CoolSPICE, v.4.0

User Manual

July 18, 2018

TABLE OF CONTENTS

1. Introduction and Getting Started

1.1. Introduction and Features

CoolSPICE, from CoolCAD Electronics, LLC, is a full-featured simulation suite. Its SPICE engine allows for mixed-level/mixed-signal circuit simulation. CoolSPICE is a complete suite including a graphical Schematics Editor, a text-based Netlist Editor and a Plotter for the results. Example circuits to demonstrate the devices and simulation features are also included in the package.

Features of CoolSPICE include:

- Standard SPICE models and analysis tools
- Cryogenic CMOS simulation
- Simulation of radiation effects on CMOS operation
- High-power, high-temperature SiC device model libraries and simulation
- Ability to set individual temperatures for devices in a design during simulation
- Self-consistent electrical/thermal simulations
- Niche device model libraries (e.g. rectennas, avalanche photodiodes...) and simulation
- TCAD simulations for a variety of structures

The Student Version only includes the standard SPICE models and analysis tools.

1.2. Installation and Setup

1.2.1. System Requirements

- Windows XP or later,
- Windows SDK 7.1 (included with the installer),
- .NET Framework 4.5 or later (included with the installer),
- Visual C++ 2010 SP1 x64, or (included with the installer),

- Visual C++ 2010 SP1 x86 (included with the installer),
- WinSCP (optional, to directly run netlist files).

1.2.2. Installation

The free student version of CoolSPICE is available to download from CoolCAD's website. Simply go to

http://coolcadelectronics.com/coolspice/

and enter your name and e-mail address. You can opt in or out of joining the mailing list. Click the Download CoolSPICE button for the download. This sends you to the download page, where you can choose the version you wish. When run, the installer creates shortcuts to the CoolSPICE Package on the desktop and in the Start menu.

Reinstalling Caution!

If you are reinstalling CoolSPICE or installing a new version, be aware that in the CoolSPICE default directories, all the files created by a previous use of the CoolSPICE installer will be overwritten. For instance, if you have modified the circuits found in the Ckts/ directory, they will be overwritten by the default versions in the installer. It is a good idea to back up your circuits and netlists in another location before starting a new CoolSPICE installation.

To obtain the full commercial version, please contact CoolCAD Electronics at

contact@coolcadelectronics.com
or +1-301-405-3363.

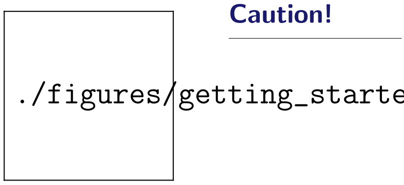


Figure 1.1: The desktop shortcut.

Folder locations

By default, CoolSPICE is installed in the folder CoolSpice/ within the My Documents/ folder of the Windows user account used for the installation. Within this folder, four subfolders will be created:

- Ckts/ This folder includes a collection of example circuits in schematic file format. These circuits can be saved under other names and edited to explore the capabilities of CoolSPICE.
- CoolCADSPICE/ This folder contains the executable files for the SPICE engine, the Schematics Editor, the Text Editor and the Plotter.
- Libraries/ This folder contains the schematic libraries used by the Schematics Editor.
- Models/ This folder contains SPICE models published by MOSIS (MOSIS.txt) and SPICE models for some other standard components (StandardParts.txt).

Within the CoolCADSPICE/ folder, the subfolder WinSCP/ is created. The open-source software **WinSCP** can optionally be installed here if the user wants the capability to run netlist files directly from the main console. **WinSCP** can be downloaded from http://coolcadelectronics.com/wp-content/uploads/2013/04/WinSCP.zip

and the extracted files from the zip archive should be placed in Documents/CoolSpice/CoolCADSPICE/WinSCP.

1.3. Quick Start

The CoolSPICE package is launched by double-clicking on the desktop shortcut or from the Start menu as CoolSpice Package under the menu item CoolSpice Package under "All Programs." The main console, as displayed in Figure **??**, comes up.

./figures/getting_started_figures/CoolSPICE_mainconsole.png

Figure 1.2: The main console of the CoolSPICE package.

1.3.1. Building a Circuit

On the main console, click on **Schematic Editor** to launch the Schematic Editor. By default, the Schematic Editor should open with frames which show the libraries and the selected symbol (when one is selected) to the left. If these frames are not visible, click the **Symbol Picker** button, which has the symbol $\boxed{\Omega}$ (or press **F6**).

For a fast-start example, we are going to build an inverter using the ON Semiconductor 0.5 μm technology and run a simulation to find its voltage and current transfer functions. The MOSFET models are included in the built-in libraries of CoolSPICE. Click on the "+" symbol next to the MOSIS library to open up the list of components in this library.

Inserting components

Double-clicking on the name of the required component (or single-clicking on the symbol which appears in the symbol frame) gives the user an instance of the component to place. It also opens up the Options window for the component. The user can place

./figures/getting_started_figures/CoolSPICE_emptySchematicEditor.png

Figure 1.3: Schematic Editor with a model selected and ready.

Use the Esc key to cancel out of an action.

multiple instances of the same component until the place mode is cancelled by hitting the **Escape** key. The Options window disappears after the place mode is cancelled (or if the user clicks on an empty spot in the design area) and can be recalled by clicking on a component after placement.

Component parameters

The component parameters can be changed by clicking on the value of the parameter in the Options window. As shown in Fig. **??**, here the width for the PMOS has been changed to 4.5 μm (by typing **4.5u**). The placement orientation and the instance reference name (**"Ref"**) for a component can also be changed from the Options window.

Note that if **"Ref"** is set to end with a number (e.g. "D42"), when more instances of the same component are placed in the schematic by copy-paste, this reference number can automatically be incremented. *However, if the user edits the reference in the* Options *box in between placements and places further instances, the reference number will* not *be incremented* until the user cancels out of the action and restarts by copy-pasting or selecting the left-frame representation of the symbol to create a new instance. This can lead to netlist conflicts.

Power and ground nets

We need to add power and ground nets and voltage sources. To add the pins for the power net, we use the **Power** button (marked with the earth ground symbol) or press **F4**. The net name is set in the dialogue box; symbol rotation and type are set by the radio buttons. Figure **??** shows the VDD pin being added. For the ground net, there are two options: If the **Power** button option is used, the pin *must* be named "**0**", which is in fact the default name; the other option is to use the "**Source - Ground**" component in the **Analog** library, as in this example circuit (see Fig. **??**).

Voltage sources

The voltage sources are also available under the Analog library. For this example we use DC voltage sources, labelled "SourceV - DC AC" in the program. Figure **??**

./figures/getting_started_figures/CoolSPICE_SchematicEditorwithDevicesandOptions.

Figure 1.4: Schematic Editor with two components placed and the Options window.

Make sure all instances have distinct reference names. Avoid reference names with only numbers (e.g. “42”).

./figures/getting_started_figures/SchematicEditor_ChangePMOSWidth.png

Figure 1.5: Changing component parameters using the Options window.

shows the placement, naming and parameters of these DC sources.

Adding wires

Wires are added using the **Wire** button (see Fig. **??**) or the F2 hotkey. Click to start the wire and once in the path to turn a corner. (The first corner-turn in a given path is also detected from the mouse movement.) The wire angle can be set in the Wire Options dialogue. Double-clicking after a straight extension or right-clicking at any time ends the piece. When the wire crosses another wire, a circle appears to present

./figures/getting_started_figures/SchematicEditor_addingVDD.png

Figure 1.6: Adding the VDD net pin.

the option of forming a connection with a single click. A connection also ends the wire segment. Figure **??** shows the wired circuit just before the last connection is placed. (Note that the mouse scroll button or the PgUp/PgDown keys zoom in/out. This is a zoomed-in figure. The user can also pan around the design by right click-hold-dragging the mouse.)

./figures/getting_started_figures/SchematicEditor_addingwires.png

Figure 1.7: Wired circuit.

1.3.2. Analyses, Simulation and Viewing Results

In order to measure and plot voltage signals, nets can be labelled with the Label

Labelling nets

Do not label the net connected to the ground node or a power pin (e.g. VDD). Do not rely on capitalization to distinguish nets.

tool (the button marked a, or the F1 hotkey). For current signals, a current meter should be used. The current meter is in the Analog library as “Probe - Ammeter”. Figure **??** shows these elements in place and the input and output nets being labelled with “Vin” and “Vout”. Note that the net names are *not case-sensitive*. In SPICE, a net named VGS and one named vgs are the same in the netlist and if they are electrically different nets in the circuit, conflicts and errors will arise.

Defining analyses

The SPICE analysis tools are in the **“Analysis”** library. For this example we use the **.dc** analysis. Figure **??** shows the DC analysis setup in place. Note especially how the voltage source to be swept is specified as “Voltage Source1”. In this case, the common gate of the MOSFETs is being swept from 0 to 3V (the rail voltage) in 0.05V increments.

./figures/getting_started_figures/SchematicEditor_labeling_and_Isense.png

Figure 1.8: Inverter with input/output net labels and a current meter at the PMOS source.

Saving and running

To run the analysis we first need to save the design. The **Save** button (floppy disk symbol) or the **File**→**Save** menu item can be used. To start the simulation, use the **Run** button (the “play” symbol) or the **SPICE**→**Create Spice Net List and Run** menu item. This will create the netlist and ask for the netlist filename. Here we use the same name as the design, **inverter_test1**. Just to see the netlist without running the simulation, use **SPICE**→**View SPICE Netlist** menu item. (The netlist is not editable from this view.)

Under the **SPICE**→**Preferences** menu item, CoolSPICE can be set to automatically launch the Plotter and/or show the simulation Log File and raw data file after the simulation is complete. By default, the ”Launch Plotter” option is enabled and the options to show log/raw data file are disabled.

Simulation failures due to path error

A common cause for simulation failure is the path specification errors for the mod-

./figures/getting_started_figures/SchematicEditor_DCrunadded.png

Figure 1.9: DC sweep specification.

💡 **The model paths are relative! Watch where the netlister is searching for the model cards.**

els. CoolSPICE comes with built-in example circuits under the folder CoolSpice/Ckts, symbol libraries under Libraries/ and model cards under Models/. During netlist creation, the netlister looks for the models in the ../Models/ folder (*i.e.* it expects the Models/ folder to be one level up from the circuit itself) unless the models are edited to point at an alternative location. If the path is not where the program searches, the simulation will fail, quietly if the user has not changed the SPICE preferences to show the logfile automatically. (If the Plotter has been set to launch automatically, a window will warn that no raw data file was found.) As shown in Fig. **??**, the log file (which is stored in the same folder as the circuit) displays where the netlister looked for the model file. To avoid this problem, save the designs either in the CoolSpice/Ckts/ folder (which is created by default) or in a folder at the same path level, such as CoolSpice/MyCircuits/.

Adding a trace

After the simulation is complete, the CoolCADPlotter window is automatically launched if the option is enabled as it is by default. To view results, click the **Add Trace** button (see Fig. **??**) or use the **Graph→Add Traces** menu item. The only option the software will present in this example is the current trace from the current probe, as seen in the figure.

Setting trace and view properties, navigating

Right-clicking on the plotting area allows the user to set properties of the trace (such as color and style), move traces between panels, make measurements or use other trace tools. Right-clicking and dragging in the plotting area pans around the plot. Single-clicking on an axis allows the user to set axis properties such as the label, ranges and step size. Click-and-dragging to create a rectangle in the plotting area zooms in on that area. Double-clicking on the trace-label box which displays the trace color and name zooms out to the full trace. Right-clicking on the trace-label box opens the trace properties. When there are multiple traces, clicking on a trace name in the legend box

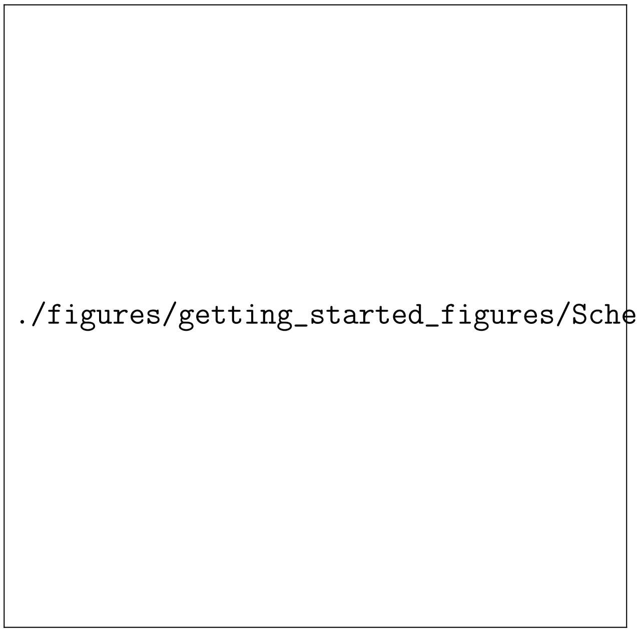

./figures/getting_started_figures/SchematicEditor_modelswrongplace.png

Figure 1.10: Error recorded in the log file if the model for devices in the circuit are not found where expected.

makes it the active trace for measurements. When there are multiple panes, a blue frame marks the active pane.

For voltage transfer function, we add a voltage probe to the schematic (the Probe - Voltmeter component under the Analog library) connected to the line labeled Vout, as shown in Fig. **??**. After re-running the simulation, we can now add both traces and view the current and voltage transfer functions of the inverter. Figure **??** shows the Plotter window when both traces have been added. Since one trace is from a current meter and the other from a voltage meter, they are by default plotted on separate y-axes. Note that the legend insets can be grabbed and moved by the mouse, as was done in this example since by default they blocked the low-input-voltage part of the voltage transfer function.

Another simulation example is given in Figure **??**, where a one-stage common-source amplifier is simulated with .tran (transient) analysis. Note that not V_{in} but $V_{in} - 1.25$ is displayed, so that the DC offset at the gate is removed to make reading the graph easier. This is achieved using the **Calculator** tool in the Plotter.

The last example here in Figure **??** is the same CS amplifier subjected to a frequency-domain analysis using the .ac analysis tool. Since the .ac tool in SPICE requires an AC source, note that the SourceV - Sinewave element in Fig. **??** has been replaced with a SourceV - DC AC element in the new circuit. The Plotter automatically plots the magnitude and phase of the selected traces, which allows the designer to evaluate the frequency domain characteristics of the circuit.

1.3.3. Example Circuits

Under the folder CoolSpice/Ckts/ there are example circuits which are bundled with the program. Running and playing around with the designs and SPICE analysis statements in these circuits is a good way to rapidly learn more about CoolSPICE and

./figures/getting_started_figures/Plotter_addingtrace.png

Figure 1.11: Adding traces to the plotter.

its capabilities. As an example, Fig. **??** shows the example circuit IV_IDVDM, a circuit designed to run I_D-V_{DS} sweeps on an TSMC 0.18 μm NMOS with V_{GS} as a parameter. Once the simulation is run as it is in the example, the curve i_vimeter is available to view in the Plotter.

1.3.4. Creating a New Symbol for Hierarchical Designs

To include a circuit or subcircuit in another schematic, a symbol needs to be associated with this circuit or subcircuit. In this example we design the symbol for the inverter circuit.

Polygon tool

The **SPICE→Add hierarchical symbol** menu item opens another tab in the design window. The original circuit remains on a tab named "Sheet 1" by default. The polygon tool, marked in Fig. **??**, can be used to draw lines and shapes. The ellipse, rectangle and arc tools are also available in buttons next to the polygon tool. When using the polygon tool, right-click and choose "Finish polygon" to stop drawing.

Adding symbol pins

The design itself should have labels on all the nodes which will correspond to pins in the symbol. We first revise the design to take out the DC analysis command and the sources, since the VDD and ground pins of the inverter symbol would presumably be connected to sources in the circuit into which the inverter symbol is inserted. Figure **??** shows, on the left, the revised circuit (note the new net names for VDD and ground) and, on the right, the corresponding symbol pins being added. The pins are added with the **Add Symbol Pin** button as marked in the figure. Before clicking to place a pin, define the pin name, direction, graphical type and electrical type with the dialogue box. Make sure that the electrical type fits what each pin needs to be for the circuit: In this example, Vin is an input pin, Vout is an output pin, and VDD and cktgnd are defined

./figures/getting_started_figures/SchematicEditor_voltageprobeadded.png

Figure 1.12: A voltage probe connected to the Vout net.

as input/output pins. If unsure which type to use, it is best to pick "input/output."

Hierarchical symbol references

Once the symbol has been saved, it can be incorporated into another circuit with the **SPICE→Insert another design as symbol...** menu item. The inverter circuit simulation results from Section are replicated using this new inverter symbol in Fig. **??**. Note that the reference names for inserted hierarchical symbols are not set automatically; nor are they incremented if you copy-paste a symbol to create multiple instances. Set a distinct reference name for each symbol instance to avoid netlist errors.

1.3.5. Creating New Component Models and Libraries

In the Schematic Editor, users can access existing libraries to edit them, or add new component libraries, by the **Libraries...** button at the top of the left-side frame (see Fig. **??**). This opens the Library Setup window. In this window, double-clicking a library or selecting it and clicking the **Edit** button opens the symbol list for this library. There, the user can right-click on an existing symbol to edit the symbol or perform other operations.

Creating a new library and adding a new component

As an example, we will create a new library and populate it with a new diode model. The **New** button in the Library Setup window opens a Save As... dialogue for the user to set the location and name for the new library. In this example, the library testlibrary1.TCLib is created and saved in a folder named MyLibraries/ under the CoolSpice folder. This sends the user back to the Library Setup window with the new library added to the list.

We will base the new diode on the symbol and part statement of an existing diode in the **"DiscreteParts"** library. Double-clicking this library brings up the devices it contains. Right-clicking on the `1N4148` diode, we choose **Copy to→...** and pick the new library we created, as shown in Fig. **??**. This brings up the Update Library Symbol

./figures/getting_started_figures/Plotter_twotraces.png

Figure 1.13: The Plotter displaying two traces.

dialogue box, showing the properties of the symbol just copied into the new library.

As seen in Figure **??**, Update Library Symbol is where the attributes and associated SPICE model for components are defined. For existing components, this window is launched by right-clicking on the symbol and choosing **"Symbol Properties..."**. For the new component, first click on the value of the Name parameter and give the new name. (Hit **Tab** or **Enter** for the value to be reflected in the left-side frame.) Note that the Reference parameter is set as **D** since our example device is a diode.

Under the SPICE tab, the "Model" box is used to define the default *part statement* the netlister uses to generate the SPICE netlist line when encountering this component. The "Prologue" or "Epilogue" boxes are used to define the .include statement which points to the model card. It is necessary to fill only one of these two; the netlister puts the .include statement either to the very beginning or the very end of the netlist depending on which one has been used. Since we copied a diode component, we do not need to edit anything in the Model box part statement except the model name (see Fig. **??**).

Specifying a device model card

The model name should match the model card (.model statement) in the text file specified by the .include statement in the Prologue or Epilogue box. For this example, the contents of the text file ../Models/ExtraParts.txt have been reproduced below.

```
   .model TestDiode D(Is=5.84n N=1.94 Rs=.7017 Ikf=44.17m Xti=3 Eg=1.11
 +Cjo=.95p M=.55 Vj=.75 Fc=.5 Isr=11.1n Nr=2.09 Bv=100 Ibv=100u Tt=11.1n)
```

Click **Store** to save the symbol attributes and SPICE information and return to the symbol display window. The new component can now be used in the Schematic Editor. The new library appears in the library list and includes and the new component (See Fig. **??**).

Another method to specify the component SPICE model is to write out the .model statement in the Prologue or Epilogue dialog box instead of putting it in a text file and pointing to the file with a .include statement as above. However, the .include method

./figures/getting_started_figures/CoolSPICE_examplesimulation_tra

Figure 1.14: A basic common-source amplifier simulated using transient analysis.

has the advantage that it is possible to incorporate more than one .model statement in a text file. Then, when defining the models for multiple components within the same library, the same .include statement works for all. This helps with organization. For instance, Fig. **??** shows another new device, this time a MOSFET, defined in the same library by the copying method described above. For it to work correctly, the file ../Models/ExtraParts.txt should now also include an NMOS .model statement with the model name "generic018nmos."

Editing existing component models

It is also possible to edit existing libraries, point existing models to new .model statements *etc.* by starting with the **Libraries** button as above. However, note that if the model for a component with existing instances in the design is changed, either by pointing it to a new model card file by editing the .include statement or by editing the .model statement in the model card text file, *only new instances of it will use the new model in the netlist.* Existing instances will have to be deleted and replaced to be associated with the modified model.

1.4. References and Further Reading

1. CoolCAD CoolSPICE, http://coolcadelectronics.com/coolspice/, last visited October 2014.

2. WinSCP, http://winscp.net, last visited October 2014.

3. List of SPICE device codes, http://www.ecircuitcenter.com/SPICEsummary.htm#devices, last visited October 2014.

./figures/getting_started_figures/CoolSPICE_examplesimulation_ac.p

Figure 1.15: A basic common-source amplifier simulated using ac analysis.

./figures/getting_started_figures/CoolSPICE_examplecircuit_IV_IDVD

Figure 1.16: The example circuit IV_IDVDM found in the Ckts/ directory and its run results.

./figures/getting_started_figures/SchematicEditor_drawingsymbol1.

Figure 1.17: Creating an inverter symbol using the hierarchical symbol definition tool in the Schematic Editor.

Make sure to set reference names for the hierarchical symbols.

./figures/getting_started_figures/SchematicEditor_revisedcircuita

Figure 1.18: The inverter circuit revised for representation as a hierarchical symbol (left) and pin definition in the symbol view (right).

./figures/getting_started_figures/CoolSPICE_invertersymbolincircu

Figure 1.19: The inverter symbol used in a simulation.

./figures/getting_started_figures/SchematicEditor_openinglibraries

Figure 1.20: The **Libraries...** button displays libraries and allows new ones to be created.

./figures/getting_started_figures/SchematicEditor_copymodeltonewlibrary.png

Figure 1.21: An existing diode model is copied from the DiscreteParts library to the newly-created testlibrary1 library for easy use of the symbol and the part statement.

./figures/getting_started_figures/SchematicEditor_UpdateLibrarySymbol.png

Figure 1.22: The name, description and other attributes of the new component are defined in the **Attributes** tab of the Update Library Symbol window.

./figures/getting_started_figures/SchematicEditor_enteringthemodel.png

Figure 1.23: The part statement and .include statement pointing to the file which includes the .model statement for the device are defined in the SPICE tab of the Update Library Symbol window.

💡 **Note that the model name (`TestDiode`) specified at the end of the part statement in the Model box in Fig. ??.**

💡 **The netlist creator does not check if models are up to date with existing instances of components! If you changed a model, replace the instances to apply the change to everything.**

./figures/getting_started_figures/SchematicEditor_NewLibraryDeviceAvailable.png

Figure 1.24: The new library and component (with its symbol and model) are now available for use in schematics.

./figures/getting_started_figures/SchematicEditor_MOSFETmodelexample.png

Figure 1.25: The part statement for a MOSFET defined in the new library by copying the MOSFET symbol from the MOSIS library.

2. The Schematic Editor

CoolCAD Schematic Editor is the schematic capture tool for the CoolSPICE suite. It comprises symbol and analysis tool libraries, symbol placement and wiring tools, and drawing tools with a built-in symbol editor for the design of new symbols. The designer can create the SPICE netlist from the schematic and automatically invoke the CoolCAD SPICE engine to run the simulation from within the Schematic Editor.

Quick references to the tool buttons available in the Schematic Editor can be at the end of this Chapter.

2.1. Drawing a Circuit

By default, the Schematic Editor opens with an empty schematic named "Cool1" in an inset window. The inset windows can be closed, maximized to cover the full Schematic Editor "desktop," or minimized. A new window is opened by the **File**→**New** menu item. An existing design is opened by the **File**→**Open...** menu item.

The mouse scroll button can be used to zooming in or out of the schematic. Clicking and holding the scroll button (or the middle button in a three-button mouse) can be used to pan around the schematic.

Schematic navigation

2.1.1. Component Placement and Options

Components are inserted by using the Symbol Picker frame, which is visible by default on the left-side of the Schematic Editor window and can be toggled on/off using the **Symbol Picker** button ($\boxed{\Omega}$, see Fig. **??**). The panel libraries can be expanded using the $\boxed{+}$ buttons.

