

MicroSim PSpice Optimizer

Analog Performance Optimization Software

User's Guide


MicroSim[®]
MicroSim Corporation
20 Fairbanks
Irvine, California 92618
(714) 770-3022

Version 8.0, June, 1997.

Copyright 1997, MicroSim Corporation. All rights reserved.
Printed in the United States of America.

TradeMarks

Referenced herein are the trademarks used by MicroSim Corporation to identify its products. MicroSim Corporation is the exclusive owners of "MicroSim," "PSpice," "PLogic," "PLSyn."

Additional marks of MicroSim include: "StmEd," "Stimulus Editor," "Probe," "Parts," "Monte Carlo," "Analog Behavioral Modeling," "Device Equations," "Digital Simulation," "Digital Files," "Filter Designer," "Schematics," "PLogic," "PCBoards," "PSpice Optimizer," and "PLSyn" and variations thereon (collectively the "Trademarks") are used in connection with computer programs. MicroSim owns various trademark registrations for these marks in the United States and other countries.

SPECCTRA is a registered trademark of Cooper & Chyan Technology, Inc.

Microsoft, MS-DOS, Windows, Windows NT and the Windows logo are either registered trademarks or trademarks of Microsoft Corporation.

Adobe, the Adobe logo, Acrobat, the Acrobat logo, Exchange and PostScript are trademarks of Adobe Systems Incorporated or its subsidiaries and may be registered in certain jurisdictions.

EENET is a trademark of Eckert Enterprises.

All other company/product names are trademarks/registered trademarks of their respective holders.

Copyright Notice

Except as permitted under the United States Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a data base or retrieval system, without the prior written permission of MicroSim Corporation.

As described in the license agreement, you are permitted to run one copy of the MicroSim software on one computer at a time. Unauthorized duplication of the software or documentation is prohibited by law. Corporate Program Licensing and multiple copy discounts are available.

Technical Support

Internet Tech.Support@microsim.com

Phone (714) 837-0790

FAX (714) 455-0554

WWW <http://www.microsim.com>

Customer Service

Internet Sales@MicroSim.com

Phone (714) 770-3022

Contents

Before You Begin

- Welcome to MicroSim xiii
- MicroSim PSpice Optimizer Overview xiv
- How to Use this Guide xv
 - Typographical Conventions xv
- Related Documentation xvi
- If You Have the Evaluation Version xvii

Chapter 1 Things You Need to Know

- Chapter Overview 1-1
- What is the PSpice Optimizer? 1-2
 - Designs that You Can Optimize 1-3
 - Designs that You Cannot Optimize 1-3
- Using the PSpice Optimizer with Other MicroSim Programs 1-4
- Terms You Need to Understand 1-5

Chapter 2 Primer: How to Optimize a Design

- Chapter Overview 2-1
- Optimizing a Diode Biasing Circuit—the Objective 2-2
- Why Use Optimization? 2-3
- Phase One: Developing the Design 2-4
 - The PSpice Optimizer Advantage 2-5
- Phase Two: Setting Up the Optimization 2-6
 - Defining Design Parameters 2-7
 - Setting Up Goals and Constraints 2-8
 - Setting up analyses for each goal and constraint 2-8
 - Developing performance measures 2-9
 - Defining specifications: goals and constraints 2-10
- Phase Three: Running an Optimization 2-11
 - Running the PSpice Optimizer 2-12
 - Adding a Constraint and Rerunning the PSpice Optimizer 2-14

Changing the Constraint and Rerunning the PSpice Optimizer	2-16
Using Standard Component Values	2-18
Producing Reports	2-18
Saving Results	2-19
Updating the Schematic	2-19

Chapter 3 Using the PSpice Optimizer

Chapter Overview	3-1
Activating and Loading the PSpice Optimizer	3-2
Activating the PSpice Optimizer	3-2
From Schematics	3-2
From the Windows 95 Start Menu	3-3
Changing Activation Options	3-3
Loading a Different Optimization File	3-4
The PSpice Optimizer Window	3-5
Specifications Area	3-6
Internal specifications	3-6
External specifications	3-7
Parameters Area	3-8
Error Gauge Area	3-9
Adding and Editing Parameters	3-10
Adding a Parameter	3-10
Selecting a Parameter to Edit	3-12
Adding and Editing Specifications	3-13
Adding a Specification	3-13
Defining an Evaluation for an External Specification	3-17
Selecting a Specification to Edit	3-18
Measuring and Optimizing Performance	3-18
Optimizing Your Design	3-18
Graphically Monitoring Progress	3-19
Exploring the Effect of Parameter and Specification Changes	3-20
Testing Performance when Changing Current Values	3-21
Automatically recalculating performance	3-22
Manually recalculating performance	3-23
Ensuring reliable results when tweaking values	3-24
Excluding Parameters and Specifications from Optimization	3-24
Testing Performance when Adding or Changing Parameters or Specifications	3-25
Saving Intermediate Values	3-26
Viewing Result Summaries	3-26
Producing Optimization Reports	3-26

Viewing the Optimization Log	3-28
Viewing Derivatives	3-28
Finalizing the Design	3-29
Using Standard Component Values	3-29
Saving Results	3-30
Updating the Schematic	3-31

Chapter 4 Understanding Optimization Principles and Options

Chapter Overview	4-1
Goals versus Constraints	4-2
Constrained Optimization	4-3
Types of Constraints	4-4
Feasible and Infeasible Points	4-5
Active and Inactive Constraints	4-6
Lagrange Multipliers	4-6
Characteristics of Functions	4-7
Global and Local Minima	4-8
Starting Points	4-8
Convergence	4-9
Parameter Bounds	4-9
Derivatives	4-10
How the PSpice Optimizer Estimates Derivatives	4-10
Limitations of Derivative Data	4-11
Target Value Scaling	4-12
Default Options	4-13
Controlling Finite Differencing when Calculating Derivatives (Delta Option)	4-13
Limiting Simulation Iterations (Max. Iterations Option)	4-15
Specifying a Probe Display (Probe File and Display Options)	4-16
Advanced Options	4-17
Controlling Cutback (Cutback Option)	4-17
Controlling Parameter Value Changes Between Iterations (Threshold Option)	4-17
Choosing an Optimization Method for Single Goal Problems (Least Squares/Minimization Options)	4-19

Chapter 5 Tutorial: Optimizing a Design (Passive Terminator)

Tutorial Overview	5-1
The Passive Terminator Design	5-2
Loading the Design into Schematics	5-3
Setting Component Values to Expressions	5-4

Defining Optimization Parameters	5-5
Defining the Analysis Type	5-6
Running an Initial Circuit Analysis	5-6
Activating the PSpice Optimizer	5-7
Viewing the Parameter Description	5-8
Defining the Goals and Constraints	5-9
Checking that the Design Will Simulate	5-11
Starting the Optimization	5-11
Changing a Goal to a Constraint	5-13
Saving Results	5-13

Chapter 6 Tutorial: Exploring Design Tradeoffs (Active Filter)

Tutorial Overview	6-1
The Active Filter Design	6-2
The Parameters	6-3
The Goals	6-3
Testing Performance	6-5
Calculating Derivatives	6-5
Tweaking Parameters	6-6
Tweaking Goals and Constraints	6-7
Completing Optimization	6-8

Chapter 7 Tutorial: Using Constrained Optimization (MOS Amplifier)

Tutorial Overview	7-1
The CMOS Amplifier Design	7-2
The Parameters	7-3
The Evaluations	7-4
The Goals and Constraints	7-6
Setting the Method for a Single-Goal Optimization	7-7
Performing the Optimization	7-8

Chapter 8 Tutorial: Fitting Model Data (Bipolar Transistor)

Tutorial Overview	8-1
Using the PSpice Optimizer to Fit Data to Model Parameters	8-2
The Bipolar Transistor Test Case	8-3
The Parameters	8-4
The Analysis	8-5
The External File of Measured Data	8-5
The Goals and Constraints	8-6
Monitoring Progress with Probe	8-8

Fitting the Data 8-10

Appendix A Error Messages

Appendix Overview A-1
 Error Message Descriptions A-2

**File Types Used by
 Appendix B the PSpice Optimizer**

Appendix Overview B-1
 File and Program Relationships B-2
 Measuring Performance Using Information in the Circuit File and .prb File B-4
 Defining Specification Criteria in the External Data File B-5
 File Type Summary B-6

**Optimizing a
 Appendix C Netlist-Based Design**

Appendix Overview C-1
 Optimizing without a Schematic C-2
 Setting Up the Circuit File C-3
 Setting Up and Running the PSpice Optimizer C-4
 Example: Parameterizing the Circuit File C-6

Index

Figures

Figure 1-1	Optimization Design Flow	1-4
Figure 2-1	Diode Biasing Design Example	2-2
Figure 2-2	“Phase One: Developing the Design” Design Flow	2-4
Figure 2-3	“Phase Two: Setting Up the Optimization” Design Flow	2-6
Figure 2-4	“Phase Three: Running an Optimization” Design Phase	2-11
Figure 2-5	PSpice Optimizer Automatic Optimization Process	2-13
Figure 2-6	Optimization Results for the Diode Design Example	2-14
Figure 2-7	Results after Adding the Power Constraint	2-16
Figure 2-8	Results after Changing the Constraint Type	2-17
Figure 2-9	Report Summary for the Diode Optimization	2-19
Figure 2-10	Updated Diode Schematic	2-20
Figure 3-1	The PSpice Optimizer Window	3-5
Figure 3-2	Example of a Specification Box	3-6
Figure 3-3	Example of a Parameter Box	3-8
Figure 3-4	Sample Format for an External Specification	3-14
Figure 3-5	Sample Excerpt from a Report	3-27
Figure 3-6	Sample Excerpt from a Log File	3-28
Figure 3-7	Sample Derivative Data	3-28
Figure 4-1	Resistive Terminator Circuit	4-2
Figure 4-2	Global and Local Minima of a Function	4-8
Figure 4-3	Hypothetical Function	4-11
Figure 4-4	Hypothetical Data Glitch	4-18
Figure 5-1	Resistive Terminator Circuit	5-2
Figure 5-2	Schematic for the Terminator Example, term.sch	5-3
Figure 5-3	Optimization Results for the Passive Terminator Example	5-12
Figure 6-1	Schematic for the Active Filter Example, bpf.sch	6-2
Figure 6-2	Optimized Values for the Active Filter Example	6-8
Figure 7-1	Schematic for CMOS Amplifier Example, m2.sch	7-2
Figure 7-2	Updated Performance Values for the Amplifier Example	7-8
Figure 7-3	Optimized Values for the Amplifier Example	7-9
Figure 8-1	Schematic for the BJT Model Fitting Example	8-3
Figure 8-2	Initial Traces for the Ic and Ib Parameters	8-9

X Figures

Figure 8-3	Optimization Results for the BJT Model Fitting Example	8-11
Figure 8-4	Probe Display after Optimization is Complete	8-11
Figure 8-5	MicroSim Program and File Interactions Important to Optimization	B-2
Figure B-1	Sample External Data File	B-5

Tables

Table 1-1	Optimization Problems	1-5
Table 1-2	Valid Operators and Functions for PSpice Optimizer Expressions	1-11
Table 3-1	Edit Parameter Dialog Box Controls	3-11
Table 3-2	Edit Specification Dialog Box Controls	3-14
Table 8-1	Error Message Descriptions	A-2
Table 8-2	Summary of PSpice Optimizer-Related File Types	B-6

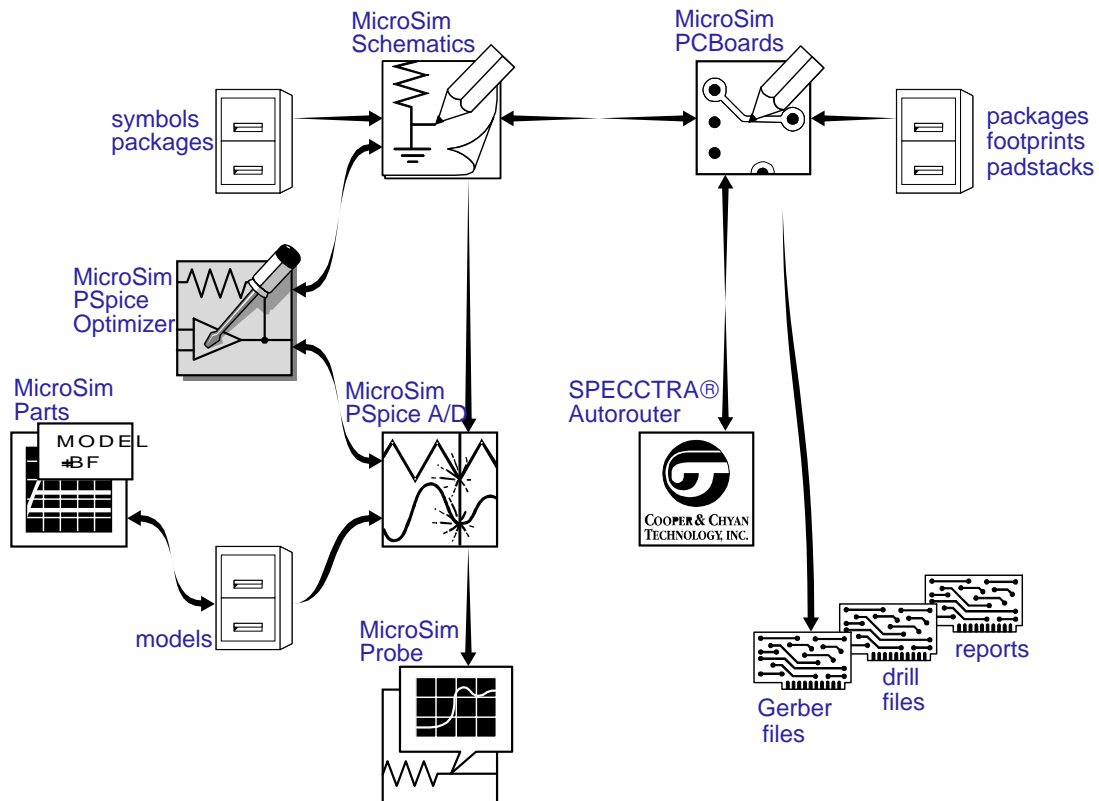
Before You Begin

Welcome to MicroSim

Welcome to the MicroSim family of products. Whichever programs you have purchased, we are confident that you will find that they meet your circuit design needs. They provide an easy-to-use, integrated environment for creating, simulating, and analyzing your circuit designs from start to finish.

MicroSim PSpice Optimizer Overview

The MicroSim PSpice Optimizer is a circuit optimization program that improves the performance of analog and mixed analog/digital circuits. The PSpice Optimizer is fully integrated with other MicroSim programs. This means you can design your circuit with MicroSim Schematics, simulate with MicroSim PSpice A/D (or MicroSim PSpice), analyze results with MicroSim Probe and optimize performance within the same environment.



How to Use this Guide

This guide is designed so you can quickly find the information you need to use the PSpice Optimizer.

This guide assumes that you are familiar with Microsoft Windows (NT or 95), including how to use icons, menus, and dialog boxes. It also assumes you have a basic understanding about how Windows manages applications and files to perform routine tasks, such as starting applications and opening, and saving your work. If you are new to Windows, please review your *Microsoft Windows User's Guide*.

Typographical Conventions

Before using the PSpice Optimizer, it is important to understand the terms and typographical conventions used in this documentation.

This guide generally follows the conventions used in the *Microsoft Windows 95 User's Guide*. Procedures for performing an operation are generally numbered with the following typographical conventions.

For UNIX users:

All screen captures in this manual are of Windows 95 dialog boxes and windows. Most options in these dialog boxes and windows are available in your operating environment. When certain options are not available to you, or you must do something differently than what is primarily outlined, information specific to your platform is provided.

Notation	Examples	Description
ALL CAPS	ANALOG.SLB or CLIPPER.SCH	Library files and file names.
Ctrl + R	Press Ctrl + R	A specific key or key stroke on the keyboard.
monospace font	Type VAC...	Commands/text entered from the keyboard.
Notation	Examples	Description

Related Documentation

Documentation for MicroSim products is available in both hard copy and online. To access an online manual instantly, you can select it from the Help menu in its respective program (for example, access the Schematics User's Guide from the Help menu in Schematics).

Note *The documentation you receive depends on the software configuration you have purchased.*

The following table provides a brief description of those manuals available in both hard copy and online.

This manual...	Provides information about how to use...
MicroSim Schematics User's Guide	MicroSim Schematics, which is a schematic capture front-end program with a direct interface to other MicroSim programs and options.
MicroSim PCBboards User's Guide	MicroSim PCBboards, which is a PCB layout editor that lets you specify printed circuit board structure, as well as the components, metal, and graphics required for fabrication.
MicroSim PSpice A/D & Basics+ User's Guide	PSpice A/D, Probe, the Stimulus Editor, and the Parts utility, which are circuit analysis programs that let you create, simulate, and test analog and digital circuit designs. It provides examples on how to specify simulation parameters, analyze simulation results, edit input signals, and create models.
MicroSim PSpice & Basics User's Guide	MicroSim PSpice & MicroSim PSpice Basics, which are circuit analysis programs that let you create, simulate, and test analog-only circuit designs.
MicroSim PLSyn User's Guide	MicroSim PLSyn, which is a programmable logic synthesis program that lets you synthesize PLDs and CPLDs from a schematic or hardware description language.
MicroSim FPGA User's Guide	MicroSim FPGA—the interface between MicroSim Schematics and XACTstep—with MicroSim PSpice A/D to enter designs that include Xilinx field programmable gate array devices.
MicroSim Filter Designer User's Guide	MicroSim Filter Designer, which is a filter synthesis program that lets you design electronic frequency selective filters.

The following table provides a brief description of those manuals available online *only*.

This online manual...	Provides this...
MicroSim PSpice A/D Online Reference Manual	Reference material for PSpice A/D. Also included: detailed descriptions of the simulation controls and analysis specifications, start-up option definitions, and a list of device types in the analog and digital model libraries. User interface commands are provided to instruct you on each of the screen commands.
MicroSim Application Notes Online Manual	A variety of articles that show you how a particular task can be accomplished using MicroSim's products, and examples that demonstrate a new or different approach to solving an engineering problem.
Online Library List	A complete list of the analog and digital parts in the model and symbol libraries.
MicroSim PCBoards Online Reference Manual	Reference information for MicroSim PCBoards, such as: file name extensions, padstack naming conventions and standards, footprint naming conventions, the netlist file format, the layout file format, and library expansion and compression utilities.
MicroSim PCBoards Autorouter Online User's Guide	Information on the integrated interface to Cooper & Chyan Technology's (CCT) SPECCTRA autorouter in MicroSim PCBoards.

If You Have the Evaluation Version

The evaluation version of the PSpice Optimizer has the following requirements and limitations:

- Requires the MicroSim PSpice A/D with Schematics evaluation package.
- Is limited to one goal, one parameter, and one constraint.

Things You Need to Know

1

Chapter Overview

This chapter introduces the purpose and function of the PSpice Optimizer, the optimization process, and related terms.

[What is the PSpice Optimizer? on page 1-2](#) describes optimizer capabilities and the criteria designs must meet for successful optimization.

[Using the PSpice Optimizer with Other MicroSim Programs on page 1-4](#) presents the high-level design flow for optimization and how other MicroSim programs are integrated into each design phase.

[Terms You Need to Understand on page 1-5](#) defines the terms that are important for optimizing designs successfully.

What is the PSpice Optimizer?

The MicroSim PSpice Optimizer is a circuit optimization program that improves the performance of analog and mixed analog/digital circuits.

Run optimizations The PSpice Optimizer performs iterative simulations, while adjusting the values of design parameters until performance goals, subject to specified constraints, are nearly or exactly met. Constraints can include simple bounds on parameter values and nonlinear functions. The PSpice Optimizer also computes Lagrange multipliers that provide information on the cost of each constraint on the solution.

Explore performance tradeoffs When you enter new values for design parameters, the PSpice Optimizer provides graphical feedback showing performance. You can also tweak goal and constraint values to examine changes to parameter values.

Fit model parameters Given a parameterized model, a set of measured data points, and a good starting point for the parameter values, the PSpice Optimizer fits a more accurate model.

Designs that You Can Optimize

A design that you can optimize must meet the following criteria:

- It works; that is, it simulates with PSpice to completion and behaves as intended.
- One or more of its components have a variable value, and each value that is varied relates to an intended performance goal.
- An algorithm exists to measure its performance as a function of the variable value.

If you can visualize what factors should be adjusted to improve performance, and how you would manually step through the optimization process (even though the computations might seem unwieldy), then the design is a good candidate for the PSpice Optimizer.

Optimization problems are not always solvable by a particular algorithm.

Designs that You Cannot Optimize

You cannot use the PSpice Optimizer to:

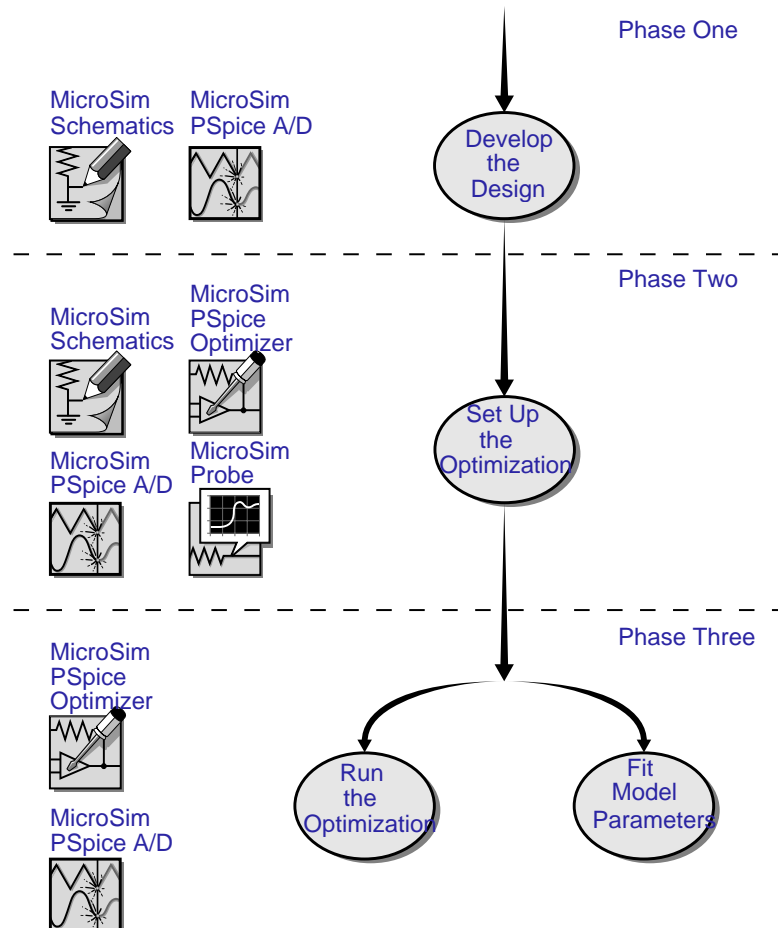
- Create a working design. This especially applies when you begin with a design that is far from meeting specifications.
- Optimize a design in which the circuit has several states where a small change in a parameter value causes a change of state.

Example: A flip-flop is *on* for some parameter value, and *off* for a slightly different value.

Using the PSpice Optimizer with Other MicroSim Programs

Because you can use Schematics, PSpice, and Probe to design and simulate at the system, subcircuit, or component level, use the PSpice Optimizer to optimize at whatever level is most appropriate.

The PSpice Optimizer is fully integrated with other MicroSim programs. This means you can design your circuit with MicroSim Schematics, simulate with MicroSim PSpice A/D (or MicroSim PSpice), analyze results with MicroSim Probe and optimize performance within the same environment. Figure 1-1 illustrates the typical design flow for circuit optimization.



See Chapter 2, “Primer: How to Optimize a Design” for a detailed description of each design phase.

Figure 1-1 Optimization Design Flow

Terms You Need to Understand

Optimization Optimization is the process of fine-tuning a design by varying design parameters between successive simulations until performance comes close to (or exactly meets) the ideal performance.

The PSpice Optimizer solves four types of optimization problems as described in Table 1-1.

Table 1-1 *Optimization Problems**

Problem Type	PSpice Optimizer Action	Example
unconstrained minimization	reduces the value of a single goal	minimize the propagation delay through a logic cell
constrained minimization	reduces the value of a single goal while satisfying one or more constraints	minimize the propagation delay through a logic cell while keeping the power consumption of the cell less than a specified value
unconstrained least squares**	reduces the sum of the squares of the individual errors (difference between the ideal and the measured value) for a set of goals	given a terminator design, minimize the sum of squares of the errors in output voltage and equivalent resistance
constrained least squares	reduces the sum of squares of the individual errors for a set of goals while satisfying one or more constraints	minimize the sum of squares of the figures of merit for an amplifier design while keeping the open loop gain equal to a specified value

*. All four cases allow simple bound constraints; that is, lower and upper bounds on all of the parameters. The PSpice Optimizer also handles nonlinear goals and constraints.

** . Use unconstrained least squares when fitting model parameters to a set of measurements, or when minimizing more than one goal.

Parameter A parameter defines a property of the design for which the PSpice Optimizer attempts to determine the best value within specified limits. A parameter can:

- Represent component values (such as resistance, R, for a resistor).
- Represent other component attribute values (such as slider settings in a potentiometer).
- Participate in expressions used to define component values or other component attribute values.

The PSpice Optimizer can optimize designs with up to eight variable parameters.

Example: A potentiometer symbol in a schematic uses the SET attribute to represent the slider position. You can assign a parameterized expression to this attribute to represent variable slider positions between 1 and 0. During optimization, the PSpice Optimizer varies the parameterized value of the SET attribute.

See [Chapter 6, Tutorial: Exploring Design Tradeoffs \(Active Filter\)](#) starting on page [6-1](#) for a working example showing parameterized slider values.

For more information, see *Goal* and [Constraint on page 1-8](#).

Specification A specification describes the ideal behavior of a design in terms of goals and constraints.

