Avant!

TCAD Business Unit

IMPORTANT NOTICE

For Users of Medici Version 2000.2

Appendix C of this User's Manual contains a complete description of the changes in Medici 2000.2.
If you are a user of Medici 1999.4, read Appendix C carefully before using version 2000.2 to make efficient use of the program.

Reader Comments: *Medici 2000.2 User's Manual*

Avant! TCAD Business Unit welcomes your comments and suggestions concerning this manual. Please mail this form (Attn.: Technical Publications Department) to the address on the reverse side of this sheet, or fax a copy to (510) 413-7743, or e-mail your comments to tcad-pubs@avanticorp.com. Attach additional pages if needed.

What model computer are you using?_____Operating system?_____

Did you find any errors in this manual? If so, please list the page number and describe the error.

Have you encountered program features that need to be better described in this manual?

What additional information should be included?

How can we improve this document?

Other comments and suggestions:

_____Fold here and tape_____

To:
Avant! Corporation
TCAD Business Unit, Bldg. 2
Attn: Technical Publications Dept.
46871 Bayside Parkway
Fremont, CA 94538
USA

_____Fold here and tape_____

From:
(Optional Information)

Name, Position: .....

Company .....

Address: .....

.....

.....

Phone, fax, or e-mail.....

Medici

Two-Dimensional Device Simulation Program

Version 2000.2

User's Manual

July, 2000

Avant! Corporation, TCAD Business Unit Fremont, California

Contacting Avant! Corporation:

Telephone: (510) 413-8000
(800) 369-0080
FAX: (510) 413-7743
e-mail tcad-app@avanticorp.com
Internet: http://www.avanticorp.com/

Avant! Corporation
TCAD Business Unit
46871 Bayside Parkway
Fremont, CA 94538

Table of Contents

Diode and Lumped Element Examples 6-1

Photogeneration Examples 7-1

Programmable Device Examples 11-1

Circuit Analysis Examples 12-1

Appendix A: Template Files A-1

Appendix B: *Avant!* TCAD Graphics B-1

Appendix D: Technology Interchange Format (TIF) D-1

Appendix E: Medici and STUDIO Command Editor E-1

Appendix F: Limitations of IBM SP2 Versions of Medici F-1

List of Figures

Introduction to Medici

Overview

Medici is a powerful device simulation program that can be used to simulate the behavior of MOS and bipolar transistors, and other semiconductor devices. It models the two-dimensional distributions of potential and carrier concentrations in a device. The program can be used to predict electrical characteristics for arbitrary bias conditions.

Analyzing Devices and Effects

The program solves Poisson's equation and both the electron and hole current-continuity equations to analyze devices such as diodes and bipolar transistors, as well as effects in which the current flow involves both carriers, such as CMOS latchup. Medici can also analyze devices in which current flow is dominated by a single carrier, such as MOSFETs, JFETs, and MESFETs. In addition, Medici can be used to study devices under transient operating conditions.

Submicron Simulation

Medici simulates the behavior of deep submicron devices by providing the ability to solve the electron and hole energy balance equations self-consistently with the other device equations. Effects, such as carrier heating and velocity overshoot, are accounted for in Medici and their influence on device behavior can be analyzed.

Medici Grid

Medici uses a non-uniform triangular simulation grid and can model arbitrary device geometries with both planar and nonplanar surface topographies. The simulation grid can also be refined automatically during the solution process. Addi-

tional nodes and elements can be added where a user-specified quantity, such as potential or impurity concentration, varies by more than a specified tolerance over existing mesh elements. This flexibility makes modeling of complicated devices and structures possible.

Electrodes can be placed anywhere in the device structure. Impurity distributions can be created by combining Medici's analytic functions with input from *Avant!* TCAD's process modeling programs, TMA SUPREM-3 and TSUPREM-4, and input from text files containing the impurity distributions.

Physical Models

A number of physical models are incorporated into the program for accurate simulations, including models for recombination, photogeneration, impact ionization, band-gap narrowing, band-to-band tunneling, mobility, and lifetime. Medici also incorporates both Boltzmann and Fermi-Dirac statistics, including the incomplete ionization of impurities.

Additional Features

Other features that make Medici a powerful tool include the ability to:

- Attach lumped resistive, capacitive, and inductive elements to contacts
- Specify distributed contact resistances
- Specify either voltage and current boundary conditions during a simulation
- Automatic I-V curve tracing
- Perform an AC small signal analysis at virtually any frequency in order to calculate frequency-dependent capacitances, conductances, admittances, and S-parameters

Advanced Application Modules

Advanced Application Modules (AAM) are optionally available. They provide some unique and powerful capabilities for special purpose applications. The presently available AAMs are introduced in Advanced Application Modules on page lxii, and are discussed in more detail in Chapter 2.

Backward Compatibility

Medici maintains backward compatibility with all previous versions of the program including TIF and structure files generated by previous versions. Refer to Chapter 3, Section 3.7 Old Statements for a listing of statements whose functions have been superseded by newer additions to the program.

Manual Overview

This manual contains 17 chapters, six appendices, and an index. Of particular note is Chapter 2, which describes the physical models that form the basis of Medici. Chapter 3, contains the input commands recognized by the program. Chapters 4 through 17 show examples of using Medici to simulate and analyze typical device behaviors.

Note:

Examples used as illustrations in this manual are not intended for use with actual simulations. They are presented as guidelines only.

Chapter 1 Discusses the execution of Medici, the required input files, the output files generated, and other files required to execute the program.

Chapter 2 Provides a description of the Medici program. This includes the physical models, the numerical methods, and the simulation grid used by the program. It also describes the Advanced Application Modules that are optionally available for Medici.

Chapter 3 Contains descriptions of the input statements recognized by Medici. The description of each statement includes a summary of the statement syntax, descriptions of the statement parameters, and a discussion of the use of the statement.

Chapter 4 Presents examples that illustrate the simulation and analysis of an n-channel MOSFET.

Chapter 5 Presents examples that illustrate the simulation and analysis of an NPN bipolar transistor.

Chapter 6 Presents examples that illustrate the transient analysis capabilities of Medici by studying the turn-on characteristics of a PN diode. It also presents examples that illustrate the use and effect of external lumped elements and contact resistance.

Chapter 7 Presents two examples that illustrate the photogeneration capabilities of Medici. The first example models a back-lit solar cell. The second is a simulation of single-event upset in an SRAM cell.

Chapter 8 Illustrates the use of parameterized template files for the automatic creation and simulation of MOS and bipolar structures.

Chapter 9 Presents examples that include the solution of the energy balance equation in the simulations. Both MOS and bipolar devices are studied.

Chapter 10 Presents examples that illustrate the interfaces between Medici and the process simulation and parameter extraction programs.

Chapter 11	Presents examples that illustrate the use of the Programmable Device AAM. For this purpose, the writing and erasing characteristics of a Flash EEPROM device are simulated.
Chapter 12	Presents examples that illustrate the use of the Circuit Analysis AAM.
Chapter 13	Presents examples that illustrate the use of the Lattice Temperature AAM.
Chapter 14	Presents examples that illustrate the use of the Heterojunction Device AAM.
Chapter 15	Presents examples that illustrate the use of the Trapped Charge AAM.
Chapter 16	Presents examples that illustrate the use of the Optical Device AAM.
Chapter 17	Presents examples that illustrate the use of the Anisotropic Material AAM.
Appendix A	Describes the structure and use of the parameterized template files for the automatic creation and simulation of MOS and bipolar devices.
Appendix B	Discusses *Avant!* TCAD graphics and describes the *mdpdev* plot device definition file. This file contains information that controls use of the graphics output devices.
Appendix C	Describes the program enhancements in Medici Revision 2000.2 relative to Revision 1999.4.
Appendix D	Describes the Technology Interchange Format (TIF) for interprogram communication.
Appendix E	Describes the limitations of using Medici with the STUDIO Command Editor.
Appendix F	Describes limitations of using Medici with IBM SP2 platform.

Advanced Application Modules

Advanced Application Modules (AAM) for Medici are optionally available from *Avant!* TCAD Business Unit. They give the program additional capabilities for special purpose applications. One or more may be licensed in addition to the Medici program. AAMs are described in detail in Chapter 2, pages 2-123 through

2-159. Input statements and parameters used with the AAMs are discussed in Chapter 3. Presently available AAMs include:

Programmable Device AAM, page 2-123	Provides the ability to simulate the programming characteristics of nonvolatile memory devices, such as EPROMs, EEPROM, and Flash EEPROMs.
Circuit Analysis AAM, page 2-127	Provides the ability to perform circuit simulation where the active circuit elements can include Medici numeric devices.
Lattice Temperature AAM, page 2-129	Provides the ability to solve the lattice heat equation self-consistently with the other device equations. This makes it possible to analyze the effects of lattice heating on a structure's electrical performance.
Heterojunction Device AAM, page 2-134	Provides the ability to simulate the behavior of a variety of heterojunction devices, such as HBTs and HEMTs.
Trapped Charge AAM, page 2-148	Provides the ability to simulate the behavior of a wide variety of devices containing traps, such as TFTs and power devices. It also provides the ability to simulate insulator traps; therefore it can be used to perform reliability simulations.
Optical Device AAM, page 2-150	Provides the ability to automatically calculate photogeneration rates that occur in a variety of optical devices, such as CCDs, photodetectors and solar cells.
Anisotropic Material AAM, page 2-159	Provides the ability to account for the anisotropic behavior of some semiconductor materials, such as SiC.

Note:

AAM input statements and parameters discussed in this manual have no effect on a simulation unless the appropriate AAMs are licensed and installed.

Typeface Conventions

The following typeface conventions are used in this manual:

Typeface	Used for
`STATEMENT`	Commands, or keyboard information you type, appear in this bold, fixed-width typeface. **`SILICON`** is an example of a parameter in this typeface.
`output text`	Text output by Medici or your system appears in this typeface. Output files are an example of this typeface.

Typeface	Used for
<*pathname*>	Variable information you type, which must be replaced with specific text, is indicated in italics enclosed by angle brackets (< >). The plot device definition file *mdpdev* is an example of this convention. Do ***not*** type the angle brackets when entering your text.

Related Publications

This manual covers all aspects of the Medici 2D Simulation program. For information on Medici installation procedures, see the *Avant! TCAD Product Installation Manual*.

Reference Materials

This manual uses many references from the changing body of literature in the industry. Where appropriate, you are directed to source material. A reference section is included in Chapter 2, beginning on page 2-175.

Problems and Troubleshooting

If you have problems or questions regarding Medici operation, first check the UNIX window from which you started Medici for warning or error messages:

- For help in resolving UNIX system errors (`cannot create` <*file*>`: Permission denied`, and others), please see your UNIX systems administrator.
- For Medici-specific problems, please see the person who installed this product or associated *Avant!* TCAD products in your company. Usually this is your UNIX systems administrator or the CAD manager.

For further help, please contact *Avant!* TCAD or *Avant!* TCAD's representative in your area.

Using Medici

Introduction

This chapter discusses starting Medici, required input files, output files generated, and miscellaneous files required to execute the program. The chapter includes discussions of the following:

- Execution command
- Syntax errors
- Error and warning messages
- Program output
- File specification
- Required input files
- Output files generated
- Miscellaneous files required to execute the program

Program Execution and Output

This section describes the Medici execution command, syntax and execution errors, and types of program output.

Execution Command

Execution of Medici is initiated with the following command:

```
medici
```

or

```
medici <filename>
```

where <*filename*> is the name of a command input file (see "Command Input File" on page 1-8 and Figure 1-1). The program responds by displaying a header

identifying the program version on your terminal. If <*filename*> is not specified, you will be prompted for this information:

```
 ************************************************************
 ***                       MEDICI (TM)                    ***
 ***                    Version 1999.4.0                  ***
 ***                System S (Sun: Solaris)               ***
 ***          Unpublished Copyright (C) 1991-1999         ***
 ***               Avant! Corp. and TMA, Inc.             ***
 ***                  All Rights Reserved                 ***
 ***     Avant! Corporation Proprietary and Confidential  ***
 ***                                                      ***
 ***    Your use of this program and its documentation are ***
 ***   covered by a license agreement, a copy of which may ***
 ***    be found in the file "license.txt" located in the ***
 ***   install directory.  Your use of this program is your ***
 ***     consent to be bound by its terms and conditions. ***
 ***                                                      ***
 ***       Use of copyright notice is precautionary and   ***
 ***        does not imply publication or disclosure.     ***
 ***                                                      ***
 ***     Use, duplication or disclosure by the Government ***
 ***        is subject to restrictions as set forth in    ***
 ***        subparagraph (c) (1) (ii) of the Rights in    ***
 ***      Technical Data and Computer Software clause at  ***
 ***                   DFARS 252.227-7013.                ***
 ***                                                      ***
 ***           Technology Modeling Associates, Inc.       ***
 ***             is a wholly owned subsidiary of          ***
 ***                  Avant! Corporation                  ***
 ***                 46871 Bayside Parkway                ***
 ***                   Fremont, CA 94538                  ***
 ***                                                      ***
 ***         Unpublished - Rights reserved under the      ***
 ***                 copyright law of the U.S.            ***
 ***                                                      ***
 ***   MEDICI is a trademark of Avant! Corp. and TMA, Inc.***
 ************************************************************

                        11-Feb-2000 10:59:28

Enter the input file name (press return for interactive mode)
File name:
```

Figure 1-1 Medici command prompt

Note:

The file specification must conform to conventions in the operating system and cannot contain more than 80 characters.

If the file specification is blank, the program enters interactive input mode, as described in Chapter 3, "INTERACTIVE" on page 3-374. In this case, the input statements in the command input file must be entered from your terminal.

Execution on IBM SP2 Systems with More than One Processor

When you execute Medici on the IBM SP2 for more than one processor, the number of processors (<n>) must follow the program command

```
medici n
```

The program responds as described above. The limitations of the IBM SP2 version of Medici are listed in Appendix F.

Syntax Errors

The input statements in the command input file are printed on the standard output (see Standard Out File—<*base*>.*out* on page 1-9) with assigned sequential line numbers. Each statement is checked for syntax errors. Syntax error messages are displayed on your terminal and on the standard output. Syntax error messages contain the following:

- The error number
- The line number of the statement responsible for the error
- The invalid portion of the statement text
- A description of the error

Execution Errors and Warnings

If no syntax errors are found, the input statements are then processed to check the validity of parameter combinations and values. The validity of a statement is also considered in the context of the preceding statements. Errors and warnings are indicated by messages displayed on your terminal and on the standard output.

Execution errors and warnings contain the following:

- The error number
- The line number of the statement responsible for the error
- A description of the error

Error and warning messages associated with file input and output include available system I/O error numbers.

The occurrence of execution errors terminates program execution after it is determined that no further reliable error checking can be performed.

Note:

Warnings are not fatal and serve only to indicate possible problems that you may choose to correct. Warning messages normally indicate the corrective action taken automatically by the program.

Program Output

Medici generates both printed and graphical output that describes the simulation results. All output is made available to the user before termination of the program, as described in "Output Files" on page 1-9.

Printed Output

The following printed output can be obtained:

- Terminal voltages and currents for each bias and/or time point
- Terminal capacitances, conductances, and admittances as the result of an AC small-signal analysis
- Quantities obtained from mathematical expressions involving terminal data and internal device variables
- Structure information at each node of the simulation mesh, such as node number, coordinates, impurity concentration, interface charge, and electrode connections
- Mesh element information, such as element number, vertex nodes, material, coupling coefficients between nodes of the element, and the area of each node assigned to the element
- Material, mobility, contact, and model parameters, including values for concentration-dependent mobilities and lifetimes at each node
- Potential, carrier concentrations, current densities, recombination and generation rates, and electric field at each node in the simulation region

Graphical Output

The following graphical output can be obtained:

- One-dimensional plots of terminal data

  This can include DC characteristics, such as applied voltage, contact voltage, terminal current, and time (in the case of transient simulations). It can also include AC quantities, such as capacitance, conductance, admittance, and frequency (if a frequency analysis is performed), and user-defined data.
- One-dimensional plots of quantities along arbitrary straight line paths through the structure

  Some of the quantities that can be plotted include potential, carrier quasi-Fermi potential, electric field, carrier concentration, impurity concentration, recombination and generation rates, and current density.
- Two-dimensional structure plots that can include the grid, boundaries, electrode and junction locations, and depletion line edges
- Two-dimensional contour plots of quantities, such as potential, carrier quasi-Fermi potential, electric field, carrier concentration, impurity concentration, recombination and generation rates, current density, and flowlines
- Two-dimensional vector plots of current density and electric field
- Three-dimensional projection plots of quantities, such as potential, carrier quasi-Fermi potential, electric field, carrier concentration, impurity concen-

tration, recombination and generation rates, and current density, against location in the structure

File Specifications

This section contains discussions of the various file specifications used by Medici, their configuration, locations, and how they might be used. This section includes the following:

- Output file identifiers
- Command input file indefinites
- Initially assigned names
- Environmental variables
- Library directory

Output File Identifiers—*<base>*

The names of standard output files are based on a common value referred to as <*base*>. The default value of <*base*> is set to the value of <*root*> (described below), which depends on the identifier for the command input file. This allows the names of output files to be uniquely associated with the name of the command input file. Multiple copies of Medici may be executed simultaneously, using different command input files, in a single directory without encountering naming conflicts among the output files.

The default value of <*base*> may be overridden by setting the *MBASE* environment variable to an alternate value before executing Medici. The value of <*base*> is available in the command input file through the *MBASE* initially assigned name. Environment variables are discussed in "Environment Variables" on page 1-7, and initially assigned names are discussed in "Initially Assigned Names" on page 1-6.

Command Input File Identifier—*<head>* and *<root>*

The identifier for the command input file is represented by the following components:

<***head***>	The directory containing the file.
<***root***>	The portion of the file name before the last period.
<ext>	The portion of the file name after the last period.

The format of the file identifier is:

<*head*>/<*root*>.<*ext*>

The following special cases can occur:

- If the file identifier is blank, Medici has entered interactive input mode, <*head*> is defined as "." and <*root*> is defined as "*md*".
- If the file identifier does not include a directory, <*head*> is defined as ".".
- If the file name does not include a period, <*root*> is defined as the entire file name.

The default values of <*head*> and <*root*> may be overridden by setting the *MDINH* and *MDINR* environment variables (see "Environment Variables" on page 1-7), to alternate values before executing Medici. The values of <*head*> and <*root*> are available in the command input file through the *MDINH* and *MDINR* initially assigned names (see "Initially Assigned Names" on page 1-6).

Initially Assigned Names

At the beginning of execution, Medici establishes values for a predetermined set of assigned names (see the description of the **`ASSIGN`** statement in Chapter 3, page 3-396). These are referred to as "initially assigned names." These names are associated with the standard file identifiers used by Medici. The following initially assigned names are defined:

Initially Assigned Names	Description	Manual Section
MDINH	<*head*>—directory containing the command input file.	See Command Input File Identifier—<*head*> and <*root*> on page 1-5.
MDINR	<*root*>—root of the command input file name.	See Command Input File Identifier—<*head*> and <*root*> on page 1-5.
MDBASE	<*base*>—base for standard output file names.	See Output File Identifiers—<*base*> on page 1-5.
MDLIB	Location of the library directory.	See Library Directory on page 1-7.
MDPATH	Path file identifier.	See Path File—*mdpath* on page 1-13.
MDAUTH	Authorization file identifier.	See Authorization File—*mdauth* on page 1-14.
MDFKY0	Formatted key file identifier.	See Formatted Key File--*mdfky0* on page 1-14.
MDUKY0	Unformatted key file identifier.	See Unformatted Key File—*mduky0* on page 1-15.
MDERR0	Error message file identifier.	See Error Message File—*mderr0* on page 1-15.
MDPDEV	Plot device definition file identifier.	See Plot Device Definition File—*mdpdev* on page 1-14.
MDFKYI	Circuit analysis formatted key file identifier.	See Circuit Analysis Formatted Key File—*mdfky1* on page 1-15.
MDOUT	Standard output file identifier.	See Standard Output File—<*base*>.*out* on page 1-9.

Initially Assigned Names	Description	Manual Section
MDINF	Informational output file identifier.	See Information Output File--<*base*>.*inf* on page 1-9.
MDDIA	Diagnostic output file identifier.	See Diagnostic Output File—<*base*>.*dia* on page 1-10.
MDPRS	Parsing scratch file identifier.	See Temporary Scratch Files on page 1-15.
MDDPLT	Formatted plot output file identifier.	See Formatted Plot Output Files on page 1-10.
MDBPLT	Binary plot output file identifier.	See Binary Plot Output Files on page 1-11.
MDCPU	Execution time file identifier.	See Execution Time Files on page 1-13.
MDIVL	I-V log file identifier.	See Log Files on page 1-12.
MDSAV	Saved solution file identifier.	See Saved Solution Files on page 1-12.
MDTTN	Triangle tree storage file identifier.	See Temporary Scratch Files on page 1-15.
MDCONT	Temporary solution file identifier.	See Temporary Scratch Files on page 1-15.

The default value of each initially assigned name may be overridden by setting the environment variable (see "Environment Variables" on page 1-7) of the same name to an alternate value before executing Medici.

Environment Variables

Environment variables can be used to override the default values for the library directory location (see "Library Directory" on page 1-7), the standard file identifiers, and the graphics output device. Graphic devices are discussed in Appendix B.

A unique environment variable is associated with each library file and each standard output file. These environment variables are described in the remainder of this chapter in the descriptions of the file identifiers that they control.

Library Directory

The library directory contains files that are required to execute Medici (see "Miscellaneous Files" on page 1-13). The files are: *mdpath*, *mdpdev*, *mdauth*, *mdfky0*, *mdfky1*, and *mderr0*. The most important file is *mdpath*, which defines the default values for the library directory location and the standard file identifiers.

The default location of the library directory is defined as the current directory. It can be overridden by setting the *MDLIB* environment variable to an alternate directory location before executing Medici.

Input Files

Medici requires several types of input files. You must always supply a command input file. All other input files are optional.

Command Input File

The command input file contains statements that direct the Medici simulation. It is a text file that is created and modified using any text editor. Chapter 3 contains a detailed description of the valid input statements and their formats.

Statement Overview

The initial structure definition consists of **MESH**, **X.MESH**, **Y.MESH**, **ELIMINATE**, **SPREAD**, **BOUNDARY**, **TSUPREM4**, **REGION**, **ELECTRODE**, and **PROFILE** statements. These statements define the structure and simulation grid. **REGRID** statements may be used to refine the simulation grid.

INTERFACE statements may be included to specify interface charge, traps, or recombination velocities. Use **CONTACT** statements to include special boundary conditions at electrodes and **MATERIAL** statements to alter the material characteristics of the structure. **MOBILITY** statements specify parameters associated with the various mobility models. **MODELS** statements specify physical models used during the simulation.

SYMBOLIC statements are used to select the solution methods in the simulation. **METHOD** statements are used to choose special techniques for use with the selected solution method. The **SOLVE** statement is used to select bias conditions and analysis type, which can be either steady state, transient, or AC small-signal.

The **PLOT.3D** statement is used to initiate a three-dimensional plot sequence. Other statements that can form part of the sequence are **3D.SURFACE**, **TITLE**, and **COMMENT**.

The **PLOT.2D** statement is used to initiate a two-dimensional plot sequence. The two-dimensional plot sequence can consist of **CONTOUR**, **VECTOR**, **E.LINE**, **LABEL**, **TITLE**, and **COMMENT** statements.

The **PLOT.1D** statement is used to initiate a one-dimensional plot. The **E.LINE**, **LABEL**, **TITLE**, and **COMMENT** statements can be used in conjunction with one-dimensional plots.

Two-Dimensional Process Files

Two-dimensional process files can contain either two-dimensional electrically active impurity distributions or boundary information describing device structures, or both.

Two-dimensional process files containing impurity distributions can be specified as input on the **PROFILE** statement to define impurity distributions in Medici. These files can be generated by TSUPREM-4, or they may be formatted text files. In addition, TSUPREM-4 files can be specified as input on the **MESH** statement in order to define both a Medici grid and its doping distribution.

Two-dimensional process files containing boundary information can be specified as input on the **BOUNDARY** statement. They are used to derive grids conforming closely to the boundaries. These files can be generated by Taurus-Lithography or TSUPREM-4, or they may be formatted text files.

One-Dimensional Process Files

One-dimensional process files contain one-dimensional electrically active impurity distributions. They are specified as input on the **PROFILE** statement to define vertical impurity distributions. One-dimensional process files can be generated by TMA SUPREM-3, or they may be formatted text (ASCII) files.

Output Files

Medici produces a variety of printed and graphical output and data files. Messages and output indicating the progress of the program are displayed on your terminal. These files are described in this section.

Standard Output File—*<base>.out*

Standard output is placed in the *<base>.out* file. It consists of a listing of the simulation input statements, error and warning messages, and printed output produced during a Medici session. Each input statement in the listing is preceded by a line number. The *<base>.out* file is formatted as standard FORTRAN list output. It should be examined after Medici completes a simulation.

The default file identifier for the file *<base>.out* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDOUT* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *<base>.out* is available within the command input file through the *MDOUT* initially assigned name.

Informational Output File—*<base>.inf*

Informational output is placed in the *<base>.inf* file and consists of supplementary information. It is formatted as standard FORTRAN list output and is normally not of interest to you.

The default file identifier for the file *<base>.inf* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting

the *MDINF* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*inf* is available within the command input file through the *MDINF* initially assigned name.

Diagnostic Output File—*<base>.dia*

Diagnostic output is placed in the <*base*>.*dia* file and consists of information that is used during the diagnosis of program errors. It is formatted as standard FORTRAN list output and is useful only if you have a detailed knowledge of the internal operation of Medici.

The default file identifier for the file <*base*>.*dia* is defined by the path file *mdpath* (see Path file—*mdpath* on page 1-13). This default may be overridden by setting the *MDDIA* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*dia* is available within the command input file through the *MDDIA* initially assigned name.

Graphical Output

Graphical output is sent to the graphics output device specified by the **DEVICE** parameter on **PLOT.1D**, **PLOT.2D**, and **PLOT.3D** statements. This is usually your terminal. Valid names for the graphics output device are defined by the *mdpdev* plot device definition file (see Appendix B).

Device drivers are available for a variety of devices, including graphics terminals, pen plotters, and graphics software libraries. In addition, custom drivers may be developed and installed at your site.

Formatted Plot Output Files

Formatted plot output files are FORTRAN formatted files containing the character sequences that control the graphics device. These files may be output to the graphics device to reproduce the graphical output.

Output is sent to the file specified by the **PLOT.OUT** parameter on **PLOT.1D**, **PLOT.2D**, and **PLOT.3D** graphics statements, if this parameter is specified. Otherwise, if the **DF** entry is "T" in the *mdpdev* plot device definition file, output is sent to the <*base*>.*dplt* formatted plot output file. Output to these files is only available for the direct device drivers, as discussed in the description of the graphics statements in Chapter 3 and Appendix B.

The default file identifier for the file <*base*>.*dplt* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDDPLT* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*dplt* is available within the command input file through the *MDDPLT* initially assigned name.

Binary Plot Output Files

Binary plot output files are unformatted (binary) files with each line containing the arguments *X* (real), *Y* (real), and *IPEN* (integer) for one call to the driver subroutine. The arguments *X* and *Y* are in units of centimeters. This output is *not* in the same format as the output sent to the output device, and is only useful as input to graphics post-processors.

Output is sent to the file specified by the **PLOT.BIN** parameter on **PLOT.1D**, **PLOT.2D**, and **PLOT.3D** statements, if this parameter is specified. Otherwise, if the **BF** entry is "T" in the *mdpdev* plot device definition file, output is sent to the *<base>.bplt* binary plot output file.

The default file identifier for the file *<base>.bplt* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDBPLT* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *<base>.bplt* is available within the command input file through the *MDBPLT* initially assigned name.

Mesh Files

Mesh files contain coordinates of simulation mesh nodes region information and doping information at each node. They can be generated and given user-specified names by using the **MESH** and **REGRID** statements. Mesh files can be used to define the simulation mesh in subsequent simulations through the **MESH** statement. By default, mesh files are created in binary format. They may also be created as formatted files.

Profile Files

Profile files contain the original specification of the impurity profiles for a Medici simulation. They can be generated and given user-specified names by using the **PROFILE** statement. Profile files are in binary format and are used to define the impurity profiles during regridding operations.

Boundary Files

Boundary files contain descriptions of the material interfaces of grids created by the **BOUNDARY** statement. Since they are formatted files, you can alter them so they can be reread as input to the **BOUNDARY** statement.

Solution Files

Solution files contain the results of solutions to the following:

- Poisson's equation
- Electron and hole current-continuity equations (if used)

- Electron and hole energy balance equations (if used)
- Lattice heat equation (if used)

Solution files can be generated and given user-specified identifiers by using the **SOLVE** statement. Use solution files to initialize the simulation solutions through the **LOAD** statement. By default, solution files are created in binary format, but they can also be created as formatted files.

TIF Files

Technology Interchange Format (TIF) files are formatted files that can contain both mesh and solution information. TIF files provide a simple method for exchanging information between Medici and other programs such as TSUPREM-4 and Michelangelo. TIF files can be generated and given user-specified identifiers by using the **SOLVE** and **SAVE** statements. You can also use TIF files to initialize a simulation through the **MESH** statement.

Note:

Technology Interchange Format (TIF) is a proprietary but publicly open Avant! TCAD format. It should not be confused with the graphics format, Tagged Image File Format (TIF or TIFF).

Saved Solution Files

Saved solution files contain the two most recent completed solutions. These files are generated by specifying the **SAVE.SOL** parameter on the **OPTION** statement. The files can be given user-specified identifiers by using the **SOL.FILE** parameter on the **OPTION** statement. Saved solution files are in binary format and can be used to initialize the simulation solutions through the **LOAD** statement.

If the user-specified identifier is not specified for the saved solution files, a default of <*base*>.*sav* is used.

The default file identifier for the file <*base*>.*sav* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDSAV* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*sav* is available within the command input file through the *MDSAV* initially assigned name.

Log Files

Log files contain the terminal characteristics (I-V, AC, or arbitrary data) for all solutions performed after opening the log file. These files can be generated and given user-specified identifiers by using the **LOG** statement. Log files have the format of standard list output and can be used as input during **PLOT.1D** statements.

If the user-specified identifier is not provided for the log files, a default of <*base*>.*ivl* is used.

The default file identifier for the file <*base*>.*ivl* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDIVL* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*ivl* is available within the command input file through the *MDIVL* initially assigned name.

Execution Time Files

Execution time files contain detailed execution-time information for the simulation. These files are generated by specifying the **CPU.STAT** parameter on the **OPTION** statement. The files can be given user-specified identifiers by using the **CPU.FILE** parameter on the **OPTION** statement. Execution time files have the format of standard list output and are usually not generated or of interest to the user.

If an user-specified identifier is not provided for the execution time file, a default of <*base*>.*cpu* is used.

The default file identifier for the file <*base*>.*cpu* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDCPU* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*cpu* is available within the command input file through the *MDCPU* initially assigned name.

Miscellaneous Files

Medici uses several miscellaneous files which are generally not of interest to the user.

CAUTION

Medici **cannot run if these files are missing or inaccessible.**

Path File—*mdpath*

mdpath defines the following:

- The default location of the library directory, and the default file identifiers for files in the library directory such as, *mdpdev*, *mdauth*, *mdfky0*, *mduky0*, and *mderr0*.
- The default file identifiers for standard output files, such as <*base*>.*out*, <*base*>.*inf*, <*base*>.*dia*, <*base*>.*prs*, <*base*>.*dplt*, and <*base*>.*bplt*.

The path file is a text file and can be modified with any text editor.

The default file identifier for *mdpath* is defined in Medici to be located in the library directory. This default may be overridden by setting the *MDPATH* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *mdpath* is available within the command file through the *MDPATH* initially assigned name.

Plot Device Definition File—*mdpdev*

The *mdpdev* file contains information that controls the use of the driver subroutines that send graphical output to a graphics device or graphics software library. It is a text file and you can modify it with any text editor. This file is described in detail in Appendix B.

The default file identifier for *mdpdev* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDPDEV* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *mdpdev* is available within the command file through the *MDPDEV* initially assigned name.

Authorization File—*mdauth*

The *mdauth* file contains authorization values that enable the execution of Medici. If this file contains invalid authorization values, error 1074 is generated. You have to contact *Avant!* TCAD Business Unit for assistance in correcting this problem. This file is a text file.

The default file identifier for *mdauth* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDAUTH* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *mdauth* is available within the command file through the *MDAUTH* initially assigned name.

Note:

This file is only applicable to the Cray system.

Formatted Key File—*mdfky0*

The *mdfky0* file defines the statement names, parameter names, and some of the default values used by Medici. It is used to check the syntax of the command input file. This file is a text file.

The default file identifier for *mdfky0* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDFKY0* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *mdfky0* is available within the command file through the *MDFKY0* initially assigned name.

Unformatted Key File—*mduky0*

The *mduky0* file contains the same information as the *mdfky0* formatted key file, but it is in binary format. Although this file does not originally exist, it can be used much more efficiently than *mdfky0* to check the syntax of the command input file.

When Medici is executed, it attempts to open the *mduky0* file. If *mduky0* does not exist or is inaccessible, the data in *pmdfky0* is used to generate *mduky0* so it can perform the syntax check. Once *mduky0* has been created, *mdfky0* is no longer needed. If it is necessary to change the statement names, parameter names, or default values by modifying *mdfky0*, *mduky0* should be deleted and the updated *mdfky0* file made available for the next execution of Medici.

The default file identifier for *mduky0* is defined by the path file *mdpath* (see Path File--*mdpath* on page 1-13). This default may be overridden by setting the *MDUKY0* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *mduky0* is available within the command file through the *MDUKY0* initially assigned name.

Error Message File—*mderr0*

The *mderr0* file defines the error messages used by Medici to report syntax and other execution errors. It also defines the information printed by the **HELP** statement. It should always be present and available to Medici. The *mderr0* file is a text file.

The default file identifier for *mderr0* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDERR0* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *mderr0* is available within the command file through the *MDERR0* initially assigned name.

Circuit Analysis Formatted Key File—*mdfky1*

The *mdfky1* file defines the statement names, parameter names, and some of the default values used by the Circuit Analysis AAM in Medici. It is used to check the syntax of the command input file. The *mdfky1*file is a text file.

The default file identifier for *mdfky1* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDFKY1* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file *mdfky1* is available within the command file through the *MDFKY1* initially assigned name.

Temporary Scratch Files

Temporary scratch files are created and deleted as needed by Medici. These files do not need to be present before or after Medici is executed, and are usually deleted when execution terminates. If the program terminates abnormally, these

files may not be deleted. If they are not deleted, you may delete them. The temporary scratch files created by Medici are described below.

<base>.prs

<*base*>.*prs* is used in the parsing of input statements. The default file identifier for <*base*>.*prs* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDPRS* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*prs* is available within the command file through the *MDPRs* initially assigned name.

<base>.ttn

<*base*>.*ttn* is used to store the history of regridding operations for **REGRID** statements that do not specify an output mesh file. This file is used by the next **REGRID** statement to initialize the history of previous regridding operations. The default file identifier for the file <*base*>.*ttn* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDTTN* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*ttn* is available within the command input file through the *MDTTN* initially assigned name.

<base>.cont

<*base*>.*cont* is used to store the potential and carrier concentrations during continuation of a solution. The default file identifier for <*base*>.*cont* is defined by the path file *mdpath* (see Path File—*mdpath* on page 1-13). This default may be overridden by setting the *MDCONT* environment variable to an alternate file identifier before executing Medici. The file identifier used for the file <*base*>.*cont* is available within the command file through the *MDCONT* initially assigned name.

Medici Description

Introduction

This chapter describes the Medici program for the analysis of electrical device operation.[1] It contains, in order of presentation, discussions and/or descriptions of the following:

- Physical equations used by the program to describe semiconductor device behavior.
- Mobility modeling capabilities, boundary conditions, numerical methods, and simulation grid.
- Analysis capabilities including, transient, AC small-signal, impact ionization, gate current, ionization integrals, and band-to-band tunneling.
- Simulation of deep submicron devices made possible by Medici's ability to include energy balance equations self-consistently within the system of device equations. (See "Energy Balance Equations" on page 2-112.)
- Modification on a region by region basis of all accessible model and material parameters. (See "Regional Specification of Semiconductor Parameters" on page 2-123.)
- Advanced Application Modules (AAM) available for Medici. (See "Programmable Device Advanced Application Module" on page 2-123 through "Anisotropic Material Advanced Application Module" on page 2-159.)

Equations and discussions include the statement parameters documented in Chapter 3.

Definitions and units for all symbols used in this chapter not documented in Chapter 3 are found on page 2-163. The section, "References" on page 2-175, lists the references for Chapter 2.

1. Much of the material in this chapter is reproduced, with permission, from the Stanford University technical reports "PISCES II: Poisson and Continuity Equation Solver", by Mark R. Pinto, Conor S. Rafferty, and Robert W. Dutton, Copyright 1984 The Board of Trustees of the Leland Stanford Junior University and "PISCES-IIB Supplementary Report", by Mark R. Pinto, Conor S. Rafferty, Hal R. Yeager, and Robert W. Dutton, Copyright 1985 The Board of Trustees of the Leland Stanford Junior University. We thank them for this courtesy.

Physical Description

This section describes the following:

- Fundamental equations solved by Medici
- Recombination mechanisms and carrier and impurity statistics
- Temperature dependencies of parameters and models

The actual temperature, T, at which the simulation is performed can be specified with the **TEMPERAT** parameter on the **MODELS** statement.

Basic Equations

The primary function of Medici is to solve the three partial differential equations, Equations 2-1, 2-2, and 2-3, self-consistently for the electrostatic potential, ψ, and for the electron and hole concentrations, n and p, respectively.

Poisson's Equation

The electrical behavior of semiconductor devices is governed by Poisson's equation:

$$\varepsilon\nabla^2\psi = -q(p - n + N_D^+ - N_A^-) - \rho_S \qquad \text{Equation 2-1}$$

Continuity Equations

Continuity equations for electrons and holes also govern electrical behavior:

$$\frac{\partial n}{\partial t} = \frac{1}{q}\vec{\nabla}\cdot\vec{J}_n - U_n = F_n(\psi,n,p) \qquad \text{Equation 2-2}$$

$$\frac{\partial p}{\partial t} = \frac{-1}{q}\vec{\nabla}\cdot\vec{J}_p - U_p = F_p(\psi,n,p) \qquad \text{Equation 2-3}$$

Throughout Medici, ψ is always defined as the intrinsic Fermi potential. That is, $\psi = \psi_{intrinsic}$. N_D^+ and N_A^- are the ionized impurity concentrations and ρ_S is a surface charge density that may be present due to fixed charge in insulating materials or charged interface states (see "Interface Charge and Traps" on page 2-53).

Boltzmann Transport Theory

From Boltzmann transport theory, $\vec{J}_n$ and $\vec{J}_p$ in Equations 2-2 and 2-3 can be written as functions of the carrier concentrations and the quasi-Fermi potentials for electrons and holes, ϕ_n and ϕ_p.

$$\vec{J}_n = -q\mu_n n\vec{\nabla}\phi_n \qquad \text{Equation 2-4}$$

$$\vec{J}_p = -q\mu_p p\vec{\nabla}\phi_p \qquad \text{Equation 2-5}$$

Alternatively, $\vec{J}_n$ and $\vec{J}_p$ can be written as functions of ψ, n, and p, consisting of drift and diffusion components:

$$\vec{J}_n = q\mu_n \vec{E}_n n + qD_n \vec{\nabla} n \quad \text{Equation 2-6}$$

$$\vec{J}_p = q\mu_p \vec{E}_p p - qD_p \vec{\nabla} p \quad \text{Equation 2-7}$$

where μ_n and μ_p are the electron and hole mobilities and D_n and D_p are the electron and hole diffusivities, neglecting the effects of bandgap narrowing and assuming Boltzmann carrier statistics (see "Boltzmann Statistics" on page 2-7),

$$\vec{E}_n = \vec{E}_p = \vec{E} = -\vec{\nabla}\psi \quad \text{Equation 2-8}$$

SRH, Auger, and Direct Recombination

U_n and U_p in Equations 2-2 and 2-3 represent net electron and hole recombination respectively. Currently Medici supports Shockley-Read-Hall, Auger, and direct recombination (also known as band-to-band or optical recombination).

That is,

$$U = U_n = U_p = U_{SRH} + U_{Auger} + U_{dir} \quad \text{Equation 2-9}$$

where

$$U_{SRH} = \frac{pn - n_{ie}^2}{\tau_p\left[n + n_{ie}\exp\left(\frac{\mathbf{ETRAP}}{kT}\right)\right] + \tau_n\left[p + n_{ie}\exp\left(\frac{-\mathbf{ETRAP}}{kT}\right)\right]} \quad \text{Equation 2-10}$$

$$U_{dir} = \mathbf{C.DIRECT}(np - n_{ie}^2) \quad \text{Equation 2-11}$$

$$U_{Auger} = \mathbf{AUGN}(pn^2 - nn_{ie}^2) + \mathbf{AUGP}(np^2 - pn_{ie}^2) \quad \text{Equation 2-12}$$

In the above, n_{ie} is the effective intrinsic concentration and τ_n and τ_p are the electron and hole lifetimes, which may be concentration dependent (see "Concentration Dependent Lifetimes" on page 2-4). The parameter **ETRAP** represents the difference between the trap energy level E_t and the intrinsic Fermi energy E_i (i.e., **ETRAP** = $E_t - E_i$) and **AUGN** and **AUGP** are specified constants. The **MATERIAL** statement can be used to modify the default values of **ETRAP**, **AUGN**, **AUGP**, and **C.DIRECT**.

Surface Recombination

In addition to the recombination mechanisms described in the previous section, Medici also includes an additional recombination component at specific insulator-semiconductor interfaces. This recombination mechanism can be described by a surface recombination velocity as described in Reference [1].

Surface recombination velocities for electrons and holes can be specified for any interface using the **INTERFACE** statement. Note that the **INTERFACE** statement can also be used to include fixed charge densities, **QF**, for the interface and interface traps (see "Interface Charge and Traps" on page 2-53).

For each node on the interface so specified, an effective SRH lifetime for each carrier, τ_n^{eff} and τ_p^{eff}, is computed based on the given recombination velocities, **S.N** and **S.P**:

$$\frac{1}{\tau_n^{eff}(i)} = \frac{\mathbf{S.N}\ d_i}{A_i} + \frac{1}{\tau_n(i)} \qquad \text{Equation 2-13}$$

$$\frac{1}{\tau_p^{eff}(i)} = \frac{\mathbf{S.P}\ d_i}{A_i} + \frac{1}{\tau_p(i)} \qquad \text{Equation 2-14}$$

where $\tau_n(i)$ and $\tau_p(i)$ are the regular SRH lifetimes at node i (possibly concentration dependent), A_i is the semiconductor area associated with node i, and d_i is the length of the interface associated with node i.

Each interface, or portions of a particular interface, can be defined separately with arbitrary recombination velocities at each interface.

Concentration Dependent Lifetimes

Electron and hole lifetimes used in Medici may be concentration dependent as follows:

$$\frac{\mathbf{TAUN0}}{\tau_n(x,y)} = \mathbf{AN} + \mathbf{BN}\left(\frac{N_{total}(x,y)}{\mathbf{NSRHN}}\right) + \mathbf{CN}\left(\frac{N_{total}(x,y)}{\mathbf{NSRHN}}\right)^{\mathbf{EN}} \qquad \text{Equation 2-15}$$

$$\frac{\mathbf{TAUP0}}{\tau_p(x,y)} = \mathbf{AP} + \mathbf{BP}\left(\frac{N_{total}(x,y)}{\mathbf{NSRHP}}\right) + \mathbf{CP}\left(\frac{N_{total}(x,y)}{\mathbf{NSRHP}}\right)^{\mathbf{EP}} \qquad \text{Equation 2-16}$$

where N_{total} is the local total impurity concentration.

The parameters **NSRHN**, **NSRHP**, **TAUN0**, **TAUP0**, **AN**, **AP**, **BN**, **BP**, **CN**, **CP**, **EN**, and **EP** are user accessible constants which can be changed from their default values on the **MATERIAL** statement (see Reference [2]). The default values of these parameters are such that the above expressions reduce to

$$\tau_n(x,y) = \frac{\mathbf{TAUN0}}{1 + N_{total}(x,y)/\mathbf{NSRHN}} \qquad \text{Equation 2-17}$$

$$\tau_p(x,y) = \frac{\mathbf{TAUP0}}{1 + N_{total}(x,y)/\mathbf{NSRHP}} \qquad \text{Equation 2-18}$$

Lattice Temperature-Dependent Lifetimes

Electron and hole lifetimes used in Medici may be lattice temperature-dependent as follows:

$$\tau_n(x,y,T) = \tau_n(x,y)\left(\frac{T}{300}\right)^{\mathbf{EXN.TAU}} \quad \text{Equation 2-19}$$

$$\tau_p(x,y,T) = \tau_p(x,y)\left(\frac{T}{300}\right)^{\mathbf{EXP.TAU}} \quad \text{Equation 2-20}$$

The default values for **EXN.TAU** and **EXP.TAU** are zero, which disables the lattice temperature-dependent lifetimes. The models can be activated by specifying nonzero values for these parameters on the **MATERIAL** statement.

Recombination Including Tunneling

A generalization of the Shockley-Read-Hall recombination model that includes tunneling in the presence of strong electric fields (Reference [3]) is available by specifying **R.TUNNEL** on the **MODELS** statement. It extends U_{SRH} to

$$U_{RTUN} = U^*_{SRH} + U_{btbt} \quad \text{Equation 2-21}$$

where U^*_{SRH} is the modified SRH recombination using field-dependent lifetimes, and U_{btbt} represents recombination due to band-to-band tunneling. The field dependent lifetimes for electrons are

$$\tau_n = \frac{\tau^0_n}{1+\Gamma_n} \quad \text{Equation 2-22}$$

where τ^0_n is electron lifetime for $E = 0$ and Γ_n is the field-enhancement factor due to trap-assisted tunneling and is given by

$$\Gamma_n = \frac{\Delta E_n}{kT}\int_0^1 \exp\left[\frac{\Delta E_n}{kT}\ u - K_n\ u^{3/2}\right]du \quad \text{Equation 2-23}$$

where K_n is given by

$$K_n = \frac{4}{3}\frac{\sqrt{2(\mathbf{M.RTUN}\cdot m_0)\Delta E_n^3}}{q\frac{h}{2\pi}E} \quad \text{Equation 2-24}$$

The expression for Γ_n is approximated by analytic expressions for low field and high field as described in Reference [3], except that here the low field expression has been enhanced to provide a smooth transition from low field to high field:

Equation 2-25

$$\Gamma_n = \sqrt{\pi}\ \frac{E}{E_{trap}}\ \exp\left[\frac{1}{3}\left(\frac{E}{E_{trap}}\right)^2\right]\left\{2 - \mathrm{erfc}\left[\frac{1}{2}\left(\frac{E_{trap}}{E} - \frac{E}{E_{trap}}\cdot\frac{\Delta E_n}{kT}\right)\right]\right\}$$

when $\frac{E}{E_{trap}} \le \sqrt{\frac{\Delta E_n}{kT}}$ and

Equation 2-26

$$\Gamma_n = \sqrt{\pi\frac{E}{E_{trap}}}\left(\frac{\Delta E_n}{kT}\right)^{1/4}\exp\left[-\frac{\Delta E_n}{kT} + \frac{E}{E_{trap}}\left(\frac{\Delta E_n}{kT}\right)^{1/2} + \frac{1}{3}\frac{E_{trap}}{E}\left(\frac{\Delta E_n}{kT}\right)^{3/2}\right] \times \mathrm{erfc}\left[\left(\frac{E}{E_{trap}}\right)^{1/2}\left(\frac{\Delta E_n}{kT}\right)^{1/4} - \left(\frac{E_{trap}}{E}\right)^{1/2}\left(\frac{\Delta E_n}{kT}\right)^{3/4}\right]$$

elsewhere. Expressions for E_{trap} and $\Delta E_n/(kT)$ are given by

$$E_{trap} = \frac{\sqrt{8(\mathbf{M.RTUN}\cdot m_0)(kT)^3}}{q\frac{h}{2\pi}}$$

Equation 2-27

$$\frac{\Delta E_n}{kT} = \begin{cases} 0 & n > n_{ie}\exp(E_g/2kT) \\ \frac{E_g}{2kT} - \ln\frac{n}{n_{ie}} & n_t \le n \le n_{ie}\exp(E_g/2kT) \\ \frac{E_g}{2kT} - \ln\frac{n_t}{n_{ie}} & n < n_t \end{cases}$$

Equation 2-28

$$n_t = n_{ie}\exp\left(\frac{\mathbf{ETRAP}}{kT}\right)$$

Equation 2-29

Similar expressions exist for τ_p, Γ_p, $\Delta E_p/kT$ and p_t is defined by

$$p_t = n_{ie}\exp\left(-\frac{\mathbf{ETRAP}}{kT}\right)$$

Equation 2-30

The recombination due to band-to-band tunneling is expressed by

$$U_{btbt} = -\mathbf{B.RTUN}\ E^{\mathbf{S.RTUN}}\ D\exp\left(-\frac{E_{btbt}}{E}\right)$$

Equation 2-31

The factor D is expressed by (Reference [3])

Equation 2-32

$$D = \begin{cases} 0 & \vec{\nabla}\phi_{n,p} \cdot \vec{E} < 0 \\ \dfrac{n_{ie}^2 - pn}{(n + n_{ie})(p + n_{ie})} & \vec{\nabla}\phi_{n,p} \cdot \vec{E} > 0, \quad J_{n,p} < 10^{-3}\ qn_{ie}v_{sat} \\ 1 & \vec{\nabla}\phi_{n,p} \cdot \vec{E} > 0, \quad J_{n,p} > 10^{-3}\ qn_{ie}v_{sat} \end{cases}$$

where v_{sat} is the saturated drift velocity. E_{btbt} is given by

$$E_{btbt} = \mathbf{E.RTUN}\left(\frac{E_g(T)}{E_g(T = 300)}\right)^{3/2}$$ Equation 2-33

All parameters associated with the **R.TUNNEL** model are accessible through the **MATERIAL** statement.

Boltzmann Statistics

The electron and hole concentrations in semiconductors are defined by Fermi-Dirac distributions and a parabolic density of states. When these are integrated, they yield:

$$n = N_C \mathbf{F}_{1/2}(\eta_n)$$ Equation 2-34

$$p = N_V \mathbf{F}_{1/2}(\eta_p)$$ Equation 2-35

where

$$\eta_n = \frac{E_{Fn} - E_C}{kT}$$ Equation 2-36

$$\eta_p = \frac{E_V - E_{Fp}}{kT}$$ Equation 2-37

In Equations 2-34, 2-35, 2-36, and 2-37

- N_C and N_V are the effective density of states in the conduction and valence bands.
- E_C and E_V are the conduction and valence band energies
- E_{Fn} and E_{Fp} are the electron and hole Fermi energies. For example, $E_{Fn} = -q\phi_n$ and $E_{Fp} = -q\phi_p$.

The Fermi-Dirac integral of order one-half is defined as:

$$\mathbf{F}_{1/2}(\eta_S) = \frac{2}{\sqrt{\pi}}\int_0^\infty \frac{\eta^{1/2}}{1 + \exp(\eta - \eta_S)} d\eta$$ Equation 2-38

For the operating range of most semiconductor devices, Equations 2-34 and 2-35 can be simplified using Boltzmann statistics

$$n \approx N_C \exp(\eta_n) = n_{ie} \exp\left[\frac{q}{kT}(\psi - \phi_n)\right] \quad \text{Equation 2-39}$$

$$p \approx N_V \exp(\eta_p) = n_{ie} \exp\left[\frac{q}{kT}(\phi_p - \psi)\right] \quad \text{Equation 2-40}$$

where, neglecting bandgap narrowing, the intrinsic carrier concentration is

$$n_{ie}(T) = n_i(T) = \sqrt{N_C N_V}\, e^{-E_g/2\mathrm{kT}} \quad \text{Equation 2-41}$$

Where E_g is the bandgap energy of the semiconductor, that is, $E_g = E_C - E_V$.

Bandgap and Effective Density of States

As described in Reference [4], for the isotropic element semiconductors, the bandgap and effective density of states in the conduction and valence band have temperature dependencies as follows:

$$E_g(T) = E_g(0) - \frac{\mathbf{EGALPH}\ T^2}{T + \mathbf{EGBETA}} \quad \text{Equation 2-42}$$

$$= E_g(300) + \mathbf{EGALPH}\left[\frac{300^2}{300 + \mathbf{EGBETA}} - \frac{T^2}{T + \mathbf{EGBETA}}\right]$$

$$N_C(T) = 2M_C\left(\frac{2\pi m_{de} kT}{h^2}\right)^{3/2} \quad \text{Equation 2-43}$$

$$N_V(T) = 2M_V\left(\frac{2\pi m_{dh} kT}{h^2}\right)^{3/2} \quad \text{Equation 2-44}$$

where for homogenous material, $E_g(300)$ = **EG300**. In Medici, the effective density of states are expressed as follows:

$$N_C(T) = \mathbf{NC300}\left(\frac{T}{300}\right)^{\mathbf{NC.F}} \quad \text{Equation 2-45}$$

$$N_V(T) = \mathbf{NV300}\left(\frac{T}{300}\right)^{\mathbf{NV.F}} \quad \text{Equation 2-46}$$

The parameters **EG300**, **EGALPH**, **EGBETA**, **NC300**, **NC.F**, **NV300**, and **NV.F** can be modified on the **MATERIAL** statement. Note that ψ is related to the bandgap and the effective density of states by the expression

$$-q\psi = E_C - \frac{E_g}{2} - \frac{kT}{2}\ln\left(\frac{N_C}{N_V}\right) \quad \text{Equation 2-47}$$

Bandgap narrowing effects due to heavy doping are included as spatial variations in the intrinsic concentration (Reference [5]),

$$n_{ie}(x,y) = n_i \exp\left\{\frac{\mathbf{V0.BGN}\ q}{2kT}\left[\ln\frac{N_{total}(x,y)}{\mathbf{N0.BGN}} + \sqrt{\left(\ln\frac{N_{total}(x,y)}{\mathbf{N0.BGN}}\right)^2 + \mathbf{CON.BGN}}\right]\right\} \quad \text{Equation 2-48}$$

The parameters **V0.BGN**, **N0.BGN**, and **CON.BGN** can be adjusted from their default values on the **MATERIAL** statement. The spatial dependence of n_{ie} results in an adjustment to the electric field terms in the transport Equations 2-6 and 2-7 obtained by substitution of Equations 2-39 and 2-40 into Equations 2-4 and 2-5,

$$\vec{E}_n = -\vec{\nabla}\left(\psi + \frac{kT}{q}\ln\ n_{ie}\right) \quad \text{Equation 2-49}$$

$$\vec{E}_p = -\vec{\nabla}\left(\psi - \frac{kT}{q}\ln\ n_{ie}\right) \quad \text{Equation 2-50}$$

Stress-Induced Bandgap Change

In addition to temperature and doping-induced variations in the bandgap, Medici can also consider variations in the bandgap due to mechanical stress and strain in silicon regions. Under Boltzmann statistics, the change in the conduction and valence band edges are as follows [97]:

$$\Delta E_C = -kT\ln\left(\sum_{i=1}^{3}\frac{\exp\left(-\frac{\Delta E_{Ci}}{kT}\right)}{3}\right) \quad \text{Equation 2-51}$$

$$\Delta E_V = kT\ln\left(\frac{x_0}{1+x_0}\exp\left(\frac{\Delta E_{Vl}}{kT}\right) + \frac{1}{1+x_0}\exp\left(\frac{\Delta E_{Vh}}{kT}\right)\right) \quad \text{Equation 2-52}$$

where:

- ΔE_{Ci} is the shift in the band edge of the ith ellipsoidal conduction minima
- ΔE_{Vl} and ΔE_{Vh} are the shifts in the band edges for the light and heavy hole maxima, respectively, that make up the valence band.
- x_0 is given by $(\mathbf{ML0}/\mathbf{MH0})^{3/2}$
- The band edge shifts are computed using the deformation potential theory from Bir and Pikus [98]:

$$\Delta E_{Ci} = \mathbf{D.STRESS}\ (\varepsilon_{11} + \varepsilon_{22} + \varepsilon_{33}) + \mathbf{U.STRESS}\ \varepsilon_{ii} \quad \text{Equation 2-53}$$

$$\Delta E_{V(h,l)} = \mathbf{A.STRESS}\ (\varepsilon_{11} + \varepsilon_{22} + \varepsilon_{33}) \pm \sqrt{\xi} \quad \text{Equation 2-54}$$

where

$$\xi = \frac{\mathbf{B.STRESS}^2}{2}\{(\varepsilon_{11} - \varepsilon_{22})^2 + (\varepsilon_{22} - \varepsilon_{33})^2 + (\varepsilon_{33} - \varepsilon_{11})^2\} \quad \text{Equation 2-55}$$
$$+ \mathbf{C.STRESS}^2(\varepsilon_{12}^2 + \varepsilon_{13}^2 + \varepsilon_{23}^2)$$

and ε is the strain tensor in the crystallographic coordinate system. The deformation potential constants, **D.STRESS**, **U.STRESS**, **A.STRESS**, **B.STRESS**, and **C.STRESS**, can be adjusted from their default values on the **MATERIAL** statement.

The spatial variation of E_C and E_V result in an adjustment to the electric field terms that are used in the transport equations. The independent variations of E_C and E_V are handled using a formulation similar to that used for graded heterostructures. Using this formulation, the effective electric fields for electrons and holes can be written as:

$$\vec{E}_n = \frac{\vec{\nabla} E_C}{q} \quad \text{Equation 2-56}$$

$$\vec{E}_p = \frac{\vec{\nabla} E_V}{q} \quad \text{Equation 2-57}$$

Stress-induced changes to the band edges can be selected with the **STRESS** parameter on the **MODELS** statement. The orientation of the y-axis relative to the crystallographic coordinate system should also be specified using the **Y.ORIENT** parameter on the **MODELS** statement. By default, a <100> oriented substrate is assumed. In addition, the nodal values of the 2D stress tensor should be read using the **PROFILE** statement. The stress tensor, σ, is converted to strain for use in Equations 2-53 through 2-55 using the compliance tensor S: $\varepsilon = S\sigma$. In silicon, S has three unique, non-zero, components which are taken as S_{11} = 7.684e-13 cm^2/dyne, S_{12} = -2.139e-13 cm^2/dyne, S_{44} = 1.257e-12 cm^2/dyne. A plane strain condition is assumed.

Fermi-Dirac Statistics

Medici implements Fermi-Dirac statistics in a form quite similar to Boltzmann statistics after Yu (Reference [6]). The form of Equations 2-39 and 2-40 is adjusted by introducing degeneracy factors γ_n and γ_p.

$$\gamma_n = \frac{\mathbf{F}_{1/2}(\eta_n)}{\exp(\eta_n)} \qquad \text{Equation 2-58}$$

$$\gamma_p = \frac{\mathbf{F}_{1/2}(\eta_p)}{\exp(\eta_p)} \qquad \text{Equation 2-59}$$

so Equations 2-39 and 2-40 generalize to

$$n = N_C \gamma_n \exp(\eta_n) \qquad \text{Equation 2-60}$$

$$p = N_V \gamma_p \exp(\eta_p) \qquad \text{Equation 2-61}$$

where $\gamma_n = \gamma_p = 1$ for Boltzmann statistics, but are less than 1 for Fermi-Dirac statistics.

The identification of the exponential term in Equations 2-60 and 2-61 facilitates the standard Scharfetter-Gummel discretization of the continuity equations. The actual code implementation requires both the Fermi-Dirac function $\mathbf{F}_{1/2}(\eta_s)$ and its inverse $\mathbf{F}^{-1}_{1/2}(X_S)$. The inverse function is calculated from the Joyce-Dixon approximation (Reference [7]) given by Equation 2-62:

$$\mathbf{F}^{-1}_{1/2}(X_S) = \eta_S = \ln(X_S) + aX_S + b(X_s)^2 + c(X_S)^3 + d(X_S)^4 \qquad \text{Equation 2-62}$$

where η_S represents either η_n or η_p and X_S denotes either n/N_C or p/N_v. The constants *a*, *b*, *c*, and *d* can be found in either Reference [6] or [7]. For values of $X_S > 8.0$, the asymptotic expansion for the Fermi-Dirac inverse is used:

$$\eta_S = \left[\left(\frac{3\sqrt{\pi}}{4}X_S\right)^{4/3} - \frac{\pi^2}{6}\right]^{1/2} \qquad \text{Equation 2-63}$$

In either case, the Fermi-Dirac function is calculated from its inverse through a straight forward inversion process. Fermi-Dirac statistics can be selected with the **FERMIDIR** parameter on the **MODELS** statement.

Note:

When using Fermi-Dirac statistics in a simulation, it is strongly recommended that incomplete ionization of impurities also be included for accurate simulations (see "Incomplete Ionization of Impurities" on page 2-12).

With regard to the correlation between the mobilities and diffusivities in Equations 2-6 and 2-7, by assuming Boltzmann statistics, the Einstein relationship has been tacitly assumed, for example:

$$D_n = \frac{kT}{q}\mu_n \qquad \text{Equation 2-64}$$

$$D_p = \frac{kT}{q}\mu_p \qquad \text{Equation 2-65}$$

However, using Fermi-Dirac statistics,

$$D_n = \left(\frac{kT}{q}\mu_n\right)\mathbf{F}_{1/2}(\eta_n)/\mathbf{F}_{-1/2}(\eta_n) \qquad \text{Equation 2-66}$$

$$D_p = \left(\frac{kT}{q}\mu_p\right)\mathbf{F}_{1/2}(\eta_p)/\mathbf{F}_{-1/2}(\eta_p) \qquad \text{Equation 2-67}$$

Incomplete Ionization of Impurities

Poisson's equation (Equation 2-1, page 2-2) includes the ionized impurity concentrations N_D^+ and N_A^- in the expression for space charge. Although, for most practical cases, full impurity ionization may be assumed (that is, $N_D^+ = N_{D,total}$ and $N_A^- = N_{A,total}$), Medici can treat impurity freeze-out (Reference [8]) using Fermi-Dirac statistics with appropriate degeneracy factors for the conduction and valence bands.

The incomplete ionization of impurities model is selected by specifying the **INCOMPLE** parameter on the **MODELS** statement. By default, the model is applied individually to each impurity in the structure. If it is desired to apply the model to only select impurities, this can be accomplished by specifying the impurities using the **IMPURITY** parameter.

For donor impurities, the expression for incomplete ionization is given by

$$N_D^+ = \frac{N_D}{1 + \mathbf{GB}\ \exp[(E_{Fn} - E_C + \Delta E_D)/(kT)]} \qquad \text{Equation 2-68}$$

while for acceptor impurities you have

$$N_A^+ = \frac{N_A}{1 + \mathbf{GB}\ \exp[(E_V - E_{Fp} + \Delta E_A)/(kT)]} \qquad \text{Equation 2-69}$$

In these expressions, **GB** is the band degeneracy factor which is an impurity dependent parameter that can be changed from its default value on the **IMPURITY** statement.

Doping and Temperature Dependent Impurity Activation Energies

In the above expressions, ΔE_D and ΔE_A are the donor and acceptor impurity activation energies. By default, these are given by the impurity dependent parameter **EB0**, which can be changed from its default value on the **IMPURITY** statement.

Medici also provides for the doping and temperature dependence of the impurity activation energies (Reference [9]). If the parameter **ENERGY.L** is specified on the **MODELS** statement (in addition to **INCOMPLE**), the activation energy for donor and acceptor impurities will be given by

Equation 2-70

$$\Delta E_{D,A} = \mathbf{EB0} - \mathbf{ALPHA}\ N_{D,A}^{1/3} + \mathbf{BETA}\left(T^{-\mathbf{GAMMA}} - 300^{-\mathbf{GAMMA}}\right)\frac{kT}{q}$$

where **ALPHA**, **BETA**, and **GAMMA** are impurity dependent parameters that can be changed from their default values on the **IMPURITY** statement.

High Doping Transition to Complete Ionization

For very high dopings, the physics of incomplete ionization becomes very complicate due to effects such as band-edge lowering, band tailing and impurity band broadening. In most cases, such effects lead to complete ionization of the impurity, even at very low temperatures.

To account for the transition from incomplete ionization to complete ionization at high doping levels, Medici uses a simple approach similar to Reference [10]. If the parameter **HIGH.DOP** is specified on the **MODELS** statement, Medici will assume incomplete ionization according to the expressions above for impurity concentrations less than **HDT.MIN** and complete ionization for impurity concentration greater than **HDT.MAX**. Linear interpolation on the fraction of ionization is used in the interval between **HDT.MIN** and **HDT.MAX**.

The parameters **HDT.MIN** and **HDT.MAX** are impurity dependent and can be changed from their default values on the **IMPURITY** statement.

Low Temperature Simulation

In conjunction with the Fermi-Dirac statistics and impurity freeze-out, Medici has been structured to allow low temperature simulations. In general, simulations can be made at temperatures as low as 50 K without loss of quadratic convergence. Below this temperature, the carrier and ionization statistics develop sharp transitions that induce severe damping, resulting in loss of quadratic convergence in the nonlinear Newton step. Since several iterations are required below 50 K, the **ITLIMIT** parameter on the **METHOD** statement should be increased.

Due to the limited exponent range on some machines, it may be difficult to calculate the quasi-Fermi level of minority carriers. As the temperature decreases, the intrinsic carrier concentration n_i also decreases. An example of this is $n_i \approx 10^{-10}$ at 100 K for silicon.

In the quasineutral regions, the minority carrier concentration can easily underflow. Previously, such situations were handled by setting those concentrations to

zero. This does not, however, allow an accurate post-calculation of the minority carrier quasi-Fermi levels.

To compensate the quasi-Fermi level calculations, the majority carrier concentration and the relation $np = n_i^2$ are used to deduce the minority carrier concentrations should they underflow. Despite these efforts, false readings are occasionally observed at low temperatures in minority quasi-Fermi levels. The current calculations, however, are not affected by these false readings as the semiconductor equations are solved with the ψ, n, and p variable set.

Quantum Mechanical Effects in MOSFET Inversion Layers

For deep submicron devices, quantum mechanical effects are becoming increasingly more important. In particular, thinner oxides and higher substrate dopings used in advanced technologies lead to high electric fields that can quantize electron motion in the inversion layer. This phenomena has an effect on:

- Threshold voltages
- CV characteristics
- Carrier distribution

In general, a solution of Schrödinger's equation is needed to correctly account for quantum mechanical effects. However, approximate methods for dealing with these effects can be extremely useful in many situations.

By specifying **QM.PHILI** on the **MODELS** statement, the Medici program accounts for quantum mechanical effects in MOSFET inversion layers in an approximate manner by using van Dort's bandgap widening approach (Reference [84]). An expression for $\Delta E_{g,\,qm}$ that approximately accounts for both the splitting of energy levels in the conduction band to higher sub-bands and for a displacement of the carrier concentration away from the semiconductor-insulator interface, in the case of electron-inducing interface field, is given by

$$\Delta E_{g,\,qm} = \mathbf{KAPPA.N} \cdot \frac{13}{9} \cdot \beta \cdot \left(\frac{\varepsilon_{\text{semi}}}{4kT}\right)^{1/3} \cdot \left|E_n\right|^{2/3}. \qquad \text{Equation 2-71}$$

In this expression, E_n is the normal electric field at the semiconductor-insulator interface and β is a factor that can be determined experimentally. In Medici, β is given by Reference [84]:

$$\beta = 4.1 \times 10^{-8} \text{ eV-cm} \qquad \text{Equation 2-72}$$

The factor **KAPPA.N** was introduced in Reference [85] to account for the effect of quantized levels above the ground state and is used as a fitting parameter. In the accumulation regime, if **QM.EXTEN** is set (the default), β is further modified as suggested in Reference [101]:

$$\beta_{\text{accum}} = \beta \times \frac{\mathbf{N.ACCUM}}{\mathbf{N.ACCUM} + \left|N_{\text{dop}}\right|} \qquad \text{Equation 2-73}$$

A similar expression evaluates $\Delta E_{g,\,qm}$ in the case of hole-inducing inteface field with **KAPPA.N** replaced by **KAPPA.P** and **N.ACCUM** by **P.ACCUM**. Choosing **KAPPA.N** or **KAPPA.P**= 1.0 (the default for silicon) and disable **QM.EXTEN** results in the van Dort model. **KAPPA.N**, **KAPPA.P**, **N.ACCUM**, and **P.ACCUM** can be specified on the **MATERIAL** statement.

The value of $\Delta E_{g,\,qm}$ is used to calculate a new value for the intrinsic carrier concentration at the interface according to the expression

$$n_{i,\,qm} = n_{i,\,conv}\exp(-\Delta E_{g,\,qm}/(2kT)) \qquad \text{Equation 2-74}$$

where $n_{i,conv}$ is the conventional intrinsic carrier concentration at the interface when quantum mechanical affects are not taken into account.

Medici includes two options for computing how the effective intrinsic carrier concentration at the interface given by Equation 2-74 falls off with distance as you move away from the interface. Specifying **QM.METHO** = 1 on the **MODELS** statement uses an approach suggested by van Dort:

$$n_{i,\,eff} = (1 - F(a))n_{i,\,conv} + F(a)n_{i,\,qm} \qquad \text{Equation 2-75}$$

where *F(a)* is given by

$$F(a) = \frac{2\exp(-a^2)}{1 + \exp(-2a^2)}; \qquad a = \frac{d}{\mathbf{DREF.QM}} \qquad \text{Equation 2-76}$$

and *d* is the distance from the interface and **DREF.QM** is a reference distance for the material (specified on the **MATERIAL** statement). If **QM.METHO** = 2 is specified, then an approach suggested by Vande Voorde, et al. (Reference [85]) is used:

$$n_{i,\,eff} = n_{i,\,conv}\exp(-F(a)\Delta E_{g,\,qm}/(2kT)) \qquad \text{Equation 2-77}$$

where *F(a)* is the same function given above.

Parameters are also provided that allow you to control the conditions for which this model is applied. Specifying **QM.NORP**=1, -1, or 0 (on the **MATERIAL** statement) causes band-gap widening to be applied in n-type regions only, p-type regions only, or both, respectively. Specifying **QM.NORP**=2 or -2 will cause the program to behave as if **QM.NORP**=1 or -1 is specified when the channel is not inverted, but allows the program to switch to a **QM.NORP**=0 behavior as the device approaches inversion. Whenever possible the default value of **QM.NORP**=0 should be used. Specifying **QM.NORP**=1 or -1 (for P-channel or N-channel devices, respectively) often helps with convergence, but can sometimes introduce artificial electric fields at the edge of the channel that can result in excessive velocity saturation. Specifying **QM.NORP**=2 or -2 will usually improve convergence for gate characteristic simulations, but does not introduce the velocity saturation problem just mentioned.

The sign of the normal electric field at the interface can also be used as a criterion for applying **QM.PHILI**. Specifying **QM.EFIEL**=1 (on the **MATERIAL** state-

ment) will cause the program to apply the model only when the electric field points into the semiconductor (confining field for electrons). A value of **QM.EFIEL**=-1 will cause the program to apply the model only when the electric field points into the insulator (confining field for holes). A value of **QM.EFIEL**=0 will apply the model regardless of the sign of the electric field.

Finally, the magnitude of the normal electric field at the interface can be used as a criterion for applying **QM.PHILI**. The parameter **QM.EMIN** on the **MODELS** statement represents the minimum interface field that must be present before the QM corrections are applied.

Note:

The parameters* QM.NORP *and* QM.EFIEL *appear on both the* MODELS *and* MATERIAL *statements. Values specified on the* MODELS *statement will effect quantum mechanical calculations for the entire device structure. Generally, however,* QM.NORP *and* QM.EFIEL *should be specified on the* MATERIAL *statement, which will allow the quantum mechanical calculations to be applied differently in different regions.

Mobility Models

The carrier mobilities μ_n and μ_p account for scattering mechanisms in electrical transport. Medici provides several mobility model choices. Mobility models to be included in a Medici simulation can be specified on the **MODELS** statement. The parameters used in the models can be modified from their default values using the **MOBILITY** statement.

Low Field Mobility

Six choices are available to account for low field mobility:

- Constant values for electron and hole mobility can be specified with the **MUN0** and **MUP0** parameters
- A concentration dependent mobility model can be selected with the **CONMOB** parameter
- Either of two analytic mobility models can be selected with the **ANALYTIC** or **ARORA** parameters
- A carrier-carrier scattering mobility model can be selected with the **CCSMOB** parameter
- A unified mobility model that includes acceptor, donor, and carrier-carrier scattering can be selected with the **PHUMOB** parameter

Constant Mobility

The simplest alternative is to choose low field mobilities for electrons and holes that are constant throughout the structure. That is,

$$\mu_{0n} = \textbf{MUN0} \qquad \text{Equation 2-78}$$

$$\mu_{0p} = \texttt{MUP0} \qquad \text{Equation 2-79}$$

This is the default if no other low field mobility model is selected.

