dB magnitude of a signal after an AC analysis in algebraic mode

1. Click *dB20* on the calculator.
2. Click *vf* on the calculator.
3. On the schematic, click the net you want to plot.
4. With the cursor in the Schematic window, press the *Esc* key.

   This cancels the *vf* function. Otherwise, the command stays active.
5. Press the right parenthesis key twice.

   The calculator buffer now contains the expression you want to plot.
6. Click *plot* to show the curve.

Two-Expression Functions and Operators

To use the two-expression functions and arithmetic operators

1. Enter the first operand.
2. Choose the function or operator.
3. Enter the second operand.

Key	Function in algebraic mode
*y***x*	y^x
+	Adds the buffer expression to the first stack register.
-	Subtracts the buffer expression from the first stack register.
*	Multiplies the buffer expression by the first stack register.
/	Divides the first stack register by the buffer expression.

Example: Instantaneous Power Dissipation

This example computes the instantaneous power dissipated by a resistor.

1. Click *vt*.
2. On the schematic, click the net connected to the appropriate pin of the resistor.
3. With the cursor in the Schematic window, press the *Esc* key.

   This cancels the *vt* function. Otherwise, the command stays active.
4. Click * on the calculator.
5. Click *it*.
6. Click the appropriate pin of the resistor and then press *Esc*.

   To select currents, click the square pin symbol. Do not click the wire stub.
7. Click *plot*.

Trigonometric Functions

The trigonometric functions work like the other single-expression functions. They are appended to the right of the buffer, rather than operating on the buffer.

1. Click a trigonometric function key.

   The system appends the function and a left parenthesis to the right side of the buffer.

2. Enter the operand expression you want to operate on.
3. Enter a right parenthesis.

Special Functions

The special functions help you analyze waveform data generated with calculator expressions. Some functions in the *Special Functions* menu, pop up a form where you enter the data required for the calculation. Other special functions are appended to the buffer.

Example: Average Value of a Current

This example shows how to compute the average of a current during the simulation period.

1. Choose *Special Functions – Average*.
2. Click *it* on the calculator.
3. On the schematic, click the terminal whose current you want to average.

   To select currents, click the square pin symbol. Do not click the wire stub.
4. With the cursor in the Schematic window, press the *Esc* key.

   This cancels the *it* function. Otherwise, the command stays active.
5. Enter a right parenthesis.
6. Click *print* on the calculator.

   The system displays the average value in a text window.

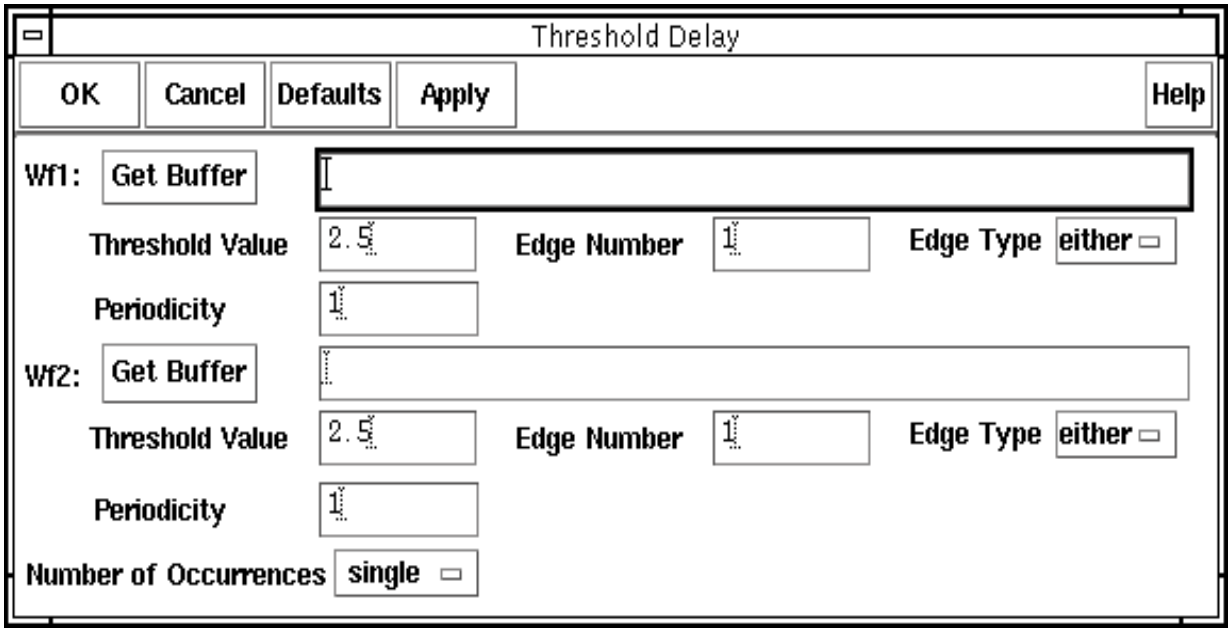
Example: Delay Function

This example shows how to append a delay special function to the calculator buffer, assuming that an expression is already in the buffer.

1. Choose *delay* from the *Special Functions* menu.

   The buffer is saved to the memory asfmem and then cleared.

   The Threshold Delay form appears.



2. To build the first expression for the delay function, do the following:

   a. Click *vt*.

   b. Click a node in the schematic.

   The expression `vt("/net1")` now appears in the buffer.

3. Click *Wf1: Get Buffer* to import the buffer expression into the Threshold Delay form.
4. Specify the threshold values, edge numbers, and edge type for the first expression.
5. To build the second waveform in the buffer, do the following:

   a. Click *clear* to clear the first expression from the buffer.

   b. The *vt* function is still active. You can now click a node in the schematic.

   The expression `vt("/net2")` now appears in the buffer.

6. Click *Wf2: Get Buffer* to import the second expression.
7. Specify the threshold values, edge numbers, and edge type for the second expression.
8. Click *OK* in the Threshold Delay form.

   The original expression in the buffer is recalled, and the delay function is appended to the right of this expression.

```
orig_expression delay(VT("/net1"),2.5,1,"either",
VT("/net2"),2.5,1,"either")
```

Algebraic Mode Special Functions

Average Function

The *average* function computes the average of a waveform over its entire range. Average is defined as the integral of the expression `f(x)` over the range of `x`, divided by the range of `x`. For example, if `y=f(x)`, `average(y)=`

$$\frac{\int_{from}^{to} f(x)dx}{to - from}$$

where `to` and `from` are the range of `x`.

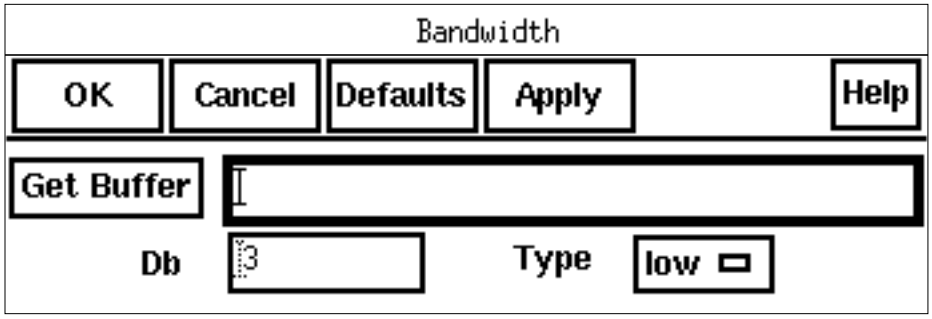
If you want a different range, use the clip function to clip the waveform to the range you want.

Bandwidth Function

The *bandwidth* function calculates the bandwidth of the waveform in the calculator buffer. Please note that the input waveform must represent a true voltage, NOT modified by a dB.

1. Set up the expression whose bandwidth you want to calculate in the buffer.
2. Select *bandwidth*.

The Bandwidth form appears.



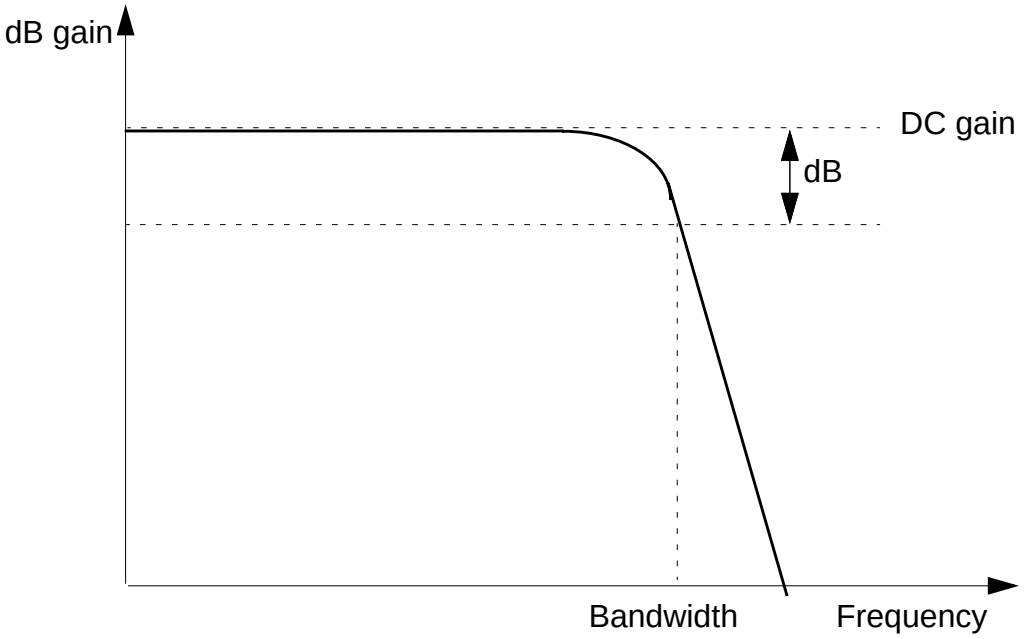
3. Click *Get Buffer*.
4. In the *Db* field, enter how far below the peak value you want to see data.
5. Choose a bandwidth response from the *Type* field.
   - low computes the bandwidth of a low-pass response.
   - high computes the bandwidth of a high-pass response.
   - band computes the bandwidth of a band-pass response.
6. Click *OK*.

Computing Low-Pass Bandwidth

The calculator computes the low-pass bandwidth by determining the smallest frequency at which the magnitude of the input waveform drops *n* decibels below the DC gain. (DC gain is obtained by zero-order extrapolation from the lowest or highest computed frequency, if

necessary.) The *dB* field specifies *n*. An error occurs if the magnitude of the input waveform does not drop *n* decibels below the DC gain.

Low-Pass Bandwidth Function

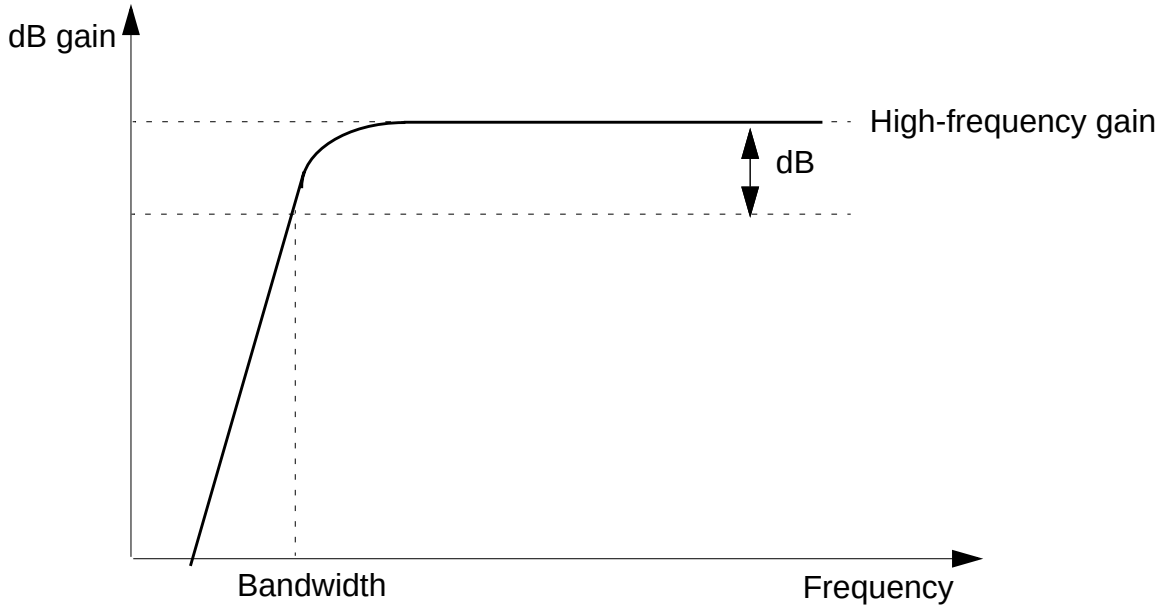


Computing High-Pass Bandwidth

The calculator computes the high-pass bandwidth by determining the largest frequency at which the magnitude of the input waveform drops *n* decibels below the gain at the highest

frequency in the response waveform. The *dB* field specifies *n*. An error occurs if the magnitude of the input waveform does not drop *n* decibels below the gain at high frequency.

High-Pass Bandwidth Function

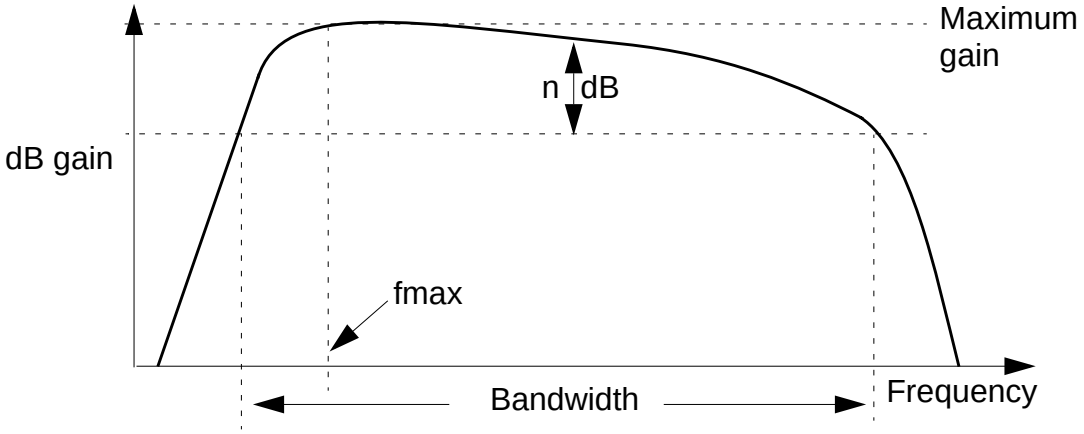


Computing Band-Pass Bandwidth

The calculator computes the band-pass bandwidth by

1. Determining the lowest frequency (f_{max}) at which the magnitude of the input waveform is maximized
2. Determining the highest frequency less than f_{max} at which the input waveform magnitude drops *n* decibels below the maximum (*n* is the number you enter in the *dB* field)
3. Determining the lowest frequency greater than f_{max} at which the input waveform magnitude drops *n* decibels below the maximum

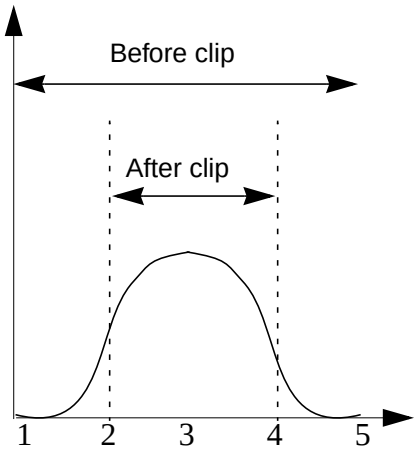
4. Subtracting the value returned by step 2 from the value returned by step 3. The value returned by step 2 or step 3 must exist.

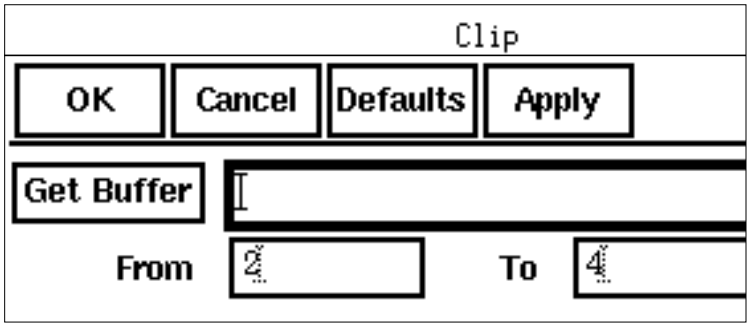


Clip Function

The *clip* function restricts the waveform defined by the buffer expression to the range entered in the *From* and *To* fields. You can use the *clip* function to restrict the range of action of other special functions of the calculator such as integ, rms, and frequency.

1. Set up the expression you want to clip in the buffer.
2. Select *clip*.
3. Click *Get Buffer*.
4. Enter the limits and click *OK*.





Note: The *clip* function does not support multi-valued functions, that is, functions that have multiple y values corresponding to a single x value.

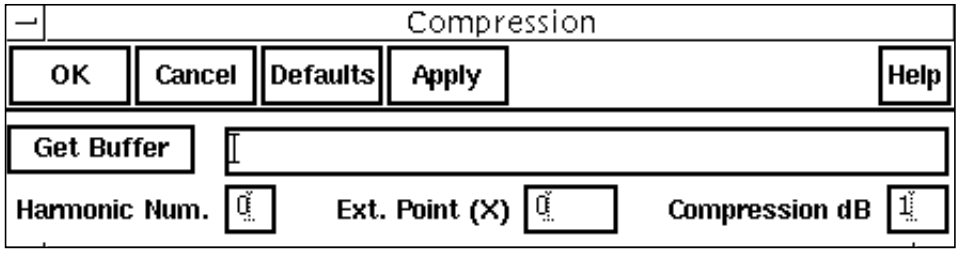
Compression Function

This function returns the *Nth* compression point value of a waveform at the extrapolation point that you specify. To use this function:

1. Set up the ne600p mixer cell from `dfII/samples/artist/rfExamples` library.The design variable frf should be set to 920MHz.
2. Ensure that the sourcetype on the rf port is 'sine' and the Amplitude (dBm) is set to prf.
3. Set up a PSS analysis. The beat frequency should be set to 40MHz. Set the number of harmonics to 2 (only two harmonics are required to determine the 1 dB compression point). Sweep the prf parameter from -30 to 10 in 10 linear steps.
4. Set the Model Library path to include the `dfII/samples/artist/models/spectre/rfModels.scs` file.
5. After running the simulation, call up the Waveform Calculator and the Results Browser. Click on *schematic->psf->Run1->sweeppss_pss_fd-sweep->sweepVariable->prf->10->sweeppss-004_pss-fd.pss->Pif* with the left mouse button. The following will appear in the calculator buffer:

   ```
   v( "/Pif" ?result "sweeppss_pss_fd-sweep" ?resultsDir "~/simulation/ne600p/spectre/schematic" ).
   ```

6. Click on *Special Functions -> compression*. The *Compression* form will be displayed.



7. Click *Get Buffer* in the Compression form.
8. Enter *Harmonic number=2.* This is the second harmonic of the 40 MHz fundamental frequency, which is the IF frequency (80MHz).
9. Enter *Ext. Point (X)* = `-25` field to specify the extrapolation point of the waveform. The extrapolation point is the X axis value.

10. Enter *Compression dB* = `1` to specify the compression coefficient (N).
11. Click on the *OK* button.
12. Click the *Evaluate Buffer* button in the Calculator. The result appears in the Calculator display.

To use this function, you must type the line below in the CIW

```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

CompressionVRI Function

This function performs an Nth compression point measurement on a power waveform.

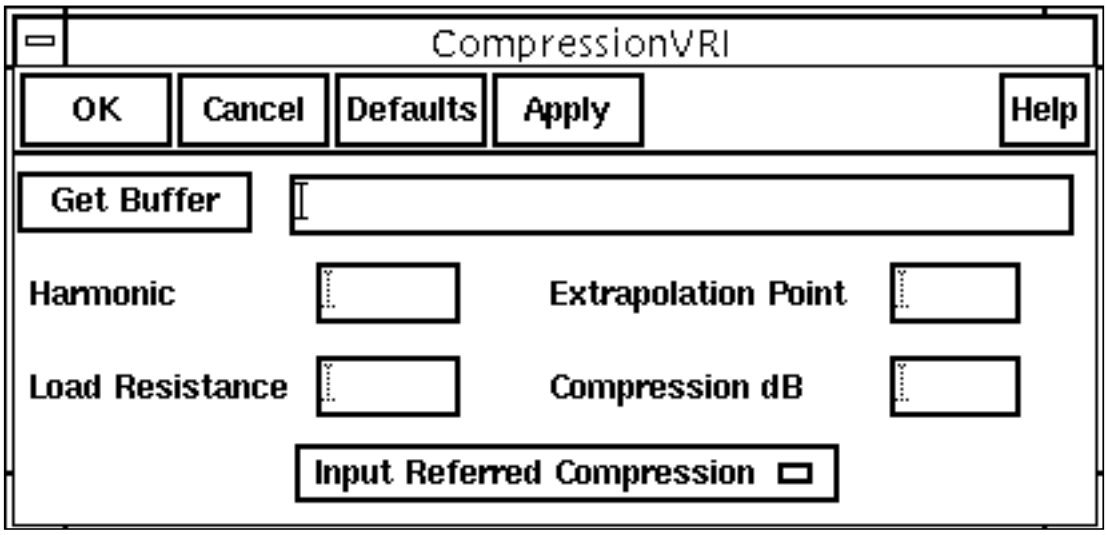
Use this function to simplify the declaration of a compression measurement. This function extracts the specified harmonic from the input waveform(s), and uses `dBm(spectralPower((i or v/r),v))` to calculate a power waveform. The function then passes this power curve and the remaining arguments to the *compression* function to complete the measurement.

The *compression* function uses the power waveform to extrapolate a line of constant slope (dB/dB) according to a specified input or output power level. This line represents constant small-signal power gain (ideal gain). The function then finds the point where the power waveform drops N dB from the constant slope line and returns either the x coordinate (input referred) or y coordinate (output referred) value.

To use this function:

1. Define the voltage waveform in the buffer.
2. Choose *compressionVRI* in the *Special Functions* menu.

The Compression form opens.



3. Click *Get Buffer* in the CompressionVRI form.
4. Type a value in the *Harmonic* field to specify the harmonic index of the waveform.
5. Type a value in the *Extrapolation Point* field to specify the extrapolation point for the waveform. The default value is the minimum x value of the input voltage waveform.

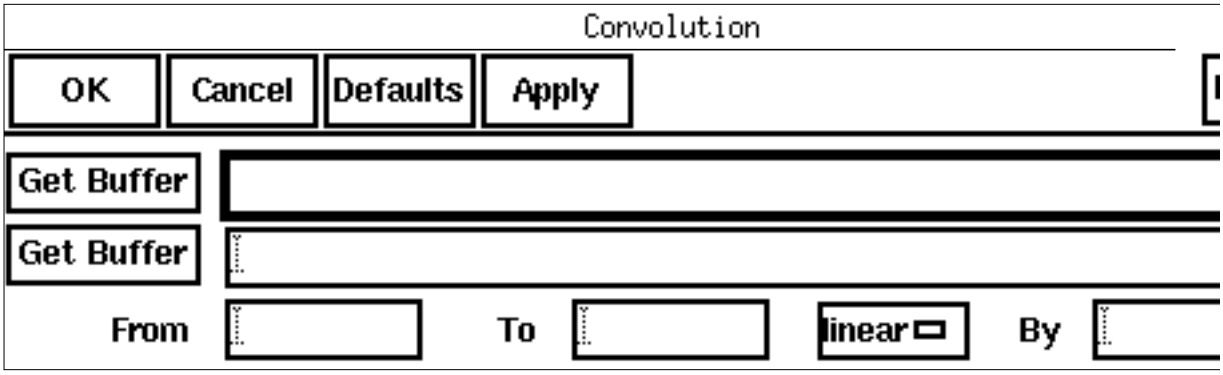
   The extrapolation point is the coordinate value in dBm that indicates the point on the output power waveform where the constant-slope power line begins. This point should be in the linear region of operation.
6. Type a numerical value in the *Load Resistance* field. The default value is 50.
7. In the *Compression dB* field, type the delta (in dB) between the power waveform and the ideal gain line that marks the compression point. The default value is 1.
8. Choose whether the measurement is for *Input Referred Compression* or *Output Referred Compression*.
9. Click *OK*.

Convolution (Convolve) Function

The *convolve* function computes the convolution of two waveforms.

1. In the *Special Functions* menu of the calculator, choose *convolve*.

The Convolution form appears.



2. In the Convolution form, define the first waveform in the upper buffer and click the upper *Get Buffer* button.
3. Define the second waveform in the lower buffer and click the lower *Get Buffer* button.
4. Specify the range, scale, and increment by values and click *OK*.

Convolution is defined as

$$\int_{from}^{to} f1(s)f2(t-s)ds$$

`f1` and `f2` are the functions defined by the first and second waveforms.

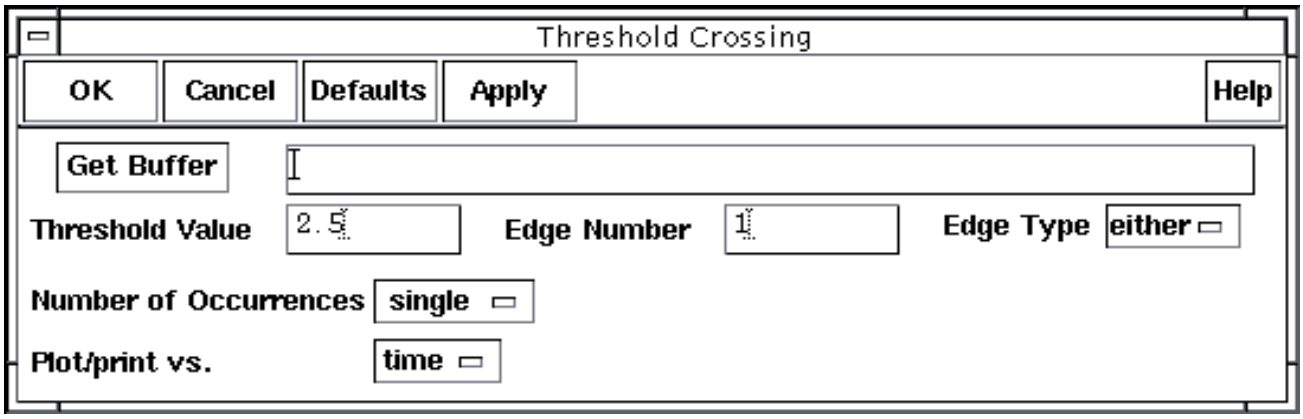
Note: The *convolve* function is numerically intensive and might take longer than the other functions to compute.

Threshold Crossing (cross) Function

The threshold crossing or *cross* function computes the x-axis value *xcross* at which the nth crossing of the specified edge type of the threshold value occurs.

1. Select *cross*.

The Threshold Crossing form appears.



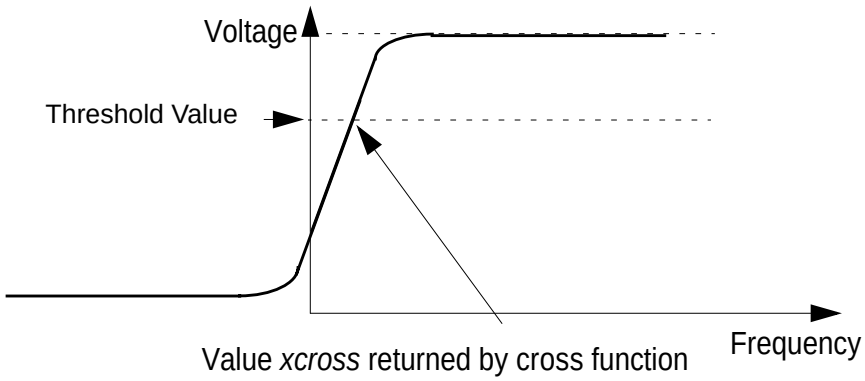
2. Set up the expression in the calculator buffer.
3. Click *Get Buffer* in the Threshold Crossing form.

   The expression is copied to the Threshold Crossing form buffer.
4. Enter the threshold value of the waveform at which to perform the calculation.
5. In *Edge Number*, enter the number of the crossing at which to perform the calculation.

   The integer you enter specifies which crossing is returned. (For example, 1 specifies the first crossing, and 2 specifies the second crossing.)

   If you specify a positive integer, the count starts at the smallest *x* value of the waveform, and the search is in the direction of increasing *x* values. If you specify a negative integer, the count starts at the largest *x* value of the waveform, and the search is in the direction of decreasing *x* values. If you enter 0, all the crossings found are returned in a list.

6. Select an *Edge Type* to determine the crossing as the rising edge, falling edge, or either edge.



7. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of a crossing event for a given waveform. If you select *multiple*, you can retrieve all occurrences of crossing for a given waveform, which you can later plot or print. The field *Edge Number* is dsabled if *Number of Occurrences* is specified as *multiple*.

8. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve crossing data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   cross(VT("/out") 2.5 0 0 t "time") (s)
--------   --------
27.87n     27.87n
823.3n     823.3n
2.027u     2.027u
2.806u     2.806u
```

Here, time values refer to each timepoint on the waveform where the crossing event occurs.

The *cycle* option helps you retrieve crossing data against cycle numbers. For example:

```
cycle      cross(VT("/out") 2.5 0 0 t "cycle") (s)
--------   --------
1          27.87n
2          823.3n
3          2.027u
4          2.806u
```

Here, cycle numbers refer to the n'th occurence of the crossing event in the input waveform.

Note: The value for this field is ignored when *Number of Occurrences* is specified as *single*.

9. Click *OK*.

dBm Function

The *dBm* function performs the operation

`dB10(x) + 30`

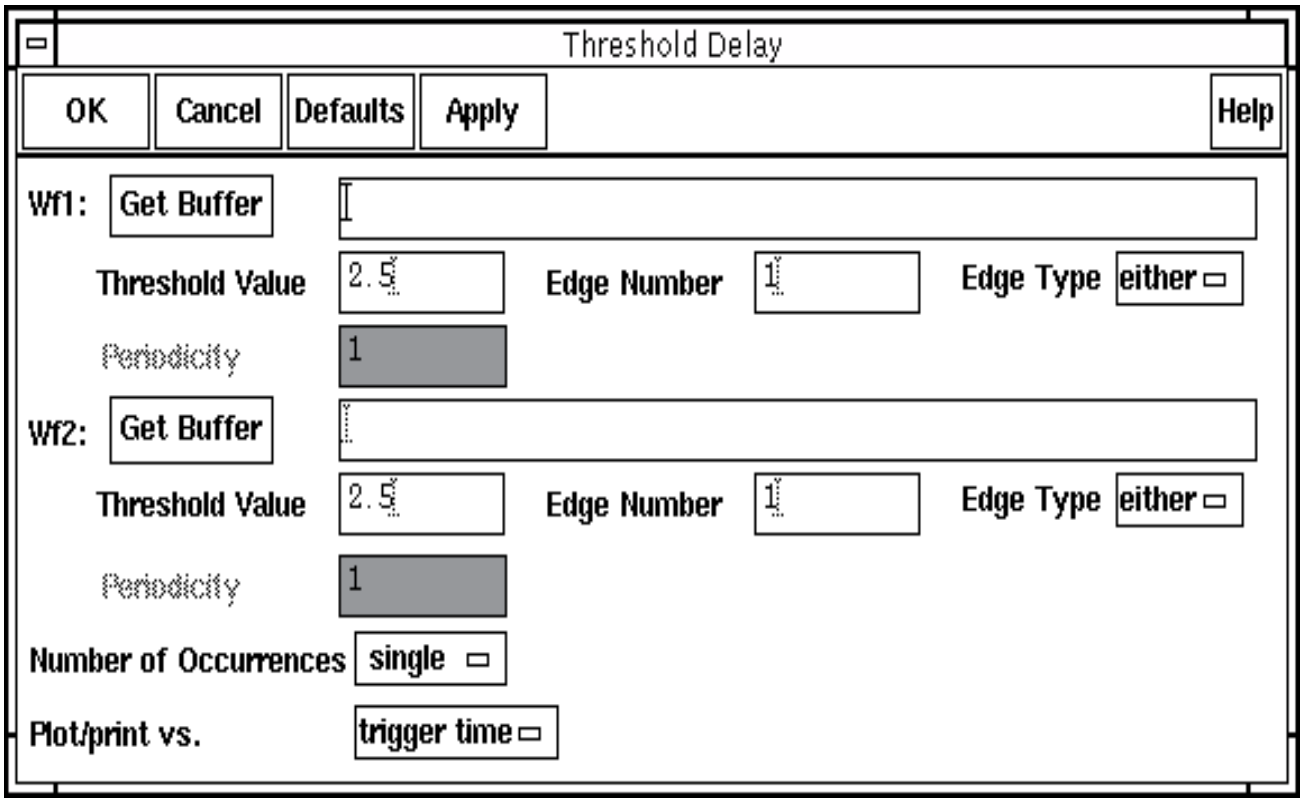
1. Select *dBm*.
2. Enter the value for `x` and close the parentheses.

Delay Function

The *delay* function computes the delay between two points or multiple sets of points in a waveform by using the cross function.

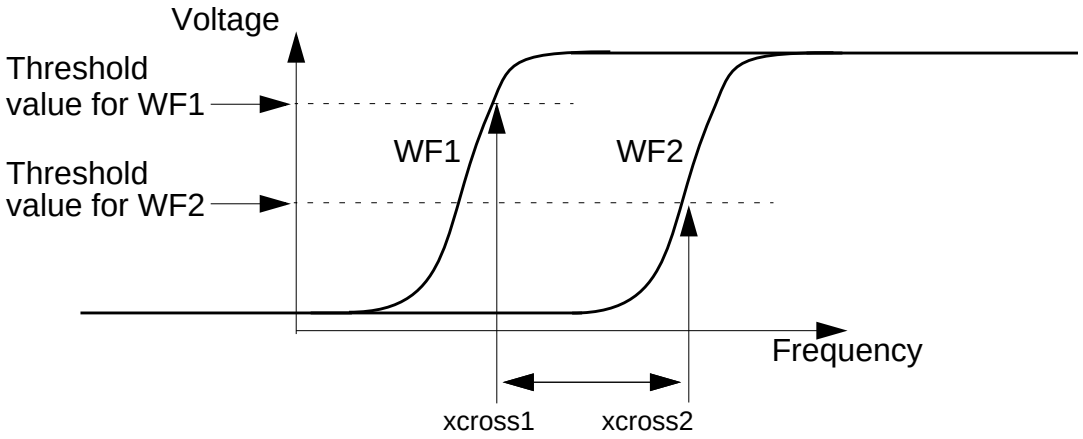
1. Select *delay*.