Double-clicking a component name, or single-clicking and pressing the **Place Symbol** button, or single-clicking the symbol representation in the frame underneath the library list, puts Schematic Editor in placing mode. Multiple copies of a symbol can be placed by clicking consecutively in new locations.

As with any action in the Schematic Editor, the placing action can be cancelled by pressing the **Esc** key.

Components are selected individually by clicking once on the symbol placed in the drawing area. This brings the Options window back up. Figure **??** shows the different elements of the Options window:

Options details

- *Orientation Options* allow the user to place the symbol in four cardinal orientations and as a mirror-image if desired.

- *The Reference Name*, which may be automatically assigned by the program or manually by the user, is the name that the netlister will use to reference the component in the netlist. Note that unclicking the tickbox next to Ref *only hides the reference name in the circuit view*; the netlister still uses the reference name to identify the component.

- *Show Power Pins* enables/disables the display of power pins.

- *Allow Re-Sizing* shows or hides a resizing box with which the user can resize the symbol to better suit the design need.

- *The Temperature Setting* sets the individual device temperature for temperature-dependent parameter calculation during simulation. This has no effect in the Student Edition.

- *The* **Show Model** *Button* shows the model card associated with the symbol. If this model card is defined within a text file with an .include statement (see Section **??**) the program will display the entire file.

- *The Parameters* change depending on the component type. Some parameters are compulsory to define for simulation. More information on component parameters can be found for analysis commands and analysis-related components such as probes, and for active and passive devices as well as behavioral devices.

- *The* **Add** *and* **Delete** *Buttons* allow the user to define a new parameter or remove it. This is useful for user-defined symbols detailed in Section **??**.

Right-clicking on the schematic window brings up a contextual menu which allows the user to perform some actions on the selected component, such as rotation, copy/paste, or cut. The first option in the contextual menu, Replace Symbol... allows the user to replace either only the selected symbol, or all instances of this symbol in the design, by another symbol which must first be selected by entering part of its name into the search box.

2.1.1.1 Reference Name Assignment

The program automatically increments names for new instances of a given component if:

- The user picks the component by double-clicking on the name in the library or single-clicking on the symbol representation and places multiple instances by repeated clicks in the circuit design area without altering the Ref box in the Options window, or

./figures/schematic_editor_figures/SchematicEditor_PartOptionsDialogue_Annotated.

Figure 2.1: The Options window for a TSMC 0.18um C018 PMOS from the MOSIS library. See the text body for the annotations.

- The user selects an already-placed component and chooses **Edit→Copy** menu item / **Edit→Paste** menu item (or uses the Ctrl-C / Ctrl-V key combinations) to place new instances without altering the Ref box in the Options window.

  However, the software will *not* increment the name in the case that the user:

Name increment exceptions

- ...picks the component by double-clicking its name in the library list or single-clicking the symbol representation, and edits the reference name in the Ref box *before* placing the first instance, or
- ...edits the reference name in the Ref box in between placing further instances, or
- ...selects an already-placed component and edits the reference name in the Ref box to a name which contains only numbers (*e.g.* "123"), then uses the copy-paste method to make further copies.

These cases may result in instances of the same component with identical names, which will lead to netlist conflicts.

It is very important to ensure that each component instance has a distinct Reference Name. Do not use reference names comprising only numbers.

Also note that if the user edits the reference name of one of the instances of the same type of component to something out of the sequence, then makes further copies of the component, the incrementer will continue incrementing from the lowest available number appended. For instance, if the circuit has instances N1 through N5, and the user edits N5 to N42 and then places more instances by copy-pasting from any of the present instances, the incrementer will continue from N5, not N43.

2.1.2. Wiring and Labels

To start laying out a wire, click the **Wire** tool button (see Fig. **??**) or press F2. The Options window allows you to set the wire angle. The first turn in the wire segment will be detected by the program depending on the mouse movement. Double-clicking after a straight extension or right-clicking at any time ends the piece. When the wire crosses another wire, a circle appears to present the option of forming a connection with a single click. A connection also ends the wire segment. Figure **??** shows the wired circuit just before the last connection is placed.

Labelling

In SPICE netlists, wires appear as the circuit nodes they represent. If the wires are not labeled in the schematic, the netlister will use an automatic numbering scheme. Labelling gives netlists that are easier to read (and after simulation, makes it easier to identify which signals to look at). The **Label** button (a) or the F1 key is used to set labels. Figure **??** shows the net Vout being labelled. In the Label Options window, the top row is used to select the orientation of the letters. The red dot that appears with the label on the schematic shows the location of the label (the net contacting this dot will take on the label). Once the label is placed, the user can write another label name in the dialogue box. The **Font** and **Colour** buttons can be used to set the font and color of the label on the schematic.

./figures/schematic_editor_figures/SchematicEditor_LabelOptionsDialogue.png

Figure 2.2: Labelling the node Vout.

With the labels, the MOSFET appears in the netlist with the following line:

```
MN1 Vdrain Vin Vs Vs on0p5nmos W=20u L=3u
```

Whereas without the labels, the netlister automatically numbers the nodes:

```
MN1 _N_4 _N_8 _N_3 _N_3 on0p5nmos W=20u L=3u
```

2.1.3. Power Nets and Sources

The power and ground nets are defined using inserted into the circuit using the **Power** button (The "earth ground" symbol, see Fig. **??**) or pressing **F4**. The name of the pin is entered using the dialogue box. Different designs and orientations for the pin are available in the Options window.

The ground pin name must be 0**.**

Multiple pins with the same name are used to denote the same node, which allows the user to avoid overly complicated wire networks in the schematic. For an example, the netlist of the circuit shown in Fig. **??** has the DC source VS2 correctly wired between the away-from-drain end of the resistor R1 and the ground. For a circuit with multiple stages with drains all connected to VDD, this would cut down on the number of required wires. This also enables multi-sheet designs, as described below in section **??**.

./figures/schematic_editor_figures/SchematicEditor_MultiPinUseExample.png

Figure 2.3: Using multiple copies of the same pin to clarify schematic design. Each arrow-paired pair of marked dots are connected to the same node.

Sources

The voltage and current sources are found under the Analog library. The voltage sources are labeled "SourceV" and current sources are labeled "SourceI". Details about the parameters of these components are found elsewhere. From the Schematic Editor the user can define DC/AC, Pulse (i.e. square wave) and Sinewave sources. Other sources, such as the piecewise linear source, can be defined using the Netlist Editor.

2.1.4. Probes

Inserting a current or voltage probe in the schematic directs the netlister to insert a `.save` statement in the netlist to store the branch current or node voltage results as appropriate for the proper analysis. The probes are available in the Schematic Editor under the Analog library. Naturally, current probes must be inserted into a branch and voltage probes connected to nodes. Figure **??** has an example circuit with a current probe measuring the drain current and three voltage probes measuring the gate, drain, and post-DC-filter output voltages.

2.1.5. Further Navigation and Schematic Commands

As mentioned above, one can use the mouse scroll button to zoom in and out of the design and the middle button/scroll button click-and-hold to pan around the circuit.

Right-clicking anywhere around the drawing area brings up the contextual menu, which has a number of general commands and a number of selected-component-specific commands that are available. (The selected-component-specific commands are greyed out unless the user right-clicks in the symbol area of a selected component.) Figure **??** shows the contextual menu invoked on the current meter Imeter1 in the displayed circuit.

The **Z-Order→Bring to front** menu item and **Send to back** commands in the contextual menu are useful for when images are imported into the Schematic Editor for use in a symbol or similar, as will be described in Section **??**.

It is possible to move the shown properties of components (including the reference names) to avoid conflicts and overlays. In Figure **??** , the reference name Voutmet1, associated with the visible voltage probe, has been moved from its default location when the component is first set (which was directly above the visible .op line in the figure) by left-click and dragging on the text line.

2.2. Multi-sheet Designs

For complicated circuits, it is possible to split the circuit drawing into multiple sheets. For a given design (.dsn file), the netlister will combine all sheets into a single circuit netlist. Node and wire names should not be repeated between sheets except where a connection is explicitly intended, and the designer should take care that multiple components do not have the same reference name between sheets either.

To start a new sheet, or to delete or rename a sheet, right click on the sheet tab at the bottom of the design window, which says Sheet 1 by default when a circuit is first started.

Use the power pins and wire labels to ensure correct connectivity between the circuit parts on different sheets. For an example, Figure **??** shows a two-stage amplifier split into two sheets for illustration. (While this is a circuit which could easily fit on a single sheet, the simplicity helps the example.) The features to note here are how the drain-side voltage source for both transistors is defined on Sheet 1 as the voltage source

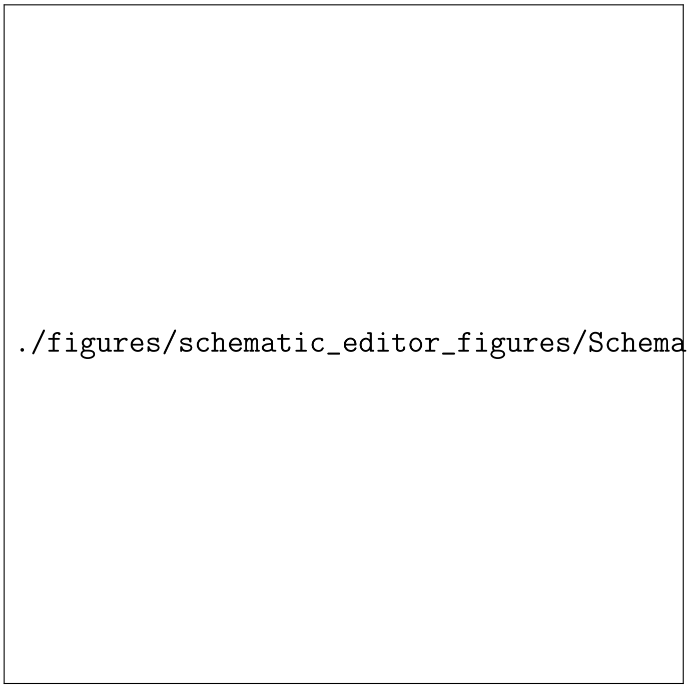

./figures/schematic_editor_figures/SchematicEditor_ContextualMenu_MovingTextLabel

Figure 2.4: The contextual menu comes up with a right-click in the schematic area. If a device's symbol area is selected, the component-specific commands are enabled. Also note that it is possible to move the reference name and property text for components, as for the case for the reference name of the voltage probe here.

VS2 as described above in Section **??**, how the gate, drain and source networks have distinguishing names between the two sheets, and how the DC-isolated output of stage 1, equivalently the DC-isolated input of stage 2, bears the same node name Vout_st1.

Part of the netlist for this circuit is shown here:

```
CC1 Vout_st1 Vdrain_st1 3n
CC2 Vout_st2 Vdrain_st2 3n
CC3 Vin_st2 _N_13 3n
MN1 Vdrain_st1 Vin_st1 Vs1 Vs1 on0p5nmos W=20u L=3u
MN2 Vdrain_st2 Vin_st2 Vs2 Vs2 on0p5nmos W=20u L=3u
RR1 Vdrain_st1 VDD 20k
RR10 _N_13 Vout_st1 10
(...)
RR4 0 Vout_st1 75k
RR5 Vdrain_st2 VDD 20k
(...)
RR7 0 Vout_st2 75k
(...)
VVS2 VDD 0 DC 3 AC 0
.save v(Vout_st2)
.save v(Vout_st1)
.save v(Vin_st2)
.save v(Vin_st1)
```

Note that the resistors RR1 and RR5 are the resistors between the two transistors' drains and VDD, and they are correctly shorted to the same node whose voltage is set

./figures/schematic_editor_figures/SchematicEditor_MultiSheetDesign.png

Figure 2.5: A two-stage amplifier as an example of multi-sheet design.

by the voltage source VVS2. When this circuit is netlisted and ran, all the traces from the four visible voltage probes will be saved and be available for plotting.

2.3. Models and Libraries

2.3.1. Symbols and Hierarchical Designs

2.4. Netlisting and Simulation

2.4.1. OP Analysis

2.5. Options and Preferences

2.6. Schematic Editor GUI Reference

./figures/appendix_buttons_menus_figures/SchematicEditor_LeftButtons.png

Figure 2.6: Left-side buttons on the Schematic Editor.

A list of menu items for the Schematic Editor can be found in the Appendix.

./figures/appendix_buttons_menus_figures/SchematicEditor_RightButtons.p

Figure 2.7: Right-side buttons on the Schematic Editor.

3. Plotter and Netlist Editor

3.1. Plotter

The CoolSPICE Plotter is used to plot simulation results, stored in .raw files. These "rawfiles" are data files containing simulation results (node voltages and branch currents indicated by probes in the schematic, or by `.save` statements in the netlist). These files have the same file name as the netlist or schematic, and are placed in the same directory by default.

3.1.1. Start-up and Loading Data

The plotter can be started from the main console by clicking on the **Plotter** button. Then using the **File**→**Open** menu item (**Ctrl**-**O**, or the **Open** button), the user loads a simulation result rawfile (.raw extension) and can plot the available traces stored in this file. A file must be loaded first for traces to be available for plotting.

Rawfiles are plain text files allowing the user to view raw data directly.

If the Open Plotter option is checked under the **SPICE**→**Preferences** menu item of the Schematic Editor, the Plotter will start automatically at the completion of each successful simulation run, or an error message will pop up if the simulation did not complete successfully and write a .raw file.

Multiple subwindows

The Plotter window can be divided into multiple inset windows the same way as the Schematic Editor can be, by using the subwindow minimize/part window/close buttons (marked as "subwindow control windows" in the figure in Section **??**). To create a new subwindow, simply load a new rawfile as described above and use the subwindow control buttons to minimize and arrange buttons. Fig. **??** in the shows an example of signals from two separate rawfiles being plotted.

3.1.2. Adding Panes and Traces, Editing Traces

The Plotter window can be subdivided into multiple "view"s and each "view" can be further subdivided into multiple panes. Each pane can be used to plot one or more traces. To split a pane and create a new pane, use the Split Pane button (see Fig. **??**

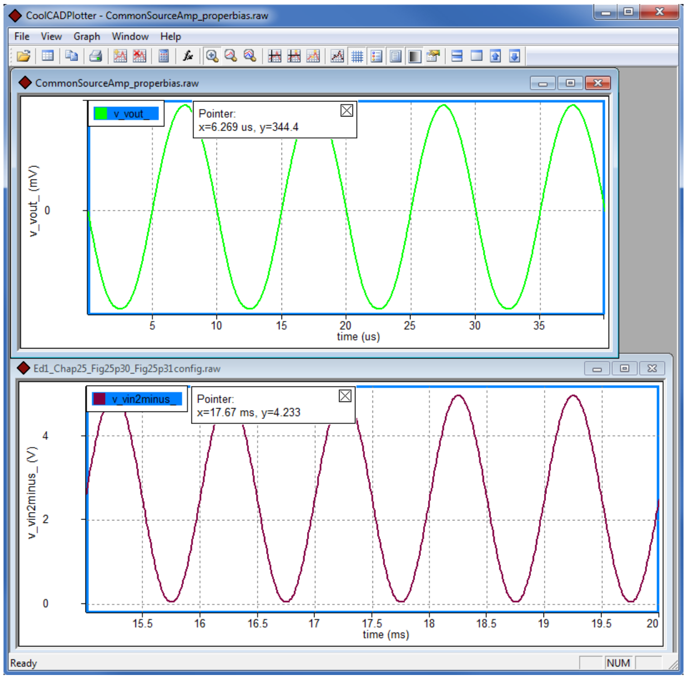


Figure 3.1: Signals from two designs plotted in the same main Plotter window.

and the annotated figures in Section **??**) or the **View→Split Pane** menu item. The active pane is highlighted with a blue border as shown in the figure. Operations such as adding and removing traces are performed in the active pane, which can be selected by left-clicking in the pane area.

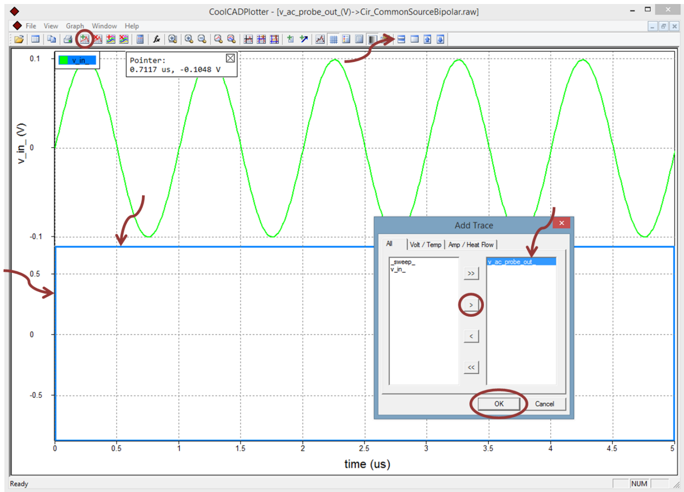


Figure 3.2: Two panes created on a plot window in which the file Cir_CommonSourceBipolar.raw has been loaded. The trace v_vin_ has already been plotted on the top trace. The bottom trace is the currently active one, into which the user is adding v_ac_probe_out_ to plot. The Add Trace and Split View buttons are marked by a circle and an arrow, respectively.

To add a trace, use the "Add Trace" button from the button panel (see Fig. **??** and the annotated figures in Section **??**), the **Graph→Add Traces** menu item, or right-click on the active panel and choose Add New Trace from the contextual menu. A list of the saved traces from the loaded rawfile (or from the recent simulation, if the Plotter was initiated from a simulation) comes up. To add one or more traces select available traces from the left hand list and move them to the right hand list, when the desired traces have been selected click the **OK** button (see **??**).

If the traces chosen have different units, i.e. if voltages and currents are chosen together, the Plotter will automatically split the view to put them on different y-axes.

The "sweep" trace is the independent variable against which the other traces are plotted.

Legend and pointer boxes

On each pane, by default there is a legend box which lists the traces and a trace info box, as shown in Figure **??**. The legend box can be toggled on/off by using the button indicated in the figure by a single dot, from the right-click contextual menu by checking "Show Legend," or from the **Graph**→**Show Legend** menu item. The trace info box can be toggled on/off by the toolbar button indicated by double dots, or from the right-click contextual menu by checking "Show Trace Info." Both boxes can be moved around in the pane, so that they do not block the traces, by left-clicking within the box and dragging.

Trace properties

The name of the active trace is highlighted in blue in the legend box, and operations such as moving the traces between panes or deletion will be applied to the active trace (see Sec. **??**). Right-clicking within the trace info box, using the **Graph**→**Properties** menu item or right-clicking within the pane area to bring up the contextual menu and choosing Properties will bring up the dialogue box which can be used to set the trace properties as shown in Fig. **??**. The user selects which trace to edit in the "Trace" drop-down box, which lists all the invoking pane's traces and their colors. This drop down list is editable allowing renaming of the selected trace. The "Style" drop-down box is used to pick the line style, choosing between solid, dashed and dotted lines. The "Thickness" text box allows sets the width of the trace. The "Pane" drop down list sets which pane the selected trace is in and can be used to move the trace between panes or to a newly created pane. "Unit" drop down list can set the unit of the selected trace, note this will move the trace to a pane of that unit type, or a new pane if none already exists. The "Unit" drop down list is editable. The color of the trace is modified with the **Color** button. The **Hide** button can hide a given trace; if this option is selected

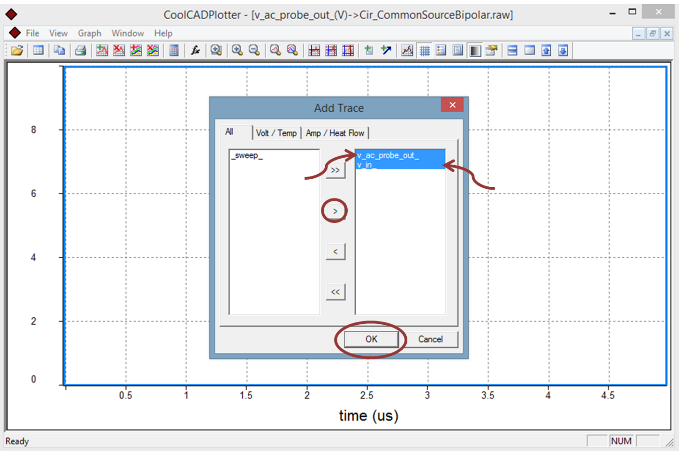
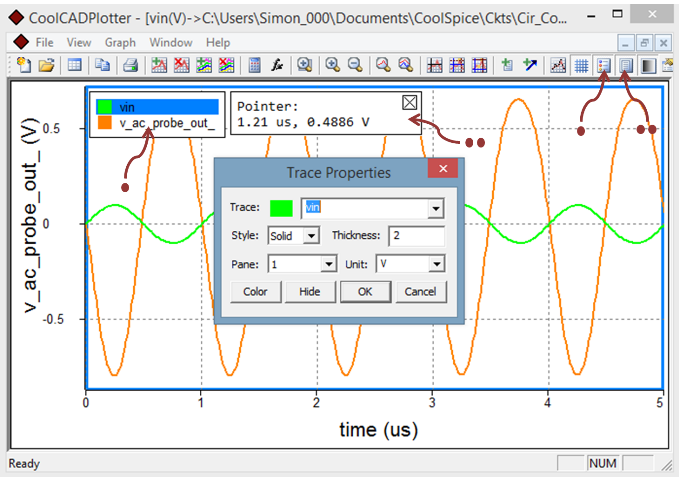


Figure 3.3: Choosing multiple traces to plot.

and applied with the OK button, the button will show up as Show the next time the Graph Properties window window is invoked to toggle the visibility of the trace back on. Changes made to a trace only take effect when the OK button is clicked OR when a different trace is selected.



CoolCADPlotter - [vin(V)->C:\Users\Simon_000\Documents\CoolSpice\Ckts\Cir_Co...

Pointer:
1.21 us, 0.4886 V

Trace Properties

Trace: 🟩 vin

Style: Solid

Thickness: 2

Pane: 1

Unit: V

Figure 3.4: The Trace Info box, the pointer box, and the Graph Properties window dialogue window.

Removing panes

Panes can be removed or merged back into a single pane by using the "Remove Split View" button (see Fig. **??**) or using the **View**→**Remove Split** menu item. All the traces present in the merged panes will be plotted in the new joint pane (assuming they have the same x-axis) and the default zoom level will fit all traces.

3.1.3. Navigation

Moving traces between panes or windows

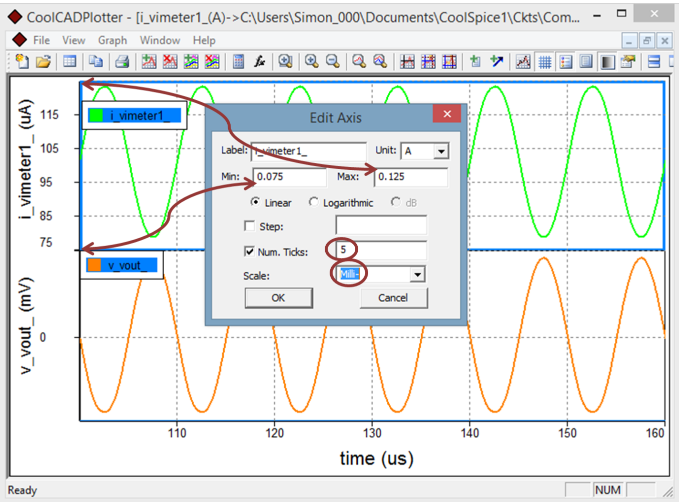
Once an active trace is selected, it can be moved between panes by either using the relevant buttons on the right-hand side of the button pane (See **??**) or the **View**→**Move Graph Up/Down** menu items. If multiple subwindows (with trace sets from different rawfiles) are open (as in Fig. **??**), the active trace may be moved to one of the other subwindows by the contextual menu option "Move Trace to New Window."

Panning and zooming

Right-clicking and dragging in a pane will pan around. If there are multiple panes sharing the same x-axis, the x-dimension panning will be mutual. The y-direction panning is always independent. Zoom-Select mode will adjust the pane to show the selected rectangle. To zoom out so that the axes span the whole of the full range of the active trace, use the “Fit Current” toolbar button (Fig. **??**) or the **Graph→Zoom to Current Trace** menu item. To zoom out to cover the full range of all traces in the active pane, use the “Fit All” toolbar button, or the **Graph→Zoom to Fit** menu item.