Examples: For a given design, the gain shall be 20 dB \pm 1 dB; for a given design, the 3 dB bandwidth shall be 1 kHz; for a given design, the rise time must be less than 1 usec.

A design can have up to eight goals and constraints in any combination, but there must *always* be at least one goal. You can easily change a goal to a constraint and vice-versa.

The PSpice Optimizer accepts specifications in two formats: internal and external.

Internal specifications

An internal specification is composed of goals and constraints defined in terms of target values and ranges, which are entered into the PSpice Optimizer through dialog boxes.

External specifications

An external specification is composed of measurement data, which are defined in an external data file that is read by the PSpice Optimizer.

Target value A target value is the ideal operating value for a characteristic of the design as defined by a goal or constraint specification.

Goal A goal defines the performance level that the design *should* attempt to meet (e.g., minimum power consumption). A goal specification includes:

- The name of the goal.
- A target value and an acceptable range.
- A circuit file to simulate.
- An evaluation for measuring performance.
- An analysis type used for simulation-based evaluations.

The goal specification can also include:

- The name of the file containing Probe goal function definitions (.prb file).
- When using an external specification, the name of the file containing measured data and the columns of data to be used as reference.

Note Typically, the PSpice Optimizer measures performance using an evaluation that requires a simulation, and therefore, you must specify the circuit file for the simulation. However, when measuring performance using PSpice Optimizer expressions which do not require a simulation, you do not need to specify a circuit file.

Constraint A constraint defines the performance level that the design *must* fulfill in which the target value exceeds, falls below, or equals a specified value (e.g., an output voltage that must be greater than a specific level). The constraint specification includes:

- The name of the constraint.
- A target value and an acceptable range.
- A circuit file to simulate. (See note on previous page.)
- An evaluation for measuring performance.
- An analysis type used for simulation-based evaluations.
- An allowed relationship between measured values and the target value, which can be one of the following:
 - <= measured value must be less than or equal to the target value
 - = measured value must equal the target value
 - >= measured value must be greater than or equal to the target value

The constraint specification can also include the name of the file containing Probe goal function definitions (.prb file).

Constraints are often nonlinear functions of the parameters in the design.

Example: Bandwidth can vary as the square root of a bias current and as the reciprocal of a transistor dimension.

Performance The performance of a design is a measure of how closely its specifications' calculated values approach their target values for a given set of parameter values. When there are multiple specifications (at least one of which is a goal), the PSpice Optimizer uses the sum of the squares of their deviations from target to measure closeness. For a single specification (goal), the PSpice Optimizer uses either the goal's value, or the square of its deviation from target.

See [Optimization on page 1-5](#) for more on least-squares and minimization algorithms.

Each aspect of a design's performance is found by either:

- first performing the appropriate simulation, then running Probe to measure characteristics of the resulting waveform(s), or
- evaluating PSpice Optimizer expressions.

In many cases (particularly if there are multiple conflicting specifications), it is possible that the PSpice Optimizer will not meet all of the goals and constraints. In these cases, optimum performance is the best *compromise* solution—that is, the solution that comes closest to satisfying each of the goals and constraints, even though it may not completely satisfy any single one.

Evaluation An evaluation is an algorithm that computes a single numerical value, which is used as the *measure of performance* with respect to a design specification.

The PSpice Optimizer accepts evaluations in one of these three forms:

- single-point Probe trace function
- Probe goal function
- PSpice Optimizer expression

Given evaluation results, the PSpice Optimizer determines whether or not the changes in parameter values are improving performance, and determines how to select the parameters for the next iteration.

Trace function A trace function defines how to evaluate a design characteristic when running a *single-point analysis* (such as a DC sweep with a fixed voltage input of 5 V).

Examples: V(out) to measure the output voltage; I(d1) to measure the current through a component.

Refer to the online *MicroSim PSpice A/D Reference Manual* for the variable formats and mathematical functions you can use to specify a trace function.

Refer to the Goal Function wizard in Probe and your PSpice user's guide for information on how to develop and specify goal functions.

Here are some quick tips. In Probe:

- To test the value returned by a specified goal function, select Eval Goal Function from the Trace menu.
- To see the waveforms and marked points used to evaluate a goal function, select Display Evaluation in the Probe Options dialog box (from the Tools menu, select Options to display this dialog box).

See [Gain on page 7-5](#) for an example of the YatX goal function definition.

Probe goal function A Probe goal function defines how to evaluate a design characteristic when running any kind of analysis other than a single-point sweep analysis. A goal function computes a single number from a Probe waveform. This can be done by finding a characteristic point (e.g., time of a zero-crossing) or by some other operation (e.g., RMS value of the waveform).

For example, you can use Probe goal functions to:

- Find maxima and minima in a trace.
- Find distance between two characteristic points (such as peaks).
- Measure slope of a line segment.
- Derive aspects of the circuit's performance which are mathematically described (such as 3 dB bandwidth, power consumption, and gain and phase margin).

To write effective goal functions, determine what you are attempting to measure, then define what is mathematically special about that point (or set of points).

Note *Be sure that the goal functions accurately measure what they are intended to measure. Optimization results highly depend on how well the goal functions behave. Discontinuities in goal functions (i.e., sudden jumps for small parameter changes) can cause the optimization process to fail.*

PSpice Optimizer expression A PSpice Optimizer expression defines a design characteristic. The expression is composed of optimizer parameter values, constants, and the operators and functions shown in Table 1-2.

Example: To measure the sum of resistor values for two resistors with parameterized values named R1val and R2val, respectively, use the PSpice Optimizer expression $R1val + R2val$.

Table 1-2 *Valid Operators and Functions for PSpice Optimizer Expressions*

Operator	Meaning
+	addition
-	subtraction
*	multiplication
/	division
**	exponentiation
exp	e^x
log	$\ln(x)$
log10	$\log_{10}(x)$
sin	sine
cos	cosine
tan	tangent
atan	arctangent

Note *Unlike trace functions and Probe goal functions, PSpice Optimizer expressions are evaluated without using a simulation or Probe.*

Derivative A derivative defines mathematically how a specification value changes with a small change in parameter value.

For a given design, the PSpice Optimizer calculates derivatives for each specification with respect to each parameter. Within an applicable range, the optimizer uses the derivatives to estimate new values for the goals and constraints.

See [Derivatives on page 4-10](#) for a detailed discussion.

Primer: How to Optimize a Design

2

Chapter Overview

This chapter guides you through the basic steps needed to setup and run an optimization using a simple diode biasing design.

[Optimizing a Diode Biasing Circuit—the Objective on page 2-2](#) describes the sample circuit and its ideal operating characteristics.

[Why Use Optimization? on page 2-3](#) explains why fine-tuning your design using the PSpice Optimizer saves time.

[Phase One: Developing the Design on page 2-4](#) walks you through the steps needed to create a working design.

[Phase Two: Setting Up the Optimization on page 2-6](#) walks you through the steps needed to define the parameters, goals, and constraints that describe the optimization.

[Phase Three: Running an Optimization on page 2-11](#) walks you through the steps needed to optimize and finalize the design.

Optimizing a Diode Biasing Circuit—the Objective

Assume that you want to design a circuit that drives a current of 1 mA ($\pm 5 \mu\text{A}$) through a diode (D1N914) using a 5 V voltage source and a series resistor to control the current through the diode. A circuit such as this is shown in Figure 2-1.

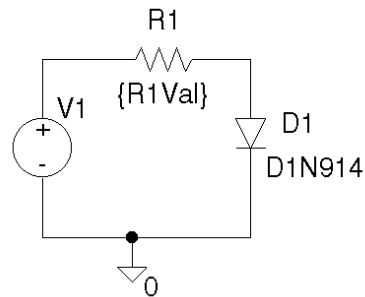


Figure 2-1 *Diode Biasing Design Example*

Your objective is to find a value for resistor R1 so that current through diode D1 falls in the range 0.995 mA to 1.005 mA.

Why Use Optimization?

To solve the problem manually, you could assign an arbitrary value to R1, manually calculate the current, then make an *educated guess* to adjust the values until a satisfactory solution is found. Or, you might use a simulation to sweep the value for R1 with a DC Sweep analysis, carefully analyzing the results to find the best solution.

These manual methods have two major disadvantages:

- Because the diode is a *non-linear* device, manual calculations can be time-consuming.
- Sweeping a parameterized value can take a large number of simulations, depending on the range and increment selected.

The PSpice Optimizer automates these processes by handling calculations for you and intelligently directing the series of simulations. Given results of the previous simulations, the optimizer automatically adjusts the parameterized value of R1 for the next run, thus eliminating unnecessary iterations, which in turn, provides a solution more quickly and with less effort.

Once the PSpice Optimizer settles on the best solution, you can still explore available tradeoffs. When done manually, this iterative process can be difficult and frustrating. With the optimizer, you can tweak the parameter(s) and immediately determine whether the design still meets specifications. You can also change the value of the specification(s) and immediately determine how parameter values change. If you are dissatisfied with the result after any change, you can always return to the last set of values.

When solving complex problems, the manual approach can be too unwieldy to consider. For example:

- When your design has multiple parameters or complicated parameter interactions, you may find it's nearly impossible to know which parameters to change, and how best to change them.
- When solving for multiple specifications, the solution often depends on the order in which goals and constraints are optimized. This sequential approach can miss possible solutions since it is impractical to repeat the process starting with a different goal or constraint each time.

Phase One: Developing the Design

Phase 1 is also the time to investigate:

- The effects of individual components by replacing component values with parameters or parameterized expressions.
- Using PSpice to perform a DC, AC, or parametric sweep of each parameterized value.

Before optimizing, you must have a *working* circuit. This means first drawing the schematic, then iteratively simulating with PSpice and adjusting the design until the circuit operates with the intended behavior.

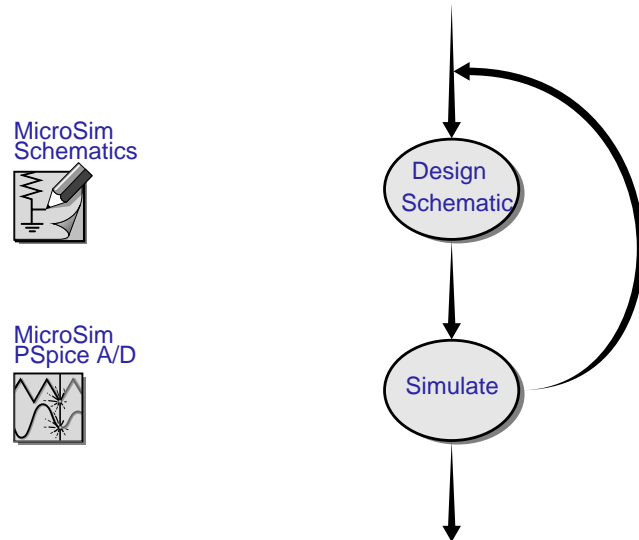


Figure 2-2 “Phase One: Developing the Design” Design Flow

To draw the schematic for the diode biasing design



Note When you initially place resistor R1, its value is 1k. Later, when you set up the optimization, you will parameterize R1’s value as shown in Figure 2-1.

- 1 In Schematics, from the Draw menu, use Get New Part to select and place the following symbols on the schematic:

R	resistor R1
D1N914	diode D1
VSRC	voltage source V1
AGND	analog ground 0

- 2 From the Draw menu, use Wire to connect the symbols as shown in Figure 2-1.
- 3 From the Edit menu, use Attributes to set V1's DC attribute to 5v.
- 4 From the File menu, select Save As and enter `mydiode.sch` in the File Name text box.



The PSpice Optimizer Advantage

To determine a value for R1 manually, you can set up a parametric analysis of a DC sweep where:

- the value of R1 steps from 4 k to 5 k in increments of 0.1 k, and
- the DC sweep analysis is a single-point voltage analysis at 5 V.

Such an analysis requires *eleven* PSpice simulations. Using Probe, the resistor value giving rise to 1 mA current through D1 is 4.14 k.

The remainder of this chapter shows how to use the PSpice Optimizer to determine the same solution automatically using *fewer* simulations.

Phase Two: Setting Up the Optimization

Now that preliminary design development is complete, you are ready to define the optimization parameters, goals, and constraints.

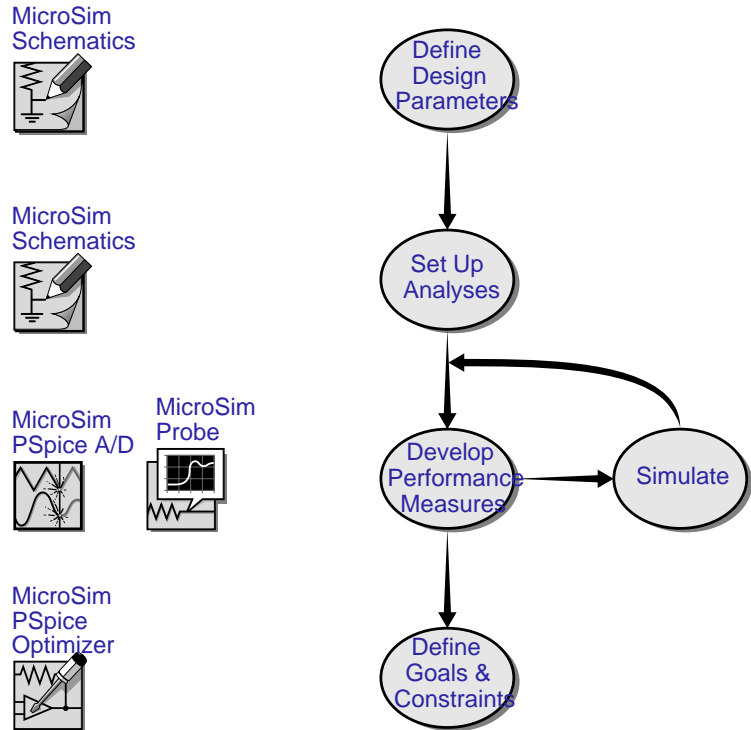


Figure 2-3 “Phase Two: Setting Up the Optimization” Design Flow

Defining Design Parameters

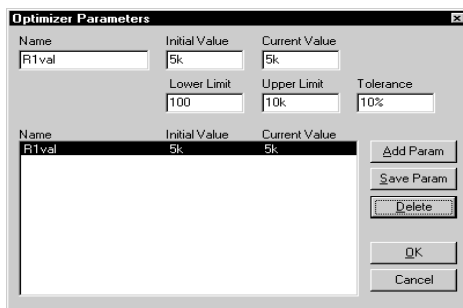
To define parameters for optimization, you must:

- Identify the parameters to adjust for optimization and assign a unique name to each one.
- Set up each parameter as a global optimization parameter using the Schematic Editor and the OPTPARAM symbol.
- Select which components in the design are affected by the parameter and, for each component, replace its value (e.g., the value of its VALUE attribute) with an expression that includes the parameter name.

To prepare the diode design example for optimization, you need to parameterize the value of R1 and specify its optimization properties.

To set up the value of R1 as a parameter named R1Val

- 1 In Schematics, from the Draw menu, use Get New Part to place an instance of the OPTPARAM symbol (from `special.slb`).
- 2 Double-click on the OPTPARAM symbol, then set R1val properties as shown in the Optimizer Parameters dialog box.



- 3 Click OK.
- 4 Double-click the 1k label for R1 and enter `{R1val}` to parameterize the value of R1.
- 5 Click OK.

You can also define optimization parameters in the PSpice Optimizer by selecting Parameters from the Edit Menu. See [Adding and Editing Parameters on page 3-10](#) for more information.



The parameter settings are:

Name = R1Val

Initial Value= 5k

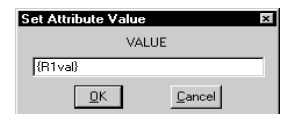
Current Value= 5k

Lower Limit= 100

Upper Limit= 10K

Tolerance= 10%

Later, in Schematics, when you select Run Optimizer from the Tune menu, the parameters specified with the OPTPARAM symbol are loaded into the PSpice Optimizer and displayed in its main window.



Setting Up Goals and Constraints

Before you can evaluate and improve the circuit's performance, you must answer these questions:

- What operating characteristics do I want to measure?
- How do the parameters affect the operating characteristics?

Once you've answered these questions, you are ready to:

- Set up the analyses needed to evaluate the performance measures.
- Develop the performance measure algorithms.
- Fully define the goals and constraints in terms of these performance measures and analyses.

Setting up analyses for each goal and constraint

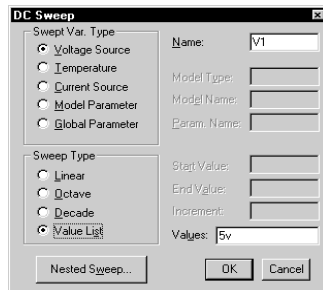
For each specification, you must define an analysis type: AC, DC, or transient. This is the analysis that PSpice will run to generate results used by the PSpice Optimizer to measure performance.

For the diode design example, you want to monitor the value of $I(D1)$ at a fixed input voltage of 5 V while the optimization parameter, $R1val$, is varied. This means setting up a single-point voltage sweep.

To set up a single-point voltage sweep analysis at 5 volts

- 1 In Schematics, from the Analysis menu, select Setup.
- 2 In the Analysis Setup dialog box, select the DC Sweep check box.

- a To fix the voltage of V1, fill in the DC Sweep dialog box as shown.



- b Click OK to return to the Setup Analysis dialog box.
- c Select the DC Sweep check box to enable it.
- 3 Clear all other analysis check boxes.
- 4 Click OK.

The DC Sweep settings are:

Swept Var Type = Voltage Source

Sweep Type= Value List

Name = V1

Values = 5v

Developing performance measures

To measure performance you must define an evaluation algorithm for each specification. There are three alternatives:

- trace function (for single-point simulations)
- Probe goal function
- PSpice Optimizer expression

When the evaluation is anything other than a single-point simulation or PSpice Optimizer expression, you must develop Probe goal functions to derive values from the simulation results. Developing goal functions is an iterative process that involves writing the goal function, simulating the design, and testing the goal functions against actual results to make sure you are measuring the waveform characteristics you intended.

A Probe goal function is not required for the diode design example. You are examining the trace of R1Val versus I(D1) which shows the relationship between the value of R1 and the diode forward current. Because only a single point on the curve is of interest, the trace function, I(D1), is the appropriate evaluation.

See [Evaluation on page 1-9](#) and the sections that follow for a definitions of trace function, Probe goal function, and PSpice Optimizer expression.

MicroSim supplies standard goal functions for AC, DC, and transient analyses in the file msim.prb. This file resides in the MicroSim root directory. You can add goal functions to this file, or create a local .prb file for use with a specific design.

See [Chapter 7, Tutorial: Using Constrained Optimization \(MOS Amplifier\)](#) for examples of Probe goal functions used to evaluate performance.

Refer to your PSpice user's guide for instructions on creating goal functions, and for a description of the global and local .prb files.

Defining specifications: goals and constraints

Now that you've completed the preliminary groundwork, you are ready to define the properties for goals and constraints. So far, you have performed all steps in Schematics. To finalize setup, you must specify the goal for the diode design example using the PSpice Optimizer.

To define the design goal, Id1, for the diode design example

- 1 In Schematics, from the Tools menu, select Run Optimizer to activate the PSpice Optimizer.

The PSpice Optimizer window appears showing the parameter R1val that you defined using the OPTPARAM symbol in the schematic.

- 2 In PSpice Optimizer, from the Edit menu, select Specifications.
- 3 In the Specifications dialog box, click Add.
- 4 Enter Id1 properties as shown in the following Edit Specification dialog box.

The goal specification settings are:

Name = Id1

Target = 1ma

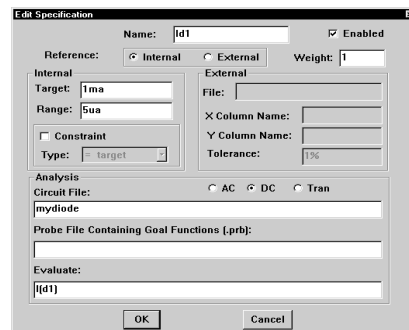
Range = 5ua

Analysis= DC

Circuit File= mydiode

Evaluate= I(d1)

The PSpice Optimizer appropriately defaults to the internal specification setting shown in the Reference control.



Phase Three: Running an Optimization

Now that you have defined the parameters, specifications, and evaluations for the design, you are ready to optimize, adjust, and finalize your design.

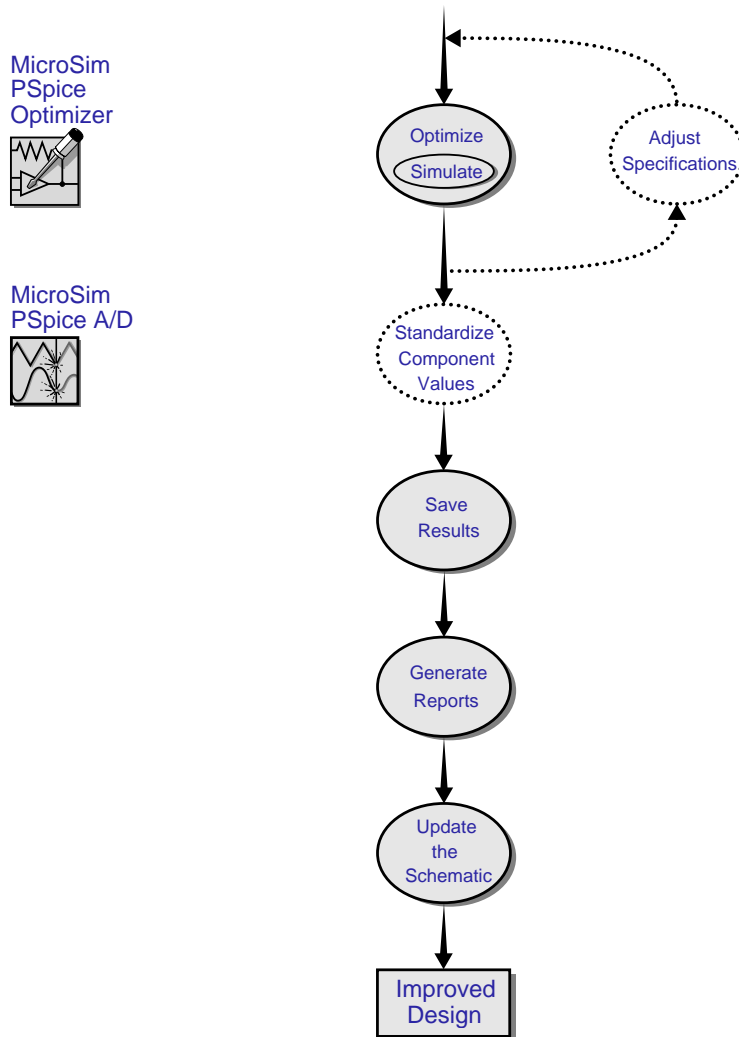


Figure 2-4 “Phase Three: Running an Optimization” Design Phase

Running the PSpice Optimizer

You can use the PSpice Optimizer to:

- Optimize the circuit to completion (from the Tune menu, select Auto).
- Evaluate performance for a single set of parameter values (from the Tune menu, select Update Performance).
- Compute derivatives of each specification with respect to each parameter (from the Tune menu, select Update Derivatives).

This is useful when initially validating the circuit or when restarting optimization after adjusting parameters or specifications.

This is useful when exploring design tradeoffs by tweaking parameter and specification values.

When you select Auto from the Tune menu, the PSpice Optimizer automatically computes the derivatives for each specification with respect to each parameter (Figure 2-5). Using the derivatives, the optimizer determines the direction in which to vary the parameters, and changes parameter values accordingly until it achieves a reduction in the overall error. After updating the parameters, the optimizer computes new derivatives and repeats the process until one of the following occurs:

- Specifications are met (success).
- No more progress can be made (failure).
- You manually interrupt the process.

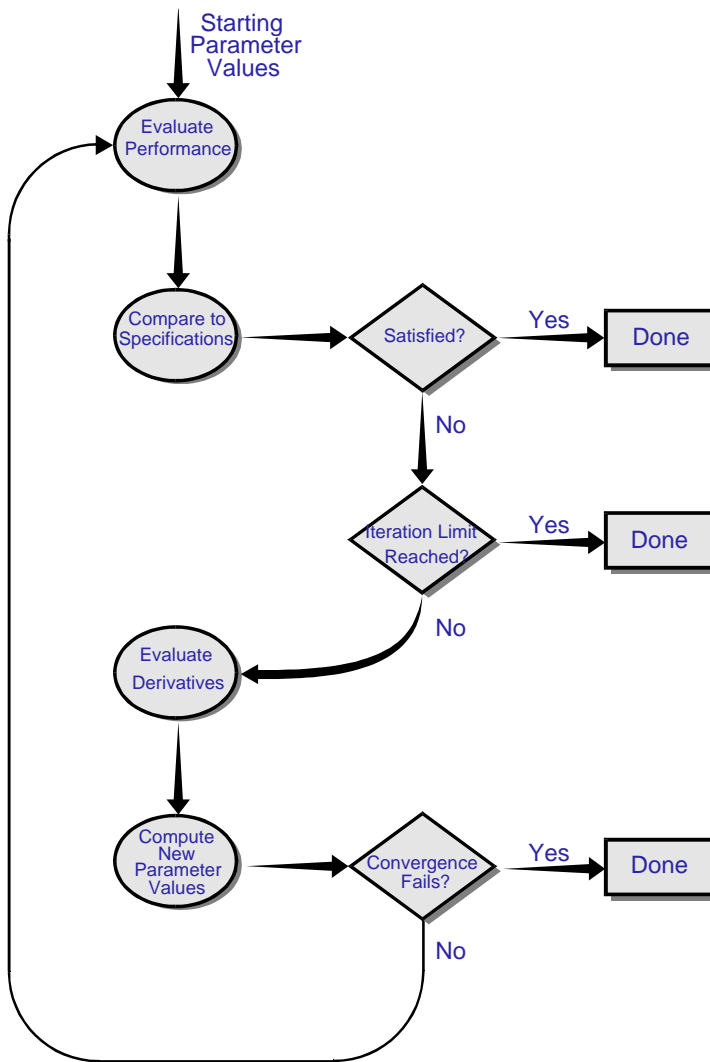


Figure 2-5 PSpice Optimizer Automatic Optimization Process

In the diode example, the derivative at $R1=5\text{ k}$ is -1.62×10^{-7} , or

$$\frac{\partial}{\partial R1}(I_{d1}) = -1.62 \times 10^{-7}$$

This indicates that a 1 ohm increase in $R1$ will produce a decrease of $0.162\text{ }\mu\text{A}$ in I_{d1} . To verify this, try reducing the value of $R1$ by 100 ohms (to 4.9 kohm) and simulate. This increases the diode current by $16.2\text{ }\mu\text{A}$ and agrees with the derivative information.