The Threshold Delay form appears.



2. Enter the first waveform into the upper buffer and click *Wf1: Get Buffer*.

3. Enter the second waveform into the lower buffer and click *Wf2: Get Buffer*.



delay = xcross2 - xcross1

4. For both waveforms, specify *Threshold Value*, *Edge Number*, *Edge Type* and *Periodicity* for the specified edge.
5. For *Number of Occurrences*, select *single* or *multiple* to indicate whether you want to calculate delay for one or more occurrences. If you select *multiple*, the delay is computed for the specified edges of the waveforms and also for the edges occuring at the periodic intervals (specified as *Periodicity*) for each waveform.
6. Specify *Plot/print vs.* as *trigger time*, *target time*, or *cycle*. The default setting is *trigger time*, which helps you retrieve delay data against trigger time. Trigger time is the time when the first waveform crosses the specified threshold value.

   For example:

```
cycle      delay(VT("/out") 1.5 1 "rising" VT("/out") 1.5 2 "rising" 1 1 t
           "trigger") (s)
--------   --------
18.19n     1.999u
2.018u     2.001u
4.018u     1.999u
6.018u     2.001u
```

   Here, time values refer to each timepoint on the waveform where the threshold delay occurs.

Target time is the time when the second waveform crosses the specified threshold value. For example:

```
cycle      delay(VT("/out") 1.5 1 "rising" VT("/out") 1.5 2 "rising" 1 1 t
           "target") (s)
--------   --------
2.018u     1.999u
4.018u     2.001u
6.018u     1.999u
8.018u     2.001u
```

The *cycle* option helps you retrieve threshold delay data against cycle numbers. For example:

```
cycle      delay(VT("/out") 1.5 1 "rising" VT("/out") 1.5 2 "rising" 1 1 t
           "cycle") (s)
--------   --------
1          1.999u
2          2.001u
3          1.999u
4          2.001u
```

Here, cycle numbers refer to the n'th occurence of the delay event in the input waveform.

Note: The value for this field is ignored when *Number of Occurrences* is specified as *single*.

7. Click *OK*.

Derivative (deriv) Function

The *deriv* function computes the derivative of an expression. You can plot the resulting waveform.

1. Select *deriv*.
2. Enter the expression and closing parenthesis into the calculator buffer.

Note: After the second derivative, the results become inaccurate because the derivative is obtained numerically.

Discrete Fourier Transform (dft) Function

The tool which converts a temporal (time domain) description of a signal (real or complex) into one, in terms of its frequency components is called the Fourier transform. DFT (Discrete Fourier Transform) is the discrete formulation of the Fourier transform, which takes such

regularly spaced data values (samples in time domain), and returns the value of the Fourier transform for a set of values in frequency domain which are equally spaced. Most of the time, however, we work on real-valued signals only.

Consider a complex series (signal) *w(k)* with *N* samples of the form

w(0), w(1), w(2), …, w(k), …, w(N-1)

Further, assume that the series outside the range *0, N-1* is extended *N*-periodic, that is, *w(k) = w(k+N)* for all *k*. The DFT of this series will be denoted *W(n)*, will also have *N* samples and will be defined as:

$$W(n) = \frac{1}{N}\sum_{k=0}^{N-1} w(k)\left(e^{-2\pi i k\frac{n}{N}}\right)$$ where *n = 0, …. , N-1*

Note:

- The first sample *W(0)* of the transformed series is the DC component, more commonly known as the average of the input series.
- The DFT of a real series results in a symmetric series about the Nyquist frequency (described below).
- The highest positive (or negative) frequency sample is called the Nyquist frequency. This is the highest frequency component that should exist in the input series for the DFT to receive ‘unpredictable’ results. More specifically, if there are no frequencies above Nyquist frequency, the original signal can be exactly reconstructed from the samples. The Nyquist Theorem (or Shannon’s Sampling Theorem) exactly specifies this, that for a band limited signal, you must sample at a frequency over twice the maximum frequency of the signal, to reconstruct it from the samples.

While the DFT transform above can be applied to any complex valued series, in practice for large series it can take considerable time to compute, the time taken being proportional to the square of the number of points (samples) in the series. A much faster algorithm has been developed by Cooley and Tukey called the FFT (Fast Fourier Transform). The only requirement of the most popular implementation of this algorithm (Radix-2 Cooley-Tukey) is that the number of points in the series be a power of *2* i.e. $N=2^n$.

Given *N* input points, the FFT returns *N* frequency components, of which the first (*N/2 +1*) are valid. (The other components are mirror images and are considered invalid since the frequencies they represent do not satisfy the Nyquist Theorem above.) They start with the DC component, and are spaced apart by a frequency of *(1 / (n deltaT))*. The magnitude of the complex number returned is the frequency's relative strength.

The *dft* function computes the discrete Fourier transform of the buffer by FFT algorithm where *deltaT= (t2-t1) / N*. The waveform is sampled at the following *N* timepoints:

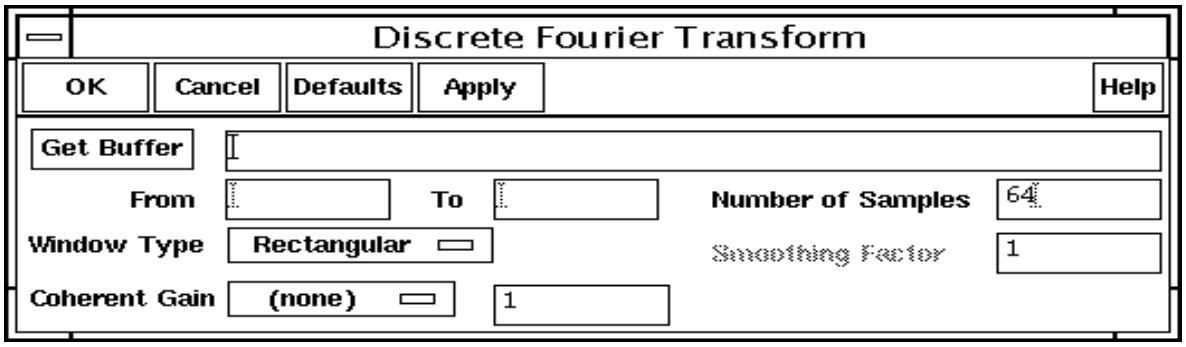
```
t1, t1 + deltaT, t1 + 2 * deltaT, º, t1 + (N - 1) * deltaT
```

The output of *dft()* is a frequency waveform, *W(f)*, which has (*N/2 + 1*) complex values: the dc term, the fundamental, and (*N/2 - 1*) harmonics.

Note: The last time point, (*t1 + (N - 1) * deltaT*), is (*t2 - deltaT*) rather than *t2*. The *dft* function assumes that *w(t1)* equals *w(t2)*. To use the *dft* function

1. Select *dft*.

   The Discrete Fourier Transform form appears.



2. Specify the range over which you want to compute the transform.

   Be sure to cover at least one complete period of your slowest frequency.

3. Enter the number of samples you want to take in expanding the Fourier transform.

   This number should be a power of *2*. If it is not, the system increases the value to the next higher power of *2*. Sample at a rate that is at least twice your highest frequency component (the Nyquist rate). Pick a sampling rate high enough that closely spaced frequency components can be resolved.

4. Select the *Window Type* option.

   For more information, see the table of window type option values later in this section.

5. Specify the *Smoothing Factor* (for the Kaiser window type only).

The *Smoothing Factor* field accepts values from *0* to *15*. The value *0* implies no smoothing and is equivalent to a rectangular window. The default value for the *Smoothing Factor* field is *1*.

6. Click *OK*.

When you run the transient analysis, keep the maximum time step small enough to represent the highest frequency component accurately. The maximum time step should be smaller than the sampling period that you use for the discrete Fourier transform (DFT) of the time domain waveform. The samples in the DFT will either hit a data point (calculated exactly by the simulator) or an interpolated point between two data points.

Choosing a maximum time step during transient simulation that is smaller than the DFT sampling period ensures that sampling maintains a resolution at least equal to that of the transient time-domain waveform.

The start and stop times should not coincide with the boundaries of the time-domain waveform. The boundary solutions might be imprecise and generate incorrect results if used in other calculations.

One of the uses of fast Fourier transform (FFT) windowing is to reduce discontinuities at window edges caused by having a nonintegral number of periods of a signal in a window. This removes the abrupt edges, making them fall off smoothly to zero, and can improve the validity of the FFT components obtained. You can also use FFT windowing to ‘dig out’ the details of signal components that are very close Gin frequency or that consist of both large and small amplitudes.

The following table was obtained from the book *The FFT, Fundamentals and Concepts* by R. W. Ramirez, Prentice Hall, 1985. As explained in this reference, the values in the table were computed from software-generated windows and might vary slightly from theoretical values. In the third column, the peak magnitude of each window is compared with that of the rectangular window. In the fourth column, the amplitude of the highest side lobe is given in decibels referenced to the major lobe peak. The fifth column contains the 3dB bandwidth of the major lobe, normalized to one over the window’s width. The last column gives the theoretical rolloff of the side lobes.

Window name	**Shape equation**	**Major lobe height**	**Highest side lobe (dB)**	**Band-width (3 dB)**	**Theor. rolloff (dB/ octave)**
Cosine4	$A=(.5(1-\cos(2\pi t/T))^2$ for t=0 to T	0.36T	-46.9	1.79/T	30

Window name	Shape equation	Major lobe height	Highest side lobe (dB)	Band-width (3 dB)	Theor. rolloff (dB/ octave)
ExtCosBell	A=0.5(1-cos(2π5t/T) for t=0 to T/10 and t=9T/10 to T A=1 for t=T/10 to 9T/10	0.9T	-13.5	0.95/T	18 (beyond 5/T)
HalfCycleSine	A=sin(2π0.5t/T) for t=0 to T	0.64T	-22.4	1.15/T	12
HalfCycleSine3	$A=\sin^3(2\pi 0.5t/T)$ for t=0 to T	0.42T	-39.5	1.61/T	24
HalfCycleSine6					
Hamming	A=.08 + .46(1-cos(2πt/T)) for t = 0 to T	0.54T	-41.9	1.26/T	6 (beyond 5/T)
Kaiser					
Parzen	$A=1 - 6(2t/T-1)^2 + 6\lvert 2t/T-1\rvert^3$ for t=T/4 to 3T/4 $A=2(1 - \lvert 2t/T-1\rvert)^3$ for t=0 to T/4 and t=3T/4 to T	0.37T	-53.2	1.81/T	24
Rectangular	A=1 for t=0 to T	T	-13.2	0.86/T	6

Sources of Errors

- *dft()* performs interpolation to determine values of *w(t)* that are not directly available from the simulator output. This interpolation can cause an inaccurate spectrum.
- If *(t2 - t1)* is not the time period of *w(t)*, the output of *dft()* might be misleading.

- If the simulator-generated values are inaccurate, *dft()* returns a frequency waveform with many 'insignificant' harmonics. You can minimize the amplitudes of these harmonics by increasing the accuracy of the simulator.

Discrete Fourier Transform Baseband (dftbb) Function

The *dftbb* function computes the discrete Fourier transform (fast Fourier transform) of a complex signal *z(t)= x(t) + j*y(t)*:

$$Z(n)= ReZ(n) +j*ImZ(n)=SUM[\ \sum_{k=0}^{N-1} z(k)*exp(-j*theta*n*k)],$$

where *theta=2*Pi/N; n=0, 1, ..., N-1*.

Both waveforms are sampled at the following N timepoints:

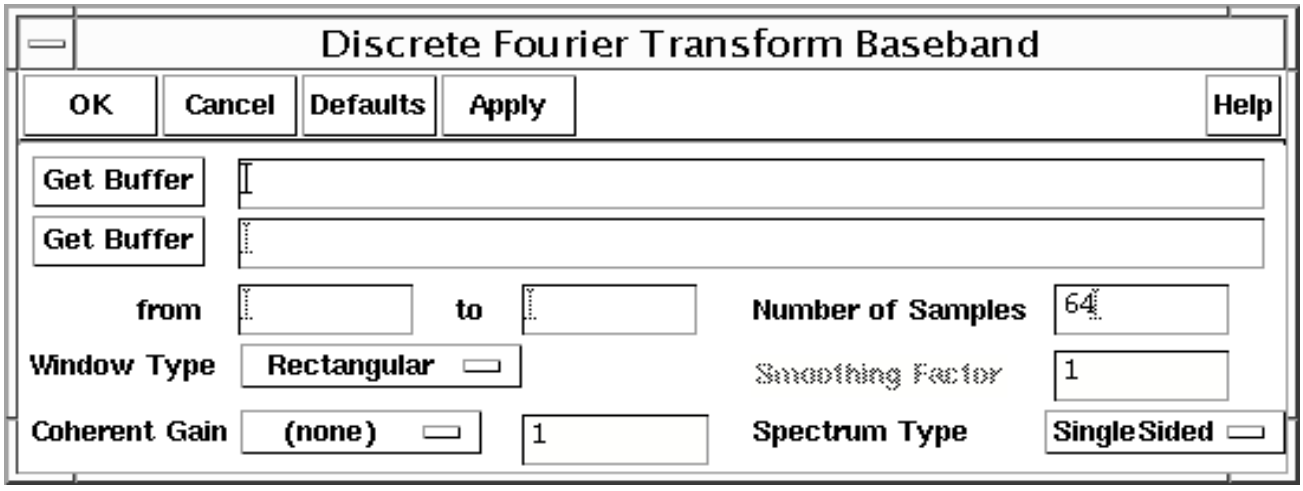
*t1, t1 + deltaT, t1 + 2 * deltaT, ..., t1 + (N - 1) * deltaT.*

The output of *dftbb(waveform1, waveform2)* are N complex values.

The above definition is for single-sided output waveforms. This holds true for double-sided output waveforms except that the previous output waveform is translated so that *n* varies from *-N/2 to (N/2)-1*.

1. Choose *dftbb* from the *Special Functions* menu in Calculator.

   The Discrete Fourier Transform Baseband form appears.



2. Click the first *Get Buffer* button to import the waveform from the Calculator buffer into the first buffer of the *dftbb* pop-up form.

3. Click the second *Get Buffer* button to import the waveform from the Calculator buffer into the second buffer of the *dftbb* pop-up form.
4. Fill in the values, as required. See the previous topic, on the *dft* function, to know more about these options.

   Note: For *Window Type*, only *Rectangular* is supported. AWD issues a warning if you create an expression by hand for a different window type and switches to the default *Rectangular* window type.
5. Click *OK*.

You get an expression for *dftbb* as *plotted*, *evaluated*, or *printed*.

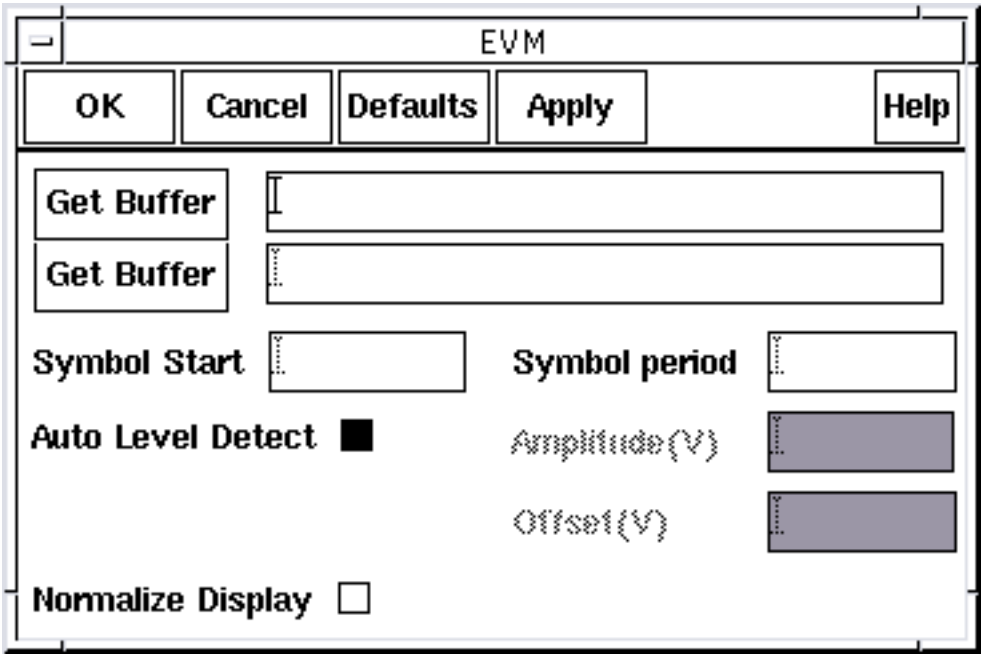
evmQpsk Function

Error Vector Magnitude (EVM) is a useful measurement to describe the overall signal amplitude and phase modulated signal quality. It is based on a statistical error distribution normalized from an ideal digital modulation. Quadrature Phase Shift Keying (QPSK) is a typical modulation scheme where EVM is useful. EVM is a signal path fidelity figure that could be affected by distortions such as background noise, amplitude modulated noise, compression effects, phase noise, and Inphase (I) & Quadrature-phase (Q) mismatches. Each of these effects have signature characteristics and can be graphically plotted in a constellation for inspection.

The *evmQpsk* function processes the I and Q waveform outputs from the transient simulation run to calculate the EVM and plot the I versus Q scatterplot. The EVM is calculated by detecting the I and Q signal levels corresponding to the four possible I and Q symbol combinations and calculating the difference between the actual signal level and the ideal signal level.

1. Choose *evmQpsk* from the *Special Functions* menu in Calculator.

The EVM form appears.



2. Click the first *Get Buffer* button to import the I signal from the Calculator buffer into the first buffer of the *EVM* form.
3. Click the second *Get Buffer* button to import the Q signal from the Calculator buffer into the second buffer of the *EVM* form.
4. Specify the start time for the first valid symbol in the *Symbol Start* field. This can be obtained from the Waveform Viewer window by recording the time of the first minimum or first maximum (whichever is earlier) on the selected signal stream.
5. Specify a period for the symbol in the *Symbol Period* field. Each period is represented by a data rate. The data rate at the output is determined by the particular modulation scheme being used. For example, if the data rate selected is `5.5 Mbps`, it corresponds to a period of `181.8 ns`.
6. The *Auto Level Detect* option button is selected by default and the values in the *Amplitude* and the *Offset* fields are automatically calculated. *Amplitude* is calculated by averaging the rectified voltage level of the signal streams and *Offset* by averaging the sum of an equal number of positive and negative symbols in each signal stream. These values are used to determine the EVM value. If you would like to specify values in the *Amplitude* and the *Offset* fields, you deselect this option button.

Caution

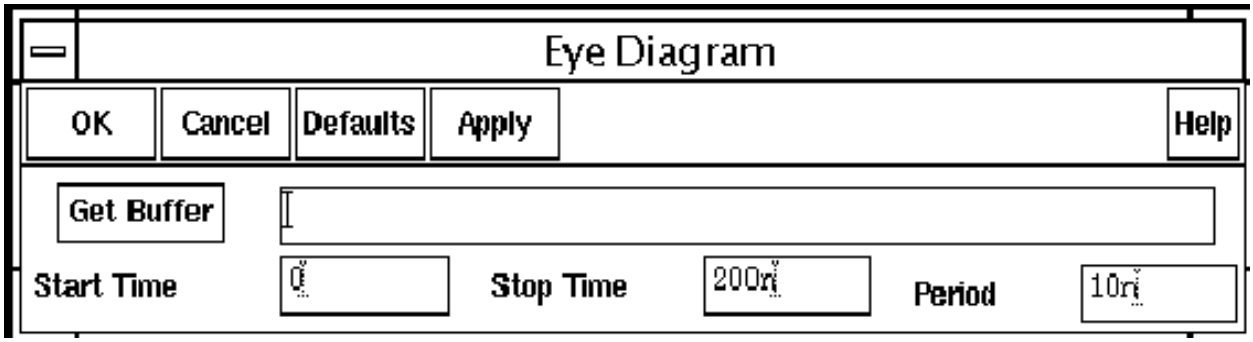
While this feature is convenient, it has to be used with caution as it tends to underestimate the EVM number. The automatic determination of levels means that for mismatching I and Q channels, the EVM would be underestimated because the I and Q channels are compared to their individual average signal level instead of a common ideal level. Also, undesired DC offset, if any, would not be included in the calculated EVM.

7. Select the *Normalize Display* option button when you want to see the scatter plot normalized to the ideal values `+1` and `-1` (for example, when superimposing scatter plots from different stages in the signal flow, where the levels may be quite different but the you want to see relative degradation or improvement in the scatter). This button does not affect the calculation of the EVM number.
8. Click *OK*.

eyeDiagram Function

The *eyeDiagram Function* gives an eye-diagram plot in which the waveform signal is divided into fixed time periods, which are then superimposed on each other. The result is a plot that has many overlapping lines enclosing an empty space known as the "*eye*". The quality of the receiver circuit is characterized by the dimension of the eye. An open eye means tha t the detector will be able to distinguish between `1's` and `0's` in its input, while a closed eye means that a detector placed on `Vout` is likely to give errors for certain input bit sequences.

1. Choose *eyeDiagram* from the *Special Functions* menu in the *Calculator*. The Eye Diagram form appears.



2. Define a waveform in the buffer.
3. Enter the values for Start Time, Stop Time and Period.

4. Click OK.
5. Click on the *plot button* in the *Calculator* to plot the eyeDiagram.

Flip Function

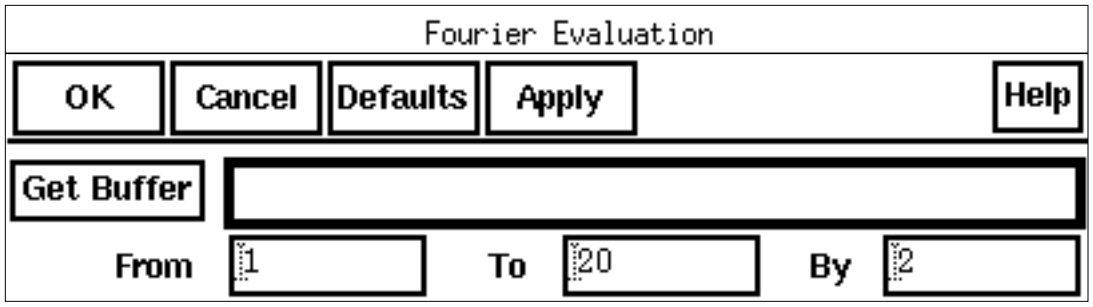
The *flip* function negates the X values. It has the effect of flipping the waveform horizontally in relation to the y axis.

Fourier Evaluation (fourEval) Function

The *fourEval* function evaluates the Fourier series represented by an expression. This function is an inverse Fourier transformation and thus the inverse of the Discrete Fourier Transform (dft) Function. The *fourEval* function transforms the expression from the frequency domain to the time domain.

1. Select *fourEval*.

   The Fourier Evaluation form appears.



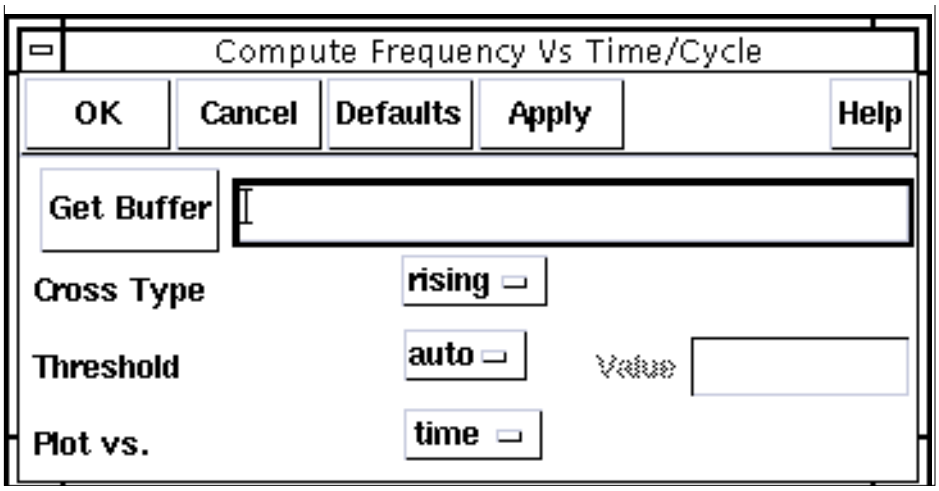
2. Set up the expression in the calculator buffer and click *Get Buffer*.
3. Specify the time range over which you want to evaluate the series.
4. Enter the increment for evaluating the series.

Freq Function

The *freq* function estimates the frequency of the input waveform(s) as a function of time or cycle.

1. Select *freq*.

The Compute Frequency Vs. Time/Cycle form appears.



2. Click *Get Buffer*.
3. Specify *Cross Type* as *rising* or *falling*.
4. Specify *Threshold* as *auto* or *user*. If you select *auto*, thethreshold value is calculated internally. If you select *user*, the *Value* field becomes editable and you need to also specify a value.
5. Specify what you want to plot the frequency against in the *Plot vs.* field. You may select *time* or *cycle*.
6. Click *OK*.
7. Click on the *plot button* in the *Calculator* to plot the frequency.

Frequency Function

The *frequency* function estimates the frequency of a periodic waveform. The system computes the reciprocal of the average time between two successive midpoint crossings of the rising waveform.

1. Select *frequency*.
2. Enter the expression and the closing parenthesis.

Gain (gainBwProd/gainMargin) Functions

The *gainBwProd* function calculates the gain-bandwidth product. This function requires one argument, the frequency response of interest over a sufficiently large frequency range.

$$gainBwProd\ (\ gain\ ) = A_o * f2$$

The gain-bandwidth product is calculated as the product of the DC gain A0 and the critical frequency f2.The critical frequency f2 is the smallest frequency for which the gain equals $1/\sqrt{2}$ times the DC gain A_0.

The *gainMargin* function computes the gain margin of the loop gain of an amplifier. It requires one argument, the loop gain of interest over a sufficiently large frequency range.

```
gainMargin( gain ) = 20 * log10( value( gain f0 ) )
```

The gain margin is calculated as the magnitude of the gain in dB at f0.
The frequency f0 is the smallest frequency in which the phase of the gain provided is -180 degrees. For stability, the gain margin must be less than 0dB.

1. Select *gainBwProd* or *gainMargin*.
2. Enter the expression and the closing parenthesis.

Group Delay Function

The *groupDelay* function computes the group delay of an expression.

Group delay is defined as the derivative of the phase with respect to frequency. Group delay is expressed in seconds. It is calculated using the *vp* function as shown below.

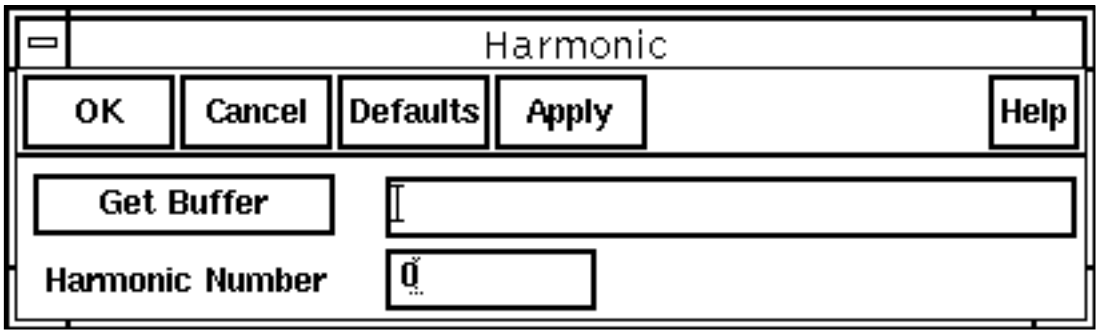
$$Group\ Delay = \frac{d\phi}{d\omega} = \frac{d}{df}\left[\frac{phase(/netX)}{360}\right]$$

1. Select *groupDelay*.
2. Enter the expression and the closing parenthesis.

Harmonic Function

This function returns the harmonic waveform of a waveform you specify.

1. Choose *harmonic* in the *Special Functions* menu of the Calculator to open the Harmonic form.



2. Enter the expression for the waveform in the Calculator buffer.
3. Click *Get Buffer* in the Harmonic form.
4. In the Harmonic form, specify the harmonic you want by typing the value in the *Harmonic Number* field.
5. Click *OK* or *Apply* in the Harmonic form.
6. Click *plot* in the Calculator to plot the waveform.

To use this function, you must type the line below in the CIW

```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

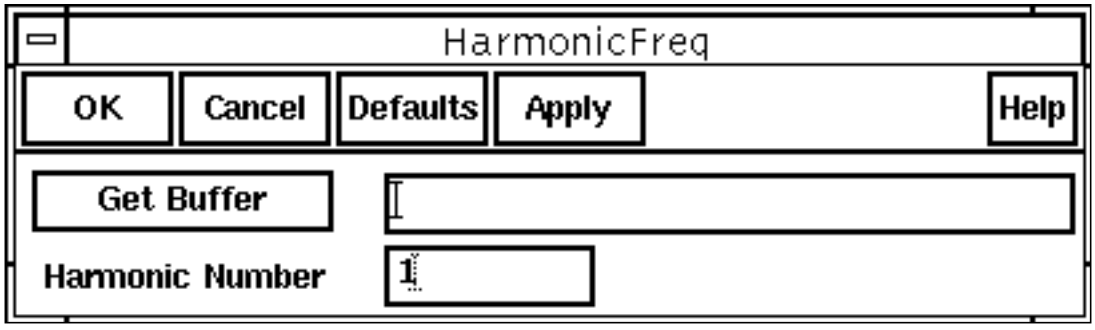
or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

Harmonic Frequency Function

This function returns the harmonic waveform of a waveform you specify.

1. Enter the expression for the waveform in the Calculator buffer.

2. Choose *harmonicFreq* in the *Special Functions* menu of the Calculator to open the HarmonicFreq form.



3. Click *Get Buffer* in the HarmonicFreq form.
4. In the HarmonicFreq form, specify the harmonic you want by typing the value in the *Harmonic Number* field.
5. Click *OK* or *Apply*.

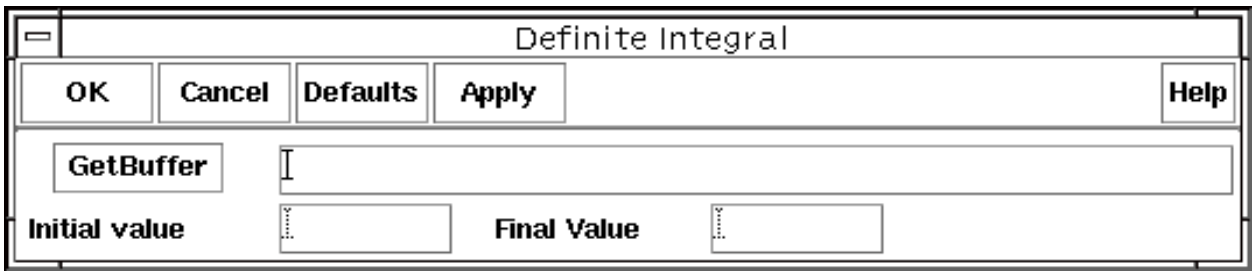
iinteg Function

The *iinteg* function computes the indefinite integral of an expression with respect to the X-axis variable. The result is a waveform that can be plotted.

integ Function

The *integ* function computes the definite integral of the expression in the buffer. The result is the value of the area under the curve over a specified range on the X-axis of the expression.

1. Choose *integ* from the *Special Functions* menu in Calculator. The Definite Integral form appears.



2. Define a waveform in the buffer.

3. Enter the values for the limits of the definite integral in the *Initial Value* and *Final Value* fields.

   Note: You should specify either both the limits or neither. In case you do specify the limits, they become the end points of the range on the X-axis for definite integration. If you do not specify the limits, then the range for definite integration is the entire range of the sweep on the X-axis.

4. Click *OK*.

intersect Function

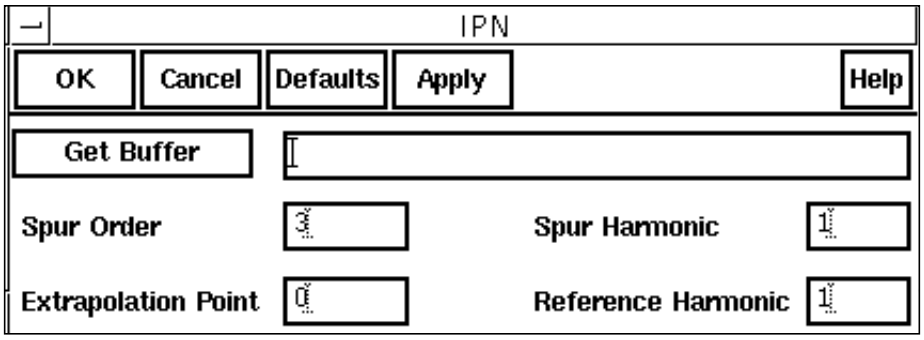
This function returns a waveform containing the points of intersection for the two waveforms passed as input.

1. Choose *intersect* from the *Special Functions* menu of the Calculator.
2. Define the expression for the first waveform.
3. Click the *space* button on the Calculator.
4. Define the expression for the second waveform.
5. Click the right parenthesis button ") " on the Calculator to close the expression.

ipn Function

This function plots the *Nth* order intercept between two harmonics of a waveform that you define.

1. Choose *ipn* from the *Special Functions* menu in the Calculator. The IPN form opens.



2. Enter the expression for the waveform in the Calculator buffer.

3. Click *Get Buffer* in the IPN form.

4. Type values for the following four quantities:

Spur Order	Spur Order determines what order of interference is calculated for the spurious and reference waves. The default value is 3; this corresponds to the *IP3* function. I you use a value other than 3, that order of interference is calculated between those two waves.
Spur Harmonic	Harmonic number for spurious waveform.
Reference Harmonic	Harmonic number for reference waveform.
Extrapolation Point	The extrapolation point for the *ipn* function. This is the X axis value.

5. Click *OK* or *Apply*.

   The expression is sent to the calculator buffer. To evaluate the expression, click the equal sign (=) button in the calculator.

To use this function, you must type the line below in the CIW

```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

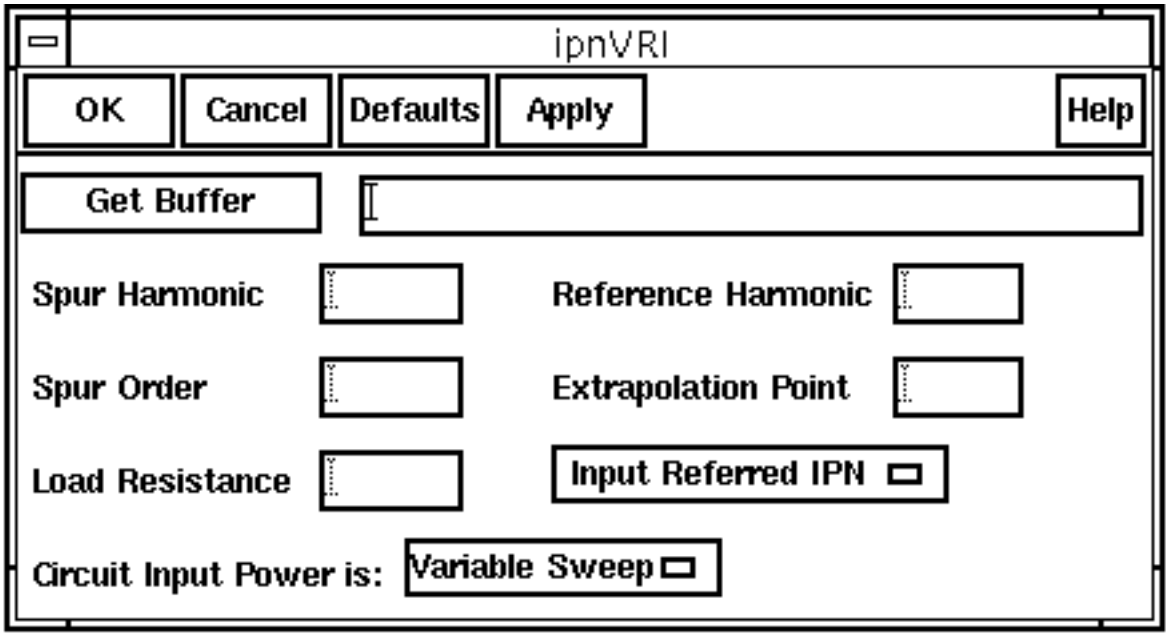
ipnVRI Function

This function performs an intermodulation Nth-order intercept point measurement.