Setting axis ra and properties

Left-click in the area of an axis where numbers are displayed to edit the axis properties. The Edit Axis window window will appear, as shown in Fig. **??**. The axis label, unit and minimum/maximum range can be specified with the dialog's text boxes. The user can select between linear, logarithmic and decibel modes of axis display. The user can also choose to either place ticks at a given step separation by choosing the checkbox next to Step and entering the separation value in the dialogue box, or the checkbox next to Num. Ticks and entering the desired number of axis ticks in the dialogue box. Only one of those two options will be valid. Finally, the axis scale can be toggled between the Plotter's automatic choice and a range of choices that make sense by using the Scale drop-down box.



CoolCADPlotter - [i_vimeter1_(A)->C:\Users\Simon_000\Documents\CoolSpice1\Ckts\Com...

Edit Axis

Label: i_vimeter1_ Unit: A

Min: 0.075 Max: 0.125

● Linear ○ Logarithmic ○ dB

☐ Step:

☑ Num. Ticks: 5

Scale: Milli-

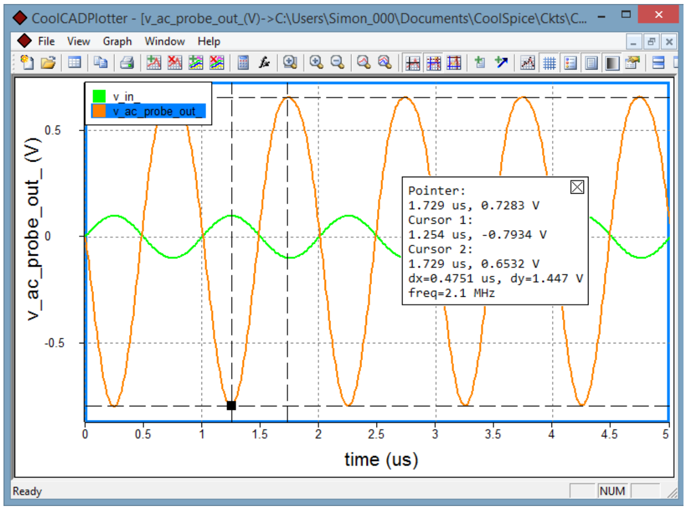
Figure 3.5: Setting the axis range and properties. The y-axis of the upper (active) pane has been selected in this example. The user has previously set the maximum and minimum values to 75 and 125 μA respectively, and selected the “number of ticks” option and set five tick values to show. The scale is being set to “milli,” (which is why Min and Max are 0.075 and 0.125 in the dialog) once **OK** is pressed, the y-axis will shift to showing the values in mA instead of the initial automatic μA choice.

3.1.4. Measurements

The measurements can be taken on the traces are displayed in the pointer/trace info box.

The “Display Difference” toolbar button or the **Graph→Get Difference from Point** menu item menu item allows the user to select a point and displays the mouse marker's difference on the trace from this point. When this button is pressed or command selected, cursor markers appear. Even though the intersection of the horizontal and vertical marker does not seem to be attached to the trace and can move freely around the active pane, when the user clicks the left mouse button, the point on the active trace that corresponds to the that x-axis location is marked and shown as "Cursor 1" in the pointer box. After this point, a new pair of horizontal/vertical cursor lines, shown as the "Cursor 2" location in the pointer box, are invoked and are attached to the active trace. As the user moves the mouse around, the cursor lines for Cursor 2 move along the trace and the x-axis and y-axis distances of the real-time location from the previously-marked point are displayed in the pointer box as dx and dy, as shown in Fig. **??**.

If the trace is a time-domain signal, the pointer box also shows "freq", which is 1/dx.



```
Pointer:
1.729 us, 0.7283 V
Cursor 1:
1.254 us, -0.7934 V
Cursor 2:
1.729 us, 0.6532 V
dx=0.4751 us, dy=1.447 V
freq=2.1 MHz
```

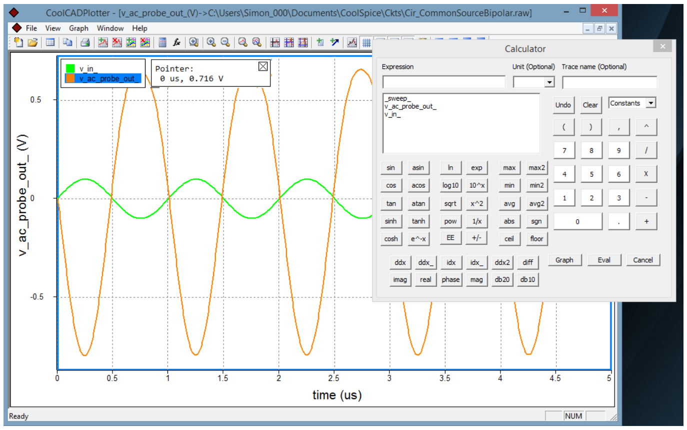
Figure 3.6: Measurement from a point. In this example, the point (x, y) = $(1.254$ μs, -0.7934 V) has been selected and the mouse pointer (not visible) has been moved to (x, y) = $(1.729$ μs, 0.7284 V), which puts the trace-attached Cursor 2 at (x, y) = $(1.729$ μs, 0.6532 V). Δx=dx=0.4751 μs, Δy=dy=1.447 V.

The "Get Difference Between Points" button (See Fig. **??** for the button marked "Two Selected Points Diff" or the **Graph→Get Difference between Points** menu item) works the same way. However, this option allows the user to mark first the point for Cursor 1, then the point for Cursor 2, and then turns off the cursors. To select new points, the user has to toggle the menu option off and back on. Once the option is turned off, the marked points and the difference value display in the pointer box will disappear.

3.1.5. Calculator

The calculator can be invoked by the Calculator button (see Fig. **??**) or the **Graph→Calculator** menu item. The traces present in the loaded rawfile are available as data items, effectively vectors, in the calculator. Calculations can be carried out using numbers and functions just as in a regular calculator. Expressions can be generated using the buttons or manually entered into the Expression text box. For calculations which evaluate to a single number, clicking the **Eval** button evaluates the expression and puts the result in the calculator's Expression text box. For these calculations, using the **Graph** button graphs a constant value vs. the same x-axis as the active trace.

For expressions which must evaluate to a trace, the **Eval** button has no function. The **Graph** button graphs the calculated variable in the active pane if the y-axis is compatible, or in a new pane if necessary. To select the trace or traces to use as a variable, the user can either enter the trace names exactly as shown in the selection box into the Expression text box, or simply click on the name of the trace in this box, which will then copy the name over to the expression box, as shown in Fig. **??**. Multiple expressions can be graphed simultaneously by separating them with a comma. A unit can be specified in the editable Unit drop down menu, this allows the user to control the destination pane for the graphed expression. A trace name can be specified in the Trace name text box to be used as a name for the graphed trace in place of the expression itself, multiple names can be specified for multiple traces separated by commas.



CoolCADPlotter - [v_ac_probe_out_(V)->C:\Users\Simon_000\Documents\CoolSpice\Ckts\Cir_CommonSourceBipolar.raw]

```
Pointer:
 0 us, 0.716 V
```

```
_sweep_
v_ac_probe_out_
v_in_
```

Figure 3.7: The calculator showing available traces.

3.1.6. Plotter Options and the Contextual Menu

The display options of the Plotter, which can be accessed through the **Graph** menu or the contextual menu and toggled on or off by clicking on them, are as follows:

- **Show Data Points**: By default, the Plotter interpolates between the data points saved in the rawfile when plotting a trace. When this option is toggled on, individual data points are marked on the curve.
- **Show Grid**: Toggle on/off the grid display.
- **Show Legend**: Toggle on/off the trace legend box display in the active pane.
- **Show Trace Info**: Toggle on/off the pointer/trace information box display in the active pane.
- **White Background**: Switch between white and dark backgrounds.

The contextua menu

The contextual menu is invoked by right-clicking anywhere within the panes. Figure **??** shows the menu items, some of which are toggle options.

The contextual menu items are as follows:

./figures/plotter_netlist_editor_figures/Plotter_ContextualMenu.png

Figure 3.8: The contextual menu items.

- Add New Trace and Delete Current Trace bring up the dialog box to plot new trace or traces, and delete the active trace, respectively.

- Move Trace to New Window moves the active trace to another subwindow, as described in Sec. **??**.

- Zoom In, Zoom to Current Trace and Zoom to Fit are zooming options described in Sec. **??**.

- Trace Cursor is a toggle option which turns on/off dashed cursors to indicate the pointer location in the pane.

- Get Difference from Point and Get Difference between Points are used to perform measurements on the traces as described in Sec. **??**.

- Show Data Points is a toggle option which shows/hides data points that were saved in the rawfile. The **Graph→Show Point Marks** menu item has the same function.

- Show Grid is a toggle option to turn on/off the grid. The **Graph→Show Grid** menu item has the same function.

- Show Legend and Show Trace Info are toggle options which display/hide the legend box and the pointer box, respectively. The **Graph→Show Legend** menu item has the same function for the former.

- White Background is a toggle option which makes the background white or black. The **Graph→Use White Background** menu item has the same function.

- Properties brings up the Graph Properties window.
- Move Graph Up/Down items move the traces between panes.

3.1.7. Plotter GUI Reference

See Appendix section for larger versions of these images.

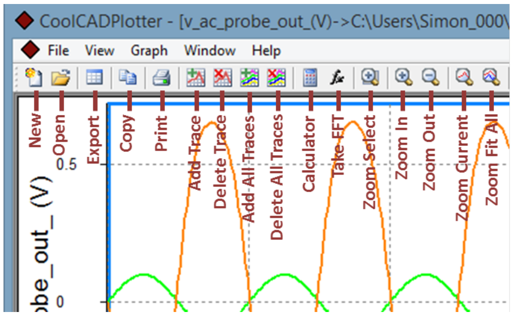


Figure 3.9: Left-side buttons on the Plotter.

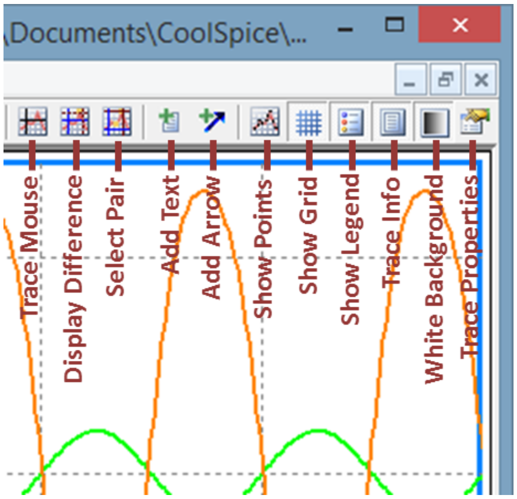


Figure 3.10: Right-side buttons on the Plotter.

A list of menu items for the Plotter can be found in the Appendix Section.

3.2. Netlist Editor

3.2.1. SPICE Netlist Syntax

All SPICE commands, including analysis commands, begin with ".", such as "`.include`", "`.tran`", or "`.end`". Component descriptions begin with a single character indicating the type of the component, such as `M` for MOSFETs, `D` for diodes and `R` for resistors.

The first line of any SPICE file must be a commented title line starting with '*'. The last line of any SPICE file must be an `.end` statement:

Beginning and ending

```
.end
```

Commenting

The comment character is '*' and is placed at the beginning of a line. For example, in the following code snippet, the first and third lines are comments:

```
* Trying two resistor values by commenting one out at a time
R12 N2 N3 40k
*R12 N2 N3 60k
C12 N2 N3 10p
```

Multiple lines can commented and uncommented CTRL-K and CTRL-SHIFT-K, useful for disabling and enabling blocks of the netlist.

Multi-line statements

When a statement continues for more than a single line in the text file, the + character must be used to indicate the new line is a continuation of the previous statement. For instance:

```
.model newpmos pmos ( LEVEL = 49
+VERSION = 3.1 TNOM = 27 TOX = 1.4E-8 XJ = 1.5E-7 NCH = 1.75E17
+VTH0 = -0.9 K1 = 0.55 K2 = 8E-3 K3 = 1.5 K3B = 0.025 W0 = 1E-8
+(...)
```

The `.include` statement

The `.include` statement is used to include subcircuit and device model definitions from another text file containing `.subckt` and `.model` statements. For long, complicated circuits, this allows the user to keep the main circuit topography, simulation options and simulation commands in one file while keeping the device and subcircuit definitions in auxiliary files. For example,

```
* Schematics Netlist *
.include "../Models/MyMOS.txt"

MN1_x line_out _HN_1_Vin 0 0 nmos0p5 W=3u L=0.6u
MP1_x line_out _HN_1_Vin _HN_1_VDD _HN_1_VDD pmos0p5 W=4.5u L=0.6u
VVG1 _HN_1_Vin 0 DC 3
VVS1 VDD 0 DC 3

.dc VVG1 0 3 0.05
.end
```

In this example, the models `nmos0p5` and `pmos0p5` must be defined in the file ../Models/MyMOS.txt. More details about the `.include` statement can be found elsewhere.

The `.subckt` statement

The `.subckt` statement is used to define a subcircuit, which acts like a device model which incorporates other device models. Once defined, multiple instances of the subcircuit can then be invoked throughout the main circuit just as the case for a regular device. Parameters defined within a subcircuit are visible only within the scope of that subcircuit, and if two parameters of the same name exist within nested scopes the parameter of the smaller scope is utilized. Subcircuits can be nested.

3.2.2. Editing and Running a Netlist

To run the Netlist Editor, click on the Netlist Editor button on the CoolSPICE main console window.

File→New menu item, **New** toolbar button or CTRL-N will open a new, empty file. A SPICE netlist can simply be typed in and saved by **File→Save** menu item or CTRL-S. For the syntax, see Section **??** above.

More often, an existing netlist generated by the Schematic Editor is opened to be edited by **File→Open** menu item or CTRL-O. Figure **??** shows an example circuit and its netlist opened in the Netlist Editor after being generated by the Schematic Editor. It is possible to save an open netlist under another name with the **File→Save As** menu item or CTRL-SHIFT-N.

To run the SPICE netlist, press F5, use the **Run→Run Netlist** menu item, or press the "Play" button. The progress bar will display the progress of the simulation. Long simulations can be paused by the "Pause" button or stopped by the "Stop" button. See Sec. **??** for an annotated screenshot showing the button locations.

3.2.3. Netlist Editor Options

The options available under the **Options** menu are the same as those available under the **SPICE→Preferences** menu item of the Schematic Editor with one exception, Syntax Highlighting. These are all toggle-on or toggle-off options.

Open Plotter, if checked, will automatically launch the Plotter after a simulation is successfully completed. If this option is checked but there are no plottable results (e.g. due to convergence problems or a netlist error) an error message window will show up instead.

Show Rawfile, if checked, will automatically display the raw result data of the simulation after the simulation is successfully completed.

Show Logfile, if checked, will automatically display the simulation log file after the simulation or simulation attempt is complete. Note that the log file for the most recent run of the netlist can be opened at anytime with CTRL-L.

Measure (No Rawfile) only applies to netlists containing a `.measure` statement and asks the program to look for and display only the log file output with the measurement results after the simulation is complete.

Highlight, if checked, will automatically color lines of the netlist according to their type, e.g. green for comments. Syntax Highlighting can also be toggled with CTRL-H or the Syntax Highlight toolbar button.

As a display option, the button "Syntax Highlight On-Off" (see Section **??**) will turn the color-coding of the SPICE commands on or off.

A list of menu items for the Netlist Editor can be found in the Appendix Section.

./figures/plotter_netlist_editor_figures/NetlistEditor_ExampleNetlist1.png

Figure 3.11: An example circuit open in the Schematic Editor and the corresponding netlist open in the Netlist Editor (the window is named “Text Editor”).

./figures/appendix_buttons_menus_figures/NetlistEditor_AllButtons.png

Figure 3.12: Buttons on the Netlist Editor.

4. SPICE Analysis Types, Commands and Options

The SPICE engine in CoolSPICE is based on ngspice. The netlist generator creates netlists in the standard SPICE syntax, for which extensive documentation is available online. The Analysis components in the Schematic Generator are translated into the netlist in the syntax set down here for each analysis type, described here. As described in Chapter **??**, these commands can also be directly added to the netlist to run the relevant analyses.

4.1. SPICE Suffixes

These are repeated here for convenience.

suffix	*name*	*magnitude*
a	atto-	10^{-18}
f	femto-	10^{-15}
p	pico-	10^{-12}
n	nano-	10^{-9}
u	micro-	10^{-6}
m	milli-	10^{-3}
k	kilo-	10^{3}
meg	mega-	10^{6}
g	giga-	10^{9}
t	tera-	10^{12}

4.2. Basic SPICE Analysis Types

In the syntax statements of this section, parameters are stated with their name between cornered parentheses (*i.e.* `[]`). Optional parameters are stated between normal parentheses.

4.2.1. The .SPICE Directive

The .Spice Directive component can be used to insert any valid SPICE statement directly into the netlist, including analysis statements. Figure **??** shows a .dc statement specified without using the .dc analysis tool (see Section **??** below) and how the statement is included in the netlist.

The most important use of this component in CoolSPICE is to define SPICE options by inserting an `.options` statement. Section **??** describes the options and their use.

./figures/spice_analysis_figures/SPICEAnalysis_spicedirective1.png

Figure 4.1: The analysis component .Spice Directive used to specify a nested DC analysis.

4.2.2. .ac (AC/Frequency Domain Analysis)

There must be one AC source with a defined value in the circuit for this analysis to run. However, more than one AC source with defined values can make results hard to interpret.

Syntax:

```
.ac dec [POINTSPERDECADE] [START] [END]
.ac oct [POINTSPEROCTAVE] [START] [END]
.ac lin [POINTS] [START] [END]
```

parameter	*description*	*Schematic Editor option*
`[POINTSPERDECADE]`	Number of points within each decade (*) of the frequency range	Dec
`[POINTSPEROCTAVE]`	Number of points within each octave (*) of the frequency range	N/A
`[POINTS]`	Number of points within the frequency range (*)	N/A
`[START]`	Start value for sweep	Start Freq
`[END]`	End value for sweep	End Freq

(*) The AC Run component in the Schematic Editor only implements the decade option; see Remarks below.

Examples:

statement	*explanation*
`.ac dec 30 10k 100meg`	Sweep the AC source frequency from 10 kHz to 100 MHz, 30 points per decade
`.ac lin 20 10meg 20meg`	Sweep the AC source frequency from 10 MHz to 20 MHz in 20 points
`.ac oct 16 2meg 64meg`	Sweep the AC source frequency from 2 MHz to 64 MHz in 16 points per octave

Remarks: Linear and Per Octave Point Separation

The AC Run component in the Schematic Editor only implements the points-per-decade option. The user can implement linearly spaced points or the points-per-octave option with the syntax shown here by using the Netlist Editor.

Further options for an AC simulation can be set through an `.options` statement as described in Section **??**.

4.2.3. .dc (DC Analysis)

Syntax:

```
.dc  [SOURCENAME] [START] [END] [INCREMENT]
     ([SOURCENAME 2] [START 2] [END 2] [INCREMENT 2])
```

parameter	*description*	*Schematic Editor option*
`[SOURCENAME]`	Reference name of the source whose DC value is swept **(*)**	Voltage Source1/ Current Source1
`[START]`	Start value for sweep	V1start/I1start
`[END]`	End value for sweep	V1end/I1end
`[INCREMENT]`	Step size for sweep	V1step/I1step
`[SOURCENAME 2]`	Reference name of the second source for DC sweep **(*)**	Voltage Source2/ Current Source2
`[START 2]`	Start value for second sweep	V1start/I1start
`[END 2]`	End value for second sweep	V1end/I1end
`[INCREMENT 2]`	Step size for second sweep	V1step/I1step

(*) The netlister adds a `"V"` before the Reference Name set in the Options window for a voltage source and a `"I"` for a current source. E.g. a DC voltage source VGATE has the name `VVGATE` in the SPICE statement. The SOURCENAME parameter must have the name as it is in the SPICE statement, i.e. `VVGATE` for this example, not `VGATE`.

Examples:

statement	*explanation*
`.dc IIS1 10u 230u 10u`	Sweep the DC current source "IIS1" **(*)** from 10 μA to 230 μA in steps of 10 μA
`.dc VVS1 0 30 0.1`	Sweep the DC voltage source "VVS1" from 0 V to 30 V in steps of 0.1 V

`.dc IIS1 10u 230u 10u`
`+VVS1 0 3 0.5`
	Step the DC voltage source "VVS1" from 0 V to 3 V in steps of 0.5 V; at each step sweep the DC current source "IIS1" from 10 μA to 230 μA in steps of 10 μA (**)

(*) The source names in all examples here are as in the SPICE statements, not as in the Reference Name boxes; see note above. (**) This statement can only be set in the netlist, not from the Schematic Editor. See "Nested Sweeps" below.

Remarks: Nested Sweeps

Nested sweeps of two voltage sources can be set from the Schematic Editor by specifying both sweeps in the DC Run component. Similarly, nested sweeps of two current sources can be set by specifying both in the DC Run Current component. However, it is not possible to use a component in the Schematic Editor to set a nested voltage/current or current/voltage sweep. A SPICE statement such as the last example given above has to be entered into the netlist for that purpose.

Further options for a DC simulation can be set through an `.options` statement as described in Section **??**.

4.2.4. .op (Operating Point)

Syntax:

```
.op
```

Remarks:

Capacitors are considered open circuit and inductors are considered as closed circuit for the operating point analysis.

The operating point analysis results can be seen in the rawfile by using **SPICE→Preferences→Sh Rawfile**. If the **SPICE→Run OP Analysis** menu item is used, the program will run the operating point analysis and display the results near the meter components in the drawing, but the .op command will not be included in the netlist when it is generated unless the .op component from the Analysis menu is used.

4.2.5. .step (Parametric Analysis)

The `.step` command is not, in itself, an analysis command. It is used in conjunction with another analysis command such as `.op` to run through that analysis multiple times.

Syntax:

```
.step  [PARAMNAME]  [START]  [END]  [INCREMENT]
```

parameter	*description*	*Schematic Editor option*
[PARAMNAME]	*Component Value* to sweep **(*)**	Parameter
[START]	Start value for sweep	Start
[END]	End value for sweep	End
[INCREMENT]	Step size for sweep	Step

(*) To run a parametric sweep, the value of a component parameter must be defined as a parameter name instead of a numerical value. [PARAMNAME] in the `.step` statement must then be the same. See Fig. ?? for an example.

./figures/spice_analysis_figures/SPICEAnalysis_parametric1.png

Figure 4.2: The transistor circuit operating point is evaluated by changing the value of the resistor connected to the drain from 100 Ω to 1000 Ω in 100 Ω steps.

4.2.6. .tran (Transient Analysis)

Syntax:

```
.tran  [STEP] [END] ([START] [MAXSTEP] [UIC])
```

parameter	*description*	*Schematic Editor option*
[STEP]	Time step size for storing results **(*)**	Time Step
[END]	Stop time for simulation	End Time
[START]	Start time for simulation	Start Time
[MAXSTEP]	Maximum time step in the adaptive transient simulation	Max Step
[UIC]	"Use initial conditions" (*)	N/A

💡 **(*) The .tran component in the Schematic Editor does not include a way to specify this option. It should be specified in the netlist by using the Netlist Editor if desired.**

Examples:

statement	*explanation*
`.tran 10n 100u`	Run a transient simulation from t=0 to $t = 100\mu$s, store the results every 10 ns
`.tran 10n 100u 0 5n`	Run a transient simulation from t=0 to $t = 100\mu$s with a maximum time step of 5 ns, store the results every 10 ns
`.tran 10n 100u 50u 5n`	Run a transient simulation from t=0 to $t = 100\mu$s with a maximum time step of 5 ns, store the results every 10 ns starting from $t = 50\mu$s
`.tran 10n 80u uic`	Run a transient simulation from t=0 to $t = 80\mu$s, do not calculate the quiescent operating point beforehand

Remarks:

If "`uic`" is specified at the end of the `.tran` statement, the simulator will not solve for the quiescent operating point before the transient analysis. If there are IC values specified on elements, CoolSPICE will use these values (see Section **??**). If there are none, all node voltages will be assumed to be zero initially. This may lead to problems.

Further options for a transient simulation can be set through an `.options` statement as described in Section **??**.