Concentration Dependent Mobility

The effect of impurity scattering can be included by using mobility values from tables which depend on the local total impurity concentration, $N_{total}(x,y)$. That is,

$$\mu_{0n} = \mu_{0n}(N_{total}(x,y)) \qquad \text{Equation 2-80}$$

$$\mu_{0p} = \mu_{0p}(N_{total}(x,y)) \qquad \text{Equation 2-81}$$

For silicon and gallium arsenide, Medici has tables of low field mobility versus total impurity concentration for both electrons and holes at T=300 K. The default values are shown in Table 2-1.

Using these tables, concentration dependent mobility can be selected with the **CONMOB** parameter on the **MODELS** statement. The table values may be modified using the **MOBILITY** statement.

Table 2-1 Mobility vs Impurity Concentration for Silicon and Gallium Arsenide (T=300 K)

Concentration (cm^{-3})	Mobility in Silicon (cm^2/V-s)		Mobility in GaAs (cm^2/V-s)	
	Electrons	Holes	Electrons	Holes
1.0E14	1350.0	495.0	8000.0	390.0
2.0E14	1345.0	495.0	7718.0	380.0
4.0E14	1335.0	495.0	7445.0	375.0
6.0E14	1320.0	495.0	7290.0	360.0
8.0E14	1310.0	495.0	7182.0	350.0
1.0E15	1300.0	491.1	7300.0	340.0
2.0E15	1248.0	487.3	6847.0	335.0
4.0E15	1200.0	480.1	6422.0	320.0
6.0E15	1156.0	473.3	6185.0	315.0
8.0E15	1115.0	466.9	6023.0	305.0
1.0E16	1076.0	460.9	5900.0	302.0
2.0E16	960.0	434.8	5474.0	300.0
4.0E16	845.0	396.5	5079.0	285.0
6.0E16	760.0	369.2	4861.0	270.0
8.0E16	720.0	348.3	4712.0	245.0
1.0E17	675.0	331.5	4600.0	240.0
2.0E17	524.0	279.0	3874.0	210.0
4.0E17	385.0	229.8	3263.0	205.0
6.0E17	321.0	203.8	2950.0	200.0
8.0E17	279.0	186.9	2747.0	186.9
1.0E18	252.0	178.0	2600.0	170.0
2.0E18	182.5	130.0	2060.0	130.0

Table 2-1 Mobility vs Impurity Concentration for Silicon and Gallium Arsenide (T=300 K)

Concentration (cm^{-3})	Mobility in Silicon (cm^2/V-s)		Mobility in GaAs (cm^2/V-s)	
	Electrons	Holes	Electrons	Holes
4.0E18	140.6	90.0	1632.0	90.0
6.0E18	113.6	74.5	1424.0	74.5
8.0E18	99.5	66.6	1293.0	66.6
1.0E19	90.5	61.0	1200.0	61.0
2.0E19	86.9	55.0		
4.0E19	83.4	53.7		
6.0E19	78.8	52.9		
8.0E19	71.6	52.4		
1.0E20	67.8	52.0		
2.0E20	52.0	50.8		
4.0E20	35.5	49.6		
6.0E20	23.6	48.9		
8.0E20	19.0	48.4		
1.0E21	17.8	48.0		

Analytic Mobility

As an alternative to the concentration dependent mobility tables for T=300 K, concentration- and temperature-dependent empirical mobility functions for silicon (References [11] and [12]) and gallium arsenide (References [4] and [6]) are also available. These are given by the expressions

$$\mu_{0n} = \texttt{MUN.MIN} + \frac{\texttt{MUN.MAX}\left(\frac{T}{300}\right)^{\texttt{NUN}} - \texttt{MUN.MIN}}{1 + \left(\frac{T}{300}\right)^{\texttt{XIN}}\left(\frac{N_{total}(x,y)}{\texttt{NREFN}}\right)^{\texttt{ALPHAN}}} \qquad \text{Equation 2-82}$$

$$\mu_{0p} = \texttt{MUP.MIN} + \frac{\texttt{MUP.MAX}\left(\frac{T}{300}\right)^{\texttt{NUP}} - \texttt{MUP.MIN}}{1 + \left(\frac{T}{300}\right)^{\texttt{XIP}}\left(\frac{N_{total}(x,y)}{\texttt{NREFP}}\right)^{\texttt{ALPHAP}}} \qquad \text{Equation 2-83}$$

where $N_{total}(x,y)$ is the local total impurity concentration (in #/cm^3), and T is the temperature (in K). This analytic mobility model can be selected with the **ANALYTIC** parameter on the **MODELS** statement.

The default values for the parameters used in the above expressions are shown in Table 2-2 for **SILICON** and **GAAS**. Default values for other materials can be found in Table 3-9, "III-V Compound Semiconductor Analytic Mobility Model

Parameters," on page 3-291 in Chapter 3. These values may be modified with the **MOBILITY** statement.

Table 2-2 Default Parameter Values for the Analytic Mobility Model

Parameter	SILICON	GAAS	Parameter	SILICON	GAAS
MUN.MIN	55.24	0.0	MUP.MIN	49.70	0.0
MUN.MAX	1429.23	8500.0	MUP.MAX	479.37	400.0
NREFN	1.072E17	1.69E17	NREFP	1.606E17	2.75E17
NUN	-2.3	-1.0	NUP	-2.2	-2.1
XIN	-3.8	0.0	XIP	-3.7	0.0
ALPHAN	0.73	0.436	ALPHAP	0.70	0.395

Aurora Mobility Model

Also available is an analytic mobility model based on work by Arora (Reference [13]) that takes into account total impurity concentration and temperature. The mobility expressions are

Equation 2-84

$$\mu_{0n} = \mathbf{MUN1.ARO}\left(\frac{T}{300}\right)^{\mathbf{EXN1.ARO}} + \frac{\mathbf{MUN2.ARO}\left(\frac{T}{300}\right)^{\mathbf{EXN2.ARO}}}{1 + \left(\frac{N_{total}(x,y)}{\mathbf{CN.ARORA}\left(\frac{T}{300}\right)^{\mathbf{EXN3.ARO}}}\right)^{\alpha_n}}$$

Equation 2-85

$$\mu_{0p} = \mathbf{MUP1.ARO}\left(\frac{T}{300}\right)^{\mathbf{EXP1.ARO}} + \frac{\mathbf{MUP2.ARO}\left(\frac{T}{300}\right)^{\mathbf{EXP2.ARO}}}{1 + \left(\frac{N_{total}(x,y)}{\mathbf{CP.ARORA}\left(\frac{T}{300}\right)^{\mathbf{EXP3.ARO}}}\right)^{\alpha_p}}$$

where

$$\alpha_n = \mathbf{AN.ARORA}\left(\frac{T}{300}\right)^{\mathbf{EXN4.ARO}}$$ Equation 2-86

$$\alpha_p = \mathbf{AP.ARORA}\left(\frac{T}{300}\right)^{\mathbf{EXP4.ARO}}$$ Equation 2-87

The Arora mobility model is selected with the **ARORA** parameter on the **MODELS** statement. Default parameter values for **SILICON** are shown in Table 2-3, and can be modified with the **MOBILITY** statement. Default parameter values for

other materials can be found in Table 3-10, "Arora Mobility Model Parameters," on page 3-291, in Chapter 3.

Table 2-3 Default Parameter Values for the Arora Model

Parameter	Default	Parameter	Default
MUN1.ARO	88.0	**MUP1.ARO**	54.3
MUN2.ARO	1252.0	**MUP2.ARO**	407.0
AN.ARORA	0.88	**AP.ARORA**	0.88
CN.ARORA	1.26×10^{17}	**CP.ARORA**	2.35×10^{17}
EXN1.ARO	-0.57	**EXP1.ARO**	-0.57
EXN2.ARO	-2.33	**EXP2.ARO**	-2.23
EXN3.ARO	2.4	**EXP3.ARO**	2.4
EXN4.ARO	-0.146	**EXP4.ARO**	-0.146

Carrier-Carrier Scattering Mobility

A semiempirical mobility model that takes into account carrier-carrier scattering effects, based on work by Dorkel and Leturcq (Reference [14]), has been incorporated in Medici These effects are important when high concentrations of electrons and holes are present in a device. The model also takes into account the effects of lattice scattering and ionized impurity scattering. The new model can be described by the expression,

$$\mu_{0n,0p} = \mu_{n,p}^{L}\left(\frac{\mathbf{A.LIC}}{1+\left[\mathbf{B.LIC}\left(\frac{\mu_{n,p}^{L}}{\mu_{n,p}^{IC}}\right)\right]^{\mathbf{EX.LIC}}} - \mathbf{C.LIC}\right) \qquad \text{Equation 2-88}$$

where the superscripts L, I, and C stand for lattice scattering, ionized impurity scattering, and carrier-carrier scattering, respectively. The term $\mu_{n,p}^{IC}$ is obtained from μ^{C} and $\mu_{n,p}^{I}$ according to Mathiessen's rule,

$$\mu_{n,p}^{IC} = \left[\frac{1}{\mu^{C}} + \frac{1}{\mu_{n,p}^{I}}\right]^{-1} \qquad \text{Equation 2-89}$$

The carrier-carrier scattering term μ^{C} is given by the expression,

$$\mu^{C} = \frac{\mathbf{A.CCS}\left(\frac{T}{300}\right)^{3/2}}{\sqrt{np}\ln\left(1+\mathbf{B.CCS}\left(\frac{T}{300}\right)^{2}(np)^{-1/3}\right)} \qquad \text{Equation 2-90}$$

The ionized impurity scattering terms μ_{n}^{I} and μ_{Lp}^{I} are given by the expressions,

$$\mu_n^I = \frac{\mathbf{AN.IIS}\left(\frac{T}{300}\right)^{3/2}}{N_{total}} \cdot g_B\left[\frac{\mathbf{BN.IIS}\left(\frac{T}{300}\right)^2}{n+p}\right] \quad \text{Equation 2-91}$$

$$\mu_p^I = \frac{\mathbf{AP.IIS}\left(\frac{T}{300}\right)^{3/2}}{N_{total}} \cdot g_B\left[\frac{\mathbf{BP.IIS}\left(\frac{T}{300}\right)^2}{n+p}\right] \quad \text{Equation 2-92}$$

where

$$g_B(x) = \left[\ln(1+x) - \frac{x}{1+x}\right]^{-1} \quad \text{Equation 2-93}$$

And finally, the lattice scattering terms μ_n^L and μ_p^L are given by,

$$\mu_n^L = \mathbf{MUN0.LAT}\left(\frac{T}{300}\right)^{-\mathbf{EXN.LAT}} \quad \text{Equation 2-94}$$

$$\mu_p^L = \mathbf{MUP0.LAT}\left(\frac{T}{300}\right)^{-\mathbf{EXP.LAT}} \quad \text{Equation 2-95}$$

This model can be selected by specifying the **CCSMOB** parameter on the **MODELS** statement. The parameters in the above equations can be adjusted from their default values on the **MOBILITY** statement.

Philips Unified Mobility

The Philips Unified mobility model (References [15] and [16]) takes into account the following:

- Distinct acceptor and donor scattering
- Carrier-carrier scattering
- Screening

It separately models majority and minority carrier mobilities and is appropriate for addressing bipolar devices. The majority mobility agrees with the work of Masetti, et al. (Reference [17]). The electron mobility is described by the following expressions,

$$\mu_n^{-1} = \mu_{latt,\,n}^{-1} + \mu_{D+A+P}^{-1} \quad \text{Equation 2-96}$$

where

$$\mu_{latt,\,n} = \mathbf{MMXN.UM}\left(\frac{T}{300}\right)^{-\mathbf{TETN.UM}} \quad \text{Equation 2-97}$$

Equation 2-98

$$\mu_{D+A+p} = \mu_{N,n}\frac{N_{sc,n}}{N_{sc,eff,n}}\left(\frac{\mathbf{NRFN.UM}}{N_{sc,n}}\right)^{\mathbf{ALPN.UM}} + \mu_{c,n}\frac{n+p}{N_{sc,eff,n}}$$

and $\mu_{N,n}$, $\mu_{c,n}$, $N_{sc,n}$ and $N_{sc,eff,n}$ are given by

$$\mu_{N,n} = \frac{\mathbf{MMXN.UM}^2}{\mathbf{MMXN.UM} - \mathbf{MMNN.UM}}\left(\frac{T}{300}\right)^{3(\mathbf{ALPN.UM})-1.5} \quad \text{Equation 2-99}$$

$$\mu_{c,n} = \frac{\mathbf{MMXN.UM}\cdot\mathbf{MMNN.UM}}{\mathbf{MMXN.UM} - \mathbf{MMNN.UM}}\left(\frac{300}{T}\right)^{0.5} \quad \text{Equation 2-100}$$

$$N_{sc,n} = N_D^* + N_A^* + p \quad \text{Equation 2-101}$$

$$N_{sc,eff,n} = N_D^* + N_A^* G(P_n) + \frac{p}{F(P_n)} \quad \text{Equation 2-102}$$

The effective impurity levels N_D^* and N_A^* take ultra-high doping effects into account and are defined by

$$N_D^* = N_D\left[1 + \frac{1}{\mathbf{CRFD.UM} + \left(\frac{\mathbf{NRFD.UM}}{N_D}\right)^2}\right] \quad \text{Equation 2-103}$$

$$N_A^* = N_A\left[1 + \frac{1}{\mathbf{CRFA.UM} + \left(\frac{\mathbf{NRFA.UM}}{N_A}\right)^2}\right] \quad \text{Equation 2-104}$$

The functions $F(P_n)$ and $G(P_n)$ that take the finite mass of scattering holes and the repulsive potential for acceptors into account, are given by

$$F(P_n) = \frac{0.7643\ P_n^{0.6478} + 2.2999 + 6.5502\frac{m_e}{m_h}}{P_n^{0.6478} + 2.3670 - 0.8552\frac{m_e}{m_h}} \quad \text{Equation 2-105}$$

$$G(P_n) = 1 - \frac{0.89233}{\left[0.41372 + P_n\left(\frac{m_0}{m_e}\frac{T}{300}\right)^{0.28227}\right]^{0.19778}} + \frac{0.005978}{\left[P_n\left(\frac{m_e}{m_0}\frac{300}{T}\right)^{0.72169}\right]^{1.80618}} \quad \text{Equation 2-106}$$

For values of $P_{n,} \leq P_{n,min}$ $G(P_{n,min})$ is used instead of $G(P_n)$, where $P_{n,min}$ is the value at which $G(P_n)$ reaches it's minimum. The P_n parameter that takes screening effects into account, is given by

$$P_n = \left(\frac{2.459}{3.97\times10^{13} N_{sc,n}^{-2/3}} + \frac{3.828}{\frac{1.36\times10^{20}}{n+p}\left(\frac{m_e}{m_0}\right)} \right)^{-1} \left(\frac{T}{300}\right)^2 \qquad \text{Equation 2-107}$$

Similar expressions hold for holes. The effective electron and hole mass used are $m_e = 1.0\ m_0$ and $m_h = 1.258\ m_0$ with m_0 being the free electron rest mass.

The model can be selected by specifying the **PHUMOB** parameter on the **MODELS** statement. The parameters in the above equations can be adjusted from their default values on the **MOBILITY** statement.

Note:

***Bandgap narrowing parameters are closely tied to the mobility model (Reference [18]). Using the* PHUMOB *model and its default settings produce sensible results when the bandgap narrowing parameters are set to* VO.BGN = 6.92 x 10^{-3}, NO.BGN = 1.3 x 10^{17}, and CON.BGN = 0.5.**

Surface Scattering

Along insulator-semiconductor interfaces, the carrier mobilities can be substantially lower than in the bulk of the semiconductor due to surface scattering. Medici can account for this difference in the following ways:

- Mobility degradation factors **GSURFN** and **GSURFP** can be applied at interfaces
- A surface mobility model can be specified with one of the parameters **SRFMOB** or **SRFMOB2**
- A transverse electric field mobility model can be specified with the parameter **PRPMOB**
- A number of MOS inversion layer models are available through the parameters **UNIMOB**, **LSMMOB**, **GMCMOB** **SHIRAMOB** and **TFLDMOB**
- Transverse field effects are also accounted for by **HPMOB**, described in "Hewlett-Packard Mobility Model" on page 2-37

Surface Mobility Degradation Factors

The mobility used for calculating current along semiconductor-insulator interfaces may be reduced by specifying degradation factors **GSURFN** and **GSURFP** that multiply the low field mobility. That is,

$$\mu_{S,n}(surface) = \textbf{GSURFN} \cdot \mu_{0n}(surface) \qquad \text{Equation 2-108}$$

$$\mu_{S,p}(surface) = \textbf{GSURFP} \cdot \mu_{0p}(surface) \qquad \text{Equation 2-109}$$

where $0 < \mathbf{GSURFN}, \mathbf{GSURFP} \leq 1$. The default value of both **GSURFN** and **GSURFP** is 1. Other values for **GSURFN** and **GSURFP** can be selected on the **MOBILITY** statement.

Surface Mobility Model

In addition to the degradation factors described above, Medici also allows the selection of an effective-field based surface mobility model that is applied only at insulator-semiconductor interfaces (Reference [19]). This model calculates effective mobilities at interfaces using the expressions

$$\mu_{S,n}(surface) = \mathbf{GSURFN} \cdot \left(\frac{E_{eff_{\perp},n}}{\mathbf{EREFN}} \right)^{-\mathbf{EXN.SM}} \mathbf{MUREFN} \qquad \text{Equation 2-110}$$

$$\mu_{S,p}(surface) = \mathbf{GSURFP} \cdot \left(\frac{E_{eff_{\perp},p}}{\mathbf{EREFP}} \right)^{-\mathbf{EXP.SM}} \mathbf{MUREFP} \qquad \text{Equation 2-111}$$

This model can be selected by specifying the **SRFMOB** parameter on the **MODELS** statement.

In Equations 2-110 and 2-111, $E_{eff_{\perp},n}$ and $E_{eff_{\perp},p}$ represent the perpendicular components of effective electric fields, $\vec{E}_{eff,n}$ and $\vec{E}_{eff,p}$, that are computed at the insulator-semiconductor interfaces. These are described in "Effective Electric Fields at Interfaces" on page 2-41.

Note:

This model is only applied at the interface because it assumes that the carrier inversion layer width is smaller than the grid spacing used at the interface. With this assumption, all the inversion charge effectively occurs at the interface and it is appropriate to use this model. If the grid spacing is smaller than the inversion layer width, then use of this model may not be appropriate.

Note:

Since* SRFMOB *is only applied at the interface, a low-field mobility model, such as* CONMOB*, should usually be selected to properly model current flow away from the surface. At the interface itself,* Medici *will use the minimum of the* SRFMOB *calculated mobility and the selected low-field mobility.

Enhanced Surface Mobility Model

An enhanced surface mobility model (Reference [20]) has been included in Medici that takes into account phonon scattering, surface roughness scattering, and charged impurity scattering. This model can be described by the expressions:

$$\mu_{S,n}(surface) = \mathbf{GSURFN} \cdot \mu_{eff,n} \qquad \text{Equation 2-112}$$

$$\mu_{S,p}(surface) = \mathbf{GSURFP} \cdot \mu_{eff,p} \qquad \text{Equation 2-113}$$

where

Equation 2-114

$$\frac{1}{\mu_{eff,n}} = \frac{1}{\texttt{MUN1.SM}}\left(\frac{E_{eff_{\perp},n}}{10^6}\right)^{\texttt{EXN1.SM}} + \frac{1}{\texttt{MUN2.SM}}\left(\frac{E_{eff_{\perp},n}}{10^6}\right)^{\texttt{EXN2.SM}} + \frac{1}{\texttt{MUN3.SM}}\left(\frac{N_B}{10^{18}}\right)\left(\frac{10^{12}}{N_{inv}}\right)^{\texttt{EXN3.SM}}$$

Equation 2-115

$$\frac{1}{\mu_{eff,p}} = \frac{1}{\texttt{MUP1.SM}}\left(\frac{E_{eff_{\perp},p}}{10^6}\right)^{\texttt{EXP1.SM}} + \frac{1}{\texttt{MUP2.SM}}\left(\frac{E_{eff_{\perp},p}}{10^6}\right)^{\texttt{EXP2.SM}} + \frac{1}{\texttt{MUP3.SM}}\left(\frac{N_B}{10^{18}}\right)\left(\frac{10^{12}}{N_{inv}}\right)^{\texttt{EXP3.SM}}$$

where the effective electric fields used in these expressions are described in "Effective Electric Fields at Interfaces" on page 2-41.

This model can be selected by specifying the **SRFMOB2** parameter on the **MODELS** statement. The parameters in the above equations can be adjusted from their default values on the **MOBILITY** statement.

Note:

This model is only applied at the interface because it assumes that the carrier inversion layer width is smaller than the grid spacing used at the interface. With this assumption, all the inversion charge effectively occurs at the interface, and it is appropriate to use this model. If the grid spacing is smaller than the inversion layer width, then use of this model may not be appropriate.

Note:

Since* SRFMOB *is only applied at the interface, a low-field mobility model, such as* CONMOB*, should usually be selected to properly model current flow away from the surface. At the interface itself,* Medici *will use the minimum of the* SRFMOB *calculated mobility and the selected low-field mobility.

Universal Mobility Model

The Universal Mobility Model is appropriate for the modeling of MOSFET inversion layers. This model is similar in form to the surface mobility model described previously, however, it does not require that the channel vertical grid spacing be greater than the inversion layer width.

The Universal Mobility Model can be selected by specifying **UNIMOB** on the **MODELS** statement. When selected, this model will be applied to all grid points

that occur within the MOSFET inversion layer. The mobility is given by the following expressions:

Equation 2-116

$$\mu_{S,n}\ (\textit{inversion layer}) = \mathbf{GSURFN} \cdot \frac{\mathbf{MUN.UNI}}{\left[1 + \left(\frac{E_{eff_{\perp},n}}{\mathbf{ECN.UNI}}\right)^{\mathbf{EXN.UNI}}\right]}$$

Equation 2-117

$$\mu_{S,p}\ (\textit{inversion layer}) = \mathbf{GSURFP} \cdot \frac{\mathbf{MUP.UNI}}{\left[1 + \left(\frac{E_{eff_{\perp},p}}{\mathbf{ECP.UNI}}\right)^{\mathbf{EXP.UNI}}\right]}$$

where

Equation 2-118

$$E_{eff,n} = \mathbf{ETAN} \cdot E_s + (\mathbf{ZETAN} - \mathbf{ETAN}) \cdot E_0$$

Equation 2-119

$$E_{eff,p} = \mathbf{ETAP} \cdot E_s + (\mathbf{ZETAP} - \mathbf{ETAP}) \cdot E_0$$

In the above expressions, E_s is the transverse semiconductor field at the interface and E_0 is the transverse field at the edge of the inversion layer.

Note:

At the present time, this model can only be used with "rectangular" grid structures in the inversion layer beneath the MOSFET gate.

Perpendicular Electric Field Mobility Model

Medici includes a model for the perpendicular electric field reduction of mobility that can be selected as an alternative to the models described previously. This model is applied at every position in the device and not just at interfaces or in the inversion layer.

The perpendicular electric field mobility model can be selected with the **PRPMOB** parameter on the **MODELS** statement. When selected, the low field mobility described in "Low Field Mobility" on page 2-16 (References [21] and [22]) is modified by the expressions

Equation 2-120

$$\mu_{S,n} = \mathbf{GSURFN} \cdot \frac{\mu_{0n}}{\sqrt{1 + \frac{E_{\perp,n}}{\mathbf{ECN.MU}}}}$$

Equation 2-121

$$\mu_{S,p} = \mathbf{GSURFP} \cdot \frac{\mu_{0p}}{\sqrt{1 + \frac{E_{\perp,p}}{\mathbf{ECP.MU}}}}$$

where $E_{\perp,n}$ and $E_{\perp,p}$ are the components of electric field that are perpendicular to the side of an element (the default) or the components of electric field perpendicular to the current direction (if **EJ.MOBIL** is selected on the **MODELS** state-

ment). At interfaces, $E_{\perp, n}$ and $E_{\perp, p}$ are replaced by effective electric field components $E_{eff_{\perp, n}}$ and $E_{eff_{\perp, p}}$ described in "Effective Electric Fields at Interfaces" on page 2-41.

Note:

The factors GSURFN and GSURFP are only applied at interfaces between semiconductor and insulator. Everywhere else, these factors should be considered to have values of unity.

Lombardi Surface Mobility Model

Medici also incorporates an empirical model that combines mobility expressions for semiconductor-insulator interfaces and for bulk silicon (Reference [23]). The basic equation is given by Mathiessen's rule:

$$\mu_S = \left[\frac{1}{\mu_{ac}} + \frac{1}{\mu_b} + \frac{1}{\mu_{sr}}\right]^{-1} \qquad \text{Equation 2-122}$$

where

- μ_S is total electron or hole mobility accounting for surface effects
- μ_{ac} is mobility degraded by surface acoustical phonon scattering
- μ_b is mobility in bulk silicon
- μ_{sr} is mobility degraded by surface roughness scattering.

The expressions for μ_{ac} for holes and electrons are

$$\mu_{ac, n} = \frac{\mathtt{BN.LSM}}{E_{\perp, n}} + \frac{\mathtt{CN.LSM}\ N_{total}^{\mathtt{EXN4.LSM}}}{T\ \sqrt[3]{E_{\perp, n}}} \qquad \text{Equation 2-123}$$

$$\mu_{ac, p} = \frac{\mathtt{BP.LSM}}{E_{\perp, p}} + \frac{\mathtt{CP.LSM}\ N_{total}^{\mathtt{EXP4.LSM}}}{T\ \sqrt[3]{E_{\perp, p}}} \qquad \text{Equation 2-124}$$

The equations for μ_b are

Equation 2-125

$$\mu_{b, n} = \mathtt{MUN0.LSM} + \frac{\mu_{max, n} - \mathtt{MUN0.LSM}}{1 + \left(\frac{N_{total}}{\mathtt{CRN.LSM}}\right)^{\mathtt{EXN1.LSM}}} - \frac{\mathtt{MUN1.LSM}}{1 + \left(\frac{\mathtt{CSN.LSM}}{N_{total}}\right)^{\mathtt{EXN2.LSM}}}$$

Equation 2-126

$$\mu_{b,p} = \mathbf{MUP0.LSM}\ \exp\left(\frac{-\mathbf{PC.LSM}}{N_{total}}\right) + \frac{\mu_{max,p}}{1+\left(\frac{N_{total}}{\mathbf{CRP.LSM}}\right)^{\mathbf{EXP1.LSM}}} - \frac{\mathbf{MUP1.LSM}}{1+\left(\frac{\mathbf{CSP.LSM}}{N_{total}}\right)^{\mathbf{EXP2.LSM}}}$$

where

$$\mu_{max,n} = \mathbf{MUN2.LSM}\left(\frac{T}{300}\right)^{-\mathbf{EXN3.LSM}}$$ Equation 2-127

$$\mu_{max,p} = \mathbf{MUP2.LSM}\left(\frac{T}{300}\right)^{-\mathbf{EXP3.LSM}}$$ Equation 2-128

And finally, the expressions for μ_{sr} are given by

$$\mu_{sr,n} = \left(\frac{\mathbf{DN.LSM}}{E_{\perp}^{\mathbf{EXN8.LSM}}}\right)$$ Equation 2-129

$$\mu_{sr,p} = \left(\frac{\mathbf{DP.LSM}}{E_{\perp}^{\mathbf{EXP8.LSM}}}\right)$$ Equation 2-130

The Lombardi surface mobility model is activated with the **LSMMOB** parameter on the **MODELS** statement, with all parameters accessible through the **MOBILITY** statement.

Generalized Mobility Curve Mobility Model

A mobility model that follows the Generalized Mobility Curve (GMC) (Reference [81]) can be selected by specifying **GMCMOB** on the **MODELS** statement. Although the model was developed for NMOS devices, it has been implemented here for both NMOS and PMOS devices. The default parameters for the electron mobility are taken from Reference [81]. The default parameters for hole mobility are the same as those for electron mobility. All parameters can be modified by on the **MOBILITY** statement.

The **GMCMOB** mobility model is a modified version of the Lombardi Surface Mobility Model (see "Lombardi Surface Mobility Model" on page 2-27). The **GMCMOB** contains additional terms to account for screened and unscreened impurity scattering. The model can be described as:

$$\mu_S = \left[\frac{1}{\mu_{universal}} + \frac{1}{\mu_{impurity}}\right]^{-1}$$ Equation 2-131

The universal part is given by the following expression:

$$\mu_{universal} = min\left\{ \left[\mu_{ac}^{-1} + \mu_{sr}^{-1}\right]^{-1}, \mu_b \right\} \quad \text{Equation 2-132}$$

where the acoustic-phonon terms for electrons and holes are given by

$$\mu_{ac,n} = \frac{\mathbf{BN.GMC}}{E_{\perp,n}} + \frac{\mathbf{CN.GMC}\ N_{total}^{\mathbf{EXN4.GMC}}}{T\sqrt[3]{E_{\perp,n}}} \quad \text{Equation 2-133}$$

$$\mu_{ac,p} = \frac{\mathbf{BP.GMC}}{E_{\perp,p}} + \frac{\mathbf{CP.GMC}\ N_{total}^{\mathbf{EXP4.GMC}}}{T\sqrt[3]{E_{\perp,p}}} \quad \text{Equation 2-134}$$

and the surface roughness terms for electrons and holes are given by

$$\mu_{sr,n} = \left(\frac{\mathbf{DN.GMC}}{E_{\perp}^{\mathbf{EXN8.GMC}}}\right) \quad \text{Equation 2-135}$$

$$\mu_{sr,p} = \left(\frac{\mathbf{DP.GMC}}{E_{\perp}^{\mathbf{EXP8.GMC}}}\right) \quad \text{Equation 2-136}$$

The bulk mobility terms for electrons and holes are exactly the same as those used in the Lombardi Surface Mobility model (see "Lombardi Surface Mobility Model" on page 2-27).

The impurity scattering term of Equation 2-131 is given by

$$\mu_{impurity} = max\left\{\mu_{screened}, \mu_{unscreened}\right\} \quad \text{Equation 2-137}$$

where the screened terms for electrons and holes are given by

$$\mu_{screened,n} = \mathbf{D1N.GMC} \cdot \frac{n^{\mathbf{EXN5.GMC}}}{N_{total}^{\mathbf{EXN6.GMC}}} \quad \text{Equation 2-138}$$

$$\mu_{screened,p} = \mathbf{D1P.GMC} \cdot \frac{p^{\mathbf{EXP5.GMC}}}{N_{total}^{\mathbf{EXP6.GMC}}} \quad \text{Equation 2-139}$$

and the unscreened terms for electrons and holes are given by

$$\mu_{unscreened,n} = \frac{\mathbf{D2N.GMC}}{N_{total}^{\mathbf{EXN7.GMC}}} \quad \text{Equation 2-140}$$

$$\mu_{unscreened,\,p} = \frac{\texttt{D2P.GMC}}{N_{total}^{\texttt{EXP7.GMC}}}$$ Equation 2-141

The default parameters for the **GMCMOB** model are given in the following table for materials defined as SILICON. Values for other materials are shown in Table 3-16, "Generalized Mobility Curve Model Parameters," on page 3-294.

Table 2-4 GMCMOB Default Parameters

Parameter	Default	Parameter	Default
BN.GMC	8.95×10^5	**BP.GMC**	8.95×10^5
CN.GMC	3.23×10^6	**CP.GMC**	3.23×10^6
DN.GMC	8.29×10^{14}	**DP.GMC**	8.29×10^{14}
D1N.GMC	1.35×10^{11}	**D1P.GMC**	1.35×10^{11}
D2N.GMC	2.89×10^8	**D2P.GMC**	2.89×10^8
EXN4.GMC	0.0284	**EXP4.GMC**	0.0284
EXN5.GMC	1.5	**EXP5.GMC**	1.5
EXN6.GMC	2.0	**EXP6.GMC**	2.0
EXN7.GMC	0.35	**EXP7.GMC**	0.35
EXN8.GMC	2.0	**EXP8.GMC**	2.0

Shirahata Mobility Model

The Shirahata mobility model [100] incorporates the transverse field effect into the Philips Unified Mobility (see "Philips Unified Mobility" on page 2-21). Specifically, the electron mobility due to lattice scattering in the Philips Unified Mobility is expanded to include the transverse field effect with the following expression:

Equation 2-142

$$\mu_{latt,\,n}^{Shirahata} = \frac{\mu_{latt,\,n}^{PhilipsUnified}}{\left(1 + \frac{E_{eff_\perp,\,n}}{\texttt{E1N.SHI}}\right)^{\texttt{EX1N.SHI}} + \left(\frac{E_{eff_\perp,\,n}}{\texttt{E2N.SHI}}\right)^{\texttt{EX2N.SHI}}}$$

A similar expression applies for hole mobility. The Shirahata mobility model is activated with the **SHIRAMOB** parameter on the **MODELS** statement, with the parameters accessible through the **MOBILITY** statement.

Note:

The* SHIRAMOB *and* PHUMOB *common parameters are accessed through the* MOBILITY *statement as* PHUMOB *parameters.

Transverse Field-Dependent Mobility Model

A MOSFET inversion layer mobility model, based on work at the University of Texas at Austin (References [79] and [80]), for simulation of MOSFETs with planar surfaces, has been generalized for use with devices implemented in Medici that have an arbitrary geometry.

The model calculates local mobility values so that an integration across the inversion layer results in the experimentally observed value of mobility.

The model has the form:

$$\mu = f(\mu_{eff}, E_{\parallel}, v^{sat}) + (E_{\perp} - E_0) \cdot \frac{df(\mu_{eff}, E_{\parallel}, v^{sat})}{dE_{\perp}} \qquad \text{Equation 2-143}$$

For electrons:

$$f = \frac{\mu_{eff,n}}{\left(1 + \left(\frac{\mu_{eff,n} E_{\parallel}}{v_n^{sat}}\right)^{\mathbf{BETAN}}\right)^{1/\mathbf{BETAN}}} \qquad \text{Equation 2-144}$$

where v_n^{sat} is computed by default (see next section) or can be specified explicitly with the **VSATN** parameter. The effective electron mobility is given by

$$\mu_{eff,n} = \left(\frac{1}{\mu_{ph,n}(E_{eff,n})} + \frac{1}{\mu_{sr,n}(E_{eff,n})} + \frac{1}{\mu_{cl,n}}\right)^{-1} \qquad \text{Equation 2-145}$$

$$E_{eff,n} = \frac{(E_{\perp} + E_0)}{2} \qquad \text{Equation 2-146}$$

$$\mu_{ph,n} = \left\{ \left(\mathbf{TEMPN.UT} \cdot \left(\frac{T}{300}\right)^{-5/2} \right)^{-1} + \right.$$

Equation 2-147

$$\left. \left(\frac{0.0388 \cdot \left(\frac{T}{300}\right)}{E_{eff,n}} + 1.73 \cdot 10^{-5} \cdot E_{eff,n}^{-1/3} \right)^{-1} \right\}^{-1}$$

$$\mathbf{PHONN.UT} \cdot \left(\frac{T}{300}\right)^{1/2} \cdot \left(0.09 \cdot \left(\frac{T}{300}\right)^{7/4} + 4.53 \cdot 10^{-8} \cdot n^{-1/4} \cdot \frac{N_f}{\left(\frac{T}{300}\right)} \right)$$

$$\mu_{sr,\,n} = \frac{\mathbf{SURFN.UT}}{E_{eff,\,n}^{2}} \qquad \text{Equation 2-148}$$

$$\mu_{cl,\,n} = \frac{\mathbf{COULN.UT} \cdot \left(\frac{T}{300}\right)^{\frac{3}{2}}}{N_a \cdot \left(\ln(1+\gamma_n) - \frac{\gamma_n}{(1+\gamma_n)}\right)} \qquad \text{Equation 2-149}$$

$$\gamma_n = \frac{2 \cdot 10^{19}}{n}\left(\frac{T}{300}\right)^{2} \qquad \text{Equation 2-150}$$

$$\mu_{inv,\,n} = \mu_{eff,\,n} \cdot \mathbf{INV.N.UT} \qquad \text{Equation 2-151}$$

$$\mu_{acc,\,n} = \mu_{eff,\,n} \cdot \mathbf{ACC.N.UT} \qquad \text{Equation 2-152}$$

For holes:

$$f = \frac{\mu_{eff,\,p}}{\left(1 + \left(\frac{\mu_{eff,\,p} \cdot E_{\parallel}}{v_p^{sat}}\right)^{\mathbf{BETAP}}\right)^{1/\mathbf{BETAP}}} \qquad \text{Equation 2-153}$$

where v_p^{sat} is computed by default (see next section) or can be specified explicitly with the **VSATP** parameter. The effective hole mobility is given by

$$\mu_{eff,\,p} = \left(\frac{1}{\mu_{ph,\,p}(E_{eff,\,p})} + \frac{1}{\mu_{sr,\,p}(E_{eff,\,p})} + \frac{1}{\mu_{cl,\,p}}\right)^{-1} \qquad \text{Equation 2-154}$$

$$E_{eff,\,p} = \frac{(E_{\perp} + 2 \cdot E_0)}{3} \qquad \text{Equation 2-155}$$

$$\mu_{ph,\,p} = \left\{ \left(\mathbf{TEMPP.UT} \cdot \left(\frac{T}{300} \right)^{-7/5} \right)^{-1} + \right.$$

Equation 2-156

$$\left. \left(\frac{\dfrac{0.039 \cdot \left(\dfrac{T}{300} \right)}{E_{eff,\,p}} + 1.51 \cdot 10^{-5} \cdot E_{eff,\,p}^{-1/3}}{\mathbf{PHONP.UT} \cdot \left(\dfrac{T}{300} \right)^{1/2} \cdot \left(0.334 \cdot \left(\dfrac{T}{300} \right)^{3/2} + 3.14 \cdot 10^{-7} \cdot p^{-0.3} \cdot \dfrac{N_f}{\left(\dfrac{T}{300} \right)} \right)} \right)^{-1} \right\}^{-1}$$

$$\mu_{sr,\,p} = \frac{\mathbf{SURFP.UT}}{E_{eff,\,p}}$$

Equation 2-157

$$\mu_{cl,\,p} = \frac{\mathbf{COULP.UT} \cdot \left(\dfrac{T}{300} \right)^{\frac{3}{2}}}{N_d \cdot \left(\ln(1+\gamma_p) - \dfrac{\gamma_p}{(1+\gamma_p)} \right)}$$

Equation 2-158

$$\gamma_p = \frac{8.4 \cdot 10^{16}}{p \cdot \left(\dfrac{T}{300} \right)^{3.4}}$$

Equation 2-159

$$\mu_{inv,\,p} = \mu_{eff,\,p} \cdot \mathbf{INV.P.UT}$$

Equation 2-160

$$\mu_{acc,\,p} = \mu_{eff,\,p} \cdot \mathbf{ACC.P.UT}$$

Equation 2-161

where

- μ_{eff} is experimentally measured effective (or average) carrier mobility in the inversion layer
- E_0 is transverse electric field at the edge of the inversion layer
- μ_{ph} is mobility degraded by acoustic phonon scattering

- μ_{sr} is mobility degraded by surface roughness scattering
- μ_{cl} is mobility degraded by Coulombic scattering
- μ_{inv} is mobility at the surface of the weak inversion layers
- μ_{acc} is mobility at the surface of the accumulation layers
- N_f is interface charge

The transverse field dependent mobility model is activated with the **TFLDMOB** parameter on the **MODELS** statement, with the parameters accessible through the **MOBILITY** statement. Note that parameters **VSATN**, **VSATP**, **BETAN**, and **BETAP** used in the Caughey-Thomas expression above are shared with the parallel field mobility model (**FLDMOB**) described in the next section.

This model can be applied to a device with a planar, vertical, tilted, or nonplanar channel. There are no restrictions in regard to the mesh when using this model, and it may be imported from a process simulator. Moreover, the model can be used for any device where a substantial transverse field is present, not just for MOSFETs.

Note:

ETAN and ETAP, two of the parameters for effective electric field at the surface, should be set to 1 when TFLDMOB is requested.

Note:

This model has been calibrated using source/drain contact resistances of approximately 500 Ohm/micron for nFETs and 900 Ohm/micron for pFETs. If the model is used without the external resistances attached to the terminals, the drain current most likely will be 50 to 100% higher than measured.

Note:

When using TFLDMOB for n-channel MOSFETs, the FLDMOB parameter (see "High Field Effects" on page 2-34) should also be specified. However, for p-channel MOSFETs, best results are obtained when FLDMOB is not specified.

High Field Effects

Mobility models are available in Medici that can account for effects due to high field in the direction of current flow:

- **FLDMOB**
- **HPMOB**
- **LUCMOB**

These models may be selected on the **MODELS** statement. The **HPMOB** model also takes into account mobility dependence on perpendicular field. The **LUCMOB**

model is an all-inclusive model accounting for low-field, transverse field and longitudinal field.

When simulations are performed that include the solution of the carrier energy balance equation, selecting the **TMPMOB** parameter on the **MODELS** statement will use an effective electric field calculated from carrier temperature in the mobility calculations. This is described in "Carrier Temperature-Based Mobility" on page 2-117.

Parallel Field Mobility

Field-dependent models for mobility can be derived that account for carrier heating and velocity saturation effects. This is done by using analytic expressions for the drift velocity v_d as a function of the electric field in the direction of current flow, $E_{||}$, and defining $\mu(E_{||}) = v_d(E_{||})/E_{||}$. Such models have been implemented for both silicon and gallium arsenide.

Medici offers great flexibility for modeling complex device structures by allowing different parallel field-dependent mobility models to be used in different regions of the structure. For example, it is possible to use a "silicon-like" mobility model in one region of the structure and a "gallium arsenide-like" mobility model in a different region. It is also possible to locally disable the parallel field-dependent mobility calculations, in which case the low field mobilities described in the previous sections are used.

To *activate* the parallel field mobility calculations, the **FLDMOB** parameter on the **MODELS** statement must be specified. To *specify* that a particular parallel field mobility model is to be used in specific materials or regions, the **FLDMOB**=<n> parameter on the **MOBILITY** statement should be used, where <n> is an integer that identifies the model.

To clarify this point, the **MODELS** statement is used to invoke the parallel field-dependent calculations, but the **MOBILITY** statement is used to select which models to use and where they are to apply. In most cases, the default model choices are appropriate and there is no need to make adjustments on the **MOBILITY** statement.

The parallel field-dependent mobility model choices in Medici are given below, as well as how these models are selected and the defaults that are used in the program.

Disabling Parallel Field-Dependent Mobility

When **FLDMOB**=0 is specified on the **MOBILITY** statement, the parallel field-dependent mobility calculation is disabled in the materials or regions for which the statement applies. In these regions, the low field values $\mu_{S,n}$ and $\mu_{S,p}$ are used (which may include the scattering mechanisms described in "Surface Scattering" on page 2-23).

Caughey-Thomas Expression

When **FLDMOB**=1 is specified on the **MOBILITY** statement, a Caughey-Thomas expression for both electron and hole mobility is used in the materials or regions for which the statement applies. This is the default for regions that are defined as **SILICON**, **POLYSILI**, **SEMICOND**, **GERMANIU**, **SIC**, or **SIGE** on the **REGION** statement. In this case, the mobility has the form (Reference [11]):

$$\mu_n = \frac{\mu_{S,n}}{\left[1 + \left(\frac{\mu_{S,n} E_{\|,n}}{v_n^{sat}}\right)^{\mathbf{BETAN}}\right]^{1/\mathbf{BETAN}}}$$ Equation 2-162

$$\mu_p = \frac{\mu_{S,p}}{\left[1 + \left(\frac{\mu_{S,p} E_{\|,p}}{v_p^{sat}}\right)^{\mathbf{BETAP}}\right]^{1/\mathbf{BETAP}}}$$ Equation 2-163

where

- $\mu_{S,n}$ and $\mu_{S,p}$ are the low field mobilities (which may include the scattering mechanisms described in "Surface Scattering" on page 2-23)
- v_n^{sat} and v_p^{sat} are saturation velocities for electrons and holes, respectively.

Values for v_n^{sat} and v_p^{sat} are computed by default from the expression (Reference [4])

$$v_{n,p}^{sat}(T) = \frac{2.4 \times 10^7}{1 + 0.8 \cdot \exp\left(\frac{T}{600}\right)}$$ Equation 2-164

Alternatively, specific values for v_n^{sat} and v_p^{sat} can be selected with **VSATN** and **VSATP** parameters on the **MOBILITY** statement. Note that in this model, the carrier drift velocity ($\mu E_{\|}$) saturates at high fields and $\partial(\mu E_{\|}) / \partial E_{\|} > 0$ for all values of $E_{\|}$.

Gallium Arsenide-Like Mobility

When **FLDMOB**=2 is specified on the **MOBILITY** statement, a mobility model that has often been used for modeling gallium arsenide is used in the materials or regions for which the statement applies. This is the default for regions that are defined as **GAAS** or **ALGAAS** on the **REGION** statement. In this case, the mobility has the form (Reference [24])

$$\mu_n = \frac{\mu_{S,n} + \frac{v_n^{sat}}{E_{\|,n}}\left(\frac{E_{\|,n}}{\mathbf{EON}}\right)^4}{1 + \left(\frac{E_{\|,n}}{\mathbf{EON}}\right)^4}$$ Equation 2-165

$$\mu_p = \frac{\mu_{S,p} + \frac{v_p^{sat}}{E_{\|,p}}\left(\frac{E_{\|,p}}{\mathbf{EOP}}\right)^4}{1 + \left(\frac{E_{\|,p}}{\mathbf{EOP}}\right)^4} \qquad \text{Equation 2-166}$$

where values for v_n^{sat} and v_p^{sat} are computed by default from the expression (Reference [25])

$$v_{n,p}^{sat}(T) = 11.3 \times 10^6 - 1.2 \times 10^4 T \qquad \text{Equation 2-167}$$

Specific values of v_n^{sat} and v_p^{sat} can be selected with **VSATN** and **VSATP** parameters on the **MOBILITY** statement. Note that as the electric field increases in this model, the carrier drift velocity ($\mu E_{\|}$) reaches a peak and then begins to decrease at high fields due to the transferred electron effect. The result is that $\partial(\mu E_{\|}) / \partial E_{\|} < 0$ for high fields.

Alternative Parallel Field-Dependent Expression

When **FLDMOB**=3 is specified on the **MOBILITY** statement, the following parallel field-dependent mobility expression will be used in the materials or regions for which the statement applies [47]:

$$\mu_{n,p} = \frac{2\mu_{S,n,p}}{1 + \left[1 + 4\left(\frac{\mu_{S,n,p} E_{\|,(n,p)}}{v_{n,p}^{sat}}\right)^2\right]^{1/2}} \qquad \text{Equation 2-168}$$

Note that this expression is used to describe the parallel field-dependence for both electrons and holes with the **LUCMOB** mobility model.

Hewlett-Packard Mobility Model

The Hewlett-Packard mobility model (References [94] and [95]) takes into account dependence on electric fields both parallel and perpendicular to the direction of current flow. The expressions for mobility used by this model are

$$\mu_n = \frac{\mu_{\perp,n}}{\sqrt{1 + \frac{\left(\frac{\mu_{\perp,n} E_{\|,n}}{\mathbf{VCN.HP}}\right)^2}{\frac{\mu_{\perp,n} E_{\|,n}}{\mathbf{VCN.HP}} + \mathbf{GN.HP}} + \left(\frac{\mu_{\perp,n} E_{\|,n}}{\mathbf{VSN.HP}}\right)^2}} \qquad \text{Equation 2-169}$$

$$\mu_p = \frac{\mu_{\perp,p}}{\sqrt{1 + \frac{\left(\frac{\mu_{\perp,p}E_{\parallel,p}}{\mathbf{VCP.HP}}\right)^2}{\frac{\mu_{\perp,p}E_{\parallel,p}}{\mathbf{VCP.HP}} + \mathbf{GP.HP}} + \left(\frac{\mu_{\perp,p}E_{\parallel,p}}{\mathbf{VSP.HP}}\right)^2}}$$ Equation 2-170

The expressions for $\mu_{\perp,n}$ and $\mu_{\perp,p}$ are given by

$$\mu_{\perp,n} = \frac{\mathbf{MUN0.HP}}{1 + \frac{E_{\perp,n}}{\mathbf{ECN.HP}}}, \quad if \quad N_{total}(x,y) < \mathbf{NRFN.HP}$$ Equation 2-171

$$\mu_{\perp,p} = \frac{\mathbf{MUP0.HP}}{1 + \frac{E_{\perp,p}}{\mathbf{ECP.HP}}}, \quad if \quad N_{total}(x,y) < \mathbf{NRFP.HP}$$ Equation 2-172

The default value for both **NRFN.HP** and **NRFP.HP** is 5 x 10^{17}. If the above conditions are not satisfied, then $\mu_{\perp,n} = \mu_{0,n}$ and $\mu_{\perp,p} = \mu_{0,p}$, where $\mu_{0,n}$ and $\mu_{0,p}$ are the low field mobility values described in "Low Field Mobility" on page 2-16.

The HP mobility model may be selected by specifying **HPMOB** on the **MODELS** statement. All mobility parameters are accessible through the **MOBILITY** statement.

Lucent Mobility Model

An all inclusive model has been developed by Darwish, et al. [96] that incorporates slightly modified versions of the Philips Unified Mobility model (**PHUMOB**) and the Lombardi Surface Mobility model (**LSMMOB**), as well as accounting for high field effects. The model is selected with the **LUCMOB** parameter on the **MODELS** statement.

For low longitudinal field, the carrier mobility is given by

$$\mu_S = \left[\frac{1}{\mu_b} + \frac{1}{\mu_{ac}} + \frac{1}{\mu_{sr}}\right]^{-1}$$ Equation 2-173

where

- μ_b is mobility in bulk silicon
- μ_{ac} is mobility degraded by surface acoustical phonon scattering
- μ_{sr} is mobility degraded by surface roughness scattering.

The equations for μ_b are very similar to those used for **PHUMOB**. For electron mobility we have

$$\frac{1}{\mu_{b,n}} = \frac{1}{\mu_{L,n}} + \frac{1}{\mu_{I,n}}$$ Equation 2-174

where

$$\mu_{L,n} = \mathbf{MMXN.UM}\left(\frac{T}{300}\right)^{-\mathbf{TETN.UM}} \quad \text{Equation 2-175}$$

Equation 2-176

$$\mu_{I,n} = \mu_{N,n}\left(\frac{N_{total}}{N_{Ieff,n}}\right)\left(\frac{\mathbf{NRFN.UM}}{N_{total}}\right)^{\mathbf{ALPN.UM}} + \mu_{c,n}\left(\frac{n+p}{N_{Ieff,n}}\right)$$

and $\mu_{N,n}$, $\mu_{c,n}$ and $N_{Ieff,n}$ are given by

$$\mu_{N,n} = \frac{\mathbf{MMXN.UM}^2}{\mathbf{MMXN.UM} - \mathbf{MMNN.UM}}\left(\frac{T}{300}\right)^{3(\mathbf{ALPN.UM})-1.5} \quad \text{Equation 2-177}$$

$$\mu_{c,n} = \frac{\mathbf{MMXN.UM} \cdot \mathbf{MMNN.UM}}{\mathbf{MMXN.UM} - \mathbf{MMNN.UM}}\left(\frac{300}{T}\right)^{0.5} \quad \text{Equation 2-178}$$

$$N_{Ieff,n} = N_D + N_A G(P_n). \quad \text{Equation 2-179}$$

For holes, $N_{Ieff,p}$ is given by

$$N_{Ieff,p} = N_A + N_D G(P_p) \quad \text{Equation 2-180}$$

The functions $G(P_n)$ and $G(P_p)$ and the screening parameters P_n and P_p are described in "Philips Unified Mobility" on page 2-21.

Similar expressions hold for hole mobility. Note that the parameters used in the above expressions are the same as those used with **PHUMOB**.

The expressions for μ_{ac} and μ_{sr} for electrons and holes are very similar to those used with the **LSMMOB**:

$$\mu_{ac,n} = \frac{\mathbf{BN.LUC}}{E_{\perp,n}} + \frac{\mathbf{CN.LUC}\ N_{total}^{\mathbf{EXN4.LUC}}}{(T/300)^{\mathbf{KN.LUC}}\sqrt[3]{E_{\perp,n}}} \quad \text{Equation 2-181}$$

$$\mu_{ac,p} = \frac{\mathbf{BP.LUC}}{E_{\perp,p}} + \frac{\mathbf{CP.LUC}\ N_{total}^{\mathbf{EXP4.LUC}}}{(T/300)^{\mathbf{KP.LUC}}\sqrt[3]{E_{\perp,p}}} \quad \text{Equation 2-182}$$

$$\mu_{sr,n} = \left(\frac{\mathbf{DN.LUC}}{E_{\perp,n}^{\gamma_n}}\right) \quad \text{Equation 2-183}$$

$$\mu_{sr,\,p} = \left(\frac{\mathbf{DP.LUC}}{E_{\perp,\,p}^{\gamma_p}} \right)$$ Equation 2-184

where

$$\gamma_n = \mathbf{AN.LUC} + \frac{\mathbf{FN.LUC}(n+p)}{N_{total}^{\mathbf{EXN9.LUC}}}$$ Equation 2-185

$$\gamma_p = \mathbf{AP.LUC} + \frac{\mathbf{FP.LUC}(n+p)}{N_{total}^{\mathbf{EXP9.LUC}}}$$ Equation 2-186

Finally, the total mobility is obtained using the expression:

$$\mu_{n,\,p} = \frac{2\mu_{S,\,n,p}}{1 + \left[1 + \left(\frac{2\mu_{S,\,n,\,p} E_{\|,n,p}}{v_{n,p}^{sat}} \right)^2 \right]^{1/2}}$$ Equation 2-187

Default silicon parameters for **LUCMOB** are given in the following table:

Table 2-5 LUCMOB Default Parameters

Parameter	Default	Parameter	Default
AN.LUC	2.58	**AP.LUC**	2.18
BN.LUC	3.61×10^{7}	**BP.LUC**	1.51×10^{7}
CN.LUC	1.70×10^{4}	**CP.LUC**	4.18×10^{3}
DN.LUC	3.58×10^{18}	**DP.LUC**	4.10×10^{15}
FN.LUC	6.85×10^{-21}	**FP.LUC**	7.82×10^{-21}
KN.LUC	1.7	**KP.LUC**	0.9
EXN4.LUC	0.0233	**EXP4.LUC**	0.0119
EXN9.LUC	0.0767	**EXP9.LUC**	0.123

Electric Field Calculations for Mobility Models

Two aspects of the electric field calculations that are used within the mobility models described in the previous sections are described in this section.

- The effective electric field calculation that is performed (by default) at semiconductor-insulator interfaces is described first.
- A discussion of how the parallel and perpendicular components of electric field are calculated, including a description of the **EJ.MOBIL** parameter.

Effective Electric Fields at Interfaces

By default, Medici calculates and uses effective electric fields at semiconductor-insulator interfaces when performing the mobility calculations described in the previous sections. The vector expressions for these effective fields are given by the following equations:

Equation 2-188

$$\vec{E}_{eff,n} = \left\{ \mathbf{ZETAN}\ E_{semi,\perp_s} + \mathbf{ETAN}\left[\left(\frac{\varepsilon_{insul}}{\varepsilon_{semi}}\right)E_{insul,\perp_s} + \frac{q\ \mathbf{QF}}{\varepsilon_{semi}} - E_{semi,\perp_s}\right]\right\}\hat{n}_{\perp} + E_{semi,\parallel_s}\hat{n}_{\parallel}$$

Equation 2-189

$$\vec{E}_{eff,p} = \left\{ \mathbf{ZETAP}E_{semi,\perp_s} + \mathbf{ETAP}\left[\left(\frac{\varepsilon_{insul}}{\varepsilon_{semi}}\right)E_{insul,\perp_s} + \frac{q\ \mathbf{QF}}{\varepsilon_{semi}} - E_{semi,\perp_s}\right]\right\}\hat{n}_{\perp} + E_{semi,\parallel_s}\hat{n}_{\parallel}$$

where $E_{semi,\perp_s}$ and $E_{insul,\perp_s}$ are components of the electric field in the semiconductor and insulator, respectively. These components are perpendicular to the interface. $E_{semi,\parallel_s}$ is the component of the electric field in the semiconductor that is parallel to the interface, and $\hat{n}_{\perp}$ and $\hat{n}_{\parallel}$ are unit vectors normal to and parallel to the interface. Note that the above effective electric fields also account for the presence of interface charge, **QF**.

When calculating the perpendicular components of the effective fields, $E_{eff_{\perp},n}$ and $E_{eff_{\perp},p}$, Medici by default calculates the components perpendicular to the interface. However, if **EJ.MOBIL** is selected on the **MODELS** statement, then $E_{eff_{\perp},n}$ and $E_{eff_{\perp},p}$ are the components of effective electric field perpendicular to current flow (see Electric Field Components below).

Electric Field Components

When assembling the current densities within a device structure, Medici calculates the Scharfetter-Gummel current densities (Reference [26]) along each side of each triangular element. For example, the electron and hole current densities between nodes 1 and 2 in Figure 2-1 are given by

$$J_{n_{12}} = \frac{q\,\mu_{n_{12}}}{(q/KT)\,d_{12}}[(n_2 - n_1)B(\Delta_{12}) - \Delta_{12}n_1] \qquad \text{Equation 2-190}$$

$$J_{p_{12}} = \frac{q\,\mu_{p_{12}}}{(q/KT)\,d_{12}}[(p_1 - p_2)B(\Delta_{12}) - \Delta_{12}p_2] \qquad \text{Equation 2-191}$$

where

- $\mu_{n_{12}}$ and $\mu_{p_{12}}$ are the electron and hole mobility values for the side
- d_{12} is the distance between nodes 1 and 2, B is the Bernoulli function,
- Δ_{12} is the potential difference between nodes 1 and 2 scaled by kT/q.

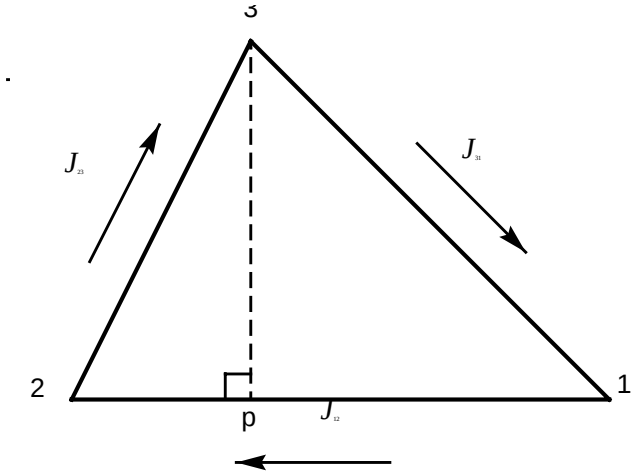


Figure 2-1 Typical triangular element used in Medici

The mobility values $\mu_{n_{12}}$ and $\mu_{p_{12}}$ used in the above expressions may depend on electric field. By default, the electric field components $E_{\parallel}$ and $E_{\perp}$ used in the mobility models are the components of electric field parallel and perpendicular to the side for which the mobility values are being calculated. For example, for the side connecting nodes 1 and 2,

$$E_{\parallel} = E_{\parallel_{12}} = \frac{|\psi_2 - \psi_1|}{d_{12}} \qquad \text{Equation 2-192}$$

$$E_{\perp} = E_{\perp_{12}} = \frac{|\psi_3 - \psi_p|}{d_{3p}} \qquad \text{Equation 2-193}$$

where ψ_1, ψ_2, ψ_3, and ψ_p are the potentials at nodes 1, 2, 3, and at point p, respectively. Mobility calculations using the electric field components obtained in this way can be quite accurate in situations where the current flow in a device is directed primarily along the edges of the triangular elements.

In general, however, current flow in a device is not directed exactly parallel to element edges. For such cases, the electric field components $E_{\parallel}$ and $E_{\perp}$ used in the mobility models should be the components of electric field that are parallel and perpendicular to current flow. If **`EJ.MOBIL`** is selected on the **`MODELS`** statement, Medici calculates electric field components for each element as follows:

$$E_{\parallel} = \frac{\max(0, \vec{E} \cdot \vec{J})}{|\vec{J}|} \qquad \text{Equation 2-194}$$

$$E_{\perp} = \frac{|\vec{E} \times \vec{J}|}{|\vec{J}|} \qquad \text{Equation 2-195}$$

where $\vec{E}$ is the electric field vector for the element and $\vec{J}$ is the current density lvector for the element that is computed as a weighted average of the Scharfetter-Gummel current densities for each side.

Although this method of obtaining electric field components generally results in more accurate mobility calculations, it is also computationally more complex and it may require more iterations for convergence when obtaining solutions.

Mobility Model Choices

The program provides several choices for mobility. Table 2-6 is intended to simplify the process of mobility model selection by illustrating the possible combinations of available mobility models.

The mobility models presently available in Medici can be classified into three categories:

- Low Field
- Transverse Field
- Parallel Field

These three categories are identified by the column headings shown in Table 2-6. The mobility choices, based on the dependencies accounted for by each model, appear in the appropriate column(s) below the headings. Note that some of the mobility models span more than one column.

Table 2-6 Mobility Choices in Medici

Low Field	Transverse Field	Parallel Field
LUCMOB		
CCSMOB	**HPMOB**	
LSMMOB		**FLDMOB**
GMCMOB		**TMPMOB**
SHIRAMOB		
ANALYTIC	**PRPMOB**	
ARORA	**SRFMOB**	
CONMOB	**SRFMOB2**	
PHUMOB	**TFLDMOB**	

When selecting mobility models for a simulation, only one model from each of the three categories shown in the table is allowed. For example, both "**CONMOB PRPMOB FLDMOB**" and "**LSMMOB FLDMOB**" represent valid choices. However, "**LSMMOB HPMOB**" is not a valid choice because it includes two entries in the "Transverse Field" category.

Comparison of Mobility Models

This section presents examples illustrating how different mobility options affect a particular simulation. For this purpose, the N-channel MOSFET structure is used (see Chapter 1)

Figure 2-2 shows the results of selecting several different methods to account for surface scattering effects. The gate and drain characteristics of the N-channel MOSFET under consideration were simulated using

1. A reduction of electron mobility at the interface by a specified factor (**GSURFN**=.75).
2. The surface mobility model (**SRFMOB**).
3. The enhanced surface mobility model (**SRFMOB2**).
4. The Lombardi surface mobility model (**LSMMOB**).
5. The Hewlett-Packard mobility model (**HPMOB**).
6. The perpendicular electric field mobility model (**PRPMOB**).
7. The perpendicular electric field mobility model (**PRPMOB**) using electric field components parallel and perpendicular to current flow (**EJ.MOBIL**).

In all cases, the parameters **CONMOB** and **FLDMOB** were also selected, except when using **HPMOB**, when simultaneous use of **FLDMOB** is not allowed, and when using **LSMMOB**, when **CONMOB** is not allowed. All cases used default values for parameters associated with the selected models.

GSURFN Results

Although the degradation factors **GSURFN** and **GSURFP** can be used to approximate the effects of surface scattering, the actual values used in most cases depend on the structure and biases involved. From the gate characteristics shown in Figure 2-2, it can be seen that specifying **GSURFN**=.75 provides reasonable agreement with most of the transverse field-dependent mobility models for V_{gs}< 2*V*. However, smaller values of **GSURFN** would be required to provide agreement at higher values of V_{gs}.

Transverse Field Mobility Results

The transverse field dependent mobility models can account for high gate biases, but as seen in Figure 2-2, the mobility reduction differs for the various cases as V_{gs} is increased. It is apparent that some characterization would be required to obtain better agreement between the models themselves, and also with measured experimental data.

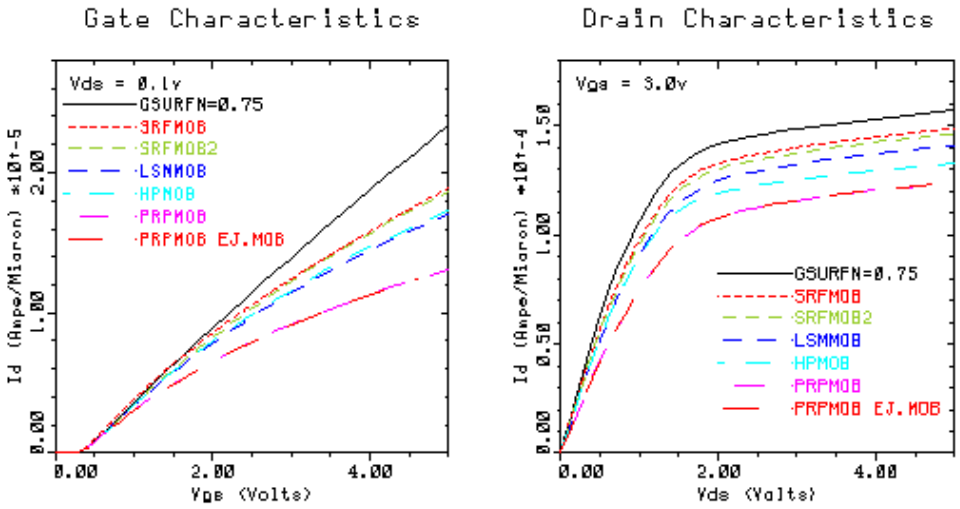


Figure 2-2 N-channel MOSFET simulation with seven mobility choices

EJ.MOBIL Results

The final two cases illustrated in this comparison use the same mobility model (**PRPMOB**), however, they differ in the way that the electric field calculations are done. In case 7, the parameter **EJ.MOBIL** is selected to force the use of the electric field components parallel and perpendicular to current flow as opposed to the electric field components parallel and perpendicular to the side of an element.

Figure 2-2 shows that the resulting gate and drain curves show no visible difference from the case when **EJ.MOBIL** is not selected (the curves for these two cases coincide in Figure 2-2). This can be explained by the fact that the current flow for this device occurs primarily at the interface and is directed along the edges of elements that make up the interface. Therefore, in this case, the components of electric field parallel and perpendicular to the sides of the interface elements are almost identical to the components of field parallel and perpendicular to the direction of current. This accounts for the nearly identical results.

Comparison of CPU Time

Table 2-7 compares the relative CPU time required to simulate the gate and drain curves shown in Figure 2-2. As the table indicates, selecting **EJ.MOBIL** can be quite expensive in terms of CPU time. Although using **EJ.MOBIL** results in more accurate solutions, as the above example illustrates, it often is not necessary.

Table 2-7 Comparison of Relative CPU Times for Various Options

Model	Gate Characteristics	Drain Characteristics
GSURFN=.75	1.00	1.00
SRFMOB	1.00	1.02
SRFMOB2	1.05	1.02
LSMMOB	1.51	1.53
HPMOB	1.30	1.22
PRPMOB	1.17	1.16

Table 2-7 Comparison of Relative CPU Times for Various Options

Model	Gate Characteristics	Drain Characteristics
PRPMOB EJ.MOBIL	1.52	1.79

EJ.MOBIL and Subsurface Current Flow

Finally, an example is presented that illustrates a case where selection of **EJ.MOBIL** does affect device behavior. The device structure used for this example is nearly identical to the N-channel MOSFET structure considered previously. The difference is that a much finer simulation mesh is used near the oxide-silicon interface so that deviations of the current flow from the surface may be studied. Such deviations are known to occur near the drain region of the channel when the drain is biased beyond the onset of saturation.

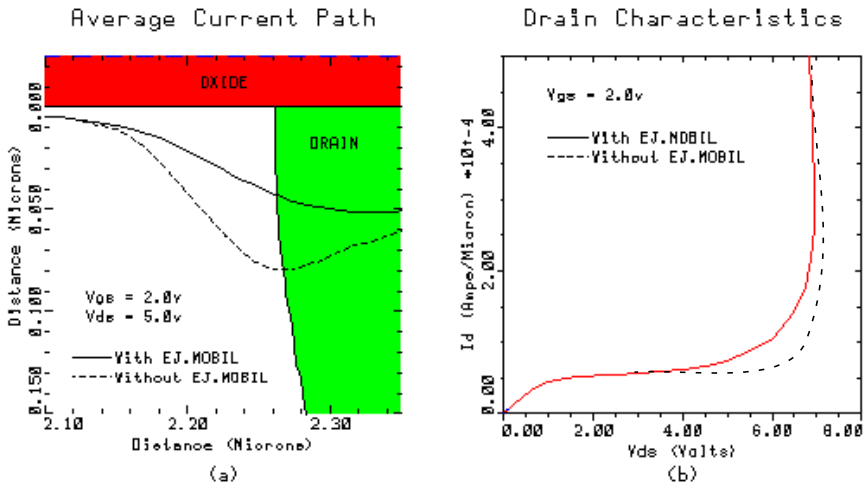


Figure 2-3 N-channel MOSFET with and without **EJ.MOBIL** selected

Figure 2-3a shows the average current path (50% flowline) for the cases when **EJ.MOBIL** is selected (solid line) and when it is not selected (dashed line). The mobility models selected for these simulations include **CONMOB**, **FLDMOB**, and **PRPMOB**.

As explained in Reference [27], the vertical deflection of the current path is exaggerated when field components parallel and perpendicular to the sides of elements are used in the mobility calculations. The resulting drain current for the two cases, including the effects of impact ionization, are shown in Figure 2-3b. The exaggerated vertical deflection of the current path when **EJ.MOBIL** is not selected causes the current to flow in a region of lower electric field. This results in a less accurate (higher) simulated breakdown voltage for the device as compared to the case when **EJ.MOBIL** is selected.

Electron-Hole Scattering

If **EHSCAT** is specified on the **MODELS** statement, electron-hole scattering is included in the electron and hole current density equations. Electron-hole scatter-

ing is important when there are large numbers of electrons and holes present simultaneously, as in power bipolar devices. When electron-hole scattering is included, the electron and hole continuity equations become (Reference [77]):

$$\vec{J}_n = -q\mu_n n\vec{\nabla}\phi_n + \sigma\vec{\nabla}\phi_p \qquad \text{Equation 2-196}$$

$$\vec{J}_p = -q\mu_p p\vec{\nabla}\phi_p + \sigma\vec{\nabla}\phi_n \qquad \text{Equation 2-197}$$

The terms μ and σ due to electron-hole scattering are given by the follow equations:

$$\mu_n = \mu_n^0 \frac{1 + n\mu_p^0 M}{1 + (p\mu_n^0 + n\mu_p^0)M} \qquad \text{Equation 2-198}$$

$$\mu_p = \mu_p^0 \frac{1 + p\mu_n^0 M}{1 + (p\mu_n^0 + n\mu_p^0)M} \qquad \text{Equation 2-199}$$

$$\sigma = qn\mu_n^0 \frac{p\mu_p^0 M}{1 + (p\mu_n^0 + n\mu_p^0)M} \qquad \text{Equation 2-200}$$

$$M = \mathbf{A.EHS}\frac{1 + \mathbf{B.EHS}^{\frac{n+p}{2}}}{1 + \mathbf{C.EHS}^{\frac{n+p}{2}}} \qquad \text{Equation 2-201}$$

In the above equations, **A.EHS**, **B.EHS**, and **C.EHS** are user-specified parameters accessible through the **MATERIAL** statement. μ_n^0 and μ_p^0 are the normal electron and hole mobilities, which include field, doping and temperature dependence. In the above equations, when M is small, the model reduces to the normal drift-diffusion case.

Note:

Use of electron-hole scattering with* CCSMOB *and* PHUMOB *is not recommended. These two models already contain mobility degradation due to electron-hole scattering. When anisotropic mobilities are used, the electron-hole scattering terms also become anisotropic due to their dependence on mobility.

Stress-Induced Mobility

Selecting **STRMOB** on the models statement causes mechanical stress effects in silicon regions to be included in the electron and hole mobilities. This model must be used in conjunction with the stress-induced bandgap model described in the **MODELS** statement. Due to the very different band structure of the conduction and valence bands, the effect of stress of the electron and hole mobilities is very differ-

ent. Using the model from Egley (Reference [97]), the electron mobility becomes anisotropic under stress and can be written as a diagonal tensor in the crystallographic coordinate system as:

$$\underline{\underline{\mu}}'_n = \mu_{n0}\begin{pmatrix} 1+\beta_1 & 0 & 0 \\ 0 & 1+\beta_2 & 0 \\ 0 & 0 & 1+\beta_3 \end{pmatrix} \qquad \text{Equation 2-202}$$

where μ_{n0} is the nominal, isotropic mobility without stress and

$$\beta_i = \left(\frac{1-\texttt{MLT.STR}}{1+2\texttt{MLT.STR}}\right)\left(\frac{\exp\left(-\frac{\Delta E_{Ci}}{kT}\right)}{\exp\left(-\frac{\Delta E_C}{kT}\right)} - 1\right) \qquad \text{Equation 2-203}$$

where:

- ΔE_{Ci} is the shift in the band edge of the ith ellipsoidal conduction minima from Equation 2-53
- ΔE_C is the net effective shift in the conduction band from Equation 2-51
- **MLT.STR**=m_L/m_T is the ratio of the longitudinal and transverse effective masses.