Use this function to simplify the declaration of an ipn measurement. This function extracts the spurious and reference harmonics from the input waveform(s), and uses `dBm(spectralPower((i or v/r),v))` to calculate the respective powers. The function then passes these power curves or numbers and the remaining arguments to the *ipn* function to complete the measurement.

From each of the spurious and reference power waveforms (or points), the *ipn* function extrapolates a line of constant slope (dB/dB) according to the specified order and input power level. These lines represent constant small-signal power gain (ideal gain). The *ipn* function calculates the intersection of these two lines and returns the value of either the x coordinate (input referred) or y coordinate.

1. Choose *ipnVRI* in the *Special Functions* menu of the Calculator. The ipnVRI form opens.



2. Enter the expression for the waveform in the Calculator buffer.
3. Click *Get Buffer* in the ipnVRI form.
4. Type values for the following quantities:

Spur Harmonic	Harmonic index for spurious waveform.
Reference Harmonic	Harmonic index for reference waveform.
Spur Order	Spur Order determines what order of interference is calculated for the spurious and reference waves. The default value is 3; this corresponds to the *IP3* function. If you use a value other than 3, that order of interference is calculated between those two waves.
Extrapolation Point	The extrapolation point for the *ipn* function. This is the X axis value. The default is the minimum x value of the input voltage waveform.
Load Resistance	The resistance into the output port. The default value is 50.

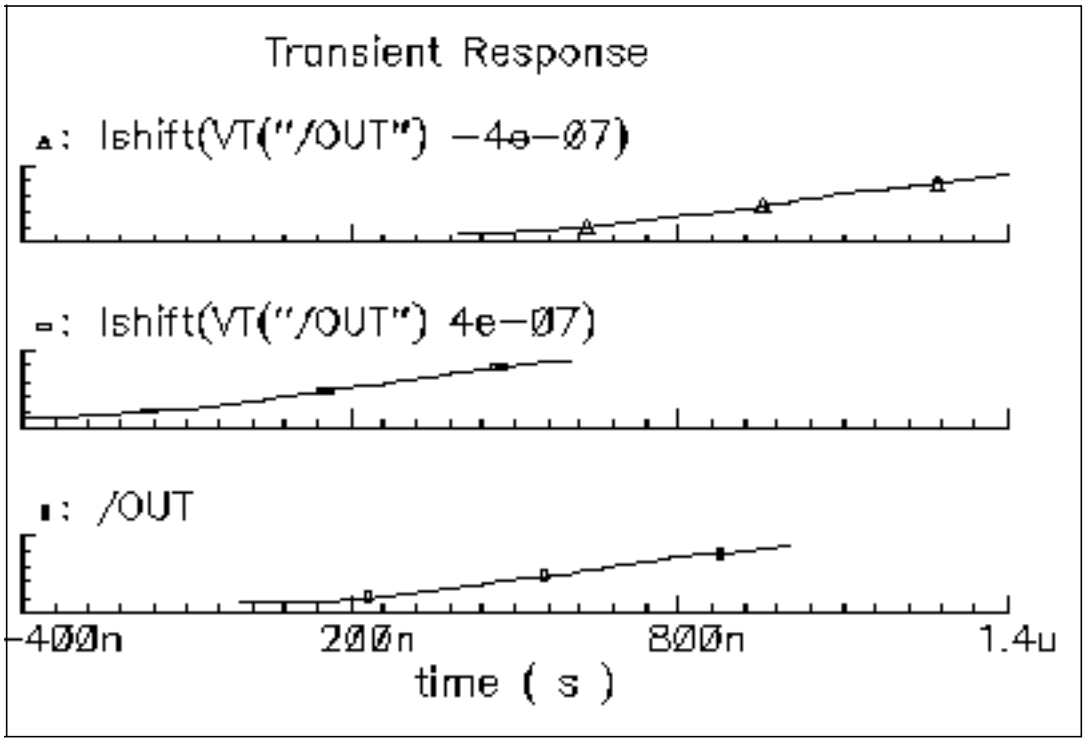
5. To get the X-coordinate of the intercept, specify *Input Referred IPN*. To get the Y-coordinate of the intercept, specify *Output Referred IPN*.

6. Indicate whether the *Circuit Input Power* is a *Variable Sweep* or a *Single Point*.
7. Click *OK* or *Apply*.

Note: For an extended design example, using ipn and ipnVRI functions, see Appendix A, "Using the Calculator Special Functions with SpectreRF Simulation Results".

Lshift Function

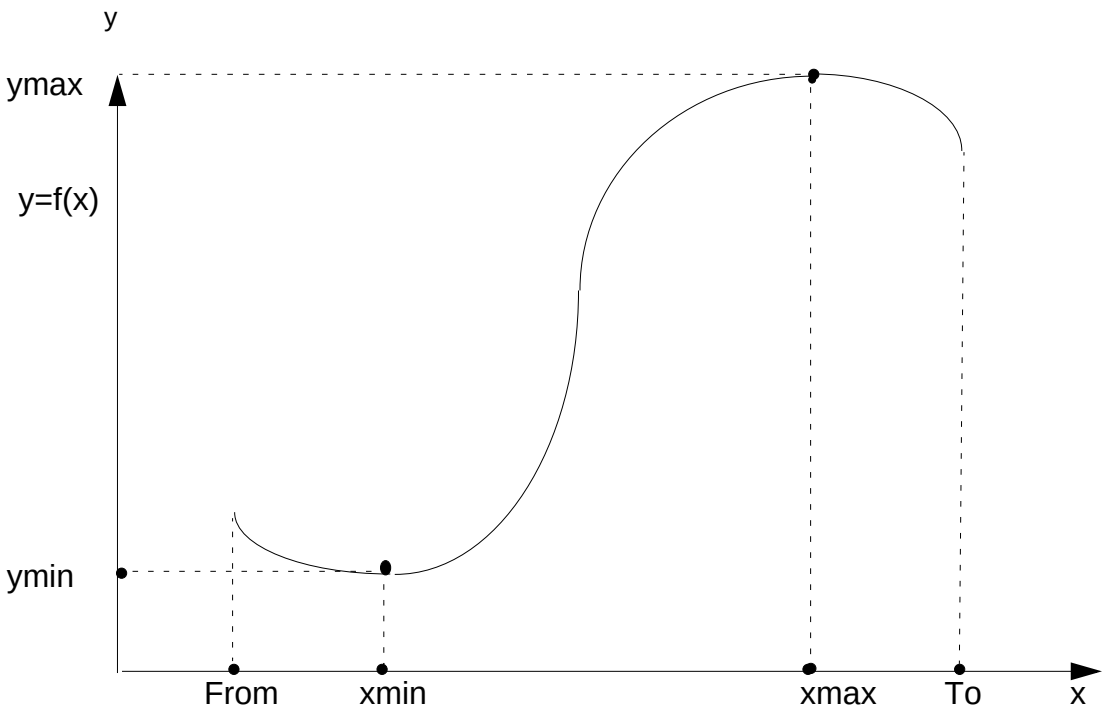
The *lshift* function shifts the data in the Waveform window to the left by a specified amount. A negative value shifts the data to the right.



Minimum and Maximum Functions

You can calculate minimum and maximum values of waveforms with the *xmax*, *xmin*, *ymax*, and *ymin* functions.

This figure shows the relationship of these functions.

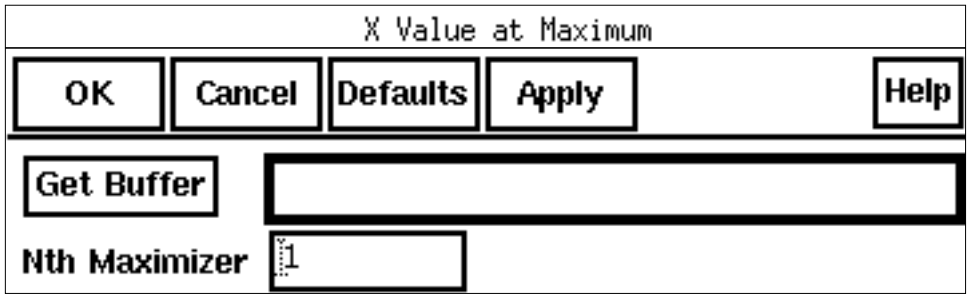


xmax and ymax

The *xmax* function computes the value of the independent variable x at which the expression attains its maximum value, that is, the value of x that maximizes y=f(x).

1. Select *xmax*.

   The X Value at Maximum form appears.



2. Enter the expression in the calculator buffer and click *Get Buffer*.

The maximum might occur at more than one point on the x axis. You must choose (in the *Nth Maximizer* field) which maximum value you want to see. The calculator returns the value of the Nth Maximizer counting from the left, that is, toward increasing X-axis values. If you enter a negative integer, the direction of search is reversed toward decreasing X-axis values (counting from the right).

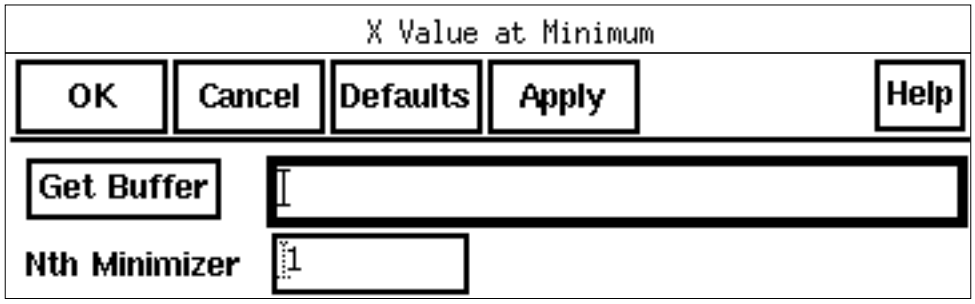
The *ymax* function computes the maximum y of the expression y=f(x).

xmin and ymin

The *xmin* function computes the value of the independent variable x at which the expression has its minimum value, that is, the value of x that minimizes y=f(x).

1. Select *xmin*.

   The X Value at Minimum form appears.



2. Enter the expression in the calculator buffer and click *Get Buffer*.

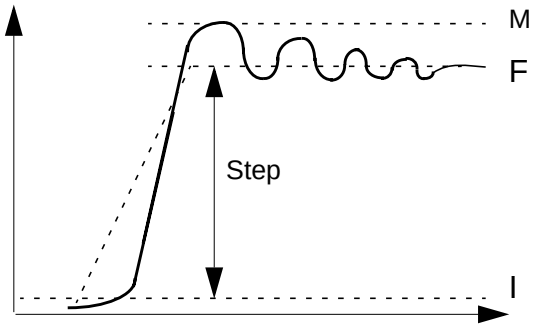
The minimum might occur at more than one point on the x axis. You must choose (in the *Nth Minimizer* field) which minimum value you want to see. The calculator returns the value of the Nth Minimizer, counting from the left, that is, toward increasing X-axis values. If you enter a negative integer, the direction of search is reversed toward decreasing X-axis values (counting from the right).

The *ymin* function computes the minimum y of the expression y=f(x).

Overshoot Function

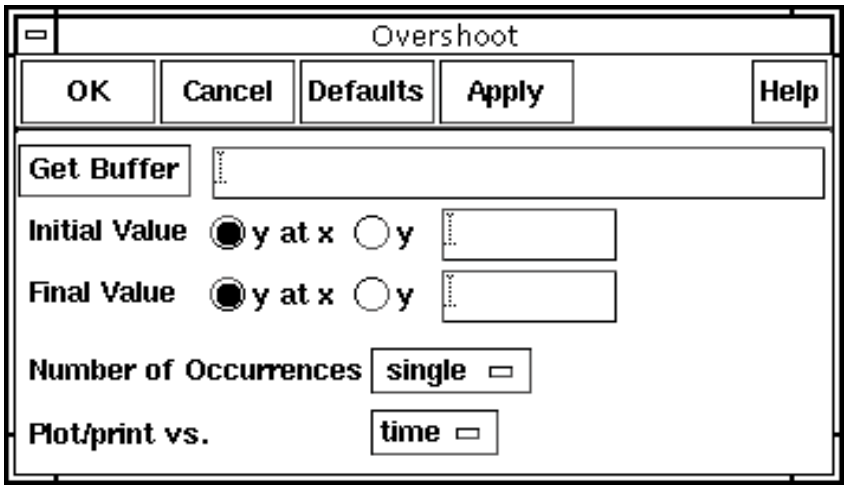
The *overshoot* function computes the percentage by which an expression overshoots a step going from the *Initial Value* to the *Final Value* you enter.

To retrieve one or multiple occurrences of overshoot in a waveform:



$$Overshoot = \frac{(M-F)x100}{F-I}$$

1. Select *overshoot* from the Special Functions menu.



2. Enter the expression in the calculator buffer and click *Get Buffer*.
3. To set the *Initial* and *Final Values*, do the following:
   - Use the *y* option and enter the y values directly (for example, 3V) or a horizontal marker name (for example, M2).
   - Use the *y at x* option to enter an x value or vertical marker name (for example, M1).
4. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of an overshoot event for the waveform. If you select *multiple*, you can retrieve all occurrences of overshoot for the waveform, which you can later plot or print.
5. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve overshoot data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   overshoot(VT("/out") 0.5 nil 4.95 nil 5 t "time")
--------   --------
7.862n     12.05
4.008u     12.05
8.008u     12.05
```

Here, time values refer to each timepoint on the waveform where the overshoot event occurs.

The *cycle* option helps you retrieve overshoot data against cycle numbers. For example:

```
cycle      overshoot(VT("/out") 0.5 nil 4.95 nil 5 t "cycle")
--------   --------
1          12.05
2          12.05
3          12.05
```

Here, cycle numbers refer to the n'th occurence of the overshoot event in the input waveform.

Note: The value for this field is ignored when *Number of Occurrences* is specified as *single*.

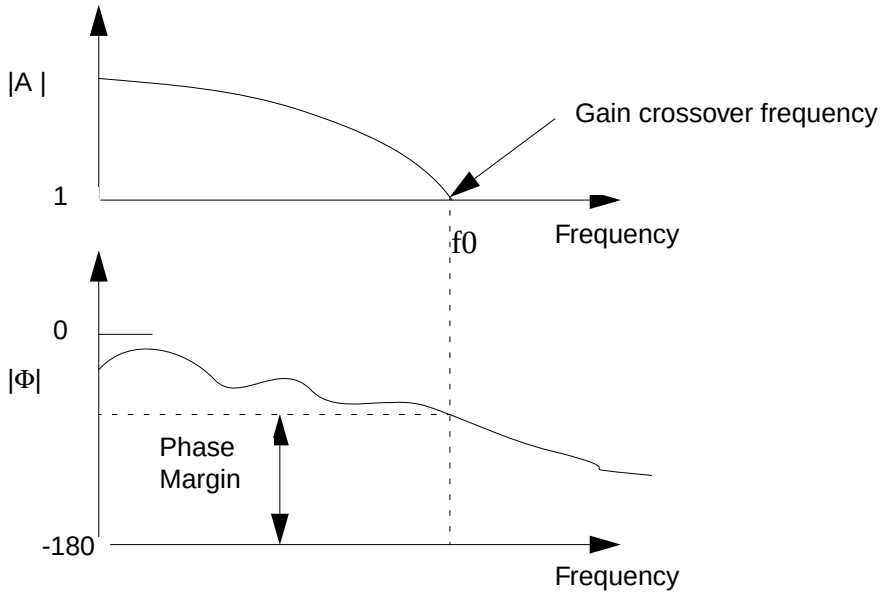
6. Click *OK*.

The system calculates the corresponding y value for the current waveform.

Phase Margin Function

The *phaseMargin* function computes the phase margin of the loop gain of an amplifier. This function is similar to the phaseDegUnwrapped Waveform Calculator SKILL command. It requires one argument, the loop gain of interest over a sufficiently large frequency range.

```
phaseMargin( gain ) = 180 + phase( value( gain f0 ) )
```



The phase margin is calculated as the difference between the phase of the gain in degrees at f0 and at -180 degrees. The frequency f0 is the smallest frequency where the gain is 1. For stability, the phase margin must be positive.

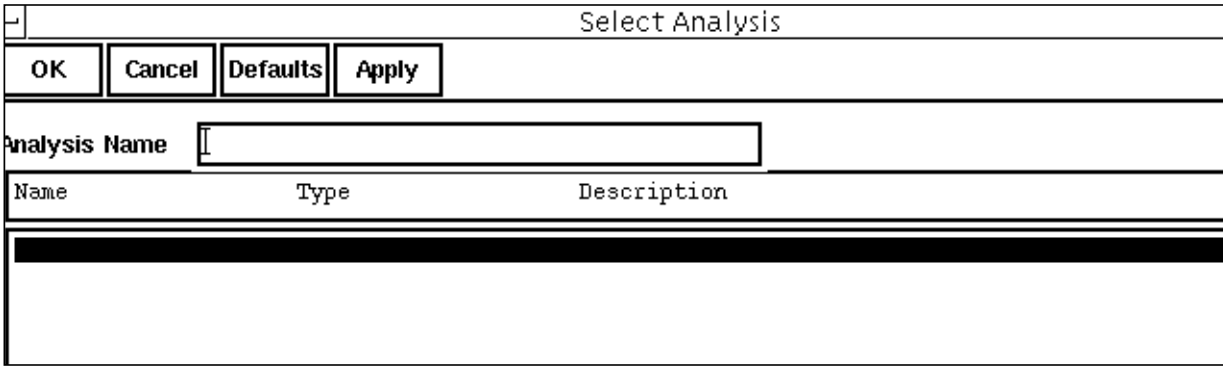
1. Select *phaseMargin*.
2. Enter the expression and the closing parenthesis.

Phase Noise Function

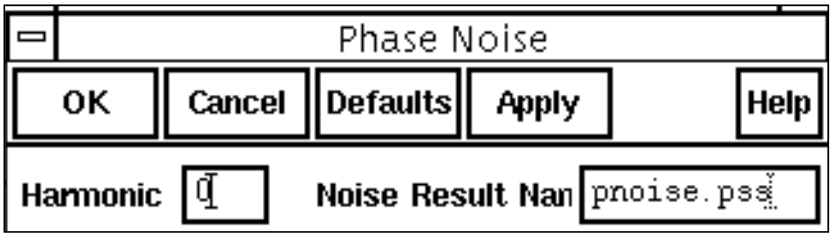
This function plots the phase noise waveform for noise analysis results. You need to follow the following steps to use this function.

1. Set up your PSS analysis.
2. Check the *oscillator button* in the *choosing analysis* form.
3. Set up a Pnoise analysis. Note the value you are entering for the relative harmonic in the *choosing analysis* form.
4. Run your simulation.

5. Invoke the Calculator . Choose *phaseNoise* in the *Special Functions* menu of the Calculator. The *Select Analysis* form appears.



6. Choose the pnoise-pnoise analysis name in the *Analysis Name* field of the *Select Analysis* form, or select the analysis name from the list box in the form. Click on the *OK* button in the form. The *Phase Noise* form appears.



7. In the Phase Noise form, enter *harmonic number=1* or the value you had entered for relative harmonic in the *pnoise choosing analysis* form.
8. Enter the *Noise Result Name* as pnoise-pnoise and click on *OK* in the form.
9. In your calculator buffer, you should see something like phaseNoise(1, "pss-fd.pss", “result "pnoise-pnoise").
10. Click on the *plot* button in the Calculator to plot the waveform.
11. The phase noise plot will appear.

To use this function, you must type the line below in the CIW

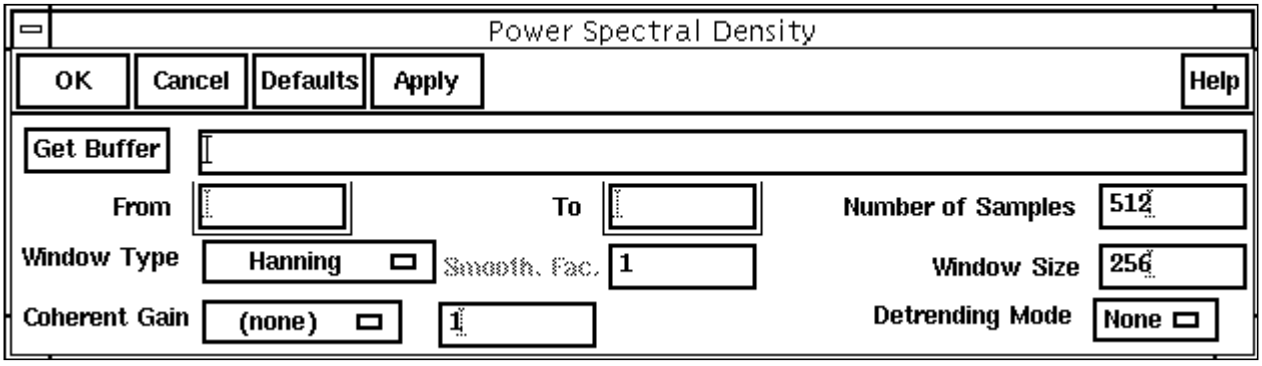
```
envSetVal("calculator" "oldexpr" ‘boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

Power Spectral Density (psd) Function

The power spectral density (*psd*) function describes how the power (or variance) of a time series (signal) is distributed with frequency. Mathematically, it is defined as the Fourier Transform of the auto correlation sequence of the time series (signal). The waveform is first interpolated, to generate evenly spaced data points in time. The spacing of the data points is the inverse of the *dft* sampling frequency. The *psd* is computed by first breaking up the time interval into overlapping segments. Each segment is multiplied, time point by time point, by the specified windowing function. The *dft* is performed on each windowed segment of the baseband waveform. At each frequency, the *dfts* from all segments are averaged together and the squared modulus of these averages, gives *psd*.

After you choose *psd* in the *Special Functions* menu of the Calculator, the Power Spectral Density form opens.



1. Enter the expression for the waveform in the Calculator buffer, and then click *Get Buffer*.
2. In the *From* field, type the starting time for the spectral analysis interval.
3. In the *To* field, type the ending time for the spectral analysis interval. You can use this parameter and the *From* parameter to exclude part of the interval. For example, you might set these values to discard initial transient data.
4. In the *Number of Samples* field, type the number of time domain points to use. The maximum frequency in the Fourier analysis is proportional to the *Number of Samples* parameter and inversely proportional to the difference between the starting time and the ending time.
5. Choose the *Window Type* that you want to use. If you select the Kaiser window type, then type in a value for the Kaiser smoothing factor. The smoothing factor must be in the range 0 <= factor <= 15, where 0 is the same as using a rectangular window.
6. In the *Window Size* field, type in the number of frequency domain points to use in the Fourier analysis.

A larger window size results in an expectation operation over fewer samples, which leads to larger variations in the power spectral density. A small window size can smear out sharp steps in the power spectral density that might really be present.

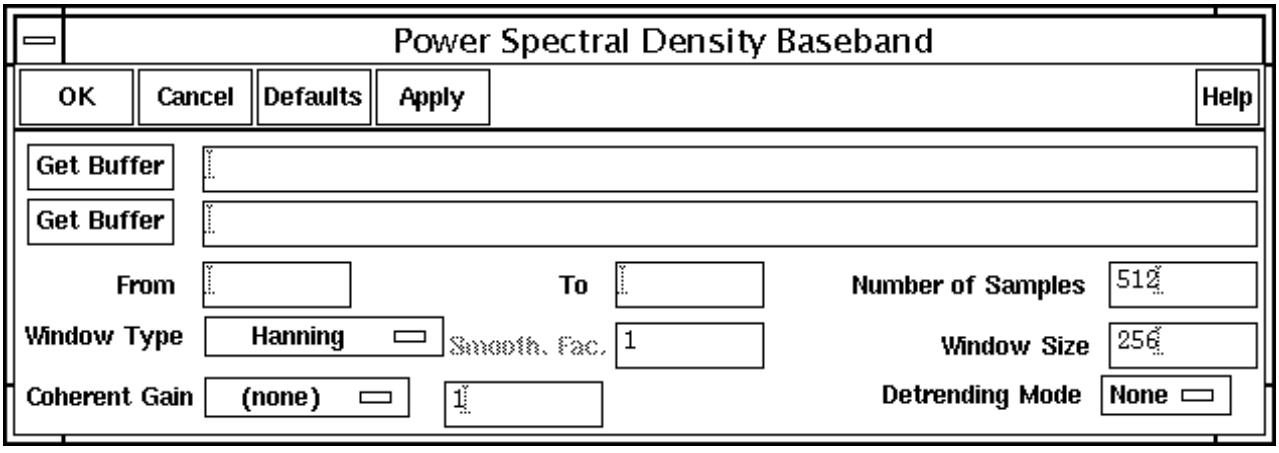
7. Choose a *Coherent Gain* factor. If you choose *magnitude*, *dB20*, or *dB10*, then enter a scaling factor. A non-zero factor scales the power spectral density by 1/(factor). Valid values for the factor are 0 < factor < 1. You can also use a value of 1 if you do not want the *Coherent Gain* factor to be used.
8. Choose which *Detrending Mode* to use.

   The *psd* function works by applying a moving windowed FFT to time-series data. If there is a deterministic trend to the underlying data, you might want to remove the trend before performing the spectral analysis. For example, consider analyzing phase noise in a VCO model. Without the noise the phase increases more or less linearly with time, so it is appropriate to set the detrending mode to *linear*. To subtract an average value, set the detrending mode to *mean*. Where the spectrum of raw data is desired, set the detrending mode to *none*.
9. Click *OK*.

Power Spectral Density Baseband (psdbb) Function

The power spectral density baseband (*psdbb*) function returns an estimate for the power spectral density of a waveform1+j * waveform2.

After you choose *psdbb* in the *Special Functions* menu of the Calculator, the Power Spectral Density Baseband form opens.



1. Enter the expression for the waveform in the Calculator buffer, and then click *Get Buffer*.

2. In the *From* field, type the starting time for the spectral analysis interval.
3. In the *To* field, type the ending time for the spectral analysis interval. You can use this parameter and the *From* parameter to exclude part of the interval. For example, you might set these values to discard initial transient data.
4. In the *Number of Samples* field, type the number of time domain points to use. The maximum frequency in the Fourier analysis is proportional to the *Number of Samples* parameter and inversely proportional to the difference between the starting time and the ending time.
5. Choose the *Window Type* that you want to use. If you select the *Kaiser* window type, then type in a value for the Kaiser smoothing factor. The smoothing factor must be in the range 0 <= factor <= 15, where 0 is the same as using a rectangular window.
6. In the *Window Size* field, type in the number of frequency domain points to use in the Fourier analysis.

   A larger window size results in an expectation operation over fewer samples, which leads to larger variations in the power spectral density. A small window size can smear out sharp steps in the power spectral density that might really be present.
7. Choose a *Coherent Gain* factor. If you choose *magnitude*, *dB20*, or *dB10*, then enter a scaling factor. A non-zero factor scales the power spectral density by 1/(factor). Valid values for the factor are 0 < factor < 1. You can also use a value of 1 if you do not want the *Coherent Gain* factor to be used.
8. Choose which *Detrending Mode* to use.

   The *psdbb* function works by applying a moving windowed FFT to time-series data. If there is a deterministic trend to the underlying data, you might want to remove the trend before performing the spectral analysis. For example, consider analyzing phase noise in a VCO model. Without the noise the phase increases more or less linearly with time, so it is appropriate to set the detrending mode to *linear*. To subtract an average value, set the detrending mode to *mean*. Where the spectrum of raw data is desired, set the detrending mode to *none*.
9. Click *OK*.

Rise Time Function

The *riseTime* function computes the riseTime of the buffer expression, that is, the time required for the waveform to rise from `theta1` (*Percent Low*) to `theta2` (*Percent High*) of the difference between the specified *Initial Value* and the specified *Final Value*.

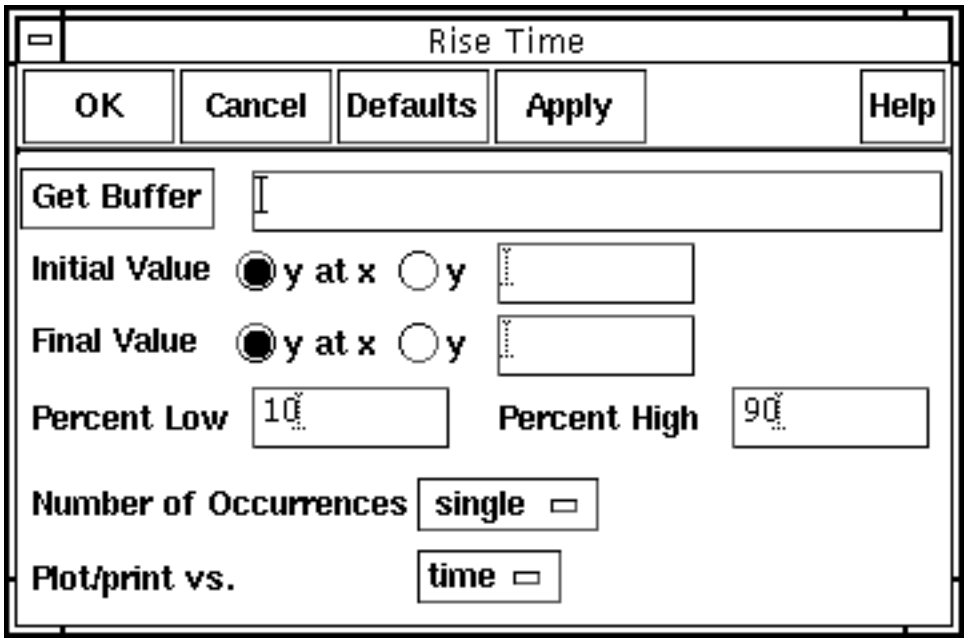
You can calculate the start and end of a riseTime event as follows:

```
Start = Initial Value + Low/100 (Final Value - Initial Value)
End = Initial Value + High/100 (Final Value - Initial Value)
```

To retrieve one or multiple occurrences of riseTime in a waveform:

1. Select *riseTime*.

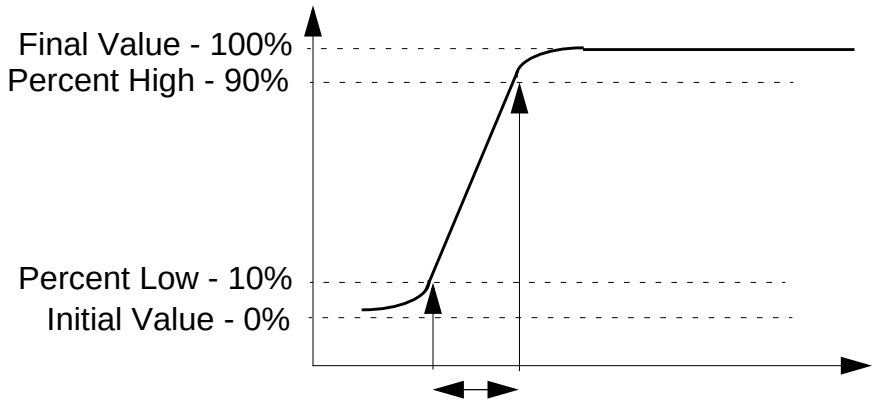
   The Rise Time form appears.



2. Set up an expression in the calculator buffer and click *Get Buffer*.
3. Set the initial and final values in the Rise Time form as follows:

- Use the *y* option for *Initial Value* and *Final Value* and enter the y values directly or a horizontal marker name.

- Use the *y at x* option for *Initial Value* and *Final Value* to enter an x value or vertical marker name.



Value returned by
riseTime function

For waveforms with multiple rise and fall edges, you should isolate edges of interest by using the `clip` function or enter values for y at x rather than entering y values directly in the Rise Time form.

4. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of a riseTime event for a given waveform. If you select *multiple*, you can retrieve all occurrences of riseTime for a given waveform, which you can later plot or print.

5. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve riseTime data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)    riseTime(VT("/out") 0.5 nil 4.5 nil 10 90 t "time") (s)
--------    --------
12.22n      32.97n
4.012u      32.96n
8.012u      32.96n
```

Here, time values refer to each timepoint on the waveform where the riseTime event occurs.

The *cycle* option helps you retrieve riseTime data against cycle numbers. For example:

```
cycle       riseTime(VT("/out") 0.5 nil 4.5 nil 10 90 t "cycle") (s)
--------    --------
1           32.97n
2           32.96n
```

```
3          32.96n
```

Here, cycle numbers refer to the n'th occurence of the riseTime event in the input waveform.

Note: The value for *Plot/print vs.* is ignored when *Number of Occurrences* is specified as *single*.

6. Click *OK*.

Root-Mean-Square (rms) Function

The *rms* function computes the root-mean-square value of the expression f(x), over the specified range of x. This is the square root of the integral of the expression squared over the specified range, divided by the range.

For example, if y=f(x),

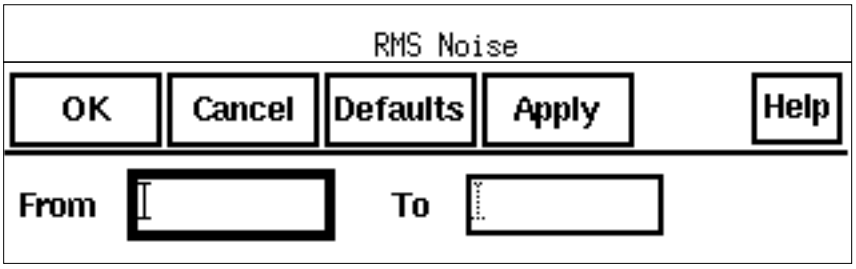
$$rms(y) = \sqrt{\frac{\int_{from}^{to} f(x)^2 dx}{to - from}}$$

1. Select *rms*.
2. Enter the expression and closing parenthesis.

To compute the rms value of the expression over a smaller range, use the clip function inside the *rms* function.

Root-Mean-Square (rms) Noise Function

The *rmsNoise* function computes the integrated root-mean-square noise over the bandwidth specified in hertz in the *From* and *To* fields.



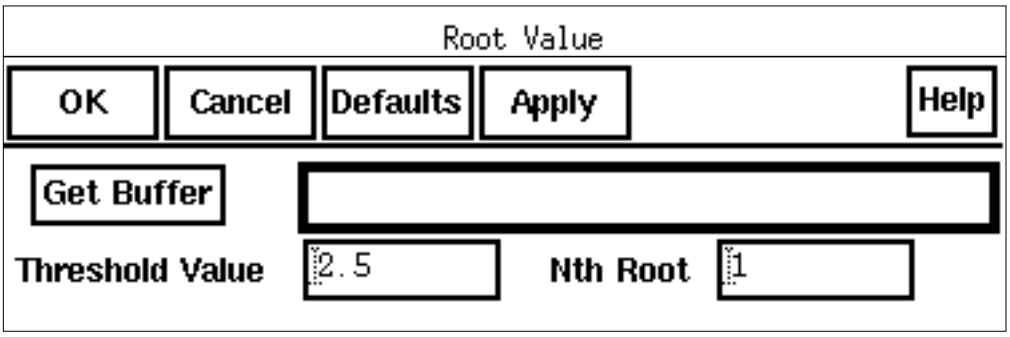
Note: To instead plot the squared noise voltage versus frequency, use the Results – Plot Noise command in the simulation window.

Root Function

The *root* function computes the value of x at which f(x) equals the specified threshold.

1. Select *root*.

   The Root Value form appears.



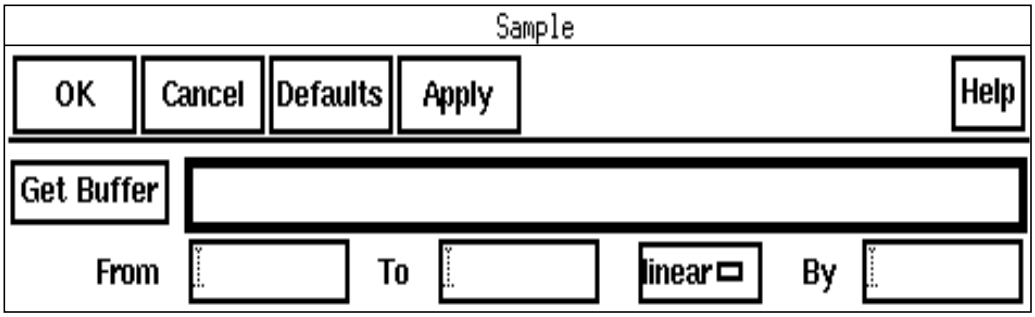
2. Set up an expression in the calculator buffer and click *Get Buffer*.
3. Enter the waveform value at which to compute the root value.
4. Enter the *root* you want to see.

Sample Function

The *sample* function samples a waveform at the interval you specify. You can use this function to speed up plotting of waveforms that have many data points.

1. Select *sample*.

   The Sample form appears.



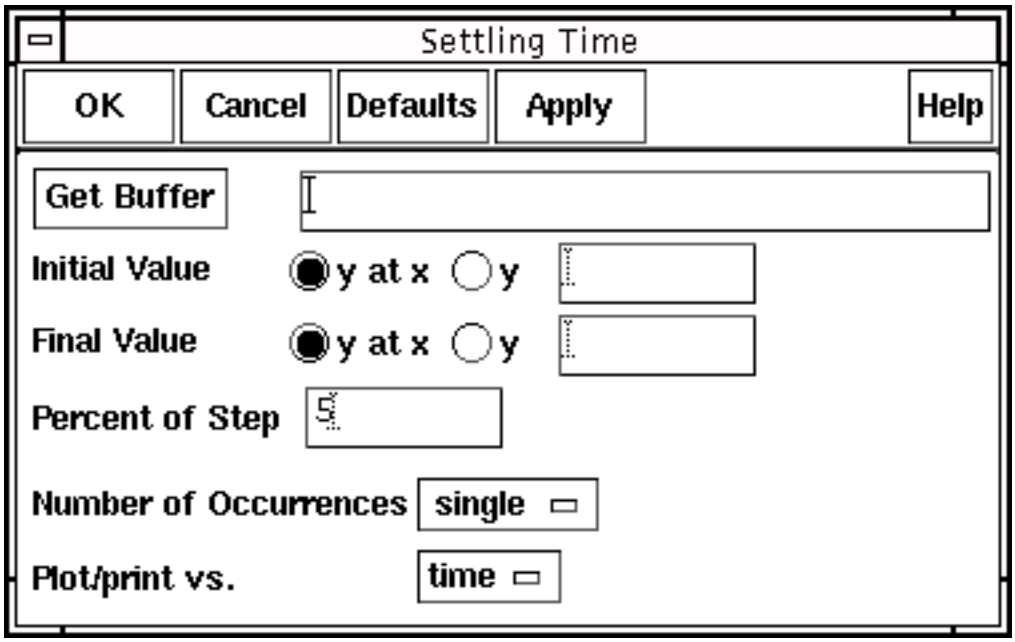
2. Define a waveform in the calculator buffer.
3. Specify the range and increment.

If you sample a waveform beyond its range, you get the final value of the waveform. You can use this function to demodulate a signal. Consider an AM modulated sine wave. Assume the carrier frequency is 1 GHz, and the modulation frequency is 1 MHz. If the waveform is sampled every 1 ns, the resulting signal will be cleanly demodulated (the 1 GHz carrier is completely eliminated by the sampling).

Settling Time Function

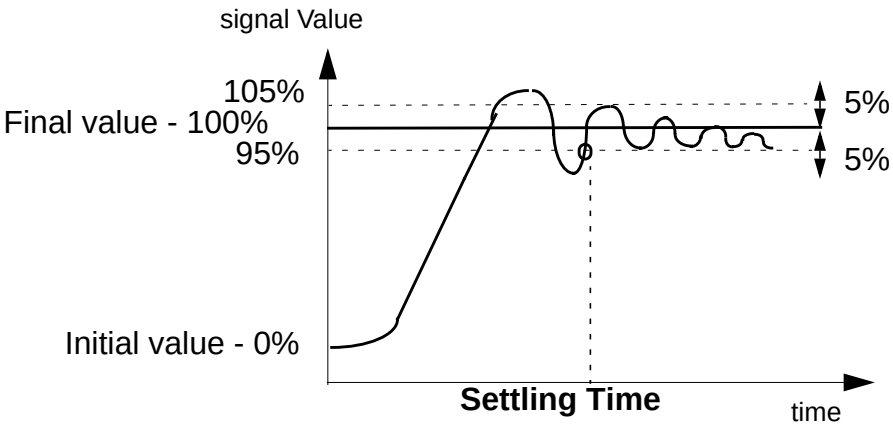
The *SettlingTime* function calculates one or multiple occurrences of the time by which the signal settles within the specified percent of step of the difference beween the *Final Value* and *Initial Value* from the *Final Value*.

1. Select *settlingTime*. The Settling Time form appears.



2. Set up an expression in the calculator buffer and click *Get Buffer*.
3. Set the *Initial* and *Final Value* in the Settling Time form as follows:

- Use the *y* option for *Initial Value* and *Final Value* and enter the y values directly or a horizontal marker name.
- Use the *y at x* option for *Initial Value* and *Final Value* to enter an x value or vertical marker name.



4. Enter the value for *Percent of Step*.
5. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of a settlingTime event for a given waveform. If you select *multiple*, you can retrieve all occurrences of settlingTime for a given waveform, which you can later plot or print.
6. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve settlingTime data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   settlingTime(VT("/out") 0.5 nil 4.95 nil 5 t "time") (s)
--------   --------
7.862n     53.36n
4.008u     4.053u
8.008u     8.053u
```

Here, time values refer to each timepoint on the waveform where the settlingTime event occurs.

The *cycle* option helps you retrieve settlingTime data against cycle numbers. For example:

```
cycle      settlingTime(VT("/out") 0.5 nil 4.95 nil 5 t "cycle") (s)
-----      --------
1          53.36n
2          4.053u
3          8.053u
```

Here, cycle numbers refer to the n'th occurence of the settlingTime event in the input waveform.

Note: The value for *Plot/print vs.* is ignored when *Number of Occurrences* is specified as *single*.

7. Click *OK*.

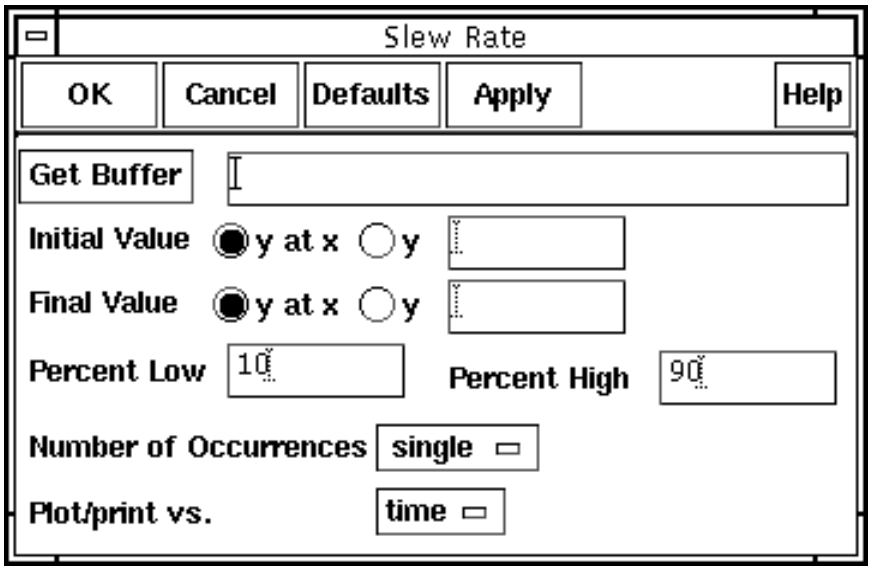
Slew Rate Function

The *slewRate* function computes the average rate at which the buffer expression changes from percent low to percent high of the difference between *Initial Value* and *Final Value*.

To retrieve one or multiple occurrences of slewRate in a waveform:

1. Select *slewRate*.

   The Slew Rate form appears.



2. Set up an expression in the calculator buffer and click *Get Buffer*.
3. Set *Initial Value* and *Final Value* using either of the following ways:

- Use the *y* option for *Initial Value* and *Final Value* and enter the y values directly or a horizontal marker name.
- Use the *y at x* option for *Initial Value* and *Final Value* to enter an x value or vertical marker name.

For waveforms with multiple rise and fall edges, you should isolate edges of interest by using the `clip` function or enter values for y at x rather than entering y values directly in the Slew Rate form.

4. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of a slewRate event for a given waveform. If you select *multiple*, you can retrieve all occurrences of slewRate for a given waveform, which you can later plot or print.

5. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve slewRate data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   slewRate(VT("/out") 0.5 nil 4.5 nil 10 90 t "time")
--------   --------
12.22n     97.07M
4.012u     97.09M
8.012u     97.09M
```

   Here, time values refer to each timepoint on the waveform where the slewRate event occurs.

   The *cycle* option helps you retrieve slewRate data against cycle numbers. For example:

```
cycle      slewRate(VT("/out") 0.5 nil 4.5 nil 10 90 t "cycle")
-----      --------
1          97.07M
2          97.09M
3          97.09M
```

   Here, cycle numbers refer to the n’th occurence of slewRate computation for the input waveform.

   Note: The value for *Plot/print vs.* is ignored when *Number of Occurrences* is specified as *single*.

6. Click *OK*.

Spectral Power Function

This function plots the spectral power for a current waveform and a voltage waveform that you define. To use this function:

1. Choose *spectralPower* in the *Special Functions* menu of the Calculator.

2. Define the voltage waveform.

3. Click the *space* button on the Calculator.
4. Define the current waveform.
5. Click the right parenthesis button " *)* " on the Calculator to close the expression.
6. Click *plot* in the Calculator to plot the waveform.

To use this function, you must type the line below in the CIW

```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

Standard Deviation (stddev) Function

The *stddev* function computes the standard deviation of a waveform (or a family of waveforms) over its entire range. Standard deviation (stddev) is defined as the square-root of the variance where variance is the integral of the square of the difference of the expression f(x) from average (f(x)), divided by the range of x.

For example, if `y=f(x)`

$$stddev(y) = \sqrt{\frac{\int_{from}^{to} (y - average(y))^2}{to - from}}$$

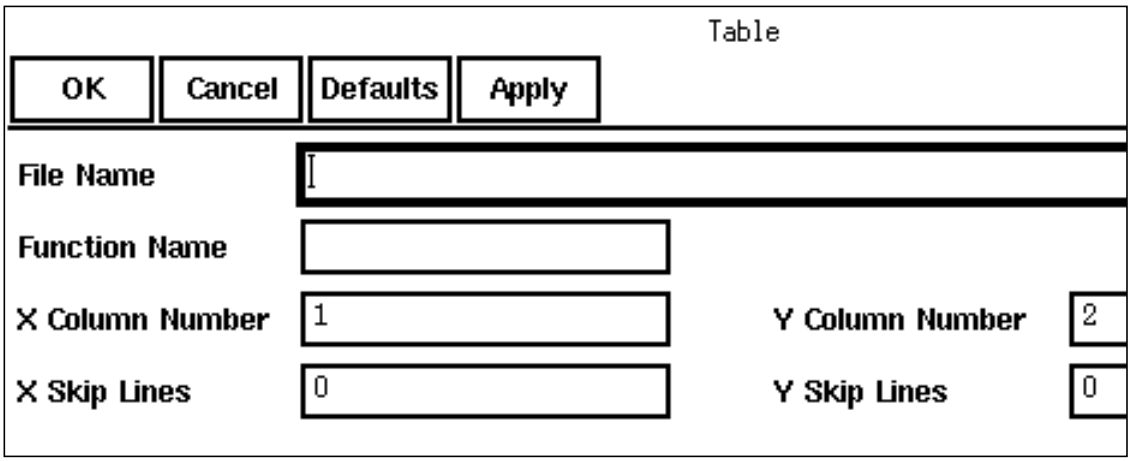
If you want a different range, use the clip function to clip the waveform to the range you want.

Table Function

The *table* function defines a piecewise linear function from a column of x and y values in a file. This function was previously named implicitX.

1. Select *table*.

The Table form appears.



2. Enter the name of the data file and any name for the function.
3. (Optional) Enter the column numbers containing the X- and Y-axis data, if they are not in columns 1 and 2 respectively.
4. (Optional) Enter the number of lines to skip in each column from the top of the file before reading the data.

Note: Do not count comment lines beginning with a semicolon and blank lines in the number of lines to skip.

The X data must be real numbers increasing monotonically. The Y data can be real numbers, or complex numbers following this syntax:

```
(real_part imag_part)
```

or

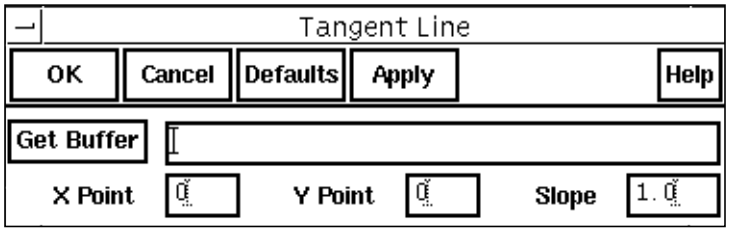
```
complex(real_part imag_part)
```

Tangent Function

This function plots a line that passes through x and y coordinates and the slope that you specify. To use this function:

1. Choose *tangent* from the *Special Functions* menu in the Calculator.

The Tangent Line form appears.



2. Define a waveform in the buffer.
3. Click *Get Buffer* in the Tangent Line form.
4. Type values for the *X point*, *Y Point*, and *Slope* fields.
5. Click *OK*.
6. Click *plot* in the Calculator to plot the tangent line.

To use this function, you must type the line below in the CIW

```
envSetVal("calculator" "oldexpr" 'boolean nil)
```

or set the calculator `oldexpr` variable to `nil` in your `.cdsenv` file.

Total Harmonic Distortion (thd) Function

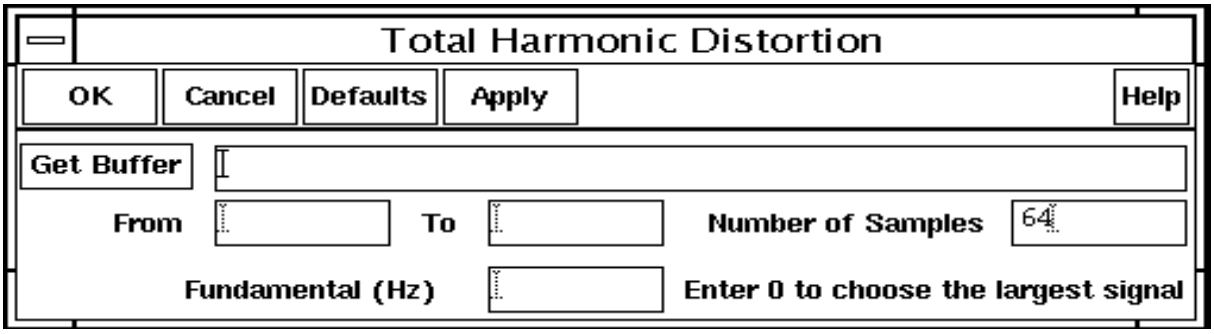
The *thd* function computes the percentage of total harmonic content of a signal with respect to the fundamental frequency. The computation uses the dft function (for information, see Discrete Fourier Transform (dft) Function). Assume that the *dft* function returns complex coefficients A_0, A_1…, A_f , ... Please note that fundamental frequency ***f*** is the frequency contributing to the largest power in the signal. A_0 is the complex coefficient for the DC component and A_i is the complex coefficient for the ***i***th harmonic where $i \neq 0, f$. Then, total harmonic distortion is computed as:

$$\frac{\sqrt{\sum_{i=1,\, i\neq 0,\, f} |A_i|^2}}{|A_f|} \times 100\%$$

To compute the *thd*, you need to perform the following steps:

1. Choose *thd* in the *Special Functions* menu of the calculator.

The Total Harmonic Distortion form appears.



2. Set up an expression in the calculator buffer and click *Get Buffer*.
3. Specify the range and the number of samples.
4. Click *OK*.
5. Click Print to see the result.

The accuracy of the total harmonic distortion measurement depends on simulator options and the analysis parameters. For an accurate measurement set the following simulation options:

Option	Suggested Value
RELTOL	1e-5
ABSTOL	1e-13
VNTOL	3e-8
TRTOL	1
METHOD	gear
MAXORD	3

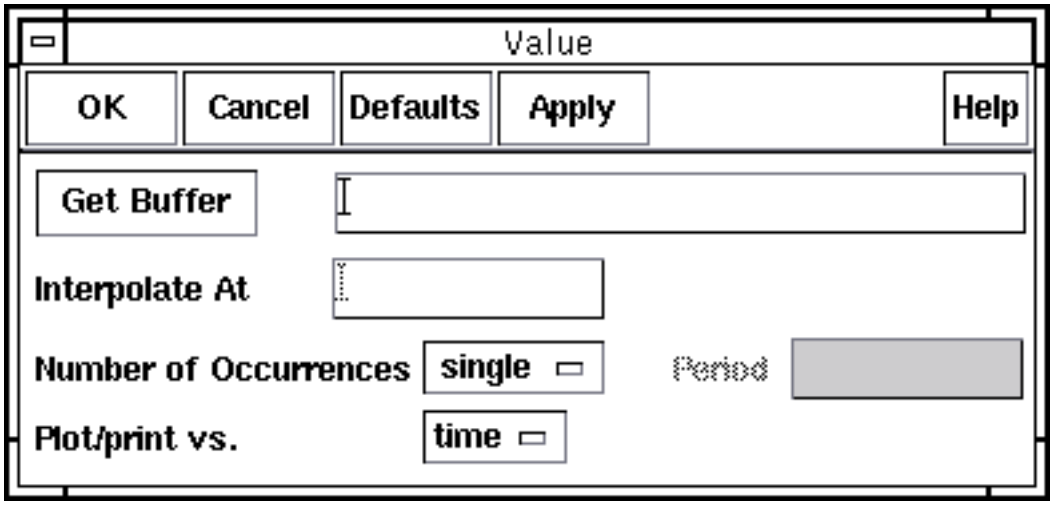
Set the simulation timestep to be 1/100th of a cycle, and simulate for ten cycles. End the simulation slightly beyond the tenth cycle. When you use the calculator, measure during the tenth cycle by specifying the beginning of the cycle as the From time and the end as the To time.

Value Function

The *value* function computes the value of the waveform at the point you specify.

1. Select *value*.

   The Value form appears.



2. Set up an expression in the calculator buffer and click *Get Buffer*.
3. Enter the point at which to compute the value in the *Intrapolate At* field.
4. Specify *Number of Occurrences* as *multiple* or *single*. The default setting is *single*, which retrieves only one occurrence of an interpolated value for a given waveform. If you select *multiple*, you can retrieve all interpolated values for a given waveform, which you can later plot or print.
5. Specify *Plot/print vs.* as *time* or *cycle*. The default setting is *time*, which helps you retrieve value data against time (or another X-axis parameter for non-transient data). For example:

```
time (s)   value(VT("/out") 2e-07 ?period 2e-07 ?xName "time") (V)
--------   --------
200n       4.965
400n       4.965
600n       4.965
800n       4.965
1u         -800.4u
```

   Here, time values refer to each timepoint on the waveform where the interpolated values occur.

The *cycle* option helps you retrieve interpolated values against cycle numbers. For example:

```
cycle      value(VT("/out") 2e-07 ?period 2e-07 ?xName "cycle") (V)
--------   --------
1          4.965
2          4.965
3          4.965
4          4.965
5          -800.4u
```

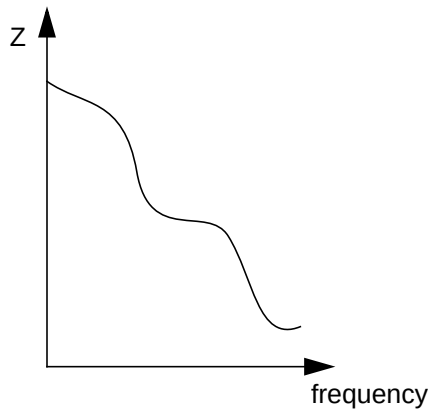
Here, cycle numbers refer to the n'th occurence of the interpolated value in the input waveform.

Note: The value for this field is ignored when *Number of Occurrences* is specified as *single*.

6. Click *OK*.

X Value (xval) Function

The X value function (*xval*) takes a single expression as an argument. It returns a waveform with the Y values equal to the X values: Y==X. This facilitates computations where the dependent variable (such as time or frequency) is needed in an expression. For example, you can use *xval* to compute the capacitance waveform for this curve:



$$\textit{Capacitance Waveform } C(f) = \frac{1}{2\pi Z \cdot xval(Z)}$$

Here are some syntax examples:

```
xval(VF("/IN"))
xval(VT("/net3"))
```

1. Select *xval*.

2. Enter the expression and the closing parenthesis.

5

RF Mode

The waveform calculator supports two sets of buttons or modes: standard and RF. To switch between the two modes, click the *RF* or *standard* button at the top right of the calculator. You can switch repeatedly between the RF and standard modes, because switching does not affect a calculation in progress.

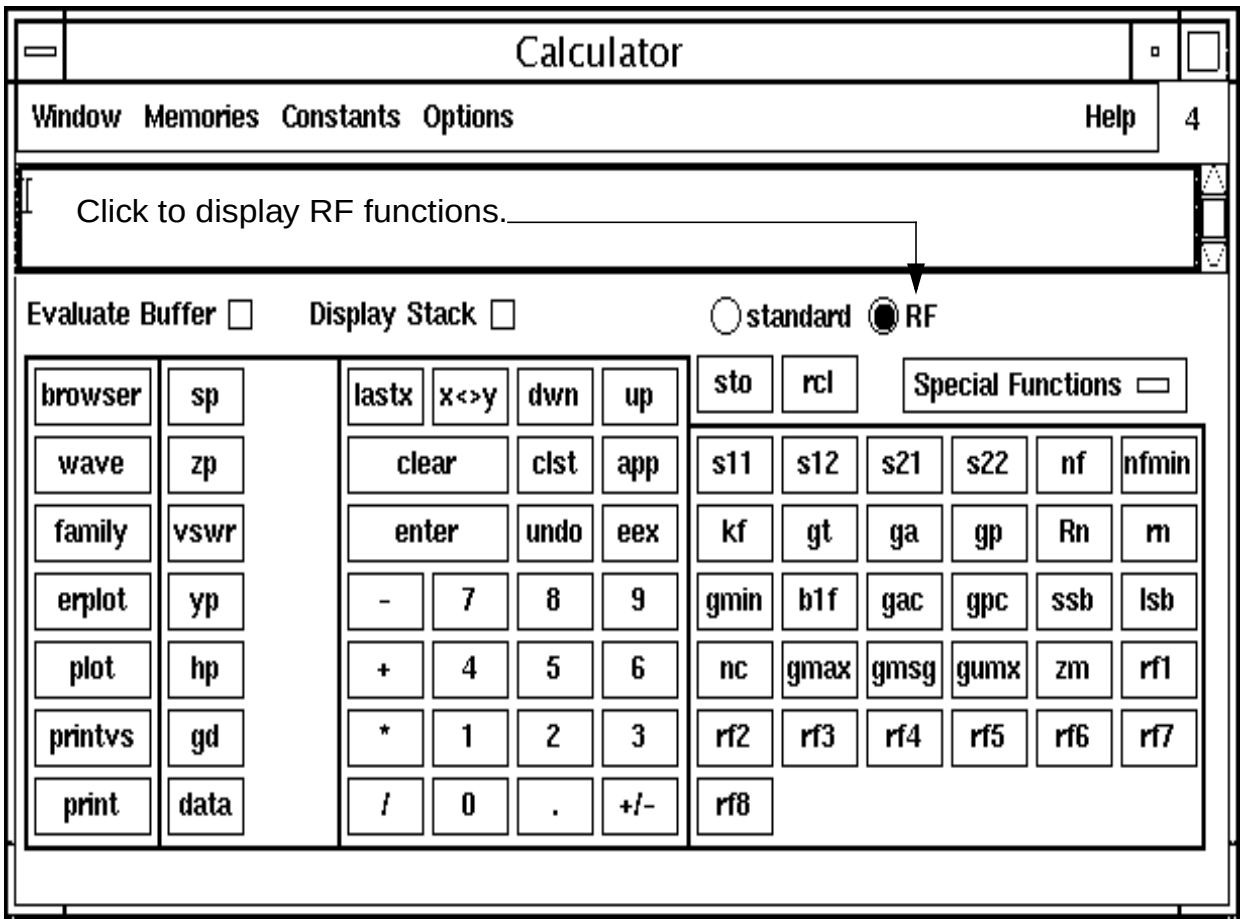
In the RF mode, the waveform calculator provides mathematical functions used for RF circuit design. In this mode the keypad includes the following:

- The same numerical keypad, special functions, and results output buttons as the standard mode.
- Two banks of function buttons for the RF function. Each bank includes functions for linear S-parameter results and for harmonic balance results.

  Note: The RF calculator function is expected to work with SP analysis. So, for these functions to work, you need to run an SP analysis along with any other chosen analyses.
- User-definable function buttons.

Many of the RF button functions are also available on the S-Parameters Results form for direct plotting.

Click *RF* to open the RF version of the calculator.



Click in the picture of the calculator for descriptions of the RF calculator functions, or continue reading here for a general description of the calculator.

The RF mode calculator buttons are used for two types of analyses: S-parameter and harmonic balance. This section describes only the functions that are used with the RF mode. You can also see descriptions and mathematical expressions for the functions associated with each calculator key.

Dual port functions relate one quantity to another. An example is SP(2 1), which expresses the reflected wave at port 2 in terms of the incident wave at port 1. The *s11* button yields the expression SP(1 1), and the expression is complete. This approach is limited to two ports because the number of *s##* buttons equals the upper limit of the number of ports squared. For more ports, use the *sp* function. Other dual port functions are *zp*, *yp*, and *hp*.

To enter the *sp* expression (or any other dual port expression):

1. Click on the *sp* button. The prompt “Select the reflected port” appears at the bottom of the CIW and the design window.
2. In the schematic, click on a *psin* element from *analogLib*. This is the reflected (signal) port. The function does not let you select another type of object. The prompts change to “Select the incident port.”
3. Click on the incident (reference) port. The complete two-port expression appears in the calculator buffer.

Single port functions involve only one port. The only single port function on the calculator is *vswr*. To use *vswr*:

1. Click on the *vswr* button. The prompt “Select the port for expression VSWR…” appears at the bottom of the CIW and the design window.
2. In the schematic, click on one of the *psin* elements of *analogLib*. The function does not let you select another type of object. The expression appears in the buffer.

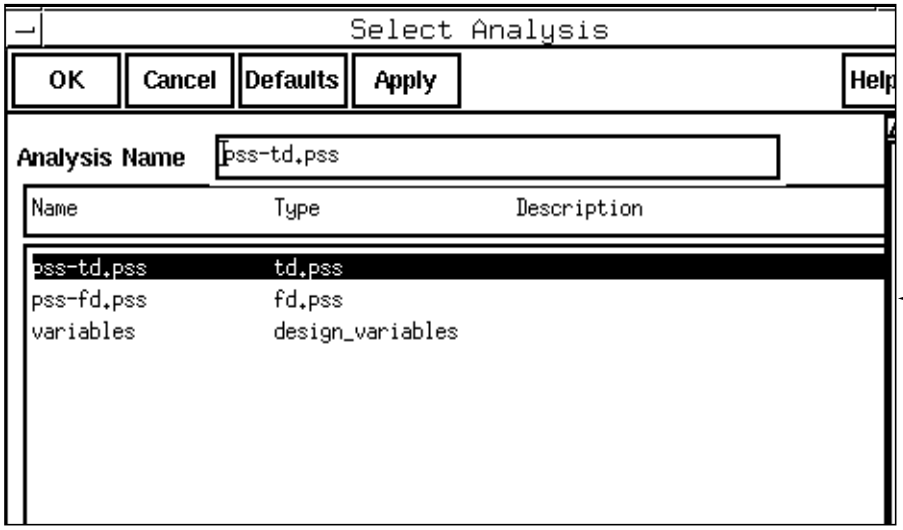
Constant performance circles include *gac*, *gpc*, *ssb*, *lsb*, and *nc***.** Use these functions only when the Waveform window is set to display a Smith chart.

Programmable Buttons

The *rf#* buttons along the bottom row of the right bank of calculator buttons are user function buttons. You can program them to perform whatever function you want, just as you can program the *f#* functions on the standard bank of buttons. Refer to “Assigning Function Keys” on page 43 for information about programming user function buttons.

Data Button

When you click on the *data* button, the Select Analysis form opens up, listing all of the analyses already performed on the current design.



The design window and CIW prompt you to “Select net, terminals, or instance for the DATA expression.” Click on the name of an analysis to select it. You can use this function to quickly review various analyses that you have just performed.

Calculator Buttons for RF

Button	Result	Arguments
For Linear S-Parameter Results:		
sp	Scattering parameters	Signal & reference port #s
yp	Admittance parameters	Signal & reference port #s
zp	Impedance parameters	Signal & reference port #s
hp	H-parameters	Signal & reference port #s
gd	Group delay	Signal & reference port #s
data	Plots a previous analysis	Net or terminal name and analysis name
gmsg	Maximum stable power gain for a two port.	Signal & reference port #s
gmax	Maximum available gain for a two port	Signal & reference port #s
gumx	Maximum unilateral power gain for a two port	Signal & reference port #s
zm	Input impedance if all other ports are matched	

Button	Result	Arguments
vswr	Voltage standing wave ratio	A port number
s11, s12, s21, s22	2-port S-parameters	None
*nf**	Noise factor For more information about this function, please refer **Note** given at the end of this table.	Γ_S (the source reflection coefficient) can be a single complex value or a waveform. Default is 0 (50 ohms).
*nfmin**	Minimum noise factor For more information about this function, please refer **Note** given at the end of this table.	None
kf	Stability factor K	None
gt	Transducer gain	Γ_S and Γ_L (source & load reflection coefficients). Each can be a single complex value or a waveform. Default is 0 (50 Ohms).
ga	Available gain	Γ_S (source reflection coefficient) can be a single complex value or a waveform. Default is 0 (50 Ohms).
gp	Power gain	Γ_L (load reflection coefficient) can be a single complex value or a waveform. Default is 0 (50 Ohms).
Rn	Equivalent noise resistance	None
rn	Normalized equivalent noise resistance	None
gmin	Optimum noise reflection coefficient for NFmin	None
b1f	Stability factor b1f	None
gac	Available gain circles	Gain circle value in dB
gpc	Power gain circles	Gain circle value in dB
ssb	Source stability circles	None

Button	Result	Arguments
lsb	Load stability circles	None
nc	Noise circles	Noise figure value in dB

Note: *nf* and *nfmin* buttons in Calculator refer to noise factors. These buttons simply retrieve the *F* and *Fmin* respectively from the psf file. In the psf file, *NF* (Noise Figure), *NFmin* (Minimum Noise Figure), *F* (Noise Factor), and *Fmin* (Minimum Noise Factor) exist. They are related to each other by the following set of equations.

NF = db10(F)

NFmin= db10(Fmin)

The user can compute the values for noise figure, by taking the *db10* value of the corresponding noise factor data, using the *db10* Calculator button.

Special Function Buttons for RF

Clicking *Special Functions* in the calculator opens a menu of functions that you can use in RF analyses. For more information, follow the cross-references in the following table.