4.3. Initial Conditions

There are two ways of specifying initial conditions within CoolSPICE:

- The IC - Initial Condition tool in the Analog library. This allows the user to set an initial condition voltage at a given node and corresponds to a .ic statement in the netlist.
- Initial conditions can be set for voltages across capacitors and currents across inductors by editing the device statements in the netlist.

This section describes the first method.

4.3.1. The `.ic` Statement

Syntax:

```
.ic  v([NODELABEL])=[VALUE]
```

parameter	*description*	*Schematic Editor option*
`[NODELABEL]`	Name or label of the node at which the IC is set	N/A, graphical
`[VALUE]`	The initial condition value	N/A

Examples:

statement	*explanation*
`.ic v(Vin)=2`	Set the initial condition on node connected to the wire labeled `Vin` to 2 V
`.ic v(_N_6)=0.5`	Set the IC on the automatically-named node `_N_6` to 0.5 V

4.4. The `.meas` Statement

The `.meas` (or `.measure`) statement is used to take measurements (such as a signal period) on the simulation output data after a simulation is successfully completed. This is a very versatile and powerful statement which serves many of the same purposes as taking measurements on a plotted result after the simulation, but yields results much faster and without user intervention or the need to use a graphical plot. It can be used to measure results from `.ac`, `.dc` or `.tran` statements.

`.measure` statements can be used to measure times elapsed between trigger voltage levels (e.g. period of a periodic signal, or rise/fall/delay times), to measure average, RMS, minimum or maximum values, or values or times when a specific event such as two signals crossing each other occurs. This section of the manual will be expanded in the next edition to describe the full use of the statement.

The example file Meas_example.dsn under the Ckts/ folder in the default CoolSPICE installation shows the .measure command being used. Note that in the Schematic Editor, the `.meas` statement is specified using the .Spice Directive component. To use this method, the Measure option must be checked under the **SPICE→Preferences** menu item. No rawfile is saved during such a run and the Plotter will not be invoked even if the option is set to open it automatically after a simulation.

4.5. The `.param` Statement

With the `.param` statement, the user can define constants or numerical values for an alphanumeric identifier, which can then be referred to throughout the rest of the circuit definition. This option is not available directly through the Schematic Editor (except when using a `.step` analysis; see below) but can be easily used through the Netlist Editor.

Syntax:

```
.param  [IDENTIFIER1]=[VALUE1] ([IDENTIFIER2]=[VALUE2]) (...)
```

parameter	*description*	*Schematic Editor option*
`[IDENTIFIER]`	Name for the identifier	N/A (*)
`[VALUE]`	The value or constant	N/A (*)

💡 **(*) When a `.step` analysis is being used to step through a range of values for a component, it is possible to set the value of the component to a parameter. In that case, the netlister will insert a `.param` statement into the netlist and set its value to the start value of the `.step` statement. Figure ?? shows an example case, for which the netlister will include the line `.param Rr=100` in the netlist.**

Examples:

statement	*explanation*
`.param pi=3.1415`	Set ‘‘pi" to 3.1415.
`.param Rd=200`	Define `Rd` as 200 Ω, to later refer to it in lines such as `Rdrain1 D1 VDD_n Rd` and `Rdrain45 D45 VDD_n Rd`.

Remarks:

The first character of the identifier must be alphabetic. The rest can be alphanumeric and use the ! # $ % [ ] _ characters. The following words are reserved keywords in SPICE and must not be used as parameter names: `abs`, `and`, `arctan`, `cos`, `defined`, `div`, `exp`, `hertz`, `ln`, `mod`, `not`, `or`, `pwr`, `sin`, `sqr`, `sqrt`, `temper`, `time`.

4.5.1. `.param` Functions

A number of functions are available for use in conjunction with .param statements.

Function Name	Description
abs(x)	Absolute value of x
agauss(nom, avar, sigma)	Nominal value plus variation drawn from Gaussian distribution with mean 0 and standard deviation absolute value of avar, divided by sigma
arcsin(x), asin(x), arccos(x), acos(x), arctan(x), atan(x),	Inverse trigonometric functions. Note these functions give the real part only.
asinh(x), acosh(x), atanh(x)	Inverse hyperbolic trigonometric functions. Note these functions give the real part only.
atan2(y, x)	Fourth quadrant inverse tangent of y/x
buf(x)	(x>0.5) ? 1.0 : 0.0
cbrt(x)	Cube root of x
ceil(x)	Smallest integer that is greater than or equal to x

defined(symbol)	1 if symbol is defined, else 0
exp(x)	e^x
flat(x)	Randomly generated number between -x and x with uniform distribution
floor(x), int(x)	Largest integer that is less than or equal to x
gauss(nom, rvar, sigma)	Nominal value plus variation drawn from Gaussian distribution with mean 0 and standard deviation rvar (relative to nominal, divided by sigma
hypot(x, y)	sqrt((x^2) + (y^2))
inv(x)	(x>0.5) ? 0.0 : 1.0
limit(nom, avar)	Nominal value ± avar, depending on random number in [-1, 1] being > 0 or < 0
ln(x), log(x)	Natural log of x
max(x, y)	x > y ? x : y
mc(x, y)	Randomly generated real number y such that x·(1-y) < y < x·(1+y) with uniform distribution
min(x, y)	x < y ? x : y
nint(x), round(x)	Round to nearest int, x.5 rounds to x+1.
pow(x, y)	x raised to the power of y (pow from C runtime library)
pwr(x, y)	pow(fabs(x),y), note that this function gives only the real part of the answer
pwrs(x, y)	sgn(x)·abs(x)^y
rand(x), random(x)	Randomly generated real number y such that 0 < y < 1 depending on the integer value of x
sgn(x)	1.0 for x > 0, 0.0 for x equal to 0, -1.0 for x < 0
sin(x), cos(x), tan(x)	Trigonometric functions

sinh(x), cos(x), tanh(x)	Hyperbolic trigonometric functions
sqr(x)	y=x^2
sqrt(x)	y=$\sqrt{x}$, note that this function gives only the real part of the answer
table(x, x_1, y_1, x_2, y_2, ...)	Piecewise linear function where the answer value y, is interpolated base on the pair of points between which x falls.
ternary_fcn(x, y, z), if(x, y, z)	x ? y : z
u(x)	Unit step, x > 0 ? 1 : 0
unif(nom, rvar)	Nominal value plus relative variation (to nominal) uniformly distributed between ±rvar
uramp(x)	x > 0 ? x : 0

Table 4.1: .param functions

4.6. The `.save` Statement

The `.save` statement is used to save node voltage and branch current values in the output rawfile. The Plotter can display these values as traces. The netlister automatically inserts the relevant `.save` statements in the netlist when the components Probe - Ammeter and Probe - Voltmeter from the Analog library are used in the Schematic Editor (see Section **??**).

💡 **Note that it is only possible to save the current through an independent voltage source. The current probe component in the Schematic Editor is actually a zero-volt DC source.**

Syntax:

```
.save  v([NODELABEL])
.save  i([VOLTAGESOURCEREF])
```

parameter	*description*	*Schematic Editor option*
`[NODELABEL]`	The label or name of a node in the circuit.	N/A
`[VOLTAGESOURCEREF]`	The Reference Name of a voltage source or a current probe, prefixed with a “`V`” if taken from the Schematic Editor Options (*)	Ref (*)

(*) The netlister will prefix the reference name of a voltage source or current probe with “`V`”. For instance, a DC voltage source whose reference name is set as “V_DS” and a current probe whose reference name is set as “Imeter1” in the Options window will appear as “`VV_DS`” and “`VImeter1`” in the netlist respectively. The `[`VOLTAGESOURCEREF`]` parameter in the `.save` statement must be in this prefixed form.

Examples:

statement	*explanation*
`.save v(_N_6)`	Save the voltage at the node “`_N_6`”
`.save v(diffampout)`	Save the voltage at the line labeled “`diffampout`,” which corresponds to a node.
`.save i(VDS1)`	Save the current through the voltage source `VDS1`, which will have the reference name as “DS1” in the Options window if defined through the Schematic Editor.

4.7. SPICE Simulation Options

SPICE simulation options can be set through the `.options` statement, which has to be entered in the netlist through the Netlist Editor. Multiple options can be entered in a single statement.

Syntax:

```
.options [OPTION1] [OPTION2] [...]
.options [OPTION1]=[VALUE] [OPTION2]=[VALUE]
```

In the tables of this section, options which are toggled on- or off- just have names listed, while options which should have a value set are shown with an “equal” sign (*e.g.* `ABSTOL=`).

4.7.1. General Options

option	*description*	*default value*
ACCT	Print runtime statistics	On
NOACCT	Do not print runtime statistics or the initial transient solution	Off
NOINIT	Do not print the initial transient solution	Off
TEMP=	Circuit operating temperature; can be overridden on devices	27 °C
TNOM=	Nominal temperature for device parameter measurements	27 °C

All the options about printing/not printing certain outputs pertain to the .log file created after the simulation. The log file is by default written in the same directory as the design (.dsn) file and shares the file name. The log file display can be invoked automatically after each simulation by using the **SPICE→Preferences** menu item of the Schematic Editor.

4.7.2. DC Simulation Options

type	*option*	*description*	*default value*
Error tolerance			
	ABSTOL=	Limit for the absolute current error tolerance	1 pA
	RELTOL=	Relative error tolerance	0.001
	VNTOL=	Limit for the absolute voltage error tolerance	1 mV
Ensuring convergence			
	GMIN=	Minimum conductance allowed	1e-12
	ITL1=	DC solution number of iterations limit	100
	ITL2=	DC transfer curve number of iterations limit	50
	PIVREL=	Ratio between the largest value in a column and the acceptable pivot value	1e-3
	PIVTOL=	The smallest acceptable pivot value	1e-13
	RSHUNT=	Value for a resistor to be inserted between each node and the ground	N/A
Other			
	KEEPOPINFO	Retain OP analysis information for AC analyses	Off

4.7.3. AC Simulation Options

	option	*description*	*default value*
	NOOPAC	Do not perform the OP analysis before the AC analysis (Invalid if there are nonlinear elements in the circuit)	

4.7.4. Transient Simulation Options

type	*option*	*description*	*default value*
Error tolerance			
	CHGTOL=	Limit for the charge tolerance	1e-14
	TRTOL=	Transient error tolerance	7
Ensuring convergence			
	GMINSTEPS=	Number of gmin steps to be attempted	1
	ITL3=	Lower limit for iteration	4
	ITL4=	Time-point iteration limit	10
	ITL5=	Total iteration limit	5000
	ITL6=	(See SRCSTEPS)	
	RAMPTIME=	Rate of change for independent supplies and passives with ICs	
	SRCSTEPS=	(If set to non-zero) Use a source-stepping method to solve for the OP	0
Integration			
	MAXORD=	Maximum order for the numerical integration method (see METHOD). (*)	2
	METHOD=	The numerical integration method (gear or trapezoidal)	trap
Other			
	AUTOSTOP	Stop the analysis after all .meas commands have been executed (see Section **??**)	Off

💡 **These options will not affect the device models, but only the device instances. The relevant option has to be set in the device line in the netlist for the reset or scale values here to be effective. If these are set in an `.options` line, they override the relevant parameter in all devices.**

4.7.5. Element-Specific Options

option	*description*	*default value*
`DEFAD=`	Define `AD`, the MOS drain diffusion area	0.0
`DEFAS=`	Define `AS`, the MOS source diffusion area	0.0
`DEFL=`	Define `L`, the MOS channel length	100 μm
`DEFW=`	Define `W`, the MOS channel width	100 μm
`SCALE=`	Element scaling factor for dimensional parameters with a default unit of meters (*)	1
`TRYTOCOMPACT`	(Applies only to transmission lines using the `LTRA` model) Attempt to condense past input voltage/current history.	OFF

(*) This applies to MOSFET parameters `AD, AS, L, PD, PS, SA, SB, SC, SD` and `W`, to diode parameters `Area, W, L`, to resistor and capacitor parameters `L` and `W`, and to JFET/MESFET parameters `Area, W` and `L`. If, for instance, a MOSFET length is defined as `L=2.5` and the line `.options SCALE=1u` is given, the MOSFET length in simulation will be 2.5 μm.

1. NGSPICE: Mixed Mode - Mixed Level Circuit Simulator, http://ngspice.sourceforge.net, last visited October 2014.

2. List of SPICE statements, http://www.ecircuitcenter.com/SPICEsummary.htm#statements, last visited October 2014.

5. SPICE Circuit Elements and Device Models

5.1. SPICE Suffixes

These suffixes are used in specifying parameter values for elements.

suffix	*name*	*magnitude*
a	atto-	10^{-18}
f	femto-	10^{-15}
p	pico-	10^{-12}
n	nano-	10^{-9}
u	micro-	10^{-6}
m	milli-	10^{-3}
k	kilo-	10^{3}
meg	mega-	10^{6}
g	giga-	10^{9}
t	tera-	10^{12}

5.2. SPICE Element Identifiers

The SPICE netlist line for each type of circuit element must start with a specific character. Here we list the characters all together; examples are presented throughout the rest of this chapter for each element type.

character	*element type*
`R`	Resistors (Section **??**)
`C`	Capacitors (Section **??**)
`L`	Inductors (Section **??**)
`K`	Coupled Inductors (Section **??**)
`V`	Independent Voltage Sources (Section **??**)
`I`	Independent Current Sources (Section **??**)
`G`	Voltage-Controlled Current Sources (Section **??**)
`E`	Voltage-Controlled Voltage Sources (Section **??**)
`F`	Current-Controlled Current Sources (Section **??**)
`H`	Current-Controlled Voltage Sources (Section **??**)
`B`	Nonlinear Dependent (Behavioral) Sources (Section **??**)
`D`	Diodes (Section **??**)
`Q`	BJTs (Section **??**)
`M`	MOSFETs (Section **??**)
`Z`	MESFETs (Section **??**)
`J`	JFETs (Section **??**)
`S, W`	Switches (Section **??**)
`T`	Transmission Lines (Section **??**)

5.3. Passive Elements

5.3.1. RXX : Resistors

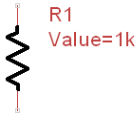
Syntax:

```
RXX  [N+] [N-] [VAL] [INLINE PARAMS]
```

Inline params are optional

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[VAL]	Resistance
[INLINE PARAMS]	*Inline parameters :* m : Current multiplication factor ac : AC value scale : Current scale factor temp : Temperature dtemp : Temperature difference with ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient noisy : Noise on/off

Schematics Editor Library:
Analog
Schematics Editor Symbol:



Schematics Editor Option:
"Value" is for assigning an alphanumeric number to [VAL]
Examples:

statement	*explanation*
`r1 1 0 1K`	Insert resistor r1, with resistance $1K\Omega$, between nets 1 and 0
`r1 1 0 1K m=2 scale=3`	Insert resistor r1, with resistance $1K\Omega\times \frac{3}{2}$, between nets 1 and 0

Remarks:

"Value" can be set to a model as described next by the alternative syntax. These alternatives can be used by editing the netlist using the netlist editor.

Alternative Syntax 1:

```
RXX [N+] [N-] [VAL] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] R [MODEL PARAMS]
```

parameter	*description*
`[N+]`	Positive node/net
`[N-]`	Negative node/net
`[VAL]`	Resistance
`[MODEL]`	Resistor model defined by a .model statement
`[INLINE PARAMS]`	*Inline parameters :* m : Current multiplication factor ac : AC value scale : Current scale factor temp : Temperature dtemp : Temperature difference with ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient noisy : Noise on/off
`[MODEL PARAMS]`	*Model parameters :* m : Current multiplication factor ac : AC value scale : Current scale factor temp : Temperature dtemp : Temperature difference with ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient noisy : Noise on/off

Example:

statement	*explanation*

`r1 1 0 1K res tc1=0.01`
`.model res R m=2`
	Insert resistor r1, with resistance 1K$\Omega\times\frac{1}{2}$, between nets 1 and 0

If it is specified for an instance, [INLINE PARAMS] override [MODEL PARAMS].

Alternative Syntax 2: Semiconductor resistor

```
RXX [N+] [N-] [VAL] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] R [MODEL PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[VAL]	Resistance
[MODEL]	Resistor model defined by a .model statement
[INLINE PARAMS]	*Inline parameters :*
	l : Length
	w : Width
	m : Current multiplication factor
	ac : AC value
	scale : Current scale factor
	temp : Temperature
	dtemp : Temperature difference with ambient
	noisy : Noise on/off
[MODEL PARAMS]	*Model parameters :*
	tc1 : Linear temperature coefficient
	tc2 : Quadratic temperature coefficient
	rsh : Sheet resistance
	defw : Default width
	narrow : Width narrowing value
	short : Length shortening value
	tnom : Nominal temperature
	kf : Flicker noise coefficient
	af : Flicker noise exponent
	r (res) : Default value

Example:

statement	*explanation*

`r1 1 0 res l=2u w=1u`
`.model res R (rsh=10`
`+narrow=0.1u short=0.05u)`
	Insert resistor r1, with resistance $10\times\frac{2u-0.05u}{1u-0.1u}\Omega$, between nets 1 and 0

If [VAL] is specified, it will override resistance calculated using the geometrical effects.

Remarks:

If resistance is not specified, but tc1 (default : 0), tc2 (default : 0), rsh (no default), short (default : 0), and narrow (default : 0) are specified, the semiconductor resistance is calculated using the circuit temperature T as follows:

$$\begin{aligned} \mathrm{r} &= \mathrm{r_o} \times (1 + \mathrm{tc1} \times (\mathrm{T} - \mathrm{tnom}) + \mathrm{tc2} \times (\mathrm{T} - \mathrm{tnom})^2) \\ \mathrm{r_o} &= \mathrm{rsh} \times \frac{\mathrm{l} - \mathrm{short}}{\mathrm{w} - \mathrm{narrow}} \end{aligned}$$

💡 **If rsh or l are not specified, r is set to 1mΩ.**

Alternative Syntax 3: Behavioral resistor

```
RXX  [N+] [N-] R= [EXPRESSION] [INLINE PARAMS]
```

parameter	*description*
`[N+]`	Positive node/net
`[N-]`	Negative node/net
`[EXPRESSION]`	An equation or expression containing voltages or currents
`[INLINE PARAMS]`	*Inline parameters :* tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient

Example:

statement	*explanation*
`r1 1 0 r=v(2)*0.1`	Insert resistor r1, with resistance v(2)×0.1Ω, between nets 1 and 0

💡 **Expression needs to be enclosed with {...} or '...'.**

5.3.2. CXX : Capacitors

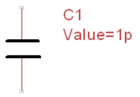
Syntax:

```
CXX  [N+] [N-] [VAL] [INLINE PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[VAL]	Capacitance
[INLINE PARAMS]	*Inline parameters :* m : Current multiplication factor scale : Current scale factor temp : Temperature dtemp : Temperature difference with ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient ic : Initial condition rser : Series resistance lser : Series inductance rpar : Parallel resistance cpar : Parallel capacitance rlshunt : Parallel resistance to lser

rser, lser, rpar, cpar and rlshunt are incorporated for Ltspice compatibility.

Schematics Editor Library:
Analog
Schematics Editor Symbol:



Schematics Editor Option:
"Value" is for assigning an alphanumeric number to [VAL]
Examples:

statement	*explanation*
c1 1 0 1p	Insert capacitor c1, with capacitance 1pF, between nets 1 and 0

`c1 1 0 1K m=2 scale=3`	Insert capacitor c1, with capacitance 1pF ×2×3, between nets 1 and 0

Remarks:

"Value" can be set to a model as described next by the alternative syntax. These alternatives can be used by editing the netlist using the netlist editor.

Default gmin is 1e-12

CoolSPICE automatically inserts a resistor with resistance 1/gmin across each capacitor to increase stability.

Alternative Syntax 1:

```
CXX [N+] [N-] [VAL] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] C [MODEL PARAMS]
```

parameter	*description*
`[N+]`	Positive node/net
`[N-]`	Negative node/net
`[VAL]`	Capacitance
`[MODEL]`	Capacitance model defined by a .model statement
`[INLINE PARAMS]`	*Inline parameters :* m : Current multiplication factor scale : Current scale factor temp : Temperature dtemp : Temperature difference with ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient ic : Initial condition rser : Series resistance lser : Series inductance rpar : Parallel resistance cpar : Parallel capacitance rlshunt : Parallel resistance to lser
`[MODEL PARAMS]`	*Model parameters :* cap : Capacitance m : Current multiplication factor scale : Current scale factor temp : Temperature dtemp : Temperature difference with ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient ic : Initial condition

Example:

statement	*explanation*

`c1 1 0 1p cap tc1=0.01`
`.model cap C m=2`
	Insert capacitor c1, with capacitance 1pF ×2, between nets 1 and 0

.ic applies only if uic is used in conjunction with .tran.

Alternative Syntax 2: Semiconductor capacitor

```
CXX [N+] [N-] [VAL] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] C [MODEL PARAMS]
```

parameter	*description*
`[N+]`	Positive node/net
`[N-]`	Negative node/net
`[VAL]`	Capacitance
`[MODEL]`	Resistor model defined by a .model statement
`[INLINE PARAMS]`	*Inline parameters :* l : Length w : Width m : Current multiplication factor scale : Current scale factor temp : Temperature dtemp : Temperature difference with ambient ic : Initial condition rser : Series resistance lser : Series inductance rpar : Parallel resistance cpar : Parallel capacitance rlshunt : Parallel resistance to lser

[MODEL PARAMS]	*Model parameters :* cap : Default value cj : Bottom/areal junction capacitance cjsw : Sidewall/perimeter junction capacitance defw : Default width defl : Default length defw : Default width narrow : Width narrowing value short : Length shortening value tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient tnom : Nominal temperature di : Relative epsilon thick : Dielectric thickness

Example:

statement	*explanation*

`c1 1 0 cap l=2u w=1u`
`.model cap C (cj=1u cjsw=1p)`
	Insert capacitor c1, with capacitance 1uF/m$^2 \times \mathrm{l} \times \mathrm{w}$ + 1pF/m $\times 2 \times \mathrm{l} \times \mathrm{w}$, between nets 1 and 0

Remarks:

If cap is not specified, but tc1 (default : 0), tc2 (default : 0), thick (default : 0), short (default : 0), narrow (default : 0), and di (default : 3.9) are specified, the semiconductor capacitance is calculated using the circuit temperature T as follows:

$$
\begin{aligned}
\mathrm{c} &= \mathrm{c_o} \times (1 + \mathrm{tc1} \times (\mathrm{T} - \mathrm{tnom}) + \mathrm{tc2} \times (\mathrm{T} - \mathrm{tnom})^2) \\
\mathrm{c_o} &= \mathrm{cj} \times (\mathrm{l} - \mathrm{short}) + 2 \times \mathrm{cjsw} \times (\mathrm{l} - \mathrm{short} + \mathrm{w} - \mathrm{narrow}) \\
\mathrm{cj} &= \frac{\mathrm{cj} \times \epsilon_{\mathrm{o}}}{\mathrm{thick}}
\end{aligned}
$$

If [VAL] is specified, it will override capacitance calculated using the geometrical effects.

Alternative Syntax 3: Behavioral capacitor

```
CXX  [N+] [N-] C= [EXPRESSION] [INLINE PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net

[EXPRESSION]	An equation or expression containing voltages or currents
[INLINE PARAMS]	*Inline parameters :* tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient rser : Series resistance lser : Series inductance rpar : Parallel resistance cpar : Parallel capacitance rlshunt : Parallel resistance to lser

Example:

statement	*explanation*
`c1 1 0 c=v(2)*0.1`	Insert capacitor c1, with capacitance v(2)×0.1F, between nets 1 and 0

5.3.3. LXX : Inductors

Syntax:

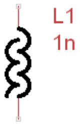
```
LXX  [N+] [N-] [VAL] [INLINE PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[VAL]	Inductance
[INLINE PARAMS]	*Inline parameters :* nt : Number of turns m : Current multiplication factor scale : Current scale factor temp : Temperature dtemp : Temperature difference with ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient ic : Initial condition rser : Series resistance rpar : Parallel resistance cpar : Parallel capacitance

rser, rpar, and cpar are incorporated for Ltspice compatibility.

Schematics Editor Library:
Analog

Schematics Editor Symbol:



Schematics Editor Option:
"Value" is for assigning an alphanumeric number to [VAL]

Examples:

statement	*explanation*
l1 1 0 1n	Insert inductor l1, with inductance 1nH, between nets 1 and 0
l1 1 0 1n m=2	Insert inductor l1, with inductance 1nF ×2, between nets 1 and 0

Remarks:

"Value" can be set to a model as described next by the alternative syntax. These alternatives can be used by editing the netlist using the netlist editor.