See [The PSpice Optimizer Window on page 3-5](#) for a complete description of the window elements and how you can interact with them.

To start optimizing the diode design

- 1 From the Tune menu, select Auto and click Start.

The PSpice Optimizer performs several simulations. For each iteration of the parameter values, the optimizer calculates overall performance and graphically displays the results. The optimizer also calculates the value of the trace function, $I(d1)$, and displays the new value in the specifications area of the PSpice Optimizer window. After three iterations, the optimizer should converge on a solution of 4.131 K , as shown in Figure 2-6.

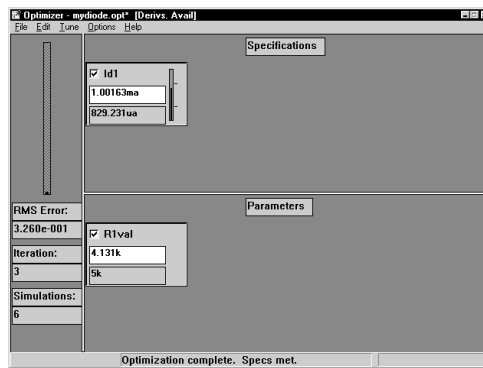


Figure 2-6 Optimization Results for the Diode Design Example

Adding a Constraint and Rerunning the PSpice Optimizer

So far, you have optimized for a single goal, I_{d1} . Now suppose you want to add a condition, or *constraint*, on the power dissipated in resistor $R1$.

Constraints are defined like goals (using the Edit Specification dialog box) with two additions. In the Internal frame, you must:

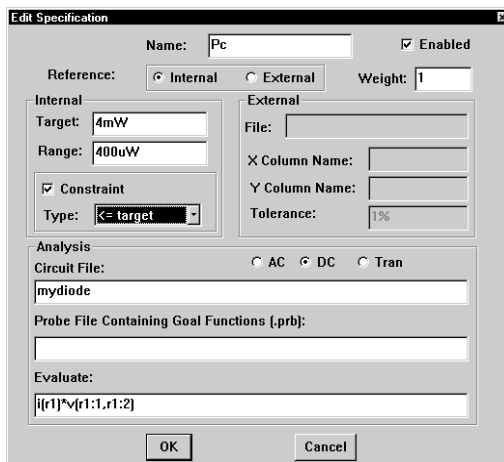
- Select the Constraint check box.
- Choose the constraint type ($>=$ target, $=$ target, or, $<=$ target).

The constraint type specifies the required relation between what is evaluated (as defined in the Evaluate text box) and the target value (defined in the Target text box).

To define the constraint for power dissipation in R1

The power dissipated in R1 must be less than or equal to 4 mW \pm 400 uW. Define the constraint by doing the following:

- 1 In PSpice Optimizer, from the Edit menu, select Specifications.
- 2 In the Specifications dialog box, click Add.
- 3 Enter the power (Pc) constraint properties as shown in the following Edit Specification dialog box.



The constraint specification settings are:

Name = Pc

Target = 4mW

Range = 400uW

Constraintselected

Type = <=target

Analysis= DC

Circuit File= mydiode

Evaluate= $i(r1)*v(r1:1,r1:2)$

The Evaluate text box contains the expression for measuring dissipated power. For each iteration, Probe will compute dissipated power by taking the product of the voltage across the resistance and the current through it.

- 4 To calculate the performance of the design for initial parameter and specification values only (one iteration):
 - a From the Edit menu, select Reset Values.
 - b From the Tune menu, select Update Performance.

Note the appearance of the progress indicator in the Pc box. Since Pc is a *less than or equal to* constraint, the



See [Progress indicator on page 3-7](#) for more on the different kinds of progress indicators and how to interpret them.

progress indicator has a tick mark 1/4 of the way up. The vertical bar within the indicator is below the tick mark; this means that the constraint is currently satisfied.

- From the Tune menu, select Auto and click Start to start optimization.

After a number of iterations, the optimization ends *without* satisfying the goal.

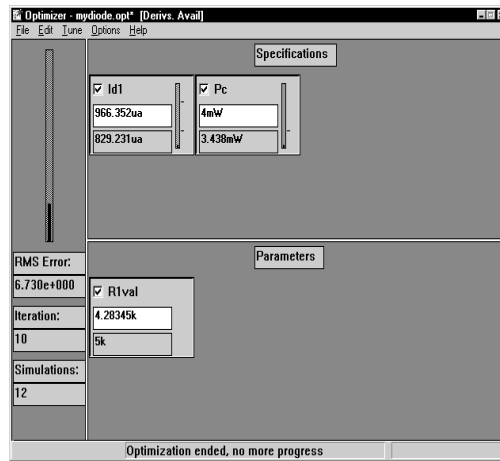


Figure 2-7 Results after Adding the Power Constraint

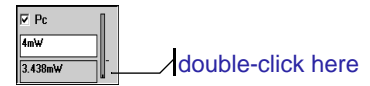
Note that the power dissipated in R1 is exactly equal to the target value of the constraint (4 mW). In this example there is no feasible solution to the problem. However, the PSpice Optimizer found the lowest value for Id1 which does not violate the constraint.

Changing the Constraint and Rerunning the PSpice Optimizer

You can examine the effect the Pc constraint has on performance by changing its constraint type so the power dissipation in the resistor must be *greater than or equal to* 4 mW.

To change the Pc constraint type to “greater than or equal”

- 1 In PSpice Optimizer, double-click the lower right-hand corner of the Pc box.
- 2 In the Edit Specifications dialog box, change Type to \geq target.
- 3 Click OK.



To run the optimization with the modified constraint

- 1 Test performance with the updated constraint:
 - a From the Edit menu, select Reset Values.
 - b From the Tune menu, select Update Performance.

The Pc constraint is initially violated because the power dissipation is less than 4 mW.

- 2 From the Tune menu, select Auto and click Start to start optimization.

The PSpice Optimizer finds a solution which satisfies both the goal (current of 1 mA) and the constraint (power dissipated in the resistor greater than 4 mW). Figure 2-8 shows the results.

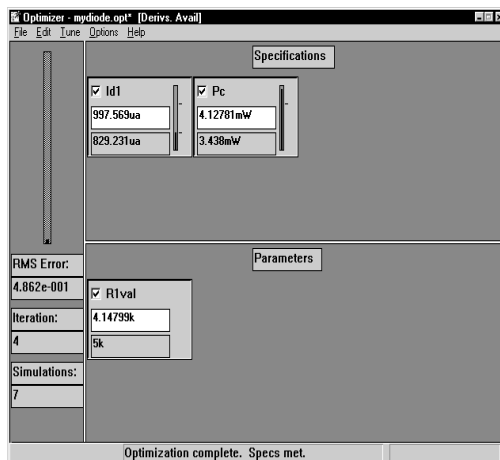


Figure 2-8 Results after Changing the Constraint Type

See [Using Standard Component Values on page 3-29](#) for more information, including how the PSpice Optimizer uses tolerances and limits when standardizing values.

Using Standard Component Values

When an optimization completes successfully, the optimizer displays the new parameter values in the PSpice Optimizer window. However, each calculated value might not correspond to an actual value that is available with off-the-shelf components. For example, resistors are not readily available in all possible values.

You can use the PSpice Optimizer to select standard component values. The optimizer either:

- rounds to the nearest value, or
- computes values based on the most recent optimization run.

To round to the nearest standard component value

- 1 From the Edit menu, select Round Nearest.

R1val's current value changes to 3.9 k in accordance with the specified 10% tolerance.



Producing Reports

You can use the PSpice Optimizer to generate a report summarizing:

- current settings for parameter, specification, and program options.
- calculated derivatives and Lagrange multipliers.

To generate a summary report

- 1 From the File menu, select Report.

The PSpice Optimizer writes the final results to `mydiode.oot` as shown in Figure 2-9.

```

PSpice Optimizer Report - mydiode.opt

*** Parameters: ***
Rival [cn]
current = 3.9k, initial = 5k, min = 100, max = 10k, tolerance = 10%

*** Specifications: ***
name: Id1 [cn], current value: 1.09687aa
weight: 1, satisfied: no
circuit: mydiode, analysis: DC, allsections: no
evaluate: i(d1)
using default goal functions file
reference: internal
type: eq, target: 1aa, range: 5ua [goal]

name: Pc [cn], current value: 4.53348aW
weight: 1, satisfied: yes
circuit: mydiode, analysis: DC, allsections: no
evaluate: i(r1)*v(r1.1.r1.2)
using default goal functions file
reference: internal
type: gt, target: 4aW, range: 400uW [constraint]

*** Options: ***
Delta: 1%
Threshold: 0
Cutback: 0.25
Max. Iterations: 20
One Spec: least squares
Recalculate: auto

*** Derivatives: ***
      Rival
Id1   -2.43e-007
Pc    9.36e-007

*** Lagrange Multipliers: ***
      Pc
Id1   0.0000e+000

```

Figure 2-9 Report Summary for the Diode Optimization

Saving Results

When you have finished optimizing, you can save all of the optimizer data, including the current values for all parameters and specifications.

To save the optimizer data for the diode design

- 1 From the File menu, select Save.

The PSpice Optimizer updates the `mydiode.opt` file. All of the options settings are also saved.

Updating the Schematic

Having completed the optimization, you can update the data in the schematic file to include the optimized parameter values.

To update the diode schematic with the current parameter values

- 1 From the Edit menu, select Update Schematic.

Recall that R1Val was initially set at 5.0 k in the schematic file. When you select Update Schematic, the PSpice Optimizer sends a message to Schematics to update the schematic file. Schematics writes the new parameter value of 3.9 k to the OPTPARAM symbol on the schematic (as the current value).

Figure 2-10 shows the updated schematic for the diode design.

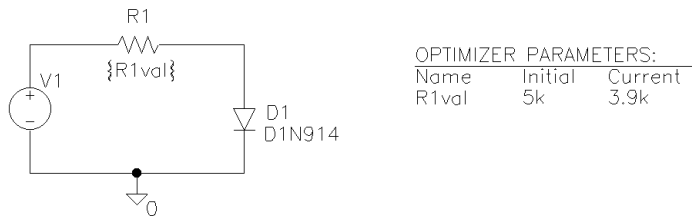


Figure 2-10 Updated Diode Schematic

Using the PSpice Optimizer

3

Chapter Overview

This chapter describes in general terms how to complete any task using the PSpice Optimizer, including:

- How to activate the PSpice Optimizer and load a design, *page [3-2](#)*.
- How to interact with the PSpice Optimizer window, *page [3-5](#)*.
- How to add and edit optimization parameters, *page [3-10](#)*.
- How to add and edit goals and constraints, *page [3-13](#)*.
- How to measure and optimize performance, *page [3-18](#)*.
- How to explore design tradeoffs, *page [3-20](#)*.
- How to generate result summaries including Lagrange multipliers and derivative values, *page [3-26](#)*.
- How to finalize the design: standardize component values, save results, and back-annotate the schematic, *page [3-26](#)*.

Activating and Loading the PSpice Optimizer

This section describes how to:

- Start the PSpice Optimizer.
- Set special startup options.
- Load a design.

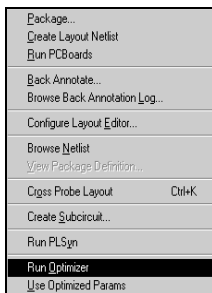
Activating the PSpice Optimizer

Start the PSpice Optimizer program either from:

- Schematics, or
- the Windows 95 Start menu.



Tools menu



From Schematics

To activate the PSpice Optimizer from Schematics

- 1 In Schematics, from the Tools menu, select Run Optimizer.

If you have an active schematic loaded into the Schematic Editor when you select Run Optimizer, any optimization parameters defined with the OPTPARAM symbol and any existing setup information contained in the corresponding optimization file (.opt) are automatically loaded into the PSpice Optimizer.

If no schematic is active, the PSpice Optimizer activates without an optimization setup. Instead, you must load an optimization file directly into the optimizer as described in [Loading a Different Optimization File on page 3-4](#).

From the Windows 95 Start Menu

From the Windows 95 Start menu, there is a program folder which contains Windows 95 shortcuts for all installed MicroSim programs, including the PSpice Optimizer.

To activate the PSpice Optimizer from the Windows 95 Start menu

- 1 From the Windows 95 Start menu, select the MicroSim program folder and then the PSpice Optimizer shortcut to start optimizer.

The optimizer activates without an optimization setup. See [Loading a Different Optimization File on page 3-4](#) for further instructions.

For UNIX users:

To activate the PSpice Optimizer on UNIX platforms, either double-click on the optimization file (.opt) in the File Manager window, or type “optimize” within a Shelltool window.

Changing Activation Options

The PSpice Optimizer supports two command line options, which are used to:

- Activate the optimizer with an initialization file other than the default (msim.ini).
- Automatically load an optimization file (.opt) after startup.

You can add one or both options to the command line.

To change the initialization file used by the PSpice Optimizer

- 1 In the optimizer command line, use the -i option as follows:

```
OPTIMIZE -i initialization_file_name
```

Because you can activate the PSpice Optimizer from either Schematics or from the Windows 95 Start menu, we recommend that you change *both* occurrences of the command line definitions as follows:

- Use a text editor to open the msim.ini file and, in the [MICROSIM OPTIONS] section, change the OPTIMIZECMD line.
- In the Windows Explorer, change the Target text box in the optimizer’s Properties dialog box (click once on the PSpice Optimizer shortcut, then, from the File menu, select Properties and click the Shortcut tab).

To automatically load an optimization file after startup

- 1 In the PSpice Optimizer command line, add the name of the optimization file as follows:

```
OPTIMIZE optimization_file_name
```

The following command line example shows how to start the optimizer *at all times* with an initialization file named `myinit.ini` and an optimization file named `mydesign.opt`:

```
OPTIMIZE mydesign.opt -i myinit.ini
```

Loading a Different Optimization File

Once you have activated the PSpice Optimizer, you can change to a new or different optimization file at any time.

To start a new optimization

- 1 From the File menu, select New.

To load an existing optimization setup

- 1 From the File menu, select Open.
- 2 Locate and select the appropriate optimization file.

The PSpice Optimizer Window

The PSpice Optimizer window contains three areas:

- specifications area
- parameters area
- error gauge area

Figure 3-1 illustrates their position in the window.

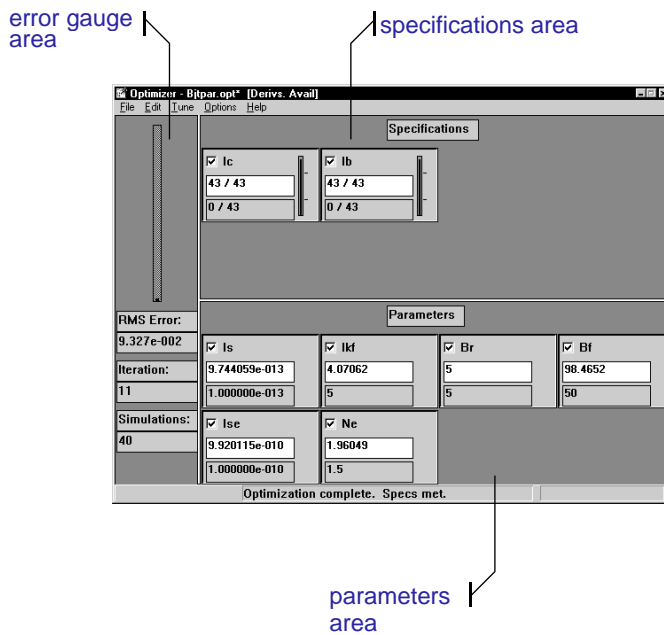


Figure 3-1 The PSpice Optimizer Window

Specifications Area

The specifications area can show up to eight specification boxes where each box represents either a goal or a constraint.

Figure 3-2 illustrates the fields contained in each box.

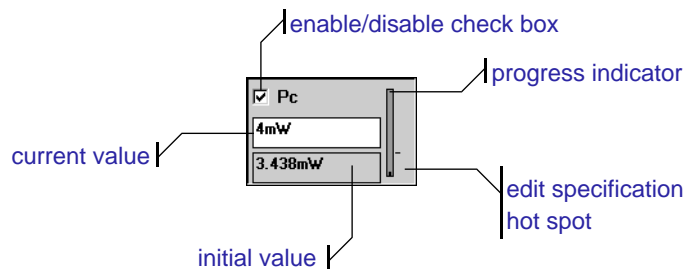


Figure 3-2 Example of a Specification Box

For more information on the enable/disable check box, see [Excluding Parameters and Specifications from Optimization on page 3-24](#). For more information on the edit hot spot, see [Selecting a Specification to Edit on page 3-18](#).

The contents of the box varies depending on the source for the specifications—either internal or external. The following sections describe the differences in the initial and current value fields, and the progress indicator.

Internal specifications

Initial value The initial value field displays a single performance measure that the PSpice Optimizer sets when you start an optimization. The optimizer derives this value from the initial optimization parameter values you defined in the schematic (or the optimizer).


If derivative data is available, you can change the value in this field to explore how parameter values might change. See [Testing Performance when Changing Current Values on page 3-21](#) for more information.

Current value The current value field displays the performance measure that corresponds to the current parameter values. Current values are updated each time an optimization iteration makes progress. When the current value satisfies the specification (that is, the current value is within the allowed range of the target) then the progress indicator turns from red to green, and the PSpice Optimizer considers the specification satisfied.


Progress indicator As mentioned above, the progress indicator shows red while the specification is violated, and changes to green when the specification is satisfied.

You can monitor progress as the optimization runs by watching the progress indicator and observing the height of the vertical black bar relative to the tick mark(s) to the right. The number and relative position of the tick mark(s) varies depending on the type of specification:


- Two tick marks for a goal or equality constraint denote the acceptable range around the target value.
- A single tick mark one-quarter of the way up denotes a *less than or equal to* constraint.
- A single mark three-quarters of the way up denotes a *greater than or equal to* constraint.



goal or equality
constraint



less than or
equal to constraint



greater than or
equal to constraint

External specifications

Initial value The initial value field is used in a different way from an internal specification: the field contains a pair of numbers separated by a '/' character. The first number is the number of subgoals in the external specification that are satisfied. The second number is the total number of subgoals in the external specification.

Current value The current value field is used in a different way from an internal specification: the field contains a pair of numbers as described above for initial value.

Progress indicator The progress indicator shows the fraction of subgoals that are satisfied. The indicator turns from red to green when all of the subgoals are satisfied.

Example: If an external specification has 20 subgoals and five are satisfied, the current value field reads 5 / 20 and the bar in the progress indicator rises to the one-quarter mark.

Parameters Area

The parameters area can show up to eight parameter boxes. Figure 3-2 illustrates the fields contained in each box.

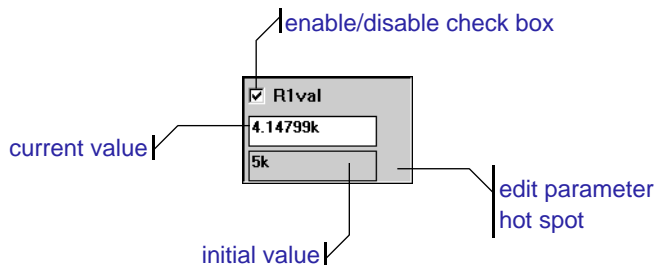


Figure 3-3 Example of a Parameter Box

For more information on the enable/disable check box, see [Excluding Parameters and Specifications from Optimization on page 3-24](#). For more information on the edit hot spot, see [Selecting a Parameter to Edit on page 3-12](#).

If derivative data is available, you can change the value in this field to explore how specification values might change. See [Testing Performance when Changing Current Values on page 3-21](#) for more information.

The following sections explain the initial and current value fields in the parameter box.

Initial value The initial value field displays a single value taken from the parameter specification that you set up either in Schematics (using the OPTPARAM symbol) or in the PSpice Optimizer (from the Edit menu, select Parameters).

Current value Initially, the current value is taken from the parameter specification you set up in either the schematic or the PSpice Optimizer. With each iteration of an optimization, the optimizer updates the current value field with the new set of parameter values.

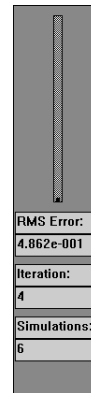
Error Gauge Area

The error gauge area has an error indicator and shows information reflecting the overall condition of the optimization.

The error gauge shows how far the goals are from their target values. Initially, the indicator displays 100%. As the optimization proceeds, the error falls as the goals approach their target values. The PSpice Optimizer displays the RMS of the normalized value of the goals in the RMS Error field.

If all of the specifications meet the target values exactly, the error is zero. If the optimization succeeds by finding a solution that works within the set ranges, but does not meet the target values exactly, the error represents the combined error of all of the specifications.

Note *The RMS error reflects only those goals that are enabled. The error does not include contributions from constraints, nor does it reflect any specifications that are disabled.*



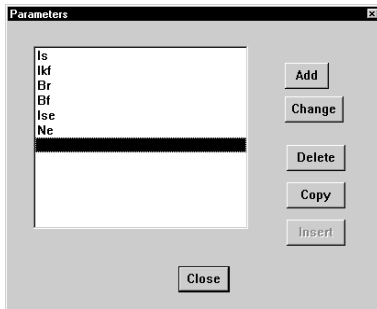
See [Target Value Scaling on page 4-12](#) for a discussion of normalization and scaling.

Adding and Editing Parameters

This section describes how to create and change optimization parameters using the PSpice Optimizer. This means you can also edit the properties of parameters that you defined in a schematic.

Note *You are limited to eight parameters per optimization file. This limit includes parameters that you have defined but disabled for a given run.*

You can also define parameters in Schematics using the OPTPARAM symbol. See [Defining Design Parameters on page 2-7](#) for an example.



Adding a Parameter

There are two ways to add a parameter using the PSpice Optimizer:

- Create the parameter from scratch.
- Copy an existing parameter and change its name.

To create a parameter from scratch

1 In the PSpice Optimizer, from the Edit menu, select Parameters.

Any optimization parameters that you have already established in the schematic appear in the selection list.

2 In the Parameters dialog box, either:

- click Add, or
- double-click the blank line following the parameter list.

- 3 In the Edit Parameter dialog box, set the controls as described in [Table 3-1](#).

Table 3-1 *Edit Parameter Dialog Box Controls*

Control Name	Meaning
Name	Parameter name; for a new parameter, double-click <<new>> and enter a text string that is unique to the current optimization file.
Current Value	Latest parameter value; for a new parameter, set Current Value equal to Initial Value.
Initial Value	Starting value for the parameter.
Upper Limit	Highest allowable value for the parameter.
Lower Limit	Lowest allowable value for the parameter.
Tolerance	Tolerance level (1%, 5%, etc.) to use when standardizing component values.
Enabled	When selected, includes the parameter in the next optimization run. If cleared, excludes the parameter.

To create a parameter based on an existing parameter

- 1 In the PSpice Optimizer, from the Edit menu, select Parameters.
- 2 Select the parameter that you want to copy and click Copy. Notice that the Insert button is now enabled (no longer grayed out)
- 3 Click Insert to add a copy of the selected parameter to the parameter list. The optimizer inserts the new parameter immediately above the highlighted parameter.
- 4 Select the new parameter in the list and click Change.
- 5 Change the Name text box to a unique name, and change any other controls as needed.

- When finished, click OK to return to the Parameters dialog box.

Note *You cannot copy a parameter description from one optimization file to another. Instead, save an entire optimization file under a new name, then edit the new version as needed.*

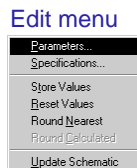
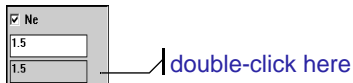
Selecting a Parameter to Edit

You can change the properties of an existing parameter at any time using the Edit Parameter dialog box.

To display the Edit Parameter dialog box for a parameter

- Do one of the following:

- In the parameters area of the PSpice Optimizer window, double-click the lower right-hand corner of the box for the parameter you want to edit.
- Select Parameters from the Edit menu, select the parameter you want to edit, and click Change.



Adding and Editing Specifications

This section describes how to create and change specifications—that is, goals and constraints—using the PSpice Optimizer.

Note *You are limited to eight specifications per optimization file. This limit includes specifications that you have defined but disabled for a given run.*

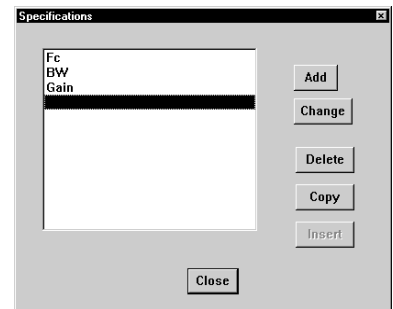
Adding a Specification

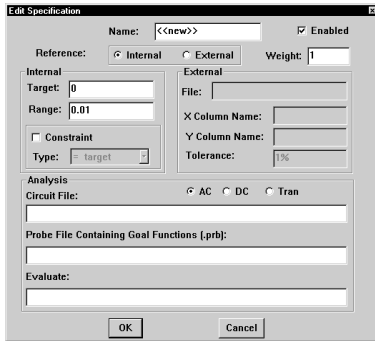
There are two ways to add a specification using the PSpice Optimizer:

- Create the specification from scratch.
- Copy an existing specification and change its name.

To create a specification from scratch

- 1 In the PSpice Optimizer, from the Edit menu, select Specifications.
- 2 In the Specifications dialog box, either:
 - click Add, or
 - double-click the blank line following the specification list.