During device simulation, the mobility tensor $\underline{\underline{\mu}}'_n$ must be transformed to the device coordinate system using a similarity transformation:

$$\underline{\underline{\mu}}_n = \begin{pmatrix} \mu_{xx} & \mu_{xy} & \mu_{xz} \\ \mu_{xy} & \mu_{yy} & \mu_{yz} \\ \mu_{xz} & \mu_{yz} & \mu_{zz} \end{pmatrix} = A\underline{\underline{\mu}}'_n A^{-1} \qquad \text{Equation 2-204}$$

where A is the transformation from the crystallographic coordinate system to device coordinate system as determined by the substrate orientation. For an arbitrary substrate orientation, the mobility tensor in the device coordinate system contains, in general, off-diagonal components. Similar to the inclusion of anisotropic transport in the Anisotropic Material Advance Application Module, however, these off-diagonal components are neglected in the discretization of the continuity equations. Under certain circumstances, however, the transformed mobility tensor in Equation 2-204 remains diagonal and is thus treated exactly. This occurs, for example, for a device under plane strain with a substrate orientation of <100> and an x-axis orientation of <110>.

In the model from Egley, the hole mobility remains isotropic and is given by:

Equation 2-205

$$\mu_p = \mu_{p0}\left\{1 + (\mathbf{MUL0.STR} - 1)\frac{x_0}{1 + x_0}\left[\exp\left(\frac{\Delta E_{Vl} - \Delta E_{Vh}}{kT}\right) - 1\right]\right\}$$

where:

- μ_{p0} is the nominal hole mobility without stress
- $x_0 = (\mathbf{ML0}/\mathbf{MH0})^{3/2}$, ΔE_{Vl} and ΔE_{Vh} are the shifts in the light and heavy hole maxima from Equation 2-54
- $\mathbf{MUL0.STR} = \mu_{pl}/\mu_{p0}$ is the ratio of the unstressed light hole mobility to the total hole mobility.

The default value of **MUL0.STR** is chosen as 2.79 from the hole mobility model of Ottaviani (Reference [99]). The coefficients **MLT.STR** and **MUL0.STR** can be changed on the **MOBILITY** statement.

Boundary Conditions

Medici supports four types of basic boundary conditions:

1. Ohmic contacts
2. Schottky contacts
3. Contacts to Insulators
4. Neumann (reflective) boundaries

In addition to these, three auxiliary boundary conditions are also included, which are useful for some types of applications.

1. To save grid space, it is beneficial to include lumped resistances, capacitances, and inductances between applied biases and semiconductor device contacts.
2. For devices such as SCRs, where current is a multivalued function of applied voltage, a current boundary condition can be used.
3. To account for the finite resistivity of contacts to the semiconductor, a true distributed contact resistance can be specified.

Both the basic and auxiliary boundary conditions are described below.

Ohmic Contacts

Ohmic contacts are implemented as simple Dirichlet boundary conditions, where the surface potential and electron and hole concentrations (ψ_s, n_s, p_s) are fixed. The minority and majority carrier quasi-Fermi potentials are equal and are set to the applied bias of that electrode, i.e. $\phi_n = \phi_p = V_{applied}$. The potential, ψ_s, is fixed at a value consistent with zero space charge, i.e.

$$n_s + N_A^- = p_s + N_D^+$$ Equation 2-206

Substituting Equations 2-34 and 2-35 for n_s and p_s, Equation 2-206 can be solved for ψ_s and hence n_s and p_s, since ϕ_n and ϕ_p are known. If Boltzmann statistics are used, substitution of Equations 2-212 and 2-213 into Equation 2-206 yields

$$n_s = \frac{1}{2}\left[(N_D^+ - N_A^-) + \sqrt{\left(N_D^+ - N_A^-\right)^2 + 4n_{ie}^2}\right]$$ Equation 2-207

$$p_s = \frac{n_{ie}^2}{n_s}$$ Equation 2-208

where

$$\psi_s = \phi_n + \frac{kT}{q}\ln\left(\frac{n_s}{n_{ie}}\right) = \phi_p - \frac{kT}{q}\ln\left(\frac{p_s}{n_{ie}}\right)$$ Equation 2-209

Schottky Contacts

Schottky contacts to the semiconductor are defined by a work function of the electrode metal and an optional surface recombination velocity. The surface potential at a Schottky contact is defined by

$$\psi_s = \chi_{semi} + \frac{E_g}{2q} + \frac{kT}{2q}\ln\left(\frac{N_C}{N_V}\right) - \mathbf{WORKFUNC} + V_{applied}$$ Equation 2-210

The work function of the metal, **WORKFUNC**, can be specified on the **CONTACT** statement. In the above expression, χ_{semi} is the electron affinity of the semiconductor which can be specified with the **AFFINITY** parameter on the **MATERIAL** statement.

Imposing Finite Surface Recombination Velocities

Finite surface recombination velocities can be imposed by specifying the **SURF.REC** parameter on the **CONTACT** statement. In this case, ϕ_n and ϕ_p are no longer equal to $V_{applied}$ and instead are defined by current boundary conditions at the surface (Reference [28])

$$J_{sn} = q\ v_{sn}\ (n_s - n_{eq})$$ Equation 2-211

$$J_{sp} = q\ v_{sp}\ (p_s - p_{eq})$$ Equation 2-212

where

- J_{sn} and J_{sp} are the electron and hole current densities at the contact
- n_s and p_s are the actual surface electron and hole concentrations

- n_{eq} and p_{eq} are the equilibrium electron and hole concentrations assuming infinite surface recombination velocities ($\phi_n = \phi_p = V_{applied}$).

The surface recombination velocities for electrons and holes, v_{sn} and v_{sp}, are calculated by default from the expressions

$$v_{sn} = \frac{\mathbf{ARICHN}\ T^2}{q\ N_C} \qquad \text{Equation 2-213}$$

$$v_{sp} = \frac{\mathbf{ARICHP}\ T^2}{q\ N_V} \qquad \text{Equation 2-214}$$

where **ARICHN** and **ARICHP** are the effective Richardson constants for electrons and holes which take into account quantum mechanical reflection and tunneling, and can be specified on the **MATERIAL** statement. Alternatively, specific values can be selected for v_{sn} and v_{sp} using the **VSURFN** and **VSURFP** parameters on the **CONTACT** statement.

Barrier Lowering

The Schottky model can also account for field-dependent barrier-lowering mechanisms, arising from image-forces and possible static dipole layers at the metal-semiconductor interface. In Medici, the barrier height is defined as (Reference [4])

$$\phi_{bn} = \mathbf{WORKFUNC} - \chi_{semi} \qquad \text{Equation 2-215}$$

$$\phi_{bp} = \frac{E_g}{q} + \chi_{semi} - \mathbf{WORKFUNC} \qquad \text{Equation 2-216}$$

Barrier lowering can be requested with the **BARRIERL** parameter on the **CONTACT** statement. The amount by which the barriers are lowered (Reference [29]) can be expressed as

$$\Delta\phi_b = \left[\frac{q}{4\pi\varepsilon_{semi}}\right]^{1/2} E^{1/2} + \mathbf{ALPHA}\ E \qquad \text{Equation 2-217}$$

where E is the magnitude of the electric field at the interface. Note that the term with the square root dependence on electric field corresponds to the image force, while the linear term corresponds to the dipole effect. The coefficient **ALPHA** can be specified on the **CONTACT** statement. Typical values for **ALPHA** can be found in Reference [29].

Barrier lowering had been previously implemented in an experimental version of the program (see Reference [30]). It has been reincorporated in Medici in a slightly different manner. The basic procedure is to solve Poisson's equation normally with the surface potential defined as in Equation 2-210. However, using the electric field consistent with the solved potentials, effective surface potentials are computed as

$$\psi_{s_{eff}} = \psi_s \pm \Delta\phi_b \qquad \text{Equation 2-218}$$

where the + is for electrons and the - for holes, respectively. The continuity equations are then solved using Equations 2-211 and 2-212 as boundary conditions, but with n_s, n_{eq}, p_s, and p_{eq} replaced with $n_{s_{eff}}$, $n_{eq_{eff}}$, $p_{s_{eff}}$, and $p_{eq_{eff}}$ which are computed using $\psi_{s_{eff}}$.

The physical interpretation of this approach is that the Poisson equation is solved consistently with the charge, but the electrons and holes see a combined Poisson and image-force potential. Note that the full barrier lowering term has been applied directly at the surface; in reality, the peak occurs a slight distance within the semiconductor.

In Medici, surface recombination is implemented on a element-by-element basis. That is, using the surface recombination velocity and geometrical data, a recombination component is calculated for each element to which an element of interest is connected. Using the electric field for each element, an adjusted recombination term can be computed if barrier lowering is to be incorporated. This is in contrast to Reference [30] where a single field value for the electrode node was used to compute a total recombination value. Also unlike Reference [30], barrier lowering can be used with any of the basic numerical solution procedures, i.e. Gummel or Newton.

Contacts to Insulators

These contacts generally have a work function, dictating a value for ψ_s similar to Equation 2-206. The electron and hole concentrations within the insulator and at the contact are forced to be zero, i.e. $n_s = p_s = 0$.

Neumann Boundaries

Along the outer (noncontacted) edges of devices to be simulated, homogeneous (reflecting) Neumann boundary conditions are imposed so that current only flows out of the device through the contacts.

Additionally, in the absence of surface charge along such edges, the normal component of the electric field goes to zero, i.e. $\hat{n} \cdot \vec{\nabla}\psi = 0$. In a similar fashion, current is not permitted to flow from the semiconductor into an insulating region.

At the interface between two different materials, the difference between the normal components of the respective electric displacements must be equal to any surface charge density, σ_s, present along the interface:

$$\hat{n} \cdot \varepsilon_1 \vec{\nabla}\psi_1 - \hat{n} \cdot \varepsilon_2 \vec{\nabla}\psi_2 = \sigma_s \qquad \text{Equation 2-219}$$

The possibilities for the surfaces charge density, σ_s, are described in the next section.

Interface Charge and Traps

The surface charge density used in Equation 2-219 may be composed of both fixed charge and interface trapped charge:

$$\sigma_s = \sigma_f + \sigma_{a,p} + \sigma_{a,n} + \sigma_{d,p} + \sigma_{d,n} \quad \text{Equation 2-220}$$

In this expression, σ_f is the fixed interface charge density and the remaining terms are charge densities due to the presence of fast interface states. These interface states can be charged or neutral depending on whether they hold a carrier or not. Fast interface states are allowed only in the semiconductor forbidden bandgap and are classified as either donor or acceptor of electrons or holes:

- $\sigma_{a,p}$ charge density of hole acceptor states (positively charged above the hole quasi-Fermi level, neutral below)
- $\sigma_{a,n}$ charge density of electron acceptor states (neutral above the electron quasi-Fermi negatively charged below)
- $\sigma_{d,p}$ charge density of hole donor states (neutral above the hole quasi-Fermi level, negatively charged below)
- $\sigma_{d,n}$ charge density of electron donor states (positively charged above the electron quasi-Fermi level, neutral below)

Expressions for the various charge density terms are given as follows:

$$\sigma_f = q \cdot \mathbf{QF} \quad \text{Equation 2-221}$$

$$\sigma_{a,p} = q \cdot \mathbf{P.ACCEPT} \cdot (E_C - E_{Fp}) \quad \text{Equation 2-222}$$

$$\sigma_{a,n} = -\ q \cdot \mathbf{N.ACCEPT} \cdot (E_{Fn} - E_V) \quad \text{Equation 2-223}$$

$$\sigma_{d,p} = -\ q \cdot \mathbf{P.DONOR} \cdot (E_{Fp} - E_V) \quad \text{Equation 2-224}$$

$$\sigma_{d,n} = q \cdot \mathbf{N.DONOR} \cdot (E_C - E_{Fn}) \quad \text{Equation 2-225}$$

In the above expressions, **QF** is a number density (#/cm^2) and **P.ACCEPT**, **N.ACCEPT**, **P.DONOR**, **N.DONOR** are number densities per unit energy (#/cm^2-eV). These parameters can be specified on the **INTERFACE** statement.

Lumped Resistance, Capacitance, and Inductance Elements

The development of lumped elements capability addresses two needs:

- To reduce the number of grid points used to discretize device structures (to save CPU time)
- To perform transient simulation more accurately (see comments at the end of "Transient Device Simulation" on page 2-87).

Note:

Medici ***requires that the Newton method be specified when using resistance, capacitance, and inductive elements.***

Simulation of an entire CMOS structure provides a typical case where a lumped resistance might be useful. Such a structure may contain a purely resistive section that could be tens or hundreds of microns away from the primary area of interest of possibly only 10 to 20 microns of the structure. If the whole structure were simulated, a large number of grid points (more than half) would be wasted to account for the purely resistive region of the device. Because CPU time is a superlinear function of the number of grid points for all numerical methods of interest (see Reference [31]), simulating such regions explicitly can prove costly. Another example is the simulation of the substrate of a MOSFET, where inclusion of grid points through to the back side of a wafer may be prohibitive. In both cases, a simple, lumped resistance might be substituted. Similar arguments hold true for capacitance.

Lumped Element Boundary Conditions

Lumped element boundary conditions are implemented by introducing extra unknowns, the voltage on the semiconductor contact (ϕ) and the current flowing in an optional inductance (I_l), to be solved for. These unknowns are defined by Kirchhoff equations:

Equation 2-226

$$\frac{V_{applied} - \phi}{R} + C\frac{d(V_{applied} - \phi)}{dt} + I_l - \sum_{i=1}^{N_b} (I_n + I_p + I_{disp})_i = 0$$

$$V_{applied} - \phi - L\frac{dI_l}{dt} = 0 \qquad \text{Equation 2-227}$$

where N_b is the number of boundary grid points associated with the electrode of interest. The first auxiliary equation, due to the currents inside the summation, has dependencies on the values of potential and carrier concentrations at the nodes on the electrode as well as all nodes directly adjacent to the electrode.

It is important to note the following:

- Temporal terms associated with the capacitor and inductor must be discretized in a manner consistent with the device equations (see "Transient Device Simulation" on page 2-87).

- In contrast to the distributed resistance to be described below, a lumped element contact has a single voltage (or potential, adjusted for possible doping nonuniformities) associated with the entire electrode.

Specification

Within Medici, values for lumped resistances, capacitances and inductance, are specified with the **RESISTAN**, **CAPACITA**, **INDUCTAN** parameters on the **CONTACT** statement.

Because auxiliary equations must be added to the system, a new symbolic matrix factorization must be performed after these boundary conditions are specified or changed. Medici detects the changes in the user-specification and takes care of this automatically. In addition, the Newton solution method must be used with lumped elements or contact resistance.

The specified values should be in units of:

- $\Omega - \mu m$ for resistance
- $F / \mu m$ for capacitance
- Henrys$-\mu m$ for inductance

Capacitance increases with device width (into the z plane), while conversely, resistance and inductance decrease. Except for the case of extremely large resistances, where the arrangement becomes similar to a pure current source (see below), no convergence degradation has been observed for a lumped element boundary in comparison to a simple ohmic contact.

The simulator should be used as much as possible to help calculate any resistance (or capacitance) components that might be included as lumped elements. For instance, in the case of the CMOS structure mentioned above, just the resistive portion of the structure may be simulated with ohmic contacts at either end. From the plot of terminal current (in A/μm) versus voltage, the resistance can be directly extracted from the slope.

CAUTION

Be careful to account for any three-dimensional effects (such as current spreading) before using such a resistance value in further simulations.

Current Boundary Conditions

One of the first applications of Medici was in the analysis of CMOS latch-up triggering (Reference [32]). The terminal current of an SCR type structure is a multivalued function of the applied voltage. This condition implies that for some voltage boundary conditions, depending on the initial guess, a numerical procedure may produce a solution in one of three distinct and stable states:

- A low current "off" state
- An intermediate region typified by negative resistance
- A high current "on" state

The condition of primary interest is the point at which $dV / dI = 0$, known as the trigger point, which is exceedingly difficult to obtain with a simple voltage boundary condition. It is nearly impossible to compute any solutions in the negative resistance regime using only voltage inputs.

Since voltage can be thought of as a single-valued function of the terminal current, a possible alternative to the problem described above would be to define a current boundary condition. Such a boundary condition has been implemented within Medici as an auxiliary equation with an additional unknown boundary potential. Like the lumped R/C case, a Kirchhoff equation is written at the electrode:

$$I_{source} - \sum_{i=1}^{N_b} (I_n + I_p + I_{disp})_i = 0 \qquad \text{Equation 2-228}$$

Unlike the lumped R/C case, I_{source} is constant and has no dependence on the boundary potential ϕ (the ϕ dependence is buried in the summation). Because of this weaker dependence on ϕ, the convergence of the nonlinear Newton iteration is affected, but not always for the worse.

Specification

To specify that current boundary conditions are to be used at an electrode, the parameter **CURRENT** should be specified on the **CONTACT** statement. The actual value of current to use at the electrode is specified when a solution is requested on the **SOLVE** statement.

Note:

Medici ***requires that the Newton method be specified when using current boundary conditions.***

In regions where dI / dV is small:

- The voltage boundary condition is preferable.

In operating regions where dI / dV is large:

- The current boundary condition may be preferable.

It is common for the negative resistance regime of an SCR to have a slope dI / dV very close to 0. Such behavior should be considered when using a current source to trace out an entire I-V curve; i.e., it might be preferable to switch back to a voltage source after passing the trigger point. Or the continuation method (see "Continuation Method" on page 2-66) can be used, which automatically selects the best boundary condition and bias step size.

Distributed Contact Resistance

For a contact with a lumped element or for a simple voltage boundary condition, a single potential is associated with the entire electrode. However, because contact materials have finite resistivities, the electrostatic potential is not truly uniform

along the metal-semiconductor interface. To account for this effect, a distributed contact resistance can be associated with any electrode in a Medici simulation.

The implementation of distributed contact resistance is as follows: Medici internally places a resistance R_i at each node *i* associated with the contact of interest. The value of each R_i is computed from the contact resistance, **CON.RESI** (in $\Omega - \text{cm}^2$, specified on the **CONTACT** statement) as

$$R_i = \texttt{CON.RESI} / d_{c,i} \qquad \text{Equation 2-229}$$

where $d_{c,i}$ is the length of the contact associated with node *i*. An auxiliary equation is then added for each electrode node, unlike the lumped element case where a single equation is added for the entire electrode. So for every node *i* that is part of the contact,

$$\frac{1}{R_i}\left[V_{applied} - \left(\psi_i \pm \frac{kT}{q}\ln\left(\frac{N}{n_i}\right)\right)\right] - (I_n + I_p + I_{disp})_i = 0 \qquad \text{Equation 2-230}$$

Extra equations added to the system are strictly local in nature. Only the current at node *i* is included in Equation 2-230 as opposed to the summation over the contact in Equation 2-226, so that there is no direct coupling between nonadjacent electrode nodes and neighbors. Distributed contact resistance is simpler to implement numerically than are the lumped element or current boundary conditions.

An example illustrating the effects of specifying contact resistance for an electrode can be found in Chapter 6.

Majority Carrier Contact

One-dimensional simulation of bipolar devices requires a majority carrier contact to set the majority carrier Fermi level at the base contact. This is established by the standard contact. The standard contact also sets the minority carrier Fermi level. It is this level that fixes the concentration of minority carriers to the equilibrium value, with the result that no minority carriers can cross the base. To allow the one-dimensional simulation of bipolar devices (using a single line of elements), a majority carrier contact was created in Medici. This contact is selected by specifying **MAJORITY** on the corresponding **ELECTRODE** statement.

The majority carrier contact is implemented by injecting a majority carrier current I_m at the nodes of the electrode. The majority carrier current is calculated from $I_m = G(\phi_n - V_a)$ for N-type material and $I_m = G(\phi_p - V_a)$ for P-type material. The conductivity G is calculated by the program and corresponds to the resistivity of 0.1 micron of silicon. V_a is the voltage applied to the contact.

Numerical Methods

Six partial differential equations (PDEs) describe the bulk behavior of semiconductor devices:

- Poisson's equation (Equation 2-1 on page 2-2) governs the electrostatic potential.
- Continuity equations for electrons and holes (Equations 2-2 and 2-3 on page 2-2) govern the carrier concentrations.
- Carrier energy balance equations for electrons and holes (Equations 2-322 and 2-323 on page 2-114) govern the carrier temperatures.
- The lattice heat equation (Equation 2-355 on page 2-129) governs the lattice temperature.

These differential equations are discretized in a simulation grid, as described in "Discretization" on page 2-59. The resulting set of algebraic equations is coupled and nonlinear. Consequently the equations cannot be solved directly in one step. Instead, starting from an initial guess, the equations must be solved by a nonlinear iteration method.

The various solution methods are detailed in "Nonlinear System Solutions" on page 2-59 through "Coupled Solutions (Newton's Method)" on page 2-63. Choice of initial guess is explained in "Initial Guesses" on page 2-61.

Relevant Statements

The solution method and the number of carriers to be solved for are specified at symbolic time on the **SYMBOLIC** statement. The program also generates a map of the matrix at symbolic time. The various parameters, acceleration factors and iteration limits are specified later on the **METHOD** statement. If defaults are not given, they are also chosen at this time.

Selecting Equations to Solve

Given a particular device and range of operation, no single solution method is optimal in all cases. Several possibilities are shown in the following:

- At zero bias, a Poisson solution alone is sufficient.
- For MOSFET I-V characteristics, only one carrier need be solved for.
- In bipolar and MOSFET breakdown simulations, both carriers are needed.
- For simulating hot-carrier effects in small geometry devices where the electric field changes rapidly, carrier energy balance equations may be added.
- When device heating effects are important, the lattice heat equation must be solved.

The equation to be solved is specified on the **SYMBOLIC** statement.

When zero or one carrier is to be solved, quasi-Fermi level needs to be set for the carrier(s) not solved for. See "Quasi-Fermi Level Adjustments" on page 2-63.

Discretization

To solve the device equations on a computer, they must be discretized on a simulation grid. The continuous functions of the PDEs are represented by vectors of function values at the nodes, and the differential operators are replaced by suitable difference operators. Instead of solving for six unknown functions, Medici solves for $6N$ unknown real numbers, where N is the number of grid points.

For further details on the discretization of the device equations, and particularly for a discussion of the impact of obtuse elements in the grid, see Price (Reference [34]). A discussion of the time-dependent discretization of the device equations can be found in "Transient Device Simulation" on page 2-87. Time discretization is completely automatic and no intervention is required.

Box Method

The key to discretizing the differential operators on a general grid is the box method (Reference [33]). Each equation is integrated over a small volume enclosing each node, yielding $6N$ nonlinear algebraic equations for the unknown variables. The integration equates the flux entering the volume with the sources and sinks inside it, so that conservation of relevant flux is built into the solution. The integrals involved are performed on an element-by-element basis, leading to a simple and elegant way of handling general surfaces and boundary conditions.

Carrier Flux Formulation

In the case of the continuity equation, the carrier fluxes must be evaluated with care. Classic finite difference formulas are modified as first demonstrated by Scharfetter and Gummel in 1969 (Reference [26]).

Nonlinear System Solutions

Newton's method with Gaussian elimination of the Jacobian is by far the most stable method of solution. For low current solutions, Gummel's method offers an attractive alternative to inverting the full Jacobian.

The discretization of the semiconductor device equations gives rise to a set of coupled nonlinear algebraic equations. These must be solved by a nonlinear iteration method. Two approaches are widely used:

- Decoupled solutions (Gummel's method)
- Coupled solutions (Newton's method)

Either approach involves solving several large linear systems of equations. The total number of equations in each system is on the order of 1-4 times the number of grid points, depending on the number of device equations being solved for.

When a carrier energy balance and/or lattice heat equations are also solved for, a block iteration approach can be specified to solve the equations self-consistently. In such an approach, up to four equations (three basic equations plus one of the advanced equations) can be specified in the Newton block, while the rest are Gummel block equations.

The total cost of a simulation is the product of the number of matrix solutions and the cost of each solution. The objective of the various methods detailed below is to minimize one component or the other of that cost.

Common Concepts

Several ideas are common to all methods of solving the equations. This section covers the following:

- Convergence rate
- Error norms
- Convergence criteria
- Error norms selection
- Linear solution options
- Initial guess

Convergence Rate

The nonlinear iteration usually converges either at a linear rate or at a quadratic rate. At a linear rate, the error decreases by about the same factor at each iteration. In a quadratic method, the error is approximately squared at each iteration, giving rise to rapid convergence.

Newton's method, which is quadratic, is more accurate than Gummel's method, which is linear in most cases.

Error Norms

The error remaining at each step can be measured in two ways:

- Right-hand-side norm (RHS norm)
- X norm ("x" at each step)

The right-hand-side norm (RHS norm) is the difference between the left and right hand sides of the equations (Equations 2-1 through 2-3, Equations 2-322 and 2-323, and Equation 2-355). Since this is the quantity to be reduced to zero, the RHS norm is the most natural measure of the error. It is measured in C / μm for the Poisson equations, in A / μm for the continuity equations, and in W / μm for the advanced equations. At zero bias, there is always a residual current due to numerical error. The RHS norm may be interpreted as the size of this current.

The size of the updates to the device variables at each iteration may also be a measure of the error. At each step, the update is the unknown "x", and is called the X norm. Potential updates are measured in kT/q. Updates to the other solution variables are measured relative to the previous value at a particular point.

Convergence Criteria

A solution is considered converged and an iteration terminates when either the X norm or the RHS norm falls below a certain tolerance. For the X norm, the default error tolerances are:

- (**PX.TOLER**) = 10^{-5} kT/q for potential
- (**CX.TOLER**) = 10^{-5} relative change in concentration
- (**ETX.TOLE**) = 10^{-2} for carrier temperature

- (**LTX.TOLE**) = 10^{-3} for lattice temperature

For the RHS norm, the error default tolerances are:

- (**PR.TOLER**) = 10^{-26} *C* / μm for the Poisson equation
- (**CR.TOLER**) = 5 x 10^{-18} *A* /μm for the continuity equations
- (**ETR.TOLE**) = 10^{-18} *W* /μm for the energy balance equations
- (**LTR.TOLE**) =10^{-11} *W* /μm for the lattice heat equation

The RHS norm tolerances are the maximum acceptable divergences of relevant fluxes.

Error Norm Selection

Depending on the current level in the device, one error norm or another may be more suitable. For example, at low current levels, the exact value of the smallest minority concentrations has little influence on the current, and these small concentrations are almost indeterminate. The size of the relative updates used by the X norm may remain as large as 1% long after the current stops changing. Therefore, the RHS norm may be more suitable at low current levels.

As the current level increases so does numerical error in the continuity equations, and the absolute criterion on the RHS becomes harder to satisfy. The relative error, however, remains the same and the X norm is usually more appropriate.

By default, Medici uses a combination of the X norm and RHS norm for determining convergence. The program assumes a solution is converged when either the X norm or RHS norm tolerances are satisfied at every node in the device. Compared to using either the X norm or RHS norm alone, the combination of the two reduces the number of iterations required to obtain a solution without sacrificing accuracy.

Linear Solution Options

Given an outer nonlinear iteration, the resulting linearized system can be solved by either:

- A direct method (Gaussian elimination)

or

- An inner (linear) iteration method

In general, the direct method is more stable than the inner iteration method. However, as the number of grid points increases, the cost of the inner iteration increases less rapidly than does the direct method. The trade-off between stability and speed must be considered in choosing an appropriate method. Medici uses the direct method as the default.

Initial Guesses

Six types of initial guesses are used in Medici:

1. **INITIAL**
2. **PREVIOUS**
3. **LOCAL**

4. **PROJECT**
5. **P.LOCAL**
6. Post-regrid initial guess

INITIAL This guess imposes the charge neutrality assumption to obtain an initial guess for the first bias point, which is the starting point of any device simulation. Any later solution with applied bias must arrive at an initial guess by modifying the bias point of one or two previous solutions.

PREVIOUS The solution currently loaded is used as the initial guess, modified by setting the applied bias at the contacts.

LOCAL This guess uses the solution in memory, sets the applied bias, and changes the majority carrier Fermi potentials throughout heavily doped regions to be equal to the bias applied to that region. This procedure is effective in the context of a Gummel iteration, particularly in reverse bias. It is less effective for a Newton method.

PROJECT This guess uses an extrapolation of two previous solutions to the new bias assuming that equivalent bias steps are taken. It is particularly economical in generating I-V data.

P.LOCAL Previous-local (**P.LOCAL**) is a local guess used in the heavily doped regions connected to electrodes while all previous initial guess is used elsewhere. This is a useful alternative to **PREVIOUS** when the lattice heat equation is solved.

Post-Regrid Initial Guess Interpolation of a solution from a coarse mesh on to a new (regridded) mesh. It can be used to start the solution of the same bias point on the new grid. In spite of being an interpolation of an exact solution, this type of guess does not give rise to rapid convergence.

Initial Guess Selection Medici selects the projection method whenever two appropriate solutions are available, otherwise it defaults to a previous guess or initial guess if this is the first bias point. In some cases you may wish to override this choice. For instance, given two solutions of a MOSFET with 0V and 1V on the gate, it is unwise to extrapolate to 2V on the gate if the device has a threshold close to 1V. The extrapolation makes the incorrect assumption that the change in surface potential, when stepping from 1V to 2V, is the same as when stepping from 0V to 1V. In this case, using the parameter **PREVIOUS** would converge more rapidly. Similar situations can arise whenever a device changes from one operating regime to another within a single bias step.

Quasi-Fermi Level Adjustments

In order to simplify the analysis of particular devices, and save large amounts of computational time, it may be desirable to do the following:

- Solve Poisson's equation and one continuity equation
- Solve Poisson's equation alone

The following devices would profit from this simplification:

- Majority carrier devices such as MOSFETs, JFETs, and, MESFETs
- Devices where all junctions are not forward biased and currents may not be required, such as CCDs or capacitors.

If a carrier is not explicitly solved for, it still must have a value consistent with the electrostatic potential throughout the device. Medici chooses appropriate quasi-Fermi potentials for these carriers. The quasi-Fermi levels are chosen to be locally constant and change only at metallurgical junctions within the device, thereby contributing no current component.

For example, if holes are not being solved for, then in a p-type region, ϕ_p is set to the local bias voltage. In any n-type region, ϕ_p is set to the lowest applied (semiconductor) potential in the device, so that excess holes do not occur as minority carriers. Similarly, if electrons are not explicitly solved for, ϕ_n is set to the local bias in any n-type region and to the maximum (semiconductor) bias in the system for all p-type regions.

Coupled Solutions (Newton's Method)

Newton's method, with Gaussian elimination of the Jacobian, is by far the most stable method of solution. Unfortunately, it can be expensive for two-carrier simulations, both in time and memory. For low current solutions, the Gummel method offers an alternative to inverting the full Jacobian.

The basic algorithm is a generalization of the Newton-Raphson method for the root of a single equation. It can be expressed as follows.

Assuming Equations 2-1 through 2-3 are solved, they can be rewritten as

$$F_\psi(\psi, n, p) = 0 \qquad \text{Equation 2-231}$$

$$F_n(\psi, n, p) = 0 \qquad \text{Equation 2-232}$$

$$F_p(\psi, n, p) = 0 \qquad \text{Equation 2-233}$$

Given an initial guess for the unknowns at each node, ψ_0, n_0, and p_0 calculate a new update ($\Delta\psi\Delta n\Delta p$) by solving the linear system

$$\begin{bmatrix} \frac{\partial F_\psi}{\partial \psi} & \frac{\partial F_\psi}{\partial n} & \frac{\partial F_\psi}{\partial p} \\ \frac{\partial F_n}{\partial \psi} & \frac{\partial F_n}{\partial n} & \frac{\partial F_n}{\partial p} \\ \frac{\partial F_p}{\partial \psi} & \frac{\partial F_p}{\partial n} & \frac{\partial F_p}{\partial p} \end{bmatrix} \begin{bmatrix} \Delta\psi \\ \Delta n \\ \Delta p \end{bmatrix} = - \begin{bmatrix} F_\psi \\ F_n \\ F_p \end{bmatrix}$$

Equation 2-234

Jacobian Matrix in Newton

This Jacobian matrix has 3 times as many columns and rows (or twice as many when solving for only one carrier) as the matrix for a single variable. The disadvantage of Newton's method is that for large grids the memory and time necessary to invert the Jacobian matrix may be excessive.

Typically the 3*N* x 3*N* matrix takes 20 times longer to invert, and the 2*N* x 2*N* seven times longer, than an *N* x *N* matrix on the same grid. Thus, the overhead per iteration is high, but the number of iterations is low, typically between three and eight.

CAUTION

A large number of Newton iterations almost guarantees that the problem is not clearly posed. The most frequent cause is that the nonphysical boundary condition set up by a depletion layer extends beyond the bottom of the device but intersects a neutral contact.

Accelerating the Newton Iteration

The biggest acceleration of a Newton iteration is the Newton-Richardson method (**AUTONR**). This method re-factors the Jacobian matrix only when necessary. Frequently, the Jacobian matrix needs to be factorized twice per bias point using Newton-Richardson, as opposed to once per iteration.

The decision to re-factor is made on the basis of the decrease per step of the error norm (**NRCRITER**). When the norm of the error falls by more than a certain criterion, the Jacobian is considered sufficiently accurate and refactorization is not needed.

The default criterion is set at 0.1. Adjustments can be made by the following:

- Downwards to increase stability
- Upwards to increase speed.

Note:

NRCRITER ***should not be increased above 0.5, to preserve the stability of the Newton iteration in high level injection.***

When to Choose Full Newton

Full Newton is the method of choice for solving the following:

- One-carrier problems after turn-on

The solution time is typically a factor of three below the corresponding Gummel time.

- Two-carrier simulations

  For this use, full Newton becomes expensive both in time and memory for increasingly complicated device structures. Gummel's method, however, becomes increasingly slow as the power level increases. It ceases to converge for on-state bipolar problems.

- When using "continuation" (see "Continuation Method" on page 2-66)

Newton Damping

There are two methods of damping the nonlinear iteration process when using the full Newton method.

- Use the **N.DVLIM** parameter on the **METHOD** statement to make the change in potential between Newton iterations to a fixed limit. Careful use of this parameter can greatly improve convergence when calculating an initial solution or when taking large bias steps. A value of about one third (or less) of the bias step is best.
- Specify **N.DAMP** on the **METHOD** statement. This method multiplies the updates (for all equations) by a factor $t\ damp < 1$, which is determined by how rapidly the RHS error norms are decreasing (a rapidly decreasing RHS norm gives $t\ damp \approx 1$). Given a reasonable initial guess, this method is usually capable of finding the solution. It is similar to that suggested by Rose and Bank (Reference [35]).

Refer to the **METHOD** statement for more details.

Decoupled Solutions (Gummel's Method)

In Gummel's method, the equations are solved sequentially.

1. The Poisson equation is solved assuming fixed quasi-Fermi potentials; since the Poisson equation is nonlinear it is itself solved by an inner Newton loop.
2. The new potential is substituted into the continuity equations, which are linear and can be solved directly to obtain carrier concentrations.
3. The new potential and carrier concentrations are substituted in the advanced equations to be solved (if any). Since these are nonlinear equations, they are also solved by an inner Newton loop.

When the convergence criteria for all the equations are met, the loop is terminated. At each state only one equation is being solved, so the matrix has N rows and N columns regardless of the number of equations being solved. This method is a decoupled method; one set of variables is held fixed while another set is solved for.

When to Use Gummel

The success of the method depends on the degree of coupling between the equations. The most important coupling is the drift term of carrier current, which is directly related to the Poisson solution. Whenever drift terms are unimportant, for instance in isolation structures, Gummel's method is suitable. When the current is drift-dominated, for example in a pure resistive structure, convergence is slow.

Accelerating Gummel

Solution of Poisson's equation can be sped up to maximize the speed of the Gummel algorithm by

- Using the ICCG for the linear matrix solution
- Using damping schemes for Poisson update

Using ICCG

Incomplete Cholesky Conjugate Gradients (ICCG) is chosen in Medici to solve symmetric matrices iteratively. With ICCG selected, the expensive Gaussian elimination step is no longer needed in the inner Newton loop of Poisson's equation, thereby speeding up the solution process. In particular, with a Poisson-only solution, no Gaussian elimination is necessary in the entire Gummel cycle. The ICCG method is recommended whenever the usual Gummel algorithm is used, particularly for large grids.

This is the fastest possible solution mode with Medici for capacitance analysis or zero bias solutions. It can sometimes be used to provide initial guesses for other methods.

ICCG Parameters

Parameters **LU1CRIT** and **LU2CRIT** govern the behavior of the iteration relating to the termination criterion. The defaults are chosen as "safe" numbers that maintain the quadratic convergence in the Newton inner loop (Reference [35]). More speed can be obtained by loosening the bounds.

Poisson Damping

It is usually necessary to damp numerical ringing in the Poisson iteration when the biases applied to electrodes are abruptly changed by large steps (more than 1V). The simplest mechanism is to limit the maximum voltage change per iteration inside the device (**DVLIMIT**). A bound that is too tight slows convergence, while too large a value can allow overflow. The default value of **DVLIMIT** is 1.0 V unless **^DAMPED** is specified, in which case it is 0.1 V. For faster but slightly more risky simulations, larger values are possible. The speed advantage is usually significant.

A more sophisticated mechanism (**DAMPED**) is the Newton damping method (Reference [35]). This damping scheme frequently speeds convergence by rejecting updates which would cause the error norm to increase. The default parameters controlling the damping are usually satisfactory.

Continuation Method

Medici contains a powerful continuation method for the automatic tracing of I-V curves. This method automatically selects the bias step and boundary conditions appropriately for the bias conditions. The continuation method is activated by specifying **CONTINUE** on the **SOLVE** statement.

CAUTION

When using the continuation method

- **A fully coupled solution method must be used (two-carrier Newton with coupled energy balance or lattice heat equation, if applicable).**
- **Only semiconductor electrodes may be selected.**

Continuation Steps

The operation of the method can best be understood by referring to Figure 2-4 in conjunction with the steps below.

1. First the program alters the boundary conditions by connecting a resistor in series with the user-specified terminal. The value of this resistor is calculated by the program and assumes different values along the I-V curve.
2. Starting at point P, the program calculates the derivative of the current I with respect to the voltage V, giving the slope of the tangent line at point P.
3. The voltage and current are then projected along the tangent line to point Q.
   a. The length of the projection is selected automatically by the program based on a the local truncation error at point P. In this way the program takes large steps in flat areas of the I-V curve and shorter steps in regions where the curvature is greatest.
   b. The initial length of the projection can be controlled by the **C.TOLER** parameter on the **SOLVE** statement. The program automatically adjusts the tolerance target (equal to **C.TOLER,** initially), depending on how easily the previous step has been solved.

Note:

The length of the first step must be specified using the* C.VSTEP *parameter on the* SOLVE *statement.

4. From point Q, the load line is computed as perpendicular to the tangent and passing through point Q.
5. The resistance R and applied voltage Va are calculated to give the desired load line. Note that the resistance R can be less than zero if the dynamic resistance of the device is less than zero.
6. The program then solves the system of equations and arrives at point S. Point S then replaces point P, and the process is repeated.

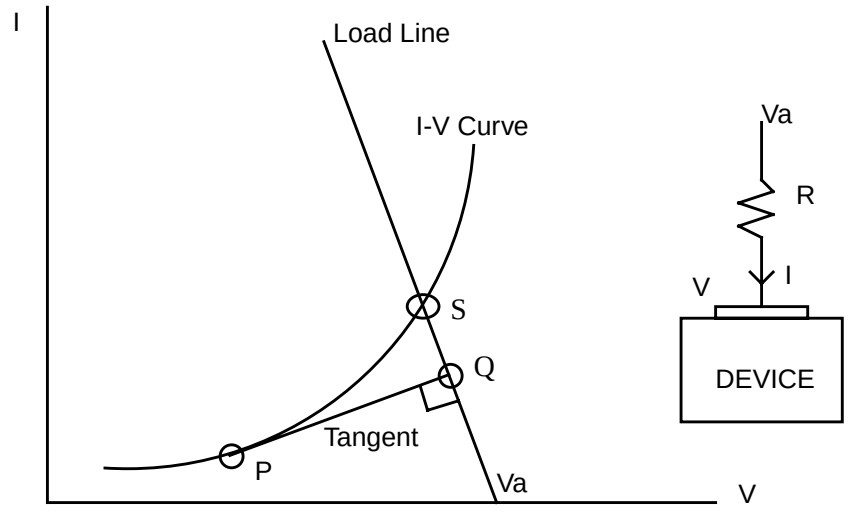


Figure 2-4 Continuation method used by Medici

When the I-V curve is rising steeply, the load line is nearly horizontal and R is very large. This case is similar to a current boundary condition (which corresponds to $R = \infty$). On the other hand, if the I-V curve is nearly horizontal, the load line is vertical and R is small. This case corresponds to a voltage boundary condition (R = 0).

The simulator continues along the I-V curve until one of the four termination criteria is met, these being:

$$V > \mathbf{C.VMAX};\quad V < \mathbf{C.VMIN};\quad I > \mathbf{C.IMAX};\quad I < \mathbf{C.IMAN}$$

These parameters are also specified on the **SOLVE** statement.

Parameter C.DVMAX

C.DVMAX specifies the maximum potential update in thermal voltage units. If the potential update exceeds this value, the program executes the following steps:

1. Bias point terminates
2. Bias step is reduced
3. Program tries again

This limit on the potential update is used to trap bias points not likely to converge. It allows the program to stop after one or two Newton iterations rather than 20 to discover that the bias point will not converge. Since projection is used, the potential updates should be small, and setting **C.DVMAX** to 50 kT / q (the default value) is a good choice.

Linear Matrix Solution

The following matrix solvers are available:

- Direct method (Gaussian Elimination)
- Incomplete Cholesky conjugate gradient method (ICCG)
- Incomplete LU conjugate gradient squared (ILUCGS) method

There advantages and disadvantages to each method. These are discussed below. For a discussion of the Jacobian matrix, see "Jacobian Matrix" on page 2-69.

Overview

Direct method—This method has the advantage of finding the answer in a predetermined number of operations without iterations. The direct method always finds the correct solution to the linear problem. The disadvantage of the direct method is that for large problems, the memory and CPU requirements quickly become intractable, particularly with the two-carrier Newton method.

ICCG—This is an iterative method and has the advantage of requiring very little memory. Typically, ICCG has good convergence properties and systems with tens of thousands of equations can be solved in less than 50 iterations. The main disadvantage of ICCG is that it can only be used with a symmetric matrix. This limits its use to solving the Poisson equation while using Gummel's method. Because the Jacobian matrices are asymmetric, ICCG cannot be used with Newton's method or to solve the continuity equations in Gummel's method.

ILUCGS—To solve the large asymmetric matrices that result from device simulation, the ILUCGS method has been recently developed. ILUCGS is an iterative method which, under most conditions, converges rapidly to an accurate solution, and requires manageable amounts of memory.

Jacobian Matrix

An essential part of the Newton or Gummel iteration is the solution of the linear problem $Jx = b$ where

- J is the Jacobian matrix
- x is the update vector $[\Delta\psi, \Delta n, \Delta p]^T$ for a two-carrier Newton solution
- b is the right hand side vector (RHS), $-[F\psi, Fn, Fp]^T$

Matrix Density

The density of the Jacobian matrix is influenced by parameter **STRIP**. When **STRIP** is true, the coupling coefficients along the hypotenuse of the right triangles are removed. This results in a more compact matrix and reduced matrix solution time. When impact ionization or perpendicular field mobility models are used, **STRIP** default to false, otherwise it is true.

Matrix Structure

For a pure rectangular grid (**STRIP** is true), J is a sparse matrix with five rough bands and zeros between the bands. The bands result from the coupling between nodes in the X and Y directions:

- Two bands, c_i and e_i resulting from Y couplings
- Two bands, b_i and f_i, resulting from X couplings
- The matrix diagonal, d_i from coupling of each node to itself

$$J = \begin{bmatrix} d_1 & e_1 & & f_1 & & & & & \\ c_1 & d_2 & e_2 & & f_2 & & & & \\ & c_2 & \ddots & \ddots & & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \ddots & & \\ b_1 & & & \ddots & \ddots & \ddots & & \ddots & \\ & b_2 & & & \ddots & \ddots & \ddots & & \ddots \\ & & \ddots & & & \ddots & \ddots & \ddots & & f_j \\ & & & \ddots & & & \ddots & \ddots & \ddots & \\ & & & & \ddots & & & \ddots & \ddots & \\ & & & & & \ddots & & & & e_k \\ & & & & & & b_j & & c_k & d_n \end{bmatrix}$$

Equation 2-235

where

- The dimension of the matrix is $N = N_x$ x N_y, where N_x and N_y, are the number of grid points in the X and Y directions
- The separation between bands b and f and the diagonal is N_y
- The separation between bands c and e and the diagonal is 1

If Gummel's method is used, each element in the above matrix consists of a single number. With Newton's method and one-carrier, each element becomes a 2 x 2 submatrix. When Newton's method with two-carriers is used, each element is a 3 x 3 submatrix. The storage for the J matrix is approximately:

- $5N$ for Gummel's method
- $20N$ for one-carrier Newton
- $45N$ for two-carrier Newton.

If impact ionization, perpendicular field mobility models (**`STRIP`** is false), or nonrectangular grids are used, additional nodal couplings occur and the storage for J expands to a maximum of:

- $7N$ for Gummel's method
- $28N$ for one-carrier Newton
- $63N$ for two-carrier Newton.

While the storage for J can become large, at least the dependence on the number of nodes is linear.

Direct Method

In the direct method, Medici uses a form of Gaussian elimination known as the LU decomposition. By default, the minimum degree algorithm (**`MIN.DEGR`**) from the Yale Sparse Matrix Package (Reference [37]) is used to reorder the nodes to reduce the size of the factorized matrix. In the LU decomposition, the matrix J is factored into an upper triangular matrix U and a lower triangular matrix L. After factorization, the system $LU_x = b$ is solved in two steps:

$$Lz = b \qquad \text{Equation 2-236}$$

$$Ux = z \qquad \text{Equation 2-237}$$

Advantages

Since L and U are triangular, these operations (known as backsolves) are easy to perform and require little CPU time. The decomposition step, where L and U are calculated, takes the most effort.

The Newton-Richardson method takes advantage of the easy backsolve sequence and simply replaces x while using the same J or equivalently the same L and U for several Newton steps. If J does not change much from iteration to iteration, large savings in CPU time can be realized.

Disadvantages

Problems arise with the direct method because, during the factorization step, fill-in between the bands occurs, and the sparse structure of the matrix is destroyed (i.e. L and U become full matrices). In particular, all the elements between bands may become nonzero requiring that their values be stored.

ICCG Solver

ICCG uses the method of conjugate gradients to iteratively minimize the residual defined as $r = b - Jx$. If r is zero, then an exact solution has been found. It can be shown that the conjugate gradient method converges to the exact solution in at most N iterations if there are N equations in N unknowns. In practice the method converges to a usable solution in much less than N iterations if the matrix has many degenerate eigenvalues, i.e. the matrix is approximately an identity matrix.

Preconditioning and Approximate Factorization

The process of transforming a matrix J to an approximate identity matrix is called preconditioning. A common approach is to construct an approximate factorization LU of J such that $U^{-1}L^{-1}J = 1 + E$ where E is an error matrix which should be small. In particular, if the positions of nonzero elements in LU are the same as those in J, you have what has been called incomplete LU (ILU) decomposition.

Incomplete Cholesky Decomposition

The conjugate gradient method only applies to symmetric matrices and if the Cholesky LU decomposition is performed on a symmetric matrix, $U = L^T$ so $J = LL^T$. The incomplete Cholesky decomposition simply computes elements of L only where the elements of J are nonzero.

Iteration Sequence

Using the incomplete Cholesky decomposition, the iteration sequence for ICCG is:

Calculate L from J

$$r_0 = b - Jx$$
$$q_0 = p_0 = (LL^T)^{-1} r_0$$

Repeat on i

$$c_i = (r_i \cdot q_i)$$
$$t_i = Jp_i$$
$$a_i = \frac{c_i}{(p_i \cdot t_i)}$$
$$x_{i+1} = x_i + a_i p_i$$
$$r_{i+1} = r_i - a_i t_i$$
$$q_{i+1} = (LL^T)^{-1} r_{i+1}$$
$$b_1 = \frac{(r_{i+1} \cdot q_{i+1})}{c_1}$$
$$p_{i+1} = q_{i+1} + b_i p_i$$

Each iteration requires one set of backsolves and one multiplication of J by the vector p. For memory storage it is necessary to retain the original J matrix and L, which is one half the size of J, plus five vectors of length N.

The iteration process ends when $||r_i||$ is sufficiently small. In Medici, the iterations stop when

$$||r_i||_\infty < 0.1 \times \textbf{PX.TOLER}$$

or

$\|r_i\|_2 < \min(\mathbf{LU1CRIT}, \mathbf{LU2CRIT} \times \|b\|_2 / \|b'\|_2)$

where b' is the RHS vector from the last Poisson iteration.

Advantages

The advantage of this method is that compared to the direct method, it requires very little memory. Typically, ICCG has good convergence properties, and systems with tens of thousands of equations can be solved in less than 50 iterations.

Disadvantages

The main disadvantage of ICCG is that it can only be used with a symmetric matrix. This limits its use to solving the Poisson equation while using Gummel's method. Because the Jacobian matrices are asymmetric, ICCG cannot be used with Newton's method or to solve the continuity equations in Gummel's method.

ILUCGS Solver

The ILUCGS solver is very similar to the ICCG solver. Since the matrices are asymmetric, the conjugate gradient squared method and incomplete LU decomposition are used instead of the conjugate gradient method and Cholesky decomposition. In the incomplete LU method, elements of L and U are only computed where there are nonzero elements in J. The ILUCGS method is as follows:

Calculate L and U from J

$$x_0 = (LU)^{-1} b$$

$$r_0 = q_0 = p_0 = (LU)^{-1}(b - Jx_0)$$

Repeat on i

$$c_i = r_0 \cdot r_i$$

$$t_1 = (LU)^{-1} J q_i$$

$$a = \frac{c_i}{(r_0 \cdot t_i)}$$

$$s_{i+1} = p_i - a_i t_i$$

$$r_{i+1} = r_i - a_i (LU)^{-1} J (p_i + s_i)$$

$$x_{i+1} = x_i + a_i (p_i + s_{i+1})$$

$$b_i = \frac{(r_0 \cdot r_{i+1})}{c_i}$$

$$p_{i+1} = r_{i+1} + b_i s_{i+1}$$

$$q_{i+1} = p_{i+1} + b_i (s_{i+1} + b_i q_i)$$

Each iteration requires two sets of backsolves and one matrix multiplication involving J. For memory storage, it is necessary to retain the original J matrix and L and U, each of which is one half the size of J. In addition, seven vectors of length N are needed.

In the Medici program, the ILUCGS iterations terminate when

$$\|r_i\|_2^2 < \textbf{ILU.TOL}$$

and when

$$|1 - \|x_i\|_2 / \|x_{i-1}\|_2| < \textbf{ILU.XTOL}$$

During AC small-signal analysis, an additional error criterion is used to insure the required accuracy:

$$\|(x_i - x_{i-1})/x_i\| < \textbf{TOLERANC}$$

Normally this is the last error criterion to be satisfied. (**ILU.TOL** and **ILU.XTOL** are found on the **METHOD** statement and **TOLERANC** is found on the **SOLVE** statement and only applies to AC analysis.)

Advantages

ILUCGS is a robust iterative solver for asymmetric matrices encountered in semiconductor device simulation. It has been found to work well in high level injection problems such as latch-up and bipolar analysis. It may be used to solve large problems where the cost of using the direct method becomes excessive.

Newton Method Implications

With ILUCGS there is no longer an exact LU decomposition. This means that the backsolve is no longer available as a quick route to a solution when using the same Jacobian. As a result, the Newton-Richardson method loses its speed advantage and may harm the convergence of the Newton loop.

This occurs since the only CPU savings would be in the initial incomplete LU factorization which is no longer the dominant factor in the solution process. Due to these differences, the default for the **AUTONR** parameter on the **METHOD** card is FALSE when ILUCGS is used.

AC Implications

With ILUCGS, the Successive Over-Relaxation (SOR) method cannot be used in the AC small-signal analysis because the backsolve operation is missing. Instead, ILUCGS is used to solve the entire Jacobian matrix obtained from the complex AC system. Although it is slower than SOR at low frequencies, ILUCGS is able to converge at much higher frequencies. As a result, bipolar transistors and MOSFETs can be analyzed well beyond their cut-off frequencies.

Convergence Degradation

The ILUCGS method is robust in its convergence behavior. ILUCGS converges very well in all the examples found in the example chapters. However, some convergence problems have been encountered.

Convergence behavior begins to degrade for floating body problems (such as SOI MOSFETS) or devices with nearly isolated PN junctions (such as a diode $100\mu m$ long with junction area of $1\mu m^2$).

Because the condition number of the Jacobian matrix becomes very large in floating body problems, any numerical errors that occur during the solution process are greatly magnified.

From a physical standpoint, (in a floating body problem), the floating regions are loosely coupled to the electrodes so their potential becomes highly dependent on the junction currents. The junction currents are in turn exponentially dependent on the potential of the floating region. This makes it difficult for the program to find the correct potential for the floating regions since small errors are amplified.

Improving ILUCGS Convergence

If convergence difficulties are experienced, the following suggestions may be of use.

1. To give the initial one- or two-carrier Newton solution a better starting point, first perform a zero-carrier Gummel solution at the same bias.
2. Occasionally ILUCGS converges to an inaccurate solution. This happens particularly in floating body problems with impact ionization. What usually happens then is that the Newton iteration either converges to the wrong solution or does not converge at all. A quick way to spot the wrong solutions is to examine the terminal currents which should *always* sum to zero.

   The exception is very small currents (<le – 15/*A*μ*m*) where harmless numerical errors may prevent the currents from summing to zero. Problems of this type can usually be cured by tightening the ILUCGS tolerances. Tolerances as small as **ILU.TOL**=1e-13 and **ILU.XTOL**=1e-8 have been used on occasion.
3. The theoretical iteration requirement for ILUCGS has been estimated as $N_{eq}^{1/3}$ where N_{eq} is the number of equations to solve. Medici uses, **ILU.ITER** = *max*(35,$N_{eq}^{1/2}$), but sometimes this requirement is not enough. If ILUCGS fails to converge, increase the requirement limit.
4. Avoid solving with impact ionization at low current levels. At zero bias (or close to zero bias), no current should be flowing in a device. Specifying impact ionization when solving at such a bias results in magnification of random numerical noise, with subsequent convergence difficulties. Since impact ionization is physically insignificant at low biases, turn it off until current flow is high enough to avoid convergence difficulties. Alternatively, bias steps can be taken will cause current to flow as soon as possible.

Grid in Medici

This section describes grid in Medici. It contains the following:

- Grid allocation
- Coordinate systems
- Maximum number of nodes
- Grid specification
- Rectangular mesh specification

- Initial mesh specification
- Regrid
- Mesh Smoothing

Grid Allocation

The correct allocation of grid is a crucial issue in device simulation. The primary goal is to achieve accurate solutions with the least amount of simulation time. Some considerations include:

- The number of nodes in the grid (N_p) has a direct influence on the simulation time. The number of arithmetic operations necessary to achieve a solution is proportional to N_p^{α} where α usually varies between 1.5 and 2.
- Because the different parts of a device have very different electrical behavior, it is usually necessary to allocate fine grid in some regions and coarse grid in others.
- The importance of accurate representation of small device geometries. In order to model the carrier flows correctly, the grid must be a reasonable fit to the device shape. This consideration becomes increasingly important as smaller, more nonplanar devices are simulated.

Note:

In order to maintain the simulation time within reasonable bounds, as far as possible, do not allow fine grid to spill over into regions where it is unnecessary.

For the reasons stated above, Medici supports a general irregular grid structure. This permits the analysis of arbitrarily shaped devices, and allows the refinement of particular regions with minimum impact on others.

Coordinate Systems

The program also gives you a choice of the Cartesian or cylindrical coordinate systems. This choice is made on the **MESH** statement. Medici is a two-dimensional simulator, so regardless of the coordinate system used, simulations are performed in only two of the three space dimensions.

Cartesian Coordinates

When using the Cartesian system, simulations are performed in the *xy*-plane. The device behavior is assumed to be identical for all values of *z*. For this reason, terminal currents are expressed in units of Amps/micron (of depth in the *z*-direction).

The Cartesian system is the one used by most two-dimensional simulators and yields good results in devices such as MOSFETs where the channel width is greater than the channel length.

Cylindrical Coordinates

The cylindrical coordinate system in Medici models the radial and *z*-dependence. The device behavior is assumed identical for all values of the azimuthal angle. By analogy with the Cartesian system, it would be reasonable to express terminal currents in units of Amps/radian. Medici, however, assumes that the structure is

rotated completely around the *z*-axis (through an angle of 2π radians), so that the terminal currents are given in units of Amps.

When using cylindrical coordinates, the Cartesian *x*-coordinate becomes the cylindrical *r*-coordinate and the Cartesian *y*-coordinate becomes the cylindrical *z*-coordinate. The cylindrical coordinate system is useful for modeling devices or problems with circular cross-sections. Some examples where cylindrical coordinates can be used are:

- Single-event upset (see Chapter 7)
- Simulation of high field breakdown at the corners of junctions
- Simulation of other devices with radial symmetry.

Maximum Number of Nodes and Memory Use

Three versions of the Medici program are routinely provided when the program is distributed. These versions are identified by the number of nodes that are allowed for a fully coupled two-carrier solution.

- 3,200 nodes version
- 10,000 nodes version
- 20,000 nodes version

For problems where it is possible to use 3,200 nodes or less, the 3,200 node version of Medici should be used. The 3,200 node version requires significantly less virtual memory than the other version and runs more efficiently on most computing resources.

For simulations requiring more than 3,200 nodes, the 10,000 or 20,000 node versions must be used. The virtual memory requirements for each version are summarized in Table 2-8.

Memory Requirements

Because of different simulation technique memory requirements, the actual number of grid points allowed differs from the maximum number of grid points available for creating a mesh if you solve the lattice temperature equation in a coupled manner.The various cases are summarized in Table 2-8.

Table 2-8 Maximum Nodes Available and Virtual Memory Required for Medici

Medici Version ⇒	3200	10,000	20,000
Maximum nodes available for mesh	3200	10,000	20,000
Maximum nodes for 0-, 1-, or 2-carrier, reduced or all couplings	3200	10,000	20,000
Maximum nodes for 2-carrier, fully coupled with energy balance	1842	5,736	11,464
Maximum nodes for 2-carrier, fully coupled with heat equation	1842	5,736	11,464
Approximate virtual memory required (Mbytes)	46	120	262

Specifying the **MAXNODES** parameter on the **OPTION** statement provides information regarding the number of nodes available for various types of analysis.

Grid Specification

User-specification is the most difficult aspect of general grid structure. To minimize this effort, Medici provides a regridding mechanism that automatically refines an initial grid wherever key variables vary rapidly.

Grid Refining Restrictions

The grid can only be refined *between* solutions, not during the solution. This approach is preferred because the overhead of repeatedly recalculating grid geometry and symbolic factorizations during every solution is quite costly. The implementation used in Medici follows the proposals in Reference [38] closely.

Initial Grid Generation

The initial grid is generated by specifying a coarse rectangular mesh using input statements. The coarse mesh is usually refined using the regrid capability until it is fine enough to accurately represent the structure. It is also possible to completely generate a mesh without using the regrid capability by varying the spacing between lines of nodes and/or by distorting the initial rectangular mesh.

Rectangular Mesh Specification

A distorted rectangular mesh can be a very effective solution mesh in some cases. Several suggested configurations are:

- For a planar device or a long-channel MOSFET, a rectangular grid is the method of choice. The same set of fine grid lines that follow the channel can be diverted around the junctions to provide reasonable resolution through all active areas of the device.
- Large aspect rectangles can be used to minimize the amount of grid allocated; the resulting matrix has properties that help reduce solution time.
- A coarse rectangular grid is a suitable candidate for regridding, particularly if the device has a complicated doping profile.

Mesh Statements

Rectangular meshes are specified by a series of mesh statements, detailed in Chapter 3, "3.1 Device Structure Specification" on page 3-13. In order of appearance, the required input is:

- **MESH** statement
- **X.MESH** statements
- **Y.MESH** statements
- **ELIMINATE** statements (optional)
- **SPREAD** statements (optional)
- **BOUNDARY** statements (optional)
- **TSUPREM4** statements (optional)
- **REGION** statements
- **ELECTRODE** statements

Note:

The order in which statements appear is important. Changing the order will change the results.

Generally, a mesh is specified by the following steps:

1. The mesh begins as a set of (nonuniformly) spaced *x*- and *y*-lines comprising a simple rectangle (**X.MESH**, **Y.MESH**).
2. Mesh lines may be terminated inside the device, and redundant nodes removed from the grid (**ELIMINATE**).
3. The rectangle can be distorted to track nonplanar geometry or match the doping profile, although strongly nonplanar structures are difficult to treat in this way (**SPREAD**).
4. Material regions and electrodes can be specified as a union of (possibly distorted) rectangles, completing the mesh specification (**REGION**, **ELECTRODE**).

CAUTION

When a rectangular grid is distorted, a large number of obtuse triangles are unavoidably introduced. When you regrid a rectangular grid, large aspect ratio rectangles (>2.8) can also give rise to very obtuse triangles. (See "Mesh Smoothing" on page 2-84.)

Initial Mesh Specification

This section details the creation of the initial mesh.

Grid Spacings

You have complete control over the specification of the initial rectangular grid spacings that define the distance between adjacent nodes. The statements used to set grid spacings are as follows:

- The grid spacings in the horizontal direction are established with one or more **X.MESH** statements.
- The spacings in the vertical direction are established with one or more **Y.MESH** statements.

Horizontal Spacing

Each **X.MESH** statement defines a section having a width specified by the **WIDTH** parameter, or alternately, the ending point of the section may be defined by the **LOCATION** or **X.MAX** parameters.

Each subsequent **X.MESH** statement generates mesh spaces starting at the right edge of the last mesh space generated by the previous **X.MESH** statement. For the first **X.MESH** statement in the sequence, the left edge of the device structure may be specified using the **X.MIN** parameter.

The order of the **X.MESH** statements corresponds to the order of the sections in the direction of increasing horizontal coordinates from left to right across the device.

Vertical Spacing

Each **Y.MESH** statement defines a section having a depth specified by the **DEPTH** parameter or, alternatively, the ending point of the section may be defined by the **LOCATION** or **Y.MAX** parameters.

Each subsequent **Y.MESH** statement generates mesh spaces starting at the bottom edge of the last mesh space generated by the previous **Y.MESH** statement. For the first **Y.MESH** statement in the sequence, the top edge of the device structure may be specified using the **Y.MIN** parameter.The order of the **Y.MESH** statements corresponds to the order of the sections in the direction of increasing vertical coordinate from top to bottom through the device.

Mesh Spacing Parameters

The mesh spacings within the section are determined by one or more of the following parameters:

- **H1**—The spacing at the beginning of the section
- **H2**—The spacing at the end of the section
- **H3**—The largest spacing in the interior of the section
- **N.SPACES**—The number of spaces
- **NODE**—The node number at the end of the section
- **RATIO**—The ratio between the sizes of adjacent spaces within the section

Any number of individual mesh sections can be defined as long as the maximum number of mesh points is not exceeded. Medici allows two additional mesh spacing control parameters:

- **SPACING**—This parameter is used to specify the local grid spacing. It behaves in the same way as setting the value of **H2** for the present grid section and the value of **H1** for the next grid section to the value specified with **SPACING**.
- **MIN.SPAC**—This parameter specifies the minimum mesh spacing allowed.

Because mesh definition allows variations in mesh spacing, the discretization error is minimized. A constant ratio between the sizes of adjacent mesh spaces is maintained within a section. The resulting matrix also reduces solution time.

The size of the mesh spaces can change at a maximum rate with a minimum adverse effect on accuracy. With this technique it is quite possible to vary the mesh spacing many orders of magnitude within a single mesh section. For instance, with a ratio of 1.2 between the sizes of adjacent mesh spaces, the spacing size changes by almost four orders of magnitude in 50 spaces.

Specifying Mesh Within a Section

Several methods can be used to specify mesh spacing within a section. Mesh spacing sizes are uniform, monotonically increasing or decreasing, or increasing or

decreasing from both ends of the section, to a maximum or minimum in the interior of the section.

Uniform Mesh Spacing

For a uniform mesh spacing, only one of the parameters **H1**, **H2**, **N.SPACES**, or **NODE** should be specified. If **N.SPACES** is specified, the size of the mesh spaces is given by

$$h(i) = \frac{\mathbf{WIDTH}}{\mathbf{N.SPACES}} \qquad \text{Equation 2-238}$$

where $h(i)$ is the mesh space size at index i within the section. The index i varies from 1 at the beginning of the section to **N.SPACES** at the end of the section. If **NODE** is specified, the mesh spacing $h(i)$ is given by

$$h(i) = \frac{\mathbf{WIDTH}}{\mathbf{NODE} - N} \qquad \text{Equation 2-239}$$

where N is the node number at the start of the section. If **H1** or **H2** is specified, the number of mesh spaces is given by

$$\mathbf{N.SPACES} = \begin{cases} \dfrac{\mathbf{WIDTH}}{\mathbf{H1}} & \mathbf{H1} \text{ specified} \\ \dfrac{\mathbf{WIDTH}}{\mathbf{H2}} & \mathbf{H2} \text{ specified} \end{cases} \qquad \text{Equation 2-240}$$

If the resulting value of **N.SPACES** is integral, then **H1** or **H2** is used for the uniform size of the mesh spaces. Otherwise, **N.SPACES** is rounded to the nearest integer, **H1** and **H2** are taken to be equal, and the mesh spacing size is increased or decreased from both ends of the section toward the center.

Monotonically Increasing or Decreasing Mesh Spacing

For a monotonically increasing or decreasing mesh spacing size, two parameters should be specified from the set consisting of **H1**, **H2**, **N.SPACES**, and **RATIO**. The size of the mesh spaces is given by

$$h(i) = \mathbf{H1} \cdot \mathbf{RATIO}^{i-1} \qquad 1 \le i \le \mathbf{N.SPACES} \qquad \text{Equation 2-241}$$

where the values of **WIDTH**, **H1**, **H2**, **N.SPACES**, and **RATIO** are related by

$$\mathbf{WIDTH} = \begin{cases} \mathbf{H1} \cdot \mathbf{N.SPACES} & \mathbf{RATIO} = 1 \\ \mathbf{H1}\left(\dfrac{\mathbf{RATIO}^{\mathbf{N.SPACES}} - 1}{\mathbf{RATIO} - 1}\right) & \text{otherwise} \end{cases} \qquad \text{Equation 2-242}$$

$$\mathbf{H2} = \mathbf{H1} \cdot \mathbf{RATIO}^{\mathbf{N.SPACES-1}} \quad \mathbf{H1} \text{ and } \mathbf{RATIO} \text{ specified} \qquad \text{Equation 2-243}$$

$$\mathbf{H1} = \mathbf{H2} \cdot \mathbf{RATIO}^{\mathbf{N.SPACES-1}} \quad \mathbf{H2} \text{ and } \mathbf{RATIO} \text{ specified} \qquad \text{Equation 2-244}$$

If **N.SPACES** is not specified and its calculated value is not integral, an integral value is chosen for **N.SPACES** that yields mesh spacings as close as possible to the specified distribution.

Increasing or Decreasing Mesh Section Spacing

For a mesh spacing size that increases or decreases from both ends of the section, the parameters **H1** and **H2** should both be specified, along with one parameter from the set consisting of **H3**, **N.SPACES**, and **RATIO**. The size of the mesh spaces is given by

$$h(i) = \begin{cases} \mathbf{H1} \cdot \mathbf{RATIO}^{i-1} & i < n_1 \\ \mathbf{H2} \cdot \mathbf{RATIO}^{\mathbf{N.SPACES}-1} & i \geq n_1 \end{cases} \quad \text{Equation 2-245}$$

where n_1 is the index of the minimum or maximum spacing in the interior of the mesh section. The values of **WIDTH**, **H1**, **H2**, **H3**, **N.SPACES**, **RATIO**, and n_1 are related by

$$\mathbf{WIDTH} = \frac{\mathbf{H1} \cdot \mathbf{RATIO}^{n_1} - 1}{\mathbf{RATIO} - 1} + \frac{\mathbf{H2} \cdot \mathbf{RATIO}^{(\mathbf{N.SPACES} - n_1)} - 1}{\mathbf{RATIO} - 1} \quad \text{Equation 2-246}$$

$$\mathbf{H3} = \mathbf{H1} \cdot \mathbf{RATIO}^{(n_1 - 1)} \quad \text{Equation 2-247}$$

$$= \mathbf{H2} \cdot \mathbf{RATIO}^{(\mathbf{N.SPACES} - n_1)} \quad \text{Equation 2-248}$$

If the **N.SPACES** parameter is not specified and its calculated value is not integral, then an integral value is chosen for **N.SPACES** that yields mesh spacings as close as possible to the specified distribution.

TCAD Compatibility

The syntax on the **X.MESH** and **Y.MESH** statements has been set up to allow compatibility with input files generated for TMA PISCES-2B and early versions of Medici, as well as other *Avant!* TCAD products. For example, early versions of TMA PISCES-2B only allowed **LOCATION**, **NODE**, and **RATIO** to be used. TSUPREM-4 uses **LOCATION** and **SPACING**. Both methods of grid specification are compatible with Medici.

Regrid

The regrid facility refines parts of the mesh to satisfy some criterion. Using regrid, fine mesh can be allocated only where necessary, with automatic mesh grading occurring between high and low density regions. The criteria for refinement are based on physical, not geometrical grounds, so they do not need to be tailored to fit a particular structure.

Regrid Algorithm

The regrid algorithm is best understood by following the steps below in relation to Figure 2-5:

1. The regrid algorithm searches the initial grid for triangles satisfying the refinement criterion.

2. Each triangle found is subdivided into four congruent subtriangles, and the various grid quantities (such as doping, potential, carrier concentrations, and mobility) are interpolated onto the new nodes, using linear or logarithmic interpolation as appropriate for that quantity. The new triangles are on level-1, while the initial grid is on level-0.
3. After all level-0 triangles have been examined, the same procedure is applied again to the level-1 triangles, and any new subtriangles become level-2 triangles.
4. At each level the grid is checked for consistency and updated to avoid abrupt changes of size from one triangle to the next.

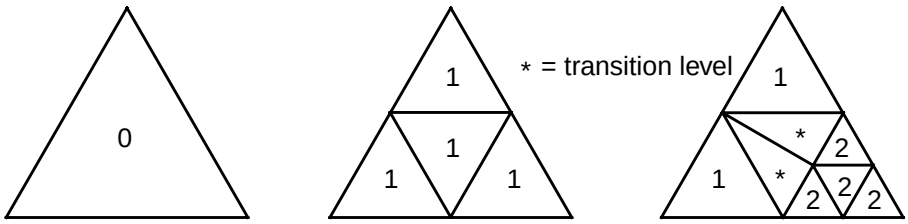


Figure 2-5 Levels of regrid in a triangular based mesh

Limiting the Refinement

Refinement continues until no triangle satisfies the criterion, or until a specified maximum level is reached. Of necessity, grids for semiconductor problems are considerably coarser than desirable, so the maximum level is usually the key factor in determining the size of the grid.

The final grid is not generated directly from the initial grid, but rather in several steps. At each step, the maximum level may be chosen to limit the refinement. The default action sets the maximum level equal to one more than the highest level in the existing mesh. However, in some cases it is necessary to specify the maximum level. For example, when a mesh has already been refined several times, and a coarse region is to be upgraded without regridding the finer regions, the maximum level should be set below the level of the finer regions.

Interpolation

If several levels of regrid are performed in immediate succession, the refinement decisions at the higher levels use interpolated data. The nonlinearity of semiconductor problems makes the use of interpolated data inadvisable; data used to refine the grid should be updated as soon as possible. Figure 2-6 illustrates the necessity for refinement.

In this case, a sharp bend in the potential contour is being refined so that all elements with steps of more than 0.1V across them are refined. With interpolated data, the whole interval would be refined as shown by the solid line in Figure 2-6.

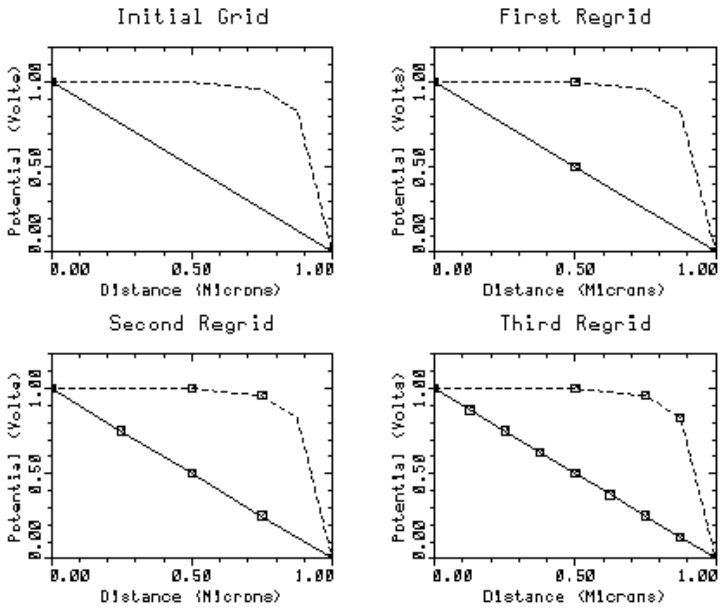


Figure 2-6 Interpolated Method 1 (solid) vs. Solution-Method 2 (dashed) regrid

If a new solution is performed between levels, however, the regrid can detect that the change in potential is localized, as in the dashed curve of Figure 2-6. To allow for this phenomenon, it is normal procedure to regrid one level at a time, rereading the doping statements and performing a new solution between levels.