Function	For RPN mode, see	For algebraic mode, see
average	"Average Function" on page 55	"Average Function" on page 110
bandwidth	"Bandwidth Function" on page 55	"Bandwidth Function" on page 110
clip	"Clip Function" on page 58	"Clip Function" on page 114
compression	"Compression Function" on page 59	"Compression Function" on page 115
compressionVRI	"CompressionVRI Function" on page 60	"CompressionVRI Function" on page 116
convolve	"Convolution (convolve) Function" on page 61	"Convolution (Convolve) Function" on page 117
cross	"Threshold Crossing (cross) Function" on page 62	"Threshold Crossing (cross) Function" on page 118
dBm	"dBm Function" on page 64	"dBm Function" on page 121
delay	"Delay Function" on page 64	"Delay Function" on page 121

Function	For RPN mode, see	For algebraic mode, see
deriv	“Derivative (deriv) Function” on page 66	“Derivative (deriv) Function” on page 124
dft	“Discrete Fourier Transform (dft) Function” on page 66	“Discrete Fourier Transform (dft) Function” on page 124
dftbb	“Discrete Fourier Transform Baseband (dftbb) Function” on page 70	“Discrete Fourier Transform Baseband (dftbb) Function” on page 129
evmQpsk	evmQpsk Function on page 71	evmQpsk Function on page 130
eyeDiagram	eyeDiagram Function on page 73	eyeDiagram Function on page 132
flip	“Flip Function” on page 74	“Flip Function” on page 133
fourEval	“Fourier Evaluation (fourEval) Function” on page 74	“Fourier Evaluation (fourEval) Function” on page 133
freq	“Freq Function” on page 75	“Freq Function” on page 133
frequency	“Frequency Function” on page 75	“Frequency Function” on page 134
gainBwProd	“Gain (gainBwProd/gainMargin) Functions” on page 76	“Gain (gainBwProd/gainMargin) Functions” on page 135
gainMargin	“Gain (gainBwProd/gainMargin) Functions” on page 76	“Gain (gainBwProd/gainMargin) Functions” on page 135
groupDelay	“Group Delay Function” on page 76	“Group Delay Function” on page 135
harmonic	“Harmonic Function” on page 76	“Harmonic Function” on page 135
harmonicFreq	“Harmonic Frequency Function” on page 77	“Harmonic Frequency Function” on page 136
iinteg	“iinteg Function” on page 78	“iinteg Function” on page 137
integ	“integ Function” on page 78	“integ Function” on page 137
intersect	“intersect Function” on page 78	“intersect Function” on page 138
ipn	“ipn Function” on page 79	“ipn Function” on page 138
ipnVRI	“ipnVRI Function” on page 80	“ipnVRI Function” on page 139

Function	For RPN mode, see	For algebraic mode, see
lshift	"Lshift Function" on page 82	"Lshift Function" on page 141
overshoot	"Overshoot Function" on page 84	"Overshoot Function" on page 143
phaseMargin	"Phase Margin Function" on page 87	"Phase Margin Function" on page 145
phaseNoise	"Phase Noise Function" on page 87	"Phase Noise Function" on page 146
psd	"Power Spectral Density (psd) Function" on page 89	"Power Spectral Density (psd) Function" on page 148
psdbb	"Power Spectral Density Baseband (psdbb) Function" on page 90	"Power Spectral Density Baseband (psdbb) Function" on page 149
riseTime	"Rise Time Function" on page 91	"Rise Time Function" on page 151
rms	"Root-Mean-Square (rms) Function" on page 93	"Root-Mean-Square (rms) Function" on page 153
rmsNoise	"Root-Mean-Square (rms) Noise Function" on page 94	"Root-Mean-Square (rms) Noise Function" on page 154
root	"Root Function" on page 94	"Root Function" on page 154
sample	"Sample Function" on page 95	"Sample Function" on page 155
settlingTime	"Settling Time Function" on page 95	"Settling Time Function" on page 155
slewRate	"Slew Rate Function" on page 97	"Slew Rate Function" on page 158
spectralPower	"Spectral Power Function" on page 99	"Spectral Power Function" on page 159
stddev	"Standard Deviation (stddev) Function" on page 99	"Standard Deviation (stddev) Function" on page 160
table	"Table Function" on page 100	"Table Function" on page 160
tangent	"Tangent Function" on page 101	"Tangent Function" on page 161
thd	"Total Harmonic Distortion (thd) Function" on page 101	"Total Harmonic Distortion (thd) Function" on page 162

Function	For RPN mode, see	For algebraic mode, see
value	“Value Function” on page 102	“Value Function” on page 164
xmax	“xmax and ymax” on page 83	“xmax and ymax” on page 142
xmin	“xmin and ymin” on page 84	“xmin and ymin” on page 143
xval	“X Value (xval) Function” on page 104	“X Value (xval) Function” on page 165
ymax	“xmax and ymax” on page 83	“xmax and ymax” on page 142
ymin	“xmin and ymin” on page 84	“xmin and ymin” on page 143

A

Using the Calculator Special Functions with SpectreRF Simulation Results

This appendix shows you how to use several of the special functions in the Analog Design Environment Waveform Calculator with RF simulation results. It illustrates how to

- Use the `ipn` and `ipnVRI` Calculator special functions in RPN and Algebraic modes to plot IP3.
- Use the `ipn` and `ipnVRI` special functions with scalar inputs.
- Use the `spectralPower` special function to plot spectral power.

This appendix is designed for users of SpectreRF in the Analog Design Environment. It assumes that users are familiar with SpectreRF and its swept PSS, PAC, and PSS analyses.

If you need more information on SpectreRF than is provided in this appendix, see the *SpectreRF User Guide*.

Using the ipn and ipnVRI Calculator Special Functions

This example uses an editable copy of the *ne600p* schematic from a copy of the *rfExamples* library. As you follow the example, you

- Perform a swept PSS analysis followed by a PAC small-signal analysis to produce data for several IP3 plots.
- Plot IP3 using the Direct Plot form.
- Plot IP3 several ways using the Calculator *ipn* and *ipnVRI* special functions.
- Compare the different results.

Setting Up the Simulation Environment

If you need more information than is provided in this appendix, see the *SpectreRF User Guide*. For example, Chapter 3 in the *SpectreRF User Guide* describes in detail how to create the editable copy of the *rfExamples* library of schematics.

Start the CIW, Open the Schematic, and Set Up the Simulator and Models

1. Start the Command Interpreter Window (CIW) by typing `icms` in a terminal window.

   Make sure that you have copied the *ne600p* schematic from the *rfExamples* library in the Cadence installation to your local library of editable schematics (*my_rfExamples* in this example) so you will have edit permissions on the schematic.

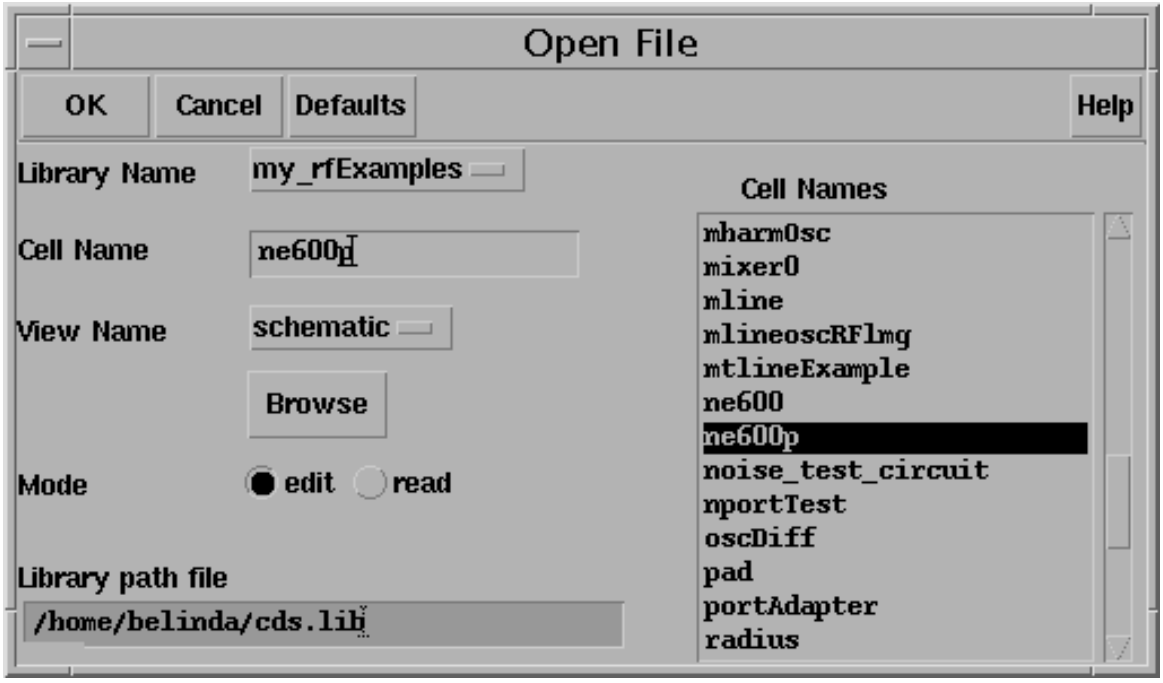
2. In the CIW, choose *File—Open.*

3. In the Open File form

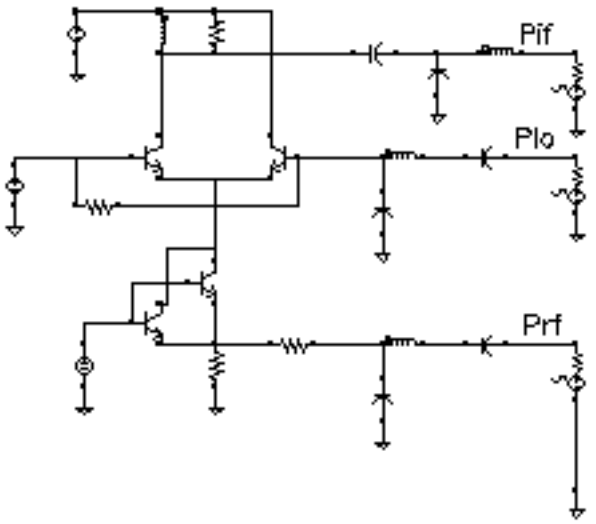
   a. Select your local library of editable schematics.

   b. Select the *ne600p* schematic for editing.

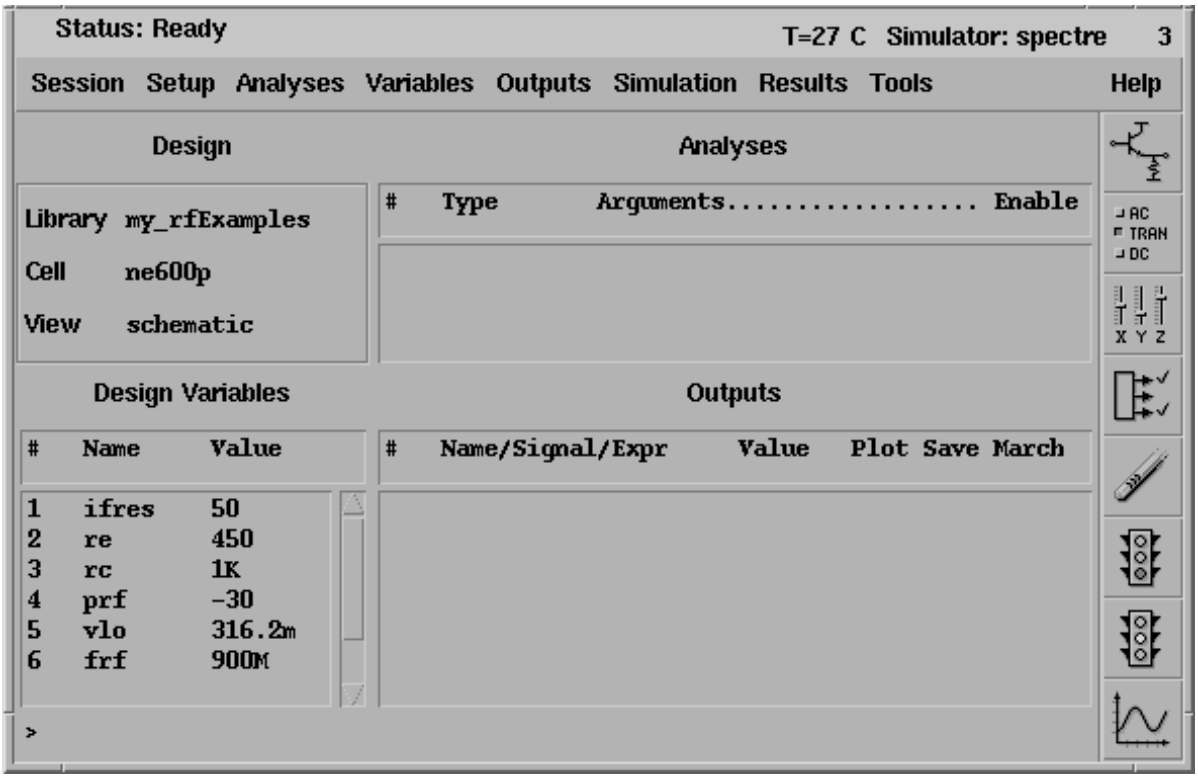
   c. Click *OK*.



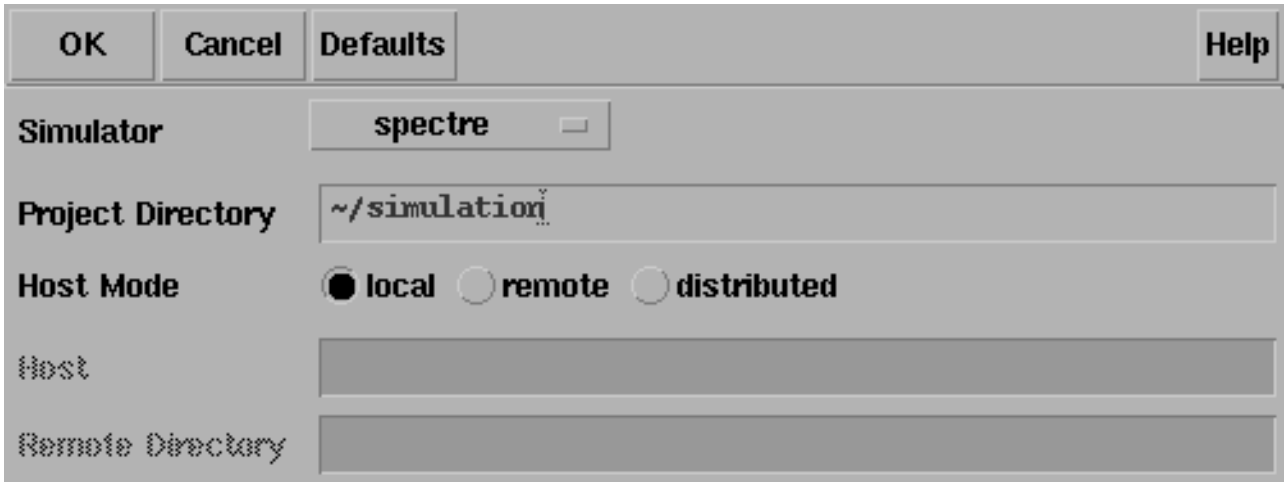
The Schematic window opens with the *ne600p* schematic displayed.



4. In the Schematic window, choose *Tools – Analog Environment* to open the Simulation window.



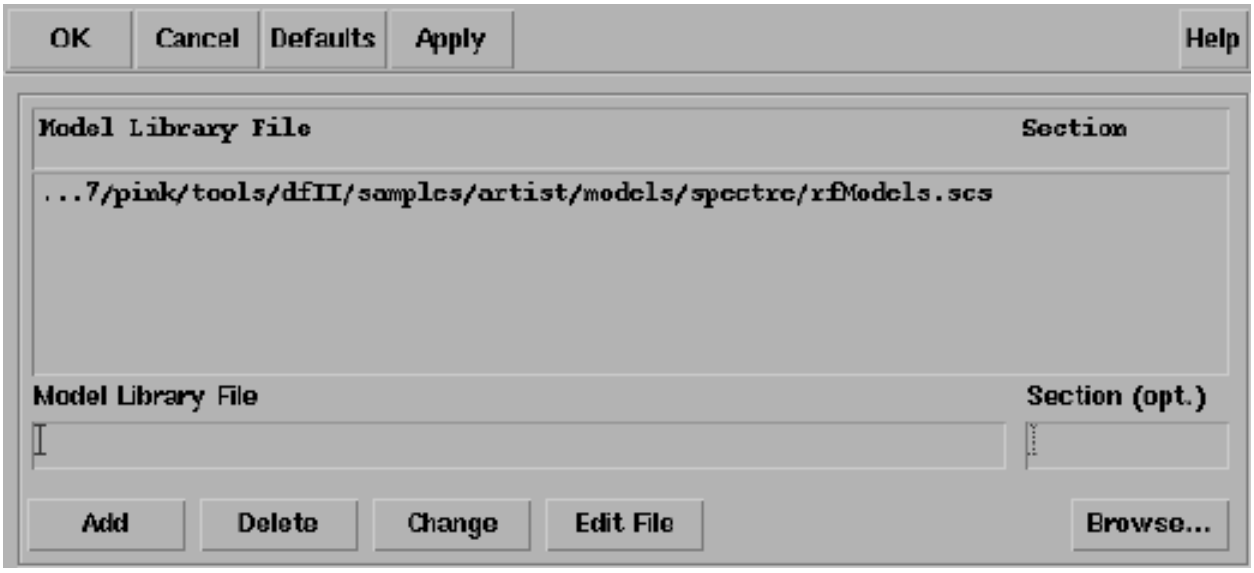
5. In the Simulation window, choose *Setup—Simulator/Directory/Host* to open the Choosing Simulator/Directory/Host form. Make the following selections and click *OK*.



a. Choose *spectre* in the *Simulator* cyclic field.

b. Click *local* to set *Host Mode*.

6. In the Simulation window, choose *Setup—Model libraries* to display the Model Library Setup form.



7. In the Model Library Setup form

a. In the *Model Library File* field, type the path to the library of models.

`<path_to>/tools/dfII/samples/artist/models/spectre/rfModels.scs`

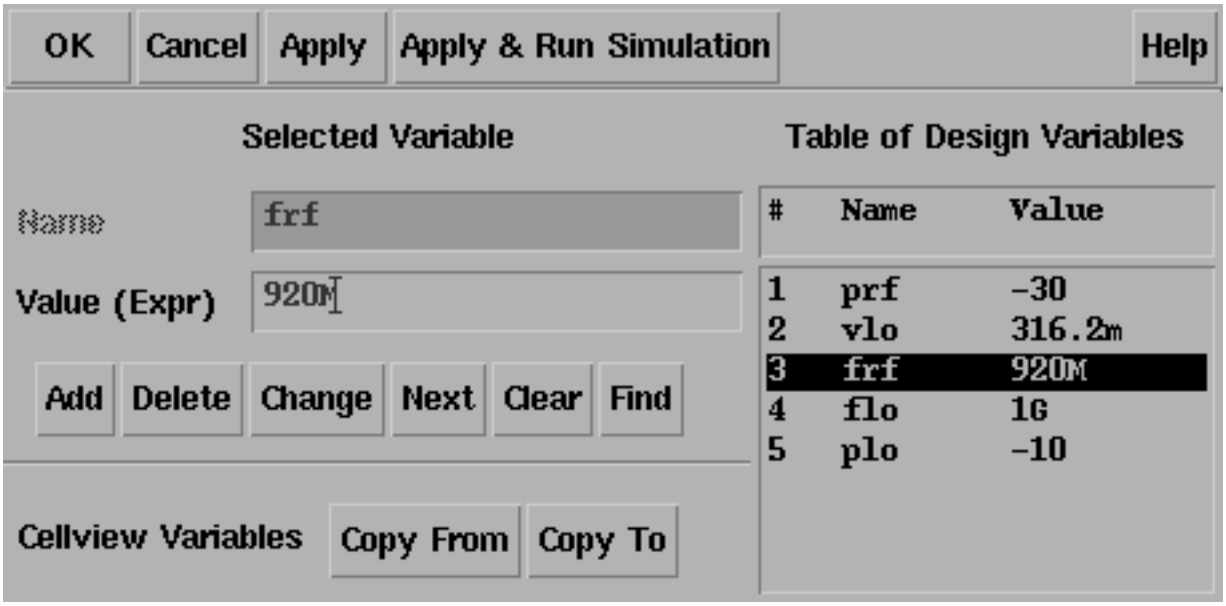
In the example, replace <*path_to*> with the installation directory for your Cadence software.

b. Click *Add*.

c. Click *OK*.

Edit the frf Design Variable

1. In the Simulation window, choose *Variables—Edit* to display the Editing Design Variables form and edit the value of the frf variable.



2. In the Editing Design Variables form

a. In the *Table of Design Variables*, click `frf 900M`.

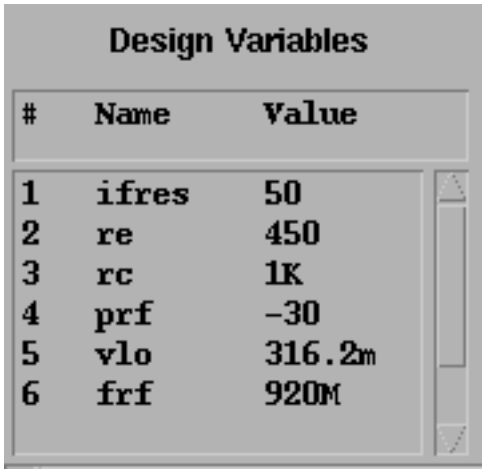
The variable name and value display in the *Name* and *Value (Expr)* fields.

b. Set the variable value in the *Value (Expr)* field to `920M` and click *Change*.

The modified value displays in the *Table of Design Variables*, `frf 920M`.

c. Click *OK*.

The modified variable value displays in the *Design Variables* area of the Simulation window.

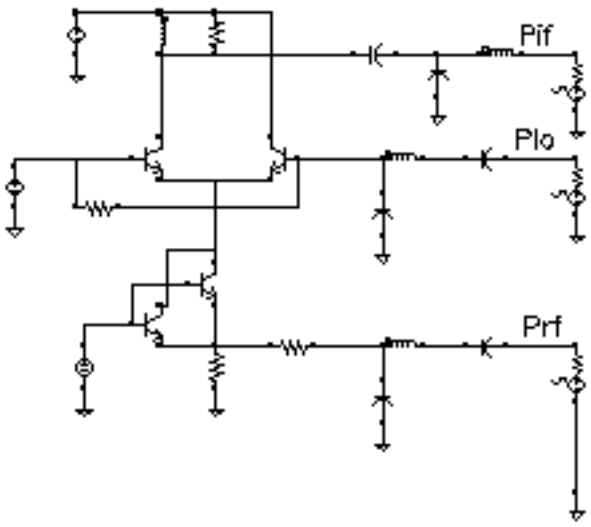


3. In the Simulation window, if necessary, choose *Analyses—Disable* to disable any analyses you might have run previously.

Editing the Schematic

Modify and verify the *port* sources in the *ne600p* schematic which is displayed in Figure A-1.

Figure A-1 The ne600p Schematic



1. In the Schematic window, click on the *rf port* source to select it.

   The *rf port* source is highlighted.

2. Choose *Edit—Properties—Objects* to display the Edit Object Properties form.

   The Edit Object Properties form displays with the current information for the *rf port* source.

3. In the Edit Object Properties form for the *rf port*, do the following.

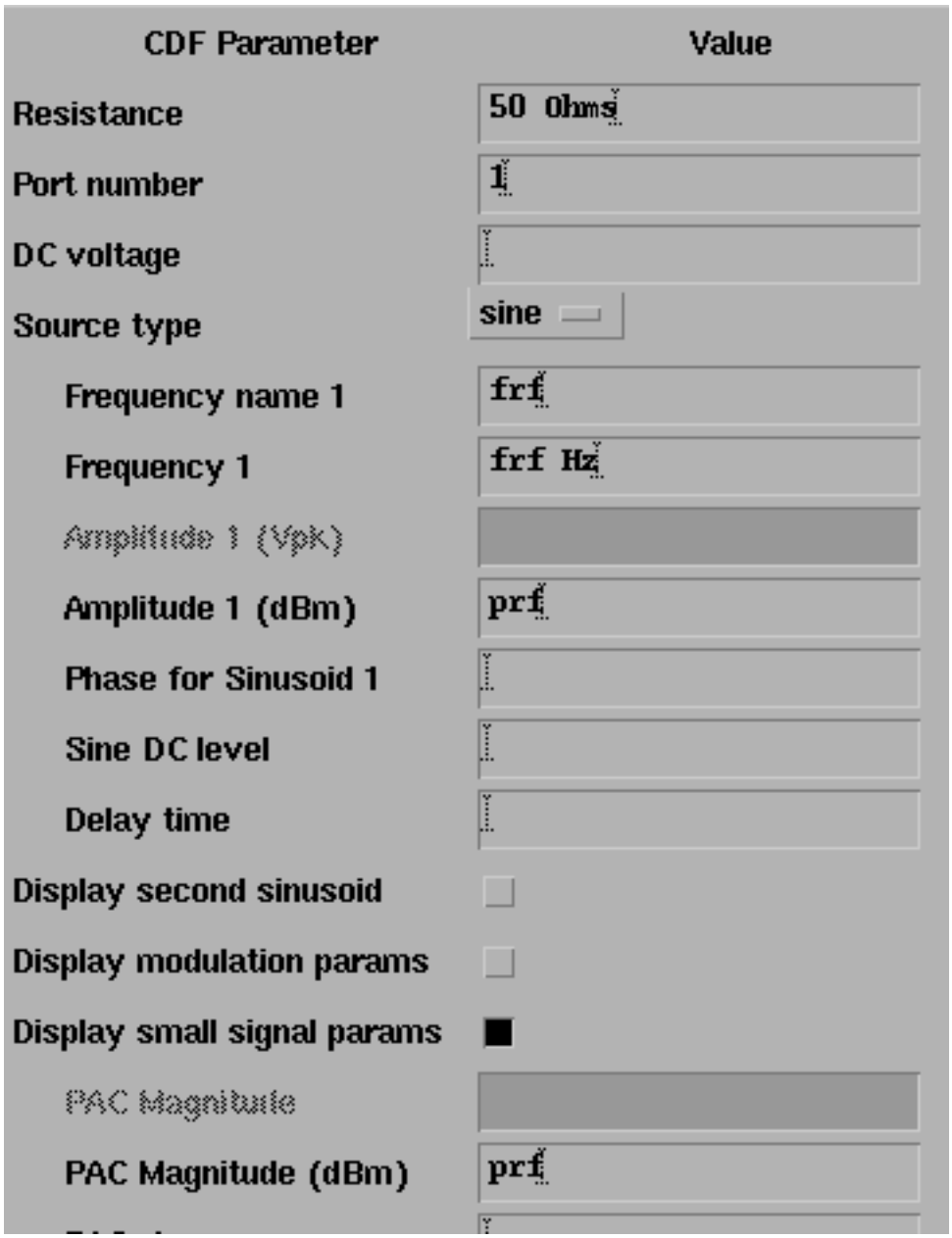
   a. Select *sine* from the *Source type* cyclic field.

   b. Type `frf` in the *Frequency 1* and *Frequency name 1* fields.

   c. Type `prf` in the *Amplitude 1(dBm)* and *PAC magnitude (dBm)* fields. (The *PAC magnitude (dBm)* field is close to the bottom of the form. You may have to scroll down to see it.) This simulation uses dBm rather than magnitude values.

   d. Click *Apply*.

The top of the CDF parameter section in the Edit Object Properties form for the *rf port* looks like the following.



4. Click on the *lo port* source to select it and verify its properties.

The Edit Object Properties form changes to display properties for the *lo port* source.

Verify that its S*ource type* is set to `sine` and click *Apply*.

5. Click on the *if port* source to select it and verify its properties.

   The Edit Object Properties form changes to display properties for the *if port* source.

   Verify that its S*ource type* is set to `sine` and click *Apply*.

6. Click *OK* in the Edit Object Properties form.

Check and Save the Design

➤ In the Schematic window, choose *Design—Check and Save*.

Messages display in the CIW as the schematic is checked and saved.

Setting Up the Analyses

➤ In the Simulation window, choose *Analyses—Choose* to display the Choosing Analyses form.

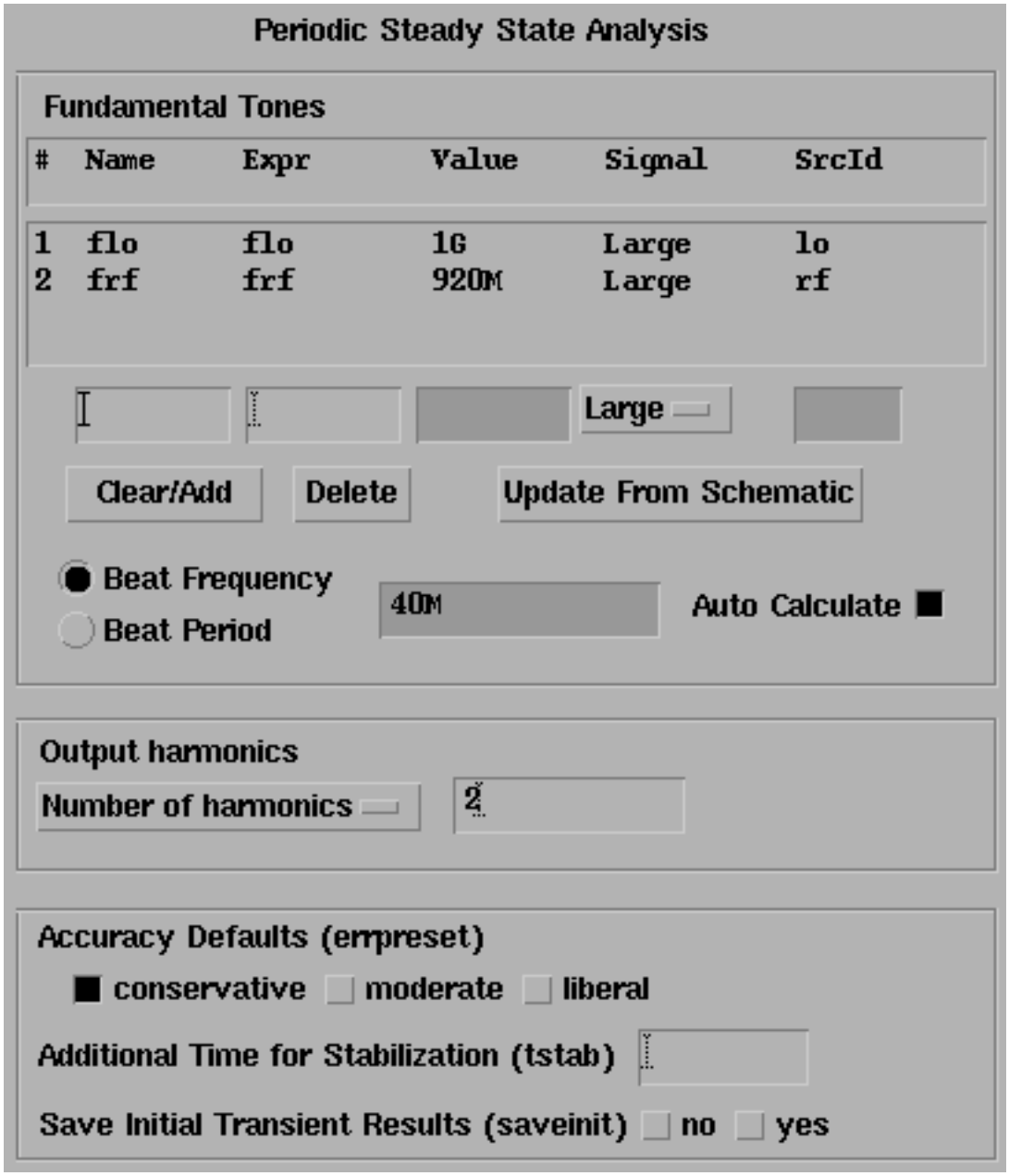
Setting Up the Swept PSS Analysis

1. In the Choosing Analyses form, click `pss` to display fields for the Periodic Steady State analysis.
2. Below the *Fundamental Tones* list box, if necessary, click *Beat Frequency* and then click the *Auto Calculate* button.

   The value `40M` displays in the *Beat Frequency* field.

   The PAC analysis is responsible for the two tones, and the PSS is now a single signal analysis. Consequently, the fundamental frequency is set to `40 MHz`.
3. In the *Output harmonics* cyclic field, select *Number of harmonics* and enter 2 in the *Number of harmonics* field.

4. Click *conservative* for the *Accuracy Defaults (errpreset)* setting. The top of the PSS Choosing Analysis form looks as follows.



5. Click *Sweep*.

The form changes to let you specify data to sweep a variable. You might need to scroll down to display all the sweep information.

a. In the *Sweep* cyclic field, select *Variable*.

b. For *Frequency Variable?*, click *no*.

c. Click *Select Design Variable*.

The Select Design Variable form displays.

a. In the Select Design Variable form, select `prf` and click *OK*.

The variable name `prf` displays in the *Variable Name* field on the Choosing Analysis form.