3 In the Edit Specification dialog box, set the controls as described in [Table 3-2](#).

Table 3-2 Edit Specification Dialog Box Controls

Control Name	Meaning
Name	Specification name; for a new specification, double-click <<new>> and enter a text string that is unique to the current optimization file.
Reference	<ul style="list-style-type: none"> • Select Internal when defining the specification's target value and range in this dialog box. • Select External when defining the specification's measurement data using a file (e.g., for curve fitting).
Weight	Relative weight of the specification. To give a specification more weight, assign a number that is higher than that for the other specifications.
Enabled	When enabled, includes the specification in the next run. If cleared, excludes the specification.
Internal Specification	
Target	Ideal value for the specification.
Range	Delta applied to Target, defining the acceptable range of values; i.e., Target ± Range. Example: If Target=10 and Range=2, then the PSpice Optimizer accepts values from 9 to 11.
Constraint	When enabled, defines the specification as a constraint (not a goal).
Type	Defines whether constraint values must equal the target value, be less than or equal to the target value, or be greater than or equal to the target value.

X column name	Y column name
Vbe	Ic
4.0E-01	6.05E-06
4.1E-01	8.90E-06
4.2E-02	1.31E-05

Figure 3-4 Sample Format for an External Specification

External Specification

Table 3-2 *Edit Specification Dialog Box Controls*

Control Name	Meaning
File	Name of the file that contains the measured data.
X Column Name	Heading for the data column in File containing the independent (X) values.
Y Column Name	Heading for the data column in File containing the dependent (Y) values
Tolerance	Tolerance value used when standardizing component values. Syntax: $< 0 \leq \text{integer value} \leq 100 > \%$
Analysis Settings	
(Analysis Type)	Kind of analysis used for simulation-based evaluations. <ul style="list-style-type: none"> • Select AC for AC sweep analysis. • Select DC for a DC sweep analysis. • Select Tran for a transient analysis.
Circuit File	Set to either: <ul style="list-style-type: none"> • the name of the circuit file PSpice uses for simulation, or • leave blank if the Evaluate text box contains a PSpice Optimizer expression.
Evaluate*	Trace function, Probe goal function, or PSpice Optimizer expression used to measure performance.

*. See [Defining an Evaluation for an External Specification on page 3-17](#) for the purpose and use of the ‘!’ symbol in the Evaluate edit control.

To create a specification based on an existing specification

- 1 In the PSpice Optimizer, from the Edit menu, select Specifications.
- 2 Select the specification that you want to copy and click Copy.

Notice that the Insert button is now enabled (no longer grayed out)

- 3 Click Insert to add a copy of the selected specification to the parameter list.

The optimizer inserts the new specifications immediately above the highlighted specification.

- 4 Select the new specification in the list and click Change.
- 5 Change the Name text box to a unique name, and change any other controls as needed.
- 6 When finished, click OK to return to the Specifications dialog box.

Note *You cannot copy a specification description from one optimization file to another. Instead, save an entire optimization file under a new name, then edit the new version as needed.*

Defining an Evaluation for an External Specification

When defining evaluations for external specifications, use the ‘!’ symbol within the evaluation as a placeholder for data in the external file. For each of the subgoals in the external file, the PSpice Optimizer first replaces the ‘!’ character with the X column data value (defined in the X Column Name text box in the Edit Specification dialog box), and then proceeds with evaluation. This approach allows a Probe goal function or PSpice Optimizer expression to *track* the independent data value.

To set up an evaluation for an external specification

- 1 Create the Probe goal function or PSpice Optimizer expression.
- 2 In the Edit Specification dialog box, enter the evaluation in the Evaluate text box as follows:

Substitute the ‘!’ character for every X value argument; i.e., wherever a measured subgoal value should appear.

Example: Suppose that you want to fit a set of data measured at different values of Vtest (2.0, 2.5, and 3.0 volts), using the Probe goal function:

```
YatX(V(out), x_value)
```

to measure the output of the design. To use this with an external specification, replace *x_value* with the ‘!’ character and enter the evaluation:

```
YatX(V(out), !)
```

into the Evaluate text box for the specification.

As the fitting process proceeds, the PSpice Optimizer passes the following goal functions (one for each measured data point) to Probe:

```
YatX(V(out), 2.0)
```

```
YatX(V(out), 2.5)
```

```
YatX(V(out), 3.0)
```

After substitution, the PSpice Optimizer does one of the following depending on the kind of evaluation:

- For a Probe goal function, the optimizer sends the substituted goal function to Probe for evaluation.
- For a PSpice Optimizer expression, the optimizer evaluates the expression directly.

See [Chapter 8, Tutorial: Fitting Model Data \(Bipolar Transistor\)](#) for another example using this technique.

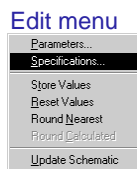
Selecting a Specification to Edit

You can change the properties of an existing specification at any time using the Edit Specification dialog box.

To display the Edit Specification dialog box for a goal or constraint

1 Do one of the following:

- In the specifications area of the PSpice Optimizer window, double-click the lower right-hand corner of the box for the specification you want to edit.
- Select Specifications from the Edit menu, then select the specification you want to edit, and click Change.



Measuring and Optimizing Performance

This section describes how to:

- Optimize your design once all of the parameters and specifications are defined.
- Monitor progress using Probe.

Optimizing Your Design

Optimization is a two-stage process:

- 1 Run one evaluation to ensure that the circuit is valid and that it simulates.
- 2 Start the optimization process.

To optimize your design

- 1 From the Tune menu, select Update Performance.

The PSpice Optimizer measures the design's performance using both the initial and current values for each of the parameters. The optimizer updates the initial and current fields, respectively, for each specification, and sets the progress indicators showing how closely these initial measures match each of the target values.

Tune menu



- 2 From the Tune menu, select Auto and click Start.

The PSpice Optimizer computes the derivatives for each specification with respect to each parameter, and uses this information to determine the direction in which to vary the parameters. With each iteration, the optimizer tries parameter changes along the chosen direction and measures performance until it achieves a reduction in the overall error. The optimizer then updates the parameters, calculates new derivatives, and repeats the process until one of the following occurs:

Tune menu



- Specifications are met (success).
- No more progress can be made (failure).
- You manually interrupt the process.

See [Viewing the Optimization Log on page 3-28](#) for information on the audit trail the PSpice Optimizer generates when running an optimization.

To abort the optimization run

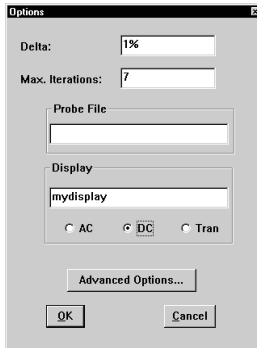
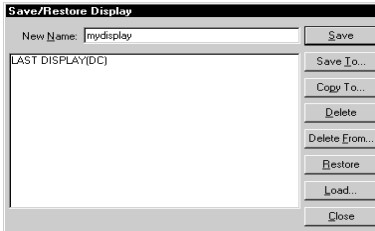
- 1 From the Tune menu, select Auto and click Terminate.

Graphically Monitoring Progress

To see how well specifications are approaching the optimization requirements, use Probe to monitor the simulation results from each of the iterations.

To monitor optimization progress

- 1 In Schematics, from the Analysis menu, select Simulate to simulate the circuit.



- 2 Create a Probe display configuration.
 - a In Probe, add the traces (from the Trace menu, select Add) and modify the axes (from the Plot menu, select X Axis Settings or Y Axis Settings) to appear as you want them to when monitoring intermediate results.
 - b From the Tools menu, select Display Control.
 - c In the New Name text box, enter a name for the Probe display and click Save.
 - d Click Close.
- 3 Define the Probe display to use when optimizing.
 - a In the PSpice Optimizer, from the Options menu, select Defaults.
 - b In the Display text box, enter the name of the Probe display.
 - c Select the analysis type corresponding to the Probe display.
 - d Click OK.

Exploring the Effect of Parameter and Specification Changes

The PSpice Optimizer provides three easy ways to examine tradeoffs between goals, constraints, and parameters. You can:

- Tweak parameter values to explore performance effects, or change specification values to see how parameters change, by entering new current values directly into the PSpice Optimizer window.
- Exclude specifications and/or parameters that you previously defined.

- Add new or edit existing parameter definitions and specification descriptions.

Testing Performance when Changing Current Values

In the PSpice Optimizer window, you can quickly examine the effects of a small change to either a parameter's current value or a specification's current value.

Recalculation modes When changing current values, you can choose between automatic and manual recalculation of performance.

- With automatic recalculation selected, you can change the current value (upper text box) for *one* parameter or a specification value, and see the impact on other values almost immediately.
- With manual recalculation selected, you can change the current value for *several* parameters or specifications, and initiate recalculation when you are ready.

Derivative calculations When tweaking values, the PSpice Optimizer does not perform any simulations. Instead, the optimizer requires derivative data and uses this to estimate what will happen when you change a parameter or specification.

The PSpice Optimizer calculates derivatives either:

- once, when you select Update Derivatives from the Tune menu, or
- for each iteration of an optimization run, when you select Auto from the Tune menu.

Use the first method (Update Derivatives) when you are exploring design tradeoffs.

See [Derivatives on page 4-10](#) for more information on how the PSpice Optimizer computes derivative data. See [Viewing Derivatives on page 3-28](#) for instructions on how to display the latest derivative data.

Note Because the performance of the design usually depends on the parameters in a highly nonlinear way, the results are typically reliable only for small changes in values. See [“Ensuring reliable results when tweaking values”](#) on page 3-24 for the steps you can take for best results.

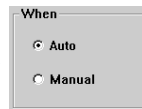
Automatically recalculating performance

The PSpice Optimizer automatically recalculates performance after each change provided that:

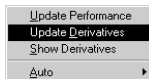
- Automatic recalculation is selected (the default).
- Derivatives are available.

To test performance using automatic recalculation

When frame in the Recalculate dialog box



Tune menu



The PSpice Optimizer accepts numerical entries in any format supported by PSpice. Refer to the online *MicroSim PSpice A/D Reference Manual* for a complete list of supported numeric forms.

- 1 Do the following to enable automatic recalculation:
 - a From the Options menu, select Recalculate.
 - b In the When frame, select Auto.
 - c Click Close.
- 2 From the Tune menu, select Update Derivatives.

When derivative calculations are complete, the message `Derivs. Avail` appears in the title bar.
- 3 Change the current value for the parameter or specification you want to investigate.
 - a Double-click in the current value (upper) field for a parameter or specification.
 - b Enter the new value.
 - c Press `Enter`.

The PSpice Optimizer immediately recalculates the values. When you change parameter values, a small “e” appears next to the progress indicator for each recalculated specification to show that the value is an *estimate* based on derivative data.

Manually recalculating performance

When you want to change multiple values before recalculating performance, disable recalculation. As with automatic recalculation, derivative data is required.

To test performance using manual recalculation

- 1 Disable automatic recalculation.
 - a From the Options menu, select Recalculate.
 - b In the When frame, select Manual.
 - c Click Close.
- 2 From the Tune menu, select Update Derivatives.

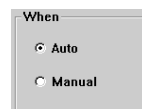
When derivative data is calculated, the message `Derivs . Avail` appears in the title bar.
- 3 Change the current value for the parameter or specification you want to investigate.
 - a Double-click in the current value (upper) field for a parameter or specification.
 - b Enter the new value.
 - c Press `Enter`.
- 4 From the Options menu, select Recalculate to recalculate performance.
- 5 Select the kind of recalculation as follows:

- If you entered a new *parameter* value, click Results.

The PSpice Optimizer recalculates performance. A small ‘e’ appears next to the progress indicator for each recalculated specification to show that the value is an *estimate* based on derivative data.

- If you entered a new *specification* value, click Parameters.

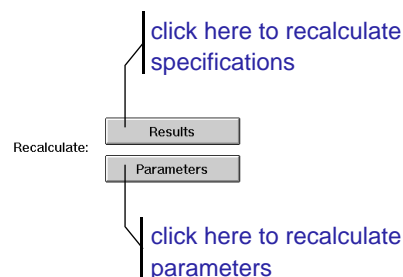
The PSpice Optimizer recalculates values for all of the parameters based on the current specification values.



When frame in the Recalculate dialog box



The PSpice Optimizer accepts numerical entries in any format supported by PSpice. Refer to the online *MicroSim PSpice A/D Reference Manual* for a complete list of supported numeric forms.



Ensuring reliable results when tweaking values

Because the PSpice Optimizer uses derivative data to estimate what will happen when a parameter or a specification is changed, and because the design usually depends on the parameters in a highly nonlinear way, results are typically reliable only for small changes in values. Once you have significantly changed values, resimulate and recompute the derivatives before adjusting values any further.

To ensure that results are reliable after significant tweaking

- 1 From the Tune menu, select Update Performance.

The PSpice Optimizer will run the appropriate simulations and update the specifications.

- 2 From the Tune menu, select Update Derivatives.

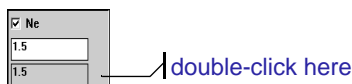
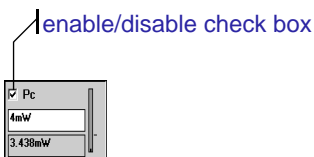
You are now ready to continue exploring the design.

Excluding Parameters and Specifications from Optimization

Every specification and parameter has a check box that you can select to exclude that specification or parameter from the next optimization run.

To exclude a parameter or specification from an optimization

- 1 Do one of the following:
 - In the PSpice Optimizer window, clear the check box (check box should be empty) to the left of the parameter or specification name.
 - Double-click the lower right-hand corner of the box for the parameter or specification you want to exclude, clear the Enabled check box (check box should be empty) in the dialog box, and click OK.



Note *When you exclude a specification from optimization, the PSpice Optimizer still re-evaluates its performance when you update the parameters or derivatives. It is also included in the matrix of partial derivatives.*

Testing Performance when Adding or Changing Parameters or Specifications

Even after running an optimization, you can add new parameters and specifications, or change the properties of existing definitions to see their effect on performance.

To test performance after changing the parameters or specifications

- 1 From the Edit menu, select Reset Values.
- 2 From the Tune menu, select Update Performance.

- 3 If you want to run a complete optimization:
 - a From the Tune menu, select Auto.
 - b Click Start.

Saving Intermediate Values

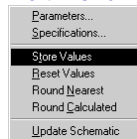
You can save a set of parameter and specification values, and then continue to investigate performance.

To save intermediate values

- 1 From the Edit menu, select Store Values.

The PSpice Optimizer copies the current values to the initial values for all specifications and parameters.

Edit menu

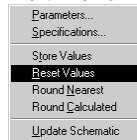


To restore the previous settings

- 1 From the Edit menu, select Reset Values.

The PSpice Optimizer copies the initial values to the current value fields.

Edit menu



Viewing Result Summaries

This section describes how to:

- Generate a report that summarizes the latest run.
- View the current derivative calculations.

Producing Optimization Reports

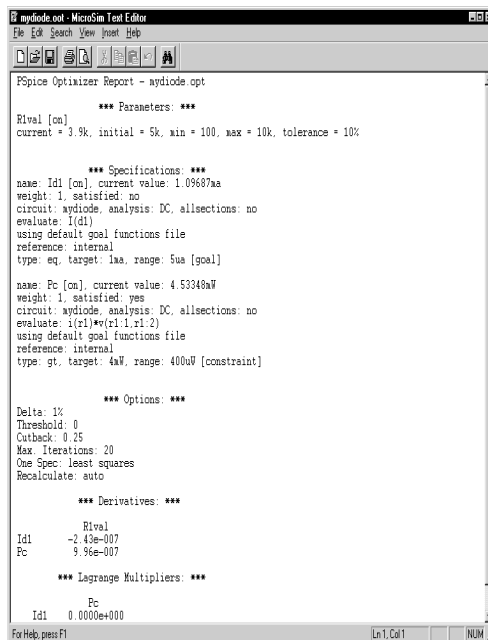
You can produce a report containing:

- settings for each of the parameters, specifications, and options
- performance results
- partial derivatives
- Lagrange multipliers

To generate an optimization report

- 1 From the File menu, select Report.

The PSpice Optimizer generates the report and saves it to an ASCII file named *design_name.oot*. The optimizer also displays the report in the text editor configured for your installation (Notepad, by default).



```

PSpice Optimizer Report - hydiode.opt
*** Parameters: ***
Rival [cn]
current = 3.9k, initial = 5k, min = 100, max = 10k, tolerance = 10%

*** Specifications: ***
name: Id1 [cn], current value: 1.09687aa
weight: 1, satisfied: no
circuit: hydiode, analysis: DC, allsections: no
evaluate: f(d1)
using default goal functions file
reference: internal
type: eq, target: 1aa, range: 5ua [goal]

name: Pc [cn], current value: 4.53346aW
weight: 1, satisfied: yes
circuit: hydiode, analysis: DC, allsections: no
evaluate: f(r1)*f(r1.1.r1.2)
using default goal functions file
reference: internal
type: gt, target: 4aW, range: 400uW [constraint]

*** Options: ***
Delta: 1%
Threshold: 0
Cutback: 0.25
Max. Iterations: 20
One Spec: least squares
Recalculate: auto

*** Derivatives: ***
          Rival
Id1      -2.43e-007
Pc       9.96e-007

*** Lagrange Multipliers: ***
          Pc
Id1      0.0000e+000
  
```

Figure 3-5 Sample Excerpt from a Report

To print an optimization report

- 1 Do one of the following:
 - If you are using MicroSim TextEdit (the default), from the File menu, select Print.

File menu

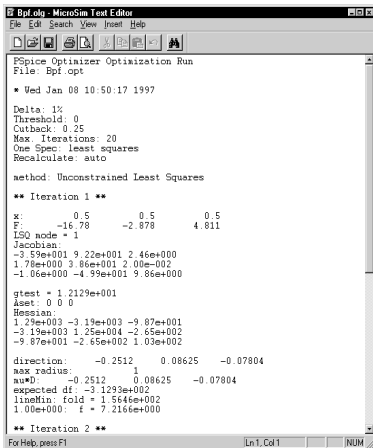


For UNIX users:

On UNIX platforms, the PSpice Optimizer displays the report using TextEdit, by default.

You can change the default text editor used by MicroSim programs by changing the path specified in the TEXTEDITCMD line in the [MICROSIM] section of the msim.ini file.

- If you are not using MicroSim TextEdit, use the Print command for the text editor showing the report.



```

PSpice Optimizer Optimization Run
File: Bpf.opt
* Wed Jan 08 10:50:17 1997

Delta: 1%
Threshold: 0
Outback: 0.25
Max Iterations: 20
One Spec: least squares
Recalculate: auto

method: Unconstrained Least Squares

** Iteration 1 **
x:          0.5          0.5          0.5
F:         -16.78         -2.878         4.811
LSQ mode = 1
Jacobian:
-3.59e+001  9.22e+001  2.46e+000
 1.78e+000  3.86e+001  2.00e+002
-1.06e+000 -4.99e+001  9.86e+000

gtest = 1.2129e+001
aset: 0 0 0
Residual:
 1.29e+003 -3.19e+003 -9.87e+001
-3.19e+003  1.25e+004 -2.65e+002
-9.87e+001 -2.65e+002  1.03e+002

direction: -0.2512  0.08625 -0.07804
max radius: 1
maxD:      -0.2512  0.08625 -0.07804
expected df: -3.1293e+002
limMin: fold = 1.5646e+002
 1.00e+000  f = 7.2166e+000

** Iteration 2 **

```

Figure 3-6 Sample Excerpt from a Log File

Viewing the Optimization Log

The PSpice Optimizer automatically creates an audit trail of optimization progress and saves the information to a file named *design_name.olg*. You can use this file as a debugging tool when an optimization fails to converge.

To view the optimization log file

- 1 Activate Notepad or another text editor.
- 2 Open the log file for browsing.

Viewing Derivatives

You can display a matrix showing the most recent derivative data the PSpice Optimizer uses to calculate performance.

To display the derivative data

- 1 From the Tune menu, select Show Derivatives.

Each entry in the matrix represents the partial derivative of one specification (the row label) with respect to one active parameter (the column heading). If a specification is completely independent of a parameter, the derivative is zero.

- 2 When finished, click Close.



	cap	res	width
PC	-1.72e+000	7.89e+000	1.23e+001
SR	1.47e+000	-1.04e+000	-1.00e+000
Area	-1.68e+001	1.11e+000	0.00e+000

Figure 3-7 Sample Derivative Data

Finalizing the Design

This section describes how to use the PSpice Optimizer to:

- Standardize component values once the optimization is complete.
- Save results.
- Back-annotate the schematic with the final component and parameter values.

Using Standard Component Values

When optimization parameters directly correspond to component values, you can use the PSpice Optimizer to select standard component values by either:

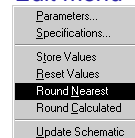
- rounding to the nearest values, or
- computing values based on the most recent optimization run.

The PSpice Optimizer considers the tolerance specified for the parameters using tables of preferred values for 1%, 5% and 10% tolerance components. Other tolerance values cause the optimizer to use the nearest calculated value for that tolerance. Parameters with zero tolerance are not changed.

To round component values to the nearest standard values

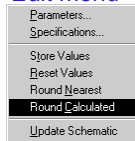
- 1 From the Edit menu, select Round Nearest.

Edit menu



To compute standard component values based on the most recent optimization

Edit menu



- 1 From the Edit menu, select Round Calculated.

The PSpice Optimizer replaces the parameter values with the standardized values only if the new values remain within the specified limits. If so, the optimizer automatically calculates new performance values (based on derivative data) using the new parameter values, and displays an ‘e’ in the upper right-hand corner of each specifications area to indicate that the performance measure is an *estimate*.

Saving Results

Performance results are not written to the optimization file until you deliberately initiate a save operation.

To save optimization results to the current optimization file

- 1 From the File menu, select Save.

By default, the PSpice Optimizer writes the results and the latest optimization settings to a file named *design_name.opt*. If you started your design from a schematic, this file already exists. If not, the optimizer displays the Save As dialog box and you must enter the name of a new file.

To save optimization results to a new or different file

- 1 From the File menu, select Save As.
- 2 Enter a new file name.
- 3 Click OK.

If an optimization file with this name already exists, the PSpice Optimizer requests confirmation to overwrite the file.

Updating the Schematic

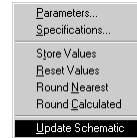
Once the optimization is complete and you have optionally standardized component values, you can update the underlying schematic with the final parameter values.

To back-annotate the schematic with the latest parameter values

- 1 In the PSpice Optimizer, from the Edit menu, select Update Schematic.

The optimizer writes the optimized parameter values to the schematic file. On the schematic, the new values appear in the Current column of the OPTPARAM symbol.

Edit menu



To use the new parameter values in subsequent simulations run from Schematics

- 1 In Schematics, from the Tools menu, select Use Optimized Params.

Understanding Optimization Principles and Options

4

Chapter Overview

This chapter explains optimization concepts and how you can influence the outcome of an optimization.

The concepts covered in this chapter include: constrained optimization, function characteristics, convergence and the effect of starting points, and how the PSpice Optimizer computes derivatives and scales target values.

[Default Options on page 4-13](#) describes the options you can set to control derivative calculations, maximum simulation iterations, and the Probe display.

[Advanced Options on page 4-17](#) describes the options you can set to control cutback, threshold, and the method (least squares/minimization) that the PSpice Optimizer uses.

Goals versus Constraints

If there is more than one goal, the PSpice Optimizer combines the errors by summing the squares of the normalized values.

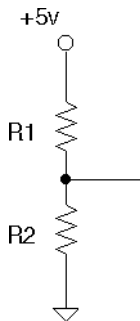


Figure 4-1 Resistive Terminator Circuit

Goals and constraints represent the ideal behavior of a design. In practice, this behavior is often unattainable.

Example: A gate cannot achieve zero propagation delay, but the goal of the optimization process might be to come as close as possible to that target value (that is, to reduce the error as much as possible).

When solving problems involving both goals and constraints, the PSpice Optimizer trades off meeting the target values for the goals against violation of the constraints. This means that the error indicator does not always reduce in value for a given iteration.

When setting up an optimization, you must decide which specifications are goals and which are constraints. In many cases, there are several legitimate ways to describe the design.

Example: Assume you want to design a resistive terminator that produces an output voltage of 3.75 V (± 0.1 V) at the junction of the two resistors, and the Thevenin equivalent resistance of the resistor combination must equal 100 Ω (± 1 Ω).

Your objective is to find the best resistor values to meet these two specifications:

- output voltage of 3.75 V (V_e)
- equivalent resistance of 100 Ω (R_e)

You can manually solve this problem using the following simultaneous equations:

$$\frac{5R_2}{R_1 + R_2} = 3.75 \quad \text{and} \quad \frac{R_1 \cdot R_2}{R_1 + R_2} = 100$$

These equations solve to $R_1 = 133.3$ Ω and $R_2 = 400$ Ω , an exact solution.

When using the PSpice Optimizer, you can set up this problem in one of three ways:

- Consider V_e and R_e as equally important; set up both as goals.
- Consider V_e as the most important requirement to meet, even at the expense of R_e ; set up V_e as a constraint and R_e as a goal.
- Consider R_e as the most important requirement to meet, even at the expense of V_e ; set up R_e as a constraint and V_e as a goal.

Note *Because at least one optimization goal is necessary, the case where both V_e and R_e are constraints is excluded.*

If the problem, like this one, has a solution, the PSpice Optimizer might arrive at the same answer for all three methods. However, most problems do not have a single, exact solution as this one does. For most designs, the result is a compromise that minimizes the goals while not violating the constraints.

Constrained Optimization

Many problems in analog circuit optimization are naturally expressed as the minimization of a function representing a goal (e.g., power consumption) which is subject to one or more constraints (e.g., bandwidth). Constraints are typically complicated nonlinear functions of the parameters of the problem, so manual optimization is a difficult task.

Most other analog circuit optimizers implement only unconstrained optimization of a single goal or a sum-of-squares of several goals. To tackle a problem like the problem outlined above, other optimizers must combine the functions for the goals and constraints and then optimize the combination. Unfortunately, this scheme does not differentiate between reduction of the goals and violation of the constraints. In

general, constraints are given much greater weight than the goals.