Reading Doping Information

As mentioned in the previous section, by default Medici calculates the doping and solution information by interpolation. To provide better values for the doping at newly created nodes (if analytical profiles are being used) it is best to use the **IN.FILE** parameter on the **REGRID** statement, as illustrated in the example below. This way the original analytical doping functions are used to calculate the doping at the new nodes rather than interpolation.

```
MESH.....
PROFILE OUT.FILE=DOPE1...
PROFILE.....
PROFILE.....
PROFILE.....
REGRID IN.FILE=DOPE1
```

Refinement Criterion

It is an open question as to what is the correct criterion to refine the grid in semiconductor problems. When solving Poisson's equation by finite-element methods, there is a natural choice in the error estimate. The grid can be refined to equidistribute the Poisson error. For on-state simulation, however, estimates for the error in the continuity equation are unreliable. In addition, there is also the problem of having three different error estimates, one from each equation.

Rather than deriving mathematically-based error estimates, physically plausible heuristics are chosen for the refinement criteria. While not optimal, this approach gives considerable control over grid generation. There are two main options:

- Refine where a particular variable exceeds some value
- Refine where the change in that variable across an element exceeds a given value

The variable can be any of the key quantities in the problem such as:

- Potential
- Electron or hole concentration
- Electron or hole quasi-Fermi potential
- Doping
- Electric field or minority carrier concentration

The value to choose depends on the size of the structure and the accuracy desired.

Ideally, no element would have a step of more than kT / q in potential or quasi-Fermi potential across it, but in practice the refinement criterion is $10 - 20\ kT / q$. Similarly, doping should not change by more than half an order of magnitude across an element, but a refinement criterion of two to three orders is more practical. In high level injection, regridding appears to be useful wherever the value of minority carrier exceeds the local doping.

Mesh Smoothing

Several procedures are available for dealing with poorly shaped elements in the mesh, and in particular with very obtuse elements. Although every step of grid generation can introduce obtuse elements, two steps in particular can cause problems.

1. Distorting a rectangular mesh unavoidably introduces a large number of very obtuse elements.
2. When a grid containing high aspect ratio (>2.8) elements is refined, very obtuse elements can be created in the transition region between high and low grid density.

The two main techniques available to treat these difficulties are:

- Node smoothing
- Element smoothing

Node Smoothing

Node smoothing includes several iterative passes during which each node is moved to a position that improves the angles of the surrounding elements.

Node smoothing is suitable only for an initial irregular grid. Node smoothing should never be used with a refined or distorted rectangular grid. In these cases, node smoothing tends to redistribute fine grid away from the physical phenomena requiring the fine grid.

Element Smoothing

Element smoothing examines each adjoining pair of elements, and if necessary the diagonal of the quadrilateral is flipped. This has the effect of stabilizing the discretization.

- When the two elements are of different materials, the diagonal is never redrawn.
- With elements of the same material but different region number, the element may or may not be flipped at your discretion.

Element smoothing is desirable in almost all cases, and should be performed both on the initial grid and on subsequent regrids. The only exception to this rule arises from an undesirable interaction of three elements: regrid, high aspect ratio elements, and smoothing.

This interaction occurs frequently in oxide regions, since they often have very long elements. In the transition region between refined and unrefined regions, nodes are added to the sides of unrefined element. If a thin element is so modified, a very obtuse angle is created. If the resulting sub-elements are smoothed, unusual patterns can result.

Usually, this is not a cause for concern since no current flows in the oxide, but the elements may look more acceptable if smoothing is omitted. There is also a provision to automatically add more points in the oxide, thus avoiding the issue entirely. See Figure 2-7.

- Figure 2-7(a) shows the vertical doping in the silicon region ($y > 0$) of a structure and the initial grid is shown in Figure 2-7(b).
- Figure 2-7(c) shows the grid after a refinement on doping without smoothing.
- Figure 2-7(d) is the same grid, with smoothing. The elements in the bulk are better shaped.
- In both Figure 2-7(c) and Figure 2-7(d), more points have been added in the oxide ($y < 0$) so the mesh maintains good angles throughout. However, the extra nodes are redundant. To avoid this, do not request extra nodes.
- In Figure 2-7(e) no points are added to the oxide and smoothing is everywhere.

  The smoothing algorithm, in its efforts to stabilize the matrix, creates a mesh hole, caused by the high aspect ratio grid spacing in the oxide.
- If desired, the smoothing can be turned off in the oxide. Figure 2-7(f) shows the grid after refinement, with no points added to the oxide, and no smoothing in the oxide.

Note:

The last two grids (e and f) differ only in how the nodes are connected; the nodes themselves lie in the same positions.

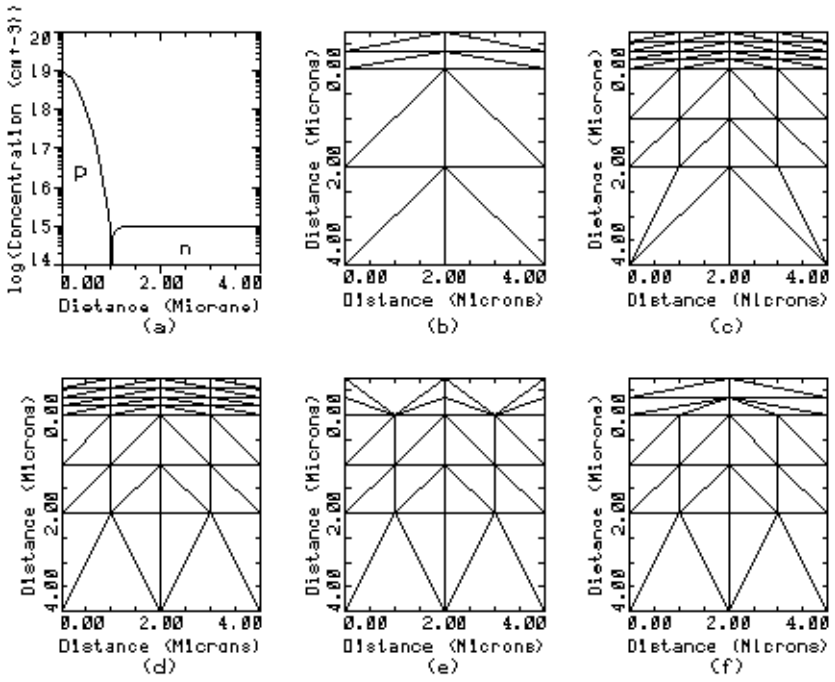


Figure 2-7 Grids in various states with and without smoothing

Undesirable Effects of Obtuse Elements

It is difficult to triangulate a general region without obtuse elements (though an algorithm has been developed, see Reference [39]). Obtuse elements have the following undesirable side effects on a simulation:

- They increase inherent roughness in the solution, making contour plots more difficult to interpret
- They can cause any solution technique to fail

Either problem can occur whenever the sum of opposite angles in a pair of elements exceeds 180 degrees. This causes the matrix coefficient coupling neighboring nodes to change sign. By flipping the diagonal, the sum of opposite angles is made less than 180 degrees.

Figure 2-7(c) and Figure 2-7(d) show a grid refined first without flipping and then with flipping. Without flipping, the discretization is potentially unstable, and may lead to unphysical solutions or poor convergence.

The roughness that obtuse elements bring into the solution is more difficult to cure. Generally, the best that can be done is to ensure that the grid is sufficiently fine where the solution varies rapidly, and that the initial grid has well-shaped elements.

Transient Device Simulation

For transient device simulation, Medici uses the BDF1 or BDF2 formulas.

Because of the extremely rigid nature of the semiconductor device equations, strong stability requirements are placed on any proposed transient integration scheme. In mathematical terms, such a scheme needs to be both A- and L-stable (Reference [40]).

The most convenient methods to use are one-step integration, so that only the solution at the most recent time step is required.

BDF1 Formula

Most device simulation codes have made use of a simple first-order (implicit) backward difference formula (BDF1) References [40] and [41]. This means that Equations 2-2, 2-3, 2-330, and 2-355 are discretized as

$$\frac{n_k - n_{k-1}}{\Delta t_k} = F_n(\psi_k, n_k, p_k, u_{n,k}, u_{p,k}, T_k) = F_n(k) \qquad \text{Equation 2-249}$$

$$\frac{p_k - p_{k-1}}{\Delta t_k} = F_p(\psi_k, n_k, p_k, u_{n,k}, u_{p,k}, T_k) = F_p(k) \qquad \text{Equation 2-250}$$

Equation 2-251

$$\frac{n_k\, u_{n,k} - n_{k-1} u_{n,k-1}}{\Delta t_k} = F_{u_n}(\psi_k, n_k, p_k, u_{n,k}, u_{p,k}, T_k) = F_{u_n}(k)$$

Equation 2-252

$$\frac{p_k\, u_{p,k} - p_{k-1} u_{p,k-1}}{\Delta t_k} = F_{u_p}(\psi_k, n_k, p_k, u_{n,k}, u_{p,k}, T_k) = F_{u_p}(k)$$

$$\frac{T_k - T_{k-1}}{\Delta t_k} = F_T(\psi_k, n_k, p_k, u_{n,k}, u_{p,k}, T_k) = F_T(k) \qquad \text{Equation 2-253}$$

where $\Delta t_k = t_k - t_{k-1}$ and ψ_k denotes the potential at time t_k, and so on. This scheme (also known as the backward Euler method) is a one-step method and is known to be both A- and L-stable (References [40] and [42]).

The disadvantage is that it suffers from a large local truncation error (LTE) which is proportional to the size of the time steps taken. This means that the error at each time step k is $0(\Delta t_k)$.

BDF2 Formula

An alternative to BDF1, a second-order backward difference formula (BDF2) could be used (Reference [40]):

$$\frac{1}{t_k - t_{k-2}}\left(\frac{2-\gamma}{1-\gamma} n_k - \frac{1}{\gamma(1-\gamma)} n_{k-1} + \frac{1-\gamma}{\gamma} n_{k-2}\right) = F_n(k) \qquad \text{Equation 2-254}$$

$$\frac{1}{t_k - t_{k-2}}\left(\frac{2-\gamma}{1-\gamma}p_k - \frac{1}{\gamma(1-\gamma)}p_{k-1} + \frac{1-\gamma}{\gamma}p_{k-2}\right) = F_p(k) \quad \text{Equation 2-255}$$

$$\frac{1}{t_k - t_{k-2}}\left(\frac{2-\gamma}{1-\gamma}n_k u_{n,k} - \frac{1}{\gamma(1-\gamma)}n_{k-1}u_{n,k-1} + \frac{1-\gamma}{\gamma}n_{k-2}u_{n,k-2}\right) = F_{u_n}(k) \quad \text{Equation 2-256}$$

$$\frac{1}{t_k - t_{k-2}}\left(\frac{2-\gamma}{1-\gamma}p_k u_{p,k} - \frac{1}{\gamma(1-\gamma)}p_{k-1}u_{p,k-1} + \frac{1-\gamma}{\gamma}p_{k-2}u_{p,k-2}\right) = F_{u_p}(k) \quad \text{Equation 2-257}$$

$$\frac{1}{t_k - t_{k-2}}\left(\frac{2-\gamma}{1-\gamma}T_k - \frac{1}{\gamma(1-\gamma)}T_{k-1} + \frac{1-\gamma}{\gamma}T_{k-2}\right) = F_T(k) \quad \text{Equation 2-258}$$

where

$$\gamma \triangleq \frac{t_{k-1} - t_{k-2}}{t_k - t_{k-2}} \quad \text{Equation 2-259}$$

Like BDF1, BDF2 is both A- and L-stable, but BDF2 requires two previous solutions at times t_{k-1} and t_{k-2}.

Time Steps Selection

Medici employs a variable order method as the default through use of the automatic time step selection (**TAUTO**) and (**2NDORDER**) time step algorithm. At a given time point, the program:

- Calculates the local truncation error (LTE) for both the BDF1 and BDF2 methods
- Calculates the new time step for both methods by matching the expected LTE with the **TOL.TIME** parameter
- Selects the method which yields the largest time step

The **TOL.TIME** parameter can be varied on the **METHOD** statement. Specifying a larger tolerance results in a quicker, but less accurate, simulation.

Dynamic Error Tolerance

Dynamic error tolerance option is invoked by setting **TOL.TIME** to zero. With this option, the error tolerance is automatically adjusted according to simulation conditions to produce accurate simulation results without user intervention.

Time Step Restrictions

Medici places the following restrictions on the time steps:

- The time step size is allowed to increase at most by a factor of two.
- If the new time step is less than one half of the previous step, the previous time step is recalculated.
- If a time point fails to converge, the time step is reduced by a factor of two, and the point is recalculated.

- If the Circuit Analysis AAM is used, breakpoints are set that force Medici to calculate time points at the edges of **PULSE** type waveforms.

User Specifications

To start a transient simulation, you must specify the following:

- An initial time step
- An input bias condition (voltage/current, ramp/step)

All the numerical parameters have defaults and most are accessible. Time steps are stored for each solution. This allows a transient simulation to be continued from a saved solution file without specifying a new initial time step. See the descriptions of the **METHOD** and **SOLVE** statements in Chapter 3, "METHOD" on page 3-139, and "SOLVE" on page 3-150. Also see the example in Chapter 6 for more details.

Impulse Response

If a voltage step or ramp (a sequence of small incremental steps) is applied directly to a semiconductor contact (not through a resistor), the resulting terminal currents include an impulse response term (delta function). This is due to the infinitely rapid change in voltage and the intrinsic capacitance of the device, which depends, in turn, on the size of the simulation space and dielectric constants. Usually, this impulse term does not affect the solution after one or two time steps.

AC Small-Signal Analysis

In addition to DC steady state and transient analysis, Medici also allows AC small-signal analysis as a post-processing step after a DC solution.

Note:

Medici ***cannot perform AC analysis with the AC signal applied to electrodes with charge boundary conditions.***

Note:

To perform AC analysis, the Newton method must be specified.

Basic Concepts

Starting from a DC bias condition, an input of given amplitude and frequency can be applied to a device structure from which sinusoidal terminal currents and voltages are calculated. Then using the relationship

$$\tilde{Y}_{ij} = G_{ij} + j\omega C_{ij} = \frac{\tilde{I}_i}{\tilde{V}_j}, \quad \tilde{V}_k = 0 \quad \text{for} \quad k \neq j \qquad \text{Equation 2-260}$$

the frequency dependent admittance matrix, and then capacitance and conductance, can be calculated. By varying the frequency and examining the various device admittances, f_T can be directly determined.

Although charge partitioning methods can be used in programs like Medici to obtain accurate estimates of capacitance, they do have some drawbacks which can be overcome by using AC small-signal analysis.

For example, the PISCES-II program has been successfully applied to the calculation of bias dependent gate capacitances for small geometry MOSFETs (Reference [43]). Solving for a bias point (V_G, V_D, V_S, V_B) and then solving again for V_G, V_D, + ΔV, V_S, V_B, produces very accurate estimates of capacitance by integrating the charge on the gate electrode at both bias points.

$$C_{ij} = C_{GD} = \frac{dQ_G}{dV_D} \approx \frac{\Delta Q_G}{\Delta V} \qquad \text{Equation 2-261}$$

The above procedure can be repeated to obtain C_{GS} and C_{GB}. This is done by applying the potential increment to the source and bulk respectively.

This technique of charge partitioning has the following disadvantages:

- It is only rigorously correct for cases where the current into electrode *i* (to calculate C_{ij}) is strictly displacement current, such as insulator contacts like MOS gates. Therefore, capacitances such as C_{DG} and C_{SG} cannot be accurately estimated by this method.
- Capacitances obtained via this method are strictly quasi-static (low frequency) values.

See Laux, Reference [44].

Sinusoidal Steady-State Analysis

Using sinusoidal steady-state analysis, both of the drawbacks associated with the charge partitioning approach are eliminated. In Medici, the approach of Laux (Reference [44]) has been followed. An AC sinusoidal voltage bias is applied to an electrode *i* such that

$$V_i = V_{i0} + \tilde{V}_i e^{j\omega t} \qquad \text{Equation 2-262}$$

where V_{i0} is the existing DC bias, $\tilde{V}_i$ is the magnitude of the AC sinusoidal bias and V_i is the actual bias (sum) to be simulated. Assuming that you are solving only the basic equations (Poisson and current continuity) in a single semiconductor material, Equations 2-1 through 2-3 can be rearranged to obtain

$$F_\psi(\psi, n, p) = \nabla^2\psi + q(p - n + N_D^+ - N_{\bar{A}}) + \rho_F = 0 \qquad \text{Equation 2-263}$$

$$F_n(\psi, n, p) = \frac{1}{q}\vec{\nabla} \cdot \vec{J}_n - U_n = \frac{\partial n}{\partial t} \qquad \text{Equation 2-264}$$

$$F_p(\psi, n, p) = -\frac{1}{q}\vec{\nabla} \cdot \vec{J}_p - U_p = \frac{\partial p}{\partial t} \qquad \text{Equation 2-265}$$

The AC solution to Equations 2-263 through 2-265 can be written as

$$\psi_i = \psi_{i0} + \tilde{\psi}_i e^{j\omega t} \quad \text{Equation 2-266}$$

$$n_i = n_{i0} + \tilde{n}_i e^{j\omega t} \quad \text{Equation 2-267}$$

$$p_i = p_{i0} + \tilde{p}_i e^{j\omega t} \quad \text{Equation 2-268}$$

where ψ_i, n_{i0} and p_{i0} are the DC potential and carrier concentrations at node i while $\tilde{\psi}_i$, $\tilde{n}_i$ and $\tilde{p}_i$ are the respective AC values, which in general are complex.

By substituting Equations 2-266 through 2-268 back into Equations 2-263 through 2-265 and expanding as a Taylor series to first-order only (the small-signal approximation), you obtain nonlinear equations of the form

$$F(\psi, n, p) = F(\psi_0, n_0, p_0) + \frac{\partial F}{\partial \psi}\tilde{\psi}e^{j\omega t} + \frac{\partial F}{\partial n}\tilde{n}e^{j\omega t} + \frac{\partial F}{\partial p}\tilde{p}e^{j\omega t} \quad \text{Equation 2-269}$$

for each of the three PDEs. If a valid DC solution at the desired DC bias has already been computed, then $F(\psi_0, n_0, p_0) = 0$ so the following linear system is obtained:

$$\begin{bmatrix} \frac{\partial F_\psi}{\partial \psi} & \frac{\partial F_\psi}{\partial n} & \frac{\partial F_\psi}{\partial p} \\ \frac{\partial F_n}{\partial \psi} & \frac{\partial F_n}{\partial n} + D_1 & \frac{\partial F_n}{\partial p} \\ \frac{\partial F_p}{\partial \psi} & \frac{\partial F_p}{\partial n} & \frac{\partial F_p}{\partial p} + D_1 \end{bmatrix} \begin{bmatrix} \tilde{\psi} \\ \tilde{n} \\ \tilde{p} \end{bmatrix} = b_1 \quad \text{Equation 2-270}$$

In this equation:

- $\partial F/\partial \psi$, $\partial F/\partial n$ and $\partial F/\partial p$ are $N \times N$ matrices, which form the DC Jacobian.
- D_1 is an $N \times N$ matrix with $-j\omega$ on the diagonal and 0 off-diagonal.
  This results from the expansion of the time-dependent portion of each continuity equation.
- b_1 is a vector of length $3N$ which contains the AC input voltage boundary conditions.

Splitting the system (Equation 2-270) into real and imaginary parts, one obtains

$$\begin{bmatrix} J & -D_2 \\ D_2 & J \end{bmatrix} \begin{bmatrix} X_R \\ X_I \end{bmatrix} = b_2 \quad \text{Equation 2-271}$$

In this equation:

- J is the $3N \times 3N$ DC Jacobian.
- D_2 is a $3N \times 3N$ diagonal matrix related to D_1 in Equation 2-270.
- b_2 is a permuted version of b_1 in Equation 2-270.
- X_R and X_1 refer to the real and imaginary AC solution vectors, each $3N$ in length.

The system (Equation 2-271) is twice the size of the DC system.

Successive Over-Relaxation Method (SOR)

Successive Over-Relaxation (SOR) method is the default method. By making use of the availability of the factored J from the DC solution, it requires the least amount of computation for each solution.

Convergence is generally very good for low to moderate frequencies. As frequencies reach high values ($\approx f_T/10$), more iterations are necessary. For frequencies above $f_T/10$, you may have to use a relaxation parameter with a value less than 1.

CAUTION

SOR does not converge for exceedingly high frequencies. See Laux, Reference [44].

Bi-CGSTAB Method

To facilitate analysis at high frequency or in difficult cases, Medici allows the use of a direct method with iterative improvement or Bi-Conjugate Gradient Stabilized solver (Bi-CGSTAB) to solve Equation 2-271. If the direct method is used, a full LU factorization is performed using single precision arithmetic (instead of normal double precision). Bi-CGSTAB is then used to iteratively improve this solution. Since the single precision LU decomposition is normally quit accurate, Bi-CGSTAB will only require 2 or 3 iterations.

The direct method is used if full LU factorization (**ILUCGS**=FALSE) was requested on the **SYMBOLIC** statement. Bi-CGSTAB will be used with a partial LU factorization if an incomplete LU factorization (**ILUCGS**=TRUE) was requested on the **SYMBOLIC** statement. These methods will work at much higher frequency ranges and for certain ill conditioned problems, such as MOS capacitors, but have the disadvantage of being much slower than SOR. The high frequency analysis mode is invoked by selecting the **HI.FREQ** parameter on the **SOLVE** statement. By default, the program will try SOR first, and if this fails, will try **HI.FREQ** analysis.

Convergence Difficulties

In situations where Bi-CGSTAB is not able to converge to the required accuracy, Medici still calculates the requested AC parameters, in case they might still be useful. These results should not be trusted unless:

1. You can determine that the values of the relevant parameters are realistic.
2. Both the real and imaginary part of the electrode current sum up to zero.

CAUTION

If warning messages are obtained from Bi-CGSTAB during AC analysis, the results are probably in error. Statement and/or parameter specifications probably need to be changed.

AC Analysis Beyond the Basic Equations

Medici performs AC analysis as a post-processing step after a DC solution. Medici also allows AC small-signal analysis when one or more temperature equations are solved with the three basic equations. The fundamental procedure remains the same, while the size of the resulting system of equations becomes larger when there is a coupled temperature equation. It should be noted, however, that the results of AC analysis with temperature equations have not been extensively investigated and you are advised to interpret the results with care. It has been observed that AC analysis produces more meaningful results, at the electrical contacts where current level is significant, with lattice temperature equation decoupled rather than coupled.

Details are documented in the **SOLVE** statement in Chapter 3, "SOLVE" on page 3-150.

Impact Ionization Analysis

Impact ionization can be selected in Medici in the following ways:

- For efficiency and speed, impact ionization is available as a post-processing capability.
- The generated carriers (due to impact ionization) can be included self-consistently in the solution of the device equations.

Post-Processing Method

When a post-processing impact ionization analysis is selected, Medici calculates the rate at which electron-hole pairs are generated as a result of impact ionization, and consequently, the generated impact ionization current. The generation rate is based on the electric fields and current densities within the structure for the most recent solution. In a post-processing analysis such as this, the generated carriers are not included in the solution. Therefore, this analysis will not allow you to study effects where the presence of the generated carriers will significantly alter the device behavior.

The post-processing analysis is useful for the following:

- Calculating substrate current for MOS devices.

- Estimating long term device degradation.

  Experimental evidence shows that the number of hot electrons being injected into the oxide can be correlated to the amount of substrate current (References [45] and [46]).
- Warning that the structure may be susceptible to other problems (breakdown or latchup, for example).

  This is a function of the magnitude of the generation rates.

Specification

A post-processing impact ionization analysis is requested by specifying the **IMPACT.I** parameter on the **SOLVE** statement. Each bias and/or time point specified on the **SOLVE** statement calculates the electron-hole pair generation rate due to each species at every node of the simulation mesh.

Output

The output from the post-processing analysis includes the following:

- Maximum generation rate in the device structure
- Location of maximum rate within the device structure
- Magnitude of electric field and current density where rate is maximum
- Total impact ionization current obtained by integrating the total generation rate over the entire device structure

Self-Consistent Method

When impact ionization is included self-consistently in the solution of the device equations, effects in which the presence of the generated carriers have a significant impact on the device behavior can be directly simulated. Examples of this include:

- Avalanche-induced breakdown of a junction
- impact ionization-induced latchup of a structure.

In cases where small voltage changes cause large changes in currents (such as near the breakdown voltage of a device), use of the continuation method (see "Continuation Method" on page 2-66) can easily trace the entire I-V characteristics.

Impact ionization in the solution is requested by specifying the **IMPACT.I** parameter on the **MODELS** statement.

Electron-Hole Generation Rate

With either the post-processing or the in-the-solution method, the generation rate for electron-hole pairs due to impact ionization can be expressed by

$$G^{II} = \alpha_{n,\,ii} \cdot \frac{\left|\vec{J}_n\right|}{q} + \alpha_{p,\,ii} \cdot \frac{\left|\vec{J}_p\right|}{q} \qquad \text{Equation 2-272}$$

where $\alpha_{n,ii}$ and $\alpha_{p,ii}$ are the electron and hole ionization coefficients, and $\vec{J}_n$ and $\vec{J}_p$ are the electron and hole current densities. As described in Reference [47], the ionization coefficients can be expressed in terms of the local electric field according to

$$\alpha_{n,ii} = \alpha^{\infty}_{n,ii}(T) \cdot \exp\left[-\left(\frac{E^{crit}_{n,ii}(T)}{E_{n,\parallel}}\right)^{\mathbf{EXN.II}}\right] \quad \text{Equation 2-273}$$

$$\alpha_{p,ii} = \alpha^{\infty}_{p,ii}(T) \cdot \exp\left[-\left(\frac{E^{crit}_{p,ii}(T)}{E_{p,\parallel}}\right)^{\mathbf{EXP.II}}\right] \quad \text{Equation 2-274}$$

where $E_{n,\parallel}$ and $E_{p,\parallel}$ are the electric field components in the direction of current flow. The critical fields used in the above expressions are obtained from

$$E^{crit}_{n,ii}(T) = \frac{E_g(T)}{q\ \lambda_n(T)}\text{(default calculation), or } \mathbf{ECN.II} \text{ if specified} \quad \text{Equation 2-275}$$

$$E^{crit}_{p,ii}(T) = \frac{E_g(T)}{q\ \lambda_p(T)}\text{(default calculation), or } \mathbf{ECP.II} \text{ if specified} \quad \text{Equation 2-276}$$

where λ_n and λ_p are the optical-phonon mean free paths for electrons and holes, and are given by

$$\lambda_n(T) = \mathbf{LAN300} \cdot \tanh\left(\frac{\mathbf{OP.PH.EN}}{2kT}\right) \quad \text{Equation 2-277}$$

$$\lambda_p(T) = \mathbf{LAP300} \cdot \tanh\left(\frac{\mathbf{OP.PH.EN}}{2kT}\right) \quad \text{Equation 2-278}$$

In these expressions, **OP.PH.EN** is the optical-phonon energy, and **LAN300** and **LAP300** are the phonon mean free paths for electrons and holes at 300 K. The factors $\alpha^{\infty}_{n,ii}$ and $\alpha^{\infty}_{p,ii}$ are given by

$$\alpha^{\infty}_{n,ii}(T) = \mathbf{N.IONIZA} + \mathbf{N.ION.1} \cdot T + \mathbf{N.ION.2} \cdot T^2 \quad \text{Equation 2-279}$$

$$\alpha^{\infty}_{p,ii}(T) = \mathbf{P.IONIZA} + \mathbf{P.ION.1} \cdot T + \mathbf{P.ION.2} \cdot T^2 \quad \text{Equation 2-280}$$

The parameters used in Equations 2-272 through 2-280 can be modified from their default values on the **MATERIAL** statement.

Non-Local Impact Ionization

To improve the accuracy of impact ionization calculations for deep submicron semiconductor devices, a non-local model has been implemented. The non-local model is based on the lucky electron model developed by Jungemann, et al. [106].

The model evaluation starts by tracking the potential drop along the current path at each point in the device. Impact ionization generation occurs only after the potential drop has reached a certain threshold. There are two generation models presented in [106], the hard threshold model and the soft threshold model. At present, only the hard threshold model is available.

To use the non-local impact ionization model, **IMPACT.I** should be specified on the **MODELS** statement. A new parameter, **II.NLOC**, is used to control whether the local or non-local model is invoked:

$$\mathbf{II.NLOC} = \begin{cases} -1 & \text{, local model (default)} \\ 0 & \text{, post-processing non-local model} \\ 1 & \text{, self-consistent non-local model, no derivatives} \\ 2 & \text{, self-consistent non-local model, derivatives included} \end{cases}$$

The threshold energy for electron and hole impact ionization is controlled by the parameters **CN.IIGAP** and **CP.IIGAP**, which can be specified on the **MATERIAL** statement. The default values for these parameters are 1, which specifies that the bandgap energy is used as the threshold:

$$\varepsilon_{TH} = \begin{cases} \mathbf{CN.IIGAP} \cdot E_g & \text{, electrons} \\ \mathbf{CP.IIGAP} \cdot E_g & \text{, holes} \end{cases}$$

Despite the inclusion of non-local derivatives when **II.NLOC**=2 is used, the convergence can sometimes be very difficult near breakdown. Therefore, the user should approach the full-blown breakdown simulation with the non-local model with care.

Examining Results

The impact ionization generation rate can be plotted in the following ways:

- Along one-dimensional slices through the structure using the **PLOT.1D** statement
- Contours of the generation rate in a two-dimensional plot using the **CONTOUR** statement
- Three-dimensional projection plots of the generation rate using the **PLOT.3D** statement

- The impact ionization current obtained in a post-processing analysis can be plotted as a function of bias or, in the case of a transient simulation, time (**PLOT.1D** statement)

The total impact ionization generation rate may be printed over a specified cross-section of the device using the **PRINT** statement.

Gate Current Analysis

Medici enables gate current analysis as a post-processing capability. After obtaining a solution, this function calculates the amount of current collected by the gate for each bias or time point from the available physical quantities within the structure (such as electric field and current density).

Gate current analysis is selected by specifying the **GATE.CUR** parameter on the **SOLVE** statement. The analysis is performed after each solution associated with the **SOLVE** statements on which the **GATE.CUR** parameter appears.

When a gate current analysis is requested, Medici computes electron and hole gate current at each point along all semiconductor-insulator interfaces. If normal insulators are used (such as **OXIDE**, **NITRIDE**, or **INSULATOR**), the program follows electric field lines in the insulator to determine the final location for the injected electron and hole current. The results of this analysis are written to the standard output listing. The total gate current can then be plotted as a function of bias or, in the case of a transient simulation, of time using the **PLOT.1D** statement.

If the insulator is a wide band-gap semiconductor (**S.OXIDE**) and the **GATE2** model is specified, the program injects hot carriers over the insulator-semiconductor barrier and then solves the continuity equations within the **S.OXIDE** region to determine where the electrons go. Due to the time-consuming nature of the gate current model and the fact that gate currents are very small, the injected hot electrons calculated from the previous time or bias point are used in the calculation. The errors from this procedure are negligible if small time or bias steps are taken and if the hot electron current changes slowly.

To select a model for gate current analysis in Medici, model parameters must be specified on the **MODELS** statement. Gate current models and their parameters are described below.

Lucky-Electron Gate Current Model

The default model for performing gate current calculations in Medici is based on the work described in Reference [46]. The principle ideas of the model are based on the lucky electron concept. This model calculates probabilities for certain scattering events to occur that result in current being injected into the gate. This model can be selected by specifying **GATE1** on the **MODELS** statement.

The total gate current (per unit length perpendicular to the simulation plane) can be obtained by integrating the flux of carriers injected into the gate from each location in the device structure. This can be expressed by

Equation 2-281

$$I_{gate} = \iint \Gamma_n(x,y)\left|\vec{J}_n(x,y)\right| \; dx \; dy + \iint \Gamma_p(x,y)\left|\vec{J}_p(x,y)\right| \; dx \; dy$$

where

- $\vec{J}_n$ and $\vec{J}_p$ are the electron and hole current densities
- The factors Γ_n and Γ_p are probabilities per unit length (in the direction of current flow) that electrons or holes, respectively, in the vicinity of the point (x, y) are injected into the gate.

  Γ_n and Γ_p can be expressed as the product of the probability factors for the various scattering events involved

$$\Gamma_n = P_{\Phi_{b,n}} P_{semi,n} P_{insul,n} / \mathbf{LAMRN} \qquad \text{Equation 2-282}$$

$$\Gamma_p = P_{\Phi_{b,p}} P_{semi,p} P_{insul,p} / \mathbf{LAMRP} \qquad \text{Equation 2-283}$$

Probability Factors

For an electron to be collected by the gate, it must acquire enough kinetic energy from the electric field to surmount the potential barrier in the insulator and then be re-directed toward the semiconductor-insulator interface.

The factor $1/\mathbf{LAMRN}$ in Equation 2-282 represents the probability per unit length (in the direction of current flow) that an electron will be re-directed without losing a significant amount of energy. The factor $P_{\Phi_{b,n}}$ is the probability that an electron will acquire enough kinetic energy to surmount the insulator potential barrier $\Phi_{b,n}$, and retains the appropriate momentum after re-direction. This can be expressed as

$$P_{\Phi_{b,n}} = .25\left(\frac{E_{n,\parallel}\mathbf{LAMHN}}{\Phi_{b,n}}\right) \exp\left[-\left(\frac{\Phi_{b,n}}{E_{n,\parallel}\mathbf{LAMHN}}\right)\right] \qquad \text{Equation 2-284}$$

where

- $E_{n,\parallel}$ is the electric field in the direction of current flow,
- **LAMHN** is the hot-electron scattering mean-free-path
- $\Phi_{b,n}$ is the semiconductor-insulator potential barrier which can be expressed as

Equation 2-285

$$\Phi_{b,n}(E_{insul,\perp_s} > 0) = \psi_{b,n} - \mathbf{BARLN}\sqrt{E_{insul,\perp_s}} - \mathbf{TUNLN}E_{insul,\perp_s}^{2/3} - \Delta\psi_{int}$$

In Equation 2-285:

- $\psi_{b,n}$ is the semiconductor-insulator interface barrier
- $E_{insul,\perp_s}$ is the electric field in the insulator ($E_{insul,\perp_s} > 0$ indicates an aiding field for electrons)
- **BARLN** $\sqrt{E_{insul,\perp_s}}$ represents barrier lowering effects due to the image field
- **TUNLN** $E_{insul,\perp_s}^{2/3}$ accounts for the possibility of tunneling
- $\Delta\psi_{int}$ is the potential difference between the interface and the point (x, y)

In the case $E_{insul,\perp_s} < 0$ (a repelling field for electrons), the probability factor $P_{\Phi_{b,n}}$ is set equal to zero.

After gaining enough kinetic energy and being re-directed in the appropriate direction, the electron must not be scattered again before reaching the peak of the potential barrier in the insulator if it is to be collected by the gate.

The factor $P_{semi,n}$ in Equation 2-282 is the probability that an electron will not be scattered in the semiconductor before reaching the interface and is given by

$$P_{semi,n} = \exp[-(d_{int}/\textbf{LAMHN})] \qquad \text{Equation 2-286}$$

where d_{int} is the distance from the point (x, y) to the interface

The factor $P_{insul,n}$ in Equation 2-282 is the probability that an electron will not be scattered in the insulator between the interface and the potential barrier peak. This can be expressed as

$$P_{insul,n} = \exp\left[-\sqrt{\textbf{ECN.GC}/E_{insul,\perp_s}}\right] \qquad \text{Equation 2-287}$$

where **ECN.GC** is a critical electric field for electron scattering in the insulator.

Gate Current Parameters

The following parameters are available on the **MATERIAL** statement:

- Electron gate current parameters **LAMHN**, **LAMRN**, **BARLN**, **TUNLN**, and **ECN.GC**.
- Corresponding hole gate current parameters **LAMHP**, **LAMRP**, **BARLP**, **TUNLP**, and **ECP.GC**.

The interface barriers $\psi_{b,n}$ and $\psi_{b,p}$ are computed from the electron affinities and energy bandgaps of the interface materials as

$$\psi_{b,n} = \chi_{semi} - \chi_{insul} \qquad \text{Equation 2-288}$$

and

$$\psi_{b,p} = \left(\chi_{insul} + \frac{E_{g,insul}}{q}\right) - \left(\chi_{semi} + \frac{E_{g,semi}}{q}\right) \qquad \text{Equation 2-289}$$

The electron affinities and energy bandgaps for both semiconductor and insulating materials can be specified on the **MATERIAL** statement.

The default values for the gate current model parameters for electrons in silicon and silicon dioxide are taken (or computed) from values given in Reference [46]. The default parameter values are as follows:

- Hole parameters in silicon and silicon dioxide are taken to be equal to the corresponding electron parameters values.
- Other semiconductor materials are taken to be equal to the corresponding silicon parameter values.
- Other insulating materials are taken to be equal to the corresponding silicon dioxide parameter values.

Angle-Dependent Gate Current Model

The angle dependent gate current model contains physical mechanisms unavailable in the other models. This model may be selected by specifying **GATE2** on the **MODELS** statement. The basic equation that governs the operation of the model is as follows (only the electron portion of the expression is written)

Equation 2-290

$$I_{gate} = \frac{61.6\text{x}10^{-7}}{\textbf{LAMRN}} \int_0^{\pi}\int\int \frac{J_n}{d_s} P_{\Phi_{b,n}} P_{semi,n} P_{insul,n} P_{\theta,n} \; dx \; dy \; d\theta$$

In this model:

- direction x is taken parallel to the channel
- y is perpendicular to the channel
- θ is the angle between the x axis and the vector V connecting the point (x,y) to a point on the interface (see Figure 2-8 on page 2-3)
- angle θ sweeps V along the length of the channel
- $P_{semi,n}$ has the same form as in the previous models: $P_{semi,n} = \exp(-|V|/\textbf{LAMHN})$
- numerical constant 61.6e-7 corresponds to the default value of **LAMRN**

The generation function is derived from the Maxwell distribution and has the following form:

Equation 2-291

$$P_{\Phi_{b,n}} = \left(1 + \frac{\Phi_{b,n}}{E_{n,\|}\textbf{LAMHN}\sin(\theta)}\right) \exp\left[-\left(\frac{\Phi_{b,n}}{E_{n,\|}\textbf{LAMHN}\sin(\theta)}\right)\right]$$

The above function has a maximum value of 1, which occurs when $\Phi_{b,n} = 0$.

Insulator Probability

The form of the insulator probability function and the barrier energy depends on whether the electric field in the insulator is an "aiding" field or a "repelling" field. For an aiding field (i.e., one that sweeps electrons from the semiconductor onto the floating gate)

$$P_{insul,n} = \exp\left[-min\left(t_{ins}/3.2e-7, \sqrt{\texttt{ECN.GC}/E_{insul,\perp}}\right)\right] \quad \text{Equation 2-292}$$

$$\Phi_{b,n} = \psi_{b,n} - \texttt{BARLN}\sqrt{E_{insul,\perp_s}} - \texttt{TUNLN}E_{insul,\perp_s}^{2/3} - \Delta\psi_{int} \quad \text{Equation 2-293}$$

In the above equation, t_{ins} is the thickness of the insulator layer, and the constant 3.2e-7 is the mean-free-path for electrons in silicon dioxide. For a repelling field in the insulator, the following equations are used:

$$P_{insul,n} = \exp(-t_{ins}/3.2e-7) \quad \text{Equation 2-294}$$

$$\Phi_{b,n} = \psi_{b,n} - \Delta\psi_{int} + \Delta\psi_{ins} \quad \text{Equation 2-295}$$

In the above equation, $\Delta\psi_{ins}$ is the potential drop across the insulator. For the repelling field, this quantity is positive and serves to increase the potential barrier the electron must surmount and, therefore, reduces the gate current.

Electron Scattering

The factor $P_{\theta,n}$ gives the probability that a hot electron will be scattered in the direction θ and thereby reach the interface. The following form is used and is based on the Rutherford scattering law for charged particles in a Coulomb field.

$$P_{\theta,n} = \left[\frac{q}{\varepsilon_{si}\tan(\alpha/2)\ d_s\ \Phi_{b,n}}\right]^2 \quad \text{Equation 2-296}$$

In the above equation, α is the angle between the vector V and the electron drift velocity vector, which is directed opposite to the electric field vector E. (See Figure 2-8.)

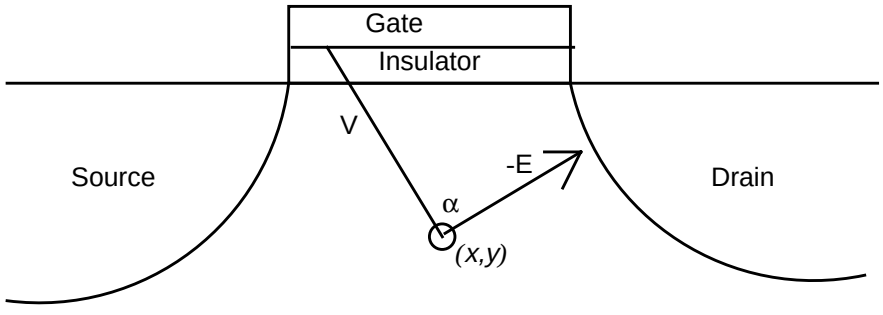


Figure 2-8 Definitions for modified current model

d_s is the distance between scattering sites (assumed to be the ionized impurities).

Therefore, d_s is given by:

$$d_s = (N_A + N_D)^{-0.3333}$$ Equation 2-297

Surface Generation

When using this model, the gate current calculations can be confined to the semiconductor-insulator interface by specifying the **GATE.SUR** on the **MODELS** statement. If this parameter is specified, then

- $P_{\theta,n}$ = for all (x,y) not at the interface
- $P_{\theta,n}$ = 1/2 for all (x,y) at the interface
- $P_{semi,n}$ is set equal to **LAMHN**, and generation only occurs for nodes at the semiconductor interface

  This mode is especially useful when course grids are used (grids with vertical spacings greater than **LAMHN**).

Non-Maxwellian Hot Carrier Generation Option

Equations 2-284 and 2-291 are similar to a Maxwellian carrier energy distribution. Recent Monte Carlo studies of the high energy tail of the distribution have shown a non-Maxwellian form to be more appropriate. In their paper (Reference [49]), Fiegna et al. developed a model based upon Monte Carlo studies with the following form:

$$I_{gate} = -q\ A\int n(x)P_{insul,n}\int_{\Phi_{b,n}}^{\infty} u^{1.5}\exp\left(\frac{-\chi\ u^3}{E_{n,\|}^{1.5}}\right)du\ dx$$ Equation 2-298

In the above equation, the integration in x is carried out at the interface along the length of the channel. Constants A and χ have the values 487 and 1.3e8, respectively.

Returning to the lucky electron model of Equation 2-281, if it is assumed that the barrier energy $\Phi_{b,n}$ and electric field $E_{n,\|}$ remain constant in the y direction (perpendicular to the interface), and that due to the high electric field, $J_n \approx qn(x)v_s$ Equation 2-281 can be reduced to:

$$I_{gate} \approx -q\ v_s\ \frac{\mathbf{LAMHN}}{\mathbf{LAMRN}}\int n(x)\ P_{\Phi_{b,n}}\ P_{insul,n}\ dx$$ Equation 2-299

New non-Maxwellian models can therefore be created by replacing $P_{\Phi,n}$ in the previous models with the following expression:

$$P_{\Phi,n} = A\ \frac{61.6\text{x}10^{-7}}{\mathbf{LAMRN}\ v_s}\int_{\Phi_b}^{\infty} u^{1.5}\exp\left(\frac{-\chi\ u^3\ \ 9.2e{-}7}{E_{n,\|}^{1.5}\ \ \mathbf{LAMRN}}\right)\ du$$ Equation 2-300

The two numerical constants, 61.6e-7 and 9.2e-7 correspond to the default values of **LAMRN** and **LAMHN**, respectively. By writing the expression this way, the model of Fiegna *et al*. results when the default values are used for **LAMHN** and **LAMRN**.

This form of $P_{\Phi_{b,n}}$ can be used in the gate current models described previously by specifying **GATE.GEN**=2 on the **MODELS** statement. If **GATE.GEN**=1 is specified (the default), the original Maxwellian expressions for $P_{\Phi_{b,n}}$ is used.

Carrier Temperature Dependent Gate Current Models

The gate current models in Medici are, by default, based on the local electric field within the structure. When knowledge of the carrier temperature obtained from an energy balance solution is available, this information can be used to obtain more accurate predictions of gate current.

Electron Probability

The function $P_{\Phi_{b,n}}$ gives the probability that an electron will have sufficient energy to surmount the semiconductor-insulator barrier. By default, Medici calculates this probability based on the electric field at a given point in the semiconductor.

Conversion to Local Electric Field Model

The energy balance model, on the other hand, gives the mean energy of the carrier distribution at each point in the semiconductor and thereby accounts for the history of the electrons as they travel through regions of varying electric field.

The local electric field model can easily be converted into an electron energy dependent model by using the homogenous steady state limit of the energy balance equation to convert the electron temperature T_n into the equivalent effective electric field $E_{n,\|}$

$$E_{n,\|} = \frac{3\ k\ T_n}{2\ q\ v_s\ \mathbf{ELE.TAUW}}. \qquad \text{Equation 2-301}$$

The carrier temperature-dependent forms of the gate current models described previously can be selected by specifying **GATE.TEM** on the **MODELS** statement.

Direct Tunneling

Medici enables direct tunneling current analysis using two different approaches: as a post-processing capability and as a self-consistent model. After obtaining a solution in post-processing mode, this function calculates the amount of direct tunneling current collected by the electrodes for each bias or time point from the band edges, quasi-fermi levels, and material properties. In self-consistent mode, the direct tunneling current acts as a self-consistent boundary condition for the electron and hole currents along semiconductor/insulator interfaces.

The post-processing mode of direct tunneling analysis is selected by specifying the **DT.CUR** parameter on the **SOLVE** statement. The analysis is performed after each solution associated with the **SOLVE** statements on which the **DT.CUR** parameter appears. The self-consistent mode of direct tunneling analysis is activated by specifying the **DT.CUR** parameter on the **MODELS** statement.

Conduction band electron tunneling (CBET) and valence band electron tunneling (VBET) can be selectively activated by specifying **DT.CBET** and **DT.VBET**,

respectively, on the **SOLVE** and **MODELS** statements. When a self-consistent solution with CBET is requested, the electron continuity equation must be solved. Likewise, when a self-consistent solution with VBET is requested, the hole continuity equation must be solved. In addition, if the VBET tunneling current is directed into a semiconducting region, such as a poly gate, then the electron continuity equation must be solved as well.

There are three different methods available for evaluating the direct tunneling current.

- Analytical Evaluation
- Numerical integration of WKB tunneling coefficient
- Numerical integration of Gundlach tunneling coefficient

The desired method can be specified via the **DT.METH** parameter on the **SOLVE** statement for post-processing mode, and via the **DT.METH** parameter on the **MODELS** statement for self-consistent mode.

Under some circumstances, such as in a MOSFET, the impact of the direct tunneling current on convergence is rather weak. In this case, the speed of the simulation can be increased by suppressing the contribution of direct tunneling to the jacobian. This can be done by using the **DT.JACOB** parameter on the **METHOD** statement. In other cases, such as a MOSCAP under inversion, direct tunneling has a large impact on the solution and the jacobian contribution must be included for good convergence.

When an analysis of the direct tunneling current is requested, Medici computes the net tunneling current due to electrons tunneling from the conduction and valence band of a semiconductor or electrode, through an insulator, into the conduction band of another semiconductor or electrode (see Figure 2-9). This model is intended to be used for devices in which the dominant contribution of direct tunneling current arises from interfaces that are approximately parallel, such as in a MOS capacitor or along the bottom of a gate in a MOSFET. The location where the tunneling current is deposited is determined by locating the closest point on the opposing boundary. In post-process mode, the results of the direct tunneling analysis are written to the standard output listing as well as stored internally for subsequent plotting and extraction via the predefined quantity *fe(i)*. In self-consistent mode, the direct tunneling current contributes to the terminal currents.

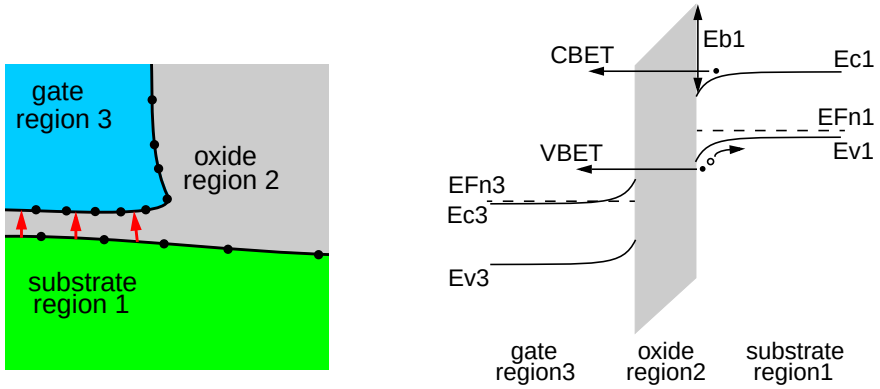


Figure 2-9 Device structure and band diagram for direct tunneling a) Device structure showing the origination of the direct tunneling current from region 1 through region 2 into region 3 b) Band diagram for direct tunneling of electrons in the conduction band (CBET) and valence band (VBET) from region 1 to region 3

Parameter Specification

When direct tunneling analysis is specified, Medici considers direct tunneling between all semiconductor/electrode interfaces separated by an insulator. The parameters used in the direct tunneling model are specified on the **MATERIAL** statement. The adjustable parameters used in the tunneling model are:

- **ME.DT**, the effective tunneling mass of electrons in the conduction band of the material or electrode.
- **MHH.DT** and **MLH.DT**, the effective tunneling mass of electrons in the heavy and light hole bands, respectively, for VBET.
- **BARR.DT**, the barrier height at an electrode/insulator interface for tunneling. This parameter can only be specified for electrodes. For a semiconductor/insulator interface, the electron affinities specified by **AFFINITY** and the band gap (for VBET) are used to calculate the barrier height.

Relationship to Fowler-Nordheim Tunneling

As shown in Figure 2-10 the injection of carriers through an insulating layer is typically separated into three regimes:

- CBET direct tunneling which occurs for electrons with energies below the minimum of the insulator conduction band.
- Fowler-Nordheim tunneling which occurs for electrons with energies between the extrema of the insulator conduction band.
- Thermionic emission; electrons with energies above the barrier are injected via thermionic emission and can be modeled using Medici's gate current model.

The direct tunneling CBET model in Medici combines both the direct tunneling regime and the Fowler-Nordheim regime. The direct tunneling model (**DT.CUR**) and the Fowler-Nordheim model (**FN.CUR**) are, therefore, mutually exclusive. Only one of them can be activated at a time. In post-processing mode, the results of the tunneling analysis for these models are stored in the predefined quantity *fe* for plotting and extraction, while in self-consistent mode the tunneling current appears as part of the terminal currents.

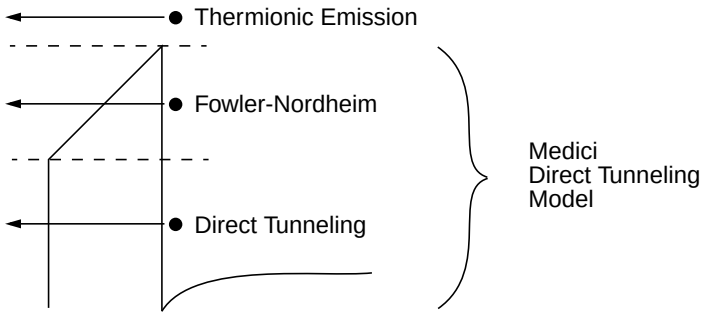


Figure 2-10 Diagram of three main types of carrier injection through an insulator

Direct Tunneling Model

Medici can calculate direct tunneling from two sources: conduction band electron tunneling (CBET) and valence band electron tunneling (VBET). As diagrammed in Figure 2-9, CBET is due to the tunneling of electrons from the *conduction* band of a semiconductor or electrode, through a thin insulating layer, into the conduction band of another semiconductor or electrode. In contrast, VBET is due to the tunneling of electrons from the *valence* band of a semiconductor with the generation of free holes. The net tunneling current across the insulator is calculated using the independent electron approximation and for CBET is given by [102]:

$$j_{DT} = \frac{4\pi q m_1 k_B T}{h^3} \int_0^{E_b} TC(E) \ln\left[\frac{e^{(E_{Fn1} - E_{c1} - E)/k_B T} + 1}{e^{(E_{Fn3} - E_{c1} - E)/k_B T} + 1}\right] dE \quad \text{Equation 2-302}$$

where the integral is over the vertical kinetic energy, E, of the incident electrons. With reference to Figure 2-9, E_{Fn1}, E_{c1}, and m_1 are the electron quasi-fermi level, the conduction band edge, and the electron effective tunneling mass, respectively, in region 1 at the insulator interface. E_{Fn3} and E_{c3} are the corresponding electron quasi-fermi level and conduction band edge in region 3. The endpoint of the integration is determined by the barrier height, E_{b1}. The electron charge is given by q, h is Planck's constant, and k_BT is the thermal energy. TC is the tunneling coefficient of an electron with energy E. For VBET, Equation 2-302 is computed for electrons in both the heavy and light hole bands with E_{c1} replaced by E_{v1}, the valence band in region 1; E_{fn1} replaced by E_{fp1}, the hole quasi-fermi level in

region 1; and the tunneling masses in region 1 are set by the light and heavy hole effective masses. VBET only occurs when the valence band in region 1 is above the conduction band in region 3, with the integration in Equation 2-302 over the band edge difference.

The method for evaluating Equation 2-302 can be specified via the **DT.METH** parameter on the **SOLVE** or **MODELS** statement. These different evaluation techniques are described below and provide the capability to trade speed for accuracy.

Direct Tunneling Method 1: Analytical Evaluation

Setting **DT.METH**=1 causes Equation 2-302 to be evaluated analytically based on a WKB calculation of the tunneling coefficient through a trapezoidal barrier and a step-function approximation for the carrier occupancy. This is the fastest method of evaluation and leads to the following modified Fowler-Nordheim equation for the tunneling current in the direct tunneling regime [103]:

Equation 2-303

$$j_{DT} = \frac{\mathbf{A.FN}(V_{ins}/t_{ins})^2}{[1-(1-V_{ins}/\tilde{\phi}_1)^{1/2}]^2}\exp\left[-\frac{\mathbf{B.FN}}{(V_{ins}/t_{ins})}(1-(1-V_{ins}/\tilde{\phi}_1)^{3/2})\right]$$

where **A.FN** and **B.FN** are the Fowler-Nordheim parameters available on the **MATERIAL** statement. The effective barrier height, $\tilde{\phi}_1$ is given by $\tilde{\phi}_1 = E_{c1} + E_{b1} - E_{Fn1}$. The potential drop across the insulator layer of thickness t_{ins} is given by V_{ins}. Equation 2-303 is only valid in the direct tunneling regime, i.e. for $V_{ins} < \tilde{\phi}_1$. For $V_{ins} > \tilde{\phi}_1$, tunneling occurs via Fowler-Nordheim injection in which case $V_{ins}/\tilde{\phi}_1$ is clamped at unity, and the standard Fowler-Nordheim current equation is recovered. Note that this method produces a non-zero current in equilibrium and cannot be used in self-consistent mode.

Direct Tunneling Method 2: Numerical Integration of WKB Tunneling Coefficient

Setting **DT.METH**=2 causes the integral in Equation 2-302 to be evaluated numerically using a WKB calculation of the tunneling coefficient through a trapezoidal barrier. The WKB tunneling coefficient is given by:

Equation 2-304

$$TC(E) = \exp\left(-\frac{4}{3}\left(\frac{8\pi^2 m_2}{h^2}\right)^{1/2}\left(\frac{t_{ins}}{qV_{ins}}\right)\left[\left(E_{b1}-E-q\frac{V_{ins}}{t_{ins}}b\right)^{3/2} - \left(E_{b1}-E-q\frac{V_{ins}}{t_{ins}}a\right)^{3/2}\right]\right)$$

where

- E_{b1} is the barrier height on the incident side
- m_2 is the effective tunneling mass of electrons in the insulator
- a and b are the classical turning points.

The effective tunneling mass of the electron in the insulator is specified via the **ME.DT** parameter on the **MATERIAL** statement. When tunneling from the conduction band of a semiconductor, the barrier height, E_{b1}, is determined from the

electron affinities of the semiconductor and insulator which can be specified on the **MATERIAL** statement. For VBET, E_{b1} is calculated similarly with the addition of the band gap. When tunneling from an electrode, the barrier height is set by the **BARR.DT** parameter on the **MATERIAL** statement. While the numerical evaluation of the integral in Equation 2-302 is slower than the analytical expression used in method 1, the result is much more accurate.

Direct Tunneling Method 3: Numerical Integration of Gundlach Tunneling Coefficient

Setting **DT.METH**=3 causes the integral in Equation 2-302 to be evaluated numerically using the Gundlach formula for the exact tunneling coefficient of a trapezoidal barrier [104]:

$$TC(E) = \frac{2(m_3/m_1)}{1+g} \qquad \text{Equation 2-305}$$

where

Equation 2-306

$$g(E) = \frac{\pi^2}{2}\left[\frac{m_3 k_1}{m_1 k_3}(Ai_0 Bi_d' - Bi_0 Ai_d')^2 + \frac{m_1 k_3}{m_3 k_1}(Bi_0' Ai_d - Ai_0' Bi_d)^2 + \frac{m_2^2 k_1 k_3}{m_1 m_3 \kappa^2}(Ai_0 Bi_d - Bi_0 Ai_d)^2 + \frac{m_1 m_3 \kappa^2}{m_2^2 k_1 k_3}(Ai_0' Bi_d' - Bi_0' Ai_d')^2\right]$$

Where *Ai* and *Bi* are the Airy functions. The Airy functions are evaluated as

$$Ai_0 = Ai\left(\frac{E_{b1} - E}{\varsigma}\right)$$
$$Ai_d = Ai\left(\frac{E_{b1} - qV_{ins} - E}{\varsigma}\right) \qquad \text{Equation 2-307}$$

where

$$\varsigma = (4\pi^2 h^2 q^2 V_{ins}^2/(2m_2 t_{ins}^2))^{1/3}. \qquad \text{Equation 2-308}$$

The wavevectors of the electrons in the two semiconductor/electrode regions are given by k_1 and k_3, and κ is given by:

$$\kappa = (8\pi^2 q m_2 V_{ins}/(h t_{ins}))^{1/3} \qquad \text{Equation 2-309}$$

For CBET, the effective tunneling mass of electrons in each of the three regions, m_1, m_2, and m_3, can be specified using the **ME.DT** parameter on the **MATERIAL** statement. For VBET, the effective tunneling mass of electrons in the heavy and light bands in region 1 can be set using **MHH.DT** and **MLH.DT,** respectively, on the **MATERIAL** statement. As in method 2, the barrier height for electrodes can be specified using the **BARR.DT** parameter on the **MATERIAL** statement. Method 3 is the most accurate, but slowest, method for evaluating the tunneling current.

However, as the number of nodes involved in tunneling is usually quite small, the time for the post-processing calculation of the direct tunneling current is typically small compared to the overall solution time.

Ionization Integrals

This section describes the following:

- Analysis of the avalanche breakdown of reversed biased junctions using the **IONIZATI** parameter on the **EXTRACT** statement
- Calculation of the electron and hole ionization integrals along potential gradient paths in a device structure using the **E.LINE** statement

Avalanche breakdown is analyzed by determining where and at what bias the ionization integrals exceed unity (corresponding to infinite carrier multiplication). Potential gradient paths and specified physical quantities, such as electric field or carrier concentration along the paths, may also be plotted.

Calculations

An ionization calculation begins by locating the potential gradient path that passes through a particular starting point within the semiconductor region of the device structure. The gradient path is followed in both directions away from the starting point until either

- The electric field becomes negligible or,
- The path intersects the boundary of the semiconductor.

For this path, ionization integrals for electrons (I_n) and holes (I_p) are calculated as

$$I_n = \int \alpha_{n,ii} \exp\left[-\int^{w} (\alpha_{n,ii} - \alpha_{p,ii}) \, dv\right] dw \qquad \text{Equation 2-310}$$

$$I_p = \int \alpha_{p,ii} \exp\left[-\int^{w} (\alpha_{p,ii} - \alpha_{n,ii}) \, dv\right] dw \qquad \text{Equation 2-311}$$

where

- v and w are distances along the gradient path
- $\alpha_{n,ii}$ and $\alpha_{p,ii}$ are given by Equations 2-273 and 2-274, respectively.

In the expression for I_n (I_p), the lower bounds for both integrals occur at the end of the gradient path where the potential is a minimum (maximum). The upper bound for the outer integral occurs at the end of the gradient path where the potential is a maximum (minimum).

Specification

To analyze the avalanche breakdown of reverse biased junctions, specify the **IONIZATI** parameter on the **EXTRACT** statement.

The electron and hole ionization integrals are calculated along gradient paths. These integrals are initiated at all nodes of the simulation mesh that lie within the rectangular region defined by the parameters **X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX**. Each calculated ionization integral is associated with one or more electrodes connected by material of the same doping type to the region where the potential gradient path terminates.

If a gradient path terminates in an isolated region with no contacting electrode, the ionization integral is associated with a special identifier named "floating". The maximum electron and hole ionization integrals associated with each electrode are calculated as

$$A_n = \max(\mathrm{I}_n) \qquad \text{Equation 2-312}$$

$$A_p = \max(\mathrm{I}_p) \qquad \text{Equation 2-313}$$

where the maximizations are performed over all integrals associated with each electrode. The peak electric field and its location along the maximum ionization path is also determined.

Avalanche breakdown of a reverse biased junction is indicated by values of A_n and A_p. These values exceed unity for electrodes that contact the doped region on either side of the junction.

The avalanche breakdown voltage for a junction can be determined by solving Poisson's equation for a series of reverse bias junctions expected to range from below to above the breakdown voltage. The avalanche breakdown voltage is the bias for which A_n and A_p equal unity.

Electric Field Lines

The **E.LINE** statement is used for the following:

- To calculate electron and hole ionization integrals along potential gradient paths

  Starting locations for the gradient paths are specified with the **X.START** and **Y.START** parameters. As with the above analysis, the peak electric field and its location along the path are determined.
- As part of a one- or two-dimensional plot sequence

  When used as part of a two-dimensional plot sequence, the potential gradient paths are plotted on the two-dimensional device cross-section. When used as part of a one-dimensional plot sequence, various physical quantities are plotted versus distance along the gradient path.

Band-to-Band Tunneling

The phenomenon of a valence band electron tunneling through the forbidden energy gap to the conduction band (leaving behind a hole) is known as band-to-band tunneling. Band-to-band tunneling has the following characteristics:

- It results in the generation of electron-hole pairs and occurs in regions of high electric field where the local band bending causes the tunneling probability to become significant.
- It is the effect used in the operation of some devices, such as Zener diodes
- It accounts for the generation of leakage currents in many submicron devices, and should be included in simulations where leakage is a concern

The carriers generated because of band-to-band tunneling can be included self-consistently in the solution of device equations by specifying the parameter **BTBT** on the **MODELS** statement. Band-to-band tunneling generation is computed in all the semiconductor regions where the current-continuity equations are solved.

The model used by Medici has the form of Kane's model (Reference [50]):

$$G^{BB} = \mathbf{A.BTBT}\frac{E^2}{E_g^{1/2}} \cdot \exp\left(-\mathbf{B.BTBT}\frac{E_g^{3/2}}{E}\right) \qquad \text{Equation 2-314}$$

In this expression, E is the magnitude of the electric field and E_g is the energy bandgap. A search along the direction opposite to the electric field is performed to determine whether there is an electric potential increase of at least $\frac{E_g}{q}$ for the band-to-band tunneling to occur.

The parameters **A.BTBT** and **B.BTBT** can be used as constants in the model. Their default values are shown in Table 2-9. These values can be modified using the **MATERIAL** statement.

Following the **MODELS** statement, parameters were recently added to improve the model implementation. The parameter **BT.MODEL** specifies the method for calculating E. The local field is used with the default value of 1. The average field

Table 2-9 Default Values for Band-to-Band Tunneling Parameters

Symbol	Default	Units
A.BTBT	3.5×10^{21}	$eV^{1/2}/\text{cm-s-}V^2$
B.BTBT	22.5×10^6	$V/\text{cm-(eV)}^{3/2}$
BT.MODEL	1	None
BT.LOCAL	1	None
BT.QUAD	False	None

Table 2-9 Default Values for Band-to-Band Tunneling Parameters

Symbol	Default	Units
BT.ATOL	1.0×10^{16}	$pairs/cm^3$
BT.RTOL	0.3	NONE
BT.TINY	5.0×10^{-4}	microns

along the tunneling path is used when **BT.MODEL** is 2. When **BT.MODEL** is 3, the average tunneling field is used in pre-exponential while a path integral in the form of Equation (3) on page 521 of Reference [4] is used in the exponential. The locality of band-to-band tunneling generation is specified using parameter **BT.LOCAL**. With the default value of 1, both electrons and holes are generated locally. When **BT.LOCAL** is set to 0, electrons are generated at the end of the tunneling path, as implied by tunneling physics. By setting the parameter **BT.QUAD**, a recursive refinement procedure is used to improved the accuracy of band-to-band tunneling calculation. When the refinement criteria are met, a triangle is divided into four congruent triangles and this process is repeated as necessary. There are three parameters for controlling the recursive refinement.

- Parameter **BT.ATOL** specifies the threshold generation rate above which refinement is considered.
- Parameter **BT.RTOL** specifies the relative change in the log10 of the generation rate above which refinement is carried out.
- Parameter **BT.TINY** specifies the minimum size above which a triangle is considered for refinement.

The total band-to-band tunneling generation rate at each node in the device can be determined by specifying the **BB.GENER** parameter on the **PRINT** statement. When **BT.LOCAL** is set to 0, the values generated are the larger of electron and hole generation rates.

One-dimensional line plots and two-dimensional contour plots of the band-to-band tunneling generation rate can be obtained by specifying the **BB.GENER** parameter on the **PLOT.1D** and **CONTOUR** statements, respectively.

Energy Balance Equations

Local carrier heating in high and spatial, rapidly varying electric fields is modeled using a self-consistent solution of the drift-diffusion and carrier energy balance equations.

The solution of the hydrodynamic model is initiated by specifying the parameters **ELE.TEMP** and **HOL.TEMP** on the **SYMBOLIC** statement. Both stationary and transient solutions are possible.

Two solution methods are available.

- The block-iterative algorithm alternately solves first the drift-diffusion equations and then one or two energy-balance equations.
- The other method solves the fully-coupled hydrodynamic system; it is initiated by invoking **COUP.ELE** or **COUP.HOL** on the **SYMBOLIC** statement.

Note:

Only one of these statements,* COUP.ELE, COUP.HOL, *or* COUP. *Currently, only one of* COUP.ELE, COUP.HOL *and* COUP.LAT *can be specified at one time (see the* SYMBOLIC *statement for explanations of* COUP.LAT*).

While the coupled method is generally faster and more robust than the uncoupled method, it is prone to divergence in case of bad initial guesses.

CAUTION

It is strongly recommended that you use the uncoupled method for the first two bias points. This allows for better initial guesses on subsequent biases thanks to the projection method (see "Initial Guesses" on page 2-61).

Implemented Differential Equations

The implemented hydrodynamic set of equations includes the standard drift-diffusion set of equations

$$\vec{\nabla}\cdot\varepsilon\vec{\nabla}\psi = -q(p-n+N_D^+-N_{\bar{A}})-\rho_s \qquad \text{Equation 2-315}$$

$$\frac{\partial n}{\partial t} = +\frac{1}{q}\vec{\nabla}\cdot\vec{J_n}-U_n \qquad \text{Equation 2-316}$$

$$\frac{\partial p}{\partial t} = -\frac{1}{q}\vec{\nabla}\cdot\vec{J_p}-U_p \qquad \text{Equation 2-317}$$

using generalized expressions for the electron and hole current densities:

$$\vec{J_n} = q\mu_n(u_n)\left[n\vec{E}+\vec{\nabla}(u_n n)\right] \qquad \text{Equation 2-318}$$

$$\vec{J_p} = q\mu_p(u_p)[p\vec{E}-\vec{\nabla}(u_p p)] \qquad \text{Equation 2-319}$$

An alternative form of the current density expressions can be used when solving the hydrodynamic set of equations by specifying **ET.MODEL** on the **MODELS** statement (References [51], [52], and [53]):

$$\vec{J_n} = q\mu_n n\vec{E}+q\mu_n\vec{\nabla}(u_n n)+nu_n\frac{\partial\mu_n(u_n)}{\partial u_n}\vec{\nabla}u_n \qquad \text{Equation 2-320}$$

$$\vec{J}_p = q\mu_p p\vec{E} - q\mu_p\vec{\nabla}(u_p p) - pu_p \frac{\partial\mu_p(u_p)}{\partial u_p}\vec{\nabla}u_p \quad \text{Equation 2-321}$$

The above set of equations, referred to as the Energy Transport Model, has been used to some advantage in small bipolar devices for reducing spurious velocity overshoot effects.

In addition, the hydrodynamic model includes the following electron and hole energy balance equations (References [54], [55], [56], and [57]), including transient effects and carrier cooling due to impact ionization:

$$\vec{\nabla}\cdot\vec{S}_n = \frac{1}{q}\vec{J}_n\cdot\vec{E} - \frac{3}{2}\left[n\frac{u_n - u_0}{\texttt{ELE.TAUW}} + \frac{\partial(nu_n)}{\partial t}\right] - \frac{1}{q}E_g G_n^{II} + H_n^R \quad \text{Equation 2-322}$$

$$\vec{\nabla}\cdot\vec{S}_p = \frac{1}{q}\vec{J}_p\cdot\vec{E} - \frac{3}{2}\left[p\frac{u_p - u_0}{\texttt{HOL.TAUW}} + \frac{\partial(pu_p)}{\partial t}\right] - \frac{1}{q}E_g G_p^{II} + H_p^R \quad \text{Equation 2-323}$$

$$\vec{S}_n = -\frac{5}{2}u_n\left[\frac{\vec{J}_n}{q} + \texttt{ELE.CQ}\ \mu_n n\vec{\nabla}u_n\right] \quad \text{Equation 2-324}$$

$$\vec{S}_p = +\frac{5}{2}u_p\left[\frac{\vec{J}_p}{q} - \texttt{HOL.CQ}\ \mu_p p\vec{\nabla}u_p\right] \quad \text{Equation 2-325}$$

$$H_n^R = U_{Auger}^n\left[\frac{E_g}{q} + \frac{3}{2}u_p\right] - \frac{3}{2}[\mathrm{SRHG}\cdot U_{SRH}u_{nL} + U_{Auger}^p u_n] \quad \text{Equation 2-326}$$

$$H_p^R = U_{Auger}^p\left[\frac{E_g}{q} + \frac{3}{2}u_n\right] - \frac{3}{2}[\mathrm{SRHG}\cdot U_{SRH}u_{pL} + U_{Auger}^n u_p] \quad \text{Equation 2-327}$$

where

$$u_{nL} = \begin{cases} u_n \text{ if } U_{SRH} > 0 \\ u_0 \text{ otherwise} \end{cases}, \quad u_{pL} = \begin{cases} u_p \text{ if } U_{SRH} > 0 \\ u_0 \text{ otherwise} \end{cases} \quad \text{Equation 2-328}$$

In Equations 2-322 through 2-325:

- $\vec{S}_n$ and $\vec{S}_p$ represent the electron and hole energy flow densities.

- u_n, u_p, and u_0 represent the electron, hole, and lattice thermal voltages kT_n/q, kT_p/q, and kT_0/q, respectively.

The value of SRHG in Equations and 2-327 can be 1 or 0 according to whether the value of flag **EB.SRH.G** is TRUE or FALSE, respectively. The default for **EB.SRH.G** is TRUE when the energy balance is fully coupled and FALSE otherwise.

This flag, along with the switch implemented in Equation 2-328, stabilizes the system of equations in difficult cases. **EB.SRH.G** is accessible on the **MODELS** statement. However, it cannot be set to TRUE when the more sensitive nonfully coupled method is used to solve EB.

When only one energy balance equation is solved, the unavailable carrier thermal voltage is replaced with the lattice thermal voltage in Equations through 2-328.