Default gmin is 1e-12

CoolSPICE automatically inserts a resistor with resistance 1/gmin across each inductor to increase stability.

CoolSPICE automatically inserts a 1 mΩ resistor in series with each inductor to increase stability. This can be disabled locally by setting inline parameter 'rser' to zero.

Alternative Syntax 1:

```
LXX [N+] [N-] [VAL] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] L [MODEL PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[VAL]	Capacitance
[MODEL]	Capacitance model defined by a .model statement
[INLINE PARAMS]	*Inline parameters :* m : Current multiplication factor scale : Current scale factor nt : Number of turns temp : Temperature dtemp : Temperature difference with ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient ic : Initial condition rser : Series resistance rpar : Parallel resistance cpar : Parallel capacitance

[MODEL PARAMS]	*Model parameters :* ind : Capacitance csect : Cross section length : Length tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient tnom : Nominal temperature nt : Number of turns mu : Relative μ

Example:

statement	*explanation*

`l1 1 0 1n myind tc1=0.01`
`.model myind L m=2`
	Insert inductor l1, with inductance 1nH ×0.5, between nets 1 and 0

.ic applies only if uic is used in conjunction with .tran.

Remarks:

If ind is not specified, but tc1 (default : 0), tc2 (default : 0), nt (default : 0), csect (default : 0), mu (default : 1), and length (default : 0) are specified, the inductance is calculated using the circuit temperature T as follows:

$$\begin{aligned} \mathrm{l} &= \mathrm{l_o} \times (1 + \mathrm{tc1} \times (\mathrm{T} - \mathrm{tnom}) + \mathrm{tc2} \times (\mathrm{T} - \mathrm{tnom})^2) \\ \mathrm{l_o} &= \frac{\mathrm{mu} \times \mu_\mathrm{o} \times \mathrm{nt}^2 \times \mathrm{csect}}{\mathrm{length}} \end{aligned}$$

Alternative Syntax 3: Behavioral inductor

```
LXX  [N+] [N-] C= [EXPRESSION] [INLINE PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[EXPRESSION]	An equation or expression containing voltages or currents
[INLINE PARAMS]	*Inline parameters :* tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient rser : Series resistance rpar : Parallel resistance cpar : Parallel capacitance

💡 **If [VAL] is specified, it will override inductance calculated using the geometrical effects.**

Example:

statement	*explanation*
`l1 1 0 l=v(2)*0.1`	Insert inductor l1, with inductance v(2)×0.1H, between nets 1 and 0

5.3.4. KXX : Coupled Inductors

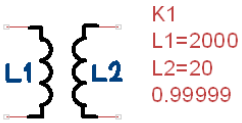
Syntax:

```
KXX  [LYY] [LZZ] [VAL]
```

parameter	*description*
[LYY]	First of the coupled inductor
[LZZ]	Second of the coupled inductor
[VAL]	Coupling value

💡 **Coupling value has to be between 0 and 1.**

Schematics Editor Library:
Analog
Schematics Editor Symbol:



Schematics Editor Option:
"Value" is for assigning an alphanumeric number to [VAL]
Examples:

statement	*explanation*
k1 l1 l2 0.99	Insert coupled inductor k1, with coupling 0.99, between nets 1 and 0

5.4. Sources

5.4.1. VXX / IXX : Independent Sources

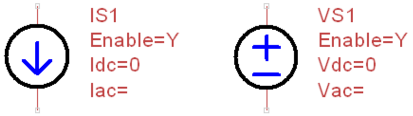
Syntax:

```
VXX  [N+] [N-] [DC] [AC] [DISTOF1] [DISTOF2] [FUNCTION]
IXX  [N+] [N-] [DC] [AC] [DISTOF1] [DISTOF2] [FUNCTION]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[DC]	DC offset during a DC or transient run
	AC parameters :
[AC]	ACMAG : AC magnitude
	ACPHASE : AC phase
	1^{st} distortion parameters :
[DISTOF1]	F1MAG : Distortion frequency
	F1PHASE : Distortion phase
	2^{nd} distortion parameters :
[DISTOF2]	F2MAG : Distortion frequency
	F2PHASE : Distortion phase
	Temporal functions :
	Pulse
	Exponential
	Sinusoidal
[FUNCTION]	Piecewise linear
	Single frequency FM
	AM
	Transient noise
	Random voltages or currents

Schematics Editor Library:
Analog

Schematics Editor Symbol:



Schematics Editor Option:

"Idc / Vdc" is for assigning an alphanumeric value to [DC]. "Iac / Vac" is for assigning an alphanumeric value to [AC].

Examples:

statement	*explanation*
`v1 1 0 1 0.1`	Insert voltage source v1, with 1V DC and 0.1V AC, between nets 1 and 0
`i1 1 0 1 0.1`	Insert current source i1, with 1V DC and 0.1V AC, between nets 1 and 0
`i1 1 0 sin(0 1 1meg)`	Insert sinusoidal current source i1, with 1V magnitude and 1MHz frequency, between nets 1 and 0

Pulse Waveform Syntax:

```
PULSE ( [V1] [V2] [TD] [TR] [TF] [PW] [PER] )
```

parameter	*description*
[V1]	Initial pulse value
[V2]	Pulsed value
[TD]	Delay time
[TR]	Rise time
[TF]	Fall time
[PW]	Pulse width
[PER]	Period

Closed Form:

$$V(t) = \begin{cases} \mathrm{V1} & \text{if } 0 \leq \mathrm{t} < \mathrm{TD} \\ \mathrm{V1} + (\mathrm{V2} - \mathrm{V1})\frac{\mathrm{t}-\mathrm{TD}}{\mathrm{TR}} & \text{if } \mathrm{TD} \leq \mathrm{t} < \mathrm{TD} + \mathrm{TR} \\ \mathrm{V2} & \text{if } \mathrm{TD} + \mathrm{TR} \leq \mathrm{t} < \mathrm{TD} + \mathrm{TR} + \mathrm{PW} \\ \mathrm{V2} + (\mathrm{V1} - \mathrm{V2})\frac{\mathrm{t}-\mathrm{TD}-\mathrm{TR}-\mathrm{PW}}{\mathrm{TF}} & \text{if } \mathrm{TD} + \mathrm{TR} + \mathrm{PW} \leq \mathrm{t} < \mathrm{TD} + \mathrm{TR} + \mathrm{PW} + \mathrm{TF} \\ \mathrm{V1} & \text{if } \mathrm{TD} + \mathrm{TR} + \mathrm{PW} + \mathrm{TF} \leq \mathrm{t} < \mathrm{PER} \end{cases}$$

Examples:

statement	*explanation*

`v1 1 0 pulse(0 1 1n 2n 3n 20n 50n)`	Insert pulsed voltage source v1 between nets 1 and 0. Pulse source generates a pulse train. The start is delayed by 1ns. The initial DC value is set to the t=0 value. v(1) voltage changes from 0 to 1V between t=1ns and t=3ns. It stays at 1V between t=3ns and t=23ns. It then falls from 1V to 0 between t=23ns and t=26ns. It stays at 0 until t=51ns. 1ns shifted (delayed canceled) version of this initial waveform repeats itself until simulation end time.

Sinusoidal Waveform Syntax:

```
SIN ( [VO] [VA] [FREQ] [TD] [THETA] )
```

parameter	*description*
`[VO]`	Voltage offset
`[VA]`	Sinusoidal wave amplitude
`[FREQ]`	Sinusoidal wave frequency
`[TD]`	Delay time
`[THETA]`	Damping factor

Closed Form:

$$V(t) = \begin{cases} \mathrm{VO} & \text{if } 0 \leq \mathrm{t} < \mathrm{TD} \\ \mathrm{VO} + \mathrm{VA}\exp(-(\mathrm{t}-\mathrm{TD})\mathrm{THETA})\sin(2\pi\mathrm{FREQ}(\mathrm{t}-\mathrm{TD})) & \text{if } \mathrm{TD} \leq \mathrm{t} < \mathrm{TSTOP} \end{cases}$$

Examples:

statement	*explanation*
`v1 1 0 sin(0 1 1meg 0 0)`	Insert sinusoidal voltage source v1 between nets 1 and 0. The initial DC value is set to the t=0 value. v(1) voltage sinusoidally changes from -1V to 1V with 1MHz frequency.

Exponential Waveform Syntax:

```
EXP ( [V1] [V2] [TD1] [TAU1] [TD2] [TAU2] )
```

parameter	*description*

[V1]	Initial value
[V2]	Pulsed value
[TD1]	Rise delay
[TAU1]	Rise time constant
[TD2]	Fall delay
[TAU2]	Fall time constant

Closed Form:

$$V(t) = \begin{cases} \mathrm{V1} & \text{if } 0 \le \mathrm{t} < \mathrm{TD1} \\ \mathrm{V1} + (\mathrm{V2} - \mathrm{V1})(1 - \mathrm{e}^{-\frac{\mathrm{t-TD}}{\mathrm{TAU1}}}) & \text{if } \mathrm{TD1} \le \mathrm{t} < \mathrm{TD2} \\ \mathrm{V1} + (\mathrm{V2} - \mathrm{V1})(1 - \mathrm{e}^{-\frac{\mathrm{t-TD1}}{\mathrm{TAU1}}}) + (\mathrm{V1} - \mathrm{V2})(1 - \mathrm{e}^{-\frac{\mathrm{t-TD2}}{\mathrm{TAU2}}}) & \text{if } \mathrm{TD2} \le \mathrm{t} < \mathrm{TSTOP} \end{cases}$$

Examples:

statement	*explanation*
`v1 1 0 exp(0 1 0 6n 20n 6n)`	Insert exponential voltage source v1 between nets 1 and 0. The initial DC value is set to the t=0 value.

Piecewise Linear Waveform Syntax:

```
PWL ( [T1] [V1] [T2] [V2] [T3] [V3] [T4] [V4] ... ) [INLINE PARAMS]
```

parameter	*description*
[V1] [V2] ...	Voltage points
[T1] [T2] ...	Time points
	Inline parameters :
[INLINE PARAMS]	r : Repeat interval from r to the last [TN]
	td : Delay

FM Waveform Syntax:

```
SFFM ( [VO] [VA] [MDI] [FS] )
```

parameter	*description*
[VO]	Offset
[VA]	Amplitude
[FC]	Carrier frequency

[MDI]	Modulation
[FS]	Signal frequency

Closed Form:

$$V(t) = \mathrm{VO} + \mathrm{VA}\sin(2\pi\mathrm{FCt} + \mathrm{MDI}\sin(2\pi\mathrm{FSt}))$$

5.4.2. GXX / EXX / FXX / HXX : Linear Dependent Sources

GXX : Voltage Controlled Current Source (VCCS):

```
GXX [N+] [N-] [NC+] [NC-] [VAL]
```

parameter	*description*
[N+]	Positive current node/net
[N-]	Negative current node/net
[NC+]	Positive controlling voltage node/net
[NC-]	Negative controlling voltage node/net
[VAL]	Transconductance

Closed Form:

$$\mathrm{I}(\mathrm{N}+ \rightarrow \mathrm{N}-) = \mathrm{VAL} \times (\mathrm{V}(\mathrm{NC}+) - \mathrm{V}(\mathrm{NC}-))$$

Examples:

statement	*explanation*
g1 1 0 2 0 2	Insert a current source between nets 1 and 0, and set its value to twice the voltage between 2 and 0.

EXX : Voltage Controlled Voltage Source (VCVS):

```
EXX [N+] [N-] [NC+] [NC-] [VAL]
```

parameter	*description*
[N+]	Positive voltage node/net
[N-]	Negative voltage node/net
[NC+]	Positive controlling voltage node/net
[NC-]	Negative controlling voltage node/net
[VAL]	Gain

Closed Form:

$$(\mathrm{V}(\mathrm{N}+) - \mathrm{V}(\mathrm{N}-)) = \mathrm{VAL} \times (\mathrm{V}(\mathrm{NC}+) - \mathrm{V}(\mathrm{NC}-))$$

Examples:

statement	*explanation*
`e1 1 0 2 0 2`	Insert a voltage source between nets 1 and 0, and set its value to twice the voltage between 2 and 0.

FXX : Current Controlled Current Source (CCCS):

```
FXX [N+] [N-] [VNAM] [VAL]
```

parameter	*description*
[N+]	Positive current node/net
[N-]	Negative current node/net
[VNAM]	Voltage source whose current is mirrored
[VAL]	Gain

Closed Form:

$$\mathrm{I}(\mathrm{N}+ \rightarrow \mathrm{N}-) = \mathrm{VAL} \times \mathrm{I}(\mathrm{VNAM})$$

Examples:

statement	*explanation*
`f1 1 0 v1 2`	Insert a current source between nets 1 and 0, and set its value to twice the current flowing on v1.

HXX : Current Controlled Voltage Source (CCVS):

```
HXX [N+] [N-] [VNAM] [VAL]
```

parameter	*description*
[N+]	Positive voltage node/net
[N-]	Negative voltage node/net
[VNAM]	Voltage source whose current is mirrored
[VAL]	Transresistance

Closed Form:

$$(\mathrm{V}(\mathrm{N}+) - \mathrm{V}(\mathrm{N}-)) = \mathrm{VAL} \times \mathrm{I}(\mathrm{VNAM})$$

Examples:

statement	*explanation*
`h1 1 0 v1 2`	Insert a voltage source between nets 1 and 0, and set its value to twice the current flowing on v1.

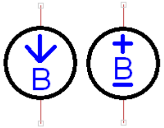
5.4.3. BXX : Behavioral Sources

Syntax:

```
BXX  [N+] [N-] [(I/V)] = [EXPRESSION] [INLINE PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[I/V]	Current/Voltage
[EXPRESSION]	Expression describing current/voltage
[INLINE PARAMS]	*Inline parameters :* temp : Temperature dtemp : Temperature difference from ambient tc1 : Linear temperature coefficient tc2 : Quadratic temperature coefficient rpar : Parallel resistance

Schematics Editor Library:
Analog
Schematics Editor Symbols:



Schematics Editor Option:
"I/V" used to describe the current/voltage of the source with an expression [I/V]

One can define the Current OR the voltage of a behavioral source but not both!

Examples:

Statement	Explanation
B1 0 1 I=cos(v(1))+sin(v(2))	Insert current source B1, with current equal to the expression cos(v(1))+sin(v(2))
B2 0 1 V=ln(cos(log(v(1,2)^2)))-v(2)^v(1)	Insert voltage source B2, with voltage equal to the expression V=ln(cos(log(v(1,2)^2)))-v(2)$^{v(1)}$

B3 0 1 V=V(1) < {Vlow} ? {Vlow} : V(1) > {Vhigh} ? {Vhigh} : V(1)	Insert voltage source B3, with voltage equal to the voltage of net/node 1 bounded by parameters 'Vlow' and 'Vhigh'
Bfib 0 1 V=pwl(time, 0,1, 1,1, 2,2, 3,3, 4,5, 5,8, 6,13, 7,21)	Insert voltage source Bfib, with voltage equal to the Fibonacci series as a function of time.

Available functions for behavioral sources:

Function Name	Description
abs(x)	Absolute value of x
absdelay(x, del), delay(x, del)	Value of x del amount of independent units before the current value of the independent variable. If the independent variable is less than the delay parameter del zero.
arcsin(x), asin(x), arccos(x), acos(x), arctan(x), atan(x)	Inverse trigonometric functions. Note these functions give the real part only.
asinh(x), acosh(x), atanh(x)	Inverse hyperbolic trigonometric functions. Note these functions give the real part only.
atan2(y, x)	Fourth quadrant inverse tangent of y/x
buf(x)	(x>0.5) ? 1.0 : 0.0
ceil(x)	Smallest integer that is greater than or equal to x
ddt(x)	derivative of x against the independent variable
exp(x)	e^x
floor(x), int(x)	Largest integer that is less than or equal to x
hypot(x, y)	sqrt((x^2) + (y^2))
idt(x[,o[,r]]), sdt(x[,o[r]])	Integral of x against the independent variable. 'o' is an optional offset parameter. 'r' is an option reset parameter. If 'r' is true the integral resets to 'o'.

idtmod(x[,ic[,mod[,off]]]),	Integral of x against the independent variable. 'ic' is an optional initial condition parameter. mod is an optional parameter, the function is reset on reaching mod. 'off' is an optional offset which is added to the output of the function. The effect is to bound the output of the function between zero + offset and 'mod' + offset.
inv(x)	(x>0.5) ? 0.0 : 1.0
limit(x, y, z)	If $y \leq x$ then x, If $x < y < z$ then y, if $x < y$ and $z \leq y$ then z.
ln(x), log(x)	Natural log of absolute value of x, causes error if x is zero
log10(x)	Log base ten of absolute value of x, causes error if x is zero
max(x, y)	x > y ? x : y
min(x, y)	x < y ? x : y
pow(x, y)	x raised to the power of y (pow from C runtime library)
pwr(x, y)	pow(fabs(x),y), note that this function gives only the real part of the answer
pwrs(x, y)	sgn(x)·abs(x)^{y}
pwl(x, x_1, y_1, x_2, y_2...), table(x, x_1, y_1, x_2, y_2...)	Piecewise linear function where the answer value y, is interpolated base on the pair of points between which x falls. Note that the x values of the points must increase monotonically.
rand(x), random(x)	Randomly generated real number y such that $0 < y < 1$ depending on the integer value of x
round(x)	Round to nearest integer, x.5 rounds up.
sgn(x)	1.0 for x > 0, 0.0 for x equal to 0, -1.0 for x < 0
sin(x), cos(x), tan(x)	Trigonometric functions
sinh(x), cosh(x), tanh(x)	Hyperbolic trigonometric functions

sqrt(x)	y=$\sqrt{x}$, note that if x is less than zero this function returns the square root of the absolute value of x
ternary_fcn(x, y, z), if(x, y, z), x ? y : z	x ? y : z, (read as "if x then y else z") note the question mark must be followed by a blank space for the parser
u(x)	Unit step, x > 0 ? 1 : 0
unif(nom, rvar)	Nominal value plus relative variation (to nominal) uniformly distributed between ±rvar
uramp(x)	x > 0 ? x : 0
white(x)	Randomly generated number y such that -0.5 < y < 0.5

Table 5.45: Behavioral source functions

Available logical operators for behavioral sources: $! =, <>, >=, <=, ==, >, <, ||, \&\&, !$

5.4.4. Special Behavioral Source Variables time, temper, temp, hertz

'time', 'temper' and 'temp' are available in transient for the simulation time and temperature. For example "B1 1 0 v=sin($\{\mathrm{time}\}$)" produces a sinusoidal voltage source. 'hertz' is available in AC analyses but may slow the simulation.