This approach has a number of pitfalls. In particular:

- If a very large value is used for the weight of the constraints, numerical problems occur.
- If a more reasonable value is used, the result is not a true solution of the original problem.
- Using a sequence of weights, and performing a series of minimizations can lead to the true solution, but at the expense of a large increase in optimization time (because of all of the extra evaluations required to solve the intermediate problems).

The PSpice Optimizer implements both constrained and unconstrained minimization algorithms. This means that the optimizer:

- Tackles constrained problems directly and efficiently.
- Calculates Lagrange multipliers for the solution, which provide valuable insight to design tradeoffs.

Types of Constraints

Constraints are restrictions placed on potential solutions to optimization problems. The simplest constraints are *bound constraints*—simple limits on the ranges of the parameters (e.g., a resistor whose value has to be at least 100 Ω).

More challenging constraints that frequently arise in analog circuit optimization have dependencies on other characteristics of the design.

Example: Consider optimizing a MOS amplifier cell which must satisfy these specifications:

- Reduce power consumption.
- Make sure gain-bandwidth product of the cell is greater than or equal to some minimum value.

The power consumption of the cell is the *goal* (the characteristic to be minimized) and the gain-bandwidth product is the *inequality constraint*.

To continue the example, consider the dependence of power consumption and gain-bandwidth product on bias current in one of the amplifier stages. Power consumption is proportional to the bias current, while the gain-bandwidth product is proportional to the square root of the bias current. Bias current must be reduced in order to reduce power consumption. Below some critical bias current the minimum gain-bandwidth requirement will be violated. This critical value is the *constrained minimum* for this problem.

The PSpice Optimizer can also handle *equality constraints* where a performance measure is required to be equal to some defined value.

Feasible and Infeasible Points

The starting point for an optimization can satisfy all the constraints (a *feasible* point) or it can violate one or more of the constraints (an *infeasible* point). Depending on the feasibility of the starting point, the PSpice Optimizer does the following:

- From an infeasible point, it attempts to reduce the goals *and* to reduce the amount by which the constraints are violated.
- From a feasible point, it attempts to reduce the goals *while* keeping the constraints satisfied.

Note *Because the PSpice Optimizer sometimes trades off reduction of the goals against violation of the constraints to make progress, an iteration can produce an infeasible point even though the initial starting point was feasible.*

Active and Inactive Constraints

An *active* constraint is one which affects the solution of the optimization problem—that is, the solution would probably be different if the constraint were removed. Equality constraints are always active (e.g., a constraint of the form $V_{\text{out}} = 3.75 \text{ V}$ is always active). Inequality constraints are considered active if the solution violates the constraint or if it is equal to the constraint (e.g., a constraint of the form $R_{\text{eq}} \geq 100$ is active if R_{eq} is less than or equal to 100 at the solution).

Lagrange Multipliers

To view Lagrange multipliers for your design, generate a report by selecting Reports from the File menu and browse the *design_name.oot* report file. See [Producing Optimization Reports on page 3-26](#) for instructions.

The result of a constrained optimization is typically a compromise between further reduction of the goals and violation of one or more constraints. A set of numbers—the *Lagrange multipliers*—provide valuable information about the tradeoffs between the goals and the constraints.

The PSpice Optimizer calculates Lagrange multipliers only for those constraints which are active at the solution. A constraint which is inactive can be removed from the problem without affecting the solution, which means there is no tradeoff between the goals and the constraint.

Think of Lagrange multipliers as the incremental *cost* of each active constraint on the solution.

Example: Consider optimizing propagation delay through a gate subject to a constraint on gate width and a constraint on bias current. Suppose that at the optimum, both constraints are active. There are two Lagrange multipliers for this problem:

- Incremental cost of propagation delay versus gate width.
- Incremental cost of propagation delay versus bias current.

For this problem, the units of the Lagrange multipliers are seconds/meter and seconds/ampere, respectively.

In a problem with several goals which have been combined as a sum-of-squares, the target value is dimensionless. In this case,

the units of the Lagrange multipliers are the reciprocal of the units of their associated constraint.

Example: A multiplier associated with a width constraint has units of meter⁻¹.

Characteristics of Functions

The success of an optimization depends highly on the behavior of the functions related to each specification.

Generally, the PSpice Optimizer obtains values (measures performance) for each of the specifications by *evaluating* a trace which results from a simulation with varying parameter values. The optimizer can experience difficulties if the accuracy of the measurement is decreased by:

- Discontinuities in the simulation results.
- An error in the Probe goal function definition.
- Inaccuracies in the simulation results on which the evaluation is based.

Generally, simulations and measurements using AC and DC analyses behave better than simulations using transient analyses. This is particularly true if the evaluations are set up to measure the value of a single point (e.g., the time when a trace crosses a specified level). This kind of measurement tends to behave discontinuously as the parameters change, creating difficulties for the optimizer.

To improve measurement accuracy, consider any of the following techniques:

- Use several points rather than a single point for the Probe goal function. That is, specify several points on a waveform instead of a single point.
- Reduce the step ceiling in Transient analysis to produce a more finely sampled set of data.

To review, the PSpice Optimizer measures performance in one of three ways: by taking the value of the single-point trace (trace function) in Probe, by applying a Probe goal function to the trace, or by evaluating a PSpice Optimizer expression. Trace functions and Probe goal functions require a simulation; PSpice Optimizer expressions do not.



- Increase simulator accuracy (e.g., reduce RELTOL).

The first proposed technique is preferred because it does not affect the time required to run each simulation (usually the determining factor in how long an optimization run takes).

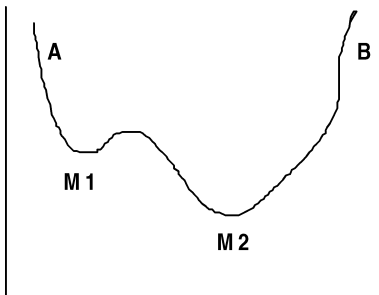


Figure 4-2 Global and Local Minima of a Function

Global and Local Minima

The curve in Figure 4-2 shows a 1-dimensional function with 2 minima. Point M1 is a local minimum; it satisfies the conditions for a minimum, but there is another minimum which is smaller. Point M2 is the global minimum for the function. There are no points within the range of the function which are smaller.

All practical optimization techniques find local minima, including the algorithms used by the PSpice Optimizer. This may or may not present a problem. The application may not have any local minima within the domain of interest. If local minima do exist, the global minimum may be the nearest solution to the starting point. This is discussed further in [Starting Points on page 4-8](#).

Starting Points

It is important to begin with a good estimate of the starting point. There are two reasons for this:

- The process may converge to the *wrong* solution (a local minimum) rather than to the *right* solution (the global minimum).

Example: Consider Figure 4-2. If point A is chosen, the PSpice Optimizer will most likely find local minimum M1. If point B is chosen, the optimizer will most likely find the global minimum M2.

- The PSpice Optimizer may require a large number of simulations to find a region close to a solution. It is usually

more efficient to find the approximate location of the desired solution (perhaps by performing a number of analyses sweeping out ranges of the parameters) before starting the optimization process.

Convergence

When running an optimization, the PSpice Optimizer varies the parameter values and measures the resulting performance. For each subsequent iteration, the optimizer chooses each parameter step to reduce the error between the design's measured and specified target performance.

If the optimizer finds a solution where all of the specifications are met, then the process has *converged*. There are several common reasons why the process may fail to converge:

- There is no solution to the problem as specified.
- The simulation and/or evaluations are not accurate enough to allow the solution to be found.
- A limit on the number of simulations or elapsed time is encountered.
- The optimizer finds a spurious numerical minimum which is not the desired solution.

To improve convergence, consider the following techniques:

- In the second case, use more accurate measurement techniques (possibly with the aid of *help* circuitry).
- In the last case, restart from a different starting point which might lead to a different solution.



Parameter Bounds

The PSpice Optimizer performs *bound-constrained* minimization. This means it will solve problems where one or

more of the parameters are limited by the specified upper or lower bound for that parameter. In other words, the optimizer finds a solution (if one exists) even if one or more parameters are at their limit.

However, solving this kind of problem is intrinsically more difficult than performing *unconstrained* minimization.



If one or more parameters appear to be limited during the optimization run, and you don't expect the final solution to have limited parameters, you could save time by using one or both of the following techniques:

- Use a starting point that is further from the parameter limits.
- Loosen the limits on the parameter(s) in question.

Derivatives

The PSpice Optimizer calculates derivatives either:

- once when you select Update Derivatives from the Tune menu, or
- automatically for each iteration when you start optimization by selecting Auto from the Tune menu.

See [Exploring the Effect of Parameter and Specification Changes on page 3-20](#) for more information.

To perform optimization, the PSpice Optimizer computes the matrix of partial derivatives—the *Jacobian*.

How the PSpice Optimizer Estimates Derivatives

The PSpice Optimizer approximates derivatives using a finite difference approach. In one-dimensional terms, this method computes an approximation to the first derivative of a function $f(x)$ by:

$$f'(x) \cong \frac{f(x+h) - f(x)}{h}$$

where h is a small perturbation.

The optimizer organizes the simulations and evaluations to compute the Jacobian in the most efficient way possible.

Example: If there are two specifications, each of which requires a DC analysis of the same circuit file, the optimizer will run a single simulation for each parameter, then load the data file (.dat) into Probe and evaluate the perturbed values of f . If there are M specifications (all using the same analysis type and the same circuit file) and N parameters, then forming the Jacobian takes:

- N simulations, and
- M Probe goal function evaluations per simulation.

Note *The time needed to simulate is usually much greater than the time needed to evaluate the Probe goal functions. This means that the time taken to optimize a design depends heavily on the number of variable parameters.*

Limitations of Derivative Data

A derivative analysis calculates a linear relationship between a parameter and a specification. It assumes that the function is linear near the initial value within a region defined by the value set for the Delta option. If the data is well-behaved in this region, then this is a valid assumption; the PSpice Optimizer can use the derivative to approximate specification values based on the linear relationship.

However, when the function is not well behaved in the region around the initial value, the approximation may not be valid.

Example: Assume that the function shown in Figure 4-3 is the plot of a specification's behavior vs. a parameter value. Note that the function is approximately linear between the dashed lines, but not necessarily linear outside of that region.

If you pick an initial value for the parameter which is between the two dashed lines, then subsequently compute the derivatives, the derivative data provides a reasonable approximation for any other parameter value between those lines. However, if you try to use the same derivative data to estimate new specification values for parameter values outside of those lines, the estimates are not reliable.

See [Controlling Finite Differencing when Calculating Derivatives \(Delta Option\)](#) on [page 4-13](#) for more information on Delta.

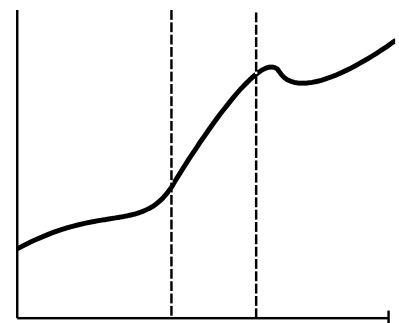


Figure 4-3 Hypothetical Function



When, for a given parameter, the difference between its initial value and the value of interest is large (that is, the relative difference is much bigger than Delta), modify its initial value and restart the simulation.

Note *When you modify the current value for a parameter to recalculate specification values (by editing the value appearing in the PSpice Optimizer window), the PSpice Optimizer uses the derivative data rather than a resimulation to determine the new values. Therefore, you should periodically verify the results with another simulation (see [“Ensuring reliable results when tweaking values”](#) on page 3-24).*

Target Value Scaling

When there is more than one goal, or a combination of goals and constraints, the PSpice Optimizer needs to scale the raw measurements before combining them.

Example: Consider a least-squares optimization with two specifications:

- collector-base capacitance
- collector resistance

The first of these can have values of several picofarads; the second of tens of kohms. Clearly, adding the squares of the errors for these specifications will lose the significance of the capacitance term. Instead, the optimizer scales the values as follows:

$$T_{Scaled} = \frac{T_{measured} - T_{target}}{Range}$$

Suppose the specified target and measured values are as shown below.

Specification	Target Value	Allowed Range	Measured Value
collector-base capacitance	10 pF	±1 pF	8 pF
collector resistance	10 k	±1 k	12 k

Then the scaled values are:

$$\frac{(8\text{pF} - 10\text{pF})}{1\text{pF}} = -2 \quad \text{and} \quad \frac{25\text{k} - 10\text{k}}{1\text{k}} = 15 ,$$

respectively.

Since these numbers are close enough in magnitude, the PSpice Optimizer can combine them without losing numerical significance.

Default Options

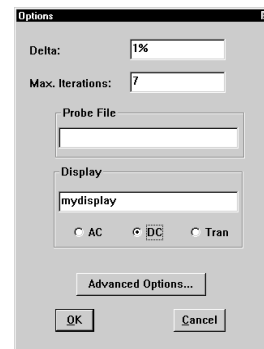
This section describes the basic configuration options for the PSpice Optimizer.

To display the Options dialog box

- 1 From the Options menu, select Defaults.

Controlling Finite Differencing when Calculating Derivatives (Delta Option)

The Delta option specifies the relative amount (as a percentage of current parameter value) by which the PSpice Optimizer perturbs each parameter from its initial value when calculating the derivatives.



The PSpice Optimizer saves option settings to the .opt file so that they remain with the optimization's parameter and specification settings.

To generate a summary of the option settings for a given design, select Report from the File menu.

The optimizer uses gradient-based optimization algorithms that use a finite difference method to approximate the gradients (gradients are not known analytically). To implement finite differencing, the optimizer:

- 1 Perturbs each parameter in turn from its current value by an amount h .
- 2 Evaluates the function at the perturbed value.
- 3 Subtracts the old function value from the new.
- 4 Divides the result by h .

Note *There is a tradeoff. If h is too small, the difference in function values is unreliable due to numerical inaccuracies. However if h is too large, the result is a poor approximation to the true gradient.*

To control parameter perturbation when calculating derivatives

- 1 Enter a value in the Delta text box that defines a fraction of the parameter's total range.

Example: If a parameter has a current value of 10^{-8} , and Delta is set to 1% (the default), then the PSpice Optimizer perturbs the parameter by 10^{-10} .

The 1% default value is suitable for the accuracy of typical simulations.



- 2 If the accuracy of your simulation is very different from typical (perhaps because of the use of a non-default value for either RELTOL or the time step ceiling for a Transient analysis), then change the value of Delta as follows:
 - If simulation accuracy is better, decrease Delta by an appropriate amount.
 - If simulation accuracy is worse, increase Delta by an appropriate amount.

Note *The optimum value of Delta varies as the square root of the relative accuracy of the simulation. For example, if your simulation is 100 times more accurate than typical, you should reduce Delta by a factor of 10.*

Limiting Simulation Iterations (Max. Iterations Option)

The Max. Iterations option defines how many attempts, at most, the PSpice Optimizer can make before *giving up* on the solution, even if the optimizer is making progress.

To limit the simulation iterations

- 1 In the Max. Iterations text box, enter an integer value that defines the maximum number of attempts you will allow the PSpice Optimizer to make.

Specifying a Probe Display (Probe File and Display Options)

The Display option defines the name of the Probe display the PSpice Optimizer uses to display simulation results. The Probe File option specifies a nondefault .prb file in which the Probe display information has been stored.

To use a specific Probe display when optimizing

Refer to online Help in Probe for information on using Probe Display Control options.

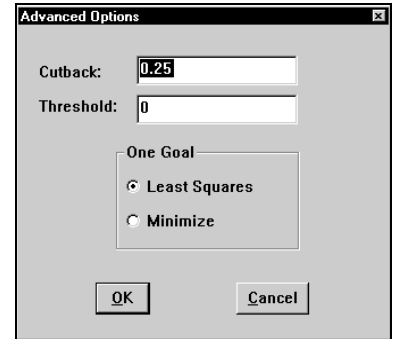
- 1** In Probe, configure the plots as you want to view them.
- 2** Save and name the display:
 - a** From the Tools menu, select Display Control.
 - b** Enter the name for the display.
 - c** Click Save.
- 3** In the PSpice Optimizer, specify the Probe display.
 - a** From the Options menu, select Defaults to display the Defaults dialog box.
 - b** In the Display frame, type the Probe display name in the text box.
 - c** Choose the appropriate analysis type (AC, DC, or Tran).
 - d** If you saved the Probe display to a nondefault .prb file, enter the name of the file in the Probe File text box.

Advanced Options

This section describes the advanced configuration options for the PSpice Optimizer.

To display the Advanced Options dialog box

- 1 From the Options menu, select Defaults.
- 2 Click the Advanced Options button.



The PSpice Optimizer saves option settings to the .opt file so that they remain with the optimization's parameter and specification settings.

To generate a summary of the option settings for a given design, select Report from the File menu.

Controlling Cutback (Cutback Option)

The Cutback option defines the minimum fraction by which an internal step is reduced while the PSpice Optimizer searches for a reduction in the goal's target value.

To set cutback

- 1 In the Cutback text box, enter a decimal fraction that defines the minimal percent reduction in internal step size.

If the data is noisy, consider increasing the Cutback value from its default of 0.25.



Controlling Parameter Value Changes Between Iterations (Threshold Option)

The Threshold option defines the minimum step size the PSpice Optimizer uses to adjust the optimization parameters during the optimization process.

The optimizer assumes that the values measured for the specifications change continuously as the parameters are varied. In practice, this assumption is not justified. For some analyses, especially transient analyses, the Probe goal function values

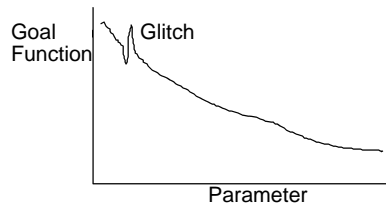


Figure 4-4 *Hypothetical Data Glitch*

show discontinuous behavior for small parameter changes. This can be caused by accumulation of errors in iterative simulation algorithms.

Figure 4-4 demonstrates a typical case. The effect of the glitch is serious—the optimizer can get stuck in the spurious local minimum represented by the glitch.

The optimizer’s threshold mechanism limits the effect of unreliable data.

To control parameter perturbation between iterations

- 1 In the Threshold text box, enter a value that defines a fraction of the current parameter value.

Example: A Threshold value of 0.01 means that when the PSpice Optimizer changes a parameter value, the value will change by at least 1% of its current value.



By default, Threshold is set to 0 so that small changes in parameter values are not arbitrarily rejected. To obtain good results, however, you may need to adjust the Threshold values. When making adjustments, consider the following:

- If data quality is good, and Threshold is greater than zero, reduce the Threshold value to find more accurate parameter values.
- If data quality is suspect (has potential for spurious peaks or glitches), increase the Threshold value to ensure that the optimizer will not get stuck during the run.

Choosing an Optimization Method for Single Goal Problems (Least Squares/Minimization Options)

The PSpice Optimizer implements two general classes of algorithm to measure design performance: least squares and minimization. These algorithms are applicable to both unconstrained and constrained problems.

Least squares A reliable measure of performance for a design with multiple targets is to take the deviation of each output from its target, square all deviations (so each term is positive) and sum all of the squares. The PSpice Optimizer then tries to reduce this sum to zero.

This technique is known as *least squares*. Note that the sum of the squares of the deviations becomes zero only if all of the goals are met.

Minimization Another measure of design performance considers a single output and reduces it to the smallest value possible.

Example: Power or propagation delay, each of which is a positive number with ideal performance corresponding to zero.

Choosing the algorithm When optimizing for more than one goal, the PSpice Optimizer always uses the least-squares algorithm. For a single goal, however, you must specify the algorithm for the optimizer.

To set the optimization method for a single goal

- 1 Do one of the following:
 - Click the Least Squares button to minimize the square of the deviation between measured and target value.
 - Click Minimize to reduce a value to the smallest possible value.



If your optimization problem is to *maximize* a single goal, then set up the specification to *minimize the negative of the value*.

Example: To maximize gain, set up the problem to minimize $-gain$.

Tutorial: Optimizing a Design (Passive Terminator)

5

Tutorial Overview

The following tutorial takes you through the steps needed to setup and run an optimization starting with the simple terminator example provided with your MicroSim programs.

In this tutorial, you will:

- Verify the schematic setup:
 - Check that component values are parameterized.
 - Check that optimization parameters are defined.
 - Check the analysis settings.
- Check the goal specification settings.
- Run the optimization.

For a complete hands-on tutorial in which you draw the schematic and set up the optimization from scratch, see [Chapter 2, Primer: How to Optimize a Design](#).

The Passive Terminator Design

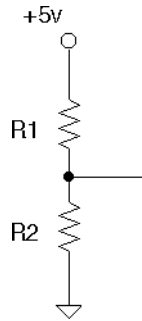


Figure 5-1 Resistive Terminator Circuit

Figure 5-1 shows a simple terminator that you could use, for example, for one line of a backplane. The top end of R1 connects to a +5 v DC supply; the lower end of R2 connects to ground. The center point of the two resistors provides the line termination.

This design must meet two specifications:

- Voltage at the junction of the two resistors (V_{center}) must lie within a specified range.
- Thevenin equivalent resistance of the combination (R_{equiv}) must equal a specified value.

Assume that the DC supply has negligible internal resistance, and that:

- V_{center} must be 3.75 ± 0.1 V.
- R_{equiv} must be 100 ± 1 Ω .

Your objective is to find the best values for R1 and R2 that meet the stated specifications. You can manually solve this problem using the following simultaneous equations:

$$\frac{5R_2}{R_1 + R_2} = 3.75 \quad \text{and} \quad \frac{R_1 \cdot R_2}{R_1 + R_2} = 100$$

These equations solve to $R_1 = 133.3$ Ω and $R_2 = 400$ Ω , an exact solution. As you complete this tutorial, compare this solution to that produced by the PSpice Optimizer.

Loading the Design into Schematics

To begin, set up a schematic with:

- parameters for the resistor values, and
- a way to measure the performance of the two specifications.

Your MicroSim installation includes a schematic for the passive terminator design.

To load the passive terminator schematic

- 1 From the Windows 95 Start menu, select the MicroSim program folder and then the Schematics shortcut to start Schematics.
- 2 From the File menu, select Open.
- 3 Move to the directory containing `term.sch` (`\MicroSim_root\examples\optimize\term`) and, in the File Name list, select the schematic file.

This schematic contains two sections. The first section connects an instance of the terminator to a DC source and to ground. The output is connected to a bubble port labeled V_c . The second section connects the top and bottom ends of an instance of the terminator to ground, and a 1 A current source to its output. This output is connected to a bubble port labeled V_r . The voltage at V_c gives the value of the V_{center} specification. The voltage at V_r gives the equivalent resistance of the network, the R_{equiv} specification.

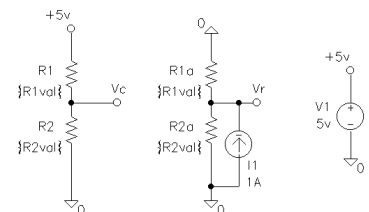


Figure 5-2 Schematic for the Terminator Example, `term.sch`

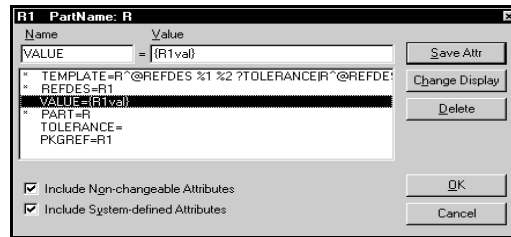
Setting Component Values to Expressions

The parameters varied between iterations must relate to the components you are optimizing. For this example, R1 and R2 resistor values are already parameterized.

To see how the resistor values are set up to use optimization parameters

- 1 Double-click the symbol graphics for R1 or R2.
- 2 Within the symbol attribute dialog box, note the value of the VALUE attribute: {R1val} or {R2val}, depending on the symbol you selected.

The curly braces are PSpice syntax for an expression. You can specify any expression that PSpice can evaluate.



- 3 Click OK to close the symbol attribute dialog box.

Defining Optimization Parameters

Next, set up the optimization parameters that the PSpice Optimizer will vary between iterations; these are the same parameters used in the value expressions for R1 and R2. For this example, R1val and R2val are already defined using an OPTPARAM symbol as shown below.

OPTIMIZER PARAMETERS:		
Name	Initial	Current
R1val	500	500
R2val	500	500

The optimization parameter settings are the same for R1val and R2val as follows:

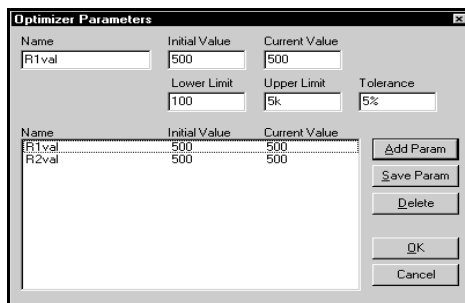
Setting	Value
Initial Value	500
Current Value	500
Lower Limit	100
Upper Limit	5K
Tolerance	5%

To see the settings for R1val and R2val

- 1 On the schematic, double-click OPTIMIZER PARAMETERS.

By default, the settings for R1val are displayed.

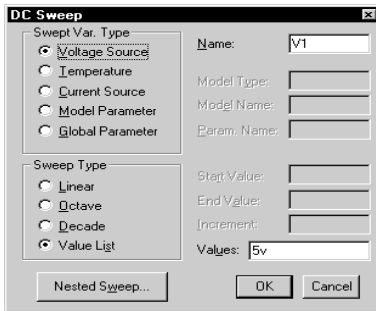
- 2 To view settings for R2val, click the R2val entry in the parameter list.



- 3 Click OK to close the Optimizer Parameters dialog box.

Defining the Analysis Type

For each specification, set up an analysis which PSpice will run for each iteration of the optimization. This example is already set up for a 1-point DC analysis, with the supply set to 5 V.



To see the analysis settings

- 1 From the Analysis menu, select Setup.
- 2 Click DC Sweep and verify the settings.
- 3 Click Cancel to return to the Analysis Setup dialog box.
- 4 Verify that DC Sweep is the only enabled analysis. If needed, select the DC Sweep check box and clear any other selected check boxes.
- 5 If wanted, select Bias Point Detail.