Compound Semiconductor EB Model

The electron and hole energy balance equations above are most appropriate for carrier transport in silicon. A different energy balance model suited for compound-semiconductors, such as GaAs, can be activated by specifying the **COMP.ET** parameter on the **MODELS** statement. As described in References [63] and [64], the equations are as follows:

$$\frac{1}{q}\left[\frac{5}{2}\vec{J}_n\cdot\vec{\nabla}u_n + u_n\vec{\nabla}\cdot\vec{J}_n\right] + \textbf{ELE.CQ}\ \vec{\nabla}\cdot\left(u_n\mu_n n\vec{\nabla}u_n\right)+\frac{1}{q}\vec{J}_n\cdot\vec{E} \qquad \text{Equation 2-329}$$
$$= \frac{3}{2}\left[n\frac{u_n-u_0}{\tau_{wn}} + \frac{\partial(nu_n)}{\partial t}\right]+\frac{E_g G_n^{II}}{q} - H_n^R$$

$$-\frac{1}{q}\left[\frac{5}{2}\vec{J}_p\cdot\vec{\nabla}u_p + u_p\vec{\nabla}\cdot\vec{J}_p\right] + \textbf{HOL.CQ}\ \vec{\nabla}\cdot\left(u_p\mu_p p\vec{\nabla}u_p\right)+\frac{1}{q}\vec{J}_p\cdot\vec{E} \qquad \text{Equation 2-330}$$
$$= \frac{3}{2}\left[p\frac{u_p-u_0}{\tau_{wp}} + \frac{\partial(pu_p)}{\partial t}\right]+\frac{E_g G_p^{II}}{q} - H_p^R$$

where τ_{wn} and τ_{wp} are the energy relaxation times for electrons and holes respectively. By default, τ_{wn} = **ELE.TAUW** and τ_{wp} = **HOL.TAUW**. An electron temperature-dependent model can be activated, however, by invoking **TMPTAUW** on the **MODELS** statement. For more information about this model, refer to "Energy Relaxation Times" on page 2-118.

COMP.ET is ignored in noncompound-semiconductor regions.

Boundary Conditions

The implemented boundary conditions for the energy balance equation are:

- Dirichlet boundary conditions at contacts:

$$u_n = u_0 \qquad u_p = u_0 \qquad \text{Equation 2-331}$$

- Homogeneous Neumann boundary conditions on all other boundaries ($\nabla_\perp$ denotes the derivative normal to the boundary):

$$\vec{\nabla}_\perp u_n = 0 \qquad \vec{\nabla}_\perp u_p = 0 \qquad \text{Equation 2-332}$$

These boundary conditions physically correspond to thermal equilibrium at contacts and to zero energy flow across insulating boundaries.

Physical Models

The solution of the energy balance equations allows the carrier temperature distribution inside a device to be determined for both electrons and holes. The knowl-

edge of carrier temperature makes it possible to more accurately model certain types of phenomena such as mobility and impact ionization. This section describes the carrier temperature models available in Medici.

Carrier Temperature-Based Mobility

The mobility model is based on the effective field approach. The effective field is the uniform electric field value that causes the carriers in a homogenous sample to attain the same temperature as at the current node in the device. Effective electric fields for electrons and holes $E_{eff,n}$, and $E_{eff,p}$ are calculated by solving locally the homogenous carrier temperature equations:

$$\mu_n(E_{eff,n})E_{eff,n}^2 = \frac{3}{2}\frac{u_n - u_0}{\tau_{wn}(u_n)} \quad \text{Equation 2-333}$$

$$\mu_p(E_{eff,p})E_{eff,p}^2 = \frac{3}{2}\frac{u_p - u_0}{\tau_{wp}(u_p)} \quad \text{Equation 2-334}$$

which are simply Equations 2-322 and 2-323 (on page 2-114) stripped of the spatially varying terms.

The default model assumes the velocity is saturated throughout the device, and, therefore, the $\dot{\mu}E_{eff}$ terms in the above equations are replaced with **VSAT**. This makes solving the equations straightforward. To turn this model on, invoke **TMPMOB** on the **MODELS** statement.

The carrier energy relaxation terms are also assumed to be constant in the default model. Introducing the effective field values obtained from Equations 2-333 and 2-334 into the field dependent model one obtains:

Equation 2-335

$$\mu_n(u_n) = \frac{\mu_{S,n}}{\left\{1 + [\alpha_n(u_n - u_0)]^{\mathbf{BETAN}}\right\}^{1/\mathbf{BETAN}}}, \quad \alpha_n = \frac{3\mu_{S,n}}{2\ \mathbf{VSATN}^2\ \mathbf{ELE.TAUW}}$$

Equation 2-336

$$\mu_p(u_p) = \frac{\mu_{S,p}}{\left\{1 + [\alpha_p(u_p - u_0)]^{\mathbf{BETAP}}\right\}^{1/\mathbf{BETAP}}}, \quad \alpha_p = \frac{3\mu_{S,p}}{2\ \mathbf{VSATP}^2\ \mathbf{HOL.TAUW}}$$

Alternatively, you may choose to activate the **HPMOB** model on the **MODELS** statement. In this case, Equations 2-169 and 2-170 on page 2-37 are used instead of Equations 2-162 and 2-163 on page 2-36.

An exact solution to the transcendental Equations 2-333 and 2-334 is always calculated in compound semiconductor regions by using the Newton-Raphson method and Equations 2-162 and 2-163 (on page 2-36). In order to use the exact solution in silicon-like materials, you have to specify the **EF.TMP** parameter on the **MODELS** statement. The effective field thus obtained is inserted in the selected

high field mobility model (Equations 2-162 and 2-163 on page 2-36, or Equations 2-169 and 2-170 on page 2-37, if **HPMOB** stated. In the case of compound semiconductors, Equations 2-165 and 2-166 on page 2-36). The energy relaxation time constants τ_{wn} and τ_{wp} are discussed in "Energy Relaxation Times" on page 2-118.

Solving the homogeneous energy-balance equations exactly leads to results consistent with drift-diffusion. Energy-balance and drift diffusion generate quasi-identical results for devices where the electric field does not vary significantly within the length of one carrier mean-free path.

In previous versions of Medici, negative values of the derivative of the mobility in compound-semiconductor regions were automatically discarded from the Jacobian matrix. Asserting **ND.MOB** on the **MODELS** statement now allows Medici to keep those terms.

Note:

It is recommended that* ND.MOB *be used with compound semi-conductors in order to accelerate convergence.

The temperature-dependent mobility model is enabled by the parameter **TMPMOB** on the **MODELS** statement. Alternatively, a local electric field-dependent mobility model can be used by specifying **FLDMOB**. Only one of the parameters **TMPMOB** and **FLDMOB** can be specified.

Energy Relaxation Times

Monte Carlo simulations show (References [64], [65], and [66]) that the energy relaxation time in compound-semiconductors varies over a quite wide range. The default values chosen by Medici are:

$$\tau_{wn} = \mathbf{ELE.TAUW} \qquad \tau_{wp} = \mathbf{HOL.TAUW} \qquad \text{Equation 2-337}$$

and the **ELE.TAUW** and **HOL.TAUW** defaults can be changed on the **MATERIAL** statement.

Carrier temperature-dependent models can be activated on the **MODELS** statement. Specifying **TMPTAUWN** (synonymous to the old **TMPTAUW**) will enable a model for the electron relaxation time and **TMPTAUWP** will enable a model for the hole relaxation time.

The empirical model used for electrons in III-V compounds is given in the equation below:

$$\tau_{wn} = \mathbf{WTN1} + (\mathbf{WTN0} - \mathbf{WTN1}) \cdot r^2 \cdot \exp(2 \cdot (1 - r)) \qquad \text{Equation 2-338}$$

where

$$r = \frac{T_n - T}{\mathbf{TNL}} \qquad \text{Equation 2-339}$$

For electrons in non-III-V materials and for holes in all materials, the models used are those suggested in (Reference [67] and Reference [93]):

Equation 2-340

$$\tau_{wn} = \begin{cases} \mathbf{WTN0} + \mathbf{WTN1} \cdot \xi_n + \mathbf{WTN2} \cdot \xi_n^2 + \mathbf{WTN3} \cdot \xi_n^3 + \mathbf{WTN4} \cdot \xi_n^4 + \mathbf{WTN5} \cdot \xi_n^5 & T_n < \mathbf{TNL} \\ \mathbf{WTNL} & T_n > \mathbf{TNL} \end{cases}$$

Equation 2-341

$$\tau_{wp} = \begin{cases} \mathbf{WTP0} + \mathbf{WTP1} \cdot \xi_p + \mathbf{WTP2} \cdot \xi_p^2 + \mathbf{WTP3} \cdot \xi_p^3 + \mathbf{WTP4} \cdot \xi_p^4 + \mathbf{WTP5} \cdot \xi_p^5 & T_p < \mathbf{TPL} \\ \mathbf{WTPL} & T_p > \mathbf{TPL} \end{cases}$$

where

Equation 2-342

$$\xi_n = \frac{T_n - T}{300} + 1$$

$$\xi_p = \frac{T_p - T}{300} + 1$$

and all coefficients in a bold font can be accessed on the **MATERIAL** statement.

Impact Ionization

Impact ionization is derived from the classical local electric field-dependent Chynoweth law and the homogeneous case temperature-field relation:

Equation 2-343

$$G_n^{II}(u_n) = \frac{\mathbf{N.IONIZA}}{q} \left|\vec{J}_n\right| \exp\left[-\left(\frac{u_{crit}}{u_n - u_0}\right)^{\mathbf{EXN.II}}\right]$$

with

Equation 2-344

$$u_{crit} = \frac{2 \ \mathbf{VSATN} \cdot \mathbf{ELE.TAUW}}{3} \cdot \mathbf{ECN.II}$$

No new fitting parameters are required when impact ionization is selected (the **IMPACT.I** parameter on the **MODELS** statement). By default, Medici uses the local electric field-dependent impact ionization model described in "Impact Ionization Analysis" on page 2-93. To use the temperature-dependent model, the parameter **II.TEMP** on the **MODELS** statement should also be specified. Expressions for holes are defined in an analogous manner. This formulation is consistent with the simplified effective field calculation introduced in "Carrier Temperature-Based Mobility" on page 2-117.

To calculate physically correct effective fields (and to obtain results consistent with drift-diffusion), the homogeneous energy balance equations must be solved exactly.

Equation 2-345

$$\mu_n(E_{eff,n})E_{eff,n}^2 = \frac{3}{2}\frac{u_n - u_0}{\tau_{wn}(u_n)} + E_g \cdot \mu_n(E_{eff,n})E_{eff,n}\,\alpha_{n,ii}(E_{eff,n})$$

Equation 2-346

$$\mu_p(E_{eff,p})E_{eff,p}^2 = \frac{3}{2}\frac{u_p - u_0}{\tau_{wp}(u_n)} + E_g \cdot \mu_p(E_{eff,p})E_{eff,p}\,\alpha_{p,ii}(E_{eff,p})$$

which are Equations 2-333 and 2-334 on page 2-117 plus impact ionization cooling terms. The impact ionization coefficients are those given in Equations 2-273 and 2-274 on page 2-95.

If **TMPMOB** and **II.TEMP** are specified, the effective fields obtained are used to calculate both the carrier mobilities and the impact ionization coefficients. In order to activate this model, the **EFI.TMP** parameter must be specified on the **MODELS** statement.

Note:

The only effect of enabling* EFI.TMP *is to add the cooling term to the homogeneous energy balance equations without precluding the choice of the impact ionization model itself. The field-dependent impact ionization model is compatible with* EFI.TMP*.

Thermally Enhanced Diffusion Current

In some cases it may be desirable to obtain an approximate energy balance solution by switching off the thermal diffusion term in the current density definition Equations 2-318 and 2-319 on page 2-113 (momentum balance equation). This can be accomplished by specifying **^TMPDIFF** on the **MODELS** statement. When **^TMPDIFF** is specified, the following expressions are used instead of Equations 2-318 and 2-319:

$$\vec{J}_n = q\mu_n(u_n)\left[n\vec{E} + \vec{\nabla}(u_0 n)\right]$$ Equation 2-347

$$\vec{J}_p = q\mu_p(u_p)\left[p\vec{E} - \vec{\nabla}(u_0 p)\right]$$ Equation 2-348

Specifying **^TMPDIFF** significantly decreases the CPU time required to obtain a solution.

Energy Balance Post-Processing Analysis

A post-processing mode for energy balance is available by specifying **EB.POST** on the **SYMBOLIC** statement. post-processing energy balance removes all dependence of the carrier concentrations on the electron (or hole) temperature. This results in some loss of accuracy but generally in much faster simulation times. The **EB.POST** option does the following:

- If temperature dependent impact ionization is enabled (**II.TEMP**=true), temperature-dependent impact ionization is disabled in the continuity equation assembly, but left enabled during the energy balance assembly. This is impor-

tant because impact ionization has a strong cooling effect on the carrier temperatures.

- If **TMPMOB** is enabled, **TMPMOB** is switched off and **FLDMOB** is enabled (and a warning given). The results of a simulation with **FLDMOB** are much closer to the desired simulation (i.e.,with **TMPMOB**) than a simulation with no mobility model.
- **TMPDIFF** is turned off and a warning is given.

Plotting Simulation Results

To plot the electron or hole temperature distribution, the parameters **ELE.TEMP** and **HOL.TEMP** on the **PLOT.1D**, **CONTOUR**, and **PLOT.3D** statements can be used. Temperature is plotted in units of Kelvin (K).

To plot electron and hole mean velocity ($v_n = -J_n/(qn)$, ($v_p = +J_p/(qp)$) the parameters **ELE.VEL** and **HOL.VEL** on the **PLOT.1D**, **CONTOUR**, and **PLOT.3D** statements can be used. Carrier mean velocity is plotted in units of cm/s.

Decoupled Solution

A decoupled approach (Figure 2-11) is chosen by default for the self-consistent solution of the hydrodynamic model. The algorithm consists of a solution of the drift-diffusion Equations 2-315 through 2-317 starting on page 2-113 for the classical variables ψ, n, and p. This is followed by a solution of the energy balance Equations 2-322 through 2-325 on page 2-114 for the electron and hole thermal voltages u_n and u_p. The new temperature distributions are used to re-evaluate ψ, n, and p. This procedure is repeated until the maximum updates of the electron and hole temperature normalized by the lattice temperature u_0 fall below the tolerance **EXT.TOLE**. The default value for **EXT.TOLE** is 0.01. It can be modified from this value on the **METHOD** statement.

The maximum number of energy balance iterations and the maximum number of block iterations are specified using the parameters **N.MAXEB** and **N.MAXBL** on the **METHOD** statement (default values are **N.MAXEB**=10 and **N.MAXBL**=25). In case of nonconvergence of either the drift-diffusion solution or energy balance solution, the applied bias voltage is reduced and the solution process is repeated.

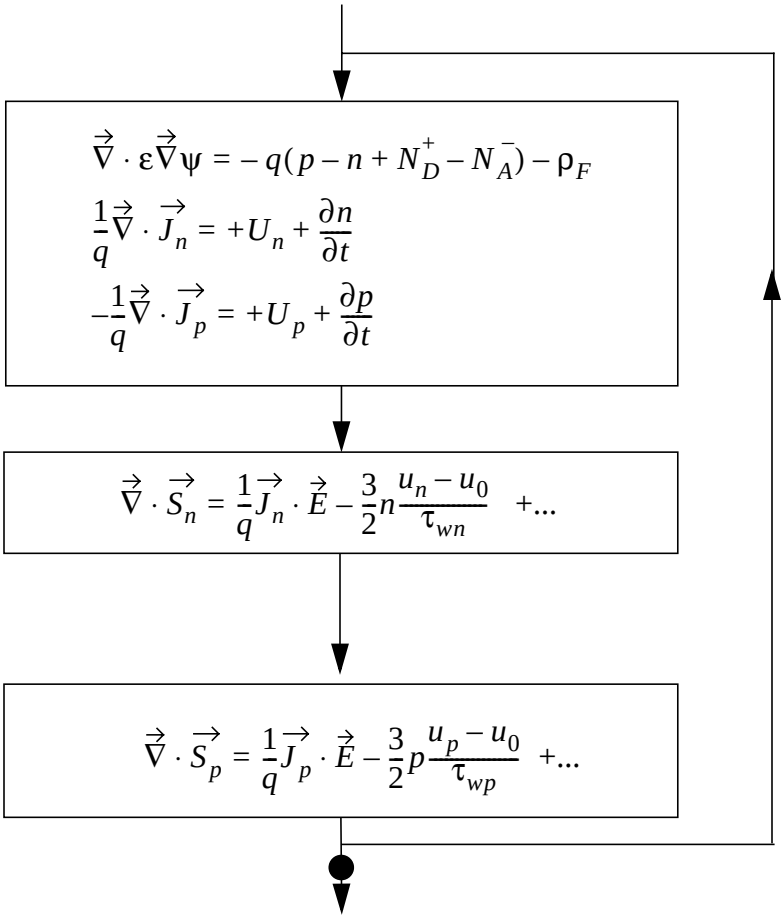


Figure 2-11 Solution strategy

Coupled Solution

The drift-diffusion equations (Equations 2-315 through 2-317 on page 2-113) can be solved simultaneously with one of the carrier temperature equations by specifying **COUP.ELE** for electrons (Equations 2-322, 2-324 starting on page 2-114) or **COUP.HOL** for holes (Equations 2-323, 2-325 or 2-330) on the **SYMBOLIC** statement.

The convergence of the carrier temperature updates is controlled, as in the uncoupled method, using **ETX.TOLE**. The tolerance on the right-hand-side of the energy balance equation is controlled with **ETR.TOLE** which is set by default to 1.0×10^{-18} *A*/micron and can be altered using the **METHOD** statement.

At the moment, only one of **COUP.ELE**, **COUP.HOL** and **COUP.LAT** can be used at a time. However, while one equation is solved using the coupled method either or both can be solved using the uncoupled method.

As mentioned at the beginning of this section, it is recommended to use the uncoupled method for the first two solutions (or bias points) and then switch to the coupled method.

Regional Specification of Semiconductor Parameters

Medici allows different semiconductor material parameters to be specified for different regions of the device structure. This capability provides an approximate method of treating structures that contain more than one semiconductor material or that have semiconductor properties that vary with location.

For example, a polysilicon emitter in a bipolar structure could be approximated by adjusting mobility and lifetime in the appropriate region of a silicon device. As a further example, there is experimental evidence that carrier ionization rates are lower at the surface in silicon devices than they are in the bulk (Reference [58]).

This behavior could be incorporated into a simulation by specifying a set of impact parameters for the surface of a device that are different than the set of parameters used in the bulk.

All parameters that appear on the **MATERIAL** and **MOBILITY** statement (see Chapter 3, "MATERIAL" on page 3-252 and "MOBILITY" on page 3-275) may be given different values in different regions. The **REGION** parameters on these statements are used to select the appropriate region number.

Programmable Device Advanced Application Module

This section describes the capabilities and use of the Programmable Device Advanced Application Module (PD-AAM) that is an option for use with the Medici program. The PD-AAM provides Medici with the ability to fully model the electrical characteristics of nonvolatile memory devices. The PD-AAM can simulate transient write and erase operations, as well as the effects of stored charge on device performance. The PD-AAM is particularly applicable for analyzing the floating gate structures found in EPROMs, EEPROMs, and flash EEPROMs.

The main features described in this section include:

- Charge boundary conditions for simulating floating gate structures.
- Implementation of the Fowler-Nordheim (FN) tunneling model to allow simulation of charging and discharging of floating gates.
- Enhancements to the gate current model for simulating the charging of floating gates using hot carrier injection.

Charge Boundary Condition

The charge boundary condition for a floating region is implemented as a floating electrode with a distributed boundary condition over all nodes of the floating electrode. For each floating electrode specified, a value of the net charge is used in the boundary condition.This net charge is either specified (for steady state) or generated by the tunneling and injection models in transient analysis and automatically placed on the floating electrodes. The charge boundary condition is specified as:

$$\oint \vec{D} \cdot d\vec{S} = Q \qquad \text{Equation 2-349}$$

where Q is the total charge on the floating electrode.

The above equation is applied to each floating electrode, with the integration being over the entire surface of the electrode. The equation forces the potential on the floating electrode to be adjusted to produce the correct total charge on the electrode.

The charge boundary condition is added as an extra equation in the Jacobian matrix of the Newton iteration. Because of this, it is necessary that the Newton method be specified on the **SYMBOLIC** statement if a floating region is present in the device. Either a one- or two-carrier solution may be performed.

Note:

Medici ***cannot perform AC analysis with floating regions. The Newton method must be specified when using charge boundary conditions.***

Fowler-Nordheim Tunneling

Fowler-Nordheim tunneling is requested by specifying the **FN.CUR** parameter on the **MODELS** or **SOLVE** statement. When **FN.CUR** is specified on the **MODELS** statement, Fowler-Nordheim currents are calculated self-consistently with other equations. Otherwise, Fowler-Nordheim currents are calculated in the post-processing mode. Fowler-Nordheim tunneling can be used in both steady-state and transient simulations.

- During steady-state simulations, Medici calculates the total tunneling current flowing to each electrode and semiconductor region in the structure.
- During transient simulations, the incremental charge obtained from integrating the tunneling current over time is immediately placed on the appropriate electrodes.

Either simulation obtains a self-consistent solution, allowing accurate analysis of programming characteristics.

Tunneling Model

The Fowler-Nordheim tunneling model used in the Medici program gives the tunneling current density as:

$$J_{FN} = \mathbf{A.FN} \cdot E_{insul}^2 \cdot \exp(-\mathbf{B.FN}/E_{insul}) \qquad \text{Equation 2-350}$$

where E_{insul} is the electric field in the insulator and **A.FN** and **B.FN** are model parameters that can be specified on the **MATERIAL** statement. The parameters **A.FN** and **B.FN** can be given separate values for:

- Tunneling that initiates in semiconductor regions
- Tunneling that initiates at floating electrodes

Use the **REGION** and **ELECTROD** parameters, respectively, on the **MATERIAL** statement. The default values (for all semiconductor regions and electrodes) are:

$$\mathbf{A.FN} = 6.32\text{x}10^{-7}\,\text{A/V}^2 \qquad \mathbf{B.FN} = 2.21\text{x}18^{8}\ \text{V/cm} \qquad \text{Equation 2-351}$$

After each solution is obtained for steady-state or transient simulations, the following occurs:

1. Each semiconductor-insulator or electrode-insulator interface is divided into segments by the mesh.
2. For each such segment, the FN current is calculated.
3. The model then tracks the electric field in the insulator by following the electric field vectors through the insulator starting from each interface segment.

   The tracking process determines where the tunneling current is likely to end up. The process assumes that the electrons that tunnel into the insulator follow the electric field. The tunneling current may end up on an electrode, or it may end up on a semiconductor region connected to an electrode.
4. The program totals these components, and the results are written to the standard output file.
5. During transient analysis, the total current flowing into each floating electrode is multiplied by the time step to calculate the charge added to each electrode during the current time step. The new value of charge is then used as the boundary condition for the next time step. This process allows accurate simulation of charging and discharging transients.

Hot Carrier Injection

The hot carrier injection (gate current) models available for use with the PD-AAM are described in "Gate Current Analysis" on page 2-97. These models can be used in both steady-state and transient analysis modes. Request gate current analysis by specifying the **GATE.CUR** parameter on the **SOLVE** statement.

The model calculates the hot carrier injection current density at each segment of each semiconductor-insulator interface as described in "Gate Current Analysis" on page 2-97. Then, analogous to the FN tunneling model, the injected current is tracked through the insulator to determine the regions or electrodes where this current ends up.

During transient simulations, the current due to hot carrier injection is treated in the same manner as described previously for the FN tunneling current. The current that flows into each floating electrode is multiplied by the time step to determine the charge added to each electrode. The new value of charge is then used as the boundary condition for the next time step.

Specifying Structure (Electrodes and Charges)

Floating electrodes are defined by specifying charge boundary conditions for each existing electrode of interest. The **CHARGE** parameter on the **CONTACT** statement is used for this purpose. The actual charge to be applied to the electrode is specified on the **SOLVE** statement in a manner analogous to specifying currents when current boundary conditions are used. All charge is specified in units of $C/\mu m$, unless cylindrical coordinates are specified, in which case charge is in units of Coulombs.

When a structure is read from TSUPREM-4, or if a previously created TMA PISCES-2B or Medici structure is read, floating silicon (or polysilicon) regions are easily converted to electrodes using the **REGION** parameter on the **ELECTRODE** statement. In the following example, region 3 is converted to a floating electrode named 5. A one-carrier solution is then performed with $10^{-15} C/\mu m$ of charge placed on the newly created floating electrode:

```
ELECTRODE   NAME=5      REGION=3
CONTACT     NAME=5      CHARGE
SYMBOLIC    NEWTON      CARRIERS=1  ELECTRONS
SOLVE       Q(5)=1E-15
```

Graphical Output

The calculated currents and charge at each electrode for both FN tunneling and hot carrier injection are saved in I-V log files for post-processing analysis. These quantities can be plotted as a function of time or bias using the **PLOT.1D** statement. The quantities of interest should be specified using the **X.AXIS** and/or **Y.AXIS** parameters where the available choices include:

FE<*name*>	FN tunneling current into Electrode <*name*>.
HE<*name*>	Hot carrier injection current into Electrode <*name*>.
QE<*name*>	The total charge on Electrode <*name*>.

Use the **EXTRACT** statement to give new names to any of these quantities, as well as to define their functions. The new names appear on the plot axis labels. See the description of the **EXTRACT** statement in Chapter 3, "EXTRACT" on page 3-166.

The following quantities can be requested on both the **PLOT.1D** and **CONTOUR** statements:

G.GAMN	Probability that an electron will be injected into the insulator
G.GAMP	Probability that a hole will be injected into the insulator
G.GAMT	Sum of **G.GAMN** and **G.GAMP**
G.IN	Hot electron current density injected into the insulator
G.IP	Hot hole current density injected into the insulator
G.IT	Sum of **G.IN** and **G.IP**

Circuit Analysis Advanced Application Module

This section describes the capabilities and use of the Circuit Analysis Advanced Application Module (CA-AAM) optionally available for use with the Medici program.

The CA-AAM provides you with the ability to embed multiple numerical device simulations within a single SPICE-like circuit simulation. Applicable to a variety of design and development problems, the CA-AAM allows semiconductor device performance to be predicted under realistic circuit operating conditions. In addition, circuits can be analyzed precisely by rigorously simulating critical device elements. The CA-AAM makes the accuracy of analog circuit simulation no longer limited by the quality of compact device models, and device analysis no longer dependent on user-estimated operating conditions.

The full set of linear and nonlinear SPICE elements is supported, including: resistors, capacitors, inductors, transformers, dependent and independent voltage and current sources, diodes, bipolar transistors, and MOSFET models.

Numerical Method Used in the CA-AAM

In the CA-AAM, the Kirchhoff equations describing the circuit and the semiconductor equations describing the devices (Poisson, continuity, energy balance, and lattice temperature) are solved as a coupled set (Reference [59]). Kirchhoff equations are formulated in the usual way, with modified nodal analysis being used for the voltage sources and the inductors.

The electron, hole, and displacement currents from the numerical model are added to the current at the circuit nodes connected to a numerical model. This is done in the same way as the lumped element boundary condition.

Construction of Circuit and Device Equations

The following example (see Figure 2-12) illustrates how the circuit and device equations are constructed. Elements *G1* and *G2* are voltage controlled, while *H1* is current controlled.

The equation at node *V2* is a simple Kirchhoff current equation:

$$0 = G2(V2) + G1(V2 - V1) + I1 \qquad \text{Equation 2-352}$$

To accommodate the current controlled source *H1*, a loop equation is needed:

$$0 = H1(I1) + V1 - V2 \qquad \text{Equation 2-353}$$

Finally, the equation at *V1* contains circuit and device quantities.

Equation 2-354

$$0 = G1(V1 - V2) - I1 + h1 \cdot J_{ca} + h2 \cdot J_{cb} + h3 \cdot J_{cd} + h4 \cdot J_{ce}$$

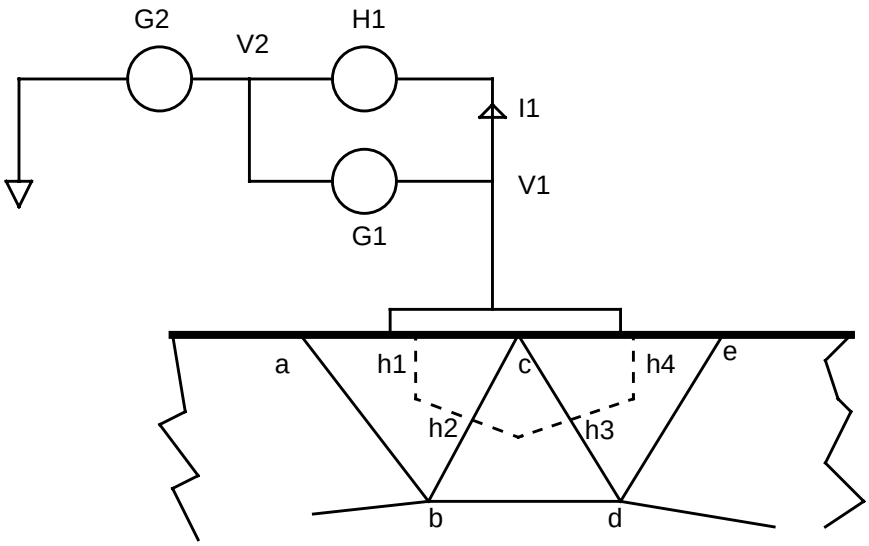


Figure 2-12 Connection of device and circuit

The terms J_x in Equation 2-354 are the current densities flowing along the indicated mesh line. In general:

- Each J_x is the sum of the electron, hole, and displacement current densities.
- Each J_x is in turn dependent upon the potential, electron, and hole concentrations, the electron and hole carrier temperatures, and the lattice temperature at each of the nodes on the element.

Therefore, current J_{cb} is dependent upon device variables at nodes *a*, *b*, *c*, and *d*.

The dotted lines in Figure 2-10 represent the perpendicular bisectors of the sides of the triangles. These form the integration volume that surrounds the electrode.

Multiple Device Matrix Description

Since the only coupling between the numerical devices occurs through the circuit, the Jacobian matrix that results from a circuit containing multiple devices has a bordered-block structure (see Figure 2-13).

In Figure 2-13 the portions of the matrix marked with a 0 contain all zeros and remain zero throughout the Gaussian elimination process. Matrix portions 1 and 2 are nearly identical to the original Jacobian matrices of the devices (without any circuit). Matrix *c* is the matrix of the circuit and matrices *a* and *b* couple the circuit to the devices. While matrices *a* and *b* have one dimension equal to the dimension of the entire system (the sum of all the device and circuit nodes), the smaller dimension is equal to the number of device terminals. In addition *a* and *b* are sparse.

An initial assessment suggests that this type of matrix leads to a N^2 dependence of the CPU time and memory requirements on the number of devices (*N*). However, as devices are added, the band width of the matrix does not increase, and the dependence of CPU time and memory is linearly dependent on the number of devices.

Note:

Convergence of a circuit containing several devices is usually as good as convergence of a single device.

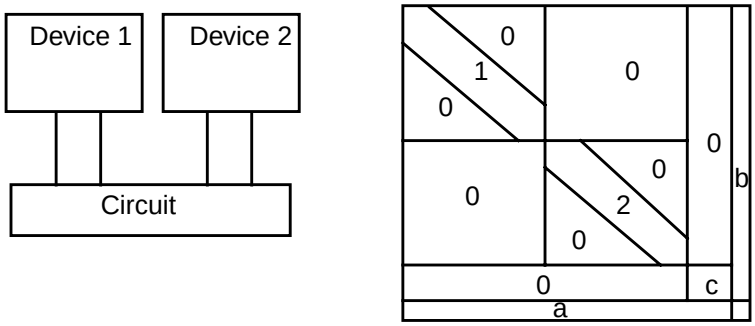


Figure 2-13 Circuit with two devices and Jacobian matrix

Lattice Temperature Advanced Application Module

This section describes the capabilities and use of the Lattice Temperature Advanced Application Module (LT-AAM) optionally available for use with the Medici program.

The LT-AAM module provides the ability to perform nonisothermal electrical analysis. By simultaneously solving the electrical and thermal equations, LT-AAM is particularly applicable to the analysis of power devices and submicron technologies where heat generation (Reference [60]) and dissipation is a concern.

Heat Equation Model

The increase in lattice temperature in a device due to current flow and recombination of carriers is included self-consistently in the solution of the device equations.

To use LT-AAM, specify the parameter **LAT.TEMP** on the **SYMBOLIC** statement. In this case, Medici solves the lattice heat equation in addition to Poisson's equation and the electron and hole current-continuity equations.

Heat Flow Equation

To compute the spatially dependent lattice temperature, the heat flow equation is used:

$$\rho c \frac{\partial T}{\partial t} = H + \vec{\nabla}(\lambda(T)\vec{\nabla}T) \qquad \text{Equation 2-355}$$

where

- ρ = the mass density of the material (g/cm^3)
- c = the specific heat of the material (J/g-K)
- H = the heat generation term (W/cm^3)

- λ = the thermal conductivity of the material (W/cm-K)

The heat generation in the semiconductor is modeled using:

$$H = H_n + H_p + H_U \qquad \text{Equation 2-356}$$

where

- H_n =lattice heating due to electron transport
- H_p =lattice heating due to hole transport
- H_U =lattice heating due to carrier recombination/generation

Overriding Default Modules

These terms have different expressions according to whether the energy balance equations are solved. The various possibilities are summarized in Equations 2-357 through 2-361.

The automatic choice of models based on the availability of solutions to the energy balance equations can be overridden by asserting **EBLT.HT** on the **MODELS** statement (default for **EBLT.HT** is TRUE) The override forces Medici to proceed as though T_n, T_p were not available.

$$H_n = \begin{cases} \dfrac{3k}{2} \cdot \dfrac{T_n - T}{\tau_{wn}} \cdot n & (1) \\ \vec{J}_n \cdot \vec{E}_n & (2) \end{cases} \qquad \text{Equation 2-357}$$

In Equation 2-357, equation (1) is used when T_n is calculated and equation (2) when it is not.

$$H_p = \begin{cases} \dfrac{3k}{2} \cdot \dfrac{T_p - T}{\tau_{wp}} \cdot p & (1) \\ \vec{J}_p \cdot \vec{E}_p & (2) \end{cases} \qquad \text{Equation 2-358}$$

In Equation 2-358, equation (1) is used when T_p is calculated and equation (2) when it is not.

Equation 2-359

$$H_R = \begin{cases} U_{SRH}\left[E_g + \dfrac{3k}{2}(T_n^L + T_p^L)\right] & (1) \\ U_{SRH}\left[E_g + \dfrac{3k}{2}T_{n,p}^L\right] + U_{Auger}^{p,n}\left[E_g + \dfrac{3k}{2}T_{n,p}\right] - \dfrac{3k}{2} \cdot U_{Auger}^{p,n}T & (2) \\ [U_{SRH} + U_{Auger}^n + U_{Auger}^p] \cdot (E_g + \mathrm{LTFACT} \cdot 3kT) & (3) \end{cases}$$

where

$$T_n^L = \begin{cases} T_n \text{ if } U_{SRH} > 0 \\ T \quad \text{otherwise} \end{cases}, \quad T_p^L = \begin{cases} T_p \text{ if } U_{SRH} > 0 \\ T \quad \text{otherwise} \end{cases} \qquad \text{Equation 2-360}$$

In Equation 2-359, equation (1) is used when T_n, T_p are both calculated, equation (2) is used when only one of T_n, T_p is calculated, and equation (3) is used in the DDE case. The switch implemented in Equation 2-360 prevents the solver from diverging when U_{SRH}<0.

The coefficient **LTFACT** (Equation 2-251) can take on one of two possible values (1 or 0) according to whether the value of the flag **3KT.LT** is TRUE or FALSE, respectively. The default value of this flag is TRUE and can be modified on the **MODELS** statement.

$$H_U = H_R - (E_g + 3kT)G_{II} \qquad \text{Equation 2-361}$$

Electric Field Terms

In the bulk of the semiconductor, the electric field terms $\vec{E}_n$, $\vec{E}_p$ are given by Equations 2-375 and 2-376. At the device contacts, however, electrons (holes) must move from the conduction (valence) band to the Fermi level to exit the device. Therefore, at the contacts, the electric field terms are given by:

$$\vec{E}_n = -\vec{\nabla}(E_c - E_{Fn}) \qquad \vec{E}_p = -\vec{\nabla}(E_v - E_{Fp}) \qquad \text{Equation 2-362}$$

Note:

Note that these electric fields are used only in Equations 2-357 through 2-361 and not in the current density relations.

When recombination with the tunneling model is enabled (**R.TUNNEL** on the **MODELS** statement), Equation 2-363 is used irrespective of the availability of carrier temperatures.

$$H_U = U_{SRH}\left(E_g - \frac{\int_0^{\Delta E_n}\left(E\frac{d\Gamma_n}{dE}\right)\,dE}{1+\Gamma_n} - \frac{\int_0^{\Delta E_p}\left(E\frac{d\Gamma_p}{dE}\right)\,dE}{1+\Gamma_p} \right) \qquad \text{Equation 2-363}$$

$$+(U_{Auger} - G^{II})\;E_g$$

Poisson's Equation

In Medici, Poisson's equation is usually solved for the intrinsic Fermi potential defined by the expression

$$-q\psi = E_c - \frac{E_g}{2} - \frac{kT}{2}\ln\left(\frac{N_c}{N_v}\right) \quad \text{Equation 2-364}$$

However, when the lattice temperature is not spatially constant, the intrinsic Fermi potential is no longer, in most cases, a solution to Poisson's equation. In this case, Poisson's equation is written as (Reference [6]):

$$\vec{\nabla}\cdot\varepsilon\vec{\nabla}(\psi-\theta) = -q(p-n+N_D^+-N_A^-)-\rho_s \quad \text{Equation 2-365}$$

where θ is the band structure parameter for the material and is given by

$$\theta = \chi + \frac{E_g}{2q} + \frac{kT}{2q}\ln\left(\frac{N_c}{N_v}\right) \quad \text{Equation 2-366}$$

Current Density Equations

If **EBLT.HT** is disabled (the default), Equations 2-318 and/or 2-319 starting on page 2-113 are used for the calculation of current density for the carrier(s) for which carrier temperatures are available. Otherwise, Equations 2-367 and/or 2-368 are used as suggested in Reference [47]. The mobility derivative term in Equations 2-317 and 2-318 is still subject to the value of **ET.MODEL**.

$$\vec{J_n} = qn\mu_n\vec{E}_n + k\mu_n(T\vec{\nabla}n + n\vec{\nabla}T) \quad \text{Equation 2-367}$$

$$\vec{J_p} = qp\mu_p\vec{E}_p - k\mu_p(T\vec{\nabla}p + p\vec{\nabla}T) \quad \text{Equation 2-368}$$

Numeric Methods

This section details the parameters and statements specified in the various methods of solving the heat equations.

Parameters and Statements

Specifying:

- **LAT.TEMP** on the **SYMBOLIC** statement causes Medici to solve the lattice heat equation. The default solves the heat equation in a decoupled manner from the other device equations, regardless of whether **GUMMEL** or **NEWTON** is specified.
- **CARRIERS** = 1 or **CARRIERS** = 2 on the **SYMBOLIC** statement causes Poisson's equation and one or both of the current-continuity equations to be solved along with the heat equation.
- **GUMMEL** on the **SYMBOLIC** statement causes Poisson's equation and the current-continuity equations to be solved in a decoupled manner.

- **NEWTON** on the **SYMBOLIC** statement causes Poisson's equation and the current-continuity equations to be solved in a completely coupled manner.
- **COUP.LAT** on the **SYMBOLIC** statement causes Poisson's equation, the current-continuity equations, and the heat equation to be solved in a completely coupled manner.

  The completely coupled method provides the fastest and most stable convergence. It also imposes increased memory requirements during the solution process, which limits the maximum number of mesh node used in comparison to those used in a decoupled method.
- **LTX.TOLE** (default = 1e-5) and **LTR.TOLE** (default = 1e-11 W/μm) on the **METHOD** statement specifies the error tolerances of the heat equation.

  These values represent the relative update tolerance for temperature and the RHS tolerance for the heat equation, respectively.

Decoupled Block Iterative Method

If most cases, **NEWTON** should be specified when **LAT.TEMP** is specified. In this case, the method used to solve the set of equations is a decoupled block iterative approach. The steps used in this method are as follows:

1. The program performs a coupled solution of Poisson's equation and the current-continuity equations to obtain ψ, n, and p at each node.
2. The heat equation is solved to obtain the temperature, T, at each node.

The program usually requires several iterations through these two sets of equations before global convergence is obtained.

Optimizing Convergence

When the heat equation is solved in a decoupled manner with the other device equations, convergence speed increases. To solve the equation in this manner, specify the factors by which the update and RHS tolerances of Poisson's equation and the continuity equations are increased for the intermediate iterations. The values of these factors are specified with the parameters **LTX.FACT** and **LTR.FACT** on the **METHOD** statement. Global convergence with the unaltered error tolerances is still required after the heat equation tolerances are satisfied.

Thermal Electrodes

Specify locations in the device structure where lattice temperature is to remain fixed by defining thermal electrodes.

Thermal electrodes are defined in exactly the same manner as electrical electrodes using the **ELECTRODE** statement, except that the parameter **THERMAL** must be added. For example, to define an electrode named "Sink" as a thermal electrode along the bottom of a structure, the following statement could be used:

```
ELECTRODE  NAME=Sink  BOTTOM  THERMAL
```

At least one thermal electrode is required in a simulation when the lattice heat equation is being solved.

The temperatures to use at thermal electrodes are specified using the ***T****(name)* parameters on the **SOLVE** statement, where *name* represents the electrode name. For example, to specify that the thermal electrode created in the example above should be set to 400 K during a solution, the following statement is used:

```
SOLVE  T(Sink)=400
```

If a temperature for a thermal electrode is not specified on a **SOLVE** statement, the previously specified temperature for that thermal electrode is used. If no previous temperature is specified, the temperature specified on the **MODELS** statement to initialize the lattice temperature in the structure is used.

Homogeneous Neumann boundary conditions are used at all boundaries not contacted by a thermal electrode.

Thermal Lumped Elements

Lumped thermal resistances and capacitances can be connected to thermal electrodes in much the same way that lumped electrical elements can be connected to electrical electrodes. To specify thermal resistances and capacitances, the parameters **R.THERMA** and **C.THERMA** on the **CONTACT** statement can be used.

Physical Models

The mass density, specific heat, and thermal conductivity of a material used in Equation 2-355 are given by the following expressions:

$$\rho = \mathbf{DENSITY} \qquad \text{Equation 2-369}$$

Equation 2-370

$$c = \mathbf{A.SP.HEA} + \mathbf{B.SP.HEA} \cdot T + \mathbf{C.SP.HEA} \cdot T^2 + \mathbf{D.SP.HEA} \cdot T^{-2} + \mathbf{F.SP.HEA} \cdot T^3 + \mathbf{G.SP.HEA} \cdot T^4$$

$$\lambda = [\mathbf{A.TH.CON} + \mathbf{B.TH.CON} \cdot T + \mathbf{C.TH.CON} \cdot T^2 + \mathbf{D.TH.CON} \cdot T^{\mathbf{E.THON.CON}}]^{-1} \qquad \text{Equation 2-371}$$

The parameters used in the above expressions can all be specified on the **MATERIAL** statement.

Heterojunction Device Advanced Application Module

This section describes the capabilities and use of the Heterojunction Device Advanced Application Module (HD-AAM) optionally available for use with the Medici program. Traditional means of increasing device speed include decreasing the device size, which can cause premature device breakdown, or modifying the dopants, which decreases current gain and emitter efficiency. Using the HD-AAM makes it possible to design faster devices without these negative side effects. Implementation of the HD-AAM includes:

- Band alignment at heterojunctions
- Velocity overshoot
- Virtual nodes
- Thermionic emission and tunneling currents through heterojunctions
- Graded (position dependent mole-fraction) and abrupt heterojunctions

Both abrupt and graded heterojunction devices are allowed. In the case of abrupt heterojunctions, the use of virtual nodes is recommended (see the Virtual Nodes section). Supported semiconductor materials include:

- Si, GaAs, Ge, $Si_{1-x}Ge_x$, $A1_xGa_{1-x}As$, $In_{1-x}Ga_xAs$, $A1_xIn_{1-x}As$, $GaAs_xP_{1-x}$, $In_xGa_{1-x}P$, and $InAs_{1-x}P_x$
- Arbitrary materials defined by the user

Typical applications include:

- The simulation and design of devices such as Si/SiGe and AlGaAs/GaAs Heterojunction Bipolar Transistors (HBTs)
- High Electron Mobility Transistor (HEMT) structures such as the GaAs/AlGaAs/InGaAs HEMT analyzed in Chapter 14, High Electron Mobility Transistor Simulation on page 14-7.

The HD-AAM may be combined with the other AAMs to provide an extremely powerful tool for analyzing the behavior of a wide variety of heterojunction devices and circuits. Some possibilities are:

- Accurate analysis of deep submicron HBTs, by including solutions of the energy balance equation in the analysis when using the HD-AAM.
- Understanding the behavior of heterojunction power devices that experience significant lattice heating by combining the HD-AAM with the LT-AAM.
- Analysis of the behavior of a circuit containing one or more high speed heterojunction devices using the CA-AAM with the HD-AAM.

Material Parameters

The parameters available for describing the properties of the heterojunction are the same parameters available for describing the properties of the materials that meet at the heterojunction. Some of these include:

- Energy bandgap parameters (**`EG300`**, **`EGALPH`**, and **`EGBETA`**)
- Electron affinity (**`AFFINITY`**)
- Densities of state (**`NC300`** and **`NV300`**)
- Various parameters for describing recombination, mobility or other qualities.

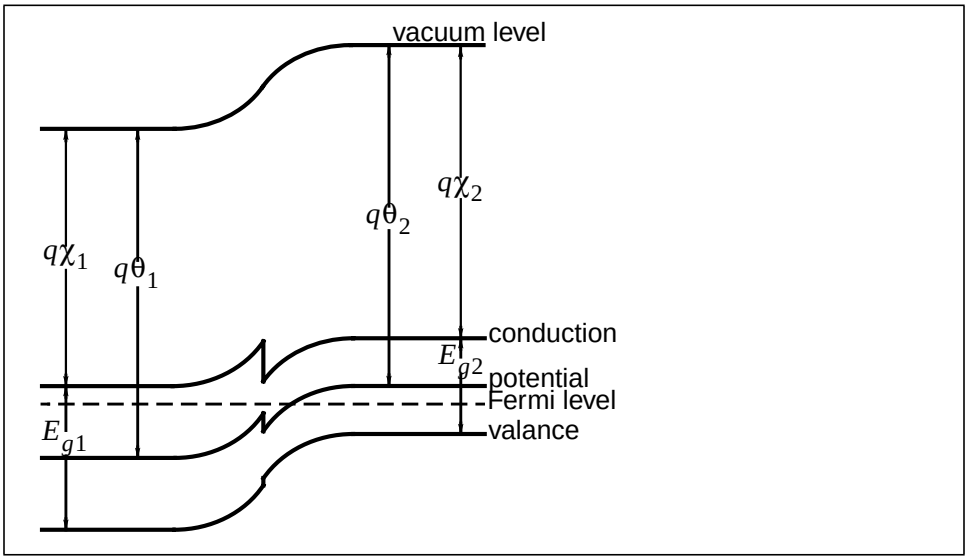


Figure 2-14 Band diagram with two different materials forming a heterojunction

Band Diagram

Figure 2-14 shows a band diagram under equilibrium conditions for a typical heterojunction involving two materials. It also identifies some of the physical parameters for the materials. In the figure:

- χ is the electron affinity
- E_g is the energy bandgap
- θ is the band structure parameter given by Reference [6].

$$\theta = \chi + \frac{E_g}{2q} + \frac{kT}{2q}\ln\left(\frac{N_c}{N_v}\right) \qquad \text{Equation 2-372}$$

Materials

For compound materials such as $Si_{1-x}Ge_x$ and $Al_xGa_{1-x}As$, the mole fraction, *x*, can be specified by using the **X.MOLE** and related parameters on the **REGION** statement that is used to define the material region. The mole fraction can either be constant or linearly graded in the region.

For example, to describe an HBT consisting of $Al_{0.3}Ga_{0.7}As$ as the emitter on top of a GaAs substrate with a 0.03 micron transition region from AlGaAs to GaAs, the following statements could be used:

```
REGION  NAME=1 ALGAAS Y.MIN=0.00 Y.MAX=0.42 X.MOLE=0.3
REGION  NAME=2 ALGAAS Y.MIN=0.42 Y.MAX=0.45 X.MOLE=0.3
+       X.END=0.0    Y.LINEAR
REGION  NAME=3 GAAS   Y.MIN=0.45 Y.MAX=1.2
```

Device Equations for Heterojunctions

The intrinsic Fermi potential (usually referred to as "potential" in Medici) is in general not a solution to Poisson's equation when solving for structures that contain heterojunctions. The vacuum level, however, is a solution to Poisson's equation. The intrinsic Fermi potential and the vacuum level are related by

$$\psi_{vacuum} = \psi - \theta \qquad \text{Equation 2-373}$$

where

- ψ is the intrinsic Fermi potential
- θ is the band structure parameter given by Equation 2-372

Note that if the band structure parameter is spatially constant, then ψ will be a solution to Poisson's equation. However, this is seldom the case in structures containing heterojunctions due to differences in bandgap, electron affinity, and densities of states in adjacent materials.

For this reason, Poisson's equation must be written in the form

$$\vec{\nabla} \cdot \varepsilon \vec{\nabla}(\psi - \theta) = -q(p - n + N_D^+ - N_A^-) - \rho_s \qquad \text{Equation 2-374}$$

Note that this is the same form of Poisson's equation that is given in the description of the Lattice Temperature AAM. In structures exhibiting a spatial dependence on lattice temperature, θ is usually not constant due to temperature dependencies of bandgap, electron affinity, and densities of states.

The form of the continuity equations remains unchanged for heterojunctions except that the electric field terms $\vec{E}_n$ and $\vec{E}_p$ in the transport equations must account for gradients in conduction and valence band edges (Reference [75]). Note that $E_c = -q(\psi_{vacuum} + X)$.

$$\vec{E}_n = \frac{\vec{\nabla} E_c}{q} - \frac{kT}{q}\vec{\nabla}\left(\ln(N_c) - \ln(T^{3/2})\right) \qquad \text{Equation 2-375}$$

$$\vec{E}_p = \frac{\vec{\nabla} E_v}{q} + \frac{kT}{q}\vec{\nabla}\left(\ln(N_v) - \ln(T^{3/2})\right) \qquad \text{Equation 2-376}$$

Models for Compound Materials

This section briefly describes various mole fraction dependent models available for compound materials such as $Si_{1-x}Ge_x$.

Permittivity

The dependence of permittivity on mole fraction is described by the expression

$$\varepsilon(x) = (\mathbf{PERMITTI} + \mathbf{EPS.X1} \cdot x + \mathbf{EPS.X2} \cdot x^2)\,\varepsilon_0 \qquad \text{Equation 2-377}$$

where the parameters **PERMITTI**, **EPS.X1**, and **EPS.X2** can be specified on the **MATERIAL** statement and ε_0 is the vacuum permittivity.

Density of States

The dependence of conduction and valence band density of states on mole fraction is given by the expressions

$$N_C(x, T) = N_C(T) \cdot \left[1 + \mathbf{NC.0}\left(\exp\left(\frac{-\mathbf{NC.E}\ x}{kT}\right) - 1\right)\right] \quad \text{Equation 2-378}$$

$$N_V(x, T) = N_V(T) \cdot \left[1 + \mathbf{NV.0}\left(\exp\left(\frac{-\mathbf{NV.E}\ x}{kT}\right) - 1\right)\right] \quad \text{Equation 2-379}$$

where the parameters **NC.0**, **NC.E**, **NV.0** and **NV.E** can be specified on the **MATERIAL** statement and $N_C(T)$ and $N_V(T)$ are described in "Bandgap and Effective Density of States" on page 2-8.

Energy Bandgap Choices

The selection of an energy bandgap model to use with a specific material or in a specific region of the device structure is made by specifying **EG.MODEL**=<n> on the **MATERIAL** statement, where <n> is an integer that represents the model of choice.

EG.MODEL=1 is the default used for all noncompound materials. This choice calculates a temperature dependent bandgap as described in "Bandgap and Effective Density of States" on page 2-8. For compound materials, a model specific to the material is used. Available choices for **EG.MODEL** are given below.

Energy Bandgap and Electron Affinity

Selecting **EG.MODEL**=0 on the **MATERIAL** statement invokes a mole fraction dependent energy bandgap given by

$$E_g(x, 300) = \mathbf{EG300} + \mathbf{EG.X1} \cdot x + \mathbf{EG.X2} \cdot x^2 \quad \text{Equation 2-380}$$

The electron affinity is given by

$$affinity(x) = \mathbf{AFFINITY} + \mathbf{AF.X1} \cdot x + \mathbf{AF.X2} \cdot x^2 \quad \text{Equation 2-381}$$

The parameters in these expressions are specified on the **MATERIAL** statement.

Lattice Temperature-Dependent Energy Bandgap

Selecting **EG.MODEL**=1 on the **MATERIAL** statement invokes the temperature-dependent energy bandgap model described in "Bandgap and Effective Density of States" on page 2-8. The expression is repeated here for completeness. This is the default for noncompound materials.

Equation 2-382

$$E_g(x, T) = E_g(x, 300) + \mathbf{EGALPH}\left[\frac{300^2}{300 + \mathbf{EGBETA}} - \frac{T^2}{T + \mathbf{EGBETA}}\right]$$

Although this is the default for noncompound materials, because of the mole fraction dependence built in to $E_g(x, 300)$, it can be used for compound materials

such as $Si_{1-x}Ge_x$ as well, with the appropriate selection of parameters. For example, a model for $Si_{1-x}Ge_x$ such as this,

$$E_g^{SiGe}(x, T) = E_g^{Si}(T) - 0.74x \qquad \text{Equation 2-383}$$

simply requires the specification of **EG.X1**=-0.74 on the **MATERIAL** statement.

Strained Energy Bandgap Model

Selecting **EG.MODEL**=2 on the **MATERIAL** statement invokes an energy bandgap model for strained $Si_{1-x}Ge_x$ (piecewise linear approximation to the lower curve of the strain split data shown in Figure 2 of Reference [61]). If the HD-AAM is enabled, this is the default for all regions that were specified as **SIGE** with **REGION** statements.

The data shown in Reference [61] is for T=90K. A conversion to the actual temperature is accomplished using the expressions given in "Bandgap and Effective Density of States" on page 2-8 after calculating the mole fraction dependence as shown below:

Equation 2-384

$$E_g(x) = \begin{cases} E_g(90) - 4.0(E_g(90) - 0.950)\ x & , x \le 0.25 \\ 0.950 - 0.66666\ (x - 0.25) & , 0.25 < x \le 0.40 \\ 0.850 - 0.57500\ (x - 0.40) & , 0.40 < x \le 0.60 \\ 0.735 - 0.43333\ (x - 0.60) & , 0.60 < x \le 0.75 \\ 0.670 & , x > 0.75 \end{cases}$$

Unstrained Energy Bandgap Model

Selecting **EG.MODEL**=3 on the **MATERIAL** statement invokes an energy bandgap model for unstrained $Si_{1-x}Ge_x$ (piecewise linear approximation to the unstrained bulk data shown in Figure 2 of Reference [61]). The temperature and mole fraction dependence can be expressed as

$$E_g^{SiGe}(x, T) = (1 - f(x))\ E_g^{Si}(T) + f(x)\ E_g^{Ge}(T) \qquad \text{Equation 2-385}$$

where the temperature dependence of E_g^{Si} and E_g^{Ge} is calculated using the expressions given in "Bandgap and Effective Density of States" on page 2-8. The mole fraction dependence is given by

Equation 2-386

$$f(x) = \begin{cases} 0.9375\ x & , x \le 0.40 \\ 0.375 - 0.108696\ (x - 0.40) & , 0.40 < x \le 0.86 \\ 0.425 - 4.107143\ (x - 0.86) & , x > 0.86 \end{cases}$$

Models Specific to III-V Compound Semiconductors

Selecting **EG.MODEL**=4 on the **MATERIAL** statement invokes mole-fraction dependent bandgap and electron affinity models specific to III-V compound semiconductors.

If the HD-AAM is enabled, this is the default for all regions that were specified as **ALGAAS**, **INGAAS**, **ALINAS**, **GAASP**, **INGAP**, or **INASP** with **REGION** statements.

The energies at the bottom of three energy bands (Γ, X, L) are modeled as functions of the mole-fraction and lattice temperature. The energies are referenced to the top of the highest valence-band and are evaluated at every node of the mesh. The bottoms of the three bands are calculated as:

Equation 2-387

$$E_\Gamma(x,T) = \mathbf{EG300} + \mathbf{EG.X0} + \mathbf{EG.X1}\cdot x + \mathbf{EG.X2}\cdot x^2 + \mathbf{EG.X3}\cdot x^3 + \mathbf{EG.X4}\cdot x^4 + \left[\frac{300^2}{300+\mathbf{EGBETA}} - \frac{T^2}{T+\mathbf{EGBETA}}\right]\cdot(\mathbf{EGALPH} + \mathbf{EGGAMM}\cdot x)$$

Equation 2-388

$$E_X(x,T) = \mathbf{EG300} + \mathbf{EG.X5} + \mathbf{EG.X6}\cdot x + \mathbf{EG.X7}\cdot x^2 + \mathbf{EG.X8}\cdot x^3 + \mathbf{EGX9}\cdot x^4 + \left[\frac{300^2}{300+\mathbf{EGBEX}} - \frac{T^2}{T+\mathbf{EGBEX}}\right]\cdot(\mathbf{EGALX} + \mathbf{EGGAX}\cdot x)$$

Equation 2-389

$$E_L(x,T) = \mathbf{EG300} + \mathbf{EG.X10} + \mathbf{EG.X11}\cdot x + \mathbf{EG.X12}\cdot x^2 + \mathbf{EG.X13}\cdot x^3 + \mathbf{EG.X14}\cdot x^4 + \left[\frac{300^2}{300+\mathbf{EGBEL}} - \frac{T^2}{T+\mathbf{EGBEL}}\right]\cdot(\mathbf{EGALL} + \mathbf{EGGAL}\cdot x)$$

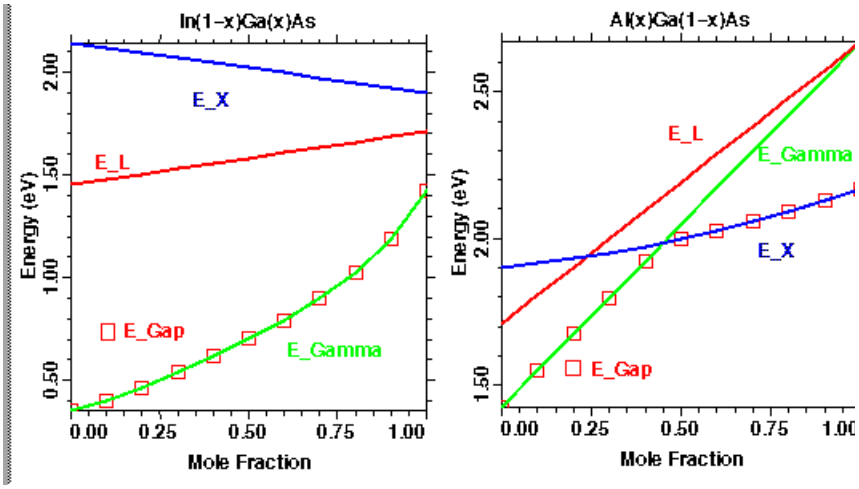


Figure 2-15 Energy bands for AlGaAs and InGaAs

The gap energy is then defined as the minimum of the three previously defined energies:

$$E_{gap}(x, T) = min(E_{\Gamma}(x, T), E_{X}(x, T), E_{L}(x, T)) \quad \text{Equation 2-390}$$

The mole fraction dependence of the three energy bands and the gap for the particular cases of $Al_xGa_{1-x}As$ and $In_{1-x}Ga_xAs$ are shown in Figure 2-15.

The electron affinity is modeled as:

Equation 2-391

$$\chi(x) = \mathbf{AFFINITY} + \begin{cases} \mathbf{AF.X0} + \mathbf{AF.X1} \cdot x + \mathbf{AF.X2} \cdot x^2 & , x \le \mathbf{AF.XL} \\ \mathbf{AF.X3} + \mathbf{AF.X4} \cdot x + \mathbf{AF.X5} \cdot x^2 & , x > \mathbf{AF.XL} \end{cases}$$

Carrier mass, carrier density of states, and Richardson's Constant models specific to compound semiconductors will be enabled if the value of the **EM.MODEL** model selection parameter appearing on the **MATERIAL** statement is 1(default for all III-V materials). The electron mass accounts for the contributions of electron masses from all three bands (mr_{Γ}^{e}, mr_{X}^{e}, mr_{L}^{e}):

Equation 2-392

$$mr_{\Gamma}^{e}(x, T) = \left[(\mathbf{MEG} + \mathbf{MEG.X1} \cdot x)^{3/2} \cdot \exp\left[\frac{E_{gap}(x, T) - E_{\Gamma}(x, T)}{k \cdot T} \right] \right]^{2/3}$$

Equation 2-393

$$mr_{X}^{e}(x, T) = \left[(\mathbf{MEX} + \mathbf{MEX.X1} \cdot x)^{3/2} \cdot \exp\left[\frac{E_{gap}(x, T) - E_{X}(x, T)}{k \cdot T} \right] \right]^{2/3}$$

Equation 2-394

$$mr_{L}^{e}(x, T) = \left[(\mathbf{MEL} + \mathbf{MEL.X1} \cdot x)^{3/2} \cdot \exp\left[\frac{E_{gap}(x, T) - E_{L}(x, T)}{k \cdot T} \right] \right]^{2/3}$$

by defining $m^e(x,T)$ as:

$$m^{e}(x, T) = \left[mr_{\Gamma}^{e}(x, T)^{3/2} + mr_{X}^{e}(x, T)^{3/2} + mr_{L}^{e}(x, T)^{3/2} \right]^{2/3} \quad \text{Equation 2-395}$$

The hole mass model accounts for light and heavy holes:

$$m^{h}(x) = \left[(\mathbf{MH0} + \mathbf{MH0.X1} \cdot x)^{3/2} + (\mathbf{ML0} + \mathbf{ML0.X1} \cdot x)^{3/2} \right]^{2/3} \quad \text{Equation 2-396}$$

If **EM.MODEL** is set to 0, $m^e(x,T)$ and $m^h(x)$ will be assigned the constant values **EL.EMAS** and **HO.EMAS**.

If **EM.MODEL** is 1, the conduction band and valence band densities of states N_c and N_v, respectively, are expressed as:

$$N_C(x, T) = 2 \cdot \left[\frac{2 \cdot \pi \cdot m^e(x, T) \cdot k \cdot T}{h^2}\right]^{3/2} \quad \text{Equation 2-397}$$

$$N_v(x, T) = 2 \cdot \left[\frac{2 \cdot \pi \cdot m^h(x) \cdot k \cdot T}{h^2}\right]^{3/2} \quad \text{Equation 2-398}$$

If **EM.MODEL** is set to 0,

$$N_c(T) = \mathbf{NC300} \cdot \left[\frac{T}{300}\right]^{3/2} \quad \text{Equation 2-399}$$

$$N_v(T) = \mathbf{NV300} \cdot \left[\frac{T}{300}\right]^{3/2} \quad \text{Equation 2-400}$$

The electron and hole Richardson's Constants (A_e^* and A_h^*, respectively) are calculated as:

$$A_e^*(x, T) = \frac{4 \cdot \pi \cdot e \cdot k^2}{h^3} \cdot m^e(x, T) \quad \text{Equation 2-401}$$

$$A_h^*(x) = \frac{4 \cdot \pi \cdot e \cdot k^2}{h^3} \cdot m^h(x) \quad \text{Equation 2-402}$$

Whereas, if **EM.MODEL** is set to 0, A_e and A_h are set to **ARICHN** and **ARICHP**, respectively. A_e and A_h appear in the calculation of thermionic emission and tunneling currents that flow across heterojunctions.

Mobility Models Specific to III-V Compound Semiconductors

The low-field and high-field mobility model parameters appearing in all the equations below have been fitted to data obtained from recent publications for $A1_xGa_{1-x}As$, $In_{1-x}Ga_xAs$, $A1_xIn_{1-x}As$, $GaAs_xP_{1-x}$, $In_xGa_{1-x}P$, and $InAs_{1-x}P_x$. Their values can be modified on the **MOBILITY** statement on a region by region basis.

In all the equations below, the mole fraction and lattice temperature are represented by x and T, respectively.

Low-Field Mobility Model

A modified version of the **ANALYTIC** model that is mole fraction-dependent is used for electrons:

$$\mu_{0_n}(x,T) = \mu_n^{min}(x) + \frac{\left[\mu_n^{max}(x)\cdot\left(\frac{T}{300}\right)^{\mathbf{NUN}} - \mu_n^{min}(x)\right]}{\left[1+\left(\frac{T}{300}\right)^{\mathbf{XIN}}\cdot\left[\left(\frac{N_{total}}{\mathbf{NREFN}}\right)^{\mathbf{ALPHAN}} + \left(\frac{N_{total}}{\mathbf{NREFN2}}\right)^3\right]\right]}$$

Equation 2-403

where $N_{total}(x)$ is the local total impurity concentration and $\mu_n^{min}(x)$ and $\mu_n^{max}(x)$ are defined as:

$$\mu_n^{min}(x) = \mathbf{MUN.MIN}(1 + \mathbf{MIN.X1}\cdot x + \mathbf{MIN.X2}\cdot x^2)$$

Equation 2-404

$$\mu_n^{max}(x) = \mathbf{MUN.MAX}(1 + \mathbf{MAN.X1}\cdot x + \mathbf{MAN.X2}\cdot x^2)$$

Equation 2-405

This model is enabled by stating the **ANALYTIC** flag on the **MODELS** statement. The theoretical data from [6] and [7] was used to fit the coefficients appearing in the equations above. Note that the temperature dependence for ternary compounds is disabled for now, i.e., **NUN**=0 and **XIN**=0. No new data is available for holes yet.

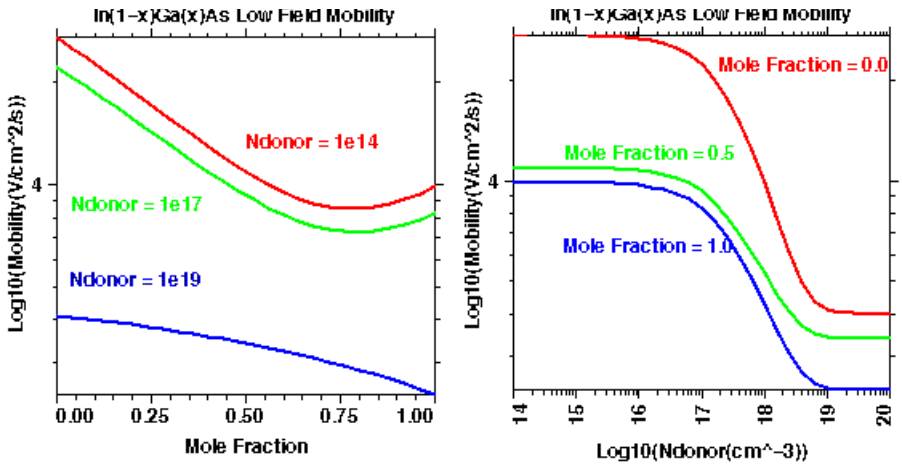


Figure 2-16 InGaAs low-field mobility as a function of mole fraction and doping

High-Field Mobility Model

If the value of the **FLDMOB** model switch, accessible on the **MOBILITY** statement, is 2, (default for III-V materials), a modified version of the III-V specific mobility degradation model is used for electrons:

$$\mu_n(x, T, E_{\|,n}) = \frac{\mu_{n0}(x, T) + \frac{v_n^{sat}(x)}{E_{\|,n}} \cdot \left(\frac{E_{\|,n}}{E_0(x)}\right)^4}{1 + \left(\frac{E_{\|,n}}{E_0(x)}\right)^4} \qquad \text{Equation 2-406}$$

where

Equation 2-407

$$v_n^{sat}(x) = \mathbf{VSATN}(1 + \mathbf{VSN.X1} \cdot x + \mathbf{VSN.X2} \cdot x^2)$$

$$E_0(x) = \mathbf{E0.N}(1 + \mathbf{EN.X1} \cdot x + \mathbf{EN.X2} \cdot x^2) \qquad \text{Equation 2-408}$$

Default coefficient values for the new model described above are only available for electrons so far. This model is designated to be used when the **FLDMOB** flag is true in the **MODELS** statement. This model is also be used if an energy balance simulation is performed and **TMPMOB** is stated on the **MODELS** statement.In that case an electron temperature-dependent effective field, $E_{eff,n}(T_n)$, will be calculated and used in $\mu_n(x, T, E_{eff,n}(T_n))$.

The values of the default coefficients were obtained by best-fitting the above equations to the data from Reference [8].

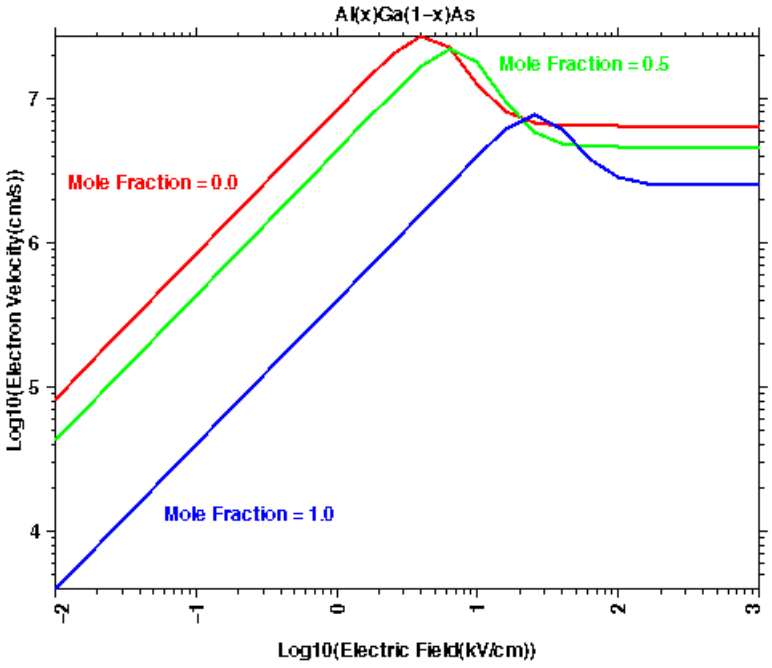


Figure 2-17 Field-dependent electron velocity in AlGaAs for $N_d=10^{17}$ cm^{-3}

Virtual Nodes

An abrupt heterojunction is formed by contact between two or more different materials. Medici automatically places nodes on heterojunctions. All material properties and system variables, such as potential or carrier concentrations, at such nodes are multivalued. Carrier transport across the heterojunction is governed by thermionic emission and tunneling, unlike transport in continuous media, which is governed by drift and (carrier and/or temperature) diffusion.

To correctly model the phenomena taking place at abrupt heterojunction nodes, a separate data structure associated with the node has to be created for each material as depicted in Figure 2-18. These data structures are referred to as *virtual nodes*, which are created by Medici if the **`VIRTUAL`** flag is stated on the **`MESH`** statement or the **`SYMBOLIC`** statement. Each such node has the properties of the material in which it is located. However, its system variable values may differ from those of other virtual nodes at the same physical location. This allows for discontinuous potentials, such as Fermi levels.

Every two virtual nodes belonging to regions sharing a side are connected by a side through which the heterojunction current flows (V_S1, V_S2, ... in Figure 2-18). When three or more semiconductor materials meet at a node not completely surrounded by semiconductor regions, two or more couples of adjacent virtual nodes are left with no semiconductor surfaces in common. Such nodes are connected by *ghost virtual sides*, with zero cross sections through which no current can flow.

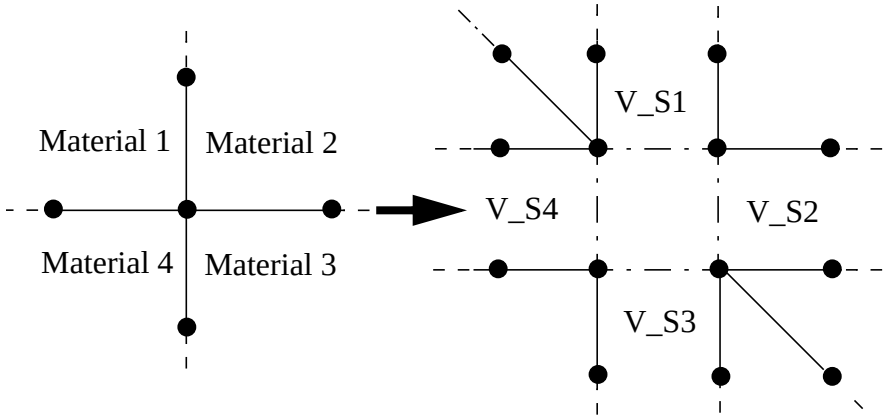


Figure 2-18 Generation of virtual nodes at an abrupt heterojunction mesh node with four different materials in contact

No virtual node may be part of the same element as a contact node. If such a situation occurs, an additional nonvirtual, noncontact node is added between the two nodes.

Once the virtual nodes are present, the mesh may not be changed. Therefore, if the mesh is to be built gradually using, for example, **`REGRID`** statements, the

VIRTUAL flag should only be stated on the first **SYMBOLIC** statement occurring after mesh completion.

Poisson Equation and Band Alignment

All virtual nodes sharing one mesh node are forced to the same vacuum level (the vacuum level is continuous across heterojunctions). In each group of virtual nodes, one virtual node is arbitrarily chosen and designated as the *base virtual node*. The Poisson equation is assembled only at base nodes. Contributions from nonheterojunction nodes connected to heterojunction nonbase nodes are inserted into the base node equation. The potential equation at nonbase nodes forces their vacuum level to be that of the base node.