5.5. Active Elements

5.5.1. DXX : Diodes

Syntax :

```
DXX [N+] [N-] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] D [MODEL PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[MODEL]	Diode model defined by a .model statement
[INLINE PARAMS]	*Inline parameters :* m : Current multiplication factor pj : Perimeter scale factor off : Initial condition for diode state ic : Initial condition temp : Temperature dtemp : Temperature difference with ambient

[MODEL PARAMS]	*description of model parameters :*
is	Saturation current
js	Saturation current
jsw	Sidewall Saturation current
tnom	Parameter measurement temperature
tref	Parameter measurement temperature
rs	Ohmic resistance
trs	Ohmic resistance 1st order temp. coeff.
trs1	Ohmic resistance 1st order temp. coeff.
trs2	Ohmic resistance 2nd order temp. coeff.
n	Emission Coefficient
tt	Transit Time
ttt1	Transit Time 1st order temp. coeff.
ttt2	Transit Time 2nd order temp. coeff.
cjo	Junction capacitance
cj0	Junction capacitance
cj	Junction capacitance
vj	Junction potential
pb	Junction potential
m	Grading coefficient

mj	Grading coefficient
tm1	Grading coefficient 1st temp. coeff.
tm2	Grading coefficient 2nd temp. coeff.
cjp	Sidewall junction capacitance
cjsw	Sidewall junction capacitance
php	Sidewall junction potential
mjsw	Sidewall Grading coefficient
ikf	Forward Knee current
ik	Forward Knee current
ikr	Reverse Knee current
tlev	Diode temperature equation selector
tlevc	Diode temperature equation selector
eg	Activation energy
xti	Saturation current temperature exp.
cta	Area junction temperature coefficient
ctc	Area junction capacitance temperature coefficient
ctp	Perimeter junction capacitance temperature coefficient
kf	Flicker noise coefficient
af	Flicker noise exponent
fc	Forward bias junction fit parameter
fcs	Forward bias sidewall junction fit parameter
bv	Reverse breakdown voltage
ibv	Current at reverse breakdown voltage
tcv	Reverse breakdown voltage temperature coefficient
cond	Ohmic conductance

Diode Equations:

Temperature Dependent Terms:

Initial variables :

$$\mathrm{T} = \mathrm{CKTtemp} + \mathrm{dtemp}$$

$$\mathrm{dt} = \mathrm{T} - \mathrm{Tnom}$$

$$\mathrm{vt} = \frac{\mathrm{k} \times \mathrm{T}}{\mathrm{q}}$$

$$\mathrm{vtnom} = \frac{\mathrm{k} \times \mathrm{Tnom}}{\mathrm{q}}$$

$$\overline{\mathrm{mj}} = \mathrm{mj} \times (1 + \mathrm{tm1} \times \mathrm{dt} + \mathrm{tm2} \times \mathrm{dt}^2)$$

$$\overline{\mathrm{Eg}} = 1.16 - \frac{7.02 \times 10^{-4} \times \mathrm{T}^2}{\mathrm{T} + 1108}$$

$$\overline{\mathrm{pb}} = -2\mathrm{vt} \times \left(1.5 \log\left(\frac{\mathrm{T}}{300.15}\right) + \mathrm{q} \times \left(-\frac{\overline{\mathrm{Eg}}}{2\mathrm{kT}} + \frac{1.1150877}{2\mathrm{k} \times 300.15}\right)\right)$$

$$\overline{\mathrm{Eg}}1 = 1.16 - \frac{7.02 \times 10^{-4} \times \mathrm{Tnom}^2}{\mathrm{Tnom} + 1108}$$

$$\overline{\mathrm{pb}}1 = -2\mathrm{vtnom} \times \left(1.5 \log\left(\frac{\mathrm{Tnom}}{300.15}\right) + \mathrm{q} \times \left(-\frac{\overline{\mathrm{Eg}}1}{2\mathrm{kT}} + \frac{1.1150877}{2 \times 300.15}\right)\right)$$

Junction capacitance variables :
if tlevc is equal to 0 then

$$\mathrm{pbo} = \frac{\mathrm{vj} - \overline{\mathrm{pb}}1}{\frac{\mathrm{Tnom}}{300.15}}$$

$$\mathrm{gmaold} = \frac{\mathrm{vj} - \mathrm{pbo}}{\mathrm{pbo}}$$

$$\overline{\mathrm{tcj}} = \frac{\mathrm{cj}}{1 + \overline{\mathrm{mj}} \times (400 \times 10^{-6} \times (\mathrm{Tnom} - 300.15) - \mathrm{gmaold})}$$

$$\overline{\mathrm{tjp}} = \overline{\mathrm{pb}} + \frac{\mathrm{T}}{300.15} \times \mathrm{pbo}$$

$$\mathrm{gmanew} = \frac{\overline{\mathrm{tjp}} - \mathrm{pbo}}{\mathrm{pbo}}$$

$$\overline{\mathrm{tcj}} = \overline{\mathrm{tcj}} \times \left(1 + \overline{\mathrm{mj}} \times (400 \times 10^{-6} \times (\mathrm{T} - 300.15) - \mathrm{gmanew})\right)$$

else

$$\overline{\mathrm{tcj}} = \mathrm{cj} \times \mathrm{cta} \times (\mathrm{T} - 300.15)$$

if tlevc is equal to 0 then

$$\mathrm{pbosw} = \frac{\mathrm{vjsw} - \overline{\mathrm{pb}}1}{\frac{\mathrm{Tnom}}{300.15}}$$

$$\mathrm{gmaold} = \frac{\mathrm{vjsw} - \mathrm{pbosw}}{\mathrm{pbosw}}$$

$$\overline{\mathrm{tcjsw}} = \frac{\mathrm{cjsw}}{1 + \overline{\mathrm{mj}} \times (400 \times 10^{-6}(\mathrm{Tnom} - 300.15) - \mathrm{gmaold})}$$

$$\overline{\mathrm{tjswp}} = \overline{\mathrm{pb}} + \frac{\mathrm{T}}{300.15} \times \mathrm{pbosw}$$

$$\mathrm{gmanew} = \frac{\overline{\mathrm{tjswp}} - \mathrm{pbosw}}{\mathrm{pbosw}}$$

$$\overline{\mathrm{tcjsw}} = \overline{\mathrm{tcjsw}}(1 + \overline{\mathrm{mj}} \times (400 \times 10^{-6} \times (\mathrm{T} - 300.15) - \mathrm{gmanew}))$$

else

$$\overline{\mathrm{tcjsw}} = \mathrm{cjsw} \times \mathrm{cta}(\mathrm{T} - 300.15)$$

Saturation current variables :

$$\overline{\mathrm{tjs}} = \mathrm{js} \times \exp\left(\left(\frac{\mathrm{T}}{\mathrm{Tnom}} - 1\right)\frac{\mathrm{eg}}{\mathrm{n} \times \mathrm{vt}} + \frac{\mathrm{xti}}{\mathrm{n}}\log\left(\frac{\mathrm{T}}{\mathrm{Tnom}}\right)\right)$$

$$\overline{\mathrm{tjsw}} = \mathrm{jsw} \times \exp\left(\left(\frac{\mathrm{T}}{\mathrm{Tnom}} - 1\right)\frac{\mathrm{eg}}{\mathrm{n} \times \mathrm{vt}} + \frac{\mathrm{xti}}{\mathrm{n}}\log\left(\frac{\mathrm{T}}{\mathrm{Tnom}}\right)\right)$$

Additional variables :

$$\mathrm{tF1} = \overline{\mathrm{tjp}} \times \frac{1 - \exp((1 - \overline{\mathrm{mj}}) \times \log(1 - \mathrm{fc}))}{1 - \overline{\mathrm{mj}}}$$

$$\mathrm{tF2} = \exp((1 + \overline{\mathrm{mj}}) \times \log(1 - \mathrm{fc}))$$

$$\mathrm{tF3} = 1 - \mathrm{fc} \times (1 + \overline{\mathrm{mj}})$$

$$\mathrm{tF2SW} = \exp((1 + \mathrm{mjsw}) \times \log(1 - \mathrm{fcs}))$$

$$\mathrm{tF3SW} = 1 - \mathrm{fcs} \times (1 + \mathrm{mjsw})$$

Depletion capacitance variables :

$$\overline{\mathrm{tCdep}} = \overline{\mathrm{tjp}} \times \mathrm{fc}$$

Vcrit :

$$\mathrm{vte} = \mathrm{n} \times \mathrm{vt}$$

$$\mathrm{tVcrit} = \mathrm{vte} \times \log\left(\frac{\mathrm{vte}}{(\sqrt{2} \times \overline{\mathrm{tjs}} \times \mathrm{area})}\right)$$

Transit time :

$$\overline{\mathrm{ttt}} = \mathrm{tt} \times \left(1 + \mathrm{TTT1} \times \mathrm{dt} + \mathrm{TTT2} \times \mathrm{dt}^2\right)$$

Series resistance :

$$\overline{\mathrm{tgs}} = \frac{1}{\mathrm{rs}} \times \frac{1}{1 + \mathrm{trs} \times \mathrm{dt} + \mathrm{trs2} \times \mathrm{dt}^2}$$

Current-Voltage Terms:

DC Current :

$$\mathrm{Isat} = (\overline{\mathrm{tjs}} \times \mathrm{area} + \overline{\mathrm{tjsw}} \times \mathrm{pj}) \times \mathrm{m}$$

$$\mathrm{Id} = \begin{array}{ll}
\left\{\begin{array}{ll}
\mathrm{Isat} \times (\exp(\mathrm{vd}/\mathrm{vte}) - 1) + \mathrm{gmin} \times \mathrm{vd} & (\mathrm{ikf} \leq 0) \\
\frac{\mathrm{Isat}\times(\exp(\mathrm{vd}/\mathrm{vte})-1)}{1+\sqrt{\frac{\mathrm{Isat}\times(\exp(\mathrm{vd}/\mathrm{vte})-1)}{\mathrm{ikf}\times\mathrm{area}\times\mathrm{m}}}} + \mathrm{gmin} \times \mathrm{vd} & (\mathrm{ikf} > 0)
\end{array}\right. & (\mathrm{vd} \geq -3 \times \mathrm{vte}) \\
\left\{\begin{array}{ll}
-\mathrm{Isat} \times \left(1 + \left(\frac{3\times\mathrm{vte}}{\mathrm{e}\times\mathrm{vd}}\right)^3\right) + \mathrm{gmin} \times \mathrm{vd} & (\mathrm{ikr} \leq 0) \\
\frac{-\mathrm{Isat}\times\left(1+\left(\frac{3\times\mathrm{vte}}{\mathrm{e}\times\mathrm{vd}}\right)^3\right)}{1+\sqrt{\frac{-\mathrm{Isat}\times\left(1+\left(\frac{3\times\mathrm{vte}}{\mathrm{e}\times\mathrm{vd}}\right)^3\right)}{-\mathrm{ikr}\times\mathrm{area}\times\mathrm{m}}}} + \mathrm{gmin} \times \mathrm{vd} & (\mathrm{ikr} > 0)
\end{array}\right. & (-\mathrm{BV} \geq \mathrm{vd} < -3 \times \mathrm{vte}) \\
\left\{\begin{array}{ll}
-\mathrm{Isat} \times \exp(-\frac{\mathrm{BV}+\mathrm{vd}}{\mathrm{vte}}) + \mathrm{gmin} \times \mathrm{vd} & (\mathrm{ikr} \leq 0) \\
\frac{-\mathrm{Isat}\times\exp(-\frac{\mathrm{BV}+\mathrm{vd}}{\mathrm{vte}})}{1+\sqrt{\frac{-\mathrm{Isat}\times\exp(-\frac{\mathrm{BV}+\mathrm{vd}}{\mathrm{vte}})}{\mathrm{ikr}\times\mathrm{area}\times\mathrm{m}}}} + \mathrm{gmin} \times \mathrm{vd} & (\mathrm{ikr} > 0)
\end{array}\right. & (\mathrm{vd} < -\mathrm{BV})
\end{array}$$

Charge Storage :

$$\begin{aligned}\mathrm{czero} &= \overline{\mathrm{tcj}} \times \mathrm{area} \times \mathrm{m}\\ \mathrm{czeroSW} &= \overline{\mathrm{tcjsw}} \times \mathrm{pj} \times \mathrm{m}\end{aligned}$$

$$\mathrm{Qd} = \begin{cases} \begin{array}{l} \overline{\mathrm{ttt}} \times \mathrm{Id} \\ \quad + \overline{\mathrm{tjp}} \times \mathrm{czero} \times \frac{1 - \left(1 - \frac{\mathrm{vd}}{\overline{\mathrm{tjp}}}\right) \times \exp\left(-\overline{\mathrm{mj}} \times \log\left(1 - \frac{\mathrm{vd}}{\overline{\mathrm{tjp}}}\right)\right)}{1 - \overline{\mathrm{mj}}} \\ \quad + \overline{\mathrm{tjswp}} \times \mathrm{czeroSW} \times \frac{1 - \left(1 - \frac{\mathrm{vd}}{\overline{\mathrm{tjswp}}}\right) \times \exp\left(-\overline{\mathrm{mj}} \times \log\left(1 - \frac{\mathrm{vd}}{\overline{\mathrm{tjswp}}}\right)\right)}{1 - \overline{\mathrm{mj}}} \end{array} & (\mathrm{vd} < \overline{\mathrm{tCdep}}) \\ \begin{array}{l} \overline{\mathrm{ttt}} \times \mathrm{Id} + \mathrm{czero} \times \mathrm{tF1} \\ \quad + \frac{\mathrm{czero}}{\mathrm{tF2}} \times \left(\mathrm{tF3} \times (\mathrm{vd} - \overline{\mathrm{tCdep}}) + \frac{\overline{\mathrm{mj}}}{2 \times \overline{\mathrm{tjp}}} \times (\mathrm{vd}^2 - \overline{\mathrm{tCdep}}^2)\right) \\ \quad + \frac{\mathrm{czeroSW}}{\mathrm{tF2SW}} \times \left(\mathrm{tF3SW} \times (\mathrm{vd} - \overline{\mathrm{tCdep}}) + \frac{\overline{\mathrm{mj}}}{2 \times \overline{\mathrm{tjswp}}} \times (\mathrm{vd}^2 - \overline{\mathrm{tCdep}}^2)\right) \end{array} & (\mathrm{vd} \geq \overline{\mathrm{tCdep}}) \end{cases}$$

Capacitance :

$$\mathrm{Cd} = \begin{cases} \begin{array}{l} \overline{\mathrm{ttt}} \times \frac{\partial \mathrm{Id}}{\partial \mathrm{vd}} \\ \quad + \mathrm{czero} \times \exp\left(-\overline{\mathrm{mj}} \times \log\left(1 - \frac{\mathrm{vd}}{\overline{\mathrm{tjp}}}\right)\right) \\ \quad + \mathrm{czeroSW} \times \exp\left(-\overline{\mathrm{mj}} \times \log\left(1 - \frac{\mathrm{vd}}{\overline{\mathrm{tjswp}}}\right)\right) \end{array} & (\mathrm{vd} < \overline{\mathrm{tCdep}}) \\ \begin{array}{l} \overline{\mathrm{ttt}} \times \frac{\partial \mathrm{Id}}{\partial \mathrm{vd}} + \mathrm{czero} \times \mathrm{tF1} \\ \quad + \frac{\mathrm{czero}}{\mathrm{tF2}} \times \left(\mathrm{tF3} + \frac{\overline{\mathrm{mj}}}{\overline{\mathrm{tjp}}} \times \mathrm{vd}\right) \\ \quad + \frac{\mathrm{czero}}{\mathrm{tF2SW}} \times \left(\mathrm{tF3SW} + \frac{\overline{\mathrm{mj}}}{\overline{\mathrm{tjswp}}} \times \mathrm{vd}\right) \end{array} & (\mathrm{vd} \geq \overline{\mathrm{tCdep}}) \end{cases}$$

5.5.2. QXX : BJTs

Syntax :

```
QXX [NC] [NB] [NE] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] [DEVICE TYPE] [MODEL PARAMS]
```

parameter	*description*
[NC]	Collector node/net
[NB]	Base node/net
[NE]	Emitter node/net (can be followed by an optional source node [NS])
[MODEL]	Bipolar device model defined by a .model statement
[INLINE PARAMS]	*Inline parameters :* areaa : Emitter area factor areab : Base area factor areac : Collector area factor m : Current multiplication factor off : Initial condition for diode state ic : Initial voltage condition temp : Temperature dtemp : Temperature difference with ambient
[DEVICE TYPE]	NPN or PNP

[MODEL PARAMS]	*description of model parameters :*
npn	NPN type device
pnp	PNP type device
subs	Vertical or Lateral device
tnom	Parameter measurement temperature
tref	Parameter measurement temperature
is	Saturation Current
bf	Ideal forward beta
nf	Forward emission coefficient
vaf	Forward Early voltage
va	Forward Early voltage
ikf	Forward beta roll-off corner current
ik	Forward beta roll-off corner current
ise	B-E leakage saturation current
ne	B-E leakage emission coefficient
br	Ideal reverse beta
nr	Reverse emission coefficient
var	Reverse Early voltage

vb	Reverse Early voltage
ikr	reverse beta roll-off corner current
isc	B-C leakage saturation current
nc	B-C leakage emission coefficient
rb	Zero bias base resistance
irb	Current for base resistance=(rb+rbm)/2
rbm	Minimum base resistance
re	Emitter resistance
rc	Collector resistance
cje	Zero bias B-E depletion capacitance
vje	B-E built in potential
pe	B-E built in potential
mje	B-E junction grading coefficient
me	B-E junction grading coefficient
tf	Ideal forward transit time
xtf	Coefficient for bias dependence of TF
vtf	Voltage giving VBC dependence of TF
itf	High current dependence of TF
ptf	Excess phase
cjc	Zero bias B-C depletion capacitance
vjc	B-C built in potential
pc	B-C built in potential
mjc	B-C junction grading coefficient
mc	B-C junction grading coefficient
xcjc	Fraction of B-C cap to internal base
tr	Ideal reverse transit time
cjs	Zero bias Substrate capacitance
csub	Zero bias Substrate capacitance
ccs	Zero bias Substrate capacitance
vjs	Substrate junction built in potential
ps	Substrate junction built in potential
mjs	Substrate junction grading coefficient
ms	Substrate junction grading coefficient
xtb	Forward and reverse beta temp. exp.
eg	Energy gap for IS temp. dependency
xti	Temp. exponent for IS
fc	Forward bias junction fit parameter
kf	Flicker Noise Coefficient
af	Flicker Noise Exponent
invearlyvoltf	Inverse early voltage:forward
invearlyvoltr	Inverse early voltage:reverse
invrollofff	Inverse roll off - forward
invrolloffr	Inverse roll off - reverse
collectorconduct	Collector conductance
emitterconduct	Emitter conductance

transtimevbcfact	Transit time VBC factor
excessphasefactor	Excess phase fact.
iss	Substrate Jct. Saturation Current
ns	Substrate current emission coefficient
tlev	Temperature equation selector
tlevc	Temperature equation selector
tbf1	BF 1. temperature coefficient
tbf2	BF 2. temperature coefficient
tbr1	BR 1. temperature coefficient
tbr2	BR 2. temperature coefficient
tikf1	IKF 1. temperature coefficient
tikf2	IKF 2. temperature coefficient
tikr1	IKR 1. temperature coefficient
tikr2	IKR 2. temperature coefficient
tirb1	IRB 1. temperature coefficient
tirb2	IRB 2. temperature coefficient
tnc1	NC 1. temperature coefficient
tnc2	NC 2. temperature coefficient
tne1	NE 1. temperature coefficient
tne2	NE 2. temperature coefficient
tnf1	NF 1. temperature coefficient
tnf2	NF 2. temperature coefficient
tnr1	NR 1. temperature coefficient
tnr2	NR 2. temperature coefficient
trb1	RB 1. temperature coefficient
trb2	RB 2. temperature coefficient
trc1	RC 1. temperature coefficient
trc2	RC 2. temperature coefficient
tre1	RE 1. temperature coefficient
tre2	RE 2. temperature coefficient
trm1	RBM 1. temperature coefficient
trm2	RBM 2. temperature coefficient
tvaf1	VAF 1. temperature coefficient
tvaf2	VAF 2. temperature coefficient
tvar1	VAR 1. temperature coefficient
tvar2	VAR 2. temperature coefficient
ctc	CJC temperature coefficient
cte	CJE temperature coefficient
cts	CJS temperature coefficient
tvjc	VJC temperature coefficient
tvje	VJE temperature coefficient
tvjs	VJS temperature coefficient
titf1	ITF 1. temperature coefficient
titf2	ITF 2. temperature coefficient
ttf1	TF 1. temperature coefficient

ttf2	TF 2. temperature coefficient
ttr1	TR 1. temperature coefficient
ttr2	TR 2. temperature coefficient
tmje1	MJE 1. temperature coefficient
tmje2	MJE 2. temperature coefficient
tmjc1	MJC 1. temperature coefficient
tmjc2	MJC 2. temperature coefficient
tmjs1	MJS 1. temperature coefficient
tmjs2	MJS 2. temperature coefficient
tns1	NS 1. temperature coefficient
tns2	NS 2. temperature coefficient
nkf	NKF High current beta rolloff exponent

Example:

statement	*explanation*

`xx`
`.model xx`
	xx

5.5.3. MXX : MOSFETs

Syntax :