Running an Initial Circuit Analysis

Before optimizing, verify that the circuit works, and check that the voltages at V_c and V_r are as you expect.

To test the circuit setup

To have Probe automatically run after simulation (the default)

- 1 In Schematics, from the Analysis menu, select Probe Setup.
- 2 In the Auto-Run Option frame, select Automatically Run Probe After Simulation.
- 3 Click OK.
- 1 In Schematics, from the Analysis menu, select Simulate.
- 2 If Probe does not activate automatically, select Run Probe from the Analysis menu.
- 3 In Probe, from the Trace menu, select Add and click $V(V_c)$ and $V(V_r)$.
- 4 Click OK.
- 5 Verify that the voltages at V_c and V_r are 2.5 V and 250 V respectively.

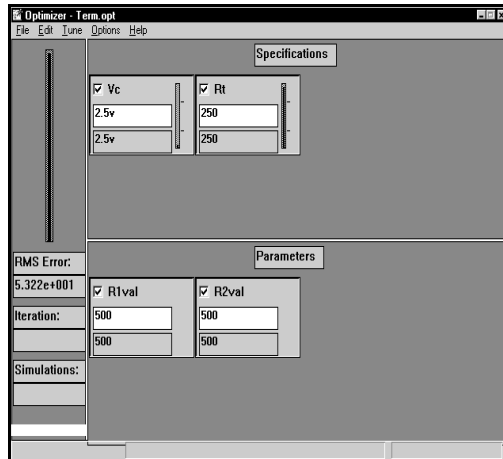
Activating the PSpice Optimizer

Assuming the circuit simulated successfully, you are now ready to activate the PSpice Optimizer and complete optimization setup.

To activate the PSpice Optimizer

- 1 In Schematics, from the Tools menu, select Run Optimizer.

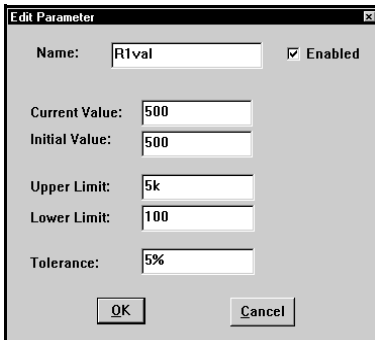
The PSpice Optimizer window displays the current optimization file, `term.opt`, in the title bar.



The two parameters, R1val and R2val, appear in the parameters area. Because the specifications, Vc and Rt, are predefined for this example circuit, they also appear in the specifications area. Initially, the error gauge area is empty.

Viewing the Parameter Description

The PSpice Optimizer automatically loads the parameter descriptions defined earlier in Schematics.



To verify the parameter descriptions

- 1 From the Edit menu, select Parameters.
- 2 To see R1val settings.
 - a In the Parameters list, click R1val.
 - b Click Change to display the Edit Parameter dialog box.
- 3 Click Cancel to leave the parameter unchanged.
- 4 Optionally repeat steps 2 and 3 for R2val.
- 5 Click Close to exit the Edit Parameter dialog box.

Defining the Goals and Constraints

To review, the specifications for this example are:

- Voltage, V_{center} , at the junction must be 3.75 ± 0.1 V.
- Thevenin equivalent resistance, R_{equiv} , must be 100 ± 1 Ω .

One way to handle the optimization problem is to treat both specifications as a goal.

For each goal, you must define:

- its name
- its target value and range
- the analysis to use for simulation
- the circuit file containing the netlist description
- the way to measure performance

For this example, these are already defined.

To see the Vc and Rt goal settings

- 1 From the Edit menu, select Specifications.
- 2 Click Vc, then click Change.

Edit Specification

Name: Enabled

Reference: Internal External Weight:

Internal

Target:
Range:

Constraint
Type:

External

File:
X Column Name:
Y Column Name:
Tolerance:

Analysis

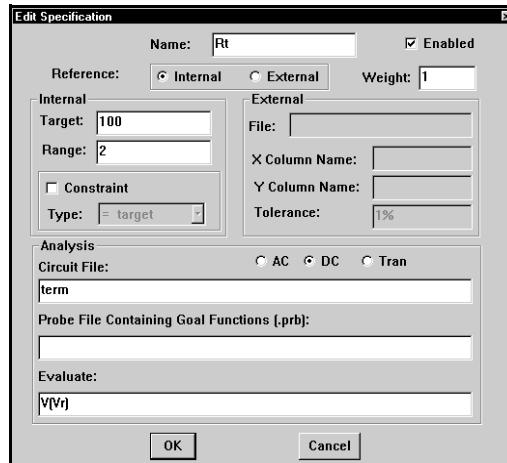
Circuit File: AC DC Tran

Probe File Containing Goal Functions (.prb):

Evaluate:

Notice that V_c is described as a goal (Constraint check box is cleared). The analysis type is DC and the evaluation to measure performance is the trace function $V(V_c)$ —as in the preliminary test simulation.

- 3 Click Cancel to leave the specification unchanged.
- 4 To see the settings for the R_t goal, click R_t , then click Change.



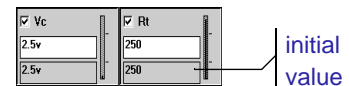
Again, the analysis type is DC, and the evaluation to measure performance is the trace function $V(V_r)$.

- 5 Click Cancel to leave the specification unchanged.
- 6 Click Close to exit the Edit Specification dialog box.

Checking that the Design Will Simulate

Before running a full optimization, you should run a single iteration to make sure the design still simulates (by selecting Update Performance from the Tune menu). The PSpice Optimizer performs the simulations and trace function evaluations required to find the current value for each active specification, and updates their initial values.

For this example, an initial iteration has already been run. The Vc specification shows 2.5 and the Rt specification shows 250 in the initial value fields. Note that this example requires one PSpice simulation to update the performance.



Starting the Optimization

You are now ready to run the full optimization to find resistor values which satisfy the two specifications (output voltage 3.75 V, equivalent resistance 100 Ω).

To start optimizing

- 1 From the Tune menu, select Auto and click Start.

First, the PSpice Optimizer computes the derivative of each specification with respect to each parameter at the initial value. While this is in progress, the message `Updating Derivatives` appears in the status bar of the optimizer window. In this example, two simulations are performed—one for each parameter.

Using the derivative information, the optimizer selects a direction in which to vary the parameters. The optimizer tries parameter changes along this direction until it achieves a reduction in the overall error. The optimizer then updates the parameters, and computes new derivatives.

The optimizer repeats this process until it achieves success, failure, or you terminate the process (from the Tune menu, point to Auto and select Terminate).

For this example, let the process run to completion. This takes 5 more simulations.

As the optimization proceeds, the vertical gauge and numeric display fields in the error gauge area (far left in the PSpice Optimizer window) show updated values. The numeric field shows the total error (the sum of the squares of the normalized errors for the active specifications) and the gauge shows the error relative to the starting point. Final results are shown in Figure 5-3.

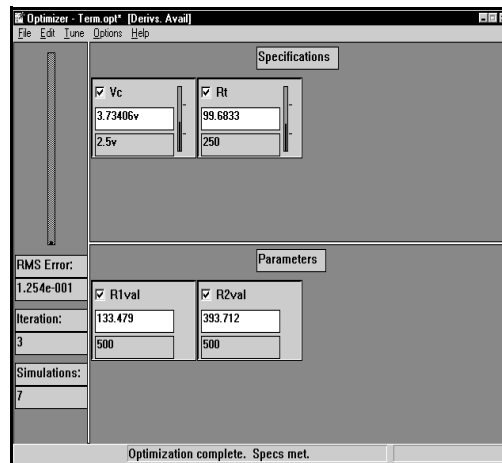


Figure 5-3 Optimization Results for the Passive Terminator Example

Compare the parameter values of 133.5 for R1val and 393.7 for R2val with the calculated values on page [5-2](#). They are very close.

Changing a Goal to a Constraint

Try examining what happens when you change one of the specifications from a *goal* to a *constraint*.

For an example, see See [Goals versus Constraints on page 4-2](#) for more information.

To change a goal to a constraint

- 1 Edit a specification:
 - a In the PSpice Optimizer window, double-click the hot-spot (lower right-hand corner) in either the Vc or Rt specification box.
 - b Select the Constraint check box.
 - c In the Type list, choose the type of constraint.
- 2 From the Edit menu, select Reset Values to set the current values back to the original initial values.
- 3 From the Tune menu, select Update Performance.
- 4 If you want to run a complete optimization:
 - a From the Tune menu, select Auto.
 - b Click Start.

For this example, the PSpice Optimizer should produce approximately the same result for each configuration of the two specifications (one goal and one constraint).

Saving Results

At this point you would ordinarily select Save from the File menu to save the results. However, to avoid modifying the tutorial files, either:

- From the File menu, select Save As to store the results under another file name, or
- omit the save process entirely.

Tutorial: Exploring Design Tradeoffs (Active Filter)

6

Tutorial Overview

The following tutorial shows you how to use the PSpice Optimizer to explore design tradeoffs. Using the simple active filter example provided with your MicroSim programs, you will:

- Review the optimization setup in Schematics: parameter definitions and parameterized expression assignments.
- Review the goal and constraint setup in the PSpice Optimizer.
- Change a parameter value to see the effect on goal and constraint values.
- Change a goal value to see the effect on parameter values.

The Active Filter Design

The filter has three adjustable resistors. These resistors adjust center frequency, bandwidth, and gain. In this case, the adjustments are interdependent.

Requirements for the filter are:

- Center frequency (F_c) must be 10 Hz with 1% accuracy.
- 3 dB bandwidth (BW) must be 1 Hz with 10% accuracy.
- Gain (Gain) must be 10 with 10% accuracy.

To load the active filter design

- 1 From the Windows 95 Start menu, select the MicroSim program folder and then the Schematics shortcut to start Schematics.
- 2 From the File menu, select Open.
- 3 Move to the directory containing `bpf.sch` (`\MicroSim_root\examples\optimize\bpf`) and select the schematic file in the File Name list.

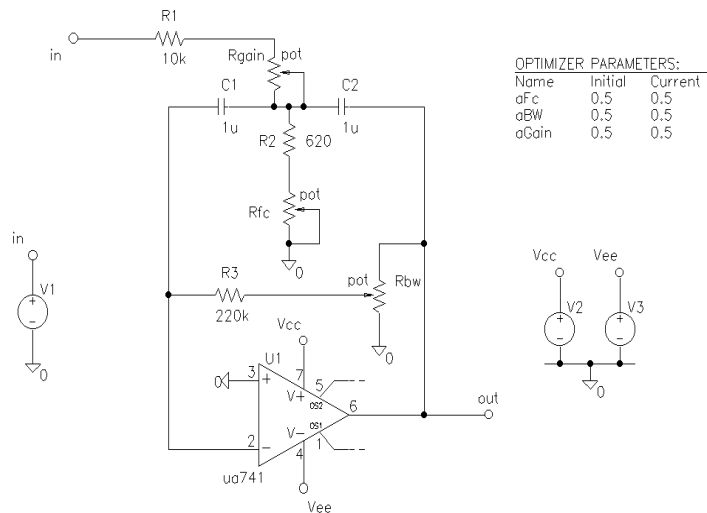


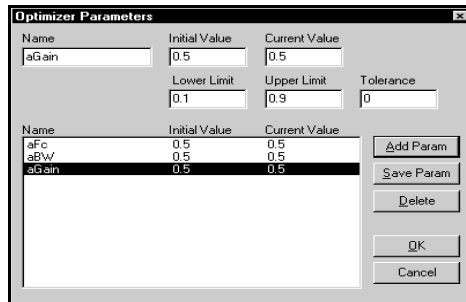
Figure 6-1 Schematic for the Active Filter Example, `bpf.sch`

The Parameters

The three variable resistors—Rfc, Rbw, and Rgain—are implemented as potentiometers. The potentiometer symbol has an attribute called SET which describes the slider position—a value between 0 and 1.

To optimize slider position, the optimization parameters afc, aBW, and AGain are assigned to the SET attribute for the symbols Rfc, Rbw, and Rgain, respectively.

The parameters are shown in the OPTPARAM symbol on the schematic. Each parameter is set up to range from 0.1 to 0.9 with an initial value for each set to 0.5 (each potentiometer's center point). The settings for aGain are shown below.



To see the assignment for SET, double-click on one of the variable resistor symbols to bring up its list of symbol attributes.

To display the Optimizer Parameters dialog box, double-click the OPTPARAM symbol in the schematic.

The Goals

The example circuit is set up with predefined goals which you can view using the PSpice Optimizer.

To activate the PSpice Optimizer

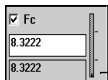
- 1 From the Tools menu, select Run Optimizer.

The PSpice Optimizer window shows the three parameters (aFc, aBW and aGain) and the three corresponding goals (Fc, BW, and Gain). The goals are defined as follows:

Setting	Center Frequency	Bandwidth	Gain
Name	Fc	BW	Gain
Target	10	1	10
Range	0.1	0.1	1
Analysis	AC	AC	AC
Circuit File	bpf	bpf	bpf
Evaluate	CenterFreq (vdb(out), 1)	Bandwidth (vdb(out), 3)	max(v(out))

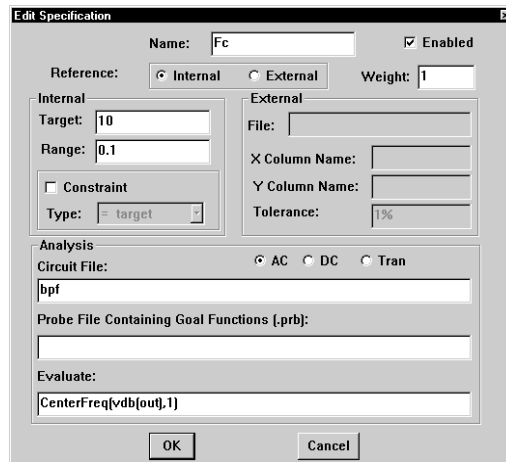
The Fc settings as they appear in the Edit Specification dialog box are shown below.

To display the Edit Specifications dialog box, double-click the lower right-hand corner of the specification box of interest in the PSpice Optimizer window.



double-click here

When finished browsing, click Cancel.



Note that, to measure performance, all three goals are evaluated using Probe goal functions.

Testing Performance

Initial performance has already been measured for the active filter ensuring that the design simulated as expected, that current values for the goals were calculated, and that initial values were updated. Initial performance calculated to:

Goal	Current/Initial Value	Target Value
Fc	8.322 Hz	10 Hz
BW	0.7122 Hz	1 Hz
Gain	14.81	10

To remeasure performance, select Update Performance from the Tune menu.

Calculating Derivatives

To estimate performance effects with small changes in parameters (or specifications), the PSpice Optimizer uses derivatives of each specification with respect to each of the parameters.

To calculate the derivatives

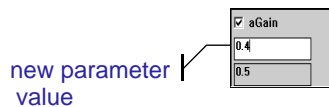
- 1 From the Tune menu, select Update Derivatives.

The message `Updating Derivatives` appears in the status bar while this takes place.

Tweaking Parameters

Once the derivatives are calculated, you can use the PSpice Optimize to explore how small changes in the parameters affect the performance of the design.

To examine the performance effect when changing aGain from 0.5 to 0.4



A screenshot of the Optimize goal table. The table has three columns: Fc, BW, and Gain. The first row shows the current values, and the second row shows the updated goal values. A callout box with the text 'updated goal values' points to the second row.

✓ Fc	✓ BW	✓ Gain
8.2976	0.712009	13.8246
8.3222	0.712189	14.8106

- 1 Highlight the current value for aGain, and type 0.4.
- 2 Press **Enter**.

The value of the Gain specification changes from the initial value of 14.811 to a new value of 13.825. The target is 10.0, which is a change in the right direction.

Center frequency and bandwidth also change. In the case of center frequency, the change from its initial value of 8.322 to a new value of 8.298 is in the wrong direction.

This frequently occurs. You can try various strategies to get closer to the target values. Two common approaches are:

- Adjust a given parameter to get the best results, then vary the other parameters.

Example: Continue to adjust aGain until you are satisfied with the performance. Then incrementally change Fc until you are satisfied. And finally, change BW.

- Change every parameter in the set by a small amount at a time, and continue in this manner to get the best results.

Example: Adjust aGain once by a small amount, then Fc, then BW. Continue with this pattern of adjustments until you are satisfied.

Tweaking Goals and Constraints

The PSpice Optimizer gives you the advantage of doing something not possible on the bench: changing the performance results and seeing what parameter values would produce these results.

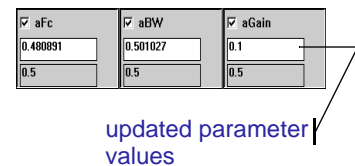
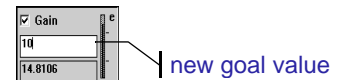
To investigate the parameter changes when changing Gain to a new target value of 10.

- 1 Make sure that automatic recalculation is selected.
 - a From the Options menu, select Recalculate.
 - b In the When frame, click Auto.
 - c Click OK.
- 2 Highlight the current value for Gain, and type 10.
- 3 Press .

The PSpice Optimizer automatically adjusts the parameters to satisfy the results, and then updates the results to match. The result changes from the value you specified to the nearest value which still satisfies all of the parameter limits.

In this case, the lower limit of the aGain parameter (0.1) is violated, so the optimizer uses the smallest allowed parameter value. This gives a value of 10.8 for Gain.

Note *Because the PSpice Optimizer computes estimates using the previously calculated derivatives, results are typically reliable for only small changes in parameter values. After significant tweaking, you should resimulate and recompute the derivatives to see true performance. See [“Ensuring reliable results when tweaking values”](#) on page 3-24.*



Completing Optimization

To finish exploring the active filter design, run an optimization.

To start optimizing

- 1 From the Tune menu, select Auto and click Start.

The PSpice Optimizer finds a set of parameters for which the specifications are met. Figure 6-2 shows the results.

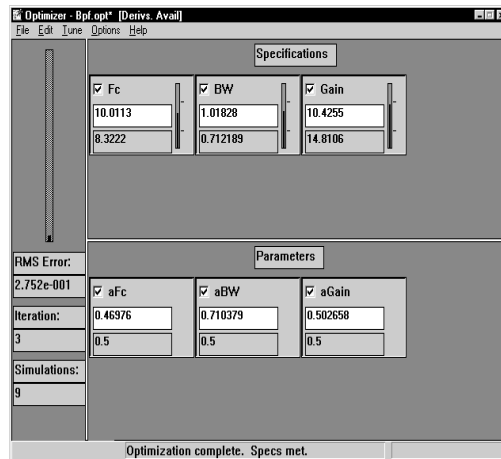


Figure 6-2 Optimized Values for the Active Filter Example

Note The number of simulations required to complete the optimization varies, depending on the starting values (in the current value fields) when you initiate optimization.

Tutorial: Using Constrained Optimization (MOS Amplifier)

7

Tutorial Overview

The following tutorial shows you how to use the PSpice Optimizer to set up and run constrained optimization. Using the MOS amplifier example provided with your MicroSim programs, you will:

- Review the optimization setup in Schematics: parameter definitions and parameterized expression assignments.
- Review the goal and constraint setup in the PSpice Optimizer including the evaluations used to measure performance.
- Review the optimization method selected for single-goal problems.
- Run the optimization.

The CMOS Amplifier Design

For this optimization, the CMOS amplifier requirements are:

- Minimize power consumption.
- Maintain gain at 20.
- Maintain 3 dB bandwidth at 1 MHz or greater.

To load the CMOS amplifier design

- 1 From the Windows 95 Start menu, select the MicroSim program folder and then the Schematics shortcut to start Schematics.
- 2 From the File menu, select Open.
- 3 Move to the directory containing `m2.sch` (`\MicroSim_root\examples\optimize\m2`) and select the schematic file in the File Name list.

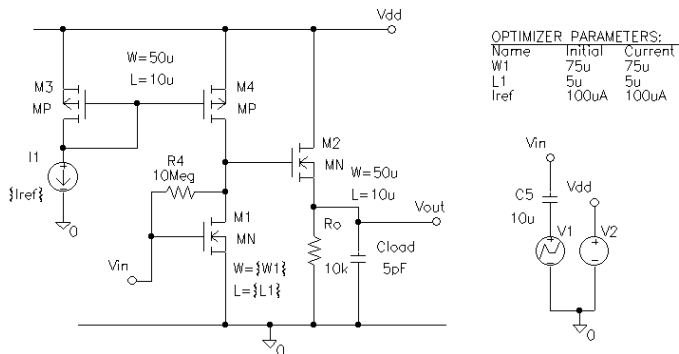


Figure 7-1 Schematic for CMOS Amplifier Example, `m2.sch`

The circuit consists of a common source stage (M1) with active load (M3 and M4) and a source follower (M2).

The Parameters

For this example, there are three circuit values that you must optimize, and three corresponding parameters that the PSpice Optimizer will vary:

- channel length for M1 ($W1$)
- channel width for M1 ($L1$)
- bias current for M3 and M4 (I_{ref})

Parameterized expression assignments In the schematic, you can see the parameterized component values for MOSFET M1 ($W=\{W1\}$, $L=\{L1\}$) and the current source I1 ($\{I_{ref}\}$).

In preliminary tests, the parameter values $W1 = 75 \text{ m}$, $L1 = 5 \text{ m}$, and $I_{ref} = 100 \text{ mA}$ produce the following performance characteristics:

gain	23.8
bandwidth	2.2 Mhz
power consumption	2.2 mW

These initial parameter values yield both excess gain and bandwidth, so a reduction in power consumption appears feasible. But because the gain, bandwidth, and power depend nonlinearly on circuit parameters such as transistor dimensions, manual optimization is impractical.

To display the Optimizer Parameters dialog box, double-click the OPTPARAM symbol in the schematic.

PSpice Optimizer parameters For optimization, the parameters for the amplifier are set up using the OPTPARAM symbol as follows:

Property	Parameters		
	W1	L1	Iref
Initial Value	75u	5u	100uA
Current Value	75u	5u	100uA
Lower Limit	10u	2u	10uA
Upper Limit	150u	50u	500uA
Tolerance	0	0	0

The Evaluations

The amplifier is set up with three performance characteristics: power, gain, and 3 dB bandwidth. To measure performance, the PSpice Optimizer needs to know how to calculate response for each of these.

To see the setup for the analyses:

- 1 In Schematics, from the Analysis menu, select Setup.
- 2 Click either the DC Sweep or AC Sweep button to browse the detail.
- 3 Click Cancel to return to the Analysis Setup dialog box.

Power The objective, to minimize power consumption, is a *single-point* goal. This means that a trace function appropriately measures the power response of the circuit.

The amplifier design is set up to measure power by performing a single-point DC analysis and then applying the trace function

$$-I(V2) * 10V$$

to the simulation results.

Gain The amplifier design is set up to measure the spot gain at 1 kHz (assuming that the bandwidth is much greater than this) by performing an AC analysis and then applying the Probe goal function:

```
YatX(V(Vout), 1k)
```

to the simulation results.

YatX is provided with your MicroSim programs. The goal function definition is as follows:

```
; value at given x
YatX(1, where) = y1
{ 1 | sf xvalue(when) !1;}
```

YatX(V(Vout), 1k) gives the Y value on the V(Vout) trace for the X value corresponding to 1 k.

3 dB bandwidth The amplifier design is set up to find the frequency where the output has fallen by 3 dB from its low-frequency value by performing an AC analysis (same as for Gain) and then applying the Probe goal function:

```
LPBW(Vdb(Vout), 3)
```

LPBW is provided with your MicroSim programs. The goal function definition is as follows:

```
; bandwidth of low pass response
LPBW(1, db_level) = y1
{ 1 | sf level(max-db_level, n) !1;}
```

LPBW(Vdb(Vout), 3) gives the low pass cutoff value on the Vdb(Vout) trace where the output is 3 dB below the maximum value on the trace.

The goal functions, YatX and LPBW, are contained in the file, msim.prb. Every MicroSim installation includes this file.

When loading a data file into Probe, Probe automatically loads the global msim.prb file (located, by default, in your MicroSim root directory), and any local .prb file (located in the working directory) corresponding to the current design.

You can freely update msim.prb or any other .prb file with new goal function definitions.

Refer to online Help in Probe for more information on the .prb files and Probe.

The Goals and Constraints

The example circuit is set up with predefined goals and constraints which you can view using the PSpice Optimizer.

To activate the PSpice Optimizer

- 1 In Schematics, from the Tools menu, select Run Optimizer.

The PSpice Optimizer window shows the three parameters (W1, L1, and Iref) and the three specifications. Power is defined as a goal, and gain and bandwidth are defined as constraints as shown below.

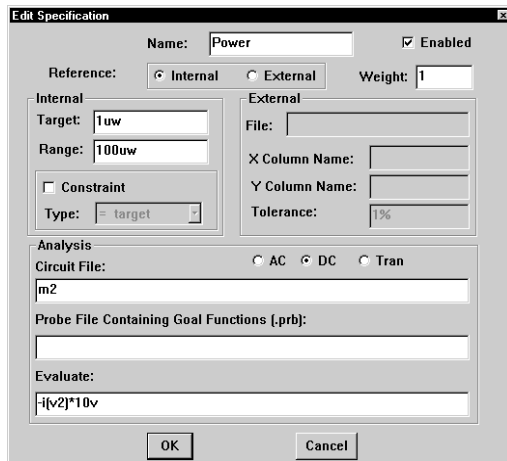
Setting	Power	Gain	Bandwidth
Name	Power	Gain	BW
Target	1uw	20	1Meg
Range	100uw	2	100k
Constraint Type	N/A *	= target **	>= target ***
Analysis	DC	AC	AC
Circuit File	m2	m2	m2
Evaluate	-i(v2)*10v	YatX (v(vout), 1k)	LPBW (vdb(vout),3)

*. Since Power is a goal, constraint type does not apply.

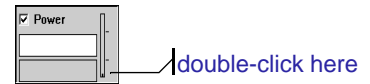
**. Gain is defined as a constraint because circuit gain *must* remain at 20±1.

***. BW is defined as a constraint because the 3 dB bandwidth *must* be at least 900 kHz or greater.

The Power settings as they appear in the Edit Specification dialog box are shown below.