Heterojunction Currents

In general, the current flowing through the virtual sides has two components: the thermionic emission current and the tunneling current. The diagram in Figure 2-19 shows the various current components flowing through the junction in the case of electron transport. The electrons from Material 1 with energies lower than the conduction band in Material 2 can only cross into Material 2 through tunneling. The remaining population flows through thermionic emission. The current flowing from Material 2 to Material 1 only has a thermionic emission component (no tunneling) because those electrons do not see a conduction band barrier. The hole current is treated similarly.

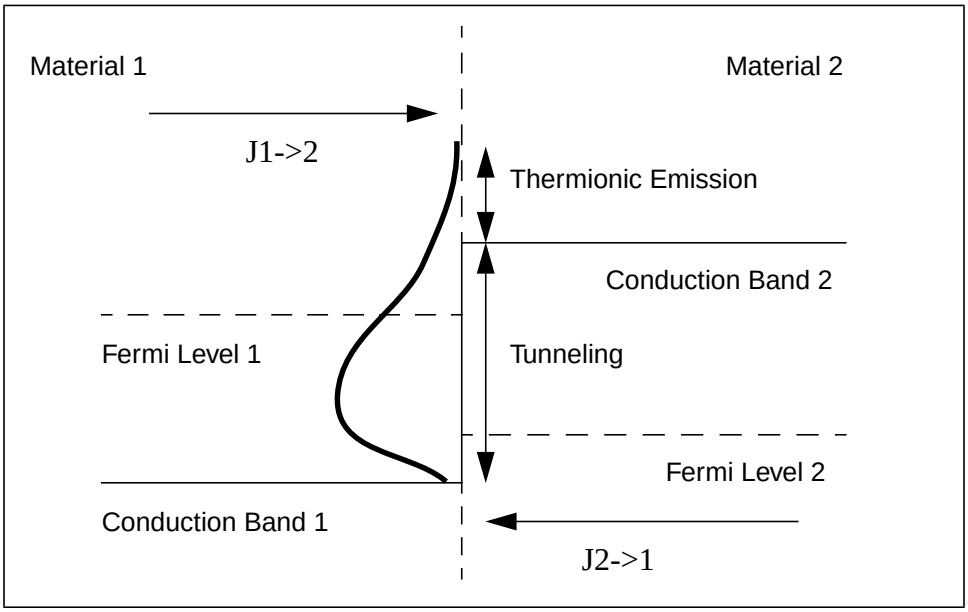


Figure 2-19 Electron current components at an abrupt heterojunction. The curve drawn with a thick line represents the electron energy density in material 1

The default heterojunction current model (**HJSC2** flag on **MODELS** statement, true by default) forces the Fermi levels to the same value on all sides of the heterojunction. This model is compatible with the old implementation of the HD-AAM which forced the Fermi levels to be continuous at heterojunctions.

Thermionic Emission Current Model

The thermionic emission electron current flowing through a virtual side has two components Reference [82]. In the case depicted in Figure 2-19, where electrons see a barrier when crossing from Material 1 to Material 2, the two expressions are:

$$J_{n(1,2)} = \mathbf{ARICHN}_1 \cdot T_1^{\,2} \cdot \exp((E_{Fn1} - E_{c2})/(k \cdot T_1)) \qquad \text{Equation 2-409}$$

$$J_{n(2,1)} = \mathbf{ARICHN}_2 \cdot T_2^{\,2} \cdot \exp((E_{Fn2} - E_{c2})/(k \cdot T_2)) \qquad \text{Equation 2-410}$$

where the 1 and 2 designate Material 1 and Material 2, respectively. Expressions for the case where $E_{c1} > E_{c2}$ are obtained from Equations 2-409 and 2-410 by symmetry. Similar expressions are used for the hole current. To turn this model on, assert **HJTEM** on the **MODELS** statement, which will automatically turn **HJSC2** off.

ARICHN is the Richardson constant for electrons (**ARICHP** for holes). They can be set on the **MATERIAL** statement.

Tunneling Current Model

There can be a nonzero tunneling current component only if the overall flow of the thermionic emission current is against the barrier. The tunneling current is obtained by multiplying the thermionic emission current by a factor Γ_{tunn}:

$$J_{tunn} = \Gamma_{tunn} \cdot J_{therm} \qquad \text{Equation 2-411}$$

In the case of electrons, if you assume the band configuration is that depicted in Figure 2-19 and that the overall thermionic emission current flows from Material 1 to Material 2, $J_{therm} = J_{n(1,2)} - J_{n(2,1)}$ and Γ_{tunn} is expressed as Reference [83]:

$$\Gamma_{tunn} = 1 + \int_0^{(E_{c2} - E_{c1})/(k \cdot T_1)} \exp(\xi - (\xi/\xi_0)^{3/2}) d\xi \qquad \text{Equation 2-412}$$

and

$$\xi_0 = \frac{1}{k \cdot T_1} \cdot \left(\frac{3 \cdot \frac{h}{(2 \cdot \pi)} \cdot q \cdot E_\perp}{4 \cdot (2 \cdot m_1^*)^{1/2}} \right)^{2/3} \qquad \text{Equation 2-413}$$

where $E_\perp$ is the field component inside Material 2 (the barrier) perpendicular to the heterojunction. Note that if $E_\perp$ is not an accelerating field for the carrier under consideration, there is no tunneling. Holes are treated similarly.

To turn this model on, assert **HJTUN** on the **MODELS** statement. Note that **HJTEM** is expected to be *on*.

Trapped Charge Advanced Application Module

The Trapped Charge Advanced Application Module (TC-AAM) allows detailed analysis of semiconductor devices containing traps, such as thin-film transistors (TFT), bipolar junction transistors (BJT), and power MOSFETs. The TC-AAM allows simulation of important carrier trapping and de-trapping mechanisms within semiconductor materials. These effects are important in a wide variety of cases such as the simulation of deep trap levels, deep donor/acceptor states, and the creation of "lifetime profiles." Four trap possibilities are allowed:

- Neutral hole traps
- Neutral electron traps
- Donor states
- Acceptor states

Analysis with Trapped Charge

For the analysis of traps, the energy gap is divided in up to 50 discrete energy levels E_{t_i}. The recombination and trapping processes are then analyzed at each level.

Recombination

For recombination, the Shockley-Read-Hall model is used. For example, the recombination rate for electron traps is:

$$U = \sum_i \frac{pn - n_{ie}^2}{\tau_{p_i}(n + n_{t_i}) + \tau_{n_i}(p + p_{t_i})} \qquad \text{Equation 2-414}$$

where $n_{t_i} = n_{ie}\exp(E_{t_i}/kT)/\mathbf{DGEN}$ and $p_{t_i} = n_{ie}\exp(-E_{t_i}/kT)\mathbf{DGEN}$.

The minority carrier lifetimes for electrons and holes τ_{n_i} and τ_{p_i} are defined separately as a function of bandgap energy as well as position (x,y). Their values are calculated from the parameters **TAUN** and **TAUP** on the **TRAPS** statement. The effects of recombination including tunneling are also included separately for each energy level (see "Recombination Including Tunneling" on page 2-5). The parameter **DGEN** accounts for degeneracy effects. The trap energy level, E_{t_i}, is specified relative to the intrinsic Fermi level.

Modeling

Trapping is also modeled using Shockley-Read-Hall. For fast traps, which instantaneously reach equilibrium, the following expression gives the trap occupation function for electron traps

$$f_i = \frac{\tau_{p_i} n + \tau_{n_i} p_{t_i}}{\tau_{p_i}(n + n_{t_i}) + \tau_{n_i}(p + p_{t_i})} \qquad \text{Equation 2-415}$$

Note that f has a maximum value of 1, indicating a completely full trap. The Poisson equation is then modified to include the number of electrons that are trapped

$$\varepsilon\nabla^2\psi = -q(p - n + N_D^+ - N_A^- - \sum_i N_{t_i} f_i) - \rho_s \qquad \text{Equation 2-416}$$

N_{t_i} is the total number of traps (in #/cm^3/eV) for the ith energy level. N_{t_i} is calculated from the **N.TOTAL** parameter on the **TRAPS** statement and is also a function of energy and position. N_{t_i} is positive for electron traps and negative for hole traps. If the trap state is specified as **CHARGED** (as in the case of a donor state) then the following form of the Poisson equation is used:

$$\varepsilon\nabla^2\psi = -q(p - n + N_D^+ - N_A^- - \sum_i N_{t_i}(f_i - 1)) - \rho_s \qquad \text{Equation 2-417}$$

When time dependent traps are modeled during transient analysis, the traps require some time to come into equilibrium with the semiconductor (it takes time for the traps to capture or emit electrons). Under these conditions it is necessary to solve an additional differential equation for each trap level. For electron traps this rate equation is:

$$\frac{\partial(f_i N_{t_i})}{\partial t} = \frac{(1 - f_i)n - f_i n_{t_i}}{\tau_{n_i}} - \frac{f_i p - (1 - f_i)p_{t_i}}{\tau_{p_i}} \qquad \text{Equation 2-418}$$

Medici uses a special numerical method to self consistently solve the above equations for each electron trap level at each mesh point. For hole traps, the corresponding SRH recombination rate, fast-trap hole occupancy, and trap rate equation are shown in the following equations:

$$U = \sum_i \frac{pn - n_{ie}^2}{\tau_{p_i}(n + n_{t_i}) + \tau_{n_i}(p + p_{t_i})} \qquad \text{Equation 2-419}$$

$$f_i = \frac{\tau_{n_i} p + \tau_{p_i} n_{t_i}}{\tau_{p_i}(n + n_{t_i}) + \tau_{n_i}(p + p_{t_i})} \qquad \text{Equation 2-420}$$

$$\frac{\partial(f_i N_{t_i})}{\partial t} = \frac{f_i n - (1 - f_i)n_{t_i}}{\tau_{n_i}} - \frac{(1 - f_i)p - f_i p_{t_i}}{\tau_{p_i}} \qquad \text{Equation 2-421}$$

where $n_{t_i} = n_{ie}\exp(E_{t_i}/kT)$**DGEN** and $p_{t_i} = n_{ie}\exp(-E_{t_i}/kT)/$**DGEN** .

The net charge of the traps can best be understood by examining Figure 2-20. A careful examination of the trap occupation equation shows that for electrons, if the trap level is below the electron Fermi level, the trap state is filled (contains a bound electron). Likewise for holes, a trap state above the hole Fermi level is filled (contains a bound hole).

Electron Trap

If an electron trap level is specified as, **CHARGED**, then an empty trap has a positive charge (like a donor state). If the **CHARGED** electron trap is filled by an electron, then the negative charge of the electron cancels the positive charge of the tribunate charge of the trap becomes zero.

Hole Trap

If a hole trap level is specified as **CHARGED**, then an empty trap has a negative charge (like an acceptor state). If the **CHARGED** hole trap is filled by a hole, then the positive charge of the hole cancels the negative charge of the trap, and the net charge of the trap becomes zero.

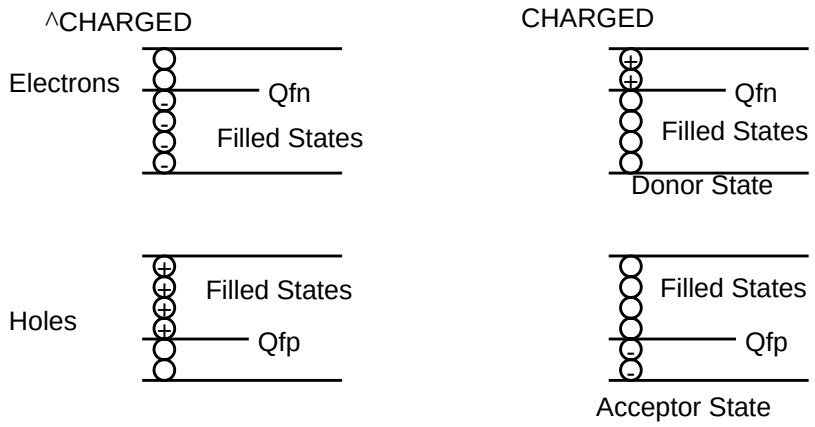


Figure 2-20 Trap net charge possibilities

Transient Simulation Trap States

Any trap states created during a transient simulation by changing the value of **N.TOTAL** are initially empty states because charge must be conserved.

If there are initially 10^{14} traps that are 50% filled (f=0.5, **N.TOTAL**=1e14), and the trap density is increased to 10^{15} traps, then immediately after there will be 10^{15} traps that are 5% filled (**N.TOTAL**=1e15, f=0.05).

Note:

The total trapped charge (which is f*N.TOTAL) does not change in this situation.

Optical Device Advanced Application Module

This section describes the capabilities and the use of the Optical Device Advanced Application Module (OD-AAM) optionally available for use with the Medici program. The OD-AAM provides an advanced treatment of photogeneration associated with radiation incident on image sensors and other optical devices. The OD-AAM includes the following features:

- Ray tracing models the propagation of light inside and outside the device

- Various light absorption mechanisms are accounted for, such as band-to-band transitions, band-tail absorption, and free carrier absorption
- Black-body spectral radiation can be used to describe the external light source or the user may provide a spectral file containing the spectral distribution
- Calculation of transmission through a stack of material layers including interference

A wide variety of materials can be used with the OD-AAM including silicon, amorphous silicon, gallium arsenide, aluminum gallium arsenide, germanium, silicon germanium, indium phosphide, indium arsenide, and diamond. To analyze results, post-processing capabilities allow plotting of ray tracing results through the device structure, internal and external collection efficiency, and transmittance and reflectance.

Transmittance and Reflectance Calculation

The relationship between the angles and the coefficients of incidence, reflection and transmission at the interface between two media is established by the Fresnel formulae for the transverse electric (TE) and the transverse magnetic (TM) components of the incident radiation.

Simple Interface

For the simple interface as shown in Figure 2-21 on page 2-5,

$$\theta_1 = \theta_r \qquad \text{Equation 2-422}$$

$$n_1 \sin\theta_1 = n_2 \sin\theta_2 \qquad \text{Equation 2-423}$$

For the TM wave:

$$r_{TM} = \frac{n_2\cos\theta_1 - n_1\cos\theta_2}{n_2\cos\theta_1 + n_1\cos\theta_2} \qquad \text{Equation 2-424}$$

$$t_{TM} = \frac{2n_1\cos\theta_1}{n_2\cos\theta_1 + n_1\cos\theta_2} \qquad \text{Equation 2-425}$$

For the TE wave:

$$r_{TE} = \frac{n_1\cos\theta_1 - n_2\cos\theta_2}{n_1\cos\theta_1 + n_2\cos\theta_2} \qquad \text{Equation 2-426}$$

$$t_{TE} = \frac{2n_1\cos\theta_1}{n_1\cos\theta_1 + n_2\cos\theta_2} \qquad \text{Equation 2-427}$$

where

- θ_1 is the angle of the incident ray,
- θ_2 is the angle of the refracted ray (transmitted ray),

- θ_r is the angle of the reflected ray,
- r_{TM} and t_{TM} are the reflection coefficient and the transmission coefficient of the TM wave, respectively,
- r_{TE} and t_{TE} are the reflection coefficient and the transmission coefficient of the TE wave, respectively,
- n_1, n_2 are the complex refractive indices of material 1 and material 2, respectively.

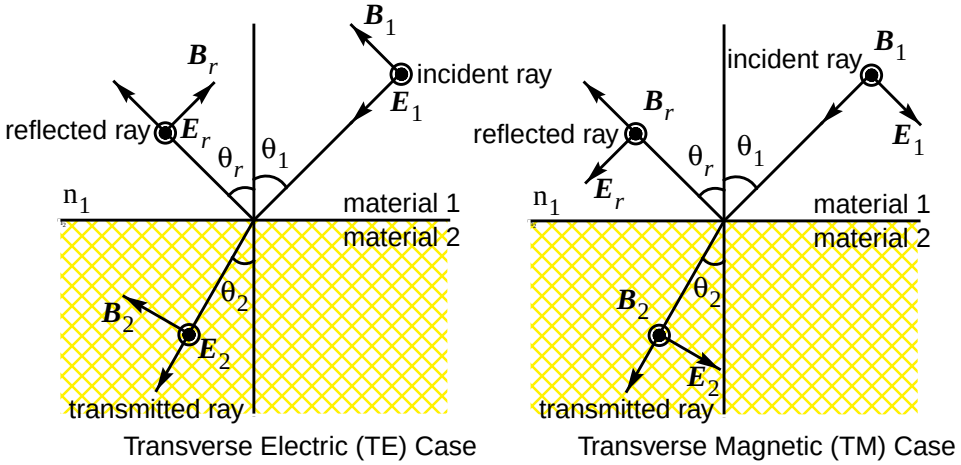


Figure 2-21 Reflection and transmission of a ray

Transmission Through Planar Optical Layers

For a stack of N material layers as shown in Figure 2-22, the reflection and transmission coefficients are expressed in terms of the characteristic matrices of the stratified media. The matrix of the *i*-th material (Reference [68]) is:

for the TE wave

$$M_i = \begin{bmatrix} \cos(kd_i p_i) & \left(\frac{-j}{p_i}\right)\sin(kd_i p_i) \\ -jp_i \sin(kd_i p_i) & \cos(kd_i p_i) \end{bmatrix} \qquad \text{Equation 2-428}$$

for $i = (2,3,...,N-1)$

where

- $p_i = n_i \cos\theta_i$
- $k = \frac{2\pi}{\lambda}$
- n_i is the complex refractive index of the *i*-th material
- d_i is the thickness of the *i*-th material
- λ is the wavelength of incident wave
- θ_i is the angle in the *i*-th material

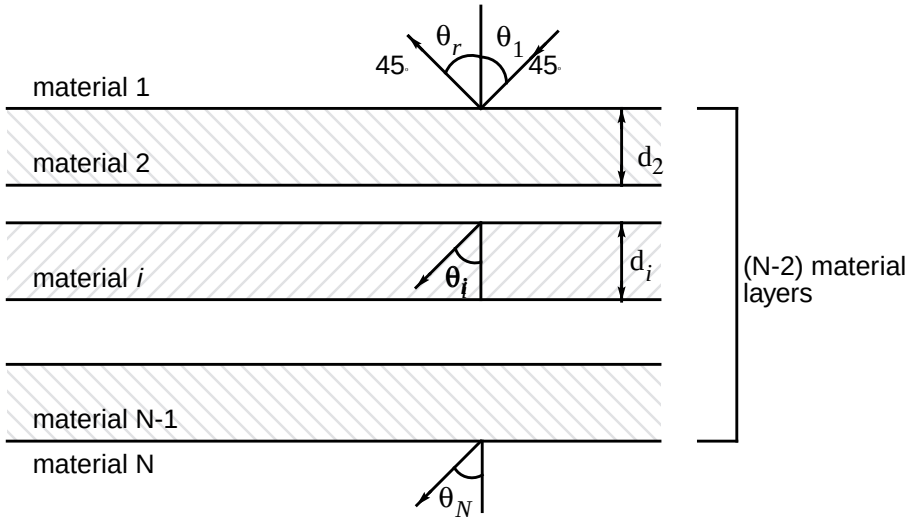


Figure 2-22 A stack of N material layers separated by parallel interfaces

For the TM wave, p_i is replaced by $q_i = \frac{\cos\theta_i}{n_i}$ and the same expressions hold.

The generalization to the case of a succession of media, which is a stack of N material layers is:

$$M = M_2 \times M_3 \times \ldots \times M_{n-1} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \quad \text{Equation 2-429}$$

Then the reflection and transmission coefficients for the TE wave are:

$$r_{TE} = \frac{(m_{11} + m_{12}p_n)p_1 - (m_{21} + m_{22}p_n)}{(m_{11} + m_{12}p_n)p_1 + (m_{21} + m_{22}p_n)} \quad \text{Equation 2-430}$$

$$t_{TE} = \frac{2p_1}{(m_{11} + m_{12}p_n)p_1 + (m_{21} + m_{22}p_n)} \quad \text{Equation 2-431}$$

For the TM wave, the same equations hold, except that p_i is replaced by q_i. It is assumed that all layers are parallel and the magnetic permeabilities are unity. Then for both simple and multilayered structures, the reflectance and transmittance for the radiation is:

$$R_{opt} = A_{opt}^2 r_{TM}^2 + B_{opt}^2 r_{TE}^2 \quad \text{Equation 2-432}$$

$$T = \frac{\mathrm{Re}(n_N \cos\theta_N)}{n_1 \cos\theta_1} (A_{opt}^2 t_{TM}^2 + B_{opt}^2 t_{TE}^2) \quad \text{Equation 2-433}$$

where A^2 and B^2 are the polarization factors (Reference [69]),

$$A^2 = \frac{\cos^2\psi_{opt} + \tan^2\chi_{opt}\sin^2\psi_{opt}}{1 + \tan^2\chi_{opt}} \quad \text{Equation 2-434}$$

$$B^2 = \frac{\sin^2\psi_{opt} + \tan^2\chi_{opt}\cos^2\psi_{opt}}{1 + \tan^2\chi_{opt}} \quad \text{Equation 2-435}$$

where

ψ_{opt} is the angle made by the major axis of the polarization ellipsis of the incident wave and the horizontal axis

$\tan\chi_{opt}$ is the ratio between the lengths of the elliptical axes

Their expressions are as follows:

$$\tan\psi_{opt} = \tan(2\phi_{opt})\cos\delta_{opt} \quad \text{Equation 2-436}$$

$$\sin\chi_{opt} = \tan(2\phi_{opt})\sin\delta_{opt} \quad \text{Equation 2-437}$$

where ϕ_{opt} is the linear polarization angle between the E vector of the incident ray and the incident plane, and δ_{opt} is the phase difference between the TM and the TE components.

Photogeneration Model

By exposing a semiconductor device structure to light or other radiation, it is possible to generate electron-hole pairs inside the device. These photogenerated carriers are created when an electron from the valence band is excited into the conduction by the absorption of a photon with energy greater than the bandgap energy. This absorption process causes the intensity of the radiation inside the device to decrease exponentially with distance according to the expression

$$I_{opt}(x, \lambda) = I_{opt_0}(\lambda)e^{-\alpha_{opt}(\lambda)x} \quad \text{Equation 2-438}$$

where

- $I_{opt_0}(\lambda)$ is the intensity at the starting point inside the device
- $\alpha_{opt}(\lambda)$ is the absorption coefficient for a particular wavelength λ of the incident radiation.

In practice, the incident radiation may consist of a spectrum of wavelengths.

If $G_{opt}(x, \lambda)$ represents the carrier generation rate at a distance x, then the number of carriers generated in the distance x to $x + \Delta x$ (per cm^2 per second) is given by

$$G_{opt}(x, \lambda)\Delta x = -\mathbf{QUAN.EFF} \cdot \frac{I_{opt}(x + \Delta x, \lambda) - I_{opt}(x, \lambda)}{(hc/\lambda)} \quad \text{Equation 2-439}$$

where (hc/λ) is the energy of the absorbed photon and the parameter **QUAN.EFF** is the quantum efficiency (number of generated electron-hole pairs per absorbed photon). The parameter **QUAN.EFF** can be specified on the **PHOTOGEN** statement, with a default value of 1.0. In the limit as $\Delta x \rightarrow 0$, it can be written

Equation 2-440

$$G_{opt}(x,\lambda) = -\mathbf{QUAN.EFF} \cdot \left(\frac{\lambda}{hc}\right)\frac{dI(x,\lambda)}{dx}$$
$$= \mathbf{QUAN.EFF} \cdot \left(\frac{\lambda}{hc}\right) I_{opt_0}(\lambda)\ \alpha_{opt}(\lambda)\ e^{-\alpha_{opt}(\lambda)x}$$

The total generation rate at a point is calculated by integrating over the wavelength spectrum of the incident radiation:

$$G_{opt}(x) = \int G_{opt}(x,\lambda)\ d\lambda \qquad \text{Equation 2-441}$$

From the above expressions, it is clear that to describe the photogeneration occurring within the device structure it is necessary to:

1. Describe the properties of the incident radiation.
2. Provide information regarding the absorption coefficient $\alpha_{opt}(\lambda)$.

Parameters associated with the incident radiation, such as wavelength, wavelength spectrum or intensity, are specified on the **PHOTOGEN** statement. The choices regarding the absorption coefficient are given in the following section.

Absorption Models

By default, the program obtains the absorption coefficient from the imaginary part of the complex index of refraction for the material. The complex index of refraction for a material can be expressed as

$$\hat{n} = \eta - ik \qquad \text{Equation 2-442}$$

where η is the normal refractive index for the material and the absorption coefficient is related to k by

$$\alpha_{opt} = \frac{4\pi k}{\lambda} \qquad \text{Equation 2-443}$$

Wavelength Dependent Empirical Tables

The Optical Device AAM provides wavelength dependent empirical tables for the complex index of refraction for a variety of materials (Reference [70]). You can provide your own data for both the real index of refraction and the imaginary index of refraction or the absorption coefficient itself by specifying the appropriate parameters on the **MATERIAL** statement.

Absorption Mechanisms

The following physical models describing various absorption mechanisms are available in the program:

- Band-to-band absorption (α_{bb})
- Band-tail absorption (α_{bt})
- Free-carrier absorption (α_{fc})

For band-to-band absorption, both direct and indirect transitions can be modeled, as well as absorption in amorphous materials:

$$\alpha_{opt} = \alpha_{bb} + \alpha_{bt} + \alpha_{fc} \quad \text{Equation 2-444}$$

$$\alpha_{bb} = \begin{cases} \textit{from table} & \textit{if } \texttt{BTBT.AB}\text{=False} \\ \alpha_{dir} + \alpha_{indir} & \textit{for crystalline materials if } \texttt{BTBT.AB}\text{=True} \\ \alpha_{amorp} & \textit{for amorphous material if } \texttt{BTBT.AB}\text{=True} \end{cases} \quad \text{Equation 2-445}$$

Models that describe these mechanisms are given in the following sections. Parameters associated with these models can be specified on the **MATERIAL** statement.

Fundamental Absorption (Band-to-Band Transitions)

A general expression for the absorption coefficient due to band-to-band transitions can be given by:

$$\alpha_{bb} = \sum_i \alpha_{bb_i} = \sum_i \frac{A_i}{\eta}\left(\gamma_1 x_{1_i}^{\texttt{EXP.BB}_i} + \gamma_{2_i} x_{2_i}^{\texttt{EXP.BB}_i}\right) \quad \text{Equation 2-446}$$

$$x_{1_i} = h\nu + \texttt{E.PHONON}_i - E_{g,\,opt} \quad \text{Equation 2-447}$$

$$x_{2_i} = h\nu - \texttt{E.PHONON}_i - E_{g,\,opt} \quad \text{Equation 2-448}$$

where the summation is over all relevant transitions:

- η is the real part of refractive index
- $\texttt{EXP.BB}_i$ is an exponent characteristic of the transition type
- $h\nu$ is the photon energy (eV) = 1.24/λ(microns)
- $\texttt{E.PHONON}_i$ is the phonon energy (eV) for indirect transitions (equal to zero for direct transitions)
- $E_{g,\,opt}$ is the optical energy gap (described in "Optical Energy Gap" on page 2-158).

The expressions for A_i, γ_{1_i}, γ_{2_i} in Equation 2-446 depend on the type of transition and the type of material under consideration.

Direct Transitions

For a direct transition, $\texttt{E.PHONON}_i$=0 and

$$A_i = \frac{\mathbf{B.BB}_i}{h\nu} \qquad \gamma_{1_i} = \frac{1 - f(x_{1_i})}{2} \qquad \gamma_{2_i} = \frac{1 - f(x_{2_i})}{2}$$ Equation 2-449

where $\mathbf{B.BB}_i$ is a parameter that can be induced by quantum mechanical treatment (References [71] and [72]) and $f(x)$ is the electron distribution function.

Indirect Transitions

Indirect transitions are characterized by a nonzero value for the phonon energy and

$$A_i = \frac{\mathbf{B.BB}_i \times \mathbf{E1.BB}_i}{h\nu(\mathbf{E1.BB}_i - h\nu)^2}$$ Equation 2-450

$$\gamma_{1_i} = \frac{1 - f(x_{1_i})}{\exp\left(\frac{\mathbf{E.PHONON}_i}{kT}\right) - 1}$$ Equation 2-451

$$\gamma_{2_i} = \frac{1 - f(x_{2_i})}{1 - \exp\left(-\frac{\mathbf{E.PHONON}_i}{kT}\right)}$$ Equation 2-452

where

- $\mathbf{E1.BB}_i$ is the energy gap for a transition to a (virtual) intermediate state
- $\mathbf{E.PHONON}_i$ is the phonon energy
- $\mathbf{B.BB}_i$ is a parameter obtainable from quantum mechanical calculations (References [71] and [72]).

Amorphous Materials

For amorphous materials, the phonon energy is also set to zero, and

$$A_i = \frac{\mathbf{B.BB}_i^2}{h\nu}\eta \qquad \gamma_{1_i} = \frac{1 - f(x_{1_i})}{2} \qquad \gamma_{2_i} = \frac{1 - f(x_{2_i})}{2}$$ Equation 2-453

where $\mathbf{B.BB}_i$ can be induced by quantum mechanical treatment (References [71] and [72]).

Band-Tail Absorption

Band-tail absorption has significance when the material is highly doped or amorphous. For direct transitions, you can expect no absorption at energies smaller than the energy gap and a steeply rising absorption at the band edge. However, in practice, due to band-tail transitions, an exponentially increasing absorption edge occurs. This behavior is known as Urbach's rule (Reference [73]). The contribution to the absorption coefficient by this mechanism is:

$$\alpha_{bt} = K_{bt}\exp\left(\frac{h\nu - \mathbf{E1.BT}}{\mathbf{E.URBACH}}\right)$$ Equation 2-454

where

- **E.URBACH** is the Urbach energy (eV)
- **E1.BT** is the optical energy gap (also called the Tauc gap)
- K_{bt} is the band-to-band absorption coefficient at the optical band edge

In amorphous materials, the Urbach energy is a function of the temperature and the structural disorder and can be calculated from the expression

$$\mathbf{E.URBACH} = \frac{\mathbf{E1.BT} - E_{g,\,opt}}{\mathbf{G.BT}} \qquad \text{Equation 2-455}$$

where **G.BT** is a fitting parameter (Reference [74]).

Free Carrier Absorption

The free carriers make transitions to higher energy states within the same valley by absorbing photons and reducing the light intensity, resulting in a decrease of the EHP generation rate. This effect is accounted for with the following expressions:

$$\alpha_{fc} = \frac{q^3}{\eta c \varepsilon_0 \omega^2 m_0^2}\left[\frac{n}{\mathbf{EL.EMAS}^2 \cdot \mu_n} + \frac{p}{\mathbf{HO.EMAS}^2 \cdot \mu p}\right] \qquad \text{Equation 2-456}$$

$$\eta^2 = \varepsilon_r - \frac{q^2}{\varepsilon_0 \omega^2 m_0}\left[\frac{n}{\mathbf{EL.EMAS}} + \frac{p}{\mathbf{HO.EMAS}}\right] \qquad \text{Equation 2-457}$$

where

- c is the free space speed of light
- η is the real part of the refractive index
- ε_0 is the free space permittivity
- ε_r is the relative permittivity
- **EL.EMAS** is the electron effective mass ratio to free electron mass
- **HO.MAS** is the hole effective mass ratio to free electron mass
- μ_n is the electron energy-dependent mobility
- μ_p is the hole energy-dependent mobility
- m_0 is the free electron mass
- ω is the radial frequency of the photon

Optical Energy Gap

If the incident photon energy is high enough to excite electrons from the valence band to the conduction band, electron-hole pairs are generated in the crystal. The critical photon energy for this generation mechanism is called the optical energy gap which is approximately equal to the electric energy gap. In order to obtain a more precise value, the following expression may be used:

$$E_{g,\,opt} = E_{g,\,opt}(T) + u(E_{F_n} - E_c - 4kT)\left(1 + \frac{\mathbf{EL.EMAS}}{\mathbf{HO.EMAS}}\right)$$ Equation 2-458

Where **EL.EMAS** and **HO.EMAS** are the effective electron and hole mass at the band edge respectively and the first term in RHS is the temperature-dependent optical gap.

Equation 2-459

$$E_{g,\,opt}(T) = \mathbf{EGO300} + \mathbf{EGOALPH}\left(\frac{300^2}{300 + \mathbf{EGOBETA}}\right) - \frac{T^2}{T + \mathbf{EGOBETA}}$$

The second term in Equation 2-458 takes into account the widening of the optical gap by degeneracy.

Equation 2-460

$$u(E_{F_n} - E_c - 4kT) = \begin{cases} E_{F_n} - E_c - 4kT & \text{for } E_{F_n} - E_c - 4kT > 0 \\ 0 & \text{for } E_{F_n} - E_c - 4kT \le 0 \end{cases}$$

where E_{F_n} is the electron quasi-Fermi energy and k is Boltzmann's constant.

The carrier concentration affects the absorption coefficient through electric or optical band-gap changing and band-tail absorption.

Anisotropic Material Advanced Application Module

Medici can account for the anisotropic nature of some semiconductor materials, such as silicon carbide, with the use of the optionally available Anisotropic Material AAM (AM-AAM). This AAM allows most physical models to be treated anisotropically by introducing tensors for various physical quantities such as the dielectric permittivity, carrier mobilities, impact ionization coefficients, and thermal conductivity.

Permittivity Tensor

To enable the AAM, use the **ANISOTROPIC** statement. The components specified should be normalized by the corresponding isotropic quantity. For example, the permittivity tensor can be expressed as:

$$\underline{\underline{\varepsilon}} = \begin{pmatrix} \varepsilon_{xx} & 0 & 0 \\ 0 & \varepsilon_{yy} & 0 \\ 0 & 0 & \varepsilon_{zz} \end{pmatrix} = \varepsilon_{mat} \cdot \begin{pmatrix} \left(\frac{\varepsilon_{xx}}{\varepsilon_{mat}}\right) & 0 & 0 \\ 0 & \left(\frac{\varepsilon_{yy}}{\varepsilon_{mat}}\right) & 0 \\ 0 & 0 & \left(\frac{\varepsilon_{zz}}{\varepsilon_{mat}}\right) \end{pmatrix} \quad \text{Equation 2-461}$$

$$= \varepsilon_{mat} \cdot \begin{pmatrix} \mathbf{PERM(1)} & 0 & 0 \\ 0 & \mathbf{PERM(2)} & 0 \\ 0 & 0 & \mathbf{PERM(3)} \end{pmatrix}$$

where ε_{mat} is the permittivity that is used by the program if the material is isotropic (that is, if **PERM(1)** = **PERM(2)** = **PERM(3)** = 1). The value of ε_{mat} can be specified for each material using the **PERMITTI** parameter on the **MATERIAL** statement.

Additional Capabilities

The AM-AAM also provides for anisotropic electron mobility, hole mobility, electron impact ionization coefficient, hole impact ionization coefficient, and thermal conductivity:

Equation 2-462

$$\underline{\underline{\mu_n}} = \begin{pmatrix} \mu_{n_{xx}} & 0 & 0 \\ 0 & \mu_{n_{yy}} & 0 \\ 0 & 0 & \mu_{n_{zz}} \end{pmatrix} = \mu_{n_{mat}} \cdot \begin{pmatrix} \mathbf{MU.N(1)} & 0 & 0 \\ 0 & \mathbf{MU.N(2)} & 0 \\ 0 & 0 & \mathbf{MU.N(3)} \end{pmatrix}$$

Equation 2-463

$$\underline{\underline{\mu_p}} = \begin{pmatrix} \mu_{p_{xx}} & 0 & 0 \\ 0 & \mu_{p_{yy}} & 0 \\ 0 & 0 & \mu_{p_{zz}} \end{pmatrix} = \mu_{p_{mat}} \cdot \begin{pmatrix} \mathbf{MU.P(1)} & 0 & 0 \\ 0 & \mathbf{MU.P(2)} & 0 \\ 0 & 0 & \mathbf{MU.P(3)} \end{pmatrix}$$

Equation 2-464

$$\underline{\underline{\alpha_n}} = \begin{pmatrix} \alpha_{n_{xx}} & 0 & 0 \\ 0 & \alpha_{n_{yy}} & 0 \\ 0 & 0 & \alpha_{n_{zz}} \end{pmatrix} = \alpha_{n_{mat}} \cdot \begin{pmatrix} \mathbf{II.N(1)} & 0 & 0 \\ 0 & \mathbf{II.N(2)} & 0 \\ 0 & 0 & \mathbf{II.N(3)} \end{pmatrix}$$

Equation 2-465

$$\underline{\underline{\alpha_p}} = \begin{pmatrix} \alpha_{p_{xx}} & 0 & 0 \\ 0 & \alpha_{p_{yy}} & 0 \\ 0 & 0 & \alpha_{p_{zz}} \end{pmatrix} = \alpha_{p_{mat}} \cdot \begin{pmatrix} \mathbf{II.P(1)} & 0 & 0 \\ 0 & \mathbf{II.P(2)} & 0 \\ 0 & 0 & \mathbf{II.P(3)} \end{pmatrix}$$

Equation 2-466

$$\underline{\underline{\kappa}} = \begin{pmatrix} x_{xx} & 0 & 0 \\ 0 & x_{yy} & 0 \\ 0 & 0 & x_{zz} \end{pmatrix} = \kappa_{mat} \cdot \begin{pmatrix} \mathbf{TH.COND(1)} & 0 & 0 \\ 0 & \mathbf{TH.COND(2)} & 0 \\ 0 & 0 & \mathbf{TH.COND(3)} \end{pmatrix}$$

The specific models or parameters used to describe $\mu_{n_{mat}}$, $\mu_{p_{mat}}$, $\alpha_{n_{mat}}$, $\alpha_{p_{mat}}$, and κ_{mat} can be specified on the **MODELS**, **MATERIAL**, and **MOBILITY** statements.

Thermal Diffusion

The program accounts for anisotropic carrier thermal diffusion coefficients through the carrier mobility. That is, in the current density relations,

$$\vec{J_n} = qn\mu_n\vec{E} + q\left(\frac{k_BT}{q}\right)\mu_n\vec{\nabla}n + qnD_n^T\vec{\nabla}T$$ Equation 2-467

$$\vec{J_p} = qn\mu_p\vec{E} - q\left(\frac{k_BT}{q}\right)\mu_p\vec{\nabla}p - qpD_p^T\vec{\nabla}T$$ Equation 2-468

the thermal diffusion terms are assumed to be proportional to mobility and are given by

$$D_n^T = \mathbf{DN.LAT}\left(\frac{k_B}{q}\right)\mu_n$$ Equation 2-469

$$D_p^T = \mathbf{DP.LAT}\left(\frac{k_B}{q}\right)\mu_p$$ Equation 2-470

The factors **DN.LAT** and **DP.LAT** have been provided as user adjustable parameters and can be specified on the **MATERIAL** statement. The default values for these parameters are 1.

Advanced Band Structure

The advanced structure model is described by the following set of equations for electrons and holes. These parameters allow for nonparabolicity and multiple bands in the calculation of the density of states (Reference [75]).

The array parameters **ALPHJ.N**, **MJ.N**, **EJ.N**, **ALPHJ.P**, **MJ.P**, and **EJ.P** take on different values for each value of the summation index i. Up to five values may be specified for each array.

Equation 2-471

$$N_c = N_c \ \mathbf{ALPH0.N} + \sum_j \left(\mathbf{MJ.N} \left(1 + \mathbf{ALPHJ.N} \frac{15 k_b T}{4q} \right) \exp\left(\frac{\mathbf{EJ.N} \ q}{k_b T} \right) \right)$$

Equation 2-472

$$N_v = N_v \ \mathbf{ALPH0.P} + \sum_j \left(\mathbf{MJ.P} \left(1 + \mathbf{ALPHJ.P} \frac{15 k_b T}{4q} \right) \exp\left(\frac{\mathbf{EJ.P} \ q}{k_b T} \right) \right)$$

Notes on Usage

The **ANISOTROPIC** statement is specified before the first **SOLVE** statement where it is desired to account for the anisotropic nature of the material. The specified parameters remain in effect until:

- They are changed by you on another **ANISOTROPIC** statement
- A **LOAD** statement reads in a saved solution that may contain different values for the anisotropic parameters
- A **MESH** statement initializes all parameters to their default values

Symbol Definitions

The following is a list of symbols encountered in this chapter, together with their definitions and units. The ordering is alphabetical.

Symbol	Definition	Units
A_i	Semiconductor area associated with node *i*	cm
A_n	Maximum ionization integral for electrons	none
A_p	Maximum ionization integral for holes	none
b_1	Vector of length 3N used in AC small-signal analysis description	
b_2	Vector of length 3N used in AC small-signal analysis description	
B	Bernoulli function	none
c	Specific heat for a material	J/q-K
C	Capacitance	F/μm
C_{GB}	Capacitance component between gate and substrate	F/μm
C_{GD}	Capacitance component between gate and drain	F/μm
C_{GS}	Capacitance component between gate and source	F/μm
C_{ij}	Capacitance matrix component	F/μm
d_{ij}	Distance between nodes *i* and *j*	cm
d_{3p}	Distance between node 3 and point *p*	cm
$d_{c,i}$	Contact length associated with node *i*	$cm^2/\mu m$
d_i	Interface length associated with node *i*	cm
d_{int}	Distance between the point (x, y) and the interface	cm
$\vec{D}$	Electric displacement vector	C/cm^2
D_1	N×N matrix used in small-signal analysis description	
D_2	3N×3N matrix used in small-signal analysis description	
D_n	Electron diffusivity	cm^2/s
D_p	Hole diffusivity	cm^2/s
E	Magnitude of electric field	V/cm
$\vec{E}$	Electric field vector	V/cm

Symbol	Definition	Units
$E_{\|\|}$	Parallel component of electric field	V/cm
$E_{\|\|_{ij}}$	Electric field parallel to the side between nodes *i* and *j*	V/cm
$E_{\|\|,\mathrm{n}}$	Parallel component of electric field for electrons	V/cm
$E_{\|\|,\mathrm{p}}$	Parallel component of electric field for holes	V/cm
$E_{\perp}$	Perpendicular component of electric field	V/cm
$E_{\perp ij}$	Electric field perpendicular to the side between nodes *i* and *j*	V/cm
$E_{\perp,\, n}$	Perpendicular component of electric field for electrons	V/cm
$E_{\perp,\, p}$	Perpendicular component of electric field for holes	V/cm
E_A	Acceptor energy level	eV
E_C	Conduction band energy level	eV
E_D	Donor energy level	eV
$E_{eff_{\perp},\, n}$	Perpendicular component of interface effective electric field for electrons	V/cm
$E_{eff_{\perp},\, p}$	Perpendicular component of interface effective electric field for holes	V/cm
$\vec{E}_{eff,\, n}$	Interface effective electric field for electrons	V/cm
$\vec{E}_{eff,\, p}$	Interface effective electric field for holes	V/cm
$E_{eff,\, n}$	Bulk effective electric field for electrons	V/cm
$E_{eff,\, p}$	Bulk effective electric field for holes	V/cm
E_{Fn}	Electron Fermi energy level	eV
E_{Fp}	Hole Fermi energy level	eV
E_g	Bandgap energy	eV
$E_{g,\, insul}$	Insulator bandgap energy	eV
$E_{g,\, semi}$	Semiconductor bandgap energy	eV
E_i	Intrinsic Fermi energy level	eV
E_{insul}	Magnitude of insulator electric field	V/cm
$E_{insul,\, \perp_s}$	Insulator electric field perpendicular to the interface	V/cm
$\vec{E}_n$	Electric field vector in electron transport equation	V/cm

Symbol	Definition	Units
$\vec{E}_p$	Electric field vector in hole transport equation	V/cm
$\vec{E}_{semi}$	Semiconductor electric field	V/cm
$E_{semi,\perp_s}$	Semiconductor electric field perpendicular to the interface	V/cm
E_t	Trap energy level	eV
E_V	Valence band energy level	eV
f_T	Cut off frequency	Hz
$\mathbf{F}_{1/2}$	Fermi-Dirac integral of order one-half	none
$\mathbf{F}^{-1}_{1/2}$	Inverse Fermi-Dirac integral of order one-half	none
$\mathbf{F}_{-1/2}$	Fermi-Dirac integral of order minus one-half	none
F_n	Right-hand-side of the electron current-continuity equation	#/cm^3-s
F_p	Right-hand-side of the hole current-continuity equation	#/cm^3-s
F_ψ	Right-hand-side of Poisson's equation	C/cm^3
g_B	Screening term in carrier-carrier scattering mobility model	none
G_{ij}	Conductance matrix component	S/μm
G^{BB}	Generation rate due to band-to-band tunneling	#/cm^3-s
G^{II}	Total generation rate due to impact ionization	#/cm^3-s
G^{II}_n	Generation rate due to electron impact ionization	#/cm^3-s
G^{II}_p	Generation rate due to hole impact ionization	#/cm^3-s
h	Planck's constant (6.626×10^{-27})	erg-s
H	Heat generation term	W/cm^3
I	Current	A/μm
I_d	Drain current	A/μm
$I_{disp,i}$	Displacement current at node *i*	A/μm
I_{gate}	Gate current	A/μm
$\tilde{I}_i$	Phasor terminal current at electrode *i*	A/μm
I_l	Current through inductor	A/μm

Symbol	Definition	Units
$I_{n,\,i}$	Electron current at node *i*	A/μm
$I_{p,\,i}$	Hole current at node *i*	A/μm
I_{source}	Specified terminal current	A/μm
τ_n	Ionization integral for electrons	none
τ_p	Ionization integral for holes	none
j	Imaginary unit	none
J	3N×3N DC Jacobian matrix	
$\vec{J}$	Current density	C/cm^2-s
$\vec{J}_n$	Electron current density	C/cm^2-s
$J_{n_{ij}}$	Scharfetter-Gummel electron current density between nodes *i* and *j*	C/cm^2-s
$\vec{J}_p$	Hole current density	C/cm^2/-s
$J_{p_{ij}}$	Scharfetter-Gummel current density for holes between nodes *i* and *j*	C/cm^2-s
$\vec{J}_{sn}$	Surface electron current density	C/cm^2-s
$\vec{J}_{sp}$	Surface hole current density	C/cm^2-s
k	Boltzmann's constant (8.617×10^{-5})	eV/K
L	Inductance	H-μm
m_0	Electron rest mass	g
m_{de}	Density of states effective mass for electrons	g
m_{dh}	Density of states effective mass for holes	g
m_e	Effective electron mass	g
m_h	Effective hole mass	g
M_C	Number of equivalent minima in the conduction band	none
M_V	Number of equivalent minima in the valence band	none
n	Electron concentration	#/cm^3
$\tilde{n}$	AC component of electron concentration	#/cm^3
$\hat{n}_{\|\|}$	Unit vector parallel to interface	none
$\hat{n}_{\perp}$	Unit vector normal to interface	none

Symbol	Definition	Units
n_0	Initial electron concentration	#/cm^3
n_{eq}	Equilibrium electron concentration	#/cm^3
$n_{eq_{eff}}$	Effective equilibrium electron concentration	#/cm^3
n_i	Electron concentration at node *i*	#/cm^3
n_i	Intrinsic carrier concentration	#/cm^3
$\tilde{n}_i$	AC component of electron concentration at node *i*	#/cm^3
n_{i0}	DC component of electron concentration at node *i*	#/cm^3
n_{ie}	Effective intrinsic carrier concentration	#/cm^3
n_k	Electron concentration at time t_k	#/cm^3
n_{k-1}	Electron concentration at time t_{k-1}	#/cm^3
n_{k-2}	Electron concentration at time t_{k-2}	#/cm^3
n_s	Surface electron concentration	#/cm^3
$n_{s_{eff}}$	Effective surface electron concentration	#/cm^3
N	Net impurity concentration	#/cm^3
N_b	Number of grid points associated with an electrode	none
N_{inv}	Inversion layer density	#/cm^2
N_p	Number of grid points	none
N_{total}	Total impurity concentration	#/cm^3
N_A	Net compensated acceptor impurity concentration	#/cm^3
$N_{\bar{A}}$	Ionized acceptor impurity concentration	#/cm^3
$N_{A,\,total}$	Total acceptor impurity concentration	#/cm^3
N_B	Bulk impurity concentration	#/cm^3
N_C	Effective conduction band density of states	#/cm^3
N_D	Net compensated donor impurity concentration	#/cm^3
N_D^+	Ionized donor impurity concentration	#/cm^3
$N_{D,\,total}$	Total donor impurity concentration	#/cm^3
N_T	Total impurity concentration	#/cm^3
N_V	Effective valence band density of states	#/cm^3

Symbol	Definition	Units
p	Hole concentration	#/cm^3
$\tilde{p}$	AC component of hole concentration	#/cm^3
p_0	Initial hole concentration	#/cm^3
p_{eq}	Equilibrium hole concentration	#/cm^3
$p_{eq_{eff}}$	Effective equilibrium hole concentration	#/cm^3
p_i	Hole concentration at node *i*	#/cm^3
$\tilde{p}_i$	AC component of hole concentration at node *i*	#/cm^3
p_{i0}	DC component of hole concentration at node *i*	#/cm^3
p_k	Hole concentration at time t_k	#/cm^3
p_{k-1}	Hole concentration at time t_{k-1}	#/cm^3
p_{k-2}	Hole concentration at time t_{k-2}	#/cm^3
p_s	Surface hole concentration	#/cm^3
$p_{s_{eff}}$	Effective surface hole concentration	#/cm^3
$P_{insul,\,n}$	Probability that an electron will not be scattered in the insulator	none
$P_{insul,\,p}$	Probability that a hole will not be scattered in the insulator	none
$P_{semi,\,n}$	Probability that an electron will not be scattered in the semiconductor	none
$P_{semi,\,p}$	Probability that a hole will not be scattered in the semiconductor	none
$P_{\Phi_{b,\,n}}$	Probability that an electron acquires energy to surmount barrier $\Phi_{b,\,n}$	none
$P_{\Phi_{b,\,p}}$	Probability that a hole acquires energy to surmount barrier $\Phi_{b,\,p}$	none
q	Electronic charge (1.602×10^{-19})	C
Q_G	Gate charge	C/μm
r	Radial coordinate used with cylindrical coordinates.	μm
R	Resistance	Ω-μm
R_i	Resistance at node *i*	Ω-μm

Symbol	Definition	Units
$\vec{S}_n$	Electron energy flow density (scaled)	W/cm^2-C
$\vec{S}_p$	Hole energy flow density (scaled)	W/cm^2-C
t	Time	s
t_k	Time after time step k	s
t_{k-1}	Time after time step $k-1$	s
t_{k-2}	Time after time step $k-2$	s
T	Temperature	K
T_n	Electron temperature	K
T_p	Hole temperature	K
u_0	Thermal voltage for lattice kT_0/q	V
u_n	Thermal voltage for electrons kT_n/q	V
u_p	Thermal voltage for holes kT_p/q	V
u_{crit}	Critical thermal voltage	V
U	Net recombination rate	#/cm^3-s
U_n	Net electron recombination rate	#/cm^3-s
U_p	Net hole recombination rate	#/cm^3-s
U_{Auger}	Auger recombination rate	#/cm^3-s
U_{btbt}	Recombination rate due to band-to-band tunneling	#/cm^3-s
U_{RTUN}	Recombination rate including tunneling	#/cm^3-s
U_{SRH}	Shockley-Read-Hall recombination rate	#/cm^3-s
U_{SRH}^*	SRH recombination rate including field dependent life-times	#/cm^3-s
$\vec{v}_n$	Electron mean velocity	cm/s
$\vec{v}_p$	Hole mean velocity	cm/s
v_n^{sat}	Electron saturation velocity	cm/s
v_p^{sat}	Hole saturation velocity	cm/s
v_{sn}	Surface recombination velocity for electrons	cm/s
v_{sp}	Surface recombination velocity for holes	cm/s

Symbol	Definition	Units
V	Voltage	V
$V_{applied}$	Applied voltage	V
V_B	Substrate voltage	V
V_{ds}	Drain to source voltage	V
V_D	Drain voltage	V
V_{gs}	Gate to source voltage	V
V_G	Gate voltage	V
V_i	AC sinusoidal voltage at electrode *i*	V
$\tilde{V}_i$	AC component of V_i	V
V_{i0}	DC component of V_i	V
$\tilde{V}_j$	Phasor terminal voltage at electrode *j*	V
$\tilde{V}_k$	Phasor terminal voltage at electrode *k*	V
V_S	Source voltage	V
x	Horizontal distance coordinate	μm
X_I	Imaginary AC solution vector of length 3N	
X_R	Real AC solution vector of length 3N	
y	Vertical distance coordinate	μm
$\tilde{Y}_{ij}$	Admittance matrix component	S/μm
z	Distance coordinate used with cylindrical coordinates	μm
a_n	Electron mobility exponent	none
a_n	Electron mobility factor	V^{-1}
$\alpha_{n,\,ii}$	Electron ionization coefficient	cm^{-1}
a_p	Hole mobility exponent	none
a_p	Hole mobility factor	V^{-1}
$\alpha_{p,\,ii}$	Hole ionization coefficient	cm^{-1}
γ	Ratio of 1st part of time step to total time step	none
γ_n	Fermi-Dirac degeneracy factor for electrons	none
γ_p	Fermi-Dirac degeneracy factor for holes	none

Symbol	Definition	Units
Γ_n	Electron field-effect function	none
Γ_n	Probability per unit length for electron injection into the gate	cm^{-1}
Γ_p	Hole field-effect function	none
Γ_p	Probability per unit length for hole injection into the gate	cm^{-1}
Δ_{12}	Potential difference between nodes 1 and 2 scaled by kT/q	none
Δn	Update to electron concentration	#/μm
Δp	Update to hole concentration	#/μm
$\Delta\psi$	Update to potential	V
ΔQ_G	Incremental gate charge	C/μm
Δt_k	Size of time step k	s
Δt_{k+1}	Size of time step $k + 1$	s
ΔV	Incremental voltage	V
$\Delta\phi_b$	Change in barrier height due to barrier lowering	V
$\Delta\psi_{int}$	Potential difference between the point (x, y) and the interface	V
ε	Dielectric permittivity	F/cm
ε_1	Dielectric permittivity in material 1	F/cm
ε_2	Dielectric permittivity in material 2	F/cm
ε_{insul}	Insulator dielectric permittivity	F/cm
ε_{semi}	Semiconductor dielectric permittivity	F/cm
η_n	Conduction band to electron Fermi level energy difference, scaled	none
η_p	Valence band to hole Fermi level energy difference, scaled	none
θ	Intrinsic work function	V
λ	Thermal conductivity	W/cm-K
λ_n	Optical phonon mean free path for electrons	cm

Symbol	Definition	Units
λ_p	Optical phonon mean free path for holes	cm
μ_{0n}	Low field electron mobility	cm^2/V-s
μ_{0p}	Low field hole mobility	cm^2/V-s
$\mu_{ac,\,n}$	Electron mobility accounting for acoustic phonon scattering	cm^2/V-s
$\mu_{ac,\,p}$	Hole mobility accounting for acoustic phonon scattering	cm^2/V-s
$\mu_{b,\,n}$	Electron bulk mobility	cm^2/V-s
$\mu_{b,\,p}$	Hole bulk mobility	cm^2/V-s
$\mu_{eff,\,n}$	Effective surface mobility for electrons	cm^2/V-s
$\mu_{eff,\,p}$	Effective surface mobility for holes	cm^2/V-s
μ_n	Electron mobility	cm^2/V-s
$\mu_{n_{ij}}$	Electron mobility between nodes *i* and *j*	cm^2/V-s
$\mu_{n,\,p}^{C}$	Electron or hole mobility including carrier-carrier scattering effects	cm^2/V-s
$\mu_{n,\,p}^{I}$	Electron or hole mobility including ionized impurity scattering effects	cm^2/V-s
$\mu_{n,\,p}^{L}$	Electron or hole mobility including lattice scattering effects	cm^2/V-s
μ_p	Hole mobility	cm^2/V-s
$\mu_{p_{ij}}$	Hole mobility between nodes *i* and *j*	cm^2/V-s
$\mu_{sr,\,n}$	Electron mobility accounting for surface roughness	cm^2/V-s
$\mu_{sr,\,p}$	Hole mobility accounting for surface roughness	cm^2/V-s
$\mu_{S,\,n}$	Electron mobility including surface scattering effects	cm^2/V-s
$\mu_{S,\,p}$	Hole mobility including surface scattering effects	cm^2/V-s
π	Numeric constant (3.14159265)	none
ρ	Mass density for a material	g/cm^3
ρ_F	Fixed interface charge concentration	C/cm^3
$\sigma_{a,\,n}$	Electron acceptor interface trap charge density	C/cm^2
$\sigma_{a,\,p}$	Hole acceptor interface trap charge density	C/cm^2

Symbol	Definition	Units
$\sigma_{d,\,n}$	Electron donor interface trap charge density	C/cm^2
$\sigma_{d,\,p}$	Hole donor interface trap charge density	C/cm^2
σ_f	Fixed surface charge density	C/cm^2
σ_s	Surface charge density	C/cm^2
τ_i	Relative local truncation error for node *i*	none
τ_n	Electron lifetime	s
τ_p	Hole lifetime	s
τ_n^{eff}	Effective electron lifetime	s
τ_p^{eff}	Effective hole lifetime	s
τ_{wn}	Electron energy relaxation time	s
τ_{wp}	Hole energy relaxation time	s
ϕ	Semiconductor contact voltage	V
ϕ_{bn}	Barrier height for electrons	V
ϕ_{bp}	Barrier height for holes	V
ϕ_k	Potential at time t_k	V
ϕ_n	Electron quasi-Fermi potential	V
ϕ_p	Hole quasi-Fermi potential	V
$\Phi_{b,\,n}$	Semiconductor-insulator potential barrier for electrons	V
$\Phi_{b,\,p}$	Semiconductor-insulator potential barrier for holes	V
χ	Electron affinity	V
χ_{insul}	Insulator electron affinity	V
χ_{semi}	Semiconductor electron affinity	V
ψ	Potential	V
$\tilde{\psi}$	AC component of potential	V
ψ_0	Initial potential	V
ψ_i	Potential at node *i* (or in material i)	V
$\tilde{\psi}_i$	AC component of potential at node *i*	V
ψ_{i0}	DC component of potential at node *i*	V

Symbol	Definition	Units
$\psi_{intrinsic}$	Intrinsic Fermi potential	V
ψ_p	Potential at point *p*	V
ψ_s	Surface potential	V
$\psi_{s_{eff}}$	Effective surface potential	V
$\Psi_{b,n}$	Semiconductor-insulator interface barrier for electrons	V
$\Psi_{b,p}$	Semiconductor-insulator interface barrier for holes	V
ω	Radial frequency	s^{-1}

References

[1] A. S. Grove, *Physics and Technology of Semiconductor Devices* (New York: John Wiley & Sons, 1967).

[2] D. J. Roulston, N. D. Arora, and S. G. Chamberlain, "Modeling and Measurement of Minority-Carrier Lifetime Versus Doping in Diffused Layers of n^+-p Silicon Diodes," *IEEE Trans. Electron Devices*, Vol. ED-29, pp. 284-291, Feb. 1982.

[3] G. A. M. Hurkx, D. B. M. Klaassen, and M. P. G. Knuvers, "A New Recombination Model for Device Simulation Including Tunneling," *IEEE Trans. Electron Devices*, Vol. ED-39, pp. 331-338, Feb. 1992.

[4] S. M. Sze, *Physics of Semiconductor Devices*, 2nd ed. (New York: John Wiley & Sons, 1981).

[5] J. W. Slotboom, "The *pn* Product in Silicon," *Solid-State Electronics*, 20, pp. 279-283, 1977.

[6] Zhiping Yu and R. W. Dutton, *SEDAN III - A Generalized Electronic Material Device Analysis Program*, Stanford Electronics Laboratory Technical Report, Stanford University, July1985.

[7] W. B. Joyce and R. W. Dixon, "Analytic Approximation for the Fermi Energy of an Ideal Fermi Gas," *Appl. Phys. Lett.*, Vol. 31, pp. 354-356, 1977.

[8] R. C. Jaeger and F. H. Gaensslen, "Simulation of Impurity Freezeout through Numerical Solution of Poisson's Equation and Application to MOS Device Behavior," *IEEE Trans. Electron Devices*, Vol. ED-27, pp. 914-920, May 1980.

[9] M.R. Shaheed, C.M. Maziar, "A Physically Based Model for Carrier Freeze-out in Si- and SiGe-Base Bipolar Transistors Suitable for Implementation in Device Simulators," *1994 Bipolar/BiCMOS Circuits & Technology Meeting*, pp. 191-194.

[10] S. Selberherr, "MOS Device Modeling at 77K," *IEEE Trans. Electron Devices*, Vol. 36, No. 8, pp. 1464-1474, Aug. 1989.

[11] D. M. Caughey and R. E. Thomas, "Carrier Mobilities in Silicon Empirically Related to Doping and Field," *Proc. IEEE*, Vol. 55, pp. 2192-2193, 1967.

[12] S. Selberherr, "Process and Device Modeling for VLSI," *Microelectron. Reliab.*, Vol. 24, No. 2, pp. 225-257, 1984.

[13] N. D. Arora, J. R. Hauser, and D. J. Roulston, "Electron and Hole Mobilities in Silicon as a Function of Concentration and Temperature," *IEEE Trans. Electron Devices*, Vol. ED-29, pp. 292-295, Feb. 1982.

[14] J. M. Dorkel and Ph. Leturcq, "Carrier Mobilities in Silicon Semi-Empirically Related to Temperature, Doping, and Injection Level," *Solid-State Electronics*, 24, pp. 821-825, 1981.

[15] D. B. M. Klaassen, "A Unified Mobility Model for Device Simulation - I," *Solid-State Electronics*, 35, pp. 953-959, 1992.

[16] D. B. M. Klaassen, "A Unified Mobility Model for Device Simulation - II," *Solid-State Electronics*, 35, pp. 961-967, 1992.

[17] G. Masetti, M. Severi, and S. Solmi, "Modelling of Carrier Mobility Against Carrier Concentration in Arsenic-, Phosphorus-, and Boron-Doped Silicon," *IEEE Trans. Electron Devices*, Vol. ED-30, pp. 746-769, July 1983.

[18] D. B. M. Klaassen, J. W. Slotboom and H. C. de Graaff, "Unified Apparent Bandgap Narrowing in n- and p-type Silicon," *Solid-State Electronics*, 35, pp. 125-129, 1992.

[19] J. T. Watt, "Improved Surface Mobility Models in PISCES," presented at Computer-Aided Design of IC Fabrication Processes, Stanford University, Aug. 6, 1987.

[20] J. T. Watt, "Surface Mobility Modeling," presented at Computer-Aided Design of IC Fabrication Processes, Stanford University, Aug. 3, 1988.

[21] K. Yamaguchi, "Field-Dependent Mobility Model for Two-Dimensional Numerical Analysis of MOSFET's," *IEEE Trans. Electron Devices*, Vol. ED-26, pp. 1068-1074, July 1979.

[22] K. Yamaguchi, "A Mobility Model for Carriers in the MOS Inversion Layer," *IEEE Trans. Electron Devices*, Vol. ED-30, pp. 658-663, June 1983.

[23] C. Lombardi, S. Manzini, A. Saporito, and M. Vanzi, "A Physically Based Mobility Model for Numerical Simulation of Nonplanar Devices," *IEEE Trans. Computer-Aided Design*, Vol. 7, No.11, pp.1164-1170, Nov. 1988.

[24] J. J. Barnes, R. J. Lomax, and G. I. Haddad, "Finite-Element Simulation of GaAs MESFET's with Lateral Doping Profiles and Sub-Micron Gates," *IEEE Trans. Electron Devices*, Vol. ED-23, pp. 1042-1048, Sept. 1976.

[25] M. A. Littlejohn, J. R. Hauser, and T. H. Glisson, "Velocity-Field Characteristics of GaAs with $\Gamma_6^c - L_6^c - X_6^c$ Conduction Band Ordering," *J. Appl. Phys.*, Vol. 48, No. 11, pp. 4587-4590, Nov. 1977.

[26] D. L. Scharfetter and H. K. Gummel, "Large-Signal Analysis of a Silicon Read Diode Oscillator," *IEEE Trans. Electron Devices*, Vol. ED-16, pp. 64-77, Jan. 1969.

[27] S. E. Laux and R. G. Byrnes, "Semiconductor Device Simulation Using Generalized Mobility Models," *IBM J. Res. Dev.*, Vol. 29, No. 3, pp. 289-301, May 1985.

[28] C. R. Crowell and S. M. Sze, "Current Transport in Metal-Semiconductor Barriers," *Solid-State Electronics*, 9, pp. 1035-1048, 1966.

[29] J. M. Andrews and M. P. Lepselter, "Reverse Current-Voltage Characteristics of Metal-Silicide Schottky Diodes," *Solid-State Electronics*, 13, pp. 1011-1023, 1970.

[30] E. Sangiorgi, C. S. Rafferty, M. R. Pinto, and R. W. Dutton, "Non-Planar Schottky Device Analysis and Applications," Proceedings International Conference on Simulation of Semiconductor Devices and Processes, Swansea, U.K., July 1984.

[31] C. S. Rafferty, M. R. Pinto, and R. W. Dutton, "Iterative Methods in Semiconductor Device Simulation," *IEEE Trans. Electron Devices*, Vol. ED-32, pp. 2018-2027, Oct. 1985.

[32] M. R. Pinto and R. W. Dutton, "Accurate Trigger Condition Analysis for CMOS Latch-Up," *IEEE Electron Device Lett.*, Vol. EDL-6, Feb. 1985.

[33] R. S. Varga, *Matrix Iterative Analysis* (Engelwood Cliffs, NJ: Prentice-Hall, 1962).

[34] C. H. Price, "Two Dimensional Numerical Simulation of Semiconductor Devices," Doctoral dissertation, Stanford University, 1982.

[35] D. J. Rose and R. E. Bank, *Global Approximate Newton Methods*, Numerische Mathematik, 37, pp. 279-295, 1981.

[36] B. A. Carré, *Computer Journal*, 4, p. 73, 1961.

[37] S. C. Eisenstat, M. C. Gursky, M. H. Schultz, and A. H. Sherman, Department of Computer Science Technical Report 112, Yale University, 1977.

[38] R. E. Bank and A. H. Sherman, "A Refinement Algorithm and Dynamic Data Structure for Finite Element Meshes," University of Texas at Austin Technical Report CS TR-159/CNA TR-166, Oct. 1980.

[39] B. S. Baker, E. H. Grosse, and C. S. Rafferty. "Non-Obtuse Triangulation of General Polygons," to be published.

[40] C. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations* (Englewood Cliffs, NJ: Prentice-Hall, 1971).

[41] A. De Mari, "An Accurate Numerical One-Dimensional Solution of the *p-n* Junction Under Arbitrary Transient Conditions," *Solid-State Electronics*, 11, pp. 1021-1053, 1968.

[42] J. Lambert, *Computational Methods in Ordinary Differential Equations* (New York: John Wiley and Sons, 1973).

[43] H. Iwai, M. R. Pinto, C. S. Rafferty, J. E. Oristian, and R. W. Dutton, "Velocity Saturation Effect on Short Channel MOS Transistor Capacitance," *IEEE Electron Device Lett.*, Vol. EDL-6, March 1985.

[44] S. E. Laux, "Techniques for Small-Signal Analysis of Semiconductor Devices," *IEEE Trans. Electron Devices*, Vol. ED-32, pp. 2028-2037, Oct. 1985.

[45] S. Tam, P. Ko, C. Hu, and R. S. Muller, "Correlation between Substrate and Gate Currents in MOSFETs," *IEEE Trans. Electron Devices*, Vol. ED-29, pp. 1740-1744, Nov. 1982.

[46] S. Tam, P. Ko and C. Hu, "Lucky-Electron Model of Channel Hot-Electron Injection in MOSFETs," *IEEE Trans. Electron Devices*, Vol. ED-31, pp. 1116-1125, Sept. 1984.

[47] S. Selberherr, *Analysis and Simulation of Semiconductor Devices* (Wien-New York: Springer-Verlag, 1984).

[48] R. VanOverstraeten and H. Deman, "Measurement of the Ionization Rates in Diffused Silicon p-n Junctions," *Solid-State Electronics*, 13, pp. 583-608, 1970.

[49] C. Fiegna, F. Venturi, M. Melanotte, E. Sangiorgi, and B. Ricco, "Simple and Efficient Modeling of EPROM Writing," *IEEE Trans. Electron Devices*, Vol. ED-38, pp. 603-610, March 1991.

[50] E. O. Kane, "Zener Tunneling in Semiconductors," *J. Phys. Chem. Solids*, Vol. 12, pp. 181-188, 1959.

[51] D. Chen, E. C. Kan, U. Ravaioli, Z. Yu, and R. W. Dutton, "A Self-Consistent Discretization Scheme for Current and Energy Transport Equations," *SISDEP'91*, Zurich, Switzerland, Sept. 1991.

[52] D. Chen, E. C. Kan, U. Ravaioli, C-W. Shu, and R. W. Dutton, "An Detonation Improved Energy Transport Model Including Nonparabolocity and Non-Maxwellian Distribution Effects," *IEEE Electron Device Lett.*, Vol. EDL-13, Jan. 1992.E.

[53] D. Chen, E. Sangiorgi, M. R. Pinto, E. C. Kan, U. Ravaioli, and R.W. Dutton, "Analysis of the Spurious Velocity Overshoot in Hydrodynamic Simulations," *NUPAD-IV*, Seattle, WA, 1992.

[54] A. Forghieri, R. Guerreri, P. Ciampolini, A. Gnudi, M. Rudan, and G. Baccarani, "A New Discretization Strategy of the Semiconductor Equations Comprising Momentum and Energy Balance," *IEEE Trans. Computer-Aided Design*, Vol. 7, No. 2, Feb. 1988.

[55] B. Meinerzhagen and W. L. Engl, "The Influence of the Thermal Equilibrium Approximation on the Accuracy of Classical Two-Dimensional Numerical Modeling of Silicon Submicrometer MOS Transistors," *IEEE Trans. Electron Devices*, Vol. 35, No.5, pp. 689-697, May 1988.

[56] V. Axelrad, "A General Purpose Device Simulator Including Carrier Energy Balance Based on PISCES-2B," *International Workshop on VLSI Process and Device Modeling*, Oiso, Japan, May 1991.

[57] V. Axelrad, G. Rollins, S. Motzny, and M. Jaczynski, "Implementation and Applications of the Hydrodynamic Model in a General Purpose Device Simulator," *SISDEP'91*, Zurich, Switzerland, Sept. 1991.

[58] J. W. Slotboom, G. Streutker, G. J. T. Davids and P. B. Hartog, "Surface Impact Ionization in Silicon Devices," *IEDM Tech. Dig.*, pp. 494-497, 1987.

[59] J. G. Rollins and J. Choma Jr., "Mixed-Mode PISCES-SPICE Coupled Circuit and Device Solver," *IEEE Trans. Computer-Aided Design*, Vol. 7, No. 8, pp. 862-867, Aug. 1988.

[60] R. Klein, G. Lecarval, J. Gautier, B. Yachou, S. Motzny, and G. Rollins, "Implementation of the Heat Equation in a General-Purpose Device Simulator PISCES-2B," submitted for publication.

[61] S. S. Iyer, G. L. Patton, J. M. C. Stork, B. S. Meyerson, and D. L. Harame, "Heterojunction Bipolar Transistors Using Si-Ge Alloys," *IEEE Trans. Electron Devices,* Vol. 36, pp. 2043-2064, Oct. 1989.

[62] M. Shur, *Physics of Semiconductor Devices* (Englewood Cliffs, NJ: Prentice Hall, 1990).

[63] C. M. Snowden and D. Loret."Two-Dimensional Hot-Electron Models for Short-Gate-Length GaAs MESFET's," *IEEE Trans. Electron Devices,* Vol. ED-34, No. 2, pp. 212-223, Feb. 1987.

[64] Y-k. Feng and A. Hintz, "Simulation of Submicrometer GaAs MESFET's Using a Full Dynamic Transport Model," *IEEE Trans. Electron Devices,* Vol. 35, No. 9, pp. 1419-1431, Sept. 1988.

[65] M. V. Fischetti, "Monte Carlo Simulation of Transport in Technologically Significant Semiconductors of the Diamond and Zinc-Blende Structures—Part I: Homogeneous Transport," *IEEE Trans. Electron Devices,* Vol. 38, pp. 634-649, March 1991.

[66] P. A. Sandborn, J. R. East and G. Haddad, "Quasi-Two-Dimensional Modeling of GaAs MESFET's," *IEEE Trans. Electron Devices,* Vol. ED-34, No. 5, pp. 985-991, May 1987.

[67] V. Agostinelli, T. Bordelon, W. Wang, C. Yeap, and A. Tasch, "An Energy-Dependent Two-Dimensional Substrate Current Model for the Simulation of Submicron MOSFET's," *IEDL,* Vol. 13, No. 11, pp. 554-556, 1992.

[68] M. Born and E. Wolf, *Principles of Optics* (New York: Pergamon, 1980).

[69] M. C. Vecchi, M. Rudan, and G. Soncini, "Numerical Simulation of Optical Devices," *IEEE Trans. on Comp. Design,* Vol. 12, pp.1557-1569, Oct. 1993.