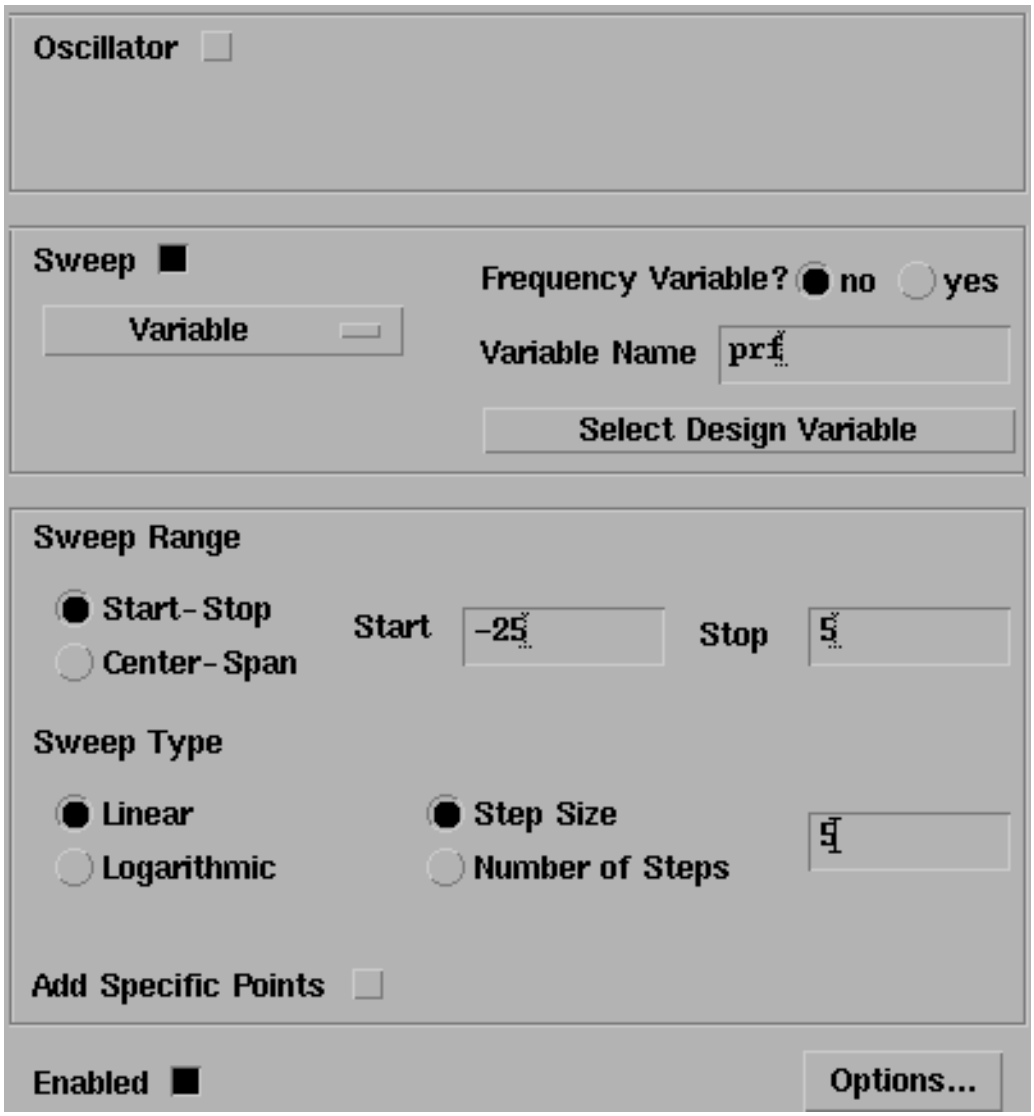
6. Click *Start-Stop* for the *Sweep Range* and then enter `-25` and `5` in the *Start* and *Stop* fields, respectively.

The sweep is skewed downward for a down converter.

7. Click *Linear* for the *Sweep Type*, then click *Step Size* and enter `5` in the *Step Size* field.

8. If necessary, click *Enabled*.

The bottom of the PSS Choosing Analysis form looks as follows.



9. Click *Apply*.

Setting Up the PAC Small Signal Analysis

1. At the top of the Choosing Analyses form, select `pac` to display fields for the Periodic AC small signal analysis.

2. In the *Frequency Sweep Range (Hz), Freq* field, enter `921M`

Because the sweep section of the PSS analysis is active, only a single point is supported for the PAC analysis.

3. In the *Sidebands* cyclic field, select *Array of indices.*
4. In the *Additional indices* field enter the values –21 and –25 separated by a space.

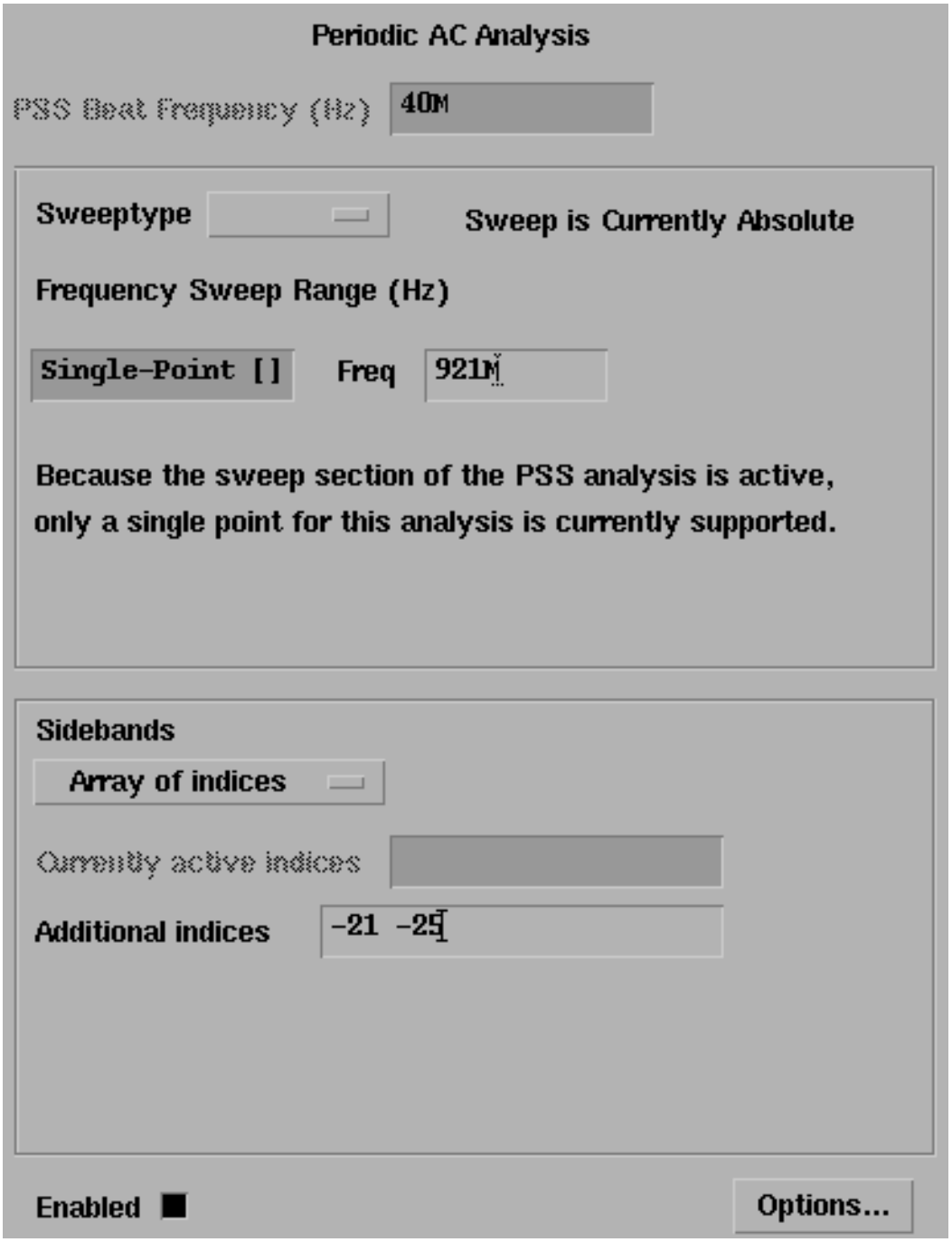
   Given the following

   - A Fundamental Tone at 40 MHz
   - The LO at 1GHz
   - Two RF tones at 920 MHz and 921 MHz

   The sidebands of –25 and –21 represent respectively

   - The first-order harmonic of the IF output at 79 MHz ($921 - 25 * 40 = 79$)
   - The third-order harmonic at 81 MHz ($921 - 21 * 40 = 81$)
5. If necessary, click *Enabled* for the PAC analysis.

6. The completed PAC section looks as follows.



7. Click Apply and then click *OK* in the Choosing Analyses form.

In the Simulation window, the *Analysis* area displays the PSS and PAC information you just entered.

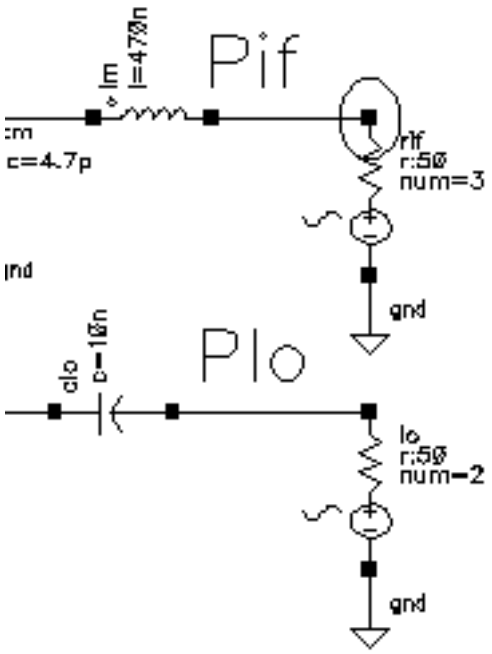
Selecting Outputs to Save

1. In the Simulation window, choose *Outputs—To Be Saved—Select on Schematic*.

   The Schematic window pops to the top.

2. In the Schematic window, click on the positive node of the *rif port*.

   The selected node is circled in the schematic. It is also listed in the *Outputs* section of the Simulation window.



Running the Simulation

In the Simulation window, choose *Simulation—Netlist and Run*.

This example compares two signals only 1 MHz apart.

Plotting the IP3 Curves Using the Direct Plot Form

1. Select *Results—Direct Plot—Main Form*.

The Direct Plot form displays.

Note: As you make selections in the Direct Plot form it might change to reflect each selection.

In the Direct Plot form, do the following:

a. Click on *Replace* for *Plot Mode*.

b. Click on *pac* for *Analysis*.

c. Click on *IPN Curves* for *Function*.

d. In the *Select* cyclic field, select *Port (fixed R(port))*.

e. Choose *Variable Sweep (prf)* for *Circuit Input Power*.

f. Type -15 for *Input Power Extrapolation Point (dBm).*

This value is the intercept point for the ideal amplification extrapolation.

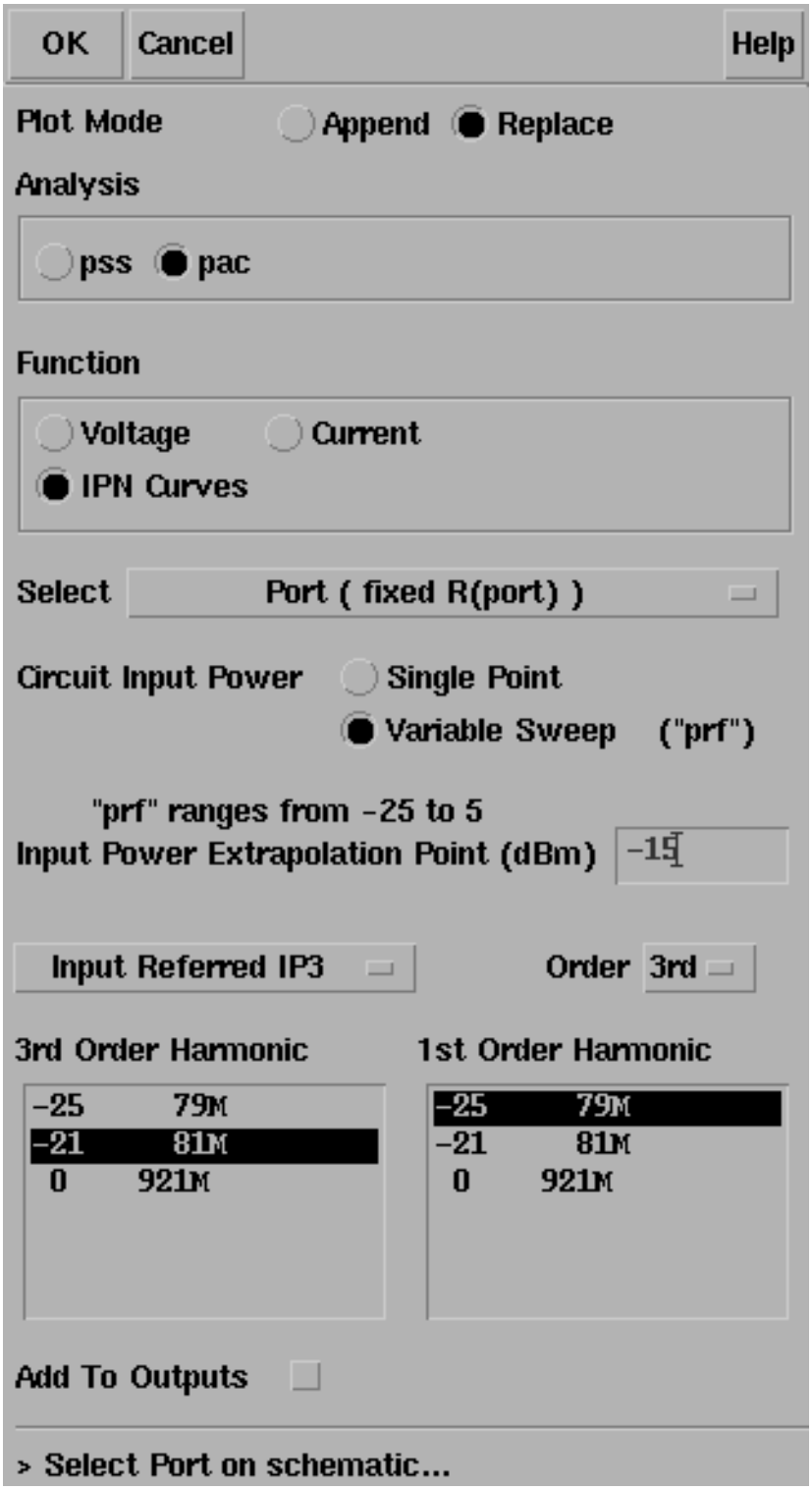
g. Select the *Input Referred IP3* and *Order 3rd* cyclic fields.

h. Highlight *-21 81M* in the *3rd Order Harmonic* list box.

i. Highlight *-25 79M* in the *1st Order Harmonic* list box.

Note: When you select the *Add to Outputs* button in the Direct Plot form, the *Input Referred IP3* Direct Plot expression is also saved and displayed in the *Outputs* pane in the Simulation Window. You will do this in "Using the IPN Special Function with Scalar Inputs" on page 206.

The completed Direct Plot form looks like the one below.



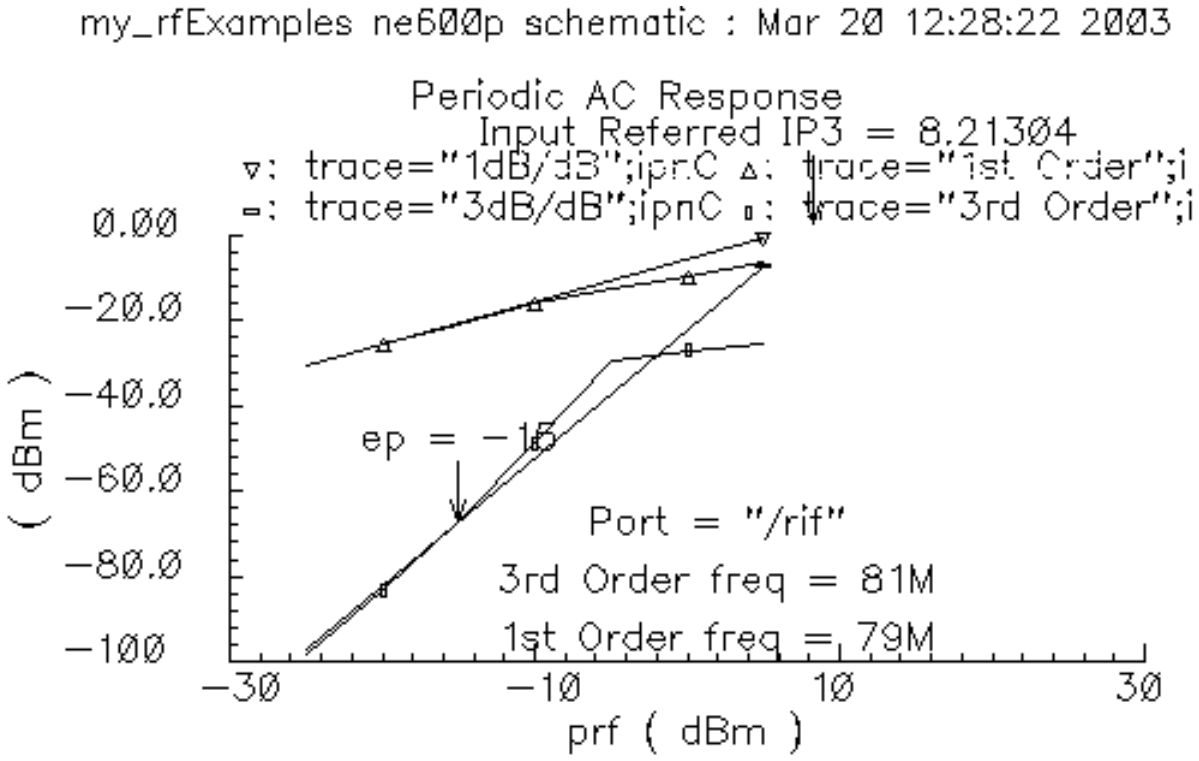
2. Following the prompt at the bottom of the Direct Plot form,

   `Select Port on schematic...`

   In the Schematic window, click on the *rif* port.

   Note: If you click on the *Pif* net (rather than on the *rif* port), you will get an IPN plot with dBV on the vertical axis.

   The Waveform window display appears as shown below



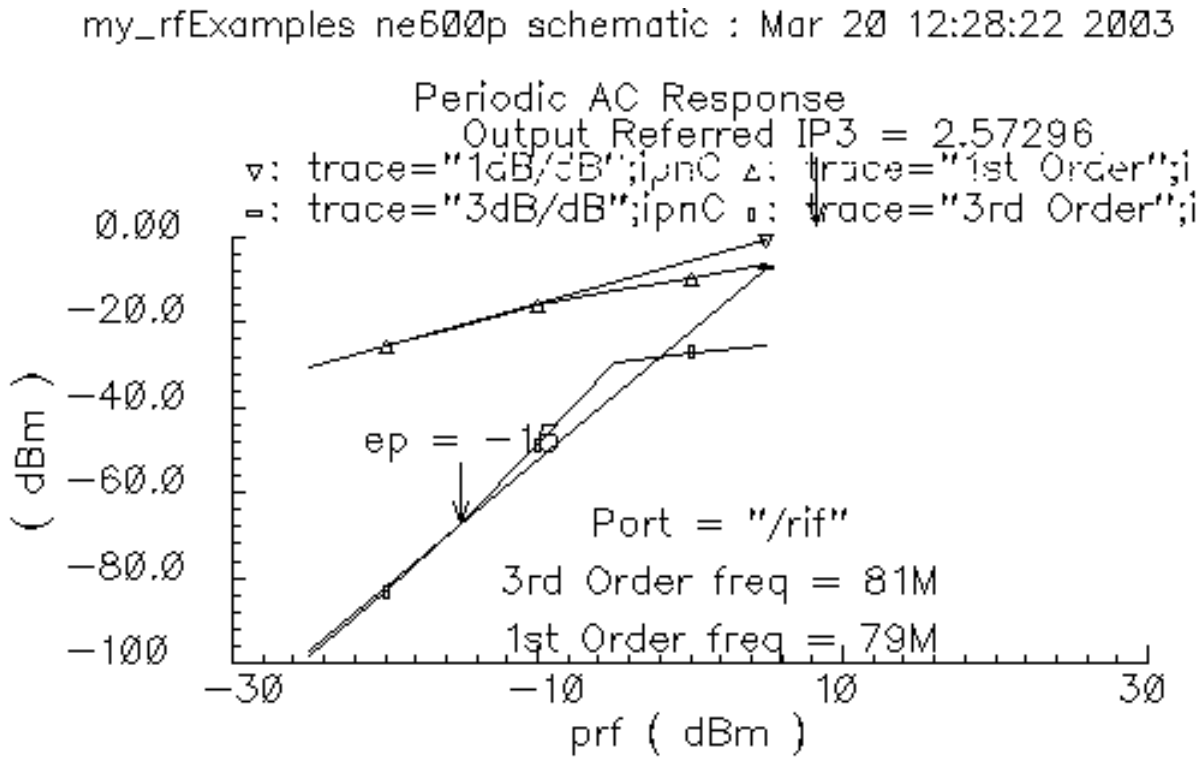
   Note: Notice that the IP3 value and it's pointer and label overwrite the legend for the traces in your plot. You can click and drag them above the legend, as in this plot, if you want.

You can also Plot Output Referred IP3 from the same Direct Plot form.

1. Change the cyclic field to *Output Referred IP3* (from *Input Referred IP3*).
2. Click *Replot*.

   Note: Note that in this case, the measured value is different, depending on whether you select a port (dBm) or a net (dBV) on the schematic.

Output Referred IP3 for the *rif* port is plotted below.



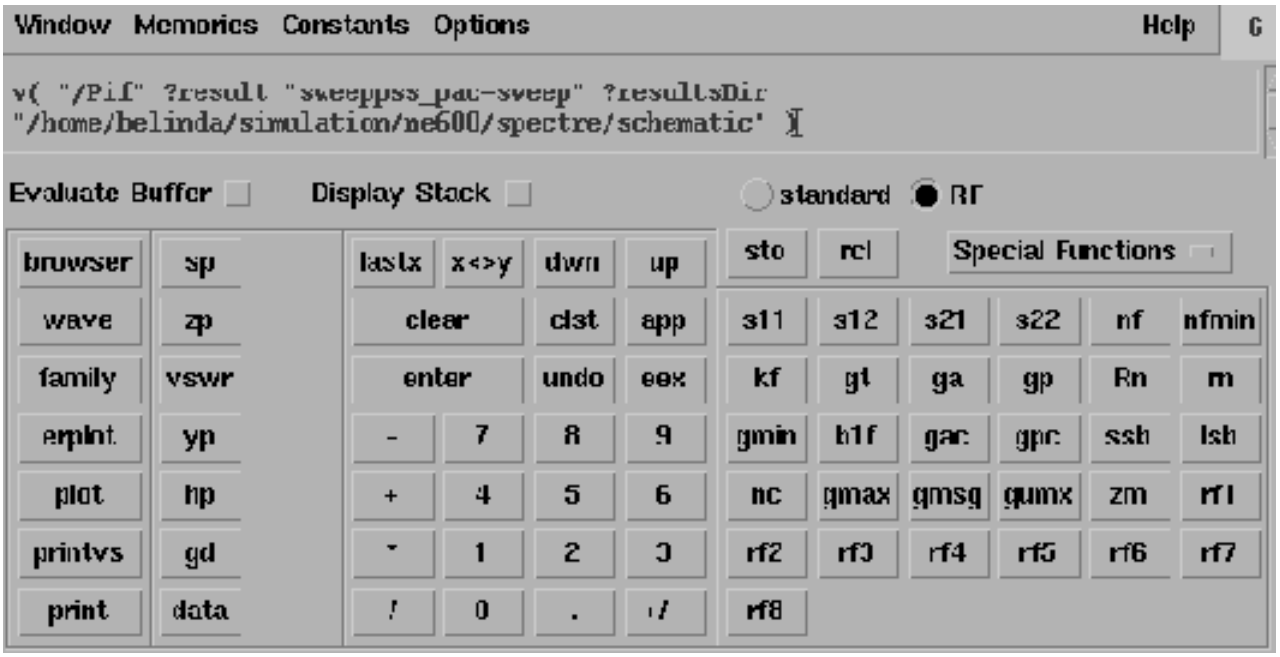
Plotting IP3 Using the Calculator ipn and ipnVRI Special Functions in the RPN Mode

The next two examples use the *ipn* and *ipnVRI* special functions in the Calculator to create IP3 plots. The *ipn* special function calculates only the Input Referred IP3 value. The *ipnVRI* special function calculates both Input Referred IP3 and Output Referred IP3 values.

Using the ipn Special Function in RPN Mode to Plot Input Referred IP3

In this example, you use the *ipn specialFunction* to plot IP3. The *ipn specialFunction* calculates only the *Input Referred IP3*.

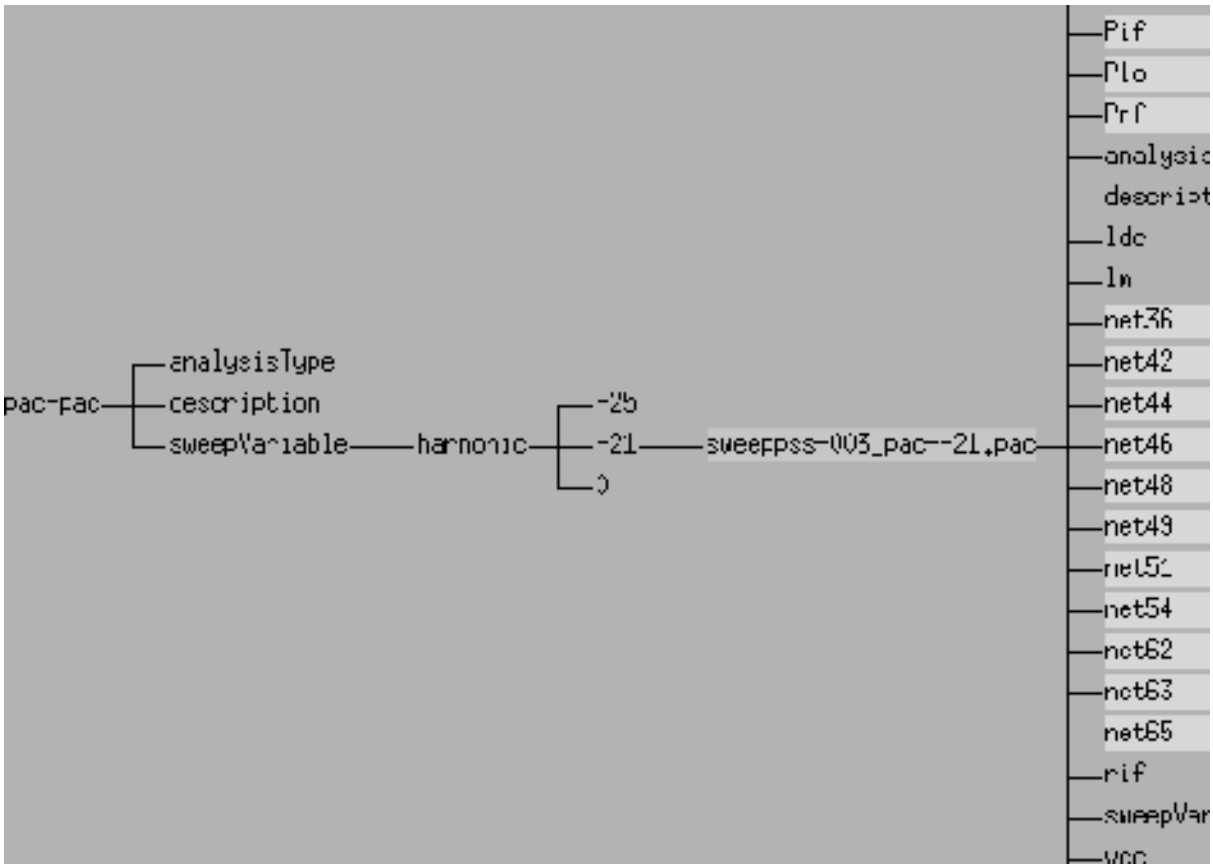
1. From the Simulation window, choose *Tools—Calculator.*



2. In this example, you use the Calculator in RPN mode. If the Calculator is in Algebraic mode, change it by selecting *Options—Set RPN*.
3. In the Calculator, click *browser* to open the Results Browser.
4. When the Browse Project Hierarchy form appears, click OK to display the results browser.
5. In the Results Browser, click on:

```
schematic->psf->Run1->sweeppss_pac-sweep->
sweepVariable->prf-> -10->sweeppss-003_pac-pac->
sweepVariable->harmonic-> -21->
sweeppss-003_pac--21.pac-> Pif
```

Or



The information displays in the Calculator buffer. It looks similar to this:

```
v( "/Pif" ?result "sweeppss_pac-sweep" ?resultsDir
"/home/belinda/simulation/ne600p/spectre/schematic" )
```

Note: Clicking on any harmonic (`-0`, `-21`, or `-25`) in the results browser gives the same results.

The Results Browser shows the basic structure of the saved simulation. data. Since you choose (in "Selecting Outputs to Save" on page 191) to save results for the *rif psin* only, the results browser displays only `Pif` data (voltage and current information).

6. In the Calculator, select *Special Functions—ipn*.

The IPN form displays. In the IPN form, do the following:

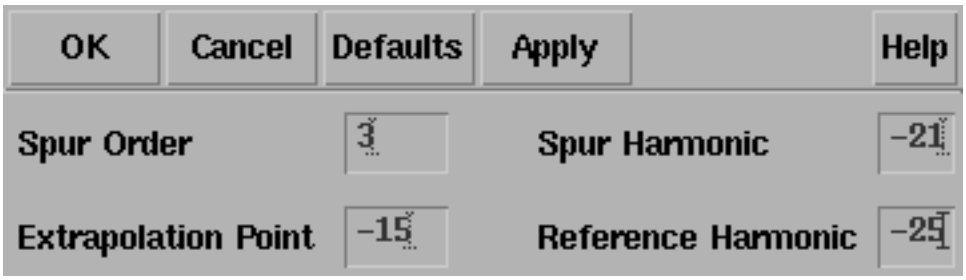
a. Enter 3 in the *Spur Order* field.

b. Enter –21 in the *Spur Harmonic* field.

c. Enter –15 in the *Extrapolation Point* field.

d. Enter –25 in the *Reference Harmonic* field.

e. Click *OK*.



7. The Calculator buffer contains an expression similar to the bottom lines of the one below:

```
ipn(dB20(harmonic(v( "/Pif" ?result "sweeppss_pac-sweep" ?resultsDir
"/home/belinda/simulation/ne600p/spectre/schematic" ), -21)),
dB20(harmonic(v( "/Pif" ?result "sweeppss_pac-sweep"
?resultsDir "/home/belinda/simulation/ne600p/spectre/schematic" ), -25)),
3,1,-15,-15)
```

8. Now you have two choices, either:

- Click *Evaluate Buffer*

  Or,

- Cut and paste the Calculator buffer into the CIW.

  The function evaluates to 8.21304. This number agrees with the IP3 value in the Waveform window.

9. Click *Clear*. if necessary, click *Evaluate Buffer* to deselect it.

  The Calculator buffer is cleared.

Using the ipnVRI Special Function in RPN Mode to Plot IP3

In this example, you use *ipnVRI* on the *specialFunctions* cyclic menu to obtain both Input Referred IP3 and Output Referred IP3 values.

1. Click on the following in the Results Browser:

```
schematic–>psf–>Run1–>sweeppss_pac-sweep–> sweepVariable–>prf–>
–10–>sweeppss-003_pac-pac–> sweepVariable–>harmonic–>
–21–>sweeppss-003_pac--21.pac–>Pif
```

  The information displays in the Calculator buffer. It looks similar to this:

```
v( "/Pif" ?result "sweeppss_pac-sweep" ?resultsDir
"/home/belinda/simulation/ne600p/spectre/schematic" )
```

Note: You should select a net voltage only when you are using the Calculator. The Load Resistance field is used in conjunction with this voltage to calculate power (V2/R).

2. Select *specialFunctions—ipnVRI* in the Calculator.

The ipnVRI form displays. In the ipnVRI form, do the following:

a. Enter –21 in the *Spur Harmonic* field.

b. Enter –25 in the *Reference Harmonic* field.

c. Enter 3 in the *Spur Order* field.

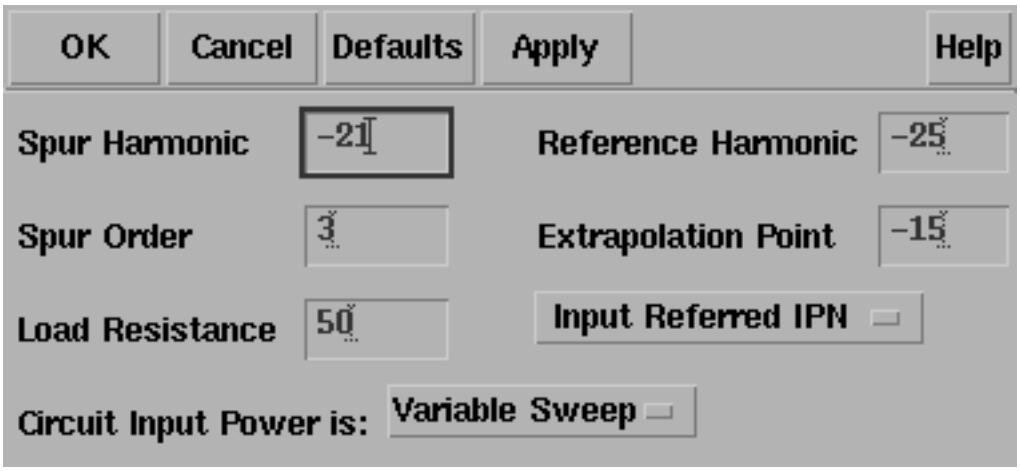
d. Enter –15 in the *Extrapolation Point* field.

e. Enter 50 in the *Load Resistance* field.

f. Select *Input Referred IPN* in the cyclic field.

g. Select *Variable Sweep* in the *Circuit Input Power is:* cyclic field.

h. Click OK.



Note: If you want to plot dBV using the *ipnVRI* function, set the *Load Resistance* to `nil`.

The Calculator buffer has an expression like the one below:

```
ipnVRI(v( "/Pif" ?result "sweeppss_pac-sweep" ?resultsDir
"/home/belinda/simulation/ne600p/spectre/schematic" ),-21,-25,
?ordspur 3,?epoint -15,?rport 50)
```

3. Click *Evaluate Buffer*.

The expression evaluates to 8.21304166.