To display the Edit Specifications dialog box, double-click the lower right-hand corner of the specification box of interest in the PSpice Optimizer window.



When finished browsing, click Cancel.

Setting the Method for a Single-Goal Optimization

The amplifier design is set up to use *minimization* to assess performance because this optimization has one goal: minimize the power consumption. The default method, *least squares*, is appropriate when minimizing the square of a function.

To see the Minimize setting, select Defaults from the Options menu, and click Advanced Options.

Performing the Optimization

You are now ready to run the optimization.

To optimize the amplifier design

- 1 From the Tune menu, select Update Performance to calculate initial performance. The results are as shown in Figure 7-2.

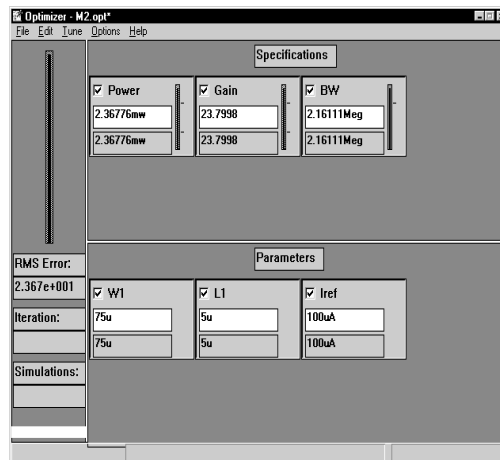


Figure 7-2 Updated Performance Values for the Amplifier Example

- 2 From the Tune menu, select Update Derivatives to compute the partial derivatives of each goal and constraint with respect to each parameter.
- 3 Display the derivatives:
 - a From the Tune menu, select Show Derivatives.
 - b Click Close, when finished.
- 4 From the Tune menu, select Auto and click Start to start the optimization.

Notice that the performance indicators change as the optimization proceeds. When complete, optimized values should appear as shown in Figure 7-3.

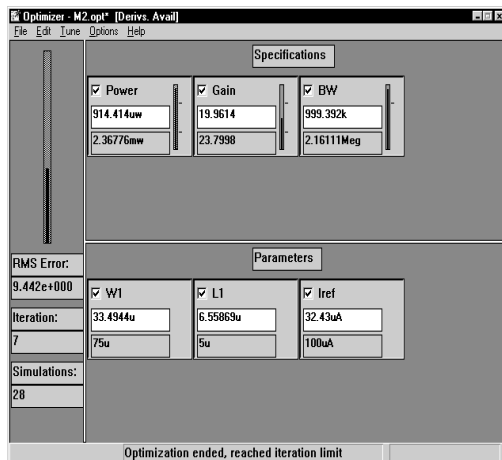


Figure 7-3 *Optimized Values for the Amplifier Example*

This optimization stops after reaching the maximum number of allowed iterations. Even so, power consumption is minimized (the objective) within the iteration limits.

Note *For a minimization problem like this one, you can still effectively use the PSpice Optimizer to improve the circuit's performance even though the goal is not attainable.*

Tutorial: Fitting Model Data (Bipolar Transistor)

8

Tutorial Overview

The following tutorial shows you how to use the PSpice Optimizer to fit a parameterized model to a set of measured data points. Using the bipolar transistor example provided with your MicroSim programs, you will:

- Review the bipolar transistor test case: schematic, parameter definitions and parameterized expression assignments, analysis, and external file with measured data.
- Review the goal setup in the PSpice Optimizer.
- Set up Probe and the PSpice Optimizer to monitor intermediate waveform results.
- Run the fitting process.

Using the PSpice Optimizer to Fit Data to Model Parameters

The PSpice Optimizer can fit a parameterized model to one or more sets of data points. The source for this *external* data might be:

- Results from measuring a real device (e.g., using a semiconductor curve tracer).
- A manufacturer's data sheet.
- A PSpice simulation.

To fit a model using the PSpice Optimizer, you need a text file containing the measured values and the measurement points.

Example: For static I-V characteristic of a diode, the external data file would contain pairs of values where each pair consists of the voltage at which the measurement was made and the current through the device at that voltage.

The PSpice Optimizer performs a *least squares* estimation of the parameter values that result in the best fit between the external data and the values produced by the model. More formally, given a set of N data points $\{x_i, y_i\}$, a model that has M parameters $\{a_j\}$, and a model relationship of the form:

$$y(x) = y(x; a_1 \dots a_M) ,$$

the optimizer chooses the parameters $\{a_j\}$ to minimize:

$$\sum_{i=1}^N [y_i - y(x_i; a_1 \dots a_M)]^2$$

The Bipolar Transistor Test Case

This tutorial fits parameters to a bipolar transistor (BJT) model using measured data for I_c and I_b versus V_{be} at constant V_{ce} .

To load the BJT design

- 1 From the Windows 95 Start menu, select the MicroSim program folder and then the Schematics shortcut to start Schematics.
- 2 From the File menu, select Open.
- 3 Move to the directory containing `bjtpar.sch` (`\MicroSim_root\examples\optimize\bjtpar`) and, in the File Name list, select the schematic file.

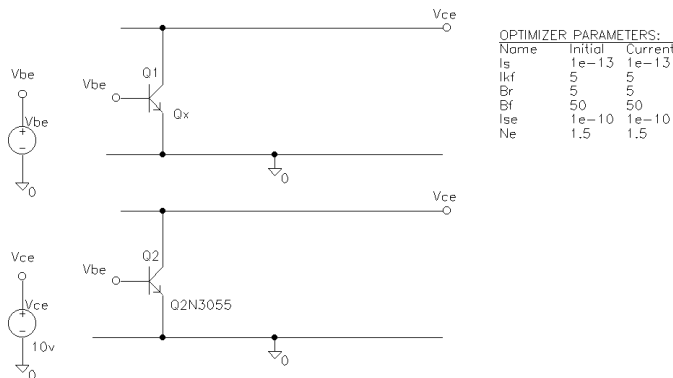


Figure 8-1 Schematic for the BJT Model Fitting Example

The circuit has voltage sources for V_{be} and V_{ce} and an instance, Q1, of a BJT. Q1 is a QbreakN device with its model reference set to Qx—the name of a BJT model.

The schematic also includes an instance of a 2N3055. This is the model that was used to produce the measured data. The 2N3055 in the schematic works as a reference during model fitting.

The QX model definition for this example is contained in the `bjtpar.inc` include file. To check whether this file is included, from the Analysis menu, select Library and Include Files.

The Parameters

Refer to Q devices in the online *MicroSim PSpice A/D Reference Manual* for more information on the parameters used for bipolar transistor models.

The model parameters to fit are: I_s , I_{kf} , B_r , B_f , I_{se} , and N_e . These parameters are a subset of the parameters PSpice uses to determine the BJT's DC characteristics. (To fit all of the BJT model parameters, you would need more measured data than is provided in this tutorial.)

To display the Optimizer Parameters dialog box, double-click the OPTPARAM symbol in the schematic.

PSpice Optimizer parameters There are six PSpice Optimizer parameters, each corresponding to one of the BJT model parameters listed above. The initial and current values for each of these parameters are set up using the OPTPARAM symbol as follows:

Property	Parameters					
	I_s	I_{kf}	B_r	B_f	I_{se}	N_e
Initial Value	1e-13	5	5	50	1e-10	1.5
Current Value	1e-13	5	5	50	1e-10	1.5
Lower Limit	1e-14	1	1	20	1e-11	1.2
Upper Limit	1e-11	10	10	200	1e-8	2
Tolerance	0	0	0	0	0	0

To see the parameterized expression assignments for the model parameters:

- 1 Click Q1.
- 2 From the Edit menu, select Model.
- 3 Click Edit Instance Model (Text).
- 4 When finished, click Cancel.

Parameterized expression assignments The model definition is set up with parameterized expression assignments for the model parameters the PSpice Optimizer will vary. All other model parameters are left unchanged.

Example: Using the Model Editor, the forward beta is specified as $b_f = \{b_f\}$.

The Analysis

The BJT example is set up for a DC sweep of the voltage source which provides a range of values for V_{be} . These values should match the measurement points contained in the external data file. I_c and I_b are measured at values of V_{be} starting from 0.4 V, incrementing by 0.01 V to a maximum of 0.82 V.

The External File of Measured Data

There are two measured curves: I_c and I_b . Each of these has an associated PSpice Optimizer specification. When fitting model parameters, specifications are different from those used in other kinds of optimizations because they reference an external data file.

For the BJT example, both sets of measured data are contained in the 3055.mdp file. A portion of this file is shown below.

Vbe	Ic	Ib
4.000E-01	6.047E-06	2.655E-06
4.100E-01	8.900E-06	3.253E-06
4.200E-01	1.310E-05	3.989E-06
4.300E-01	1.928E-05	4.897E-06
	•	
	•	
	•	
8.100E-01	8.002E+00	1.922E-01
8.200E-01	9.092E+00	2.371E-01

The file contains 3 columns of data: V_{be} , I_c , and I_b . The first non-blank line in the file contains names for the columns of data. These map to the specification setup in the PSpice Optimizer.

The Goals and Constraints

The example circuit is set up with predefined goals which you can view using the PSpice Optimizer.

To activate the PSpice Optimizer

1 From the Tools menu, select Run Optimizer.

The PSpice Optimizer window shows the six parameters (Is, Ikf, Br, Bf, Ise, and Ne) and the two specifications (Ic and Ib). Ic and Ib are defined as a goals as shown below.

Setting	Ic	Ib
Name	Ic	Ib
Reference	External	External
File	3055.mdp	3055.mdp
X Column Name	Vbe	Vbe
Y Column Name	Ic	Ib
Tolerance	5%	5%
Analysis	DC	DC
Circuit File	bjtpar	bjtpar
Evaluate	YatX(ic(q1),!)	YatX(ib(q1),!)

To display the Edit Specifications dialog box, double-click the lower right-hand corner of the specification box of interest in the PSpice Optimizer window.



double-click here

When finished browsing, click Cancel.

The Ib settings as they appear in the Edit Specification dialog box are shown below.

Edit Specification

Name: Enabled

Reference: Internal External Weight:

Internal

Target:
 Range:

Constraint
 Type:

External

File:
 X Column Name:
 Y Column Name:
 Tolerance:

Analysis

Circuit File: AC DC Tran

Probe File Containing Goal Functions (.prb):

Evaluate:

Mappings to the external data file Note the reference to the external data file, `3055.mdp`, and the values assigned to X Column Name and Y Column Name. The column values map to the column names defined in `3055.mdp`, such that the X column gives the data points at which the Y column values were measured. For the BJT example, the Y column contains measured collector current corresponding to the specified base-emitter voltages in the X column.

The evaluations To measure performance, both goals use the Probe goal function, `YatX` (see [The Evaluations on page 7-4](#)). Notice the ‘!’ character appearing in the expression. This character allows Probe to make measurements at the multiple X column values.

When the PSpice Optimizer encounters this character, it replaces it with the current X column value before sending it to Probe for evaluation.

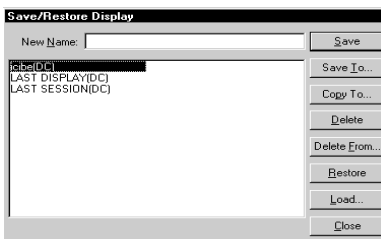
Example: If Evaluate is set to $Y_{atX}(I_C(Q1), !)$ and if the X column of the external data file contains the values (1v, 2v, 3v), then the PSpice Optimizer will form the Probe goal function expressions $Y_{atX}(I_C(Q1), 1v)$, $Y_{atX}(I_C(Q1), 2v)$, and $Y_{atX}(I_C(Q1), 3v)$, and send each one to Probe for evaluation against the simulation results.

Monitoring Progress with Probe

When optimizing, the PSpice Optimizer (with the help of PSpice) generates intermediate results which you can have sent to Probe for viewing. This is especially useful when fitting model parameters. For the BJT example, this means you can monitor how closely the I_c and I_b values match the target values.

To set up Probe and the PSpice Optimizer to monitor the fitting process

- 1 In Schematics, from the Analysis menu, select Simulate to simulate the circuit.
- 2 Create or use an existing Probe display configuration. For the BJT example, a display named `icibe` is predefined. To see what the `icibe` display looks like:
 - a In Probe, from the Tools menu, select Display Control.
 - b In the display list, click `icibe(DC)`.
 - c Click Restore.
 - d Click Close.
 - e From the File menu, select Close.



Refer to online Help in Probe for more on Probe display control.

- 3 Define the display configuration to use when optimizing. For the BJT example, this is predefined. To verify the display name:
 - a In the PSpice Optimizer, from the Options menu, select Defaults.

The display name, `icibe`, appears in the Display text box and the DC analysis type is selected.
 - b When finished, click Cancel.
- 4 Measure performance and redisplay the traces in Probe:
 - a In the PSpice Optimizer, from the Tools menu, select Update Performance.

The optimizer automatically updates the Probe display. When the iteration is complete, the Probe display should look something like the one shown in Figure 8-2. This display compares the curves for I_c and I_b with curves for an instance of a 2N3055. The percentage error for I_c and I_b is also displayed in the uppermost plot.

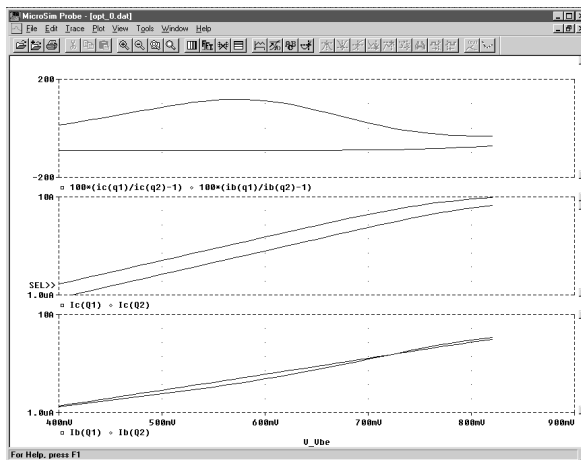
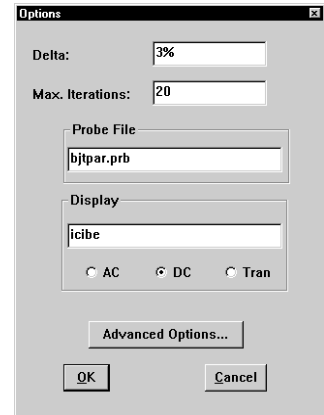


Figure 8-2 Initial Traces for the I_c and I_b Parameters



In some regions of the device characteristic, the measured currents are relatively insensitive to changes in the parameters. Because of this, the value of Delta is increased to 3% from its default value of 1%. See [Controlling Finite Differencing when Calculating Derivatives \(Delta Option\)](#) on page 4-13 for more information.

Fitting the Data

The PSpice Optimizer looks for a set of parameters which minimizes the total squared error between the measured and simulated curves. When using external data, the optimizer uses the relative error at each measured point. This means that all points have equal weight, regardless of the absolute value. This is essential for fitting semiconductor curves, where the currents may range over many orders of magnitude. Automatic normalization is discussed in detail in [Target Value Scaling on page 4-12](#).

To fit the data to the model parameters

- 1 From the Tune menu, select Auto and click Start.

The following table compares the model parameter values *estimated* by the PSpice Optimizer to the data values *measured* by PSpice for the 2N3055 transistor.

Parameter	Estimated by the PSpice Optimizer	Measured from PSpice
Is	9.744e-13	9.744e-13
Ikf	4.071	4.029
Br	5	2.949
Bf	98.465	99.49
Ise	9.920e-10	9.025e-10
Ne	1.960	1.941

The estimated parameters compare well with the original values.

The results of the optimization are shown in Figure 8-3.

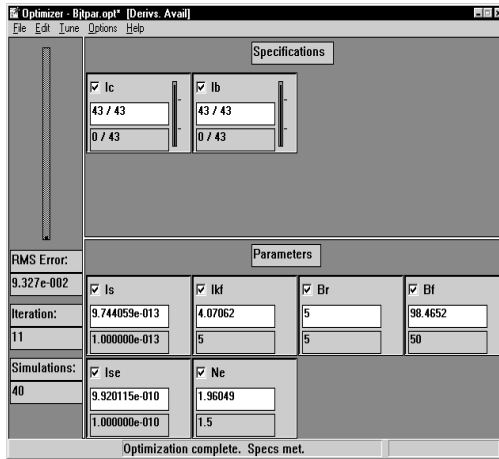


Figure 8-3 Optimization Results for the BJT Model Fitting Example

The Probe plot in Figure 8-4 compares fitted and measured curves, and shows the percent relative error in I_c and I_b .

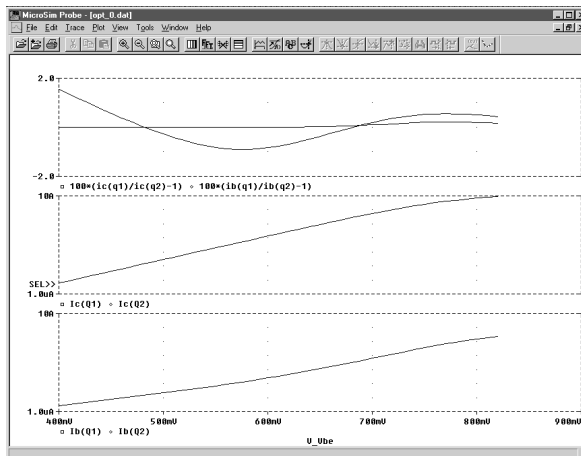


Figure 8-4 Probe Display after Optimization is Complete

Error Messages

A

Appendix Overview

This appendix lists and explains the error messages you might encounter while using the PSpice Optimizer, and, where appropriate, what action to take.

Error Message Descriptions

Table 8-1 *Error Message Descriptions*

Error Message	Description
[11002] No More Resets	No improvement was made by going back to the last good step and recalculating derivatives. Try refining the specifications.
[11003] Mismatched Parentheses	Check for mismatched parentheses in the evaluation expression.
[11004] Undefined Symbol	A symbol in a PSpice Optimizer expression was not recognized.
[11005] Divide by 0	A division by zero in a PSpice Optimizer expression occurred. Try adding a small offset, e.g., 1e-3, to the denominators of fractional terms in the expressions. You may need to change the lower or upper limit of a parameter.
[11006] 0 to the y	Try adding a small offset, e.g., 1e-3, to bases in PSpice Optimizer expressions. You may need to change the lower or upper limit of a parameter.
[11007] Unknown Operator	Valid operators and functions in PSpice Optimizer expressions are (+, -, *, /, **, exp, log, log10, sin, cos, tan, atan).
[11008] Invalid Number	PSpice Optimizer numbers are reals with an optional suffix (T, G, Meg, k, m, u, n, p, f, mil, %) and an optional symbol for units.
[11009] Log of Number <= 0	Log function (log or log10) can not have a zero argument. You may need to change the lower or upper limit of a parameter.
[11010] Can't Start PSpice	<p>Try running a PSpice simulation (with the PSpice Optimizer window still open). From the Windows 95 Start menu, select the MicroSim program folder and then the PSpice A/D (or PSpice) shortcut. Open a .cir file to run the simulation. Observe any messages that come up. Here are typical problems and resolutions:</p> <ul style="list-style-type: none"> • “insufficient memory”; at least 8 MB is required and 16 MB is recommended. • “Cannot obtain a license”. <p>In Schematics, from the Options menu, select Editor Configuration. Click App Settings. Verify that the correct path, used to run PSpice from Schematics, is set for PSPICECMD.</p>

Error Message	Description
[11011] Can't Start Probe	<p>Try running Probe (with the PSpice Optimizer window still open). From the Windows 95 Start menu, select the MicroSim program folder and then the Probe shortcut. Open a .dat file after simulation. Observe any messages that come up. Here are typical problems and resolutions:</p> <ul style="list-style-type: none">• “insufficient memory”; at least 8 MB is required and 16 MB is recommended.• “Can not obtain a license”. <p>In Schematics, from the Options menu, select Editor Configuration. Click App Settings. Verify that the correct path, used to start Probe from Schematics, is set for PROBECMD.</p>
[11012] .dat File In Use	<p>You must unload the data file from Probe for the circuit being optimized. Click on the Probe window to make it active, then select Close from the File menu.</p>
[11013] No Simulation Result	<p>Probe could not open the .dat file for the circuit being optimized. The PSpice Optimizer uses two files for optimization (opt_0.dat and opt_1.dat). Try running a PSpice simulation to investigate the problem. If PSpice does not create a .dat file, check that a DC sweep, AC sweep, or transient analysis has been selected, and that there is adequate hard disk space.</p>
[11014] G.F. Evaluation Failed	<p>Probe could not find one or more of the marked points in the goal function (error message will contain goal function name). Try loading the .dat file into Probe, select Eval Goal Function from the Trace menu to test the goal function. Refer to online Help in Probe for more information on writing goal functions.</p>
[11015] Aborted	<p>Optimization was aborted because of an error or user termination.</p>
[11016] Bad Numeric Field	<p>PSpice Optimizer numbers are real numbers with an optional suffix (T, G, Meg, k, m, u, n, p, f, mil, %) and an optional symbol for units.</p>
[11017] Name Required	<p>Specifications must be given names. Type in an ASCII string in the Name text box in the Edit Specification dialog box.</p>
[11018] MAX Must Be > MIN	<p>The current and initial values for a parameter must be greater than the Lower Limit. These values can be changed on the OPTPARAM symbol in Schematics, or select Parameters from the Edit menu in the PSpice Optimizer.</p>
[11019] Must Be Between MIN & MAX	<p>The Upper Limit of a parameter must be greater than the Lower Limit and less than the Upper Limit specified. These values can be changed on the OPTPARAM symbol in Schematics, or select Parameters from the Edit menu in the PSpice Optimizer.</p>

Error Message	Description
[11020] Evaluation Field Required	The Evaluate text box in the Specification dialog box must contain a valid Probe output variable, Probe goal function, or PSpice Optimizer expression. Refer to application examples in the your PSpice user's guide.
[11021] Column Name Required	The first line of an external data file must contain names for each column of data. The first column should contain the X axis data, and each succeeding column should contain Y axis data for each external specification. The column names (on the first line) must be identical to those used in the Specification dialog box (X Column Name, Y Column Name).
[11022] Would Exceed Max #Params	There is a maximum of 8 parameters allowed for an optimization.
[11023] Would Exceed Max #Specs	There is a maximum of 8 specifications allowed for an optimization.
[11024] All Params Are Disabled	At least one parameter must be selected to perform an optimization.
[11025] Can't Open File	The .opt file could not be opened by the PSpice Optimizer. Make sure that this file is not locked by a word processor or other application.
[11026] Problem With External Spec	See [11021] for a description of the external specification data file.
[11027] Max #Iterations Exceeded	The maximum number of iterations can be increased by selecting Defaults from the Options menu. The default limit is 20 iterations. It is often more helpful to refine the specifications than to increase this limit.
[11028] Terminated	The optimization was terminated by the user.
[11029] Can't Make Progress	No improvement was made by going back to the last good step and recalculating derivatives. Try refining the specifications.
[11030] No External File	The external file typed in the Edit Specification dialog box could not be found. See [11021] for a description of the format of this file.
[11031] External Spec - Bad Format	A column could not be found in the external data file. See [11021] for a description of the format for this file.
[11032] Unrecognized Expression	The expression used in the Evaluate text box of the Edit Specification dialog box could not be parsed. See errors [11014] and [11016] for aids in debugging the problem. PSpice Optimizer expressions can only use the following operators: (+, -, *, /, **, exp, log, log10, sin, cos, tan, atan).

Error Message	Description
[11033] Need At Least 1 Enabled Goal	At least one goal must be used in an optimization. An optimization can not be performed with constraints only.
[11034] Max External Items = 250	A maximum of 250 lines of data can be used in the external data file.
[11035] Missing External Column	The first line of an external data file must contain names for each column of data. The first column should contain the X axis data, and each succeeding column should contain Y axis data for each external specification. The column names (on the first line) must be identical to those used in the Specification dialogs (X Column Name, Y Column Name).
[11036] Eval Limits: 1 Goal, 1 Constraint	The evaluation version is limited to a maximum of 1 goal and 1 constraint.

File Types Used by the PSpice Optimizer

B

Appendix Overview

This appendix describes how MicroSim programs, used to capture and optimize your design, interact and share information with each other.

[File and Program Relationships on page B-2](#) illustrates and describes the data flow amongst MicroSim programs.

[File Type Summary on page B-6](#) explains the contents of each file type.