[70] E. D. Palik, *Handbook of Optical Constants of Solids* (Academic Press, 1985).

[71] C. M. Wolfe, et al., *Physical Properties of Semiconductors* (Englewood Cliffs, NJ: Prentice-Hall, 1989).

[72] B. K. Ridley, *Quantum Process in Semiconductors* (Oxford, England: Oxford Press, 1989).

[73] F. Urbach, *Phys. Rev.*, 140, A2059, 1965.

[74] G. D. Cody, *Phys. Rev. Lett.*, Vol. 47, No. 20, pp. 1480, 1981.

[75] U. Lindefelt, "Current-Density Relations for Nonisothermal Modeling of Degenerate Heterostructure Devices," *J. Appl. Phys.*, pp. 958-966, Jan. 1994.

[76] U. Lindefelt, "Equations for Electrical and Electrothermal Simulation of Anisotropic Semiconductors," J. *Appl. Phys.*, pp. 4162-4167, Oct. 1994.

[77] E. Velmre, et al., Proceedings of Simulation of Semiconductor Devices and Processes, Vienna, Austria, pp. 433-440, Sept. 1993.

[78] U. Lindefelt, "Heat Generation in Semicomductors," *J. Appl. Phys.*, pp. 942-957, Jan. 1994.

[79] H. Shin, A. F. Tasch, C. M. Mazier, and S. K. Banerjee, "A New Approach to Verify and Derive a Transverse-Field-Dependent Mobility Model for Electrons in MOS Inversion Layers," *IEEE Trans. Electron Devices*, Vol. 36 pp. 1117-1123, June 1989.

[80] V. M. Agostinelli, H. Shin, and A. F. Tasch, "A Comprehensive Model for Inversion Layer Hole Mobility for Simulation for Submicrometer MOSFET's," Vol. 38, pp. 151-159, Jan. 1991.

[81] S. A. Mujtaba, R. W. Dutton, and D. L. Scharfetter, "Semi-Empirical Local NMOS Mobility Model for 2-D Device Simulation Incorporating Screened Minority Impurity Scattering," *International Workshop on Numerical Modeling of Processes and Devices for Integrated Circuits: NUPAD V*, Honolulu, Hawaii, USA, June 5-6, 1994.

[82] K. Hess and G. J. Iafrate, "Modern Aspects of Heterojunction Transport Theory," *Heterojunction Band Discontinuities*, Edited by F. Capasso and G. Margaritondo (Amsterdam: Elsevier Science Publications), pp. 469, 1987.

[83] S. Luryi, "Hot Electron Injunction and Resonant Tunneling Heterojunction Devices," *Heterojunction Band Discontinuities*, Edited by F. Capasso and G. Margaritondo (Amsterdam: Elsevier Science Publications), pp. 526, 1987.

[84] M. J. van Dort, P. H. Woerlee, and A. J. Walker, "A Simple Model for Quantization Effects in Heavily-Doped Silicon MOSFET's at Inversion Conditions," S*olid-State Electronics*, 37, pp. 411-414, 1994.

[85] P. Vande Voorde, P. B. Griffin, Z. Yu, S.-Y. Oh, and R. W. Dutton, "Accurate Doping Profile Determination Using TED/QM Models Extensible to Sub-Quarter Micron nMOSFETs," *IEDM Tech. Dig.*, pp. 811-814, 1996.

[86] J.S. Blakemore, "Semiconducting and Other Major Properties of GaAs," *Journal of Applied Physics*, **53**(10), October 1982.

[87] M. Shur, *Physics of Semiconductor Devices*, pp 41, Prentice Hall 1990.

[88] M. Shur, *Physics of Semiconductor Devices*, Appendixes 8-12, Prentice Hall 1990.

[89] V. W. L. Chin, "Calculation of Carrier Concentrations and Fermi Energies in Intrinsic and Donor-Doped $In_{1-X}Ga_XAs$ ", *Solid-State Electronics*, **34**(11), pp. 1187-1190, 1991.

[90] Landolt-Bornstein, "Numerical Data and Functional Relationships in Science and Technology," **17e**. Springer-Verlag, Berlin, 1982.

[91] V. W. L. Chin, T. L. Tansley, "Alloy Scattering and Lattice Strain Effects on the Electron Mobility in $In_{1-X}Ga_XAs$ ", *Solid-State Electronics*, **34**(10), pp. 1055-1063, 1991.

[92] J. R. Hauser, M. A. Littlejohn, T. H. Glisson, "Velocity-Field Relationship of InAs-InP Alloys Including the Effects of Alloy Scattering," *Applied Physics Letters,* **28**(8), pp. 458-461, 1976.

[93] Choh-Fei Yeap, "UT-MINIMOS : A Hierarchical Transport Model Based Simulator for Deep Submicron Silicon Devices," the University of Texas at Austin, 1995.

[94] K.M. Cham, S-Y Oh, D. Chin, J.L. Moll, *Computer-Aided Design and VLSI Device Development*, 100, (Hingham, Massachhetts, Kluwer Academic Publishers, 1986).

[95] P. Vande Voorde, Private Communication, 1991.

[96] M.N. Darwish, J.L. Lentz, M.R. Pinto, P.M. Zeitzoff, T.J. Krutsick, and H.H. Vuong, "An Improved Electron and Hole Mobility Model for General Purpose Device Simulation," *IEEE Trans. on Electron Devices,* Vol. 44, No. 9, pp. 1529-1538, Sept., 1997.

[97] J.L. Egley and D. Chidambarrao, "Strain Effects on Device Characteristics: Implementation in Drift-Diffusion Simulators," *Solid-State Electronics,* **36**(12), pp. 1653-1664, 1993.

[98] G.L. Bir and G. E. Pikus, *Symmetry and Strain-Induced Effects in Semiconductors,* Wiley, New York (1974).

[99] G. Ottaviani, L. Reggiani, C. Canali, F. Nava, and A. Alberigi-Quaranta, "Hole drift velocity in silicon," *Phys. Rev B,* pp. 3318-3329 (1975).

[100] M. Shirahata, H. Kusano, N. Kotani, S. Kusanoki, and Y. Akasaka, "A Mobility Model Including the Screening Effect in MOS Inversion Layer," *IEEE Transactions on Computer-Aided Design,* pp. 1114-1119 (1992).

[101] S. Hareland, M. Manassian, W.-. Shih, S. Jallepalli, H. Wang, G. Chindalore, A. Tasch C. Maziar, "Computationally Efficient Models for Quantization Effects in MOS Electron and Hole Accumulation Layers MOS", *IEEE Transaction on Electron Devices*, pp. 1487-1492, July 1998.

[102] W. Harrison, "Tunneling from an independent-particle point of view," *Phys. Rev.*, (1961).

[103] J. Shiely, "Simulation of Tunneling in MOS Devices," Ph.D. Dissertation, Duke University, 1999.

[104] K.H. Gundlach, *Solid State Electron.* 949 (1966).

[105] M.V. Fischetti and S.E. Laux, "Band structure, deformation potentials, and carrier mobility in strained Si, Ge, and SiGe alloys," *Journal of Applied Physics,* pp. 2234-2252, Aug. 1996.

[106] C.Jungemann, R.Thoma and W.L.Engl, "A Soft Threshold Lucky Electron Model for Efficient and Accurate Numerical Device Simulation," *Solid-State Electronics,* Vol. 39, No. 7, pp.1079-1086, 1996.

Input Statement Descriptions

Introduction

The Medici program is directed via input statements. These statements may appear in a command input file or may be entered interactively from the terminal.

This chapter describes the statements recognized by Medici. The first section gives the general format of the input and defines the syntax used in the detailed documentation contained in the following sections. At the end of this chapter is a summary of the input statements.

Note:

Examples used as illustrations in this manual are not intended for use with actual simulations. They are presented as guidelines only.

Input Statements

This section describes the input statements and includes format. input limits, and syntax.

Format

Medici input statements are specified in a free format, and have the following characteristics:

- Each statement consists of a statement name followed by a list of parameter names and values.
- A statement may occupy more than one line by using continuation lines.

- An input line is a continuation line if the first nonblank character is a plus (+) (not available during interactive input mode) or if the last nonblank character of the previous input line was a plus (+).

Note:

Statements may only be broken for continuation between parameter specifications.

- Only the first 80 characters (including blanks) of each line are processed. If the first 80 characters of a line are all blank, the line is ignored.
- Nonprinting characters such as backspace, horizontal tabulation, line feed, form feed, and carriage return are converted to blanks.

Input Limits

The input to Medici may consist of at most:

- 1000 input statements
- 2000 input lines (including blank lines)
- 60,000 characters used to specify the input statements

These limits apply to the complete input, including statements entered interactively and through **`CALL`** statements.

Syntax

Valid statement and parameter names are those defined by the Medici keyfiles, described in the Introduction. Each name consists of one to eight consecutive nonblank characters. Names may be abbreviated by omitting characters from the end, provided that the abbreviation is unambiguous.

Appending

It is permissible to append extra characters to a name. For example, **`I.P`** and **`I.PRINTALL`** are both acceptable statement names, but **`I.`** is not because it is ambiguous (it could be **`I.PRINT`** or **`I.SAVE`**), and **`I.PALL`** is not because it does not match a valid statement name.

Statements with Parameters

Every statement begins with a statement name, which may be followed by parameter names with associated values. Some parameters must be assigned values; in this case the value is separated from the parameter name by an equals character (=).

Parameter name/value pairs are separated from the statement name and from each other by blanks. Blanks are permitted anywhere, except within a name or a value.

Statements without Parameters

Some statements, such as the **BATCH**, **RETURN**, and **STOP** statements, have no associated parameters. In this case the first input line of the statement consists of the statement name followed by a character value, while each continuation line contains only a character value.

The character value on each input line may be either a single character expression (see "Character Expressions" on page 3-9) or an arbitrary character string in which the first nonblank character is *not* a quote (**"**) or a commercial at (**@**).

Parameters

Parameters in Medici may be one of four types:

- Logical
- Numerical
- Array
- Character

The syntax for specifying the value of a parameter depends on its type.

Logical

Logical parameters have specific values which are assigned by various methods.

True or False Value

A logical parameter has a value of "true" or "false". The value is "true" if the parameter name appears by itself. The value is "false" if the parameter name is preceded by a *NOT* character (^, **!**, or **#**).

Character and Numerical Value

A logical parameter may also be assigned a logical value by following the parameter name with an equals character (=) and the logical value. Blanks on either side of the equals character are ignored.

The logical value may be specified with any valid numerical expression (see "Numerical Expressions" on page 3-5). The logical value is negated if the parameter name is preceded by a *NOT* character (^, **!**, or **#**).

Numerical

A numerical parameter is assigned a numerical value by following the parameter name with an equals character (=) and the numerical value. Blanks on either side of the equals character are ignored. The numerical value may be specified with any valid numerical expression.

Array

The value of an array-type parameter consists of a list of one or more numerical values.

Format

The general form of an array specification is

PARM(*i*)=(<V_1>, <V_2>, ..., <V_n>)

where <V_1>, <V_2>, and <V_n> are numerical values. The numerical values are enclosed in parentheses and separated by commas and/or blanks. If only one list value is specified, the parentheses may be omitted.

The index *i* specifies that the first value in the list should be assigned to element number *i* of the array; subsequent list values are assigned to subsequent array elements.

Each array specification must be contained on a single input line. To specify large arrays, increment the starting array index as shown in the following example:

PARM(01)=(<V_{01}>, <V_{02}>, ..., <V_{10}>) +
PARM(11)=(<V_{11}>, <V_{12}>, ..., <V_{20}>) +
PARM(21)=(<V_{21}>, <V_{22}>, ..., <V_{30}>) +
PARM(31)=(...

The index *i* and its enclosing parentheses may be omitted, in which case the assignment of list values starts with the first element of the array. The index *i* and the numerical list values may be specified with any valid numerical expressions.

A pair of commas (possibly separated by one or more blanks) in the value list denotes a null list value and leaves the corresponding array element unspecified.

Character

Character parameters have specific values which are assigned by various methods.

Value

A character parameter assumes the character value it is assigned by following the parameter name with an equals character (=) and the character value. Blanks on either side of the equals character are ignored. The character value may be specified with any valid character expression (see "Character Expressions" on page 3-9).

Length

The length of a character value may not exceed 80 characters.

Numerical Expressions

Numerical expressions may be used to specify the indices for array parameters and the values of logical, numerical, and array parameters. Blanks are not allowed in numerical expressions because blanks are used to separate parameter names and values.

Components and Use

Numerical expressions may contain the components listed in the following sections.

Numerical Values

Numerical values may be used as arguments to arithmetic operators, relational operators, numerical functions, and logical functions. Numerical constants may be specified with any valid FORTRAN integer, fixed point, or floating point decimal number representation.

As an example, the following are equivalent valid numerical constants:

```
.5
0.5
0.005E+2
+05D-1
```

Logical Values

Logical values may be used as arguments to logical operators and logical functions. The following logical constants are available:

true values	false values
`true`	`false`
`t`	`f`
`yes`	`no`
`y`	`n`

Assigned Names

Assigned names may be used in place of numerical or logical values as long as the assigned names are of type numerical or logical (see "ASSIGN" on page 3-396).

Character Expressions

Character expressions may be used as arguments to relational operators and logical functions. Character expressions may also be used as arguments to conversion functions as long as the values of the expressions represent valid numerical or logical values.

Delimiters

Delimiters establish precedence and separate function arguments:

( ) parentheses for delimiting groups

; semicolons for delimiting multiple arguments in functions

Arithmetic Operators

Arithmetic operators operate on a pair of numerical values and return a numerical value:

x **+** *y* addition

x **-** *y* subtraction

x ***** *y* multiplication

x **/** *y* division ()

x ****** *y* exponentiation ()

Relational Operators

Relational operators operate on a pair of numerical or character values and return a logical value:

x **<** *y* less than

x **<=** *y* less than or equal to

x **=** *y* equal to

x **^=** *y* not equal to

x **>** *y* greater than

x **>=** *y* greater than or equal to

Logical Operators

Logical operators operate on a single logical value or on a pair of logical values and return a logical value:

^ *x* logical negation (not)

x **&** *y* logical and

x **|** *y* logical or

Numerical Functions

Numerical functions operate on numerical values and return a numerical value:

exp(*x***)** exponential

log(*x***)** natural (base *e)* logarithm ($x > 0$)

log10(*x***)** common (base 10) logarithm ($x > 0$)

erf(*x***)** error function

erfc(*x***)** complementary error function

sqrt(*x***)** square root ($x \geq 0$)

sin(*x***)**	sine (*x* in radians)
cos(*x***)**	cosine (*x* in radians)
tan(*x***)**	tangent (*x* in radians)
asin(*x***)**	arcsine ($\lvert x\rvert \le 1\ ;\ -\pi/2 \le \mathrm{asin}(x) \le \pi/2$)
acos(*x***)**	arccosine ($\lvert x\rvert \le 1\ ;\ 0 \le \mathrm{acos}(x) \le \pi$)
atan(*x***)**	arctangent ($-\pi/2 \le \mathrm{atan}(x) \le \pi/2$)
atan2(*x***;***y***)**	arctangent ($\mathrm{atan}(x/y)\ ;\ y \ne 0$ if $x = 0\ ; -\pi < \mathrm{atan2}(x;y) \le \pi$)
sinh(*x***)**	hyperbolic sine
cosh(*x***)**	hyperbolic cosine
tanh(*x***)**	hyperbolic tangent
abs(*x***)**	absolute value
int(*x***)**	truncation
nint(*x***)**	nearest whole number ($\mathrm{int}(x + .5)$ if $x \le 0\ ;\ \mathrm{int}(x - .5)$ if $x < 0$)
mod(*x***;***y***)**	remaindering ($x - \mathrm{int}(x/y) \cdot y\ ;\ y \ne 0$)
sign(*x***;***y***)**	transfer of sign ($\lvert x\rvert$ if $y \ge 0\ ;\ -\lvert x\rvert$ if $y < 0$)
dim(*x***;***y***)**	positive difference ($x - y$ if $x > y\ ;\ 0$ if $x \le y$)
min(*x***;***y***;...)**	choosing smallest value (maximum of 80 arguments)
max(*x***;***y***;...)**	choosing largest value (maximum of 80 arguments)

Logical Functions

Logical functions operate on a logical, numerical, or character value and return a logical value:

ltype(*x***)**	true if *x* is a logical value; false otherwise
ntype(*x***)**	true if *x* is a numerical value; false otherwise
ctype(*x***)**	true if *x* is a character value; false otherwise

Conversion Functions

Conversion functions operate on a character value and return a logical or numerical value:

lval(*x***)**	convert the character value *x* to the equivalent logical value
nval(*x***)**	convert the character value *x* to the equivalent numerical value

Component Precedence

The components of numerical expressions are evaluated according to the following precedence order:

1. Groups delimited by parentheses evaluated from inner to outer
2. Functions evaluated from left to right
3. Exponentiation operators evaluated from right to left
4. Multiplication and division arithmetic operators evaluated from left to right
5. Addition and subtraction operators evaluated from left to right
6. Relational operators evaluated from left to right
7. Logical negation operators evaluated from left to right
8. Logical and operators evaluated from left to right
9. Logical or operators evaluated from left to right

Note:

Conversion occurs between logical and numerical values in some situations. Logical values are converted from true to 1.0 and from false to 0.0 when they appear as arguments to arithmetic operators, relational operators, and numerical functions. Numerical values are converted from nonzero to true and from zero to false when they appear as arguments to logical operators.

Examples of Numerical Expressions

The following table shows some examples of numerical expressions:

Expression	Value
`(2+sqrt(5))*(4/2**3)`	2.12
`@VAL1*(@VAL2+1.E12)`	2.2E13 (for *VAL1*=2, *VAL2*=1E13)
`2*3/4*5`	7.5
`2*3+4+exp(6/5)`	13.32
`2<5`	true
`"aa">"ab"`	false
`^(2<5)|"aa">"ab"`	false
`(2<5)&true`	true
`1 + 1`	(***invalid***—contains blanks)

In the above examples *VAL1* and *VAL2* are assigned names, discussed with the description of the **`ASSIGN`** statement (see "Controlling Program Execution" on page 3-362).

Character Expressions

Character expressions may be used to specify the values of character parameters and the contents of statements, such as the **BATCH**, **RETURN**, and **STOP** statements, which have no associated parameters.

They may also appear in numerical expressions (see "Numerical Expressions" on page 3-5) as arguments to relational operators, logical functions, and conversion functions.

Syntax

Character expressions may be either nonblank character strings or concatenations of any combination of character strings enclosed in quotes (**"**) and assigned names. Blanks are only allowed in character expressions within quoted character strings. Character expressions may not be continued from one input line to the next.

Length

The length of a character expression may not exceed 80 characters after replacement of assigned names and removal of quotes around quoted character strings.

Character Expression Examples

The following table shows some examples of character expressions:

Expression	Value
string	string
"this string"	this string
"this"" string"	this string
@VAL1" string"	this string (for *VAL1*="this")
@VAL1@VAL2	this string (for *VAL1*="this", *VAL2*="string")
this string	(***invalid***—contains blanks and is not enclosed in quotes)
'this string'	(***invalid***—contains blanks and is not enclosed in quotes)

Statement Description Format

The description of each statement in this manual consists of a formatted list of the parameters associated with the statement. This list indicates if the parameter is optional, and valid combinations of parameters. This is followed by a parameter definition table.

Parameter Definition Table

The parameter definition table includes the following:

- Parameter name
- Parameter type
- The parameter's function, and synonyms (if any) which can be used instead of the standard parameter name
- The default value that Medici uses for the parameter in the absence of a user-specified value
- Physical units (if any) for a numerical or array parameter

Parameter Type

The parameter type is specified as one of the following:

- **logical** - logical parameter
- **number** - numerical parameter
- **array** - array parameter
- **char** - character parameter

Example

The following is an example of the statement description format used in this manual.

X.MESH

```
{ LOCATION=<n> | ( {WIDTH=<n> | X.MAX=<n>} [X.MIN=<n>] ) }
[ {NODE=<n> | N.SPACES=<n>} ]
[ {SPACING=<n> | H2=<n>} ] [H1=<n>] [H3=<n>]
[RATIO=<n>] [MIN.SPAC=<n>] [SUMMARY]
```

Parameter	Type	Definition	Default	Units
LOCATION	number	The x location where a node is to be placed.	none	microns
WIDTH	number	The width of the grid section.	none	microns
X.MAX	number	The x location of the right edge of the grid section. **synonyms: X.RIGHT**	none	microns

Syntax of Parameter Lists

The following special characters are used in the formatted parameter list that appears at the beginning of each statement description.

- Angle brackets < >
- Square brackets **[ ]**
- Braces **{ }**
- Vertical bar **|**
- Parentheses **( )**

Note:

The special characters, **< >, [ ], |, { }, *and* ( ), *indicate parameter types, optional groups, alternate choices, and group hierarchy. They should not form part of the actual input to*** Medici ***(i.e., special characters are not typed in). Only the information enclosed in the special characters is typed into command strings.***

Value Types < >

A lower case letter in angle brackets represents a value of a given type. The following types of values are represented:

<n> - numerical value

<a> - array value

<c> - character value

For example,

```
PARM1=<n>
```

indicates that the **PARM1** parameter is assigned a numerical value.

Defining Groups { }, [ ], ( ), |

Braces, parentheses, square brackets, and vertical bars are used to define groups of parameters or groups of groups. For example,

```
{( PARM1 [PARM2 [PARM3]] PARM4 ) PARM5}
```

is a valid group, composed of the subgroups **(PARM1 [PARM2 [PARM3]] PARM4)** and **PARM5**.

The first subgroup may further be subdivided into the subgroups **PARM1**, **[PARM2 [PARM3]]**, and **PARM4**, and so on.

Optional Groups [ ]

Square brackets enclose groups that are optional. For example,

```
STMT1 [PARM1] [ PARM2 PARM3 ] [ PARM4 [PARM5] ]
```

indicates that in the **STMT1** statement, the parameter **PARM1** is optional. The group **[PARM2 PARM3]** is optional, but if **PARM2** is specified, **PARM3** must also be specified.

The group **[PARM4 [PARM5]]** is optional, but **PARM5** may be specified only if **PARM4** is specified.

List of Groups {}, |

When one of a list of groups must be selected, the groups are enclosed in braces and separated by vertical bars. For example,

```
STMT2 {PARM1 | PARM2 | ( PARM3 PARM4 )}
```

indicates that the **STMT2** statement requires that one of the three groups **PARM1**, **PARM2**, or **(PARM3 PARM4)** be specified.

Group Hierarchy ( )

Parentheses enclose groups that are to be considered as single items in higher level groupings. For example, in the above **STMT2** statement, the group **(PARM3 PARM4)** constitutes one of three possible choices and is therefore enclosed in parentheses.

3.1 Device Structure Specification

The following statements specify the device structure used by Medici:

Statement	Definition	Page
MESH	Initiates the mesh generation.	3-15
X.MESH	Specifies the placement of grid lines perpendicular to the x-axis.	3-32
Y.MESH	Specifies the placement of grid lines perpendicular to the y-axis.	3-35
ABC.MESH	Specifies the spacing parameters for ABC meshing.	3-38
ELIMINATE	Eliminates nodes along grid lines.	3-50
SPREAD	Adjusts the vertical position of nodes along horizontal grid lines.	3-52
BOUNDARY	Modifies a simulation mesh to conform to boundary interfaces read from a file.	3-55
TSUPREM4	Defines the regions and profiles in a Medici rectangular grid structure by importing a TSUPREM-4 structure saved in Medici format.	3-67
REGION	Specifies the location of material regions in the structure.	3-70
ELECTRODE	Specifies the location of electrodes in the structure.	3-76
RENAME	Changes the name of an electrode or region.	3-81
PROFILE	Specifies impurity profiles for the structure.	3-82
REGRID	Refines the simulation mesh.	3-94
STITCH	Appends the generated structure to the simulation mesh.	3-101

Rectangular Mesh Specification

A rectangular mesh can be a very effective solution mesh. Some alternatives for mesh creation include:

- For planar devices, long-channel MOSFETs, and large structures used to simulate some power devices, a rectangular grid is the method of choice. Large aspect rectangles minimize the amount of grid allocated. Certain properties of the resulting matrix can be exploited to reduce solution time.
- For other structures, especially those with complicated doping profiles, start with a coarse rectangular mesh and use regridding.

Mesh Statements

Rectangular meshes are specified by a sequence of mesh statements, detailed in Chapter 2, "Grid in Medici" on page 2-74. The required input order is:

- **MESH** statement
- **X.MESH** statements
- **Y.MESH** statements
- **ELIMINATE** statements (optional)
- **SPREAD** statements (optional)
- **BOUNDARY** statements (optional)
- **TSUPREM4** statements (optional)
- **REGION** statements
- **ELECTRODE** statements

Note:

The order in which statements appear is mandatory. Changing the order will change the results.

Generally, a mesh is specified by the following steps:

1. The mesh begins as a set of (nonuniform) spaced *x*- and *y*-lines comprising a simple rectangle.
2. The rectangle can then be distorted to track non-planar geometry or match the doping profile, although strongly non-planar structures are difficult to treat in this way.
3. Mesh lines may be terminated inside the device, and redundant nodes removed from the grid.
4. Material regions and electrodes can then be specified as a union of (possibly distorted) rectangles, completing the mesh specification.

CAUTION

When a rectangular grid is distorted, a large number of obtuse triangles are unavoidably introduced. When you regrid a rectangular grid, large aspect ratio rectangles (>2.8) can also give rise to very obtuse triangles. See "Mesh Smoothing" on page 2-84 for more information.

MESH

The **MESH** statement initiates the mesh generation or reads a previously generated mesh.

```
MESH

 Initial Mesh Generation
   {  ( [ {RECTANGU | CYLINDRI} ] [DIAG.FLI] )

 Mesh File Input
    | ( IN.FILE=<c> [PROFILE]
        [ {  ASCII.IN
           | ( TSUPREM4 [ ELEC.BOT [Y.TOLER=<n>] ] [POLY.ELE]
               [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
               [FLIP.Y] [SCALE.Y=<n>]
             )
           | ( TIF [ ELEC.BOT [Y.TOLER=<n>] [POLY.ELE] ] )
          }
        ]
 New Automatic Boundary Conforming Mesh Generation
       [{ ( ABC
              [GRIDTOP] [VOIDELEC] [RFN.CRNR] [JUNC.ABC]
              [CRITICAL=<n>] [SPACING=<n>]
              [N.SEMICO=<n>] [N.INSULA=<n>] [N.CONDUC=<n>]
              [NORMGROW=<n>] [LATERAL=<n>] [ELIMINAT]
            ) |
 Old Automatic Boundary Conforming Mesh Generation
          ( ABC.OLD
              [SPACING=<n>][RATIO=<n>] [ANGLE=<n>]
              [MAX.SPAC=<n>] [CORNER=<n>] [LAYERS=<n>]
              [ATTEMPTS=<n>] [RELAX=<n>] [ELIMINAT]
              [EXTERNAL=<n>] [OPTIMIZE]   [JUNC.ABC]
              [N.SEMICO=<n>] [N.INSULA=<n>] [N.CONDUC=<n>]
              [N.SILICO=<n>] [N.POLYSI=<n>] [N.OXIDE=<n>]
              [N.NITRID=<n>] [N.OXYNIT=<n>] [N.SAPPHI=<n>]
              [N.BPSG=<n>]   [N.INAS=<n>]   [N.GAAS=<n>]
              [N.ALGAAS=<n>] [N.HGCDTE=<n>] [N.S.OXID=<n>]
              [N.SIC=<n>]    [N.INGAAS=<n>] [N.INP=<n>]
              [N.GERMAN=<n>] [N.DIAMON=<n>] [N.ZNSE=<n>]
              [N.ZNTE=<n>]   [N.A-SILI=<n>] [N.SIGE=<n>]
          )  |
 Quadtree Mesh Generator
          ( QUADTREE
              [MIN.THIC=<n>] [MINDEL=<n>] [MAXDEL=<n>]
          )
        }
       ]
      )
   }
```

(**MESH** statement continued on next page)

(**MESH** statement continued from the previous page)

```
Mesh Adjustments
  [PERIODIC [PBC.TOL=<n>] ] [ORDER] [ADJUST] [VIRTUAL]
  [CENTROID] [ WIDTH=<n> N.SPACES=<n> [X.SPLIT=<n>] ]
  [OBTUSE.A=<n>]

Mesh File Output
  [ OUT.FILE=<c> [ASCII.OU] [NO.TTINF] ] [SMOOTH.K=<n>]
```

Parameter	Type	Definition	Default	Units
Initial Mesh Generation				
RECTANGU	logical	Specifies that the simulation mesh uses rectangular coordinates.	true	
CYLINDRI	logical	Specifies that the simulation mesh uses cylindrical coordinates. If this parameter is specified, the horizontal axis represents the radial direction and the vertical axis represents the z-direction.	false	
DIAG.FLI	logical	Specifies that the direction of diagonals is changed about the horizontal center of the grid. If this parameter is false, all diagonals are in the same direction.	true	
Mesh File Input				
IN.FILE	char	The identifier for the file containing a previously generated mesh. Unless **ASCII.IN** or **TSUPREM4** is specified, this file is in binary format. **synonym: INFILE**	none	
PROFILE	logical	Specifies that impurity profiles are input from the data file.	true	
ASCII.IN	logical	Specifies that the input mesh is stored in a formatted file.	false	
TSUPREM4	logical	Specifies that the input mesh was generated by TSUPREM-4 or by an external grid editor.	false	
ELEC.BOT	logical	Specifies that an electrode is added to the structure at the maximum y coordinate value.	false	
Y.TOLER	number	The maximum distance by which the y coordinate of a node can deviate from the maximum y coordinate value in the device and still be considered part of an electrode that is added using the **ELEC.BOT** parameter. This is useful if the bottom edge of the device is non-planar. This parameter is valid only if **TSUPREM4** is specified.	0	microns
POLY.ELE	logical	Specifies that regions defined as polysilicon in the data file are treated as electrodes.	true	
X.MIN	number	The minimum x coordinate read in from the data file. All nodes and elements with x coordinates less than the value specified by **X.MIN** are eliminated from the structure. This parameter is valid only if **TSUPREM4** is specified.	The minimum x location in the structure.	microns
X.MAX	number	The maximum x coordinate read in from the data file. All nodes and elements with x coordinates greater than the value specified by **X.MAX** are eliminated from the structure. This parameter is valid only if **TSUPREM4** is specified.	The maximum y location in the structure.	microns

Parameter	Type	Definition	Default	Units
Y.MIN	number	The minimum y coordinate read in from the data file. All nodes and elements with y coordinates less than the value specified by **Y.MIN** are eliminated from the structure. This parameter is valid only if **TSUPREM4** is specified.	The minimum y location in the structure.	microns
Y.MAX	number	The maximum y coordinate read in from the data file. All nodes and elements with y coordinates greater than the value specified by **Y.MAX** are eliminated from the structure. This parameter is valid only if **TSUPREM4** is specified.	The maximum y location in the structure.	microns
FLIP.Y	logical	Specifies that the direction of the vertical coordinate is reversed when the file is read. This parameter is valid only if **TSUPREM4** is specified.	false	
SCALE.Y	number	The scale factor to multiply all coordinate values by when reading the mesh file. This parameter is valid only if **TSUPREM4** is specified.	none	none
TIF	logical	Specifies that the file to be read in is in the TIF (Technology Interchange Format).	false	

New Automatic Boundary Conforming Mesh Generator

ABC	logical	Specifies that the device structure imported from the input file is remeshed using the new automatic boundary conforming mesh generator.	false	
GRIDTOP	logical	Grid the top boundary of the device.	true	
VOIDELEC	logical	Mesh volume electrodes without introducing any interior nodes.	false	
RFN.CRNR	logical	Automatically refine the grid spacing on region boundaries near corners where two or more regions meet.	true	
JUNC.ABC	logical	Specifies that the mesh should conform to junctions in addition to boundaries.	false	none
CRITICAL	number	Maximum allowed deviation of the region boundaries in the new mesh from those in the original mesh.	.001*minimum of device width and device height	microns
SPACING	number	Desired grid spacing along the boundaries and interfaces. The actual mesh spacing along the boundaries and interfaces may be smaller in order to satisfy the **CRITICAL** parameter.	(device width) / 50	microns
N.SEMICO	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for all semiconductor regions.	0.1	none
N.INSULA	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for all insulator regions.	1.0	none
N.CONDUC	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for all conductor regions.	.2	none
NORMGROW	number	Multiplication factor for the grid spacing normal to the region boundary. The element size grows in the normal direction from the region boundary toward the inside of the region. Each next grid spacing in the direction toward the inside of the region is larger than the previous spacing by the specified factor. If the factor is one, the mesh spacing is uniform throughout the region. The factor should be larger than one to obtain a coarser mesh away from the region boundaries. **synonym: RATIO**	1.2	none

Parameter	Type	Definition	Default	Units
LATERAL	number	Target lateral spacing during region meshing. This parameter is used to indicate the desired lateral spacing during the interior meshing of the regions. The actual lateral spacing obtained during meshing will be refined or unrefined as necessary in order to stay close to the specified value. Refinement or unrefinement of the lateral spacing will only occur, however, if non-obtuse elements can be produced. Very large values of **LATERAL** lead to unrefinement whenever non-obtuse elements can be produced. This is equivalent to using the **ELIMINAT** parameter.	maximum of device width and device height	microns
ELIMINAT	logical	Minimizes the number of nodes in the mesh by eliminating interior nodes without creating obtuse elements. **synonym: COARSEN**	true	

Old Automatic Boundary Conforming Mesh Generation

ABC.OLD	logical	Specifies that the device structure imported from the input file is remeshed using the automatic boundary conforming mesh generator.	false	
SPACING	number	Desired grid spacing along the boundaries and interfaces. The actual mesh spacing along the boundaries and interfaces can be smaller due to surface curvature. **synonym: STEP**	(device width) / 50	microns
RATIO	number	Multiplication factor for the grid spacing normal to the region boundary. The element size grows in the normal direction from the region boundary toward the inside of the region. Each next grid spacing in the direction toward the inside of the region is larger than the previous spacing by the specified factor. If the factor is one, the mesh spacing is uniform throughout the region. The factor should be larger than one to obtain a coarser mesh away from the region boundaries. **synonym: FACTOR**	1.2	none
ANGLE	number	Threshold angle for geometry smoothing. Allows unwanted small zig-zag type geometry features in the imported structure to be removed if deviation from a straight line is less than the specified threshold angle for any pair of adjacent edges along the region's boundary.	30	degrees
RELAX	number	Relaxation factor for mesh refinement near curved boundaries. The mesh along curved region boundaries and interfaces is automatically refined to avoid obtuse elements. A **RELAX** factor value of less than one guarantees no obtuse elements near the boundaries, but the mesh is harder to generate. On the contrary, a **RELAX** factor value larger than one allows some obtuse elements near the boundaries, but the mesh is easier to generate. **synonym: LOOSE**	1	none
MAX.SPAC	number	Maximum allowed grid spacing in the normal direction to the region boundaries. **synonym: MAX.STEP**	50	microns
ELIMINAT	logical	Minimizes the number of nodes in the mesh by eliminating interior nodes without creating obtuse elements. **synonym: COARSEN**	true	
CORNER	number	Corner smoothing factor for the interior boundary conforming grid lines. If this factor is zero, the interior grid lines adjacent to the region's boundary maintain the boundary shape. If it is nonzero, then each subsequent grid line adjacent to the boundary is smoother than the previous one (the curvature of the "corner" is reduced). The corner smoothing factor should be larger or equal to zero and smaller than one.	0.9	none
LAYERS	number	Number of allowed mesh "layers" of the boundary conforming type before switching to the unstructured Delaunay mesh in the region's interior.	500	none

Parameter	Type	Definition	Default	Units
ATTEMPTS	number	Number of allowed attempts to mesh the device when the meshing fails for the specified set of meshing parameters. Each subsequent attempt "relaxes" the meshing parameters.	4	none
EXTERNAL	number	Ratio of lateral spacing along external boundaries of the structure to the lateral spacing along internal boundaries (which is specified by the parameter **SPACING**). It is recommended that the value of the ratio is larger than one and less than ten in order to create a good mesh with minimum nodes on external boundaries.	4.0	none
OPTIMIZE	logical	Optimizes mesh quality by flipping all mesh edges whenever it improves quality of the adjacent triangles.	true	none
JUNC.ABC	logical	Specifies that the mesh should conform to junctions in addition to boundaries.	false	none
N.SEMICO	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for all semiconductor regions.	0.1	none
N.INSULA	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for all insulator regions.	0.1	none
N.CONDUC	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for all conductor regions. **synonyms: N.ELECTR**, **N.ALUMIN**, **N.TERMIN**, **N.METAL**	0.2	none
N.SILICO	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for silicon.	0.1	none
N.POLYSI	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for polysilicon.	0.2	none
N.OXIDE	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for oxide. **synonym: N.SIO2**	100	none
N.NITRID	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for nitride. **synonym: N.SI3N4**	100	none
N.OXYNIT	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for oxynitride.	100	none
N.SAPPHI	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for sapphire.	100	none
N.BPSG	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for BPSG.	100	none
N.INAS	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for InAs.	0.1	none
N.GAAS	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for GaAs.	0.1	none
N.ALGAAS	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for AlGaAs.	0.1	none
N.SIGE	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for SiGe.	0.1	none
N.HGCDTE	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for HgCdTe.	0.1	none
N.INGAAS	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for InGaAs.	0.1	none
N.SIC	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for SiC.	0.1	none
N.S.OXID	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for oxide when it is simulated as a wide band gap semiconductor.	0.1	none

Parameter	Type	Definition	Default	Units
N.GERMAN	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for Ge.	0.1	none
N.INP	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for InP.	0.1	none
N.DIAMON	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for diamond.	0.1	none
N.ZNSE	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for ZnSe.	0.1	none
N.ZNTE	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for ZnTe.	0.1	none
N.A-SILI	number	Mesh aspect ratio (normal spacing/lateral spacing) at boundaries for amorphous silicon.	0.1	none
Quadtree Mesh Generator				
QUADTREE	logical	Specifies that the device structure imported from the input file is remeshed using the Quadtree mesh generator.	false	
MIN.THIC	number	The minimum thickness of any region in the device.	50e-4	microns
MINDEL	number	The minimum mesh spacing allowed in the new Quadtree mesh.	**MIN.THIC**	microns
MAXDEL	number	The maximum mesh spacing allowed in the new Quadtree mesh.	device width/4	microns
Mesh Adjustments				
PERIODIC	logical	Specifies that periodic boundary conditions are used in the horizontal direction. If this parameter is false, then reflection symmetry boundary conditions are used at the left and right edges of the device structure.	false	
PBC.TOL	number	The tolerance allowed between left and right edge nodes when determining whether a structure is valid for periodic boundary conditions.	0.1	fraction of local grid spacing
ORDER	logical	Specifies that the mesh nodes and mesh elements are ordered from left to right and from top to bottom.	True for structures created by Medici or if **PERIODIC** is specified; otherwise, false.	
ADJUST	logical	Specifies that the triangle diagonals are flipped whenever it improves (smooths) the shape of regions and electrode.	false	
VIRTUAL	logical	Generates virtual nodes at heterojunctions.	false	
CENTROID	logical	Specifies that, when an element produces a zero coupling term along an interface, a centroidal type discretization will be applied to the element. Use of this parameter normally results in a more stable discretization with reduced sensitivity to the mesh. A few cases have been observed where it is better not to use the centroids. These cases occur when Fowler-Nordheim or Band-to-Band currents are large, such as in some EEPROM devices.	true	
WIDTH	number	The width of a grid section that is inserted into the completed simulation structure at the x coordinate location specified with **X.SPLIT**. This parameter is particularly useful for varying the size of simulation structures that already exist and are read in using the **IN.FILE** parameter.	none	microns

Parameter	Type	Definition	Default	Units
N.SPACES	number	The number of grid spaces to use in the grid section that is inserted into the completed simulation structure.	none	none
X.SPLIT	number	The x coordinate location where an additional grid section is inserted into the completed simulation structure.	The midpoint of the structure.	microns
OBTUSE.A	number	The threshold angle that is considered obtuse. All elements with angles larger than the specified threshold angle will be reported as obtuse on the mesh statistics print-out and shown as obtuse on the plot. This parameter does not affect the way the elements are treated for the carrier continuity equations, where obtuse elements are those with negative coupling coefficients.	93	degrees
Mesh Output File				
OUT.FILE	char	The identifier for the output file to store the generated mesh. **synonym: OUTFILE**	none	none
ASCII.OU	logical	Specifies that the output mesh is stored in a formatted file. If this parameter is not specified, the output mesh is stored in a binary file.	false	
NO.TTINF	logical	Specifies that no triangle tree information is written to the output mesh.	false	
SMOOTH.K	number	Specifies a method for mesh smoothing. **SMOOTH.K**=1 indicates triangular smoothing is used, maintaining all region boundaries fixed. **SMOOTH.K**=2 indicates triangular smoothing is used, maintaining only material boundaries. **SMOOTH.K**=3 indicates that node averaging is used.	none	none

Description

A **MESH** statement can be used to either initiate the generation of a device structure or to read a previously generated device structure from a data file. It is possible to include several **MESH** statements in a single input file in order to perform simulations for multiple device structures. Whenever a **MESH** statement is encountered in an input file, Medici performs an initialization that allows a completely new simulation to be started.

To further illustrate the **MESH** statement refer to input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Mesh" on page 4-3.

Generating an Initial Mesh

This section describes how the **MESH** statement is used to generate an initial simulation structure.

Cartesian and Cylindrical Coordinates

When using the **MESH** statement to initiate the generation of a structure, you have the choice of the following coordinates:

- Cartesian
- Cylindrical

The default is to use Cartesian coordinates (**RECTANGU**). In this case, simulations are be performed in an xy-plane with the x-axis going from left to right and the y-axis going from top to bottom.

If cylindrical coordinates are chosen (**CYLINDRI**), the simulations are performed in an rz-plane. In this case, the Cartesian x-axis becomes the cylindrical r-axis and the Cartesian y axis becomes the cylindrical z-axis.

Statement Specification

Generating an initial simulation structure requires the specification of several statements in the proper order. Statement specification should be as follows:

1. Specify the **MESH** statement.
2. Specify the **X.MESH** and **Y.MESH** statements.

   These statements are used to define the initial placement of nodes.
3. There is now a choice between three sets of optional statements which can be used to modify the node placement.
   a. **ELIMINATE** statements may be used to eliminate unnecessary nodes and **SPREAD** statements may be used to distort the initial mesh.
   b. A **BOUNDARY** statement may be used to modify the initial mesh to conform to arbitrary boundaries.
   c. **TSUPREM4** statements may be used to define regions and profiles from the results of a TSUPREM-4 simulation.
4. Define the following for the structure:
   a. Material regions using the **REGION** statement.
   b. Electrode placement using the **ELECTRODE** statement.
   c. Impurity profiles using the **PROFILE** statement.
5. The generated mesh can be saved for a future simulation using the **OUT.FILE** parameter on the **MESH** statement, or by using the **SAVE** statement.
6. The initial mesh can be refined further, if desired, using the **REGRID** statement. In this case, the refined mesh should be saved using the **OUT.FILE** parameter on the **REGRID** statement, or by using a **SAVE** statement following the regrid.

Mesh Smoothing

The **SPREAD** statement has a tendency to create triangles with very obtuse angles which may lead to unphysical solutions or poor convergence. If the **SPREAD** statement is to be used in the mesh generation sequence, the **SMOOTH.K** parameter should be specified on the **MESH** statement.

SMOOTH.K=1 or 2 indicates triangle smoothing is used. Each adjoining pair of triangles is examined, and the diagonal of the quadrilateral is flipped if it improves

the quality of the triangles. When two elements are of different materials, the diagonal is never flipped.

With elements of the same material but different region number, the diagonals are not flipped if **SMOOTH.K**=1 and may be flipped if **SMOOTH.K**=2. **SMOOTH.K**=3 indicates node smoothing, which repositions nodes in order to improve the angles of the triangles surrounding it.

Note:

Node smoothing is not recommended for a refined or distorted mesh since it tends to redistribute fine grid away from areas where the physical properties of the structure require it.

Periodic Boundary Conditions

Periodic boundary conditions in the horizontal direction are imposed if the parameter **PERIODIC** is specified. In this case, the left and right edges of the simulation region should be identical in terms of number of nodes, vertical location of nodes, doping, and electrode specifications.

If a different number of nodes exist on the left and right edges (defined as the minimum and maximum horizontal coordinates) or if the corresponding nodes are located at different vertical locations, an error message is issued. To ensure that the same doping is used on the left and right edges, the program automatically copies the doping from the left edge to the right edge.

Finally, if an electrode contacts one edge of the structure, but not the other, the program automatically converts the corresponding nodes on the other edge into electrode nodes, as well.

Note:

Periodic boundary conditions are not allowed in structures that are used in circuits when using the CA-AAM.

Automatic Boundary Conforming Mesh Generation

If a structure is read in from a TSUPREM-4, TIF, or Medici file using the **IN.FILE** parameter, a new mesh for the structure can be generated using the **ABC** parameter on the **MESH** statement.

If you specify the **ABC** parameter, the previous mesh from the imported structure is discarded and a new mesh is created using the automatic boundary conforming meshing algorithm. To provide better spacing control and improved robustness, a new version of the **ABC** mesh generator has been developed. You can still access the old version using the **ABC.OLD** parameter.

Algorithm

If a mesh is generated using **X.MESH** and **Y.MESH** statements, grid lines are aligned with the cartesian X and Y axes. Manual specification of the mesh spacings is required throughout the mesh. In some cases, the original geometry is distorted by shifting the location of the region boundaries to the nearest mesh nodes.

The automatic boundary conforming (ABC) meshing algorithm creates grid lines that conform to the region boundaries. The ABC algorithm also conserves the shape of the original regions and may be used without specifying meshing parameters. However, the ABC algorithm may fail to generate a mesh for a region with extremely complicated geometry.

The ABC mesh generator creates "layers" of grid that conform to the boundaries (both internal and external) of the original structure. The lateral grid spacing at the boundaries along these layers can be specified on a global basis with the **SPACING** parameter. In the new ABC mesh generator, non-uniform lateral spacing along the boundaries can be specified on a region-by-region basis using the **ABC.MESH** statement. The actual spacing along the boundaries may be smaller than the specified spacing in locations where the boundary is curved.

The normal grid spacing at the boundaries (thickness of the first grid layer) is specified as a ratio of the desired normal spacing to the value of the **SPACING** parameter using the **N.SEMICO**, **N.INSULA**, and **N.CONDUC** parameters for semiconductors, insulators, and conductors, respectively. The old version of ABC also allows specification of normal spacing ratios for other materials. The new version allows a more precise specification of normal spacing on a material or region basis and also allows the specification of non-uniform normal spacings through the **ABC.MESH** statement. The normal grid spacing away from the boundary increases toward the region's interior by a factor of **RATIO** or **NORMGROW** for each subsequent grid layer. The boundary conformal grid layers are created for materials that have values of the normal grid spacing much less than 1.0. For materials with the values of the normal grid spacing that are comparable or larger than 1.0, an unstructured mesh is often generated.

New Automatic Boundary Conforming Mesh Generator

The new version of the ABC mesh generator uses the same basic algorithm as the original ABC mesh generator. Layers of mesh elements are constructed to conform to the region boundaries according to spacing parameters provided by the user. The new version has been made more robust and, in particular, is capable of handling jagged boundaries and regions with multiple embedded regions. In addition, you can now specify spacing parameters on a local basis via the **ABC.MESH** statement. The **MESH** statement in conjunction with the **ABC** parameter and other associated parameters is used to initiate the ABC meshing process and to specify global meshing properties. For details on the new ABC mesh generator, please see the description of the **ABC.MESH** statement on page 3-42.

Spacing Parameters

The **MESH** statement in conjunction with the **ABC** parameter is used to specify global values of spacing parameters used for all regions and region boundaries. Some of the parameters used by the original ABC mesh generator are also used by

the new ABC mesh generator. Many of the original parameters, however, are not applicable to the new ABC mesh generator, and a number of new parameters have been created to support additional functionality.

Gridding Region Boundaries

The ABC meshing of a device begins by re-gridding the boundaries between regions according to the spacing parameters specified by you. One key feature of this re-gridding process is how closely the newly gridded boundaries match the original boundaries. The deviation between an original and new boundary is controlled by using the **CRITICAL** parameter, which is an indication of the critical feature size that should be maintained. During boundary re-gridding, the new boundaries are guaranteed to deviate from the original boundaries by no more than the specified **CRITICAL** value. Specifying a small, but non-zero, value of **CRITICAL** allows the new boundaries to closely track the original boundaries while at the same time allows redundant nodes that are very close to each other to be eliminated. By default, **CRITICAL** is set at 1/1000 of the minimum of the device width and the device height.

The grid spacing used during the re-gridding of the region boundaries is specified on a global basis using the **SPACING** parameter, which by default is set at 1/50 of the device width. This produces a uniform spacing except along highly curved boundaries which are refined to satisfy the **CRITICAL** spacing parameter. More detailed control of the boundary gridding is obtained with the **ABC.MESH** statement which allows the specification of non-uniform gridding on a local basis.

All boundaries that are re-gridded generate layers during the meshing of the region interior. Boundary re-gridding is performed for all interior boundaries, and by default, the top boundary of the device. In some applications, however, it is desirable to leave the top ungridded and propagate mesh layers from the interior up to the top. Specify this by using the **GRIDTOP** parameter. The sides and bottom of the device are never gridded.

Gridding Region Interiors

The grid spacing into the regions, i.e. the normal spacing, beginning at the region boundaries is specified for general semiconductors, conductors, and insulators by using the ratio parameters **N.SEMICO**, **N.CONDUC**, and **N.INSULA**. These parameters specify the aspect ratio (normal spacing/boundary lateral spacing) of the elements near the region boundaries.The actual normal spacing that is used is given by the value of the **SPACING** parameter times the aspect ratio. By default, the semiconductor aspect ratio is set at 0.1, the conductor ratio at 0.2, and the insulator ratio at 1.0. The growth of the normal spacing into the region as layers are created is controlled by the **NORMGROW**, or **RATIO**, parameter which is set by default to 1.2.

During meshing of the regions, the lateral spacing is controlled using the **LATERAL** and **ELIMINA** parameters. The **LATERAL** parameter acts like a target value. Whenever the lateral spacing during meshing falls below **LATERAL**, the lateral spacing is unrefined. Likewise, if the lateral spacing during meshing rises above **LATERAL**, then the lateral spacing is refined. Refinement or unrefinement is only performed, however, if non-obtuse elements can be produced. Setting a very large value of **LATERAL**, or using the **ELIMIN** parameter, causes the mesh

generator to unrefine the lateral spacing whenever non-obtuse elements are produced. By default, **LATERAL** is set at the maximum of the device width and device height.

Automatic Refinement Near Corners

To improve the mesh quality near corners where different regions meet, an automatic refinement algorithm has been developed to reconcile the different mesh spacings that often occur in corners. The **RFN.CRNR** causes the new ABC mesh generator to refine the lateral spacing along region boundaries to better match the normal spacing at the boundaries. For this to work effectively, however, it is necessary that the normal spacings in the different regions that meet at a corner be approximately the same.

Junctions

As in the original ABC mesh generator, use the **JUNC.ABC** parameter to conform grid lines to metallurgical junctions in addition to region boundaries. The position of the junction is determined from a logarithmic interpolation of the original doping profile.

Electrodes

The new ABC meshing generator correctly handles flat electrodes, including maintaining the original endpoints of the electrodes during the re-gridding of the region boundaries. Region electrodes can be voided using the **VOIDELEC** parameter which causes the region to be meshed without introducing any new nodes in the interior of the region. Both flat and volume electrodes from the original structure are maintained in the ABC-created mesh and do not need to be re-specified using the **ELECTRODE** statement.

Old Automatic Boundary Conforming Mesh Generator

The old ABC mesh generator uses the **MESH** statement to specify spacing parameters on a material-by-material basis. The **SPACING** parameter specifies a uniform lateral spacing along the region boundaries. The aspect ratio parameters such as **N.SILICO** can be used to specify the normal spacing at the region boundaries.

Lateral and Normal Mesh Spacings

As an example, suppose that a 0.25 micron MOSFET were meshed with the default set of parameter values. The structure width (the size of the simulation structure in the horizontal direction) for such a device is usually about one micron. By default, the spacing in silicon along the channel is 1/50 of the structure width (about 0.02 microns or 200 Å). The spacing in silicon in the normal direction to the oxide/silicon interface is a product of 200 Å and the mesh aspect ratio, defined for silicon by the parameter **N.SILICO**, which is 0.1 by default. For this example, 200 * 0.1 = 20 Å. Inside the silicon region from the oxide/silicon interface, each subsequent step size is larger than the previous one by the multiplication factor, defined by the parameter **RATIO**, which is 1.2 by default. In this example, the grid spacings in the channel in the vertical direction are 20 Å, 24 Å, 29 Å, 35Å, 41Å, 50 Å, ….

For regions with complex geometry, the grid layers are constructed along the entire boundary. For regions with straight left, right, and bottom sides, the grid layers are constructed from the top boundary only. Layer after layer, the boundary conforming elements are generated until the entire region volume is meshed.

Unstructured Delaunay Mesh

If it is not possible to mesh an entire region with a boundary conforming mesh, an unstructured Delaunay mesh generator is used for the residual subregion in the center of the region.

By default, the mesh aspect ratio (the ratio of the normal to lateral mesh spacings at the region boundary) is 0.1 for semiconductors and conductors, and 100 for insulators. This means that the elements along the boundaries of semiconductors and conductors are stretched, making the element aspect ratio about 10. The insulator aspect ratio of 100 usually requires that the unstructured Delaunay mesh generator be used for the entire region.

Selecting Aspect Ratios

The aspect ratios specified for the neighbor regions should be either similar to each other or very different, for example, 0.1 and 0.1, 0.2 and 0.2, or 0.1 and 100. When using the ABC algorithm, it is easier to generate non-obtuse elements when the aspect ratios in the neighbor regions are similar. If a large aspect ratio is specified for a region, the ABC algorithm is forced to mesh the entire region with the unstructured Delaunay mesh.

Automatic Failure Handling

If there are thin sections in a region, where too many nodes are placed in front of too few nodes, the Delaunay mesh generator may fail to mesh the region or a residual subregion. If such a failure occurs, the boundary conforming meshing algorithm automatically relaxes the meshing parameters to allow the Delaunay generator to mesh the region. The algorithm then attempts to generate the mesh until it is successful or reaches the allowed number of attempts. By default, the allowed number is four, which can be modified using the **ATTEMPTS** parameter.

Obtuse Elements

Usually, even along the curved boundaries, the boundary conforming meshing algorithm generates non-obtuse elements. Some obtuse elements may occur at sharp corners or in the center of the region. Statistics often show an increase in the number of obtuse elements when the ABC algorithm is used compared to the conventional mesh. However, most obtuse angles are of the order of only 90.00001 to 91 degrees and do not reduce the mesh quality.

Junction Conforming Mesh

If **JUNC.ABC** is specified the mesh generator will try to conform grid lines to metallurgical junctions that occur within the structure in addition to boundaries. The junction is determined by using linear interpolation from the doping at the nodes of the mesh that is read in. If this junction is very jagged, the ABC meshing algorithm may fail in its attempt to place a conforming grid around it. Boundary smoothing, which is invoked for values of **ANGLE** greater than zero, may help in this regard.

Terminal Handling

In the current version of the boundary conforming mesh generator, volume-less terminals are not supported. Only volume terminals are imported into simulation structures from the input file when the ABC parameter is specified. Electrodes can be also specified explicitly using the **ELECTRODE** statement.

Polysilicon is converted to electrode when reading an input file if **POLY.ELE** parameter is specified on the **MESH** statement. In such a case, the aspect ratio for polysilicon should be specified using the **N.ELECTR** parameter instead of the **N.POLYSI** parameter.

Large Geometry Features

If the simulation structure contains large geometry features, such as deep trenches, the simulation domain boundaries (the extreme left, right, and bottom edges of the structure) must be far enough from the feature to allow the ABC algorithm to mesh the entire device.

Quadtree Mesh Generation

If a structure is read in from a TSUPREM-4, TIF, or Medici file using the **IN.FILE** parameter, a new mesh for the structure can be generated using the quadtree option by specifying the **QUADTREE** parameter on the **MESH** statement.

If you specify the **QUADTREE** parameter, the previous mesh from the imported structure is discarded and a new mesh is created using the quadtree meshing algorithm.

Algorithm

The quadtree meshing algorithm generates high quality mesh elements that conform to the region boundaries and allow for anisotropic refinement. The quadtree algorithm recursively refines the mesh in order to accurately conform to the region boundaries and to meet user-specified mesh spacing requirements. During the initial quadtree mesh construction on the **MESH** statement, the mesh spacing may be controlled using three parameters: **MIN.THIC**, **MINDEL**, and **MAXDEL**. The **MIN.THIC** parameter is used to specify the minimum thickness of any region in the device, for example the gate oxide thickness in a MOSFET. This parameter is used to control the extent of the fine mesh around region boundaries. The **MINDEL** parameter is used to specify the smallest mesh spacing that should be generated in the new quadtree mesh. This mesh spacing is used primarily at the region boundaries. In order to obtain good boundary fidelity, **MINDEL** is restricted to be less than or equal to **MIN.THIC**. The **MAXDEL** parameter specifies the maximum allowed spacing in the new quadtree mesh and should be set fairly coarse. Once the initial mesh is generated, coarse sections of the mesh can be selectively refined using the **REGRID** statement.

Immediate Regridding

The quadtree mesh generated from the **MESH** statement will contain a fairly coarse mesh in the interior and a fine mesh at the region boundaries. To support the common case of doping regridding, it is possible to specify **REGRID** statements immediately after the **MESH** statement. This will cause mesh refinement to be done during the initial mesh construction saving time and improving the dop-

ing interpolation. See the statement, "REGRID" on page 3-94 for more details. If an immediate doping regrid is not performed, then a doping file should be created using the **PROFILE** statement in order to assure that the best possible doping profile is used during subsequent regrids.

Quadtree Files

The quadtree mesh used by Medici is generated and refined using an external mesh generator. This mesh generator makes use of two support files in addition to the primary mesh file used by Medici. Medici automatically manages these additional files, however, you should be aware of them in case the primary mesh file is moved or deleted. These additional files are a tree file describing the element hierarchy and a TDF file used to hold additional structure information. During a **SAVE** statement, these support files are renamed to be consistent with the primary mesh file. For example, if a Quadtree mesh is saved to a file called *device.tif*, then the support files will be named *device.tree* and *device.tdf*. These support files are needed if subsequent regrids are performed.

CAUTION

- **Do not delete or move the Quadtree support files if additional regrids are needed.**
- **The Quadtree mesh generator currently supports devices with less than 21 regions.**
- **The Quadtree mesh generator currently requires that all contiguous regions have unique names.**

Previously Generated Meshes

This section describes how to read in previously generated meshes. It contains the following:

- Reading a previously generated mesh
- Restrictions and Limitations
- Compatibility

Read a previously generated two- or three-dimensional mesh by using the **INFILE** parameter.

Restrictions and Limitations

When a previously generated mesh is read, no additional processing is allowed using the **X.MESH**, **Y.MESH**, **ELIMINATE**, **SPREAD**, and **BOUNDARY** statements.

You may alter a previously read mesh in the following ways:

- Additional electrodes may be added to the structure using **ELECTRODE** statements.

- Regions may be redefined or modified using **REGION** statements
- Impurity profiles may be added using **PROFILE** statements.
- Additional grid refinement may be done by using the **REGRID** statement.
- Region shape can be smoothed by using the **ADJUST** parameter.

TSUPREM-4

A structure generated by TSUPREM-4 for input into Medici can be read by specifying the parameter **TSUPREM4** and using **IN.FILE** to identify the file where the structure is stored.

In addition to reading the mesh which was created by TSUPREM-4, the net and total impurity concentration at each node is read. By default, the entire structure is read. It may be truncated by specifying one or more of the parameters **X.MIN**, **X.MAX**, **Y.MIN**, or **Y.MAX**.

An electrode may be added to the bottom of the structure (which is determined by the maximum y coordinate of the structure read in) by specifying **ELEC.BOT**.

Other Programs and TIF

A structure generated by other programs for input into Medici can be read by specifying the parameter **TIF** and using **IN.FILE** to identify the file where the structure is stored. In addition to reading the mesh, the doping at each node is read.

A TIF file generated by Medici with the solution information saved in the TIF file, serves as an initial guess for further simulations. An electrode may be added to the bottom of the structure (which is determined by the maximum y coordinate of the structure read in) by specifying **ELEC.BOT**. The saved solution may also be used for plotting.

A Quadtree mesh saved in a TIF file or Medici mesh file may be read back into Medici using the **MESH** statement. Anisotropic regrids may be performed using the **REGRID** statement.

Smoothing Region Shape

This section describes how to improve the way the region and electrode shape is handled by using the **ADJUST** parameter.

The **ADJUST** parameter may be used when the mesh is generated using the **X.MESH** and **Y.MESH** statements.

If an **ADJUST** parameter is provided with a **MESH** statement, then the triangle diagonals at the region interfaces are flipped whenever it smooths the interface shape. A piece-wise linear region shape interpolation is used instead of the default staircase interpolation.

This parameter is especially useful if the region/electrode shape is non-rectangular (polygonal or circular). It is not recommended if strictly rectangular region/elec-

trode shape is desired, because it smooths both "convex" and "concave" corners of the rectangular region/ electrode.

X.MESH

The **X.MESH** statement specifies the placement of nodes in the x direction.

X.MESH

```
{ LOCATION=<n> | ( {WIDTH=<n> | X.MAX=<n>} [X.MIN=<n>] ) }
[ {NODE=<n> | N.SPACES=<n>} ]
[ {SPACING=<n> | H2=<n>} ] [H1=<n>] [H3=<n>]
[RATIO=<n>] [MIN.SPAC=<n>] [SUMMARY]
```

Parameter	Type	Definition	Default	Units
LOCATION	number	The x location where a node is to be placed.	none	microns
WIDTH	number	The width of the grid section.	none	microns
X.MAX	number	The x location of the right edge of the grid section. **synonym: X.RIGHT**	none	microns
X.MIN	number	The x location of the left edge of the grid section. **synonym: X.LEFT**	The right edge of the previous grid section.	microns
NODE	number	The x node number at the location specified by **LOCATION** or at the right edge of a grid section. There can be at most 1000 nodes in the x direction. Nodes are assigned consecutively, beginning with the first and ending with the last. **synonym: N**	none	none
N.SPACES	number	The number of grid spaces in the grid section.	none	none
SPACING	number	The size of the x grid space at the location specified with **LOCATION** or at the right edge of the grid section.	none	microns
H2	number	The size of the x grid space at the right edge of the grid section.	none	microns
H1	number	The size of the x grid space at the left edge of the grid section.	none	microns
H3	number	The size of the minimum or maximum grid spacing in the interior of the grid section.	none	microns
RATIO	number	The ratio between the sizes of adjacent grid spaces in the grid section. **RATIO** should usually lie between 0.667 and 1.5.	1.0	none
MIN.SPAC	number	The size of the minimum allowed grid spacing in this grid section.	1.0e-4	microns
SUMMARY	logical	Specifies that summary information describing the grid section is sent to the standard output.	false	

Description

If an initial mesh is being generated, **X.MESH** and **Y.MESH** statements should immediately follow the **MESH** statement.

See Also... To further illustrate the **X.MESH** statement refer to input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Dimensions and Properties" on page 4-3.

Node Placement

Although the parameters on the **X.MESH** statement can be used in a variety of ways for specifying the horizontal placement of nodes, the three most common methods are described here. Any reasonable combination of the following three methods is also acceptable.

NODE, LOCATION, and RATIO

This is the only method that was available in early versions of TMA PISCES-2B for specifying the node placement. In this method, each **X.MESH** statement places the node at the location specified with **LOCATION**. Additional nodes are added automatically between the explicitly specified ones. The **RATIO** parameter can be used to vary the spacing between adjacent nodes.

The problem with this method is that it is difficult to specify a mesh with smooth grid spacing transitions between adjacent grid sections when the **RATIO** parameter is used.

Example: The following statements creates a 1-micron grid section with a uniform grid spacing of 0.1 microns:

```
X.MESH   NODE=1   LOCATION=0.0
X.MESH   NODE=11  LOCATION=1.0
```

LOCATION and SPACING

This is the method that is used for setting up the initial grid in TSUPREM-4. In this method, each **X.MESH** statement places a node at the location specified with **LOCATION**. The local grid spacing at that location is given by **SPACING**.

Additional nodes are added automatically so that the grid spacing varies geometrically between the **SPACING** values at the user-specified locations. This method guarantees a smooth grid spacing transition between adjacent grid sections.

Example: The following statements create a 1 micron grid section with a grid spacing of 0.1 microns at x=0 and a grid spacing of 0.01 at x=1:

```
X.MESH   LOCATION=0.0  SPACING=0.1
X.MESH   LOCATION=1.0  SPACING=0.01  SUMMARY
```

The grid summary generated by the **SUMMARY** parameter shows that 25 grid spaces are used and that the ratio between adjacent grid spaces is 0.9081.

WIDTH, N.SPACES, H1, H2, H3, and RATIO

This is the most versatile of the node placement methods. It allows the **WIDTH** of a grid section to be specified along with various combinations of the number of grid spaces (**N.SPACES**), grid spacings (**H1**, **H2**, and **H3**), and grid spacing ratio (**RATIO**).

The grid spacing sizes within a grid section varies in one of three manners, depending on which parameters are specified. In all cases, a single ratio is maintained between the sizes of adjacent grid spaces throughout the section.

- The specification of the **WIDTH** parameter and one parameter from the set **H1**, **H2**, and **N.SPACES** results in a uniform grid spacing within the grid section.
- The specification of the **WIDTH** parameter and any two parameters from the set **H1**, **H2**, **N.SPACES**, and **RATIO** results in grid spacing sizes which change monotonically between the ends of the grid section.
- The specification of the **H1**, **H2**, and **WIDTH** parameters and one parameter from the set **H3**, **N.SPACES**, and **RATIO** results in grid spacing sizes that increase (or decrease) from both ends of the grid section to a maximum (or minimum) in the interior of the section.

Examples

The following **X.MESH** statement sets up a uniform grid distribution with 10 spaces to span the 1 micron width of the grid section:

```
X.MESH   WIDTH=1  N=10
```

This **X.MESH** statement causes the grid spacing to be 0.01 microns at the left edge of the section and to increase by a ratio of 1.2 between adjacent spaces:

```
X.MESH  WIDTH=1  H1=0.01  RATIO=1.2  SUMMARY
```

The grid summary generated by the **SUMMARY** parameter shows that 17 spaces are used and that the grid spacing at the right end of the section is 0.17 microns.

The following **X.MESH** statement specifies that the spacing at both ends of the section is to be 0.01 microns and that the maximum spacing in the interior of the section is to be 0.1 microns:

```
X.MESH WIDTH=1 H1=0.01 H2=0.01 H3=0.1 SUMMARY
```

The grid summary generated by the **SUMMARY** parameter shows that 26 spaces are required and that the left edge of the 0.1 micron space occurs at 0.4355 microns from the left edge of the grid section.

Y.MESH

The **Y.MESH** statement specifies the placement of nodes in the y direction.

Y.MESH

```
{ LOCATION=<n> | ( {DEPTH=<n> | Y.MAX=<n>} [Y.MIN=<n>] ) }
[ {NODE=<n> | N.SPACES=<n>} ]
[ {SPACING=<n> | H2=<n>} ] [H1=<n>] [H3=<n>]
[RATIO=<n>] [MIN.SPAC=<n>] [SUMMARY]
```

Parameter	Type	Definition	Default	Units
LOCATION	number	The y location where a node is to be placed.	none	microns
DEPTH	number	The depth of the grid section. **synonym: WIDTH**	none	microns
Y.MAX	number	The y location of the bottom edge of the grid section. **synonym: Y.BOTTOM**	none	microns
Y.MIN	number	The y location of the top edge of the grid section. **synonym: Y.TOP**	The bottom edge of the previous grid section.	microns
NODE	number	The y node number at the location specified by **LOCATION** or at the bottom edge of a grid section. There can be at most 1000 nodes in the y direction. Nodes are assigned consecutively, beginning with the first and ending with the last. **synonym: N**	none	none
N.SPACES	number	The number of grid spaces in the grid section.	none	none
SPACING	number	The size of the y grid space at the location specified with **LOCATION** or at the bottom edge of the grid section.	none	microns
H2	number	The size of the y grid space at the bottom edge of the grid section.	none	microns
H1	number	The size of the y grid space at the top edge of the grid section.	none	microns
H3	number	The size of the minimum or maximum grid spacing in the interior of the grid section.	none	microns
RATIO	number	The ratio between the sizes of adjacent grid spaces in the grid section. **RATIO** should usually lie between 0.667 and 1.5.	1.0	none
MIN.SPAC	number	The size of the minimum allowed grid spacing in this grid section.	1.0e-4	microns
SUMMARY	logical	Specifies that summary information describing the grid section is sent to the standard output.	false	

Description

If an initial mesh is being generated, **X.MESH** and **Y.MESH** statements should immediately follow the **MESH** statement.

See Also... To further illustrate the **Y.MESH** statement, refer to the input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Dimensions and Properties" on page 4-3.

Node Placement

Although the parameters on the **Y.MESH** statement can be used in a variety of ways for specifying the vertical placement of nodes, the three most common methods are described here. Any reasonable combination of the following three methods is also acceptable.

NODE, LOCATION, and RATIO

This is the only method that was available in early versions of TMA PISCES-2B for specifying the node placement. In this method, each **Y.MESH** statement places the **NODE**th node at the location specified with **LOCATION**. Additional nodes are added automatically between the explicitly specified ones. The **RATIO** parameter can be used to vary the spacing between adjacent nodes.

The problem with this method is that it is difficult to specify a mesh with smooth grid spacing transitions between adjacent grid sections when the **RATIO** parameter is used.

Example: The following statements create a 1 micron grid section with a uniform grid spacing of 0.1 microns:

```
Y.MESH   NODE=1   LOCATION=0.0
Y.MESH   NODE=11  LOCATION=1.0
```

LOCATION and SPACING

This is the method that is used for setting up the initial grid in TSUPREM-4. In this method, each **Y.MESH** statement places a node at the location specified with **LOCATION**. The local grid spacing at that location is given by **SPACING**.

Additional nodes are added automatically so that the grid spacing varies geometrically between the **SPACING** values at the user-specified locations. This method guarantees a smooth grid spacing transition between adjacent grid sections.

Example: The following statements create a 1-micron grid section with a grid spacing of 0.1 microns at y=0 and a grid spacing of 0.01 at y=1:

```
Y.MESH   LOCATION=0.0  SPACING=0.1
Y.MESH   LOCATION=1.0  SPACING=0.01  SUMMARY
```

The grid summary generated by the **SUMMARY** parameter shows that 25 grid spaces are used and that the ratio between adjacent grid spaces is 0.9081.

DEPTH, N.SPACES, H1, H2, H3, and RATIO

This is the most versatile of the methods allowing the **DEPTH** of a grid section to be specified along with various combinations of the number of grid spaces (**N.SPACES**), grid spacings (**H1**, **H2**, and **H3**), and grid spacing ratio (**RATIO**).

The grid spacing sizes within a grid section varies in one of three ways, depending on which parameters are specified. In all cases, a single ratio is maintained between the sizes of adjacent grid spaces throughout the section.

- The specification of the **DEPTH** parameter and one parameter from the set **H1**, **H2**, and **N.SPACES** results in a uniform grid spacing within the grid section.
- The specification of the **DEPTH** parameter and any two parameters from the set **H1**, **H2**, **N.SPACES**, and **RATIO** results in grid spacing sizes which change monotonically between the ends of the grid section.
- The specification of the **H1**, **H2**, and **DEPTH** parameters and one parameter from the set **H3**, **N.SPACES**, and **RATIO** results in grid spacing sizes that increase (or decrease) from both ends of the grid section to a maximum (or minimum) in the interior of the section.

Examples

The following **Y.MESH** statement sets up a uniform grid distribution with 10 spaces to span the 1 micron depth of the grid section:

```
Y.MESH   DEPTH=1  N=10
```

This **Y.MESH** statement causes the grid spacing to be 0.01 microns at the top edge of the section and to increase by a ratio of 1.2 between adjacent spaces:

```
Y.MESH   DEPTH=1  H1=0.01  RATIO=1.2  SUMMARY
```

The grid summary generated by the **SUMMARY** parameter shows that 17 spaces are used and that the grid spacing at the bottom end of the section is 0.17 microns.

The following **Y.MESH** statement specifies that the spacing at both ends of the section is to be 0.01 microns and that the maximum spacing in the interior of the section is to be 0.1 microns:

```
Y.MESH DEPTH=1 H1=0.01 H2=0.01 H3=0.1 SUMMARY
```

The grid summary generated by the **SUMMARY** parameter shows that 26 spaces are required and that the top edge of the 0.1 micron space occurs at 0.4355 microns from the top edge of the grid section.