```
MXX [ND] [NG] [NS] [NB] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] [DEVICE TYPE] [MODEL PARAMS]
```

parameter	*description*
[ND]	Drain node/net
[NG]	Gate node/net
[NS]	Source node/net
[NB]	Body node/net
[MODEL]	MOSFET device model defined by a .model statement
[INLINE PARAMS]	*Inline parameters :* m : Current multiplication factor l : Length w : Width ad : Drain area as : Source area pd : Drain perimeter ps : Source perimeter nrd : Number of drain squares nrs : Number of source squares off : Initial condition for diode state ic : Initial voltage condition temp : Temperature
[DEVICE TYPE]	NMOS or PMOS

[MODEL PARAMS]	*description of model parameters :*
cvchargemod	Capacitance Charge model selector
capmod	Capacitance model selector
diomod	Diode IV model selector
rdsmod	Bias-dependent S/D resistance model selector
trnqsmod	Transient NQS model selector
acnqsmod	AC NQS model selector
mobmod	Mobility model selector
rbodymod	Distributed body R model selector
rgatemod	Gate R model selector
permod	Pd and Ps model selector
geomod	Geometry dependent parasitics model selector
fnoimod	Flicker noise model selector

tnoimod	Thermal noise model selector
mtrlmod	parameter for non-silicon substrate or metal gate selector
igcmod	Gate-to-channel Ig model selector
igbmod	Gate-to-body Ig model selector
tempmod	Temperature model selector
paramchk	Model parameter checking selector
binunit	Bin unit selector
version	parameter for model version
eot	Equivalent gate oxide thickness in meters
vddeot	Voltage for extraction of Equivalent gate oxide thickness
tempeot	Temperature for extraction of EOT
leffeot	Effective length for extraction of EOT
weffeot	Effective width for extraction of EOT
ados	Charge centroid parameter
bdos	Charge centroid parameter
toxe	Electrical gate oxide thickness in meters
toxp	Physical gate oxide thickness in meters
toxm	Gate oxide thickness at which parameters are extracted
toxref	Target tox value
dtox	Defined as (toxe - toxp)
epsrox	Dielectric constant of the gate oxide relative to vacuum
cdsc	Drain/Source and channel coupling capacitance
cdscb	Body-bias dependence of cdsc
cdscd	Drain-bias dependence of cdsc
cit	Interface state capacitance
nfactor	Subthreshold swing Coefficient
xj	Junction depth in meters
vsat	Saturation velocity at tnom
at	Temperature coefficient of vsat
a0	Non-uniform depletion width effect coefficient.
ags	Gate bias coefficient of Abulk.
a1	Non-saturation effect coefficient
a2	Non-saturation effect coefficient
keta	Body-bias coefficient of non-uniform depletion width effect.
phig	Work function of gate
epsrgate	Dielectric constant of gate relative to vacuum
easub	Electron affinity of substrate
epsrsub	Dielectric constant of substrate relative to vacuum
ni0sub	Intrinsic carrier concentration of substrate at 300.15K
bg0sub	Band-gap of substrate at T=0K
tbgasub	First parameter of band-gap change due to temperature
tbgbsub	Second parameter of band-gap change due to temperature
nsub	Substrate doping concentration
ndep	Channel doping concentration at the depletion edge
nsd	S/D doping concentration

phin	Adjusting parameter for surface potential due to non-uniform vertical doping
ngate	Poly-gate doping concentration
gamma1	Vth body coefficient
gamma2	Vth body coefficient
vbx	Vth transition body Voltage
vbm	Maximum body voltage
xt	Doping depth
k1	Bulk effect coefficient 1
kt1	Temperature coefficient of Vth
kt1l	Temperature coefficient of Vth
kt2	Body-coefficient of kt1
k2	Bulk effect coefficient 2
k3	Narrow width effect coefficient
k3b	Body effect coefficient of k3
w0	Narrow width effect parameter
dvtp0	First parameter for Vth shift due to pocket
dvtp1	Second parameter for Vth shift due to pocket
lpe0	Equivalent length of pocket region at zero bias
lpeb	Equivalent length of pocket region accounting for body bias
dvt0	Short channel effect coeff. 0
dvt1	Short channel effect coeff. 1
dvt2	Short channel effect coeff. 2
dvt0w	Narrow Width coeff. 0
dvt1w	Narrow Width effect coeff. 1
dvt2w	Narrow Width effect coeff. 2
drout	DIBL coefficient of output resistance
dsub	DIBL coefficient in the subthreshold region
vth0	Threshold voltage
vtho	Threshold voltage
ua	Linear gate dependence of mobility
ua1	Temperature coefficient of ua
ub	Quadratic gate dependence of mobility
ub1	Temperature coefficient of ub
uc	Body-bias dependence of mobility
uc1	Temperature coefficient of uc
ud	Coulomb scattering factor of mobility
ud1	Temperature coefficient of ud
up	Channel length linear factor of mobility
lp	Channel length exponential factor of mobility
u0	Low-field mobility at Tnom
eu	Mobility exponent
ucs	Colombic scattering exponent
ute	Temperature coefficient of mobility
ucste	Temperature coefficient of colombic mobility
voff	Threshold voltage offset

minv	Fitting parameter for moderate inversion in Vgsteff
minvcv	Fitting parameter for moderate inversion in Vgsteffcv
voffl	Length dependence parameter for Vth offset
voffcvl	Length dependence parameter for Vth offset in CV
tnom	Parameter measurement temperature
cgso	Gate-source overlap capacitance per width
cgdo	Gate-drain overlap capacitance per width
cgbo	Gate-bulk overlap capacitance per length
xpart	Channel charge partitioning
delta	Effective Vds parameter
rsh	Source-drain sheet resistance
rdsw	Source-drain resistance per width
rdswmin	Source-drain resistance per width at high Vg
rsw	Source resistance per width
rdw	Drain resistance per width
rdwmin	Drain resistance per width at high Vg
rswmin	Source resistance per width at high Vg
prwg	Gate-bias effect on parasitic resistance
prwb	Body-effect on parasitic resistance
prt	Temperature coefficient of parasitic resistance
eta0	Subthreshold region DIBL coefficient
etab	Subthreshold region DIBL coefficient
pclm	Channel length modulation Coefficient
pdiblc1	Drain-induced barrier lowering coefficient
pdiblc2	Drain-induced barrier lowering coefficient
pdiblcb	Body-effect on drain-induced barrier lowering
fprout	Rout degradation coefficient for pocket devices
pdits	Coefficient for drain-induced Vth shifts
pditsl	Length dependence of drain-induced Vth shifts
pditsd	Vds dependence of drain-induced Vth shifts
pscbe1	Substrate current body-effect coefficient
pscbe2	Substrate current body-effect coefficient
pvag	Gate dependence of output resistance parameter
jss	Bottom source junction reverse saturation current density
jsws	Isolation edge sidewall source junction reverse saturation current density
jswgs	Gate edge source junction reverse saturation current density
pbs	Source junction built-in potential
njs	Source junction emission coefficient
xtis	Source junction current temperature exponent
mjs	Source bottom junction capacitance grading coefficient
pbsws	Source sidewall junction capacitance built in potential
mjsws	Source sidewall junction capacitance grading coefficient
pbswgs	Source (gate side) sidewall junction capacitance built in potential
mjswgs	Source (gate side) sidewall junction capacitance grading coefficient
cjs	Source bottom junction capacitance per unit area

cjsws	Source sidewall junction capacitance per unit periphery
cjswgs	Source (gate side) sidewall junction capacitance per unit width
jsd	Bottom drain junction reverse saturation current density
jswd	Isolation edge sidewall drain junction reverse saturation current density
jswgd	Gate edge drain junction reverse saturation current density
pbd	Drain junction built-in potential
njd	Drain junction emission coefficient
xtid	Drainjunction current temperature exponent
mjd	Drain bottom junction capacitance grading coefficient
pbswd	Drain sidewall junction capacitance built in potential
mjswd	Drain sidewall junction capacitance grading coefficient
pbswgd	Drain (gate side) sidewall junction capacitance built in potential
mjswgd	Drain (gate side) sidewall junction capacitance grading coefficient
cjd	Drain bottom junction capacitance per unit area
cjswd	Drain sidewall junction capacitance per unit periphery
cjswgd	Drain (gate side) sidewall junction capacitance per unit width
vfbcv	Flat Band Voltage parameter for capmod=0 only
vfb	Flat Band Voltage
tpb	Temperature coefficient of pb
tcj	Temperature coefficient of cj
tpbsw	Temperature coefficient of pbsw
tcjsw	Temperature coefficient of cjsw
tpbswg	Temperature coefficient of pbswg
tcjswg	Temperature coefficient of cjswg
acde	Exponential coefficient for finite charge thickness
moin	Coefficient for gate-bias dependent surface potential
noff	C-V turn-on/off parameter
voffcv	C-V lateral-shift parameter
dmcg	Distance of Mid-Contact to Gate edge
dmci	Distance of Mid-Contact to Isolation
dmdg	Distance of Mid-Diffusion to Gate edge
dmcgt	Distance of Mid-Contact to Gate edge in Test structures
xgw	Distance from gate contact center to device edge
xgl	Variation in Ldrawn
rshg	Gate sheet resistance
ngcon	Number of gate contacts
xrcrg1	First fitting parameter the bias-dependent Rg
xrcrg2	Second fitting parameter the bias-dependent Rg
lambda	Velocity overshoot parameter
vtl	Thermal velocity
lc	back scattering parameter
xn	back scattering parameter
vfbsdoff	S/D flatband voltage offset
tvfbsdoff	Temperature parameter for vfbsdoff
tvoff	Temperature parameter for voff

lintnoi	lint offset for noise calculation
lint	Length reduction parameter
ll	Length reduction parameter
llc	Length reduction parameter for CV
lln	Length reduction parameter
lw	Length reduction parameter
lwc	Length reduction parameter for CV
lwn	Length reduction parameter
lwl	Length reduction parameter
lwlc	Length reduction parameter for CV
lmin	Minimum length for the model
lmax	Maximum length for the model
wr	Width dependence of rds
wint	Width reduction parameter
dwg	Width reduction parameter
dwb	Width reduction parameter
wl	Width reduction parameter
wlc	Width reduction parameter for CV
wln	Width reduction parameter
ww	Width reduction parameter
wwc	Width reduction parameter for CV
wwn	Width reduction parameter
wwl	Width reduction parameter
wwlc	Width reduction parameter for CV
wmin	Minimum width for the model
wmax	Maximum width for the model
b0	Abulk narrow width parameter
b1	Abulk narrow width parameter
cgsl	New C-V model parameter
cgdl	New C-V model parameter
ckappas	S/G overlap C-V parameter
ckappad	D/G overlap C-V parameter
cf	Fringe capacitance parameter
clc	Vdsat parameter for C-V model
cle	Vdsat parameter for C-V model
dwc	Delta W for C-V model
dlc	Delta L for C-V model
xw	W offset for channel width due to mask/etch effect
xl	L offset for channel length due to mask/etch effect
dlcig	Delta L for Ig model
dlcigd	Delta L for Ig model drain side
dwj	Delta W for S/D junctions
alpha0	substrate current model parameter
alpha1	substrate current model parameter
beta0	substrate current model parameter

agidl	Pre-exponential constant for GIDL
bgidl	Exponential constant for GIDL
cgidl	Parameter for body-bias dependence of GIDL
egidl	Fitting parameter for Bandbending
agisl	Pre-exponential constant for GISL
bgisl	Exponential constant for GISL
cgisl	Parameter for body-bias dependence of GISL
egisl	Fitting parameter for Bandbending
aigc	Parameter for Igc
bigc	Parameter for Igc
cigc	Parameter for Igc
aigsd	Parameter for Igs,d
bigsd	Parameter for Igs,d
cigsd	Parameter for Igs,d
aigs	Parameter for Igs
bigs	Parameter for Igs
cigs	Parameter for Igs
aigd	Parameter for Igd
bigd	Parameter for Igd
cigd	Parameter for Igd
aigbacc	Parameter for Igb
bigbacc	Parameter for Igb
cigbacc	Parameter for Igb
aigbinv	Parameter for Igb
bigbinv	Parameter for Igb
cigbinv	Parameter for Igb
nigc	Parameter for Igc slope
nigbinv	Parameter for Igbinv slope
nigbacc	Parameter for Igbacc slope
ntox	Exponent for Tox ratio
eigbinv	Parameter for the Si bandgap for Igbinv
pigcd	Parameter for Igc partition
poxedge	Factor for the gate edge Tox
ijthdfwd	Forward drain diode forward limiting current
ijthsfwd	Forward source diode forward limiting current
ijthdrev	Reverse drain diode forward limiting current
ijthsrev	Reverse source diode forward limiting current
xjbvd	Fitting parameter for drain diode breakdown current
xjbvs	Fitting parameter for source diode breakdown current
bvd	Drain diode breakdown voltage
bvs	Source diode breakdown voltage
jtss	Source bottom trap-assisted saturation current density
jtsd	Drain bottom trap-assisted saturation current density
jtssws	Source STI sidewall trap-assisted saturation current density
jtsswd	Drain STI sidewall trap-assisted saturation current density

jtsswgs	Source gate-edge sidewall trap-assisted saturation current density
jtsswgd	Drain gate-edge sidewall trap-assisted saturation current density
jtweff	TAT current width dependance
njts	Non-ideality factor for bottom junction
njtssw	Non-ideality factor for STI sidewall junction
njtsswg	Non-ideality factor for gate-edge sidewall junction
njtsd	Non-ideality factor for bottom junction drain side
njtsswd	Non-ideality factor for STI sidewall junction drain side
njtsswgd	Non-ideality factor for gate-edge sidewall junction drain side
xtss	Power dependence of JTSS on temperature
xtsd	Power dependence of JTSD on temperature
xtssws	Power dependence of JTSSWS on temperature
xtsswd	Power dependence of JTSSWD on temperature
xtsswgs	Power dependence of JTSSWGS on temperature
xtsswgd	Power dependence of JTSSWGD on temperature
tnjts	Temperature coefficient for NJTS
tnjtssw	Temperature coefficient for NJTSSW
tnjtsswg	Temperature coefficient for NJTSSWG
tnjtsd	Temperature coefficient for NJTSD
tnjtsswd	Temperature coefficient for NJTSSWD
tnjtsswgd	Temperature coefficient for NJTSSWGD
vtss	Source bottom trap-assisted voltage dependent parameter
vtsd	Drain bottom trap-assisted voltage dependent parameter
vtssws	Source STI sidewall trap-assisted voltage dependent parameter
vtsswd	Drain STI sidewall trap-assisted voltage dependent parameter
vtsswgs	Source gate-edge sidewall trap-assisted voltage dependent parameter
vtsswgd	Drain gate-edge sidewall trap-assisted voltage dependent parameter
gbmin	Minimum body conductance
rbdb	Resistance between bNode and dbNode
rbpb	Resistance between bNodePrime and bNode
rbsb	Resistance between bNode and sbNode
rbps	Resistance between bNodePrime and sbNode
rbpd	Resistance between bNodePrime and bNode
rbps0	Body resistance RBPS scaling
rbpsl	Body resistance RBPS L scaling
rbpsw	Body resistance RBPS W scaling
rbpsnf	Body resistance RBPS NF scaling
rbpd0	Body resistance RBPD scaling
rbpdl	Body resistance RBPD L scaling
rbpdw	Body resistance RBPD W scaling
rbpdnf	Body resistance RBPD NF scaling
rbpbx0	Body resistance RBPBX scaling
rbpbxl	Body resistance RBPBX L scaling
rbpbxw	Body resistance RBPBX W scaling
rbpbxnf	Body resistance RBPBX NF scaling

rbpby0	Body resistance RBPBY scaling
rbpbyl	Body resistance RBPBY L scaling
rbpbyw	Body resistance RBPBY W scaling
rbpbynf	Body resistance RBPBY NF scaling
rbsbx0	Body resistance RBSBX scaling
rbsby0	Body resistance RBSBY scaling
rbdbx0	Body resistance RBDBX scaling
rbdby0	Body resistance RBDBY scaling
rbsdbxl	Body resistance RBSDBX L scaling
rbsdbxw	Body resistance RBSDBX W scaling
rbsdbxnf	Body resistance RBSDBX NF scaling
rbsdbyl	Body resistance RBSDBY L scaling
rbsdbyw	Body resistance RBSDBY W scaling
rbsdbynf	Body resistance RBSDBY NF scaling
lcdsc	Length dependence of cdsc
lcdscb	Length dependence of cdscb
lcdscd	Length dependence of cdscd
lcit	Length dependence of cit
lnfactor	Length dependence of nfactor
lxj	Length dependence of xj
lvsat	Length dependence of vsat
lat	Length dependence of at
la0	Length dependence of a0
lags	Length dependence of ags
la1	Length dependence of a1
la2	Length dependence of a2
lketa	Length dependence of keta
lnsub	Length dependence of nsub
lndep	Length dependence of ndep
lnsd	Length dependence of nsd
lphin	Length dependence of phin
lngate	Length dependence of ngate
lgamma1	Length dependence of gamma1
lgamma2	Length dependence of gamma2
lvbx	Length dependence of vbx
lvbm	Length dependence of vbm
lxt	Length dependence of xt
lk1	Length dependence of k1
lkt1	Length dependence of kt1
lkt1l	Length dependence of kt1l
lkt2	Length dependence of kt2
lk2	Length dependence of k2
lk3	Length dependence of k3
lk3b	Length dependence of k3b
lw0	Length dependence of w0

ldvtp0	Length dependence of dvtp0
ldvtp1	Length dependence of dvtp1
llpe0	Length dependence of lpe0
llpeb	Length dependence of lpeb
ldvt0	Length dependence of dvt0
ldvt1	Length dependence of dvt1
ldvt2	Length dependence of dvt2
ldvt0w	Length dependence of dvt0w
ldvt1w	Length dependence of dvt1w
ldvt2w	Length dependence of dvt2w
ldrout	Length dependence of drout
ldsub	Length dependence of dsub
lvth0	Length dependence of vto
lvtho	Length dependence of vto
lua	Length dependence of ua
lua1	Length dependence of ua1
lub	Length dependence of ub
lub1	Length dependence of ub1
luc	Length dependence of uc
luc1	Length dependence of uc1
lud	Length dependence of ud
lud1	Length dependence of ud1
lup	Length dependence of up
llp	Length dependence of lp
lu0	Length dependence of u0
lute	Length dependence of ute
lucste	Length dependence of ucste
lvoff	Length dependence of voff
lminv	Length dependence of minv
lminvcv	Length dependence of minvcv
ldelta	Length dependence of delta
lrdsw	Length dependence of rdsw
lrsw	Length dependence of rsw
lrdw	Length dependence of rdw
lprwg	Length dependence of prwg
lprwb	Length dependence of prwb
lprt	Length dependence of prt
leta0	Length dependence of eta0
letab	Length dependence of etab
lpclm	Length dependence of pclm
lpdiblc1	Length dependence of pdiblc1
lpdiblc2	Length dependence of pdiblc2
lpdiblcb	Length dependence of pdiblcb
lfprout	Length dependence of pdiblcb
lpdits	Length dependence of pdits

lpditsd	Length dependence of pditsd
lpscbe1	Length dependence of pscbe1
lpscbe2	Length dependence of pscbe2
lpvag	Length dependence of pvag
lwr	Length dependence of wr
ldwg	Length dependence of dwg
ldwb	Length dependence of dwb
lb0	Length dependence of b0
lb1	Length dependence of b1
lcgsl	Length dependence of cgsl
lcgdl	Length dependence of cgdl
lckappas	Length dependence of ckappas
lckappad	Length dependence of ckappad
lcf	Length dependence of cf
lclc	Length dependence of clc
lcle	Length dependence of cle
lalpha0	Length dependence of alpha0
lalpha1	Length dependence of alpha1
lbeta0	Length dependence of beta0
lagidl	Length dependence of agidl
lbgidl	Length dependence of bgidl
lcgidl	Length dependence of cgidl
legidl	Length dependence of egidl
lagisl	Length dependence of agisl
lbgisl	Length dependence of bgisl
lcgisl	Length dependence of cgisl
legisl	Length dependence of egisl
laigc	Length dependence of aigc
lbigc	Length dependence of bigc
lcigc	Length dependence of cigc
laigsd	Length dependence of aigsd
lbigsd	Length dependence of bigsd
lcigsd	Length dependence of cigsd
laigs	Length dependence of aigs
lbigs	Length dependence of bigs
lcigs	Length dependence of cigs
laigd	Length dependence of aigd
lbigd	Length dependence of bigd
lcigd	Length dependence of cigd
laigbacc	Length dependence of aigbacc
lbigbacc	Length dependence of bigbacc
lcigbacc	Length dependence of cigbacc
laigbinv	Length dependence of aigbinv
lbigbinv	Length dependence of bigbinv
lcigbinv	Length dependence of cigbinv

lnigc	Length dependence of nigc
lnigbinv	Length dependence of nigbinv
lnigbacc	Length dependence of nigbacc
lntox	Length dependence of ntox
leigbinv	Length dependence for eigbinv
lpigcd	Length dependence for pigcd
lpoxedge	Length dependence for poxedge
lvfbcv	Length dependence of vfbcv
lvfb	Length dependence of vfb
lacde	Length dependence of acde
lmoin	Length dependence of moin
lnoff	Length dependence of noff
lvoffcv	Length dependence of voffcv
lxrcrg1	Length dependence of xrcrg1
lxrcrg2	Length dependence of xrcrg2
llambda	Length dependence of lambda
lvtl	Length dependence of vtl
lxn	Length dependence of xn
leu	Length dependence of eu
lucs	Length dependence of lucs
lvfbsdoff	Length dependence of vfbsdoff
ltvfbsdoff	Length dependence of tvfbsdoff
ltvoff	Length dependence of tvoff
wcdsc	Width dependence of cdsc
wcdscb	Width dependence of cdscb
wcdscd	Width dependence of cdscd
wcit	Width dependence of cit
wnfactor	Width dependence of nfactor
wxj	Width dependence of xj
wvsat	Width dependence of vsat
wat	Width dependence of at
wa0	Width dependence of a0
wags	Width dependence of ags
wa1	Width dependence of a1
wa2	Width dependence of a2
wketa	Width dependence of keta
wnsub	Width dependence of nsub
wndep	Width dependence of ndep
wnsd	Width dependence of nsd
wphin	Width dependence of phin
wngate	Width dependence of ngate
wgamma1	Width dependence of gamma1
wgamma2	Width dependence of gamma2
wvbx	Width dependence of vbx
wvbm	Width dependence of vbm

wxt	Width dependence of xt
wk1	Width dependence of k1
wkt1	Width dependence of kt1
wkt1l	Width dependence of kt1l
wkt2	Width dependence of kt2
wk2	Width dependence of k2
wk3	Width dependence of k3
wk3b	Width dependence of k3b
ww0	Width dependence of w0
wdvtp0	Width dependence of dvtp0
wdvtp1	Width dependence of dvtp1
wlpe0	Width dependence of lpe0
wlpeb	Width dependence of lpeb
wdvt0	Width dependence of dvt0
wdvt1	Width dependence of dvt1
wdvt2	Width dependence of dvt2
wdvt0w	Width dependence of dvt0w
wdvt1w	Width dependence of dvt1w
wdvt2w	Width dependence of dvt2w
wdrout	Width dependence of drout
wdsub	Width dependence of dsub
wvth0	Width dependence of vto
wvtho	Width dependence of vto
wua	Width dependence of ua
wua1	Width dependence of ua1
wub	Width dependence of ub
wub1	Width dependence of ub1
wuc	Width dependence of uc
wuc1	Width dependence of uc1
wud	Width dependence of ud
wud1	Width dependence of ud1
wup	Width dependence of up
wlp	Width dependence of lp
wu0	Width dependence of u0
wute	Width dependence of ute
wucste	Width dependence of ucste
wvoff	Width dependence of voff
wminv	Width dependence of minv
wminvcv	Width dependence of minvcv
wdelta	Width dependence of delta
wrdsw	Width dependence of rdsw
wrsw	Width dependence of rsw
wrdw	Width dependence of rdw
wprwg	Width dependence of prwg
wprwb	Width dependence of prwb

wprt	Width dependence of prt
weta0	Width dependence of eta0
wetab	Width dependence of etab
wpclm	Width dependence of pclm
wpdiblc1	Width dependence of pdiblc1
wpdiblc2	Width dependence of pdiblc2
wpdiblcb	Width dependence of pdiblcb
wfprout	Width dependence of pdiblcb
wpdits	Width dependence of pdits
wpditsd	Width dependence of pditsd
wpscbe1	Width dependence of pscbe1
wpscbe2	Width dependence of pscbe2
wpvag	Width dependence of pvag
wwr	Width dependence of wr
wdwg	Width dependence of dwg
wdwb	Width dependence of dwb
wb0	Width dependence of b0
wb1	Width dependence of b1
wcgsl	Width dependence of cgsl
wcgdl	Width dependence of cgdl
wckappas	Width dependence of ckappas
wckappad	Width dependence of ckappad
wcf	Width dependence of cf
wclc	Width dependence of clc
wcle	Width dependence of cle
walpha0	Width dependence of alpha0
walpha1	Width dependence of alpha1
wbeta0	Width dependence of beta0
wagidl	Width dependence of agidl
wbgidl	Width dependence of bgidl
wcgidl	Width dependence of cgidl
wegidl	Width dependence of egidl
wagisl	Width dependence of agisl
wbgisl	Width dependence of bgisl
wcgisl	Width dependence of cgisl
wegisl	Width dependence of egisl
waigc	Width dependence of aigc
wbigc	Width dependence of bigc
wcigc	Width dependence of cigc
waigsd	Width dependence of aigsd
wbigsd	Width dependence of bigsd
wcigsd	Width dependence of cigsd
waigs	Width dependence of aigs
wbigs	Width dependence of bigs
wcigs	Width dependence of cigs

waigd	Width dependence of aigd
wbigd	Width dependence of bigd
wcigd	Width dependence of cigd
waigbacc	Width dependence of aigbacc
wbigbacc	Width dependence of bigbacc
wcigbacc	Width dependence of cigbacc
waigbinv	Width dependence of aigbinv
wbigbinv	Width dependence of bigbinv
wcigbinv	Width dependence of cigbinv
wnigc	Width dependence of nigc
wnigbinv	Width dependence of nigbinv
wnigbacc	Width dependence of nigbacc
wntox	Width dependence of ntox
weigbinv	Width dependence for eigbinv
wpigcd	Width dependence for pigcd
wpoxedge	Width dependence for poxedge
wvfbcv	Width dependence of vfbcv
wvfb	Width dependence of vfb
wacde	Width dependence of acde
wmoin	Width dependence of moin
wnoff	Width dependence of noff
wvoffcv	Width dependence of voffcv
wxrcrg1	Width dependence of xrcrg1
wxrcrg2	Width dependence of xrcrg2
wlambda	Width dependence of lambda
wvtl	Width dependence of vtl
wxn	Width dependence of xn
weu	Width dependence of eu
wucs	Width dependence of ucs
wvfbsdoff	Width dependence of vfbsdoff
wtvfbsdoff	Width dependence of tvfbsdoff
wtvoff	Width dependence of tvoff
pcdsc	Cross-term dependence of cdsc
pcdscb	Cross-term dependence of cdscb
pcdscd	Cross-term dependence of cdscd
pcit	Cross-term dependence of cit
pnfactor	Cross-term dependence of nfactor
pxj	Cross-term dependence of xj
pvsat	Cross-term dependence of vsat
pat	Cross-term dependence of at
pa0	Cross-term dependence of a0
pags	Cross-term dependence of ags
pa1	Cross-term dependence of a1
pa2	Cross-term dependence of a2
pketa	Cross-term dependence of keta

pnsub	Cross-term dependence of nsub
pndep	Cross-term dependence of ndep
pnsd	Cross-term dependence of nsd
pphin	Cross-term dependence of phin
pngate	Cross-term dependence of ngate
pgamma1	Cross-term dependence of gamma1
pgamma2	Cross-term dependence of gamma2
pvbx	Cross-term dependence of vbx
pvbm	Cross-term dependence of vbm
pxt	Cross-term dependence of xt
pk1	Cross-term dependence of k1
pkt1	Cross-term dependence of kt1
pkt1l	Cross-term dependence of kt1l
pkt2	Cross-term dependence of kt2
pk2	Cross-term dependence of k2
pk3	Cross-term dependence of k3
pk3b	Cross-term dependence of k3b
pw0	Cross-term dependence of w0
pdvtp0	Cross-term dependence of dvtp0
pdvtp1	Cross-term dependence of dvtp1
plpe0	Cross-term dependence of lpe0
plpeb	Cross-term dependence of lpeb
pdvt0	Cross-term dependence of dvt0
pdvt1	Cross-term dependence of dvt1
pdvt2	Cross-term dependence of dvt2
pdvt0w	Cross-term dependence of dvt0w
pdvt1w	Cross-term dependence of dvt1w
pdvt2w	Cross-term dependence of dvt2w
pdrout	Cross-term dependence of drout
pdsub	Cross-term dependence of dsub
pvth0	Cross-term dependence of vto
pvtho	Cross-term dependence of vto
pua	Cross-term dependence of ua
pua1	Cross-term dependence of ua1
pub	Cross-term dependence of ub
pub1	Cross-term dependence of ub1
puc	Cross-term dependence of uc
puc1	Cross-term dependence of uc1
pud	Cross-term dependence of ud
pud1	Cross-term dependence of ud1
pup	Cross-term dependence of up
plp	Cross-term dependence of lp
pu0	Cross-term dependence of u0
pute	Cross-term dependence of ute
pucste	Cross-term dependence of ucste

pvoff	Cross-term dependence of voff
pminv	Cross-term dependence of minv
pminvcv	Cross-term dependence of minvcv
pdelta	Cross-term dependence of delta
prdsw	Cross-term dependence of rdsw
prsw	Cross-term dependence of rsw
prdw	Cross-term dependence of rdw
pprwg	Cross-term dependence of prwg
pprwb	Cross-term dependence of prwb
pprt	Cross-term dependence of prt
peta0	Cross-term dependence of eta0
petab	Cross-term dependence of etab
ppclm	Cross-term dependence of pclm
ppdiblc1	Cross-term dependence of pdiblc1
ppdiblc2	Cross-term dependence of pdiblc2
ppdiblcb	Cross-term dependence of pdiblcb
pfprout	Cross-term dependence of pdiblcb
ppdits	Cross-term dependence of pdits
ppditsd	Cross-term dependence of pditsd
ppscbe1	Cross-term dependence of pscbe1
ppscbe2	Cross-term dependence of pscbe2
ppvag	Cross-term dependence of pvag
pwr	Cross-term dependence of wr
pdwg	Cross-term dependence of dwg
pdwb	Cross-term dependence of dwb
pb0	Cross-term dependence of b0
pb1	Cross-term dependence of b1
pcgsl	Cross-term dependence of cgsl
pcgdl	Cross-term dependence of cgdl
pckappas	Cross-term dependence of ckappas
pckappad	Cross-term dependence of ckappad
pcf	Cross-term dependence of cf
pclc	Cross-term dependence of clc
pcle	Cross-term dependence of cle
palpha0	Cross-term dependence of alpha0
palpha1	Cross-term dependence of alpha1
pbeta0	Cross-term dependence of beta0
pagidl	Cross-term dependence of agidl
pbgidl	Cross-term dependence of bgidl
pcgidl	Cross-term dependence of cgidl
pegidl	Cross-term dependence of egidl
pagisl	Cross-term dependence of agisl
pbgisl	Cross-term dependence of bgisl
pcgisl	Cross-term dependence of cgisl
pegisl	Cross-term dependence of egisl

paigc	Cross-term dependence of aigc
pbigc	Cross-term dependence of bigc
pcigc	Cross-term dependence of cigc
paigsd	Cross-term dependence of aigsd
pbigsd	Cross-term dependence of bigsd
pcigsd	Cross-term dependence of cigsd
paigs	Cross-term dependence of aigs
pbigs	Cross-term dependence of bigs
pcigs	Cross-term dependence of cigs
paigd	Cross-term dependence of aigd
pbigd	Cross-term dependence of bigd
pcigd	Cross-term dependence of cigd
paigbacc	Cross-term dependence of aigbacc
pbigbacc	Cross-term dependence of bigbacc
pcigbacc	Cross-term dependence of cigbacc
paigbinv	Cross-term dependence of aigbinv
pbigbinv	Cross-term dependence of bigbinv
pcigbinv	Cross-term dependence of cigbinv
pnigc	Cross-term dependence of nigc
pnigbinv	Cross-term dependence of nigbinv
pnigbacc	Cross-term dependence of nigbacc
pntox	Cross-term dependence of ntox
peigbinv	Cross-term dependence for eigbinv
ppigcd	Cross-term dependence for pigcd
ppoxedge	Cross-term dependence for poxedge
pvfbcv	Cross-term dependence of vfbcv
pvfb	Cross-term dependence of vfb
pacde	Cross-term dependence of acde
pmoin	Cross-term dependence of moin
pnoff	Cross-term dependence of noff
pvoffcv	Cross-term dependence of voffcv
pxrcrg1	Cross-term dependence of xrcrg1
pxrcrg2	Cross-term dependence of xrcrg2
plambda	Cross-term dependence of lambda
pvtl	Cross-term dependence of vtl
pxn	Cross-term dependence of xn
peu	Cross-term dependence of eu
pucs	Cross-term dependence of ucs
pvfbsdoff	Cross-term dependence of vfbsdoff
ptvfbsdoff	Cross-term dependence of tvfbsdoff
ptvoff	Cross-term dependence of tvoff
saref	Reference distance between OD edge to poly of one side
sbref	Reference distance between OD edge to poly of the other side
wlod	idth parameter for stress effect
ku0	Mobility degradation/enhancement coefficient for LOD

kvsat	Saturation velocity degradation/enhancement parameter for LOD
kvth0	Threshold degradation/enhancement parameter for LOD
tku0	Temperature coefficient of KU0
llodku0	Length parameter for u0 LOD effect
wlodku0	Width parameter for u0 LOD effect
llodvth	Length parameter for vth LOD effect
wlodvth	Width parameter for vth LOD effect
lku0	Length dependence of ku0
wku0	Width dependence of ku0
pku0	Cross-term dependence of ku0
lkvth0	Length dependence of kvth0
wkvth0	Width dependence of kvth0
pkvth0	Cross-term dependence of kvth0
stk2	K2 shift factor related to stress effect on vth
lodk2	K2 shift modification factor for stress effect
steta0	eta0 shift factor related to stress effect on vth
lodeta0	eta0 shift modification factor for stress effect
web	Coefficient for SCB
wec	Coefficient for SCC
kvth0we	Threshold shift factor for well proximity effect
k2we	K2 shift factor for well proximity effect
ku0we	Mobility degradation factor for well proximity effect
scref	Reference distance to calculate SCA, SCB and SCC
wpemod	Flag for WPE model (WPEMOD=1 to activate this model)
lkvth0we	Length dependence of kvth0we
lk2we	Length dependence of k2we
lku0we	Length dependence of ku0we
wkvth0we	Width dependence of kvth0we
wk2we	Width dependence of k2we
wku0we	Width dependence of ku0we
pkvth0we	Cross-term dependence of kvth0we
pk2we	Cross-term dependence of k2we
pku0we	Cross-term dependence of ku0we
noia	Flicker noise parameter
noib	”Flicker noise parameter
noic	Flicker noise parameter
tnoia	Thermal noise parameter
tnoib	Thermal noise parameter
rnoia	Thermal noise coefficient
rnoib	Thermal noise coefficient
ntnoi	Thermal noise parameter
em	Flicker noise parameter
ef	Flicker noise frequency exponent
af	Flicker noise exponent
kf	Flicker noise coefficient

Example:

statement	*explanation*

`xx`
`.model xx`
	xx

5.5.4. JXX : JFETs

Syntax :