File and Program Relationships

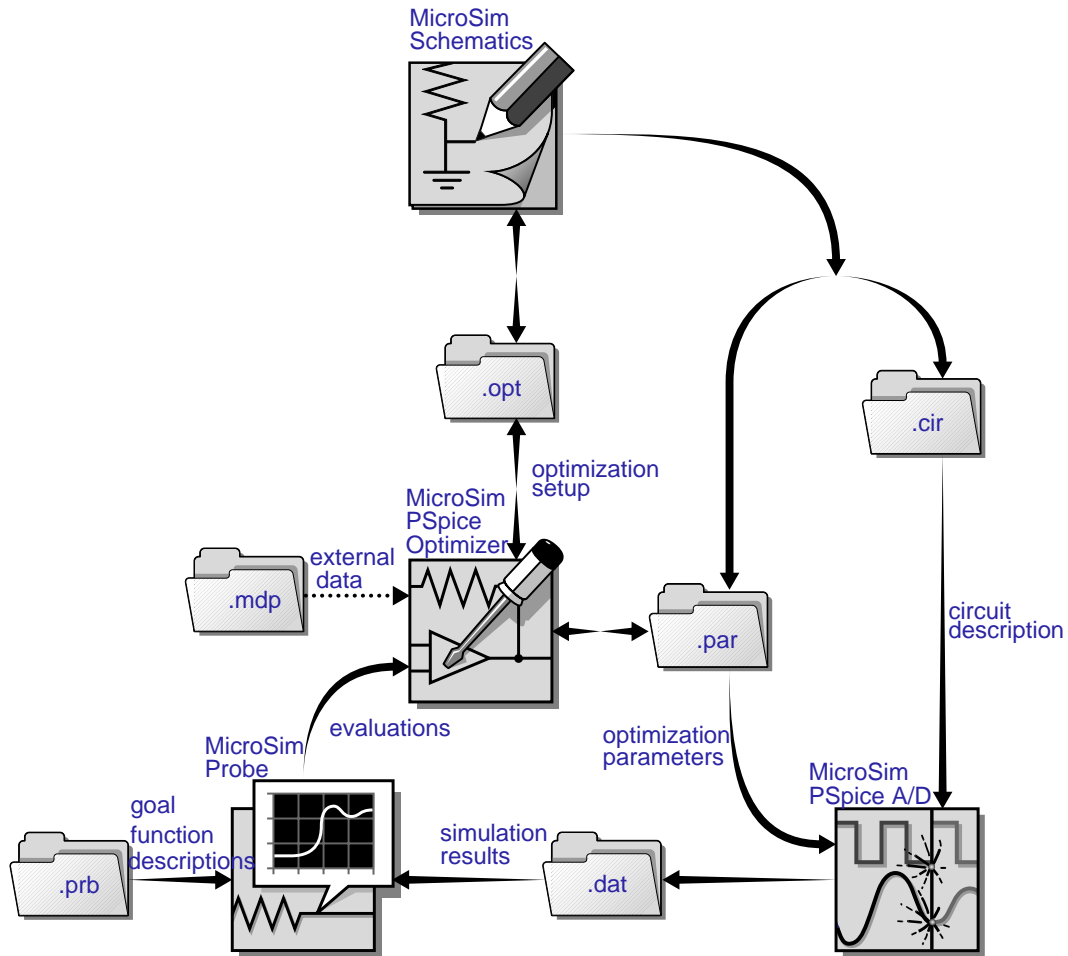


Figure 8-5 *MicroSim Program and File Interactions Important to Optimization*

Typically, you capture your design using Schematics and use this as the basis for optimization with the PSpice Optimizer. Each schematic produces a single circuit file (*design_name.cir*) which is created when either you:

- select Create Netlist or Simulate in the Analysis menu, or
- select Run Optimizer in the Tools menu.

In the latter case, Schematics also produces:

- An optimization file (*design_name.opt*) which contains optimization settings describing parameters, specifications, and other options. The PSpice Optimizer adds to and keeps the information in this file up-to-date.
- A parameters file (*design_name.par*) which contains parameter values as of the last optimization iteration. The PSpice Optimizer keeps the information in this file up-to-date.

Note *The circuit file produced by Schematics automatically references the .par file. If you are entering your design using a circuit file instead of a schematic, add a .inc (“include”) command naming the .par file. See [“Appendix C, Optimizing a Netlist-Based Design”](#) for details.*

Measuring Performance Using Information in the Circuit File and .prb File

Remember that for a simulation-based optimization, you must define how to evaluate performance. This means that, for each specification (goal or constraint), you must define the following.

What circuit file to use Typically, an optimization uses only one schematic/circuit file—based on the schematic that is active when you activate the PSpice Optimizer. But you can set up the optimization to use a different circuit file for each specification.

Which analysis to run The circuit file contains the analysis directives for any PSpice simulations (e.g., DC sweep, transient, etc.) that you set up for the schematic. However, you control which analyses are actually run for an optimization when you define how to evaluate each specification in the Edit Specifications dialog box.

How to measure performance If, to evaluate performance, your optimization is based on a multi-point simulation, you must define goal functions. Goal functions reside in the [GOAL FUNCTIONS] section of a .prb file. By default, a given design references the global .prb file shipped with all MicroSim programs (`msim.prb` in the MicroSim root directory) and a local .prb file (`design_name.prb` in the working directory).

Defining Specification Criteria in the External Data File

The external data file contains the measured data used in the optimization (e.g., when fitting data to model parameters). Data is formatted as labeled columns of values.

Typically, the first column contains the independent values at which all other data was measured; this corresponds to the X Column Name in the optimizer's Edit Specification dialog box. All other columns contain the dependent measured values and correspond to the Y Column Name in the dialog box.

To set up an external file

Using a standard text editor:

- 1 On line 1, enter the column headings for the data separated by spaces or tabs.
- 2 On all lines following line 1:
 - a Enter data values that correspond to the column headings, separated by spaces or tabs.
 - b Sort the lines of data in ascending X Column Name order.
- 3 Save the file using any file extension that does not conflict with other MicroSim standard file extensions. We recommend using the naming convention *design_name.mdp*.

For an example of an optimization that uses an external data file, see [Chapter 8, Tutorial: Fitting Model Data \(Bipolar Transistor\)](#).

Vbe	Ic	Ib
4.000E-01	6.047E-06	2.655E-06
4.100E-01	8.900E-06	3.253E-06
4.200E-01	1.310E-05	3.989E-06
4.300E-01	1.928E-05	4.897E-06
	•	
	•	
	•	
8.100E-01	8.002E+00	1.922E-01
8.200E-01	9.092E+00	2.371E-01

Figure B-1 Sample External Data File

File Type Summary

Table 8-2 briefly describes all of the file types that the PSpice Optimizer uses directly.

Table 8-2 *Summary of PSpice Optimizer-Related File Types*

File Type	Source	Description
.cir .net .als	generated by Schematics or user-entered with a text editor	Circuit file, netlist file, and alias file, respectively. Define components and connectivity, analysis directives, and simulation control directives for the circuit. Refer to your PSpice user's guide for more information.
.mdp	user-entered with a text editor	External data file. Contains measured data to which the optimization must adhere. (You can use any file extension for an external data file.)
.olg	generated by the PSpice Optimizer when running an optimization	Optimization log file. Contains an optimization audit trail that is useful as a debugging aid when the optimization doesn't converge. Have this file available when contacting Technical Support.
.oot	generated by the PSpice Optimizer when generating a report	Optimization report file. Provides a formatted description of the optimization specifications and parameters, and results including derivatives and Lagrange multipliers.
.opt	generated by Schematics when activating the PSpice Optimizer, or by the PSpice Optimizer when saving results	Optimization file. Defines the optimization settings for the circuit. When an optimization is complete, the PSpice Optimizer saves results (on command) to this file. Schematics also reads this file (on command) to back-annotate the schematic.
opt_0.dat opt_1.dat	generated by the PSpice Optimizer for each simulation in a run	Interim data files. Contain simulation results which are read by Probe to evaluate performance.
.par	generated by Schematics when activating the PSpice Optimizer and updated by the optimizer when running an optimization	Parameters file. Contains the latest optimization parameter values.

Optimizing a Netlist-Based Design

C

Appendix Overview

This appendix describes how to set up and run an optimization for a design defined in a circuit file.

[Optimizing without a Schematic on page C-2](#) gives an overview on optimizing a netlist-based design.

[Setting Up the Circuit File on page C-3](#) describes the steps that you must complete using a text editor to parameterize the circuit file for optimization.

[Setting Up and Running the PSpice Optimizer on page C-4](#) describes the steps that you must complete using the PSpice Optimizer to define parameters and specifications, run the optimization, and save the results.

[Example: Parameterizing the Circuit File on page C-6](#) steps through parameterizing a simple diode biasing circuit file.

Optimizing without a Schematic

Although the PSpice Optimizer is designed to optimize a Schematics-based design, you can optimize a design for which a netlist, but no schematic, is available.

To optimize your design without using Schematics

- 1** Implement your design as a circuit file in PSpice-compatible format.
- 2** Create a separate file containing the optimization parameter definitions.
- 3** Using the PSpice Optimizer, complete the setup and run the optimization as usual.

Refer to your PSpice user's guide for any questions you might have on circuit-file syntax. The remaining sections explain what you need to do once your design is defined as a circuit file.

Setting Up the Circuit File

To use the PSpice Optimizer to optimize a design defined as a netlist

- 1 Decide on the design parameters you want the optimizer to vary.
- 2 Place parameter definitions (.PARAM) in a separate parameters file (.par).
- 3 Include the parameters file in the circuit file using the .INC command.
- 4 Replace component values that are dependent on the parameters with expressions.

All of these steps use standard PSpice syntax; see your PSpice user's guide for details. The following section explains why you need to set up a separate parameters file, and how to do it (steps 2 and 3 above).

The parameters file (.PAR) When optimizing, the PSpice Optimizer uses a separate parameters file to track changes to parameter values with each iteration. This file contains nothing but PSpice parameter definitions reflecting the last-used values.

The optimizer automatically looks for a file with the name *circuit_file_name.par*.

Example: If the design is contained in *myamp.cir*, the optimizer writes updated parameter definitions to *myamp.par*.

To create a parameters file for a design

- 1 Activate any text editor.
- 2 For each parameter in the circuit, enter a line using the syntax:

```
.PARAM parameter_name = starting_value
```
- 3 Save the file as *circuit_file_name.par*.

To make the parameter definition in the .par file available to PSpice

- 1 Insert a `.INC` command anywhere after the first line of the circuit file using the syntax:

```
.INC parameters_file_name
```

where *parameters_file_name* is the same file in which you entered the `.PARAM` statements for the circuit.

Example: To `myamp.cir`, add the following statement anywhere after the first line in the file:

```
.INC "myamp.par"
```

Setting Up and Running the PSpice Optimizer

Before optimizing, you need to define for the PSpice Optimizer all of the parameters declared in the circuit file, and the goals and constraints. Setup is exactly the same as for a schematic-based design.

To complete setup and optimize the design

- 1 Activate the PSpice Optimizer by doing one of the following:
 - From the Windows 95 Start menu, select the MicroSim program folder and then the PSpice Optimizer shortcut.
 - From the Windows 95 Start menu, select Run, enter the command line for `optimize.exe`, and click OK.
- 2 Add a parameter definition for each of the parameters used in the circuit file:
 - a From the Edit menu, select Parameters.
 - b Click Add.
 - c Fill in the dialog box.

- 3** Add a specification definition for each goal and constraint:
 - a** From the Edit menu, select Specifications.
 - b** Click Add.
 - c** Fill in the dialog box.

In the Analysis frame, be sure to enter the name of the circuit file you parameterized earlier.
- 4** Run the optimization.
 - a** From the Tune menu, select Update Performance.
 - b** From the Tune menu, select Auto and click Start.

To save the optimization results

- 1** Write the results to the optimization file:
 - a** From the File menu, select Save As.
 - b** Enter the name of the optimization file (.opt) to contain the setup and results information.
- 2** From the File menu, select Report to create a PSpice Optimizer report file (.oot) containing a readable summary of the setup and results information.

Had you started with a design in Schematics, this step is equivalent to selecting Update Schematic from the Edit menu.

Example: Parameterizing the Circuit File

The following circuit file (call it `dbias.cir`) contains the netlist for a voltage source/resistor/diode series combination.

```
* diode bias circuit
.LIB
V1 1 0 5v
R1 1 2 5k
D1 2 0 D1N914
.DC V1 LIST 5V
.Probe
.END
```

Suppose you want to optimize the bias current in a voltage source/resistor/diode series combination. Specifically, your goal is to achieve a current of 1 mA through the diode. To realize this goal, you must vary the value of resistor R1 (call the variable value, or parameter, `Rbias`).

To parameterize the circuit file

- 1 Create the `dbias.par` file containing the parameter definition for `Rbias` using this syntax:

```
.PARAM Rbias = 5k
```

- 2 Parameterize the circuit file as shown (added or changed items are in bold):

```
* diode bias circuit
.INC "dbias.par"           ; this line added
.LIB
V1 1 0 5v
R1 1 2 {Rbias}           ; this line modified
D1 2 0 D1N914
.DC V1 LIST 5V
.Probe
.END
```

Now you are ready to activate the PSpice Optimizer, complete setup, and run the optimization.

Index

Symbols

!, 3-17
.als file, B-6
.cir file, B-3, B-6
.dat file, 4-11
.mdp file, B-5, B-6
.net file, B-6
.olg file, 3-28, B-6
.oot file, 3-27, B-6
.opt file, 3-2, 3-30, B-3, B-6
.par file, B-3, B-6, C-3
.prb file, 2-9, B-4

A

accuracy
 and Delta value, 4-15
 and derivatives, 4-11
 and failed convergence, 4-9
 and RELTOL, 4-15
 and Threshold value, 4-18
 improving, 4-7, 4-15
activation
 automatically loading an optimization file, 3-4
 changing startup options, 3-3

 from Schematics, 3-2
 from Windows 95, 3-3
 with a different initialization file, 3-3
active constraint, 4-6
Add, Parameters command (Edit menu), 3-10
Add, Specifications command (Edit menu), 3-13
Advanced Options, Defaults command (Options menu),
 4-17
alias file (.als), B-6
analysis type options, 3-15
Auto submenu, Tune menu, 3-19
automatic recalculation, 3-21, 3-22

B

back-annotation, 3-31
bound constraint, 4-4

C

Change, Parameter command (Edit menu), 3-11
Change, Specifications command (Edit menu), 3-16
circuit file (.cir), B-3, B-6
 as alternative to a schematic, C-2

- including a parameters file (netlist-based design), C-4
- Circuit File text box, 3-15
- component values
 - tolerance, 3-29
 - using standard, 3-29
- constraint, 1-6, 1-8
 - active, 4-6
 - bound, 4-4
 - compared to goals, 4-2
 - equality, 4-5
 - inactive, 4-6
 - inequality, 4-4
 - nonlinear, 4-4
 - progress indicator, 3-7
 - target value, 1-7
 - See Also specification
- Constraint check box, 3-14
- convergence, 4-9
- Copy, Parameters command (Edit menu), 3-11
- Copy, Specifications command (Edit menu), 3-16
- current value
 - changing interactively, 3-22, 3-23
 - external specification, 3-7
 - internal specification, 3-6
 - parameter, 3-8
- Current Value text box, 3-11
- Cutback option, 4-17

D

- Defaults command, Options menu, 4-13
- Delta option, 4-13
- derivatives, 1-11
 - accuracy, 4-11
 - and linearity, 4-11
 - and tweaking values, 3-21
 - calculating, 3-21, 4-10, 4-13
 - finite differencing, 4-10, 4-13
 - limitations, 4-11
 - viewing, 3-28
 - when excluding specifications, 3-25
- Display Control option, 4-16

E

- Edit menu
 - Parameters command
 - Add, 3-10

- Change, 3-11
- Copy, 3-11
- Insert, 3-11
- Reset Values command, 3-25, 3-26
- Round Calculated command, 3-30
- Round Nearest command, 3-29
- Specifications command
 - Add, 3-13
 - Change, 3-16
 - Copy, 3-16
 - Insert, 3-16
- Store Values command, 3-26
- Update Schematic command, 3-31
- Edit Parameter dialog box
 - Current Value, 3-11
 - Enabled, 3-11
 - Initial Value, 3-11
 - Initial Value text box, 3-11
 - Lower Limit, 3-11
 - Name, 3-11
 - Tolerance, 3-11
- Edit Specification dialog box
 - analysis type, 3-15
 - Circuit File, 3-15
 - Constraint, 3-14
 - Enabled, 3-14
 - Evaluate, 3-15
 - File, 3-15
 - Name, 3-14
 - Range, 3-14
 - Reference, 3-14
 - Target, 3-14
 - Tolerance, 3-15
 - Type, 3-14
 - Weight, 3-14
 - X Column Name, 3-15
 - Y Column Name, 3-15
- Enabled check box, 3-11, 3-14
- equality constraint, 4-5
- error gauge area, 3-9
- Evaluate text box, 3-15
- evaluation, 1-9
 - for external specifications, 3-17
 - See Also goal function, Probe
 - See Also PSpice Optimizer expression
 - See Also trace function, Probe
- examples
 - active filter, 6-1
 - bipolar transistor, 8-1
 - diode biasing circuit (primer), 2-2

- MOS amplifier, 7-1
- parameterizing a diode-biasing circuit file, C-6
- passive terminator, 5-1
- exploration, design, 3-20
 - active filter example, 6-1
 - ensuring reliable results, 3-24
 - tweaking values, 3-21
- expression, PSpice Optimizer, 1-9, 1-10
- external data file (.mdp), B-5, B-6
- external specification
 - progress indicator, 3-7
 - setting up the data file, B-5

F

- File menu
 - New command, 3-4
 - Open command, 3-4
 - Print command, 3-28
 - Report command, 3-27
 - Save As command, 3-30
 - Save command, 3-30
- File text box, 3-15
- files
 - alias (.als), B-6
 - circuit (.cir), B-3, B-6
 - external data (.mdp), B-5, B-6
 - log (.olg), 3-28, B-6
 - netlist (.net), B-6
 - opt_0.dat, B-6
 - opt_1.dat, B-6
 - optimization (.opt), 3-2, 3-30, B-3, B-6
 - parameter (.par), C-3
 - parameters (.par), B-3, B-6
 - Probe goal functions (.prb), 2-9, B-4
 - report (.oot), 3-27, B-6
- fitting data to model parameters, 8-10
 - bipolar transistor example, 8-1

G

- global minima, 4-8
- goal, 1-6, 1-7
 - compared to constraints, 4-2
 - progress indicator, 3-7
 - target value, 1-7
 - See Also specification
- goal function, Probe, 1-10
 - defining .prb file, 2-9, B-4

- discontinuities, 1-10

I

- inactive constraint, 4-6
- inequality constraint, 4-4
- initial value
 - external specification, 3-7
 - internal specification, 3-6
 - parameter, 3-8
- Initial Value text box, 3-11
- Insert, Parameters command (Edit menu), 3-11
- Insert, Specifications command (Edit menu), 3-16
- interactive design exploration, 2-3
- interim data file (opt_x.dat), B-6
- iterations
 - and Cutback, 4-17
 - controlling parameter value changes, 4-17
 - limiting, 4-15

L

- Lagrange multipliers, 4-4, 4-6
- least squares, 4-19
 - constrained, 1-5
 - unconstrained, 1-5
- Least Squares option, 4-20
- local minima, 4-8
- log file (.olg), B-6
 - viewing, 3-28
- Lower Limit text box, 3-11

M

- manual recalculation, 3-23
- Max. Iterations option, 4-15
- minima
 - and starting points, 4-8
 - constrained, 4-5
 - global, 4-8
 - local, 4-8
- minimization, 4-19
 - bound-constrained, 4-9
 - constrained, 1-5
 - unconstrained, 1-5, 4-10
- Minimize option, 4-20
- model parameters, fitting, 8-10
 - bipolar transistor example, 8-1

N

Name text box, 3-11, 3-14
netlist file (.net), B-6
netlist-based design, C-2
New command, File menu, 3-4
nonlinear constraint, 4-4

O

Open command, File menu, 3-4
opt_0.dat file, B-6
opt_1.dat file, B-6
optimization, 1-5

- aborting, 3-19
- adding/editing parameters and specifications, 3-25
- and specification functions, 4-7
- choosing least squares or minimization, 4-19
- constrained least squares, 1-5
- constrained minimization, 1-5
- constrained vs. unconstrained, 4-3
- controlling parameter
 - perturbation, 4-13
 - value changes, 4-17
- excluding parameters and specifications, 3-24
- for one goal, 4-19
- improving convergence, 4-9
- limiting iterations, 4-15
- log, 3-28
- loosening parameter bounds, 4-10
- purpose, 2-3
- restoring previous results, 3-26
- running, 3-19
- saving final results, 3-30
- saving intermediate values, 3-26
- scaling, 4-12
- setting Cutback value, 4-17
- setting the Probe display, 4-16
- starting points, 4-5
- tweaking values, 3-21
- unconstrained least squares, 1-5
- unconstrained minimization, 1-5
- using a netlist-based design, C-2

optimization file (.opt), 3-2, B-3, B-6

- loading, 3-2, 3-4
- saving results, 3-30

optimizer window, 3-24

- error gauge area, 3-9
- parameters area

- current value, 3-8
- edit hot spot, 3-24
- initial value, 3-8

specifications area, 3-6

- current value, 3-6, 3-7
- edit hot spot, 3-24
- Enable check box, 3-24
- initial value, 3-6, 3-7
- progress indicator, 3-7

options

- activation, 3-3
- Cutback, 4-17
- Delta, 4-13
- Display Control, 4-16
- Least Squares, 4-20
- Max. Iterations, 4-15
- Minimize, 4-20
- Threshold, 4-17

Options menu

- Defaults command, 4-13
 - Advanced Options, 4-17
- Recalculate command, 3-22, 3-23

OPTPARAM symbol, Schematics, 2-7, 3-2, 3-8, 3-31

P

parameter, 1-6

- adding from scratch, 3-10
- back-annotating values to the schematic, 3-31
- bounds, 4-9
- controlling changes between iterations, 4-17
- controlling perturbation, 4-13
- copying, 3-11
- Edit Parameter dialog box controls, 3-11
- editing, 3-12
- enable/disable, 3-24
- excluding from optimization, 3-24
- fitting, 8-10
- loosening bounds, 4-10
- tolerance on component values, 3-29
- tweaking values, 3-21
- using standard component values, 3-29

parameters area, Optimizer Window

- Enable check box, 3-24

parameters area, optimizer window, 3-8

- current value, 3-8
- edit hot spot, 3-24
- initial value, 3-8

parameters file (.par), B-3, B-6, C-3

creating for netlist-based designs, [C-3](#)
 including in a circuit file (netlist-based design), [C-4](#)
 Parameters, Recalculate command (Options menu), [3-23](#)
 performance, [1-8](#)
 derivative calculations, [3-21](#)
 evaluating, [3-19](#)
 excluding parameter and specifications, [3-24](#)
 measuring
 when adding/editing parameters and specifications, [3-25](#)
 recalculating
 automatically, [3-22](#)
 manually, [3-23](#)
 saving results, [3-30](#)
 scaling measured values, [4-12](#)
 tweaking values, [3-21](#)
 Print command, File menu, [3-28](#)
 Probe, [1-4](#), [1-9](#)
 .prb file, [2-9](#), [B-4](#)
 data file (.dat), [4-11](#)
 display, [4-16](#)
 goal function, [1-10](#)
 See Also goal function, Probe
 monitoring simulations, [3-19](#)
 trace function, [1-9](#)
 Probe data file (.dat), [4-11](#)
 Probe goal functions file (.prb), [2-9](#), [B-4](#)
 progress indicator
 external specification, [3-7](#)
 internal specification, [3-7](#)
 PSpice, [1-4](#)
 PSpice Optimizer
 activating, [3-2](#)
 loading, [3-2](#), [3-4](#)
 PSpice Optimizer expression, [1-9](#), [1-10](#)

R

Range text box, [3-14](#)
 Recalculate command, Options menu, [3-22](#), [3-23](#)
 recalculation, [3-21](#)
 automatic, [3-22](#)
 manual, [3-23](#)
 Reference frame, [3-14](#)
 RELTOL option, [4-15](#)
 Report command, File menu, [3-27](#)
 reports, [3-26](#)
 .oot file, [3-27](#)
 generating, [3-27](#)
 printing, [3-27](#)
 reports file (.oot), [3-27](#), [B-6](#)
 requirements, see specifications, [1-9](#)
 Reset Values command, Edit menu, [3-25](#), [3-26](#)
 results
 .opt file, [3-30](#)
 improving convergence, [4-9](#)
 restoring, [3-26](#)
 saving final values, [3-30](#)
 saving for netlist-based design, [C-5](#)
 saving intermediate values, [3-26](#)
 setting the Probe display, [4-16](#)
 Results, Recalculate command (Options menu), [3-23](#)
 RMS error, [3-9](#)
 Round Calculated command, Edit menu, [3-30](#)
 Round Nearest command, Edit menu, [3-29](#)

S

Save As command, File menu, [3-30](#)
 Save command, File menu, [3-30](#)
 Schematics, [1-4](#)
 back-annotating, [3-31](#)
 defining optimization parameters (OPTPARAM symbol), [2-7](#)
 OPTPARAM symbol, [3-2](#), [3-8](#), [3-31](#)
 Show Derivatives command, Tune menu, [3-28](#)
 simulation
 monitoring with Probe, [3-19](#)
 single-point analyses, [1-9](#)
 specification, [1-6](#)
 adding from scratch, [3-13](#)
 conflicting, [1-9](#)
 copying, [3-16](#)
 Edit Specification dialog box controls, [3-14](#)
 enable/disable, [3-24](#)
 excluding from optimization, [3-24](#)
 external, [1-7](#), [3-7](#), [B-5](#)
 function behavior and accuracy, [4-7](#)
 internal, [1-6](#), [3-6](#)
 target value, [1-7](#)
 tweaking values, [3-21](#)
 See Also constraint
 See Also goal
 specifications area, Optimizer Window
 Enable check box, [3-24](#)
 specifications area, optimizer window, [3-6](#)
 edit hot spot, [3-24](#)
 external

- current value, 3-7
- initial value, 3-7
- internal
 - current value, 3-6
 - initial value, 3-6
 - progress indicator, 3-7
- progress indicator, 3-7
- Start command, Auto submenu (Tune menu), 3-19
- starting points
 - and global and local minima, 4-8
 - and parameter bounds, 4-10
 - feasible, 4-5
 - improving convergence, 4-9
 - infeasible, 4-5
- Store Values command, Edit menu, 3-26
- subgoal
 - progress indicator, 3-7

T

- Target text box, 3-14
- target value, 1-7
 - scaling raw measurements, 4-12
- Terminate command, Auto submenu (Tune menu), 3-19
- Threshold option, 4-17
- Tolerance text box, 3-11, 3-15
- tolerance, component values, 3-29
- trace function, Probe, 1-9
- tradeoffs, design, 2-3, 3-20
 - active filter example, 6-1
 - and starting points, 4-5
 - between goals and constraints, 4-2
 - ensuring reliable results, 3-24
 - tweaking values, 3-21
- Tune menu
 - Auto submenu, 3-19
 - Start command, 3-19
 - Terminate command, 3-19
 - Show Derivatives command, 3-28
 - Update Derivatives command, 3-22, 3-23
 - Update Performance command, 3-19
- Type list (constraints), 3-14

U

- Update Derivatives command, Tune menu, 3-22, 3-23
- Update Performance command, Tune menu, 3-19
- Update Schematic command, Edit menu, 3-31

W

- Weight text box, 3-14

X

- X Column Name text box, 3-15

Y

- Y Column Name text box, 3-15