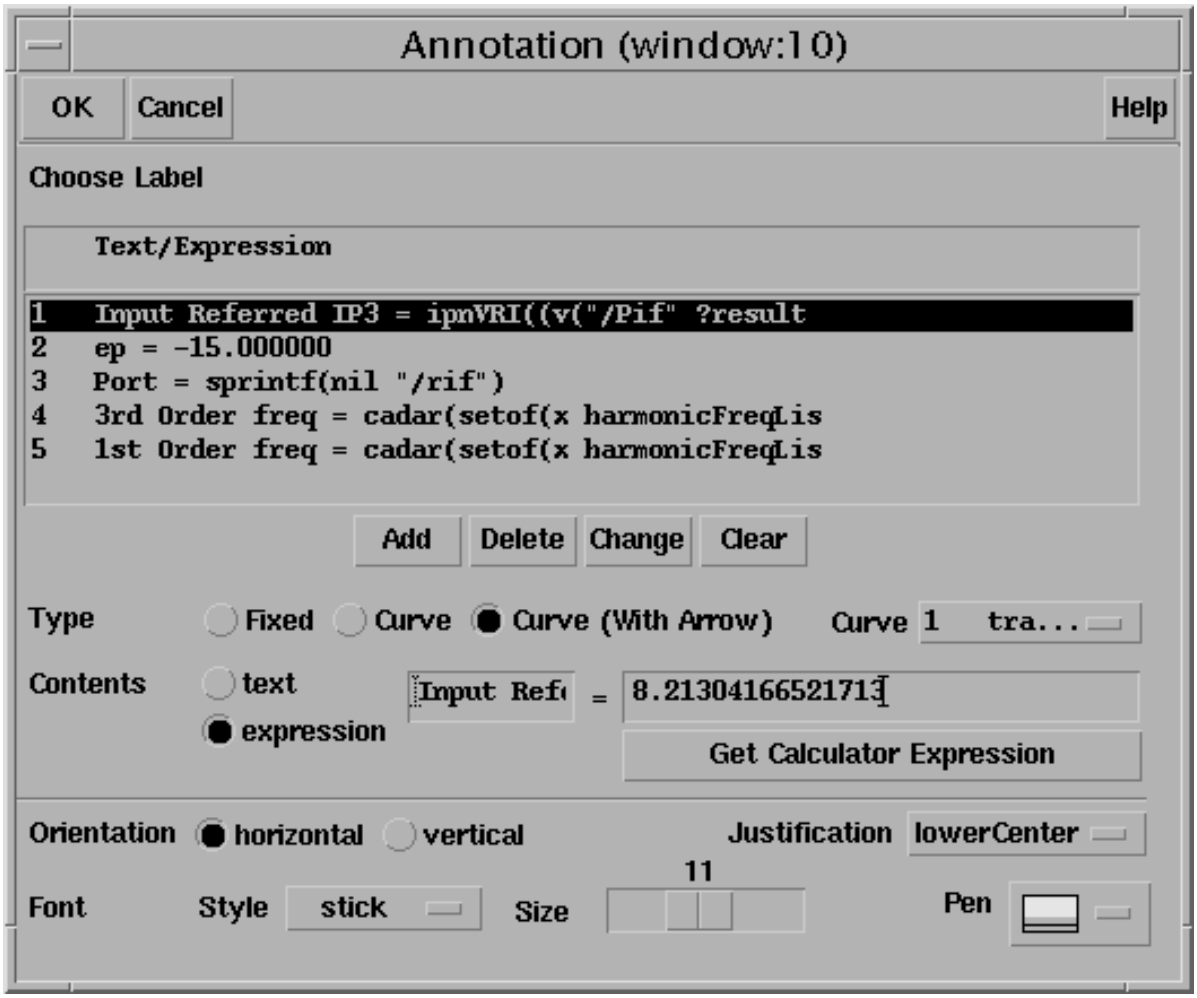
Getting Input Referred IP3 Using the Direct Plot Form

There is yet another way to obtain input referred IP3 used by the Direct Plot form:

1. Go to the Waveform Window and select *Annotation—Edit* to display the Annotation form.
2. In the Annotation form list box, click on *Input Referred IP3*. In the expression field, it shows you the expression for that label.

```
ipnVRI(v("/Pif" ?result "pac")-0. '-21 '-25 ?rport
resultParam("rif:r" ?result "pac") ?epoint -15)
```

3. Click *Get Calculator Expression* to obtain the same IP3 value (8.21304166). The Direct Plot expression uses the *ipnVRI* function, which is used to simplify the declaration of an IPN measurement.



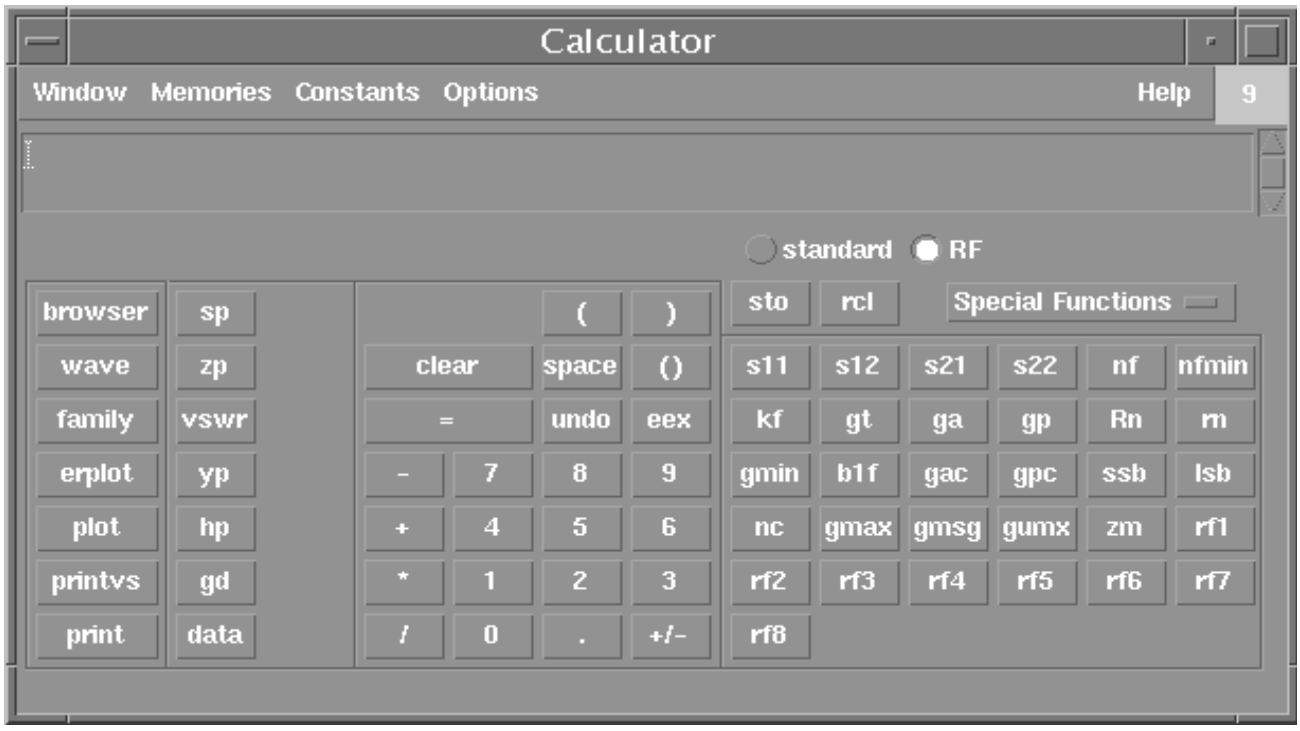
Plotting IP3 Using the Calculator ipn and ipnVRI Special Functions in the Algebraic Mode

The next two examples use the *ipn* and *ipnVRI* special functions in the Calculator to create IP3 plots in the Algebraic mode. The *ipn* special function calculates only the Input Referred IP3 value. The *ipnVRI* special function calculates both Input Referred IP3 and Output Referred IP3 values.

Using the ipn Special Function in Algebraic Mode to Plot Input Referred IP3

In this example, you use the *ipn* special function to plot IP3. The *ipn* special function calculates only the *Input Referred IP3*.

1. From the Simulation window, choose *Tools—Calculator.*



2. In this example, you use the Calculator in the Algebraic mode. If the Calculator is in the RPN mode, change it by selecting *Options—Set Algebraic*.
3. In the Calculator, select *Special Functions—ipn*.

   The IPN form displays.
4. In the Calculator, click *browser* to open the Results Browser.

5. When the Browse Project Hierarchy form appears, click OK to display the results browser.

6. In the Results Browser, click on:

```
schematic->psf->Run1->sweeppss_pac-sweep->
sweepVariable->prf-> -10->sweeppss-003_pac-pac->
sweepVariable->harmonic-> -21->
sweeppss-003_pac--21.pac-> Pif
```

Or

The information displays in the Calculator buffer. It looks similar to this:

```
v( "/Pif" ?result "sweeppss_pac-sweep" ?resultsDir
"/hm/uma/simulation/ne600p/spectre/schematic" )
```

Note: Clicking on any harmonic (`-0`, `-21`, or `-25`) in the results browser gives the same results.

The Results Browser shows the basic structure of the saved simulation. data. Since you choose (in "Selecting Outputs to Save" on page 191) to save results for the *rif psin* only, the results browser displays only *Pif* data (voltage and current information).

7. In the IPN form, do the following:

   a. Click *Get Buffer*. The contents of the Calculator buffer appear.

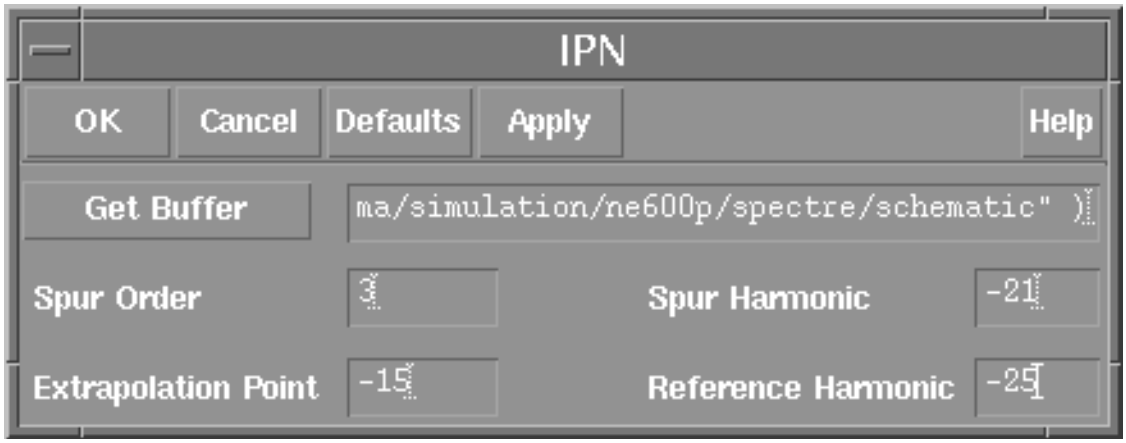
   b. Enter 3 in the *Spur Order* field.

   c. Enter –21 in the *Spur Harmonic* field.

   d. Enter –15 in the *Extrapolation Point* field.

   e. Enter –25 in the *Reference Harmonic* field.

   f. Click *Apply* and then click *OK*. .



8. Click the = button in Calculator to evaluate the expression.

Using the ipnVRI Special Function in Algebraic Mode to Plot IP3

In this example, you use *ipnVRI* on the *specialFunctions* cyclic menu to obtain both Input Referred IP3 and Output Referred IP3 values.

1. In the Calculator, select *Special Functions—ipnVRI*.

   The IPN form displays.

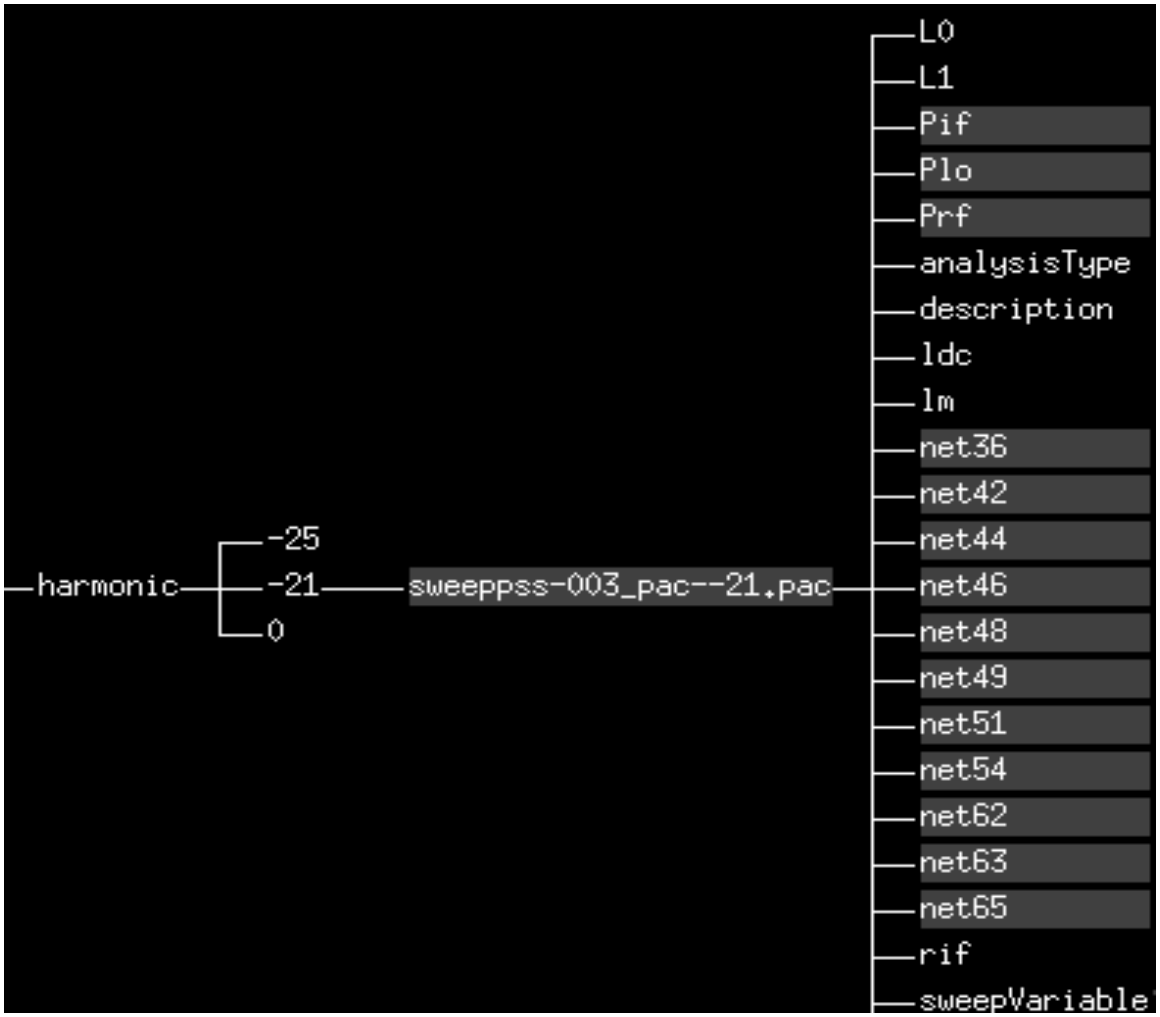
2. In the Calculator, click *browser* to open the Results Browser.

3. When the Browse Project Hierarchy form appears, click OK to display the results browser.

4. In the Results Browser, click on:

```
schematic–>psf–>Run1–>sweeppss_pac-sweep–>
sweepVariable–>prf–> –10–>sweeppss-003_pac-pac–>
sweepVariable–>harmonic–> –21–>
sweeppss-003_pac--21.pac–> Pif
```

Or



The information displays in the Calculator buffer. It looks similar to this:

```
v( "/Pif" ?result "sweeppss_pac-sweep" ?resultsDir
"/hm/uma/simulation/ne600p/spectre/schematic" )
```

Note: Clicking on any harmonic (`–0`, `–21`, or `–25`) in the results browser gives the same results.

The Results Browser shows the basic structure of the saved simulation. data. Since you choose (in "Selecting Outputs to Save" on page 191) to save results for the *rif psin* only, the results browser displays only *`Pif`* data (voltage and current information).

5. In the IPN form, do the following:

   a. Click *Get Buffer*. The contents of the Calculator buffer appear.

   b. Enter –21 in the *Spur Harmonic* field.

   c. Enter –25 in the *Reference Harmonic* field.

   d. Enter 3 in the *Spur Order* field.

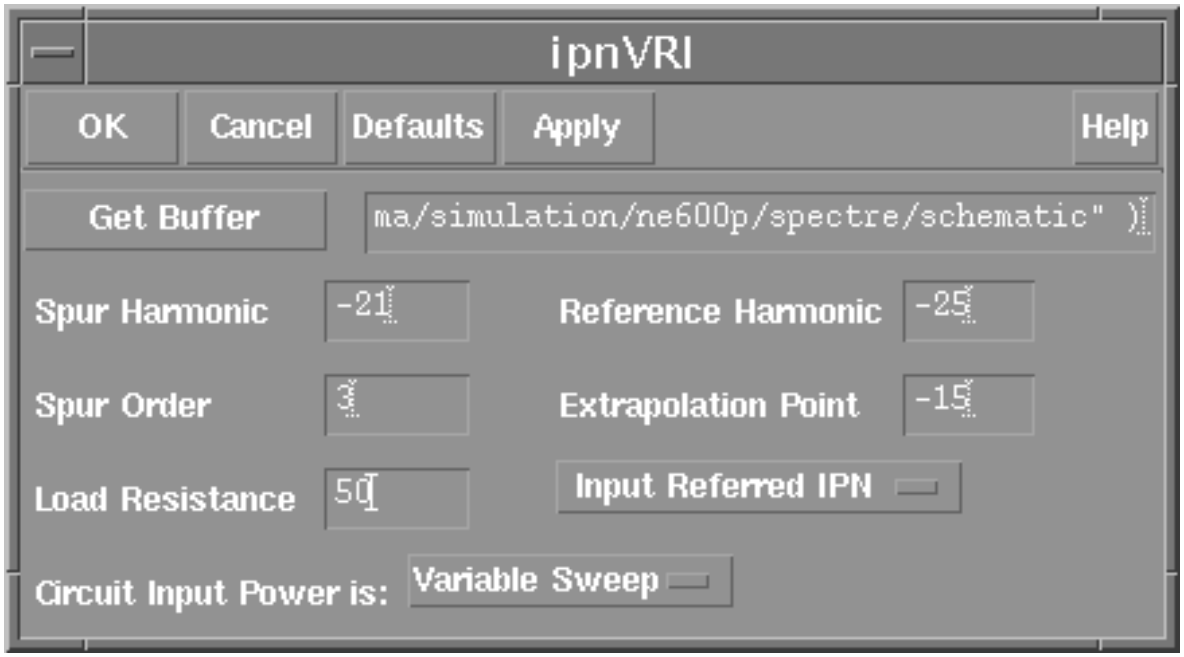
   e. Enter –15 in the *Extrapolation Point* field.

   f. Enter 50 in the *Load Resistance* field.

   g. Select *Input Referred IPN* in the cyclic field.

   h. Select *Variable Sweep* in the *Circuit Input Power is:* cyclic field.

   i. Click *Apply* and then click *OK*.



Note: If you want to plot dBV using the *ipnVRI* function, set the *Load Resistance* to `nil`.

6. Click the = button in Calculator to evaluate the expression.

Using the IPN Special Function with Scalar Inputs

This example uses the editable copy of the *ne600p* schematic with scalar, rather than swept, inputs. Two reasons that you might want to use scalar inputs with the IPN function (assuming that the region of operation is linear) are that

- Simulation is faster
- IPN optimization is faster

Note: Whenever you use the *ipn* function with scalar inputs, always follow up by doing a final *ipn* verification with a power sweep (non-scalar inputs), as described in "Using the ipn and ipnVRI Calculator Special Functions" on page 177. This will verify your results.

Setting Up the Simulation Environment

This example uses the editable copy of the *ne600p* schematic from the *rfExamples* library.

If necessary, start *icms* and edit the *ne600p* schematic as described in the previous example.

- See "Start the CIW, Open the Schematic, and Set Up the Simulator and Models" on page 178.
- Edit the *frf* design value as described in the previous example. See "Edit the frf Design Variable" on page 181.
- Edit the *ne600p* schematic. See "Editing the Schematic" on page 182.
- Finally, see "Check and Save the Design" on page 185.

Setting Up the Analyses

Before you set up the analyses, use *Analyses—Disable* in the Simulation window to disable any analyses you may have run previously.

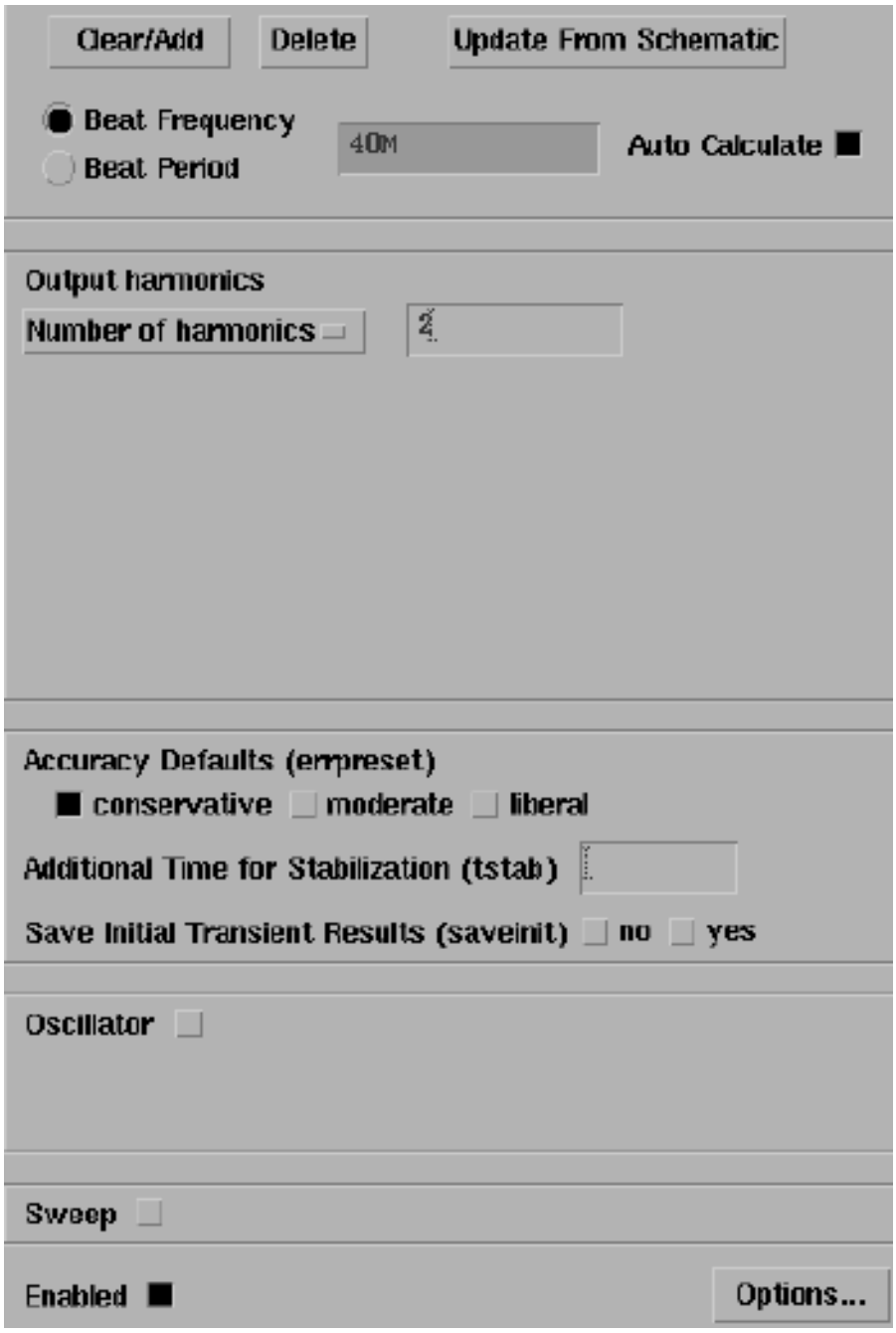
➤ In the Simulation window, use *Analyses—Choose* to display the Choosing Analyses form.

Setting Up the PSS Analysis

The PSS analysis is set up similar to the previous example, except that the *Sweep* button is disabled.

1. In the Choosing Analyses form, click *pss* to display fields for the Periodic Steady State analysis.
2. Below the *Fundamental Tones* list box, if necessary, click *Beat Frequency* and then click *Auto Calculate*.
3. In the *Output harmonics* cyclic field, select *Number of harmonics* and enter 2 in the *Number of harmonics* field.
4. Click on *conservative* for the *Accuracy Defaults (errpreset)* setting.
5. Make sure *Sweep* is not selected.
6. Click *Enabled*.

The bottom of the PSS analysis form looks like this



7. Click *Apply*.

8. At the top of the Choosing Analyses form, click *pac*.

9. Set the *Frequency Sweep Range (Hz)* cyclic field to *Single-Point*. The value `921M` displays in the *freq* field.

10. In the *Sidebands* cyclic field, select *Array of indices* and type `–21` and `–25` separated by a space in the *Additional indices* field.

    Given the following

    - A Fundamental Tone at `40` MHz
    - The LO at `1`GHz
    - Two RF tones at `920` MHz and `921` MHz

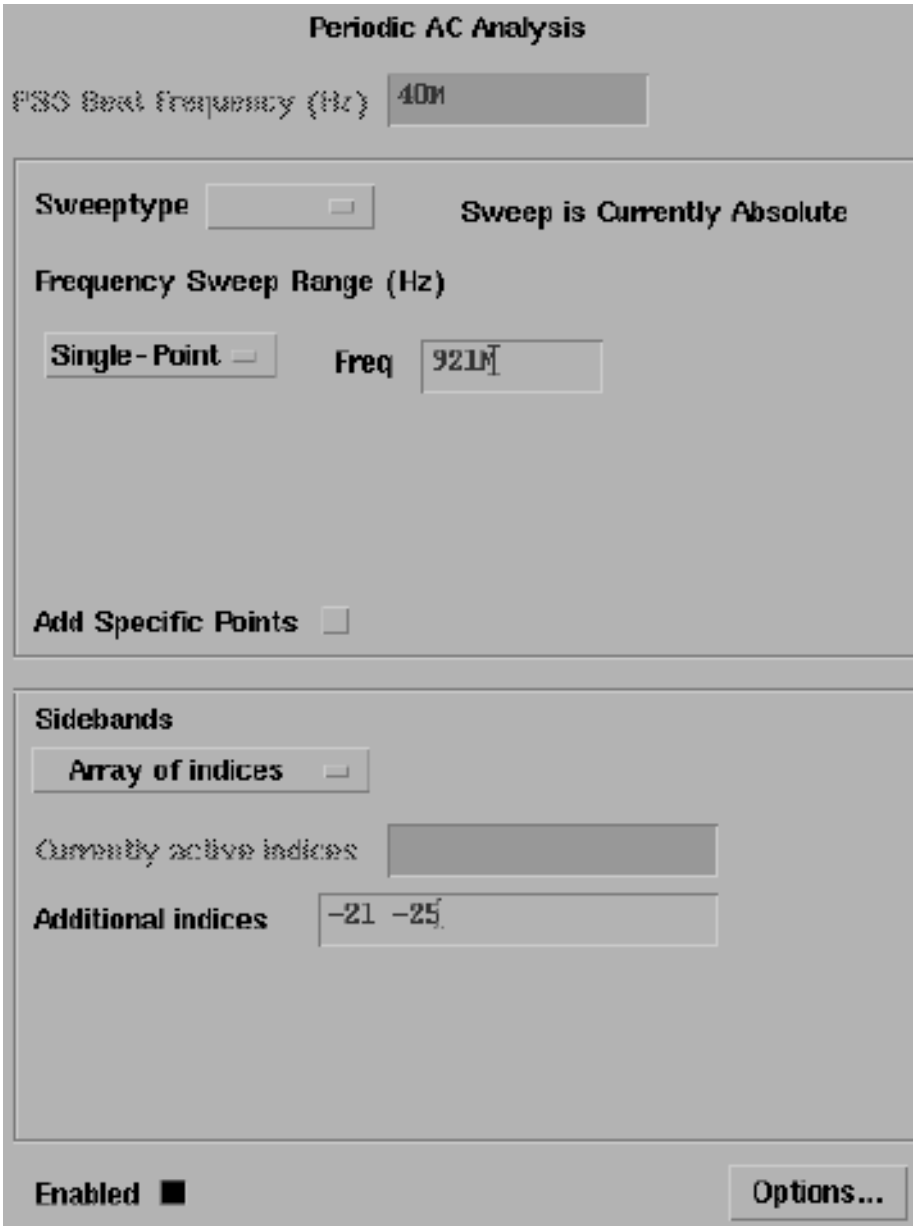
    The sidebands of `–25` and `–21` represent respectively

    - The first-order harmonic of the IF output at `79` MHz ($921 - 25 * 40 = 79$)
    - The third-order harmonic at `81` MHz ($921 - 21 * 40 = 81$)

11. Click *Enabled*.

12. Click *Apply*.

13. The completed PAC section of the Choosing Analyses form looks like this.



14. Click *OK* in the Choosing Analyses form.

Edit the prf Design Variable

In the Simulation window, choose *Variables—Edit* and change the value of the *prf* design variable to –15.

This value (–15) is the extrapolation value used for the IPN plot in the previous example. See "Edit the frf Design Variable" on page 181 for information on using the Edit Variables form.

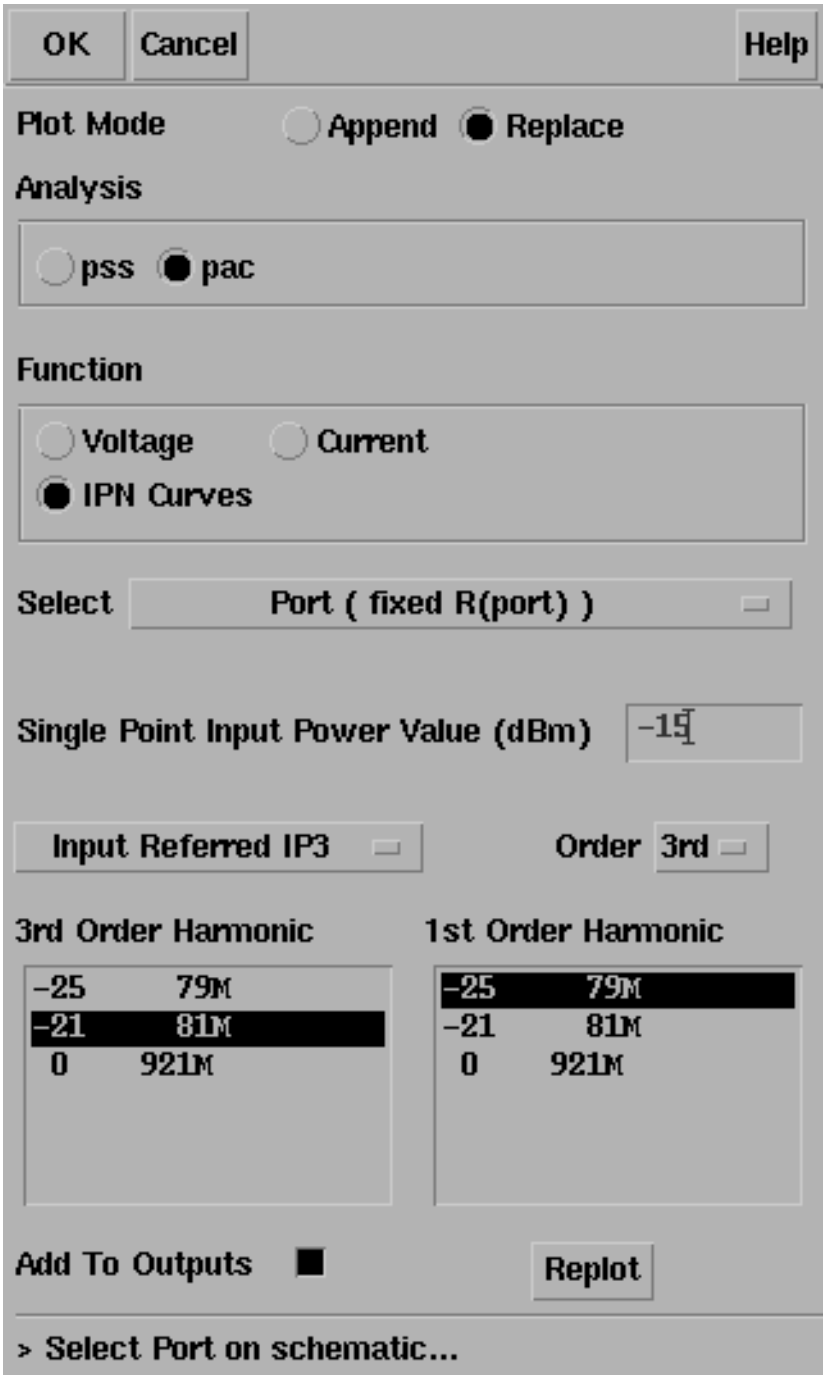
Running the Simulation

➤ Select *Simulation—Netlist and Run*.

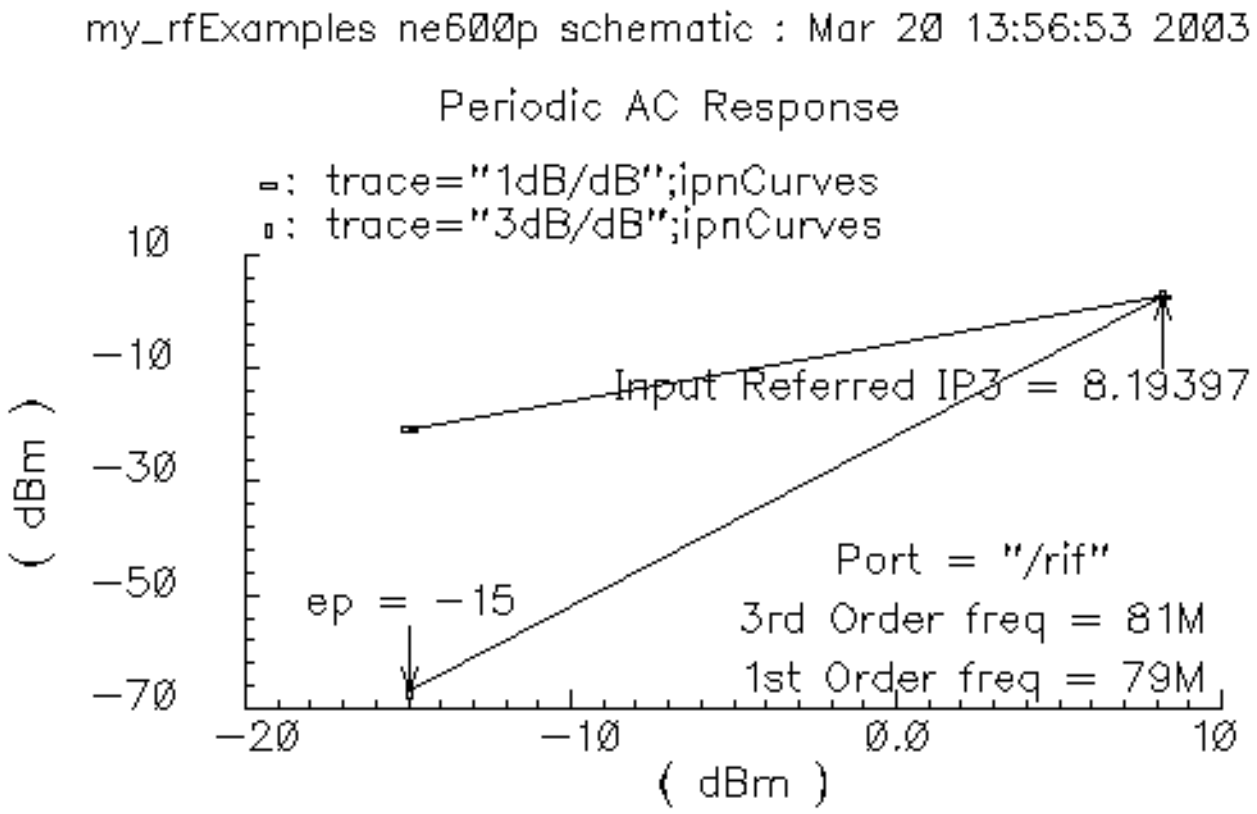
Plotting the Single Point IP3 Using the Direct Plot Form

1. After simulation, select *Results—Direct Plot—Main Form*. Enter the values in the PSS Results form as follows:
   - **a.** Click on *Replace* for *Plot Mode*.
   - **b.** Click on *pac* for *Analysis Type*.
   - **c.** Click on *IPN Curves* for *Function*.
   - **d.** In the *Select* cyclic field, select *Port (fixed R(port))*.
   - **e.** Type -15 for *Single Point Input Power Value.*
   - **f.** Select *Input Referred IP3* and *Order 3rd*.
   - **g.** Highlight *-21 81M* in the *3rd Order Harmonics* list box.
   - **h.** Highlight *-25 79M* in the *1st Order Harmonic* list box.
   - **i.** Click *Add To Outputs*.
   - **j.** Following the prompt at the bottom of the Direct Plot form, in the Schematic window, click on the *rif* port

The Direct Plot form looks like this.