```
JXX [ND] [NG] [NS] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] [DEVICE TYPE] [MODEL PARAMS]
```

parameter	*description*
[ND]	Drain node/net
[NG]	Gate node/net
[NS]	Source node/net
[MODEL]	JFET model defined by a .model statement
[INLINE PARAMS]	*Inline parameters :* area : Area factor off : Initial condition for jfet state ic : Initial voltage conditions temp : Temperature
[DEVICE TYPE]	NJF or PJF

[MODEL PARAMS]	*description of model parameters :*
vt0	Threshold voltage
vto	Threshold voltage
beta	Transconductance parameter
lambda	Channel length modulation param.
rd	Drain ohmic resistance
gd	Drain conductance
rs	Source ohmic resistance
gs	Source conductance
cgs	G-S junction capactance
cgd	G-D junction cap
pb	Gate junction potential
is	Gate junction saturation current
fc	Forward bias junction fit parm.
b	Doping tail parameter
tnom	parameter measurement temperature

kf	Flicker Noise Coefficient
af	Flicker Noise Exponent

Example:

statement	*explanation*

`xx`
`.model xx`
	xx

5.5.5. ZXX : MESFETs

Syntax :

```
ZXX [ND] [NG] [NS] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] [DEVICE TYPE] [MODEL PARAMS]
```

parameter	*description*
[ND]	Drain node/net
[NG]	Gate node/net
[NS]	Source node/net
[MODEL]	MESFET model defined by a .model statement
[INLINE PARAMS]	*Inline parameters :* area : Area factor off : Initial condition for jfet state ic : Initial voltage conditions
[DEVICE TYPE]	NMF or PMF

[MODEL PARAMS]	*description of model parameters :*
vt0	Pinch-off voltage
vto	Pinch-off voltage
alpha	Saturation voltage parameter
beta	Transconductance parameter
lambda	Channel length modulation parm.
b	Doping tail extending parameter
rd	Drain ohmic resistance
gd	Drain conductance
rs	Source ohmic resistance
gs	Source conductance
cgs	G-S junction capacitance
cgd	G-D junction capacitance
pb	Gate junction potential

is	Junction saturation current
fc	Forward bias junction fit parm.
depl_cap	Depletion capacitance
vcrit	Critical voltage
kf	Flicker noise coefficient
af	Flicker noise exponent”)

Example:

statement	*explanation*

`xx`
`.model xx`
	xx

5.6. Other Elements

5.6.1. SXX / WXX : Switches

SXX : Voltage Controlled Switch:
Syntax :

```
SXX [N+] [N-] [NC+] [NC-] [MODEL] [INLINE PARAM]
.MODEL [MODEL] SW [MODEL PARAMS]
```

parameter	*description*
[N+]	Positive node/net
[N-]	Negative node/net
[NC+]	Positive controlling node/net
[NC-]	Negative controlling node/net
[MODEL]	Diode model defined by a .model statement
[INLINE PARAM]	on/off : Switch initial state
	Model parameters :
	vt : Threshold voltage
[MODEL PARAMS]	vh : Hysteresis voltage
	ron : On resistance
	roff : Off resistance

Example:

statement	*explanation*
s1 1 0 2 0 .model s1 sw vt=0 ron=1m roff=1meg	Between nets 1 and 0, insert an ideal switch that is controlled by voltage between 2 and 0. Short circuit resistance of the switch is 1mΩ and the open circuit resistance of the switch is 1MΩ. Switch closes for positive voltages, and stays open for negative voltages.

WXX : Current Controlled Switch:
Syntax :

```
WXX [N+] [N-] [VC] [MODEL] [INLINE PARAM]
.MODEL [MODEL] CSW [MODEL PARAMS]
```

parameter	*description*

[N+]	Positive node/net
[N-]	Negative node/net
[VC]	Voltage source whose current is used for control
[MODEL]	Diode model defined by a .model statement
[INLINE PARAM]	on/off : Switch initial state
[MODEL PARAMS]	*Model parameters :* it : Threshold current ih : Hysteresis current ron : On resistance roff : Off resistance

Example:

statement	*explanation*
w1 1 0 v1 .model s1 csw it=0 ron=1m roff=1meg	Between nets 1 and 0, insert an ideal switch that is controlled by current on v1. Short circuit resistance of the switch is 1mΩ and the open circuit resistance of the switch is 1MΩ. Switch closes for positive currents, and stays open for negative currents.

5.6.2. TXX / OXX / UXX / YXX /PXX : Transmission Lines

TXX : Lossless Transmission Lines:
Syntax :

```
TXX [N1] [N2] [N3] [N4] [INLINE PARAMS]
```

parameter	*description*
[N1]	Port 1 positive node/net
[N2]	Port 1 negative node/net
[N3]	Port 2 positive node/net
[N4]	Port 2 negative node/net
[INLINE PARAMS]	*Inline parameters :* zo : Characteristic impedance td : Delay time f : Frequency nl : Normalized length at frequency given ic : Initial voltage and current conditions

OXX : Lossy Transmission Lines:
Syntax :

```
OXX [N1] [N2] [N3] [N4] [MODEL] .MODEL [MODEL] LTRA [MODEL PARAMS]
```

parameter	*description*
[N1]	Port 1 positive node/net
[N2]	Port 1 negative node/net
[N3]	Port 2 positive node/net
[N4]	Port 2 negative node/net
[MODEL PARAMS]	*Model parameters :* r : Resistance per length l : Inductance per length g : Conductance per length c : Capacitance per length len : Length of line rel : Rel. rate of change of deriv. for bkpt abs : Abs. rate of change of deriv. for bkpt nocontrol : No timestep control flag steplimit : Always limit timestep to 0.8*(delay of line) flag nosteplimit : Don't always limit timestep to 0.8*(delay of line) flag lininterp : Use linear interpolation flag quadinterp : Use quadratic interpolation flag mixedinterp : Use linear interpolation if quadratic results look unacceptable flag truncnr : Use N-R iterations for step calculation in LTRAtrunc flag truncdontcut : Don't limit timestep flag compactrel : Special reltol for straight line checking flag compactabs : Special abstol for straight line checking flag

UXX : Uniform Distributed RC Lines:
Syntax :

```
UXX [N1] [N2] [N3] [MODEL] [INLINE PARAMS]
.MODEL [MODEL] URC [MODEL PARAMS]
```

parameter	*description*
[N1]	Port 1 node/net
[N2]	Port 2 node/net
[N3]	Capacitance node/net
[INLINE PARAMS]	*Inline parameters :* l : Length n : Number of lumped segments

`[MODEL PARAMS]`	*Model parameters :* k : Propagation constant fmax : Maximum frequency rperl : Resistance per length cperl : Capacitance unit length isperl : Saturation current per length rsperl : Diode resistance per length

YXX : Single Lossy Transmission Line:
Syntax :

```
YXX [N1] 0 [N2] 0 [MODEL] [INLINE PARAM]
.MODEL [MODEL] TXL [MODEL PARAMS]
```

Please check NGSPICE manual.

PXX : Coupled Multiconductor Line:
Syntax :

```
PXX [NI1] [NI2] [NI3] ...  [GND1] [NO1] [NO2] [NO3] ...  [GND2] [MODEL] [INLINE
.MODEL [MODEL] CPL [MODEL PARAMS]
```

Please check NGSPICE manual.

5.7. Built-in Subcircuits and Behavioral Elements in CoolSPICE

5.7.1. Probes

Probes, which are found in the Analog library of CoolSPICE, are described in detail in Section **??**, which is about their associated `.save` statement.

5.7.2. The `.include` Statement

5.7.3. The `.subckt` Statement and Subcircuit Definition

6. Thermal Simulations with CoolSPICE

CoolSPICE now supports thermal analysis for certain devices in a circuit. When considering thermal analysis, there are two sources of temperature effects that need to be modeled: *self heating* from a device, and *heat coupling* from other devices. While self heating models the heat generated by a device, heat coupling can model not only the temperature effects on a given device from other devices in the circuit (coupling), but also the thermal effects from intrinsic physical properties of the packaging of a device, or an attached heatsink, or any other external factors that may affect a device's temperature. As such, thermal circuits can be created in CoolSpice to model such factors, and solved concurrently with the electrical circuit.

Whether or not a thermal circuit is included in the netlist, the *junction temperature* (TJ), and the *case temperature* (TC) of a device can be plotted if thermal analysis is enabled. A thermal circuit may also be included in the netlist if desired. For example, a heatsink attached to a MOSFET can be modeled by a thermal circuit; however, there is an additional, simpler way to model a heatsink on a MOSFET if complex thermal circuits are unnecessary. Further information is provided in Section 1.2.1.

Thermal analysis for a device is calculated using parameters RTH/CTH. These parameters may be specified as model parameters within a model definition, or as instance parameters, included inline when an instance of one of the devices is called within the netlist. These parameters, RTH/CTH, represent a thermal resistor/capacitor between the internals of a device and the packaging, ie. between TJ and TC. If unrealistic values are provided for RTH/CTH (RTH ≤ 0, CTH < 0), CoolSPICE assumes TJ is shorted to TC.

If thermal analysis is enabled, proper RTH/CTH values are given, and only an electrical circuit is included in the netlist, then TJ is calculated using a device's electrical characteristics (power), but TC is assumed to be shorted to the circuit's ambient temperature. TC will only change if there is a thermal circuit.

6.1. Analyses

Thermal analysis is supported for transient and steady state (DC sweep) simulation. Although the parameters TJ/TC have the same interpretation - to set the initial junction or case temperature - for transient and DC sweep simulation, it is useful to remember how they are used in those simulations, as CoolSPICE may output plots that may not make sense at first. Further description is provided in the following sections.

6.1.1. Transient

Thermal effects for transient simulations are calculated from the previous timepoint. Thus, thermal analysis follows the "marching in time" algorithm for transient analysis. If a thermal circuit is included in the netlist, the thermal and electrical circuits are solved concurrently through the typical transient analysis algorithm. After a converging solution for the circuit has been found for the current timepoint, the calculated electrical characteristics (ie. the power of a device), are used to calculate new TJ/TC values. Lastly, if this new TJ has changed by a significant amount (1 degree) from the previous timepoint's TJ, the electrical characteristics are recalculated before moving on to the next timepoint.

For transient simulation, specifying TJ/TC amounts to setting the respective initial temperatures for that device. This can be thought of as setting the operating point values for TJ/TC. It is also possible to specify the difference between TJ and TC using the parameter dTJ.

6.1.2. DC Sweep

At any given sweep point in DC sweep, thermal calculations are run until the temperature reaches a steady state value. The electrical characteristics are first solved through the usual DC sweep algorithm, then the thermal effects are calculated based on this electrical solution. This process is repeated until the thermal convergence criteria is satisfied. As with transient analysis, thermal convergence is reached if the temperatures stop changing by more than 1 degree; if thermal analysis has not converged, the electrical characteristics are updated again before attempting to find a new solution.

The potential source of confusion here is how CoolSPICE uses the TJ/TC parameters with DC sweep, specifically when temperature is being swept. As with transient analysis, these parameters set the initial values for the temperatures. However, CoolSPICE might output unexpected (but logical) plots.

Consider the following partial netlist:

```
* DC Temperature Sweep *

.dc VSRC1 0 4 0.4 TEMP list 25 125 150
.option heaton
MOSFET1 D G S S BSM120D12P2C005 TJ=27 rd=0.01003 rg=3
+ rth=0.25 cth=4e-3
...
```

!!! CHECK THIS PARAGRAPH FOR CORRECTNESS !!!

Here we are sweeping a voltage source (not shown) at three temperatures, and we are specifying the initial TJ. When CoolSPICE runs this circuit, there will only be one visible curve for TJ, and a constant curve for TC. Although the user requested three voltage sweeps, each at a different temperature, because TJ was specified, the initial TC/TJ of the MOSFET at each of the three sweeps will be 27°C instead of the listed temperatures. TC is constant because there is no heatsink or thermal circuit of any kind, so it is shorted to the circuit temperature.

Below is a table that describes the result of different combinations of: "is thermal analysis enabled?", "is TJ specified?", "is TC specified?".

!!! FINISH THIS TABLE !!!

heaton	*TJ*	*TC*	*Result*
N	N	N	TJ = TC = CKTtemp
N	N	Y	TJ = TC, values are constant even if CKTtemp is swept
N	Y	N	TJ = TC, values are constant
N	Y	Y	TJ, TC are constant
Y	N	N	idk
Y	N	Y	idk
Y	Y	N	idk
Y	Y	Y	idk

6.2. Devices

Currently, CoolSPICE models thermal effects for the following devices:

- SiC/GaN MOSFETs
- Diodes
- Resistors

For all of these devices, "standard" thermal analysis has been implemented. This means that self heating effects can be modeled - where the electrical behavior of a device is used to calculate a dynamic TJ - as well as heat coupling effects - where a device can be used as a heat source in a thermal circuit. Recall that if there is no thermal circuit, then TC is assumed to be shorted to the ambient circuit temperature.

Specifically for MOSFETs, CoolSPICE also allows for more intricate *thermal models* that can model the effect of a simple heatsink attached to the MOSFET, or model the thermal effect from the internal diode of SiC MOSFETs.

6.2.1. MOSFETs

Users can choose a thermal model for SiC/GaN MOSFETs by specifying the model parameter `thermalmodel` in the model definition for a MOSFET. There are four thermal models for a heatsink, specified by an integer from 0 to 3:

0 - (MOSFET only) Assumes that the thermal circuit representing the MOSFET is a resistor and capacitor in parallel between TJ and TC.

1 - (MOSFET, heatsink) Assumes that there is an additional resistor and capacitor in parallel, between TC and the ambient temperature.

2 - (MOSFET, diode) Assumes that there is an additional resistor and capacitor in parallel, between TDIO (temperature due to the internal diode) and TC, and also that there is a resistor between TJ and TDIO.

3 - (MOSFET, heatsink, diode) Assumes that there is a heatsink and diode in the thermal circuit.

Note: `thermalmodel` applies to all instances of a MOSFET model, and cannot be individually specified for each instance in the circuit.

If the proper thermal model is specified, users may use the following parameters (as model or instance parameters) to further define the internal thermal circuit:

parameter	*description*
`TJ`	Initial junction temperature
`TC`	Initial case temperature
`TDIO`	Initial internal diode temperature
`RTH`	Value of the thermal resistance for MOSFET
`CTH`	Value of the thermal capacitance for MOSFET
`RTHHS`	Value of the thermal resistance for Heat Sink
`CTHHS`	Value of the thermal capacitance for Heat Sink
`RTHD`	Value of the thermal resistance for Diode
`CTHD`	Value of the thermal capacitance for Diode
`RTHDM`	Value of the thermal resistance for Diode and MOSFET thermal coupling

If any of the parameters are unrealistic (R ≤ 0, C < 0), or they are unspecified and the chosen thermal model expects values, the thermal model is adjusted to the correct one, and CoolSPICE assumes that the nets that the trouble thermal device bridges are shorted. For example, if thermal model is 1, and RTHHS ≤ 0 or CTHHS < 0, the thermal model is ajusted to 0 (MOSFET only), and TC is assumed to be shorted to the ambient temperature. Or if the thermal model is 3, and RTHDM is unspecified, TJ and

TDIO are assumed to be shorted.

Note: The effect of the heatsink can also be modeled through an external thermal circuit (refer to Section 1.3.1). The more significant application of these thermal models is to model the thermal effect of the internal diode of a MOSFET. However, using this representation of a heatsink may be a simple and fast way to "unshort" TC from the ambient temperature.

6.2.2. Diodes & Resistors

Diodes and resistors do not have any special thermal models. The only thermal effects for these devices are those calculated from the heat generated by the device. As such, if there is only an electrical circuit, TJ will be dynamic, but TC will be constant. However, if a thermal circuit is included, TC could potentially be dynamic as well. Refer to Section 1.3.1 for further information about thermal circuits.

Except for the parameters relating to thermal models, diodes and resistors have the same thermal parameters as MOSFETs - TJ, TC, RTH, and CTH.

6.3. Netlist Syntax

In order to first enable thermal analysis, the following option card must be included in the netlist:

```
.option heaton
```

Next, RTH/CTH for a device must be set to realistic values (RTH > 0, CTH ≥ 0). Otherwise, TJ will be shorted to TC. Recall that RTH/CTH can either be instance or model parameters. In this example, RTH/CTH are inline instance parameters for a CMF10120 MOSFET:

```
Mxxx D G S S CMF10120 TC=27 Rd=0.08 Rg=10 Rth=0.2 Cth=5e-5
```

Lastly, to save the trace of either TJ or TC of a device, a save card using the device instance parameter syntax should be included in the netlist (refer to Section **??**).

```
.save @Mxxx[TJ] @Mxxx[TC]
```

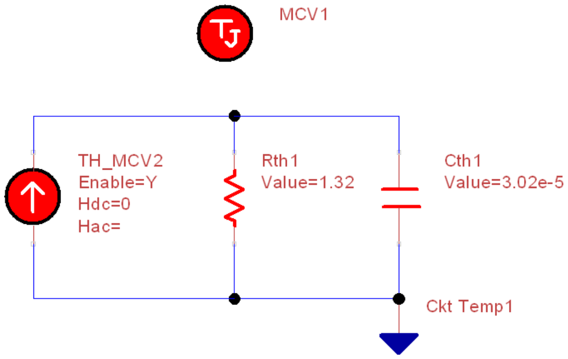
Note: Previously, the parameter TAMB was used where TC is in the above MOSFET card. Before thermal analysis was implemented, TAMB was interpreted as the case temperature of a device (not the ambient temperature). Now, to avoid confusion, we strongly encourage you to use the parameters TJ/TC to set a device's junction/case temperatures and ignore the TAMB parameter. Additionally, to achieve the effect of setting a device's initial case temperature, we even more strongly recommend using the TJ parameter rather than TC, as TC will be set to TJ if TJ is specified. Using TJ will ensure that you will get expected results.

6.3.1. Thermal Circuits

If a thermal circuit is not included in the netlist, standard thermal analysis will only show a dynamic TJ (refer to Section 1.2.1 for information about "nonstandard" thermal analysis for MOSFETs).

In order to simulate complex thermal effects for a given device, thermal circuits can be included in a netlist. Although the thermal and electrical circuits will logically be separate, CoolSPICE will treat them as a single circuit in order to solve for net values. In the electrical circuit, net values will be voltages, whereas in the thermal circuit, net values will be temperatures.

Previously, thermal circuits could not linked to the electrical device, ie. the heat generated by a device could not be used as a heat source in a thermal circuit. Now, the heat from a device can be used in a thermal circuit. For example, a heatsink on a MOSFET can be modeled on either using a MOSFET thermal model (Section 1.2.1), or by creating an external thermal circuit consisting of a thermal resistor/capacitor. The following schematic is an example of this heatsink scenario, using the heat generated by a device as the heat source.



The temperature at the net of the heat source linked to the MOSFET named MCV1 is TC. Without such a thermal circuit, TC would be shorted to the ambient circuit temperature (CktTemp1 in the schematic). However, since in this thermal circuit there is a resistor and capacitor representing a heatsink on the MOSFET, TC will be dynamic. In the SchematicsEditor, once a thermal and electrical circuit has been created, they will be automatically merged into a single netlist before being passed into the CoolSPICE engine.

When writing a netlist, rather than a schematic, the thermal and electrical netlists can be written in a single netlist, as long as the net names used in each are distinct from the other netlist, except for ground. Thus, the netlist will represent two circuits, connected only through ground. In order to include a heat source in the thermal part of the netlist, a current source (ISRC) can be used. However, the key part here is that the name of the ISRC must follow a special format so that CoolSPICE can link the heat source to the heat generated by a device in the electrical netlist. If the MOSFET in the electrical netlist has the name `MCV1`, then the current source linked to that MOSFET must have the name `ITH_MCV1`. Note the `TH` after the `I`, indicating that the current source is in fact a thermal heat source, and the `_MCV1`, which specifies the electrical device from which the heat is generated.

The following is a netlist for a Boost Converter using a CMF10120 SiC MOSFET, including a thermal circuit representing a heatsink on the MOSFET. The schematic can be found in the CoolSPICE example SiC circuits as ”HP_SiC_Cree_Boost.dsn”. The thermal circuit is the schematic shown above.

```
* Schematics Netlist *
.include "C:\Users\Simon_000\Documents\CoolSpice\Models\SiCMOS.txt"
.include "C:\Users\Simon_000\Documents\CoolSpice\Models\sic_schottky.txt"

.tran 100n 300u
.param gnd_N_9=0

MCV1 Drain _N_6 0 0 CMF10120 Tj=27 Rd=0.08 Rg=13.6 Rth=0.66 Cth=1.51e-5

CC1 0 Out 2u
DD1 _N_7 Out CSD10120
VIL _N_4 Drain 0

LL1 _N_4 _N_5 25u
RR1 _N_7 Drain 0.03
RRL1 0 Out 60

VVPul1 _N_6 0 pulse(-5 20 0 0.01u 0.01u 0.5u 1u)
VVS1 _N_5 0 DC 100

.save v(Out)
.save i(VIL)

* thermal heatsink *
CCth1 gnd_N_9 _N_8 3.02e-5
RRth1 gnd_N_9 _N_8 1.32
ITH_MCV1 gnd_N_9 _N_8 DC 0

* thermal only
.options heaton

.save @MCV1[Tj]
.save @MCV1[Tc]

.end
```

6.3.2. Outputting TJ/TC in PSIM

7. Quick Reference: Device Parameters and Commands

7.1. Circuit Elements

7.2. Sources

7.3. Probes, Analyses and SPICE Command Elements

7.4. References and Further Reading