The single point IP3 is plotted in the waveform window.

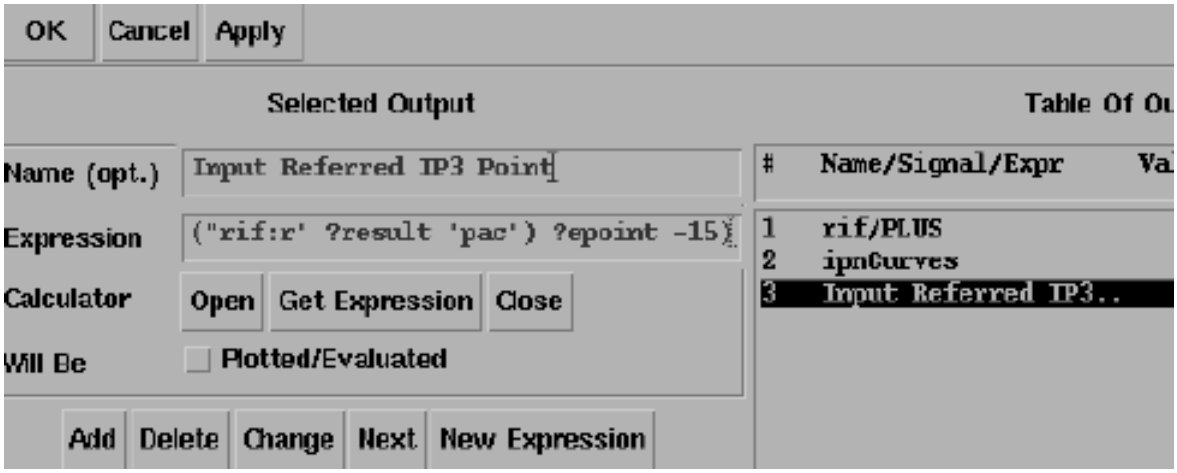


Because you selected the *Add to Outputs* button in the PSS Results form, the `Input Referred IP3 Point` direct plot expression is also saved to the *Outputs* pane in the Simulation Window

Outputs

#	Name/Signal/Expr	Value	Plot	Save	March
1	rif/PLUS		no	yes	no
2	ipnCurves		no		
3	Input Referred IP3..		no		

2. If you double click on the *Input Referred IP3...* entry in the Simulation window, the Setting Outputs form appears with *Input Referred IP3...* selected. The Table of Outputs on the right side of the form is reproduced from the Simulation window.



3. Click on *Get Expression* to enter the IP3 into the Calculator buffer or CIW window and obtain the value 8.21304.

```
ipnVRI((v("/Pif" ?result "pac") - 0.0) '-21 '-25 ?rport
resultParam("rif:r" ?result "pac") ?epoint -15)
```

   Note: You can copy the expressions stored in the *Outputs* section of the Simulation Window into the *Goals* section of the Optimizer. Use Tools—*Optimizer* in the Simulation window to open the Optimizer. In the Optimizer, choose *Goals—Retrieve Outputs* menu function.

4. The procedure for using the Calculator *ipn* and *ipnVRI specialFunctions* are essentially the same as described in the previous example.

Using the spectralPower Special Function

This example uses an editable copy of the *ne600p* schematic copied from the *rfExamples* library. As you follow the example, you

- Perform a swept PSS analysis to produce data for several spectral power plots.
- Plot spectral power using the Direct Plot form
- Plot spectral power using the Calculator *spectralPower* special function
- Compare the results

Setting Up the Simulation Environment

If you need more information than is provided in this appendix, see the *SpectreRF User Guide* for information. For example, Chapter 3 in the *SpectreRF User Guide* describes in detail how to create the editable copy of the *rfExamples* library of schematics.

This example uses the editable copy of the *ne600p* schematic from the *rfExamples* library.

If necessary, start *icms* and edit the *ne600p* schematic as described in the first example in this chapter.

- See "Start the CIW, Open the Schematic, and Set Up the Simulator and Models" on page 178.
- Edit the *frf* design value as described in the first example. See "Edit the frf Design Variable" on page 181.
- Edit the *ne600p* schematic. See "Editing the Schematic" on page 182.
- Finally, see "Check and Save the Design" on page 185.

Setting Up the PSS Analysis

Before you set up the PSS analyses, use *Analyses—Disable* in the Simulation window to disable any analyses you may have run previously.

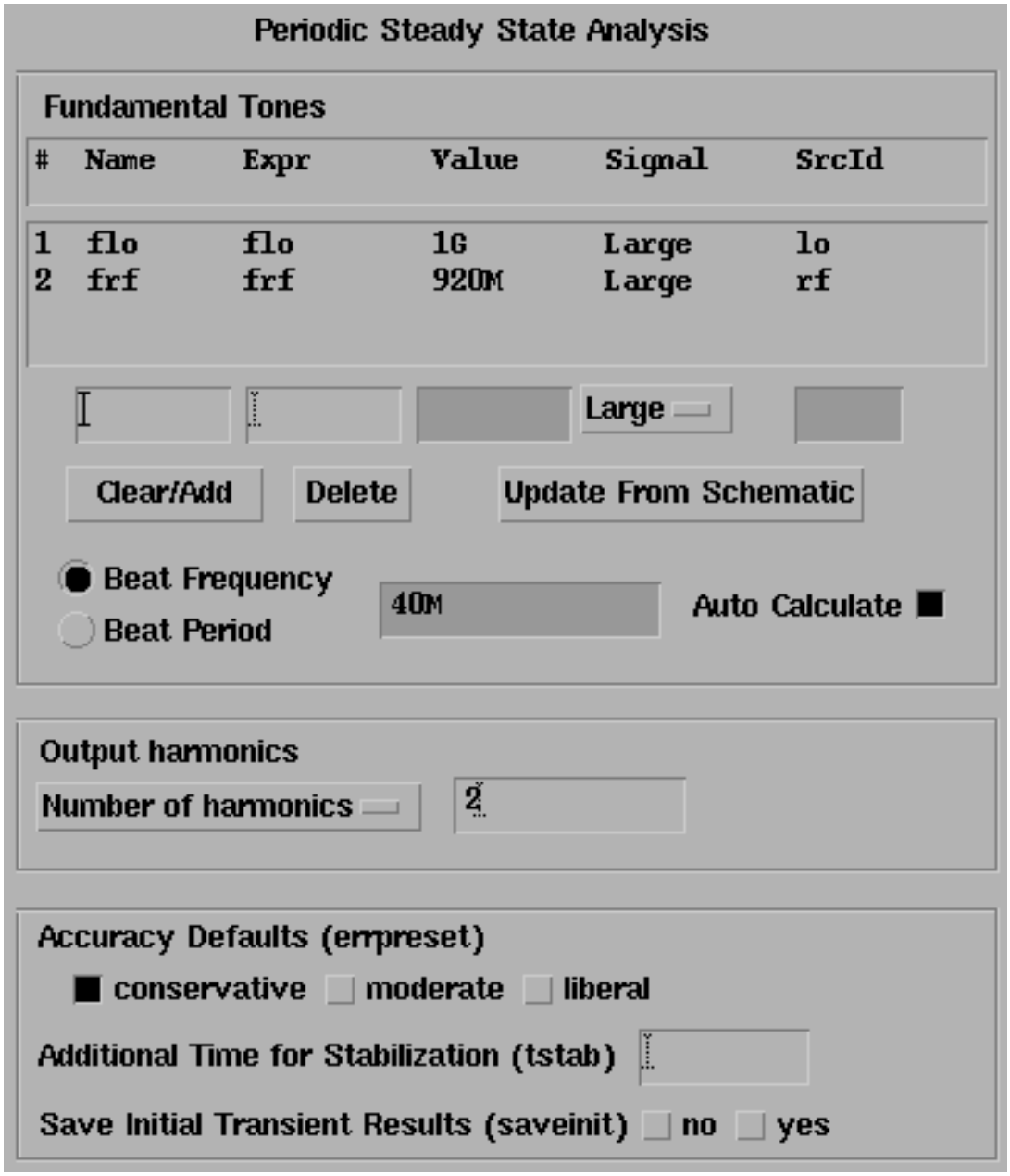
1. In the Simulation window, choose *Analyses—Choose* to display the Choosing Analyses form.
2. In the Choosing Analyses form, select pss to display fields for the Periodic Steady State analysis.

3. Below the *Fundamental Tones* list box, with *Beat Frequency* selected, click the *Auto Calculate* button.

   The value `40M` is specified as the *Beat Frequency*. Consequently, the fundamental frequency is set to `40 MHz`.

4. In the Output harmonics cyclic field, select *Number of harmonics* and enter 2 in the field.

5. Click on `conservative` for the *Accuracy Defaults (errpreset)* setting. The top of the PSS Choosing Analysis form looks as follows.



6. Click on the *Sweep* button.

The form changes to let you specify data for the variable sweep.

a. In the *Sweep* cyclic field, select `Variable`.

b. For *Frequency Variable?*, click *no*.

c. Click on the *Select Design Variable* button.

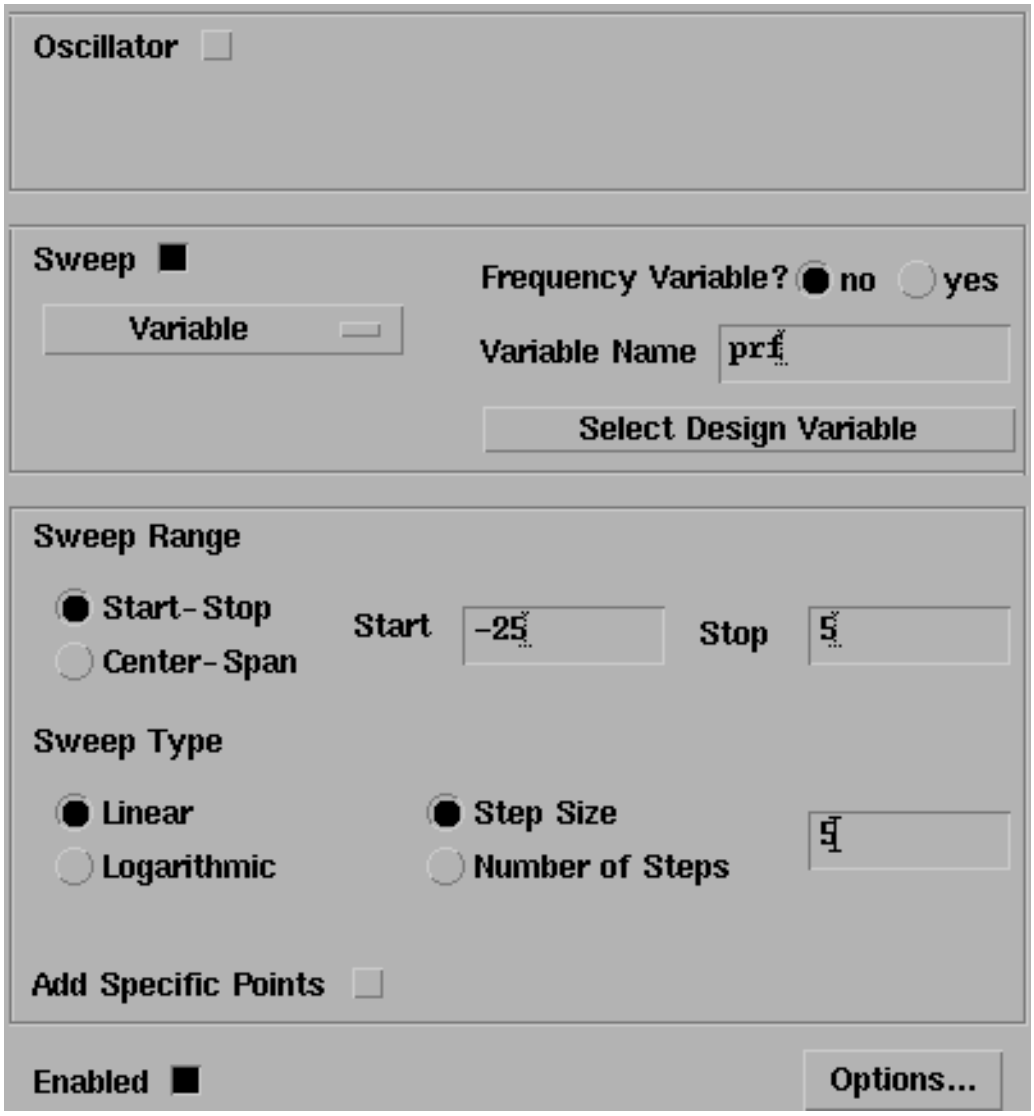
d. In the Select Design Variable form, click `prf` and click *OK*.

The variable name `prf` displays in the *Variable Name* field on the Choosing Analysis form.

7. Click *Start-Stop* for the *Sweep Range* and then enter `-25` and `5` for the *Start* and *Stop* values, respectively. The sweep is skewed downward for a down converter.

8. Click *Linear* for the *Sweep Type*, then click *Step Size* and enter `5` in the *Step Size* field.

9. If necessary, click *Enabled*. The bottom of the PSS Choosing Analysis form looks as follows.



10. Click *Apply*. Then click *OK* in the Choosing Analysis form.

In the Simulation window, the *Analysis* area displays the PSS analysis information you just entered.

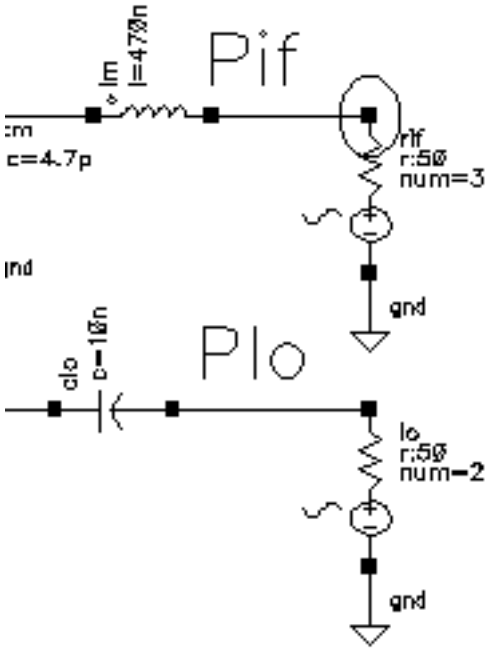
Selecting Outputs to Save

1. In the Simulation window, choose *Outputs—To Be Saved—Select on Schematic*.

The Schematic window pops to the top.

2. In the Schematic window, click on the positive node of the *rif* psin.

   The selected node is circled in the schematic. It is also listed in the *Outputs* section of the Simulation window.



Running the Simulation

In the Simulation window, choose *Simulation—Netlist and Run*.

Plotting the Power Spectrum Using the Direct Plot Form

1. Select *Results—Direct Plot—Main Form*.

   In the Direct Plot form, do the following:

   a. Click on *Replace* for *Plot Mode*.

   b. Click on *pss* for *Analysis Type*.

c. Click on *Power* for *Function*.

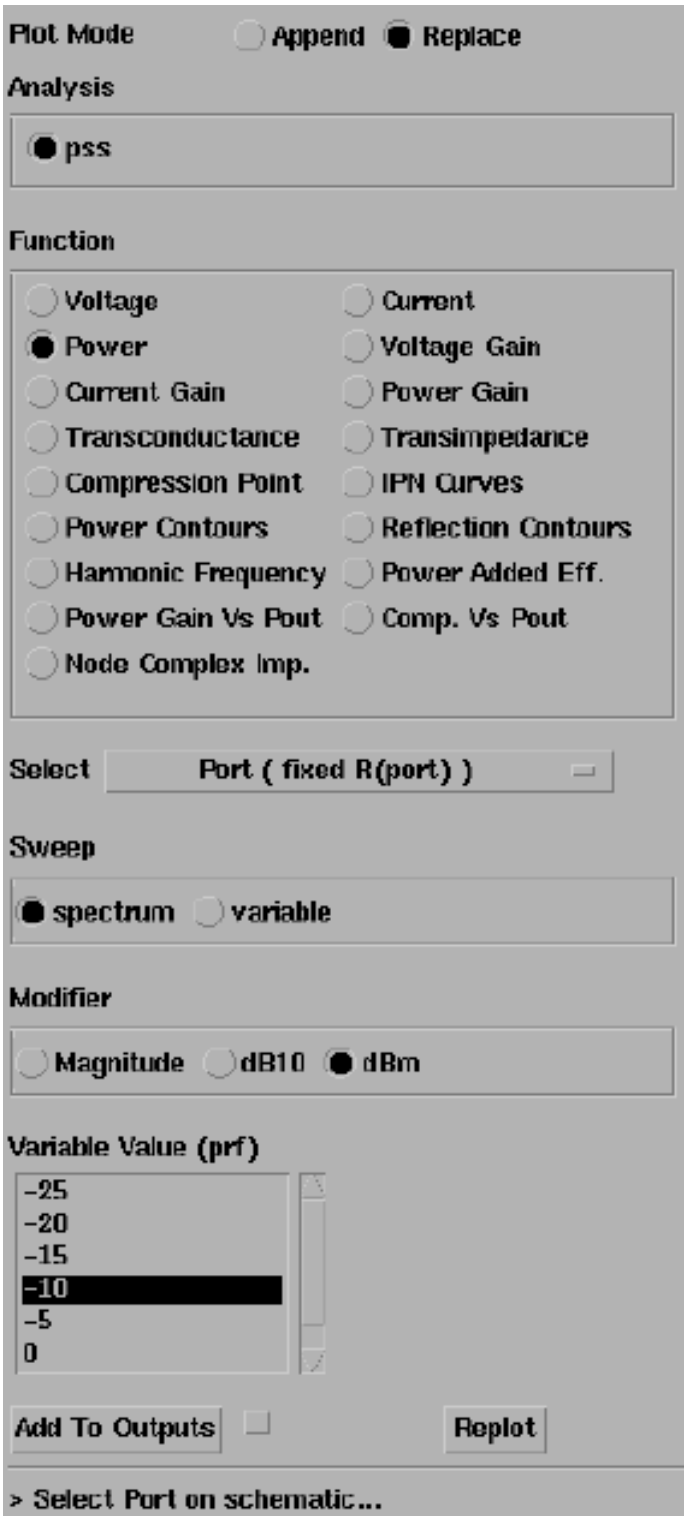
d. In the *Select* cyclic field, select *Port (fixed R(port))*.

e. Click *spectrum* for *Sweep*.

f. Click dBm for modifier

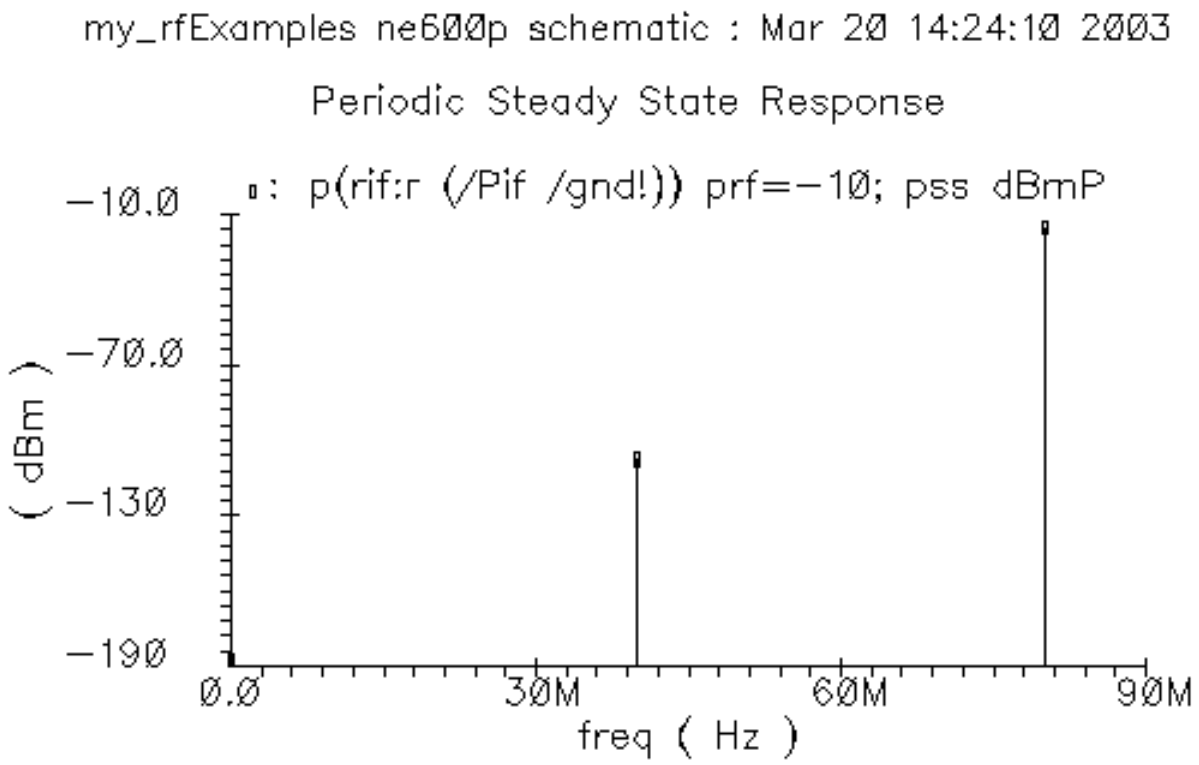
g. Highlight *-10* in the *Variable Value (prf)* list box.

The completed form looks like the one below.



2. Following the prompt at the bottom of the Direct Plot form, in the Schematic window, click on the *rif* port.

   The Waveform window display appears as shown below

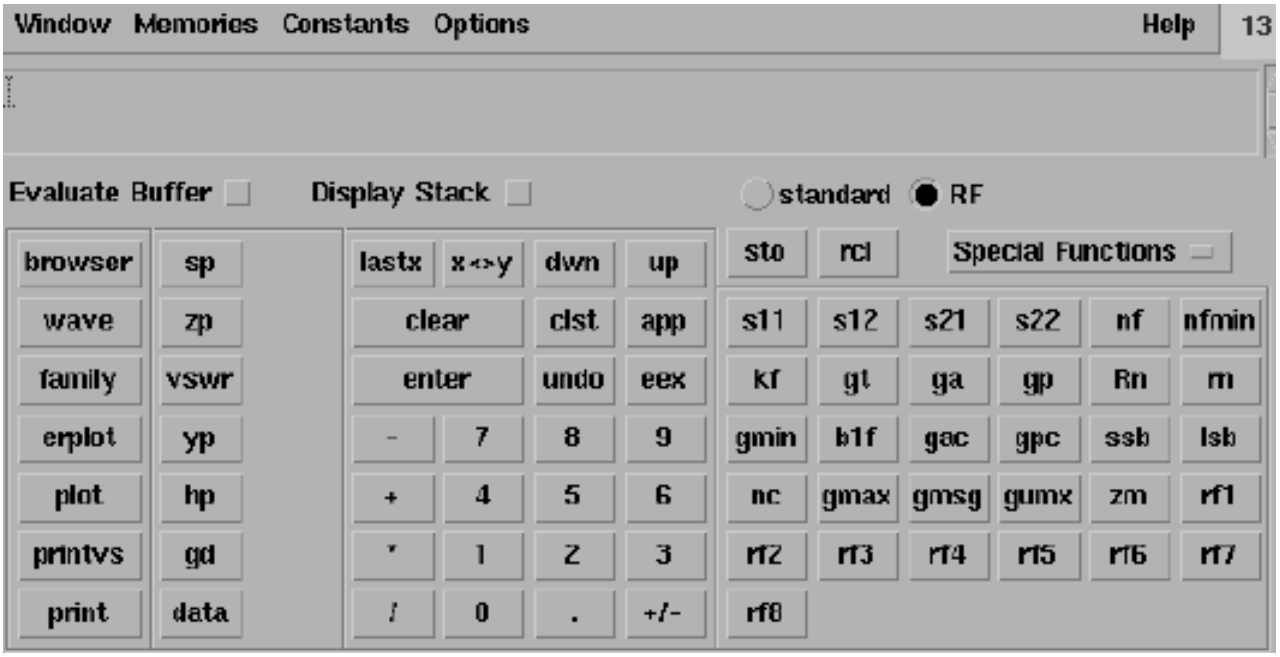


Plotting the Power Spectrum Using the Calculator spectralPower Special Function

The next example uses the *spectralPower* special function in the Calculator to create spectral power plots.

1. From the Simulation window, choose *Tools—Calculator.*

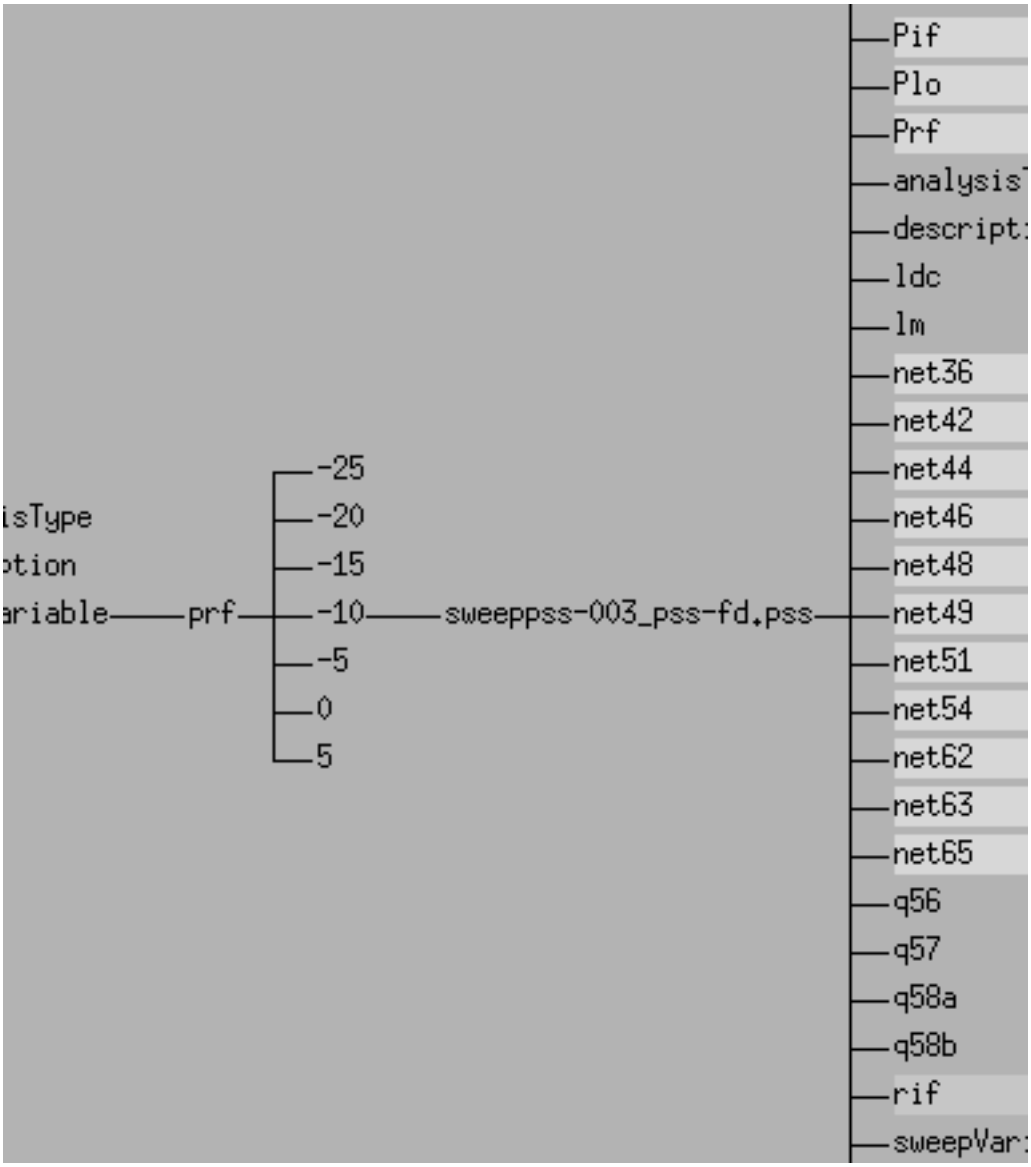
The Waveform Calculator opens.



2. In this example, you use the Calculator in RPN mode. If the Calculator is in Algebraic mode, change it by selecting *Options—Set RPN*.
3. In the Calculator, click *browser* to open the Results Browser.

   When the Browse Project Hierarchy form appears, click *OK* to display the results browser.

The Results Browser shows the basic structure of the saved simulation. data. Since you choose (in “Selecting Outputs to Save” on page 219) to save all results, the results browser displays all saved data (both voltage and current information).



4. When you use the spectralPower function, you must first enter (or select) the spectral voltage waveform and then enter (or select) the spectral current waveform

   a. To select the spectral voltage waveform, in the Results Browser, click on:

   ```
   schematic–>psf–>Run1–>sweeppss_pss_fd-sweep–>
   sweepVariable–>prf–> –10–>sweeppss-003_pss-fd.pss–> Pif
   ```

   The information displays in the Calculator buffer. It looks similar to this:

```
v( "/Pif" ?result "sweeppss_pss_fd-sweep" ?resultsDir
"/ home/belinda/simulation/ne600p/spectre/schematic" )
```

b. To select the spectral current waveform, in the Results Browser, click on:

```
schematic->psf->Run1->sweeppss_pss-fd-sweep->
sweepVariable->prf-> -10->sweeppss-003_pss-fd.pss-> rif
```

The information displays in the Calculator buffer. It looks similar to this:

```
i( "/rif/PLUS" ?result "sweeppss_pss_fd-sweep" ?resultsDir
"/home/belinda/simulation/ne600p/spectre/schematic" )
```

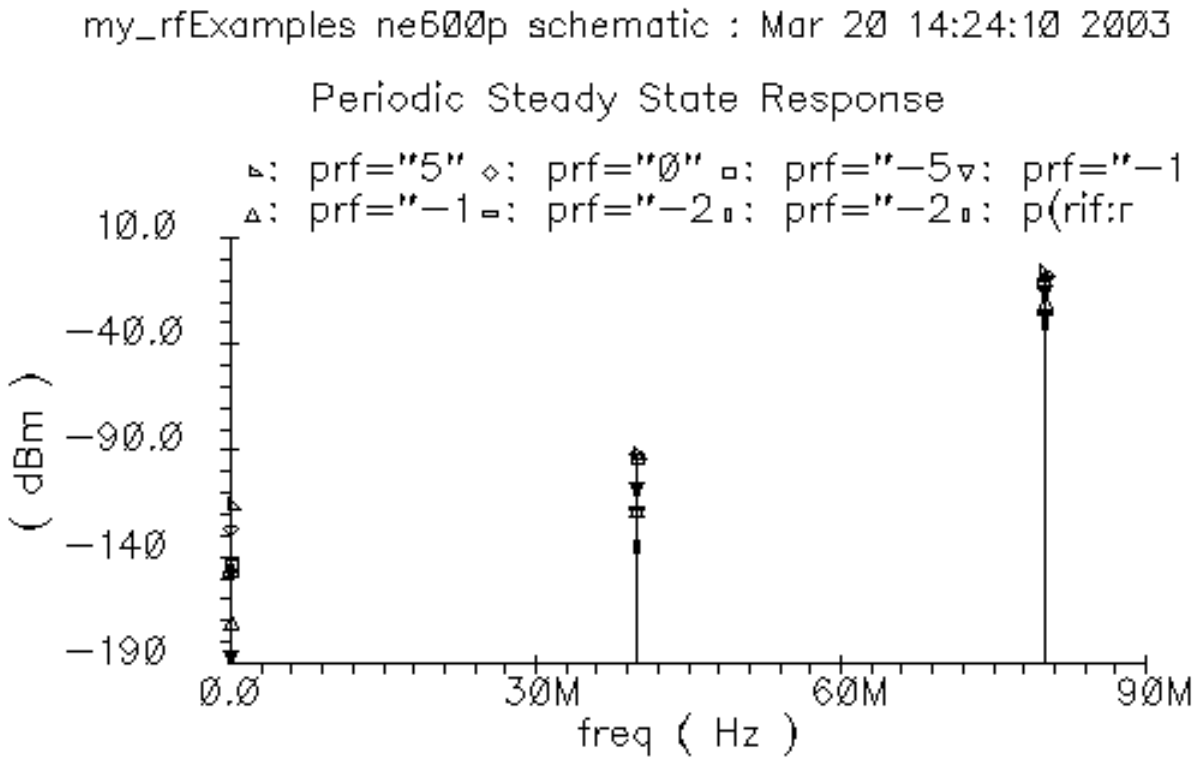
The voltage waveform previously displayed in the Calculator buffer is passed to the Calculator stack.

5. In the Calculator, select *Special Functions—spectralPower*.

6. In the Calculator, select *Special Functions—dBm*.

7. In the Calculator, click *plot*.

The power spectrum displays in the Waveform window.



Index

Symbols

Numerics

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

V

W

X

Y

Z