ABC.MESH

The **ABC.MESH** statement specifies local spacing parameters for the new Automatic Boundary Conforming (ABC) mesh generator.

```
ABC.MESH

  Local Spacing Control
   { ( [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
       {  (ALIGN REGION1=<c> REGION2=<c> REGION3=<c> REGION4=<c>)
        | ( [ {  REGION=<c> | (BOUNDARY REGION1=<c> REGION2=<c>)
                | {  SILICON | GAAS | POLYSILI | GERMANIU | SIC | SIGE
                   | ALGAAS | A-SILICO | DIAMOND | HGCDTE | INAS | INGAAS
                   | INP | S.OXIDE | ZNSE | ZNTE | ALINAS | GAASP | INGAP
                   | INASP | OXIDE | NITRIDE | SAPPHIRE | OXYNITRI
                  }
              }
            ]
            [CRITICAL=<n>] [H1=<n>] [H2=<n>]
            [{GRDRIGHT | GRDLEFT | GRDUP | GRDDOWN | GRDCNTR}]
            [NEIGHBOR=<c>] [{NORMAL=<n> | (NORMAL1=<n> NORMAL2=<n>)}]
            [NORMGROW=<n>] [LATERAL=<n>]
          )
       }
     )
  Automatic MOSFET Meshing
   | ( MOSFET
       [N.CHANN=<n>] [N.JUNC=<n>] [N.GATEOX=<n>] [N.POLY=<n>]
       [MAX.CHAN]=<n> [MAX.SD=<n>] [SI.NFACT=<n>] [GATE.NFA=<n>]
       [RATIO=<n>]
     )
   }
```

Parameter	Type	Definition	Default	Units
X.MIN	number	The minimum x-coordinate of the spacing box. The spacing parameters are only applicable to areas of the device with x-coordinates greater than or equal to **X.MIN**.	The minimum x location in the structure	microns
X.MAX	number	The maximum x-coordinate of the spacing box. The spacing parameters are only applicable to areas of the device with x-coordinates less than or equal to **X.MAX**.	The maximum x location in the structure	microns
Y.MIN	number	The minimum y-coordinate of the spacing box. The spacing parameters are only applicable to areas of the device with y-coordinates greater than or equal to **Y.MIN**.	The minimum y location in the structure	microns
Y.MAX	number	The maximum y-coordinate of the spacing box. The spacing parameters are only applicable to areas of the device with y-coordinates less than or equal to **Y.MAX**.	The maximum y location in the structure	microns
REGION	char	The name of the region to which the spacing parameters should be applied.		

Parameter	Type	Definition	Default	Units
BOUNDARY	logical	Specifies that the spacing parameters should be applied to the boundary between **REGION1** and **REGION2**.	false	
ALIGN	logical	Specifies that the nodes on the boundary between **REGION1** and **REGION2** should be projected onto the boundary between **REGION3** and **REGION4** to facilitate node alignment.	false	
REGION1	char	In conjunction with **BOUNDARY**, the name of one of the regions on the boundary to which the spacing parameters should be applied. In conjunction with **ALIGN**, the name of one of the regions on the source boundary for node alignment.		
REGION2	char	In conjunction with **BOUNDARY**, the name of one of the regions on the boundary to which the spacing parameters should be applied. In conjunction with **ALIGN**, the name of one of the regions on the source boundary for node alignment.		
REGION3	char	In conjunction with **ALIGN**, the name of one of the regions on the destination boundary for node alignment.		
REGION4	char	In conjunction with **ALIGN**, the name of one of the regions on the destination boundary for node alignment.		
SILICON	logical	Specifies that the spacing parameters should be applied to all regions of material silicon.	false	
GAAS	logical	Specifies that the spacing parameters should be applied to all regions of material GaAs.	false	
POLYSILI	logical	Specifies that the spacing parameters should be applied to all regions of material polysilicon.	false	
GERMANIU	logical	Specifies that the spacing parameters should be applied to all regions of material germanium.	false	
SIC	logical	Specifies that the spacing parameters should be applied to all regions of material SiC.	false	
SIGE	logical	Specifies that the spacing parameters should be applied to all regions of material SiGe.	false	
ALGAAS	logical	Specifies that the spacing parameters should be applied to all regions of material AlGaAs.	false	
A-SILICO	logical	Specifies that the spacing parameters should be applied to all regions of material amorphous silicon.	false	
DIAMOND	logical	Specifies that the spacing parameters should be applied to all regions of material diamond.	false	
HGCDTE	logical	Specifies that the spacing parameters should be applied to all regions of material HgCdTe.	false	
INAS	logical	Specifies that the spacing parameters should be applied to all regions of material InAs.	false	
INGAAS	logical	Specifies that the spacing parameters should be applied to all regions of material InGaAs.	false	

Parameter	Type	Definition	Default	Units
INP	logical	Specifies that the spacing parameters should be applied to all regions of material InP.	false	
S.OXIDE	logical	Specifies that the spacing parameters should be applied to all regions of material oxide when it is simulated as a wide band gap semiconductor.	false	
ZNSE	logical	Specifies that the spacing parameters should be applied to all regions of material ZnSe.	false	
ZNTE	logical	Specifies that the spacing parameters should be applied to all regions of material amorphous ZnTe.	false	
ALINAS	logical	Specifies that the spacing parameters should be applied to all regions of material AlInAs.	false	
GAASP	logical	Specifies that the spacing parameters should be applied to all regions of material GaAsP.	false	
INGAP	logical	Specifies that the spacing parameters should be applied to all regions of material InGaP.	false	
INASP	logical	Specifies that the spacing parameters should be applied to all regions of material InAsP.	false	
OXIDE	logical	Specifies that the spacing parameters should be applied to all regions of material oxide.	false	
NITRIDE	logical	Specifies that the spacing parameters should be applied to all regions of material nitride.	false	
SAPPHIRE	logical	Specifies that the spacing parameters should be applied to all regions of material sapphire.	false	
OXYNITRIDE	logical	Specifies that the spacing parameters should be applied to all regions of material oxynitride.	false	
CRITICAL	number	Specifies the critical feature size that should be maintained during re-gridding of the region boundaries. The new region boundaries are guaranteed to deviate from the original boundaries by less than **CRITICAL**.	**CRITICAL** value from the **MESH** statement.	microns
H1	number	Desired initial grid spacing on the region boundary.	**SPACING** value from the **MESH** statement.	microns
H2	number	Desired final grid spacing on the region boundary.	**SPACING** value from the **MESH** statement.	microns
GRDRIGHT	logical	Specifies that gridding along region boundaries should proceed from left to right.	false	
GRDLEFT	logical	Specifies that gridding along region boundaries should proceed from right to left.	false	
GRDUP	logical	Specifies that gridding along region boundaries should proceed from down to up.	false	
GRDDOWN	logical	Specifies that gridding along region boundaries should proceed from up to down.	false	
GRDCNTR	logical	Specifies that gridding along region boundaries should proceed from the center of the boundary out toward the ends.	true	

Parameter	Type	Definition	Default	Units
NEIGHBOR	char	In conjunction with **REGION**, specifies a particular boundary of **REGION** by giving the name of the neighboring region on the opposite side of the boundary. Used to specify the following parameters at a particular boundary of a particular region: **NORMAL**, **NORMAL1**, **NORMAL2**, **NORMGROW**.		
NORMAL	number	Specifies a uniform normal spacing along the region boundary.	Determined from the **SPACING** and **N.SEMICO**, **N.CONDUC,** and **N.INSULA** parameters on the **MESH** statement.	microns
NORMAL1	number	Specifies the normal spacing at the beginning of the region boundary.		microns
NORMAL2	number	Specifies the normal spacing at the end of the region boundary.		microns
NORMGROW	number	Specifies the growth factor of the normal spacing into the interior of the region.	**NORMGROW** value from the **MESH** statement.	
LATERAL	number	Specifies a target value for the lateral spacing during gridding of the interior of the region.	**LATERAL** value from the **MESH** statement.	microns
Automatic MOSFET Meshing				
MOSFET	logical	Specifies that automatic meshing for symmetric MOSFET's should be used.	false	
N.CHANN	number	Specifies the normal spacing in the channel at the silicon-oxide interface.	4.0e-4	microns
N.JUNC	number	Specifies the maximum normal spacing at metallurgical junctions.	40.0e-4	microns
N.GATEOX	number	Specifies the normal spacing in the oxide below the gate.	tox/2	microns
N.POLY	number	Specifies the normal spacing at the bottom of the polysilicon gate.	4.0e-4	microns
MAX.CHAN	number	Specifies the maximum lateral spacing along the top of the channel.	0.5	microns
MAX.SD	number	Specifies the maximum lateral spacing along the boundaries of the source and drain regions.	(SD junction length)/ 10	microns
SI.NFACT	number	Specifies the growth factor along the boundaries of the silicon regions for the normal spacing.	20.0	1/ microns
GATE.NFA	number	Specifies the growth factor along the gate boundaries for the normal spacing.	30.0	1/ microns
RATIO	number	Specifies the gridding ratio for the lateral spacing. Also used to set the **NORMGROW**.	1.35	

Description

ABC.MESH statements should immediately follow a **MESH** statement when using the new ABC mesh generator. The **ABC.MESH** statement is used to create a "Spacing Box" in which spacing parameters can be specified on a local basis.

Spacing Boxes

Spacing boxes allow local control of the meshing parameters for both region boundaries and region interiors. Spacing boxes work by following an override convention for parameter specification that is similar to other Medici statements. The value of a parameter is used until overridden by a later specification of the parameter in an **ABC.MESH** statement. The spacing boxes specified by the **ABC.MESH** statement are thought of as a stacked layer of boxes with the first **ABC.MESH** statement specifying the box on the bottom. Subsequent **ABC.MESH** statements place boxes on top of each other with the last **ABC.MESH** statement specifying the top-most box. The value of a parameter is found by searching the list of spacing boxes from the top to the bottom of the stack. The parameter value is determined by the first spacing box that contains a valid specification of the parameter. In the event no spacing box provides a valid specification for a parameter, either the global value specified in the **MESH** statement is used, or the default value listed in the parameter table is used.

The extent of a spacing box is constrained by using the **X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX** parameters to specify the horizontal and vertical limits of the spacing box. The parameters specified in such a box will only be valid within the specified limits. Further restriction of the parameter specification is made by specifying a particular region, region boundary, or material to which the parameters are applied by using the **REGION**, **BOUNDARY**, **REGION1**, **REGION2**, and logical material parameters. This restriction is used in conjunction with the bounding box and override convention to determine the value of a parameter at a particular point in the mesh. The stacking of spacing boxes by sequential **ABC.MESH** statements in conjunction with specifying the extent of the boxes and specifying the region(s) or region boundary(s) to which the spacing boxes apply are used to buildup a detailed mesh specification in a piecewise fashion. Examples of this are provided below.

Gridding Region Boundaries

The ABC algorithm breaks the meshing of a device into two distinct stages: gridding of the boundaries between regions and then the meshing of the interior of regions. The new version of the ABC mesh generator gives greater control over the accuracy of the newly created region boundaries and the spacing of grid points along the boundaries. The parameters involved in gridding the region boundaries are **CRITICAL**, **H1**, **H2**, **GRDRIGHT**, **GRDLEFT**, **GRDUP**, **GRDDOWN**, and **GRDCNTR**. These parameters can be specified for all boundaries of a particular region by using the **REGION** parameter, for all boundaries of a certain material type by

using the logical material parameters, or for a particular boundary between two regions by specifying **BOUNDARY** in conjunction with **REGION1** and **REGION2**.

One key feature of this re-gridding process is how closely the newly gridded boundaries match the original boundaries. Control the deviation between an original and new boundary by using the **CRITICAL** parameter, which is an indication of the critical feature size that should always be maintained. During boundary re-gridding, the new boundaries are guaranteed to deviate from the original boundaries by no more than the specified **CRITICAL** value. Specifying a small, but non- zero, value of **CRITICAL** allows the new boundaries to closely track the original boundaries while at the same time allows redundant nodes that are very close to each other to be eliminated.

Region boundaries can now be gridded with non-uniform spacing using the spacing parameters called **H1** and **H2**, similar to the **H1** and **H2** parameters in the **X.MESH** and **Y.MESH** statements. Use **H1** to set the initial spacing along the boundary, and use **H2** to set the final spacing. The intermediate grid spacings are calculated using a constant ratio between adjacent grid spaces, which is similar to the operation of the **X.MESH** and **Y.MESH** statements. Except in special circumstances, the initial and final spacings are guaranteed to be **H1** and **H2**, respectively.

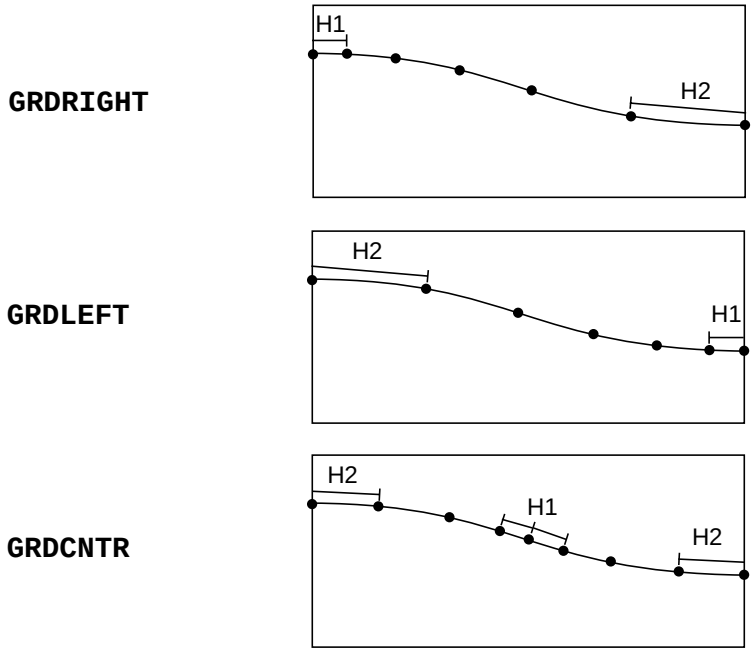


Figure 3-1 Examples of different gridding directions

As shown in Figure 3-1, the direction of the gridding along the boundary is controlled by setting one of direction parameters, **GRDRIGHT**, **GRDLEFT**, **GRDUP**, **GRDDOWN**, and **GRDCNTR**. For example, using **GRDRIGHT** along a boundary causes **H1** to be used as the left-most spacing, and **H2** to be used as the right-most spacing. Boundaries can be gridded symmetrically using **GRDCNTR** which causes **H1** to be used as the grid spacing in the center of the region boundary and **H2** as

the grid spacing at both ends of the boundary. During gridding, the actual grid direction that is used is determined by finding the specified grid direction at the center of the boundary.

Alignment

To obtain a good quality mesh in a region that is very thin, it is often necessary to align the nodes on the boundaries. As shown in Figure 3-2, a common example of this occurs in the gate oxide of a MOSFET. Misaligned nodes on the silicon-oxide and gate-oxide interfaces can result in mesh elements of poor quality. The **ALIGN** parameter can be used to align the nodes on two different boundaries.

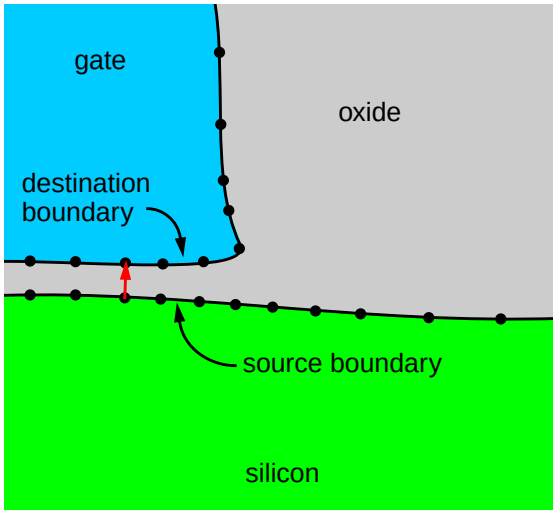


Figure 3-2 Example of node alignment across a thin gate oxide

With **ALIGN** specified, the **REGION1** and **REGION2** parameters are used to specify the source boundary that will provide the nodes for alignment. The **REGION3** and **REGION4** parameters are used to specify the destination boundary to which these nodes are projected. As shown in Figure 3-2, node alignment is achieved by projecting a node on the source boundary along a line perpendicular to the source boundary and finding the intersection with the destination boundary. Using the region names shown in Figure 3-2, this node alignment would be specified using the following **ABC.MESH** statement:

```
ABC.MESH ALIGN REGION1=silicon REGION2=oxide
+        REGION3=gate REGION4=oxide
```

The bounding box parameters can be used to restrict the area over which alignment occurs.

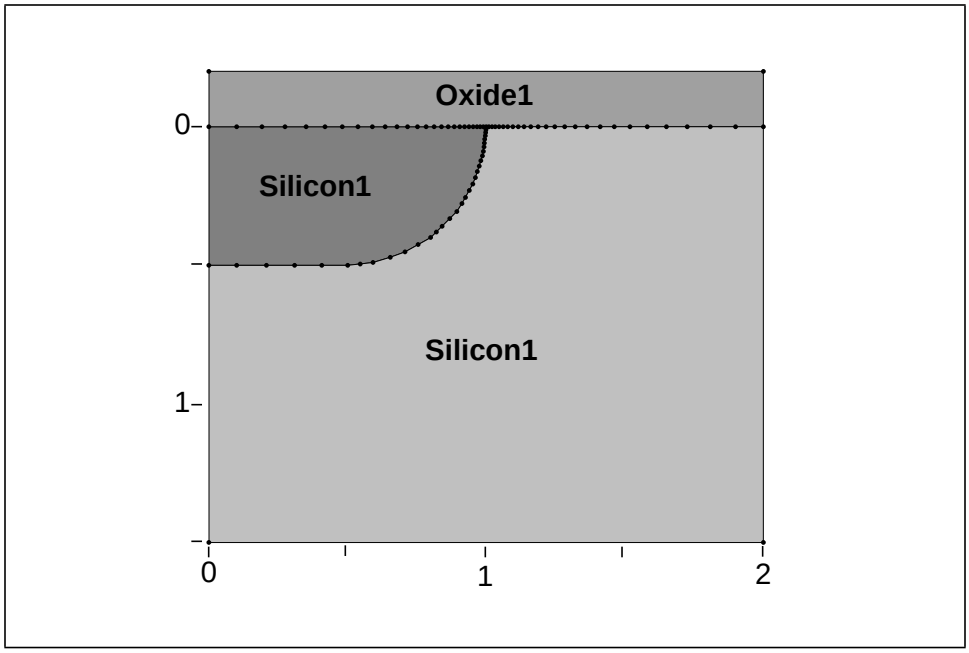


Figure 3-3 Gridding region boundaries using the new ABC mesh generator

Example

Consider the device shown in Figure 3-3. The original structure consisted of a single silicon region named Silicon1 and an oxide region named Oxide1. The figure shows the resulting junction and region boundaries after the following set of statements is processed. The single silicon region is broken into two regions with the same name as a result of the **JUNC.ABC** parameter.

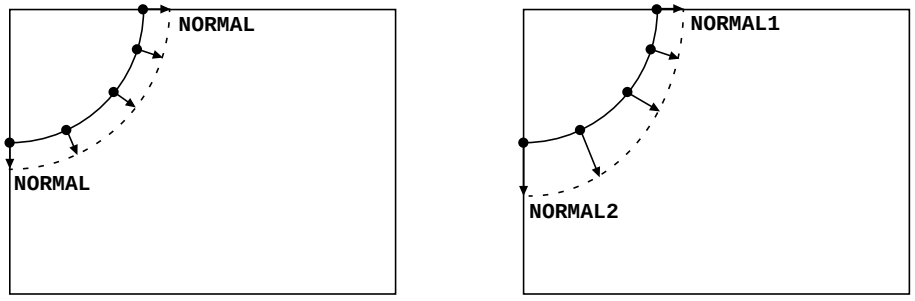
```
1... MESH      IN.FILE=device.tif TIF ABC JUNC.ABC
+              ^GRIDTOP ^RFN.CRNR
2... ABC.MESH BOUNDARY REGION1=Silicon1 REGION2=Silicon1
               GRDLEFT H1=0.01 H2=0.1
3... ABC.MESH BOUNDARY REGION1=Silicon1 REGION2=Oxide1
+              X.MIN=1.0 GRDRIGHT H1=0.01 H2=0.1
4... ABC.MESH BOUNDARY REGION1=Silicon1 REGION2=Oxide1
+              X.MAX=1.0 GRDLEFT H1=0.01 H2=0.1
```

The **MESH** statement in line 1 initializes the new ABC mesh generator and uses **JUNC.ABC** to cause the mesh to conform to junctions in addition to region boundaries. In this example, both the gridding of the top of the structure and automatic corner refinement are turned off. The automatic corner refinement is turned off so that complete control over the spacing is obtained. Line 2 uses a spacing box to specify the parameters for gridding the junction boundary. The junction is identified by specifying the two identically named silicon regions on either side of the junction. The grid is produced from right to left with an initial spacing 0.01μm and a final spacing 0.1μm. The spacing parameters for the boundary between the two silicon regions and the oxide region are specified using two **ABC.MESH** statements. Line 3 specifies the spacing parameters for the right part of the boundary by using **X.MIN** to restrict the parameters to the right side of the junction. Likewise, Line 4 specifies the spacing parameters for the left part of the boundary.

Gridding Region Interiors

Control region gridding by specifying the desired normal spacing away from the boundaries and target values of lateral spacing within the interior of the region. The parameters used for gridding the interior of a region are **NEIGHBOR**, **NORMAL**, **NORMAL1**, **NORMAL2**, **NORMGROW**, and **LATERAL**. Specify these parameters in conjunction with the **REGION** parameter or one of the logical material parameters. If the **REGION** parameter is used, the spacing parameters are used for all regions with the given name. If a logical material parameter such as **SILICO** is used, then the spacing parameters are used for all regions of that material type.

Currently, Medici only allows the normal spacing to be specified at the start of the gridding process along the region boundaries. Specify normal spacing parameters for a particular boundary of a region by using the **REGION** parameter to specify the desired region and the **NEIGHBOR** parameter to specify the region on the other side of the boundary. Specify uniform normal spacing along a boundary by using the **NORMAL** parameter.



Uniform normal spacing Non-uniform normal spacing

Figure 3-4 Specification of non-uniform normal spacing

As shown in Figure 3-4, specify non-uniform spacing by using the **NORMAL1** and **NORMAL2** parameters in conjunction with one of the gridding directions. Similar to the specification of **H1** and **H2** for boundary gridding, **NORMAL1** specifies the initial normal spacing along the boundary, and **NORMAL2** specifies the final normal spacing along the boundary. The normal spacing of intermediate points along the boundary is calculated using a constant ratio between adjacent points.

The growth of the normal spacing into the region is controlled by the **NORMGROW** parameter. The normal spacing used in each subsequent layer is larger than the previous one by the multiplication factor given by **NORMGROW**. Control the lateral spacing within region by using the **LATERAL** parameter. This parameter acts like a target value. If the lateral spacing produced during gridding is less than twice **LATERAL**, then the lateral spacing is unrefined in order to bring the spacing up to the desired value. Likewise, if the lateral spacing produced during gridding is more than twice **LATERAL**, then the lateral spacing is refined in order to bring the spacing down to the desired value. Refinement and unrefinement are only performed, however, if a non-obtuse adjustment triangle can be produced. The activity of the **LATERAL** parameter is deactivated by specifying a negative value. A

common use of **LATERAL** is to coarsen the mesh away from the boundaries by specifying a very large value. This causes the lateral spacing to be unrefined whenever non-obtuse elements are produced.

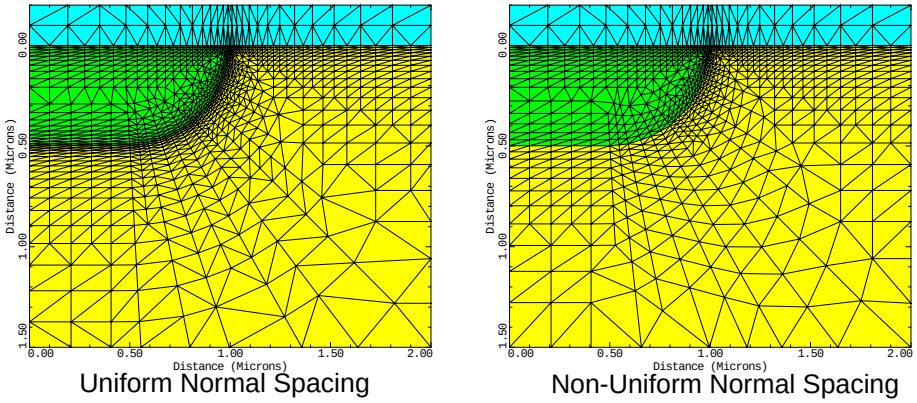


Figure 3-5 Gridding region interiors using the new ABC mesh generator

Example

Continuing with the example started in Figure 3-3, Figure 3-5 shows the result of using the following additional **ABC.MESH** statements to specify parameters for gridding the interior of the regions.

```
5... ABC.MESH  OXIDE NORMAL=0.1
6... ABC.MESH  REGION=Silicon1 NORMAL=0.01
7... ABC.MESH  REGION=Silicon1 NEIGHBOR=Silicon1 X.MIN=0.5
+              NORMAL1=0.01 NORMAL2=0.03 GRDLEFT
8... ABC.MESH  REGION=Silicon1 NEIGHBOR=Silicon1 X.MAX=0.5
+              NORMAL1=0.03 NORMAL2=0.03 GRDLEFT
```

Line 5 uses an **ABC.MESH** statement to specify that a uniform normal spacing of 0.1µm should be used in all regions of material type oxide; i.e. the region Oxide1. Likewise, line 6 specifies that all regions with name Silicon1 should be gridded with a uniform normal spacing at the boundary of 0.01µm. These two statements produce the uniform normal spacing mesh shown on the left of Figure 3-5. Line 7 and 8 are used to produced the non-uniform normal spacing mesh shown on the right of Figure 3-5. These two statements override the normal spacing specified in line 6 for points on the junction. Line 7 specifies a non-uniform normal spacing along the junction with an initial spacing of 0.01µm and a final spacing of 0.03µm at the end of the bounding box at x=0.5µm. The normal spacing increases from right to left along the junction since **GRDLEFT** is specified. The specification in line 6 is overridden by line 8 for points to the left of 0.5µm, producing a uniform normal spacing of 0.03µm along the bottom of the junction.

Automatic MOSFET Meshing

In order to facilitate the use of the new ABC mesher for one of the most popular applications, a symmetrical MOSFET, an automatic meshing algorithm has been developed. This algorithm, activated with the **MOSFET** parameter on the

ABC.MESH statement, identifies the key active regions of a symmetrical MOSFET. The mesh spacing in these key regions can be easily controlled through a small set of parameters. In addition, you can also specify how quickly the mesh should coarsen away from the key device areas in order to reduce the node count. Additional **ABC.MESH** statements can be used in conjunction with automatic meshing to further control the mesh. Note that the **JUNC.ABC** parameter must be specified on the **MESH** statement to enable meshing around junctions. If the automatic meshing algorithm is unable to identify the primary regions and boundaries of the device, meshing will continue without the automatic analysis.

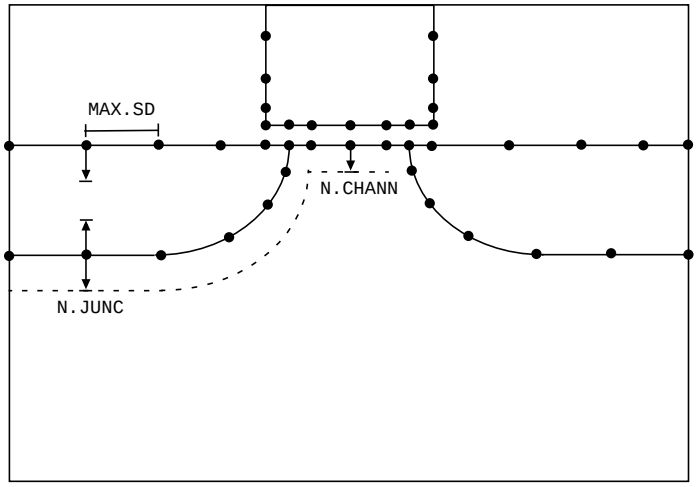


Figure 3-6 Illustration of the gridding scheme used by the automatic meshing algorithm for symmetric MOSFETs

The normal spacing parameters used to parameterize the mesh in the silicon regions are shown in Figure 3-6. A uniform normal spacing of **N.CHANN** is used at the top of the channel, increasing to a maximum of **N.JUNC** along the boundaries of the source and drain. The rate at which the normal spacing increases along these boundaries is governed by **SI.NFACT**. The lateral spacing at the source and drain junction points is set to **N.CHANN** to obtain good mesh quality there. The lateral spacing is then increased away from the junction points: outward toward the sides and inward above the channel. The rate at which the lateral spacing increases is governed by **RATIO**. The maximum allowed lateral spacing on the boundaries of the source and drain regions is set by **MAX.SD**, while the maximum allowed lateral spacing above the channel is set by **MAX.CHANN**.

The gridding scheme near the bottom of the gate on the drain side is shown in Figure 3-7. The nodes on the bottom of the gate are aligned to the nodes on the top of the silicon regions. The lateral spacing along the sides of the gate is coarsened away from the bottom of the gate. The normal spacing is set to a uniform value of **N.POLY** along the bottom of the gate, and is then increased along the sides of the gate. The rate at which the normal spacing increases is set by **GATE.NFACT**. A uniform normal spacing of **N.GATEOX** is used in the oxide under the gate.

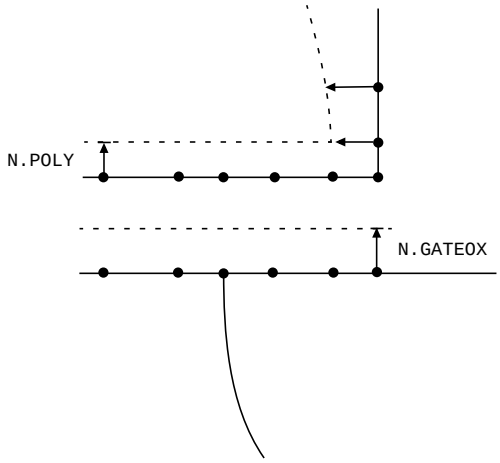


Figure 3-7 A blow-up of the automatic MOSFET gridding scheme around the drain junction point

The listing below shows an example of using the automatic meshing capability. A channel spacing of 6Å is specified along with a 10Å spacing at the bottom of the polysilicon region. The rate at which the normal spacing along the sides of the gate increases is set to $40\mu m^{-1}$ to decrease the node count. An example of a MOSFET gridded using this specification is shown in Figure 3-8.

```
ABC.MESH    MOSFET N.CHANN=6E-4
+           N.POLY=10E-4 GATE.NFA=40
```

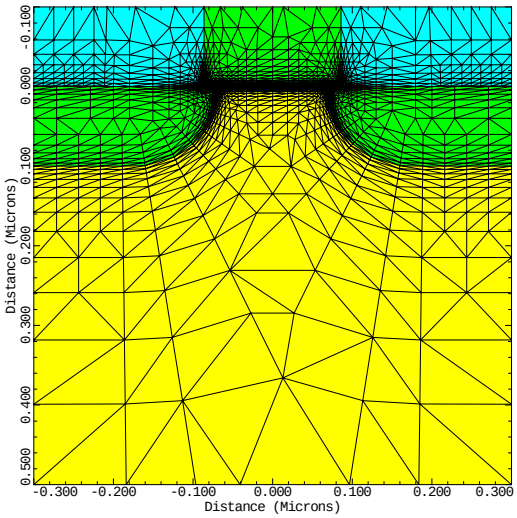


Figure 3-8 An example of a symmetric MOSFET automatically meshed

ELIMINATE

The **ELIMINATE** statement eliminates mesh points along planes in a rectangular grid over a specified volume.

```
ELIMINATE

{ROWS | COLUMNS}
[ {X.MIN=<n> | IX.MIN=<n>} ] [ {X.MAX=<n> | IX.MAX=<n>} ]
[ {Y.MIN=<n> | IY.MIN=<n>} ] [ {Y.MAX=<n> | IY.MAX=<n>} ]
```

Parameter	Type	Definition	Default	Units
ROWS	logical	Specifies that horizontal lines of nodes are eliminated. **synonym: X.DIREC**	false	
COLUMNS	logical	Specifies that vertical lines of nodes are eliminated. **synonym: Y.DIREC**	false	
X.MIN	number	The minimum x location of the rectangular volume in which nodes are eliminated.	The minimum x location for the device structure.	microns
IX.MIN	number	The minimum x node index of the rectangular volume in which nodes are eliminated. **synonym: IX.LOW**	1	none
X.MAX	number	The maximum x location of the rectangular volume in which nodes are eliminated.	The maximum x location for the device structure.	microns
IX.MAX	number	The maximum x node index of the rectangular volume in which nodes are eliminated. **synonym: IX.HIGH**	The maximum x node index for the device structure.	none
Y.MIN	number	The minimum y location of the rectangular volume in which nodes are eliminated.	The minimum y location for the device structure.	microns
IY.MIN	number	The minimum y node index of the rectangular volume in which nodes are eliminated. **synonym: IY.LOW**	1	none
Y.MAX	number	The maximum y location of the rectangular volume in which nodes are eliminated.	The maximum y location for the device structure.	microns
IY.MAX	number	The maximum y node index of the rectangular volume in which nodes are eliminated. **synonym: IY.HIGH**	The maximum y node index for the device structure.	none

Description

The **ELIMINATE** statement is useful for eliminating nodes in regions of the device structure where the grid is more dense than necessary.

Points along every second line in the chosen direction within the chosen range are removed. Successive eliminations of the same range remove points along every fourth line, eighth line, and so on.

Multiple eliminations over the same region using both **ROWS** and **COLUMNS** are allowed.

See Also...

To further illustrate the **ELIMINATE** statement refer to input file *mdex1* N-Channel MOSFET Examples, Chapter 4, "Triangular Grid" on page 4-4.

Restrictions

Do not use the **ELIMINATE** statement with the following:

- A distorted mesh, and therefore must precede any **SPREAD** statements that may be present
- A mesh which has undergone a **REGRID** operation
- A mesh read in from TSUPREM-4

SPREAD

The **SPREAD** statement provides a way to adjust the y position of nodes along grid lines parallel to the x-axis in a rectangular mesh to follow surface and junction contours.

SPREAD

```
{LEFT | RIGHT} WIDTH=<n> UPPER=<n> LOWER=<n> [ENCROACH=<n>]
{Y.LOWER=<n> | FIX.LOWE | ( THICKNES=<n> [VOL.RAT=<n>] )}
[GRADING=<n>] [ MIDDLE=<n> Y.MIDDLE=<n> [GR1=<n>] [GR2=<n>] ]
```

Parameter	Type	Definition	Default	Units
LEFT	logical	Specifies that the left side of the grid is distorted.	false	
RIGHT	logical	Specifies that the right side of the grid is distorted.	false	
WIDTH	number	The width of the distorted region measured from the **LEFT** or **RIGHT** edge of the structure. If **LEFT** is specified, the middle of the transition region between distorted and undistorted grid regions lies at the horizontal coordinate minimum(x)+**WIDTH**. If **RIGHT** is specified, the middle of the transition region between distorted and undistorted grid regions lies at the horizontal coordinate maximum(x)-**WIDTH**.	none	microns
UPPER	number	The index of the upper y-grid line of the distorted region.	none	none
LOWER	number	The index of the lower y-grid line of the distorted region.	none	none
ENCROACH	number	The factor which defines the abruptness of the transition between distorted and undistorted grid. The transition region becomes more abrupt with smaller **ENCROACH** factors. Depending on the characteristics of the undistorted grid, very bad triangles (long, thin, and obtuse) may result if **ENCROACH** is set too low. The minimum allowed value is 0.1.	1.0	none
Y.LOWER	number	The vertical location in the distorted region where the line specified by **LOWER** is moved. The grid line specified by **UPPER** does not move if this parameter is specified.	none	microns
FIX.LOWE	logical	Specifies that the line specified by **LOWER** is fixed during the spread.	false	
THICKNES	number	The thickness of the distorted region. Specifying **THICKNES** usually causes the positions of both the **UPPER** and **LOWER** grid lines to move.	none	microns
VOL.RAT	number	The ratio of the displacement of the lower grid line to the net change in thickness. If **VOL.RAT** is 0, the location of the lower grid line does not move. If **VOL.RAT** is 1, the upper grid line does not move.	0.44	none

Parameter	Type	Definition	Default	Units
GRADING	number	The vertical grid spacing ratio in the distorted region between the y-grid lines specified with **UPPER** and **LOWER** (if **MIDDLE** is not specified). The spacing grows or shrinks by **GRADING** in each interval between lines. **GRADING** should usually lie between 0.667 and 1.5.	1.0	none
MIDDLE	number	The index of a y-grid line which lies between the y-grid lines specified with **UPPER** and **LOWER**.	none	none
Y.MIDDLE	number	The vertical location in the distorted region where the line specified by **MIDDLE** is moved.	none	microns
GR1	number	The vertical grid spacing ratio in the distorted region between the y-grid lines specified with **UPPER** and **MIDDLE**. The spacing grows or shrinks by **GR1** in each interval between lines. **GR1** should usually lie between 0.667 and 1.5.	1.0	none
GR2	number	The vertical grid spacing ratio in the distorted region between the y-grid lines specified with **MIDDLE** and **LOWER**. The spacing grows or shrinks by **GR2** in each interval between lines. **GR2** should usually lie between 0.667 and 1.5.	1.0	none

Description

SPREAD is useful in reducing the amount of grid nodes for some specific problems, most notably MOSFET's. With the **SPREAD** statement, it is possible to redistribute existing grid lines to increase the node density in critical areas, for example at junctions. **SPREAD** can also be used to create a nonplanar surface topography in the device structure.

See Also...

To further illustrate the **SPREAD** statement, refer to input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Initiating and Smoothing" on page 4-3.

Examples

Figure 3-9 illustrates the use of the **SPREAD** statement for six different cases. For each case, the initial grid is spatially uniform and consists of 11 horizontal and 11 vertical grid lines. In each case, the left side of the grid is distorted, with the middle of the transition region between distorted and undistorted mesh occurring at approximately x=5 microns.

Cases (a) through (c) all specify **VOL.RAT**=0.5 so that the displacement of the grid lines specified with **UPPER** and **LOWER** are equal and opposite. Case (a) decreases the thickness of the region between the **UPPER** and **LOWER** grid lines to 1 micron while (b) increases the thickness to 7 microns. Case (c) is identical to (b) except the third grid line is placed at y=3 microns in the distorted region.

Case (d) specifies **VOL.RAT**=0 which causes the grid line specified with **LOWER** to remain at its original location. The thickness is increased to 7 microns. In case (e), the grid line specified by **LOWER** is moved to y=7 microns in the distorted

region and the line specified by **UPPER** remains fixed. Case (f) is identical to case (e) except that **GRADING** is specified to vary the spacing between grid lines in the distorted region.

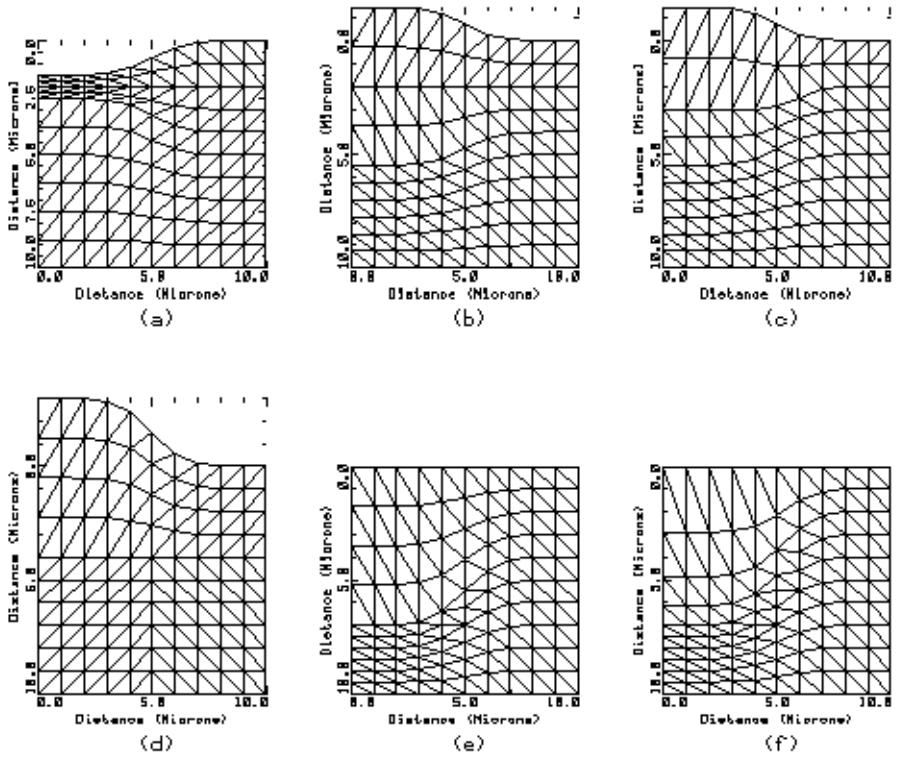


Figure 3-9 Six different spread statements on an initially uniform mesh

(a) **SPREAD LEFT WID=5. UP=1 LOW=5 THICK=1. VOL.RAT=0.5**
(b) **SPREAD LEFT WID=5. UP=1 LOW=5 THICK=7. VOL.RAT=0.5**
(c) **SPREAD LEFT WID=5. UP=1 LOW=5 THICK=7. VOL.RAT=0.5**
+ MID=3 Y.MID=3.0
(d) **SPREAD LEFT WID=5. UP=1 LOW=5 THICK=7. VOL.RAT=0.0**
(e) **SPREAD LEFT WID=5. UP=1 LOW=5 Y.LOW=7.**
(f) **SPREAD LEFT WID=5. UP=1 LOW=5 Y.LOW=7. GRADING=.67**

BOUNDARY

After the specification of an initial mesh, the **BOUNDARY** statement reads boundary information representing material interfaces from a file and adapts the mesh to the boundaries. The resulting grid closely conforms to the boundary information and can accurately represent highly nonplanar structures.

```
BOUNDARY
   IN.FILE=<c> [ASCII.IN] [ {2D.PROC | TSUPREM4} ]
   [ OUT.FILE=<c> [ASCII.OU] ]
   [X.SCALE=<n>] [Y.SCALE=<n>] [X.OFFSET=<n>] [Y.OFFSET=<n>]
   [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
   [X.TOLER=<n>] [Y.TOLER=<n>] [X.AVERAG] [Y.AVERAG]
   [ SHARP [ANGLE.CR=<n>] ] [ ASPECT [LOW.ASPE=<n>] [HIGH.ASP=<n>] ]
```

Parameter	Type	Definition	Default	Units
IN.FILE	char	The identifier of the input data file containing the boundaries to be read. **synonym: INFILE**	none	
ASCII.IN	logical	Specifies that the input data file is formatted.	False if **2D.PROC** is specified, else true.	
2D.PROC	logical	Specifies that the input data file was generated in the *Avant!* TCAD standard file format. If neither **2D.PROC** nor **TSUPREM4** is specified, boundaries are read from a simple boundary file.	false	
TSUPREM4	logical	Specifies that the input data file was generated by TSUPREM-4 for Medici. If neither **2D.PROC** nor **TSUPREM4** is specified, boundaries are read from a simple boundary file. **synonym: TS4**	false	
OUT.FILE	char	The identifier of the output data file containing the boundaries to be written. **synonym: OUTFILE**	none	
ASCII.OU	logical	Specifies that the output data file is formatted.	true	
X.SCALE	number	The factor by which boundary x coordinates are multiplied.	1.0	none
Y.SCALE	number	The factor by which boundary y coordinates are multiplied.	1.0	none
X.OFFSET	number	The distance by which boundary x coordinates are offset.	0.0	microns
Y.OFFSET	number	The distance by which boundary y coordinates are offset.	0.0	microns
X.MIN	number	The minimum allowed boundary x coordinate. Coordinates less than **X.MIN** are clipped to **X.MIN**.	The minimum x coordinate of the user-specified grid.	microns
X.MAX	number	The maximum allowed boundary x coordinate. Coordinates greater than **X.MAX** are clipped to **X.MAX**.	The maximum x coordinate of the user-specified grid.	microns

Parameter	Type	Definition	Default	Units
Y.MIN	number	The minimum allowed boundary y coordinate. Coordinates less than **Y.MIN** are clipped to **Y.MIN**.	The minimum y coordinate of the user-specified grid.	microns
Y.MAX	number	The maximum allowed boundary y coordinate. Coordinates greater than **Y.MAX** are clipped to **Y.MAX**.	The maximum y coordinate of the user-specified grid.	microns
X.TOLER	number	Before initiation of the boundary matching iterations, the minimum horizontal distance allowed between a grid line and a key point.	1% of the total width of the grid.	microns
Y.TOLER	number	Before initiation of the boundary matching iterations, the minimum vertical distance allowed between a grid line and a key point.	1% of the total height of the grid.	microns
X.AVERAG	logical	Specifies that the locations of vertical grid lines are averaged where possible to equidistribute the spacings between adjacent lines.	false	
Y.AVERAG	logical	Specifies that the locations of horizontal grid lines are averaged where possible to equidistribute the spacings between adjacent lines.	false	
SHARP	logical	Specifies that key points are to be allocated at boundary nodes that form the vertices of angles greater than **ANGLE.CR**.	false	
ANGLE.CR	number	The critical angle associated with the **SHARP** parameter.	45	degrees
ASPECT	logical	Specifies that rectangular grid elements are to be subdivided either vertically or horizontally if the ratio of their height to width is greater than **HIGH.ASP** or less than **LOW.ASPE**. Elements are subdivided so that the aspect ratios of the child elements are closer to 1 than that of their parents.	false	
LOW.ASPE	number	The low limit for aspect-ratio-based rectangular grid element subdivision. Elements whose ratio of height to width is less than **LOW.ASPE** are subdivided vertically. It is an error to set **LOW.ASPE** greater than 0.625.	0.5	none
HIGH.ASP	number	The high limit for aspect-ratio-based rectangular grid element subdivision. Elements whose ratio of height to width is greater than **HIGH.ASP** are subdivided horizontally. It is an error to set **HIGH.ASP** less than 1.6.	2.0	none

Description

The **BOUNDARY** statement allows sequences of Cartesian coordinate locations ("boundaries") representing essentially arbitrary material interfaces to be read from a file. These are then combined with a coarse Medici grid specification to generate a triangular grid. This grid contains a very low percentage of obtuse triangles, such that the boundaries coincide almost exactly with triangle sides.

The concept behind this capability was abstracted from a paper by Ciampolini et al., "Adaptive Mesh Generation Preserving the Quality of the Initial Grid," IEEE Transactions on Computer-Aided Design, Vol. 8, No. 5, May 1989.

The **BOUNDARY** statement was implemented primarily to afford the user a method of creating grids for applications in which device topography or structure is critical. Additionally, it can serve as a unidirectional interface between the *Avant!* TCAD process simulators, Taurus-Lithography and TSUPREM-4, and Medici.

See Also… To further illustrate the **BOUNDARY** statement, refer to Interface Examples, Chapter 10, "Medici Simulation of Electric Field Distribution" on page 10-23.

Input and Boundaries

Input to the grid generator consists of a set of boundaries plus an initial Medici mesh specification via the **MESH**, **X.MESH**, and **Y.MESH** statements. An analysis of the boundaries is performed to extract geometrical data such as the locations of boundary endpoints, kinks, and intersections. The mesh lines are manipulated slightly to take these locations into account, and are then used to form a terminating-line rectangular grid.

The boundaries are then matched to the grid and elements are subdivided iteratively until the grid can resolve the boundaries. Finally, the rectangular grid is converted to a standard Medici triangular grid.

Statement Ordering

To use the **BOUNDARY** statement, structure specification statements must occur in the following order:

MESH, **X.MESH**, **Y.MESH**, **BOUNDARY**, **REGION**, **ELECTRODE**, **PROFILE**.

Restrictions

ELIMINATE and **SPREAD** statements are not allowed in conjunction with the **BOUNDARY** statement.

File I/O

Reading boundaries from a file is accomplished by specifying **IN.FILE**, **ASCII.IN** (optional), and at most one of **2D.PROC** and **TSUPREM4**. For example, the following statement

```
BOUNDARY  IN.FILE=test.inb  ASCII.IN
```

reads boundaries from a formatted simple boundary file called *test.inb*. The file must contain the following information:

1. One integer: NB

   NB is the number of boundaries contained in this file.

2. NB integers: N_1, N_2,..., N_{NB}

   Each N_i is the number of points or nodes in boundary i.

3. $2(N_1 + N_2 + ... + N_{NB})$ reals: x_1^1, y_1^1,..., $x_{N_{NB}}^{NB}$, $x_{N_{NB}}^{NB}$

Each superscript indicates the boundary to which each coordinate belongs, and each subscript indicates the node number to which each coordinate belongs. The first real of each pair is an x coordinate (in microns) of a boundary node; the second is a y coordinate (in microns).

4. One 20-character string describing the date at which the file was written.

Termination, Overlap, and Intersect Examples

This section details some of the properties of the **BOUNDARY** statement using files as examples.

Termination

The following file describes a set of three boundaries containing 6, 2, and 2 nodes, respectively.

```
3
6  2  2
0.625000      -0.100000
0.625000      -0.046000
0.750000      -0.025000
2.250000      -0.025000
2.375000      -0.046000
2.375000      -0.100000

0.500000      -0.100000
0.500000       0.033000

2.500000      -0.100000
2.500000       0.033000
' 21-Aug-90 00:59:57 '
```

Boundaries must partition the simulation space. Thus, every boundary string must terminate on a boundary (a boundary's endpoints may lie on itself). Boundaries may terminate at grid edges, which are considered implicit boundaries.

Overlap

There are several restrictions to be aware of when creating a simple boundary file. The first is that boundaries that overlap (i.e., contain sections in common) must overlap at exactly the same points.

For example, the boundary file at left causes errors, while the file in the middle does not. It is actually preferable that boundaries not overlap at all. Superior to either of the other files is the one at right.

```
2                       2                       3
6  6                    6  6                    6  2  2

0.00       1.00         0.00       1.00         0.00       1.00
1.50       2.00         1.00       2.00         1.00       2.00
2.50       2.00         2.00       2.00         2.00       2.00
3.50       2.00         3.00       2.00         3.00       2.00
4.00       2.00         4.00       2.00         4.00       2.00
5.00       1.00         5.00       1.00         5.00       1.00

0.00       3.00         0.00       3.00         4.00       2.00
1.00       2.00         1.00       2.00         5.00       3.00
2.00       2.00         2.00       2.00
3.00       2.00         3.00       2.00         0.00       3.00
4.00       2.00         4.00       2.00         1.00       2.00
5.00       3.00         5.00       3.00
' 1-Sep-90 14:23:11 '  ' 1-Sep-90 14:23:15 '  ' 1-Sep-90 14:23:20 '
```

Intersect

Furthermore, it is preferable for boundaries which intersect—to intersect explicitly at exactly the same point. The boundary file at right is preferable to the one at left.

```
2                                       2
2  2                                    3  3

0.000000       1.000000                 0.000000       1.000000
1.000000       0.000000                 0.500000       0.500000
                                        1.000000       0.000000
0.000000       0.000000
1.000000       1.000000                 0.000000       0.000000
'  1-Sep-90 14:24:33 '                  0.500000       0.500000
                                        1.000000       1.000000
                                        '  1-Sep-90 14:24:40 '
```

Compatibility

This section details using the **BOUNDARY** statement with files written by *Avant!* TCAD products other than Medici, and other programs.

Avant! TCAD

The following Medici statement reads boundaries from a file *STRINGS* written in the *Avant!* TCAD standard file format by Taurus-Lithography:

```
BOUNDARY  2D.PROC  IN.FILE=STRINGS
```

The file must have been written with the Taurus-Lithography statement:

```
SAVEFILE  STRUCTURE  FILE=STRINGS
```

The following Medici statement reads boundaries from a file *STRUCT.MD* written by TSUPREM-4:

```
BOUNDARY TSUPREMA4 IN.FILE=STRUCT.MD OUT.FILE=struct.oub
+       ASCII.OU
```

The file must have been written with the TSUPREM-4 statement:

```
STRUCTURE  MEDICI  OUT.FILE=STRUCT.MD
```

Most regions are assigned automatically when gridding a boundary file written by TSUPREM-4; reading in a file written by Taurus-Lithography, however, currently requires the user to assign regions to the resulting grid. In either case, electrodes must be assigned by you.

Process Simulator Data Files

To write out boundaries read from process simulator data files in a form which can be edited, use the **OUT.FILE** and **ASCII.OU** parameters. The last **BOUNDARY** statement example above writes the simple boundary file *struct.oub.* Since the simple boundary file input and output formats are identical, the file *struct.oub* may be edited as required and then read again.

Transformation and Truncating

After boundaries are read, they are transformed (scaled and translated), in that order. That is, if x and y are the coordinates of a boundary node then the sequence of operations performed on x and y is:

$$x = (x \cdot \mathbf{X.SCALE}) + \mathbf{X.OFFSET} \qquad \text{Equation 3-1}$$

$$y = (y \cdot \mathbf{Y.SCALE}) + \mathbf{Y.OFFSET} \qquad \text{Equation 3-2}$$

The boundaries are then truncated to fit within the rectangle defined by **X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX**.

Boundary Analysis

After having been transformed and truncated, the boundaries are analyzed in order to extract significant geometric information, such as the locations of:

- Aspect ratio
- Kinks
- Intersections

Kinks

Kinks in a boundary are defined to occur when the **SHARP** parameter is specified and three successive boundary nodes form an angle greater than **ANGLE.CR** degrees (see Figure 3-10). These locations, known as "key points," are then used in two ways.

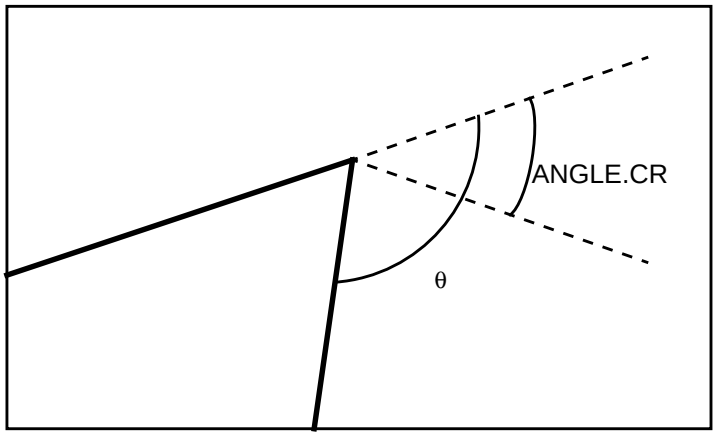


Figure 3-10 Kink Definition

- Any vertical grid line less than **X.TOLER** microns from a key point is snapped to that key point. Similarly, if the distance between any horizontal grid line and any key point is less than **Y.TOLER**, that line is also snapped to that key point.

  This process helps ensure that the initial rectangular mesh subdivisions do not create rectangles with extreme aspect ratios.

- Initial subdivisions of the mesh are performed at the key points.

  Every element that contains a key point is subdivided at that key point (see Figure 3-11).

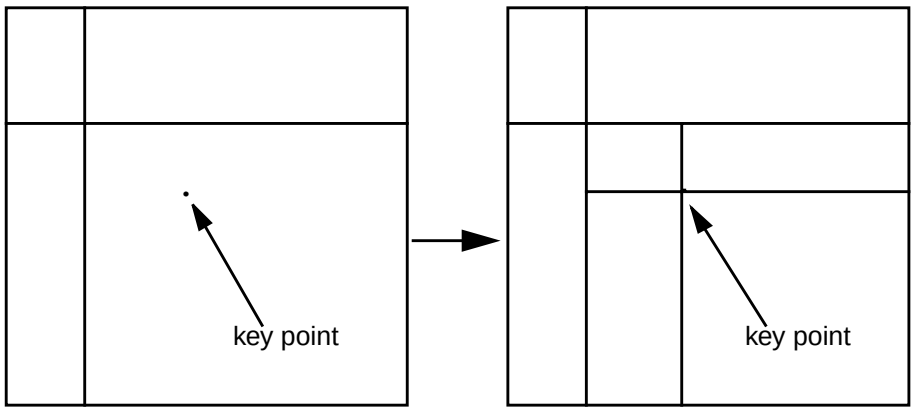


Figure 3-11 Subdivision performed on a rectangle grid element

Boundary Matching (Intersections)

Boundary matching computes the points of intersection between the boundaries and the rectangular grid elements. This iterative process causes subdivision of elements until every element can be triangulated using certain simple triangulation schemes. In addition, an element is subdivided whenever the **ASPECT** parameter

is specified and the height-to-width ratio of that element exceeds **LOW.ASPE** or **HIGH.ASP**.

Aspect-Ratio-Based Subdivision

In general, grids conditioned with aspect-ratio-based subdivision have more smoothly-varying element sizes than grids without; they have, in addition, a smaller percentage of obtuse triangles. Setting **LOW.ASPE** to 0.5 or greater and **HIGH.ASP** to 2.0 or less results in grids with the fewest obtuse triangles, but the tighter the aspect-ratio subdivision bounds, the more elements are created during generation of the grid.

Troubleshooting

As the **BOUNDARY** statement is still experimental, problems may occur during boundary analysis or matching. This section offers a cross-section of potential problems and hints on attacking them.

Unsuccessful Termination, Subdivision Error

SYMPTOM: Boundary matching does not terminate successfully, and at least one message of the form:

```
Warning during subdivision of rectangular element  1393.
Attempted subdivision near (x,y) = (    2.9087,   -1.3521)(microns)
which would have caused the resulting elements to be smaller than
the allowed minimum size.  Turning subdivision for this element off
```

occurs, followed by an error message of the form:

```
Error number  885 detected in line number  45
Unable to refine the rectangular grid of  1510 elements further
because the elements are becoming too small.
```

DIAGNOSIS AND POSSIBLE SOLUTIONS

Examine the *differences* in the number of grid points between successive boundary matching iterations. If these differences are decreasing monotonically to zero, then the original grid may have been too fine. Medici does not allow rectangular grid elements with any dimension less than 0.01 Angstrom.

If the differences just before the occurrence of the error message are large (on the order of 10% of the total number of grid points), or the differences are *not* decreasing monotonically to zero, then there may be certain grid features which are attracting large numbers of element subdivisions.

For example, if an element is subdivided into elements with extreme aspect ratios and **ASPECT** has been specified, grid consistency requires that these elements be subdivided multiple times (see Figure 3-12).

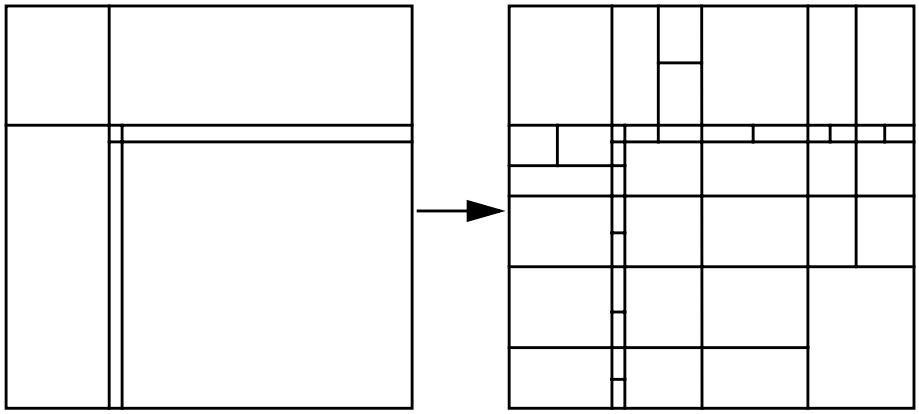


Figure 3-12 Effect of grid consistency requirements on elements with extreme aspect ratios

If the **ASPECT** and **SHARP** parameters are on, it is helpful to loosen their tolerances (i.e., decrease **LOW.ASPE**, increase **HIGH.ASP**, or increase **ANGLE.CR**) or to turn them off, at least temporarily, in order to isolate the problems.

Other aids include varying the number and placement of grid lines, and as a last resort, simplifying the boundaries contained in the boundary file to retain essential features while eliminating unnecessary details.

Unassigned Elements

SYMPTOM: Error messages are received of the form:

```
Elements extending through the ranges below were not assigned a region
number.
xmin =     0.0000;  xmax =    13.0000
ymin =    -1.0000;  ymax =     1.0000

Error number  199 detected in line number  45
Some elements have not been assigned a region number.
```

DIAGNOSIS AND POSSIBLE SOLUTIONS: A **REGION** statement contained an error or was omitted, causing a set of contiguous elements bounded by the rectangle defined by xmin, xmax, ymin, and ymax not to be assigned region numbers. Identify a point (x,y) within this set of elements and insert into the input file a **REGION** statement of the form:

REGION NUM=<*n*> **X=**<*x*> **Y=**<*y*> <*material*>

Overwriting Region

SYMPTOM: A warning message is received of the form:

Warning: Overwriting region <*r1*> with region <*r2*>.

DIAGNOSIS AND POSSIBLE SOLUTIONS: The point (specified with the parameters X and Y) associated with region <*r2*> was contained within region <*r1*>. If region <*r1*> is not to be identified as region <*r2*>, use different point specifications for the region statements associated with regions <*r1*> and <*r2*>. It is also possible that the boundaries may have been inconsistent (e.g., not all

boundary endpoints terminated on some other boundary node), resulting in "spilling" of one region onto another.

Gridlines Removed

SYMPTOM: The final grid looks as if some initially specified grid lines had been removed.

DIAGNOSIS AND POSSIBLE SOLUTIONS: **X.TOLER** and **Y.TOLER** may be too large for the specified grid. It may be necessary to set **X.TOLER** and **Y.TOLER** smaller than the minimum grid line spacing.

Examples

This section details several different uses of the boundary statement using example files and illustrations.

TSUPREM-4 LOCOS Process

The following example illustrates the use of the boundary statement to read boundaries from a TSUPREM-4 simulation of a LOCOS process (see Figure 3-13). The following is a partial output listing:

```
                Statements input from file mds4bnd

  1... TITLE     MEDICI File Boundary Matching
  2... COMMENT   TSUPREM-4 ==> MEDICI:  LOCOS process + doping

  3... COMMENT   Plot TSUPREM-4 grid for reference.
  4... MESH      TSUPREM  IN.FILE=S4LOCOS
  5... PLOT.2D   GRID  TITLE="T-SUPREM4 LOCOS Grid"  FILL
   ... +         X.MIN=-1.5  X.MAX=0  Y.MIN=-0.5  Y.MAX=0.5

  6... COMMENT   Now create an initial mesh and read in boundaries.
  7... MESH
  8... X.MESH    X.MIN=-1.5  WIDTH=1.5  N.SPACES=14
  9... Y.MESH    Y.MIN=-0.5  WIDTH=1.0  N.SPACES=9

 10... COMMENT   Boundaries.
 11... BOUNDARY  TSUPREM4  IN.FILE=S4LOCOS  OUT.FILE=MDS4OUT
   ... +         SHARP  ASPECT

 12... COMMENT   Regions:  Most regions have been assigned automatically.
   ... +         Region 10 = deposited oxide
 13... REGION    NUM=10  X=-0.5  Y=-0.3  OXIDE

 14... COMMENT   Electrodes:  1 = substrate
 15... ELECTROD  NUM=1  PERIMETER  Y.MIN=0.5

 16... COMMENT   Profiles.
 17... PROFILE   TSUPREM4  IN.FILE=S4LOCOS

 18... COMMENT   Plot of MEDICI grid.
 19... PLOT.2D   GRID  TITLE="MEDICI LOCOS Grid"  FILL

Rectangular grid element size tolerances (microns):
  xtol =     0.0150;  ytol =     0.0100

Read TSUPREM-4 file from S4LOCOS

Boundaries extracted from TSUPREM4 file: S4LOCOS

Boundary limits (microns):
  xmin =    -1.5000;  xmax =     0.0000
```

```
  ymin =    -0.3322;  ymax =     0.5000

Boundaries written to MDS4OUT

Mesh statistics (finite box) :
    Total iterations =  5
    Total grid points =  371
    Total no. of rectangles =  315

Mesh statistics (rectangular) :
    Total grid points =  475
    Total no. of triangles =  892
    Obtuse triangles =   32  ( 3.6%)

  Region    Material      X-min        X-max        Y-min        Y-max
   Name       Type      (microns)    (microns)    (microns)    (microns)
----------  --------    ---------    ---------    ---------    ---------
1           Silicon       -1.5000       0.0000       0.0000       0.5000
2           Oxide         -1.5000       0.0000      -0.3131       0.2225
3           Nitride       -1.5000      -0.2500      -0.3322      -0.0300
10          Oxide         -1.5000       0.0000      -0.5000      -0.0800

Electrode    Number       X-min        X-max        Y-min        Y-max
   Name     of Nodes    (microns)   (microns)    (microns)    (microns)
----------  --------    ---------   ---------    ---------    ---------
1              15         -1.5000      0.0000       0.5000       0.5000
```

Initial Grid Lines

This example uses a uniform mesh with a relatively small number of initial grid lines since this simulation places no particular requirements on the grid itself.

Partitioning the Mesh

Since the boundaries and the mesh edges must partition the mesh, you must know where to place the mesh limits in the **X.MESH** and **Y.MESH** statements. If you performed the original TSUPREM-4 simulation, then the knowledge is already available; otherwise, the **BOUNDARY** statement may by used to read the boundaries first and print the extents of the boundaries or write a formatted simple boundary file for perusal before allocating the final mesh.

Extraction, Conformity, and Subdivision

The **BOUNDARY** statement reads the TSUPREM-4 file *S4LOCOS* and extracts from it a set of boundaries. This file is the same file used by Medici in the **MESH** statement to read in TSUPREM-4 grids, and to read in doping profiles. The extracted boundaries are written immediately to the formatted simple boundary file *MDS4OUT* for the user to peruse or edit.

The **SHARP** parameter was specified to ensure that the resulting grid conforms as closely as possible to the boundaries of the original TSUPREM-4 grid. Aspect-ratio-based subdivision was turned on to reduce the number of obtuse triangles.

Regions

The **BOUNDARY** statement interface to TSUPREM-4 assigns most regions automatically, but as can be seen from the original TSUPREM-4 grid, there is no mesh above the nitride layer. Consequently, the **BOUNDARY** statement requires the user to specify the material type of the elements in the grid above the nitride. Region specification is easy because the user only need pick a point within a region.

The addition of a substrate electrode satisfies the Medici requirement that there be at least one electrode in any simulation. This electrode was defined by specifying all nodes on the grid perimeter *and* at or below y = 0.5.

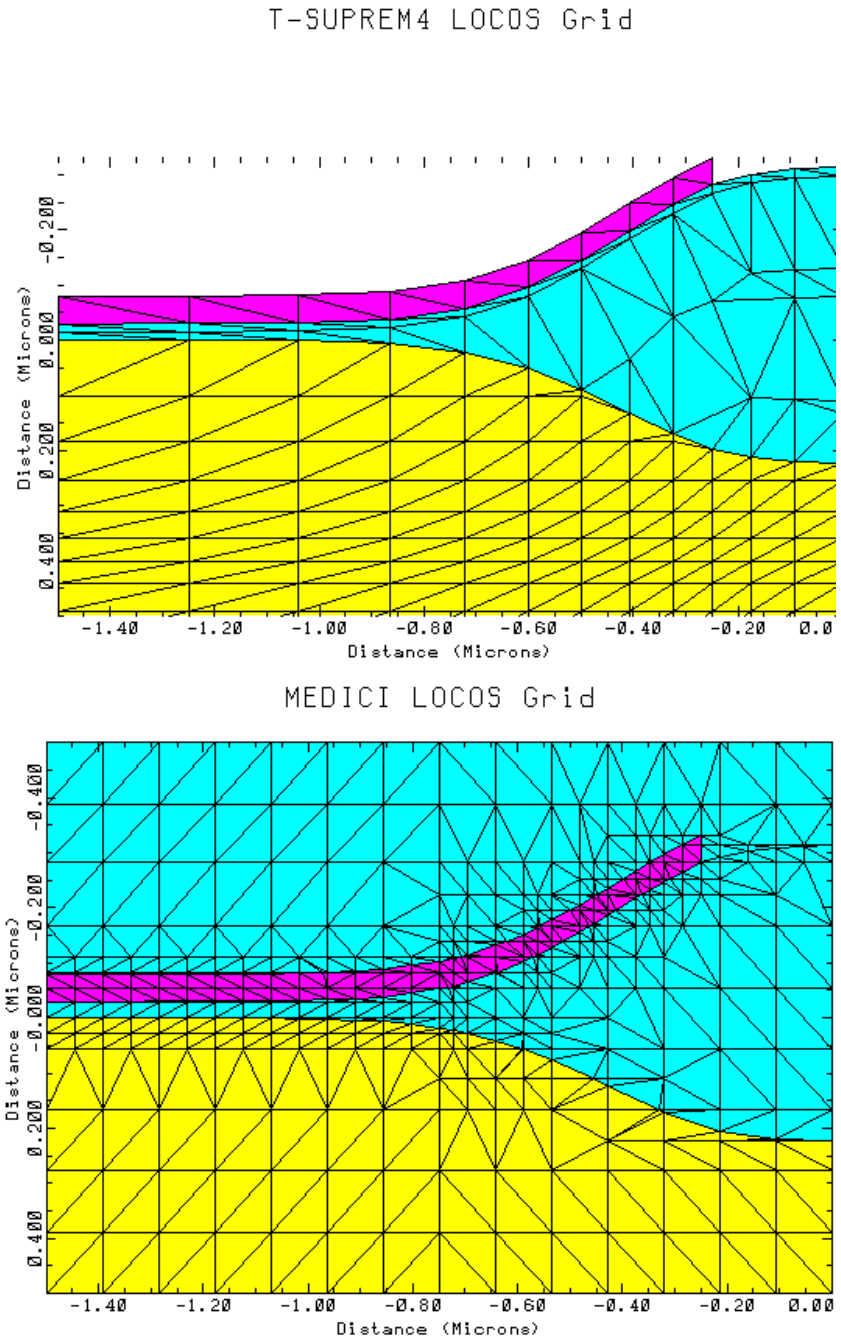


Figure 3-13 Comparison of a TSUPREM-4 grid and one possible Medici counterpart

TSUPREM4

The **TSUPREM4** statement provides a means for defining the regions and profiles in a Medici rectangular grid structure by importing a TSUPREM-4 structure saved in **MEDICI** format.

TSUPREM4

```
IN.FILE=<c>
[X.LEFT=<n>] [X.RIGHT=<n>] [Y.TOP=<n>] [Y.BOT=<n>]
[X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
[X.OFFSET=<n>] [Y.OFFSET=<n>] [X.INTERF=<n>]
[IMPURITY] [FLIP] [SYMMETRI]
```

Parameter	Type	Definition	Default	Units
IN.FILE	char	The name of the file containing the TSUPREM-4 structure saved in Medici format.	none	
X.LEFT	number	The minimum x coordinate in the Medici mesh that will receive structure and profile information from the TSUPREM-4 file.	The minimum x location in the Medici mesh.	microns
X.RIGHT	number	The maximum x coordinate in the Medici mesh that will receive structure and profile information from the TSUPREM-4 file.	The maximum x location in the Medici mesh.	microns
Y.TOP	number	The minimum y coordinate in the Medici mesh that will receive structure and profile information from the TSUPREM-4 file.	The minimum y location in the Medici mesh.	microns
Y.BOT	number	The maximum y coordinate in the Medici mesh that will receive structure and profile information from the TSUPREM-4 file.	The maximum y location in the Medici mesh.	microns
X.MIN	number	The minimum x coordinate in the TSUPREM-4 structure that will be used when importing structure and profile information.	The minimum x location in the TSUPREM-4 structure.	microns
X.MAX	number	The maximum x coordinate in the TSUPREM-4 structure that will be used when importing structure and profile information.	The maximum x location in the TSUPREM-4 structure.	microns
Y.MIN	number	The minimum y coordinate in the TSUPREM-4 structure that will be used when importing structure and profile information.	The minimum y location in the TSUPREM-4 structure.	microns
Y.MAX	number	The maximum y coordinate in the TSUPREM-4 structure that will be used when importing structure and profile information.	The maximum y location in the TSUPREM-4 structure.	microns
X.OFFSET	number	The horizontal offset of **X.MIN** relative to **X.LEFT**.	0.0	microns

Parameter	Type	Definition	Default	Units
Y.OFFSET	number	The vertical offset of **Y.MIN** relative to **Y.TOP**. However, if **X.INTERF** is specified then the insulator-semiconductor interface will be aligned with the location y=0 in Medici. In this case, **Y.OFFSET** represents an additional vertical offset of the TSUPREM-4 structure after this alignment takes place.	0.0	microns
X.INTERF	number	The TSUPREM-4 horizontal location that will be used to find the insulator-semiconductor interface. The interface at this location will be aligned with y=0 in the Medici mesh.	none	microns
IMPURITY	logical	Specifies that the TSUPREM-4 impurity profiles are read.	true	
FLIP	logical	Specifies that the TSUPREM-4 coordinates are flipped.	false	
SYMMETRI	logical	Specifies that the TSUPREM-4 structure is used to create a Medici structure that is symmetric about the horizontal mid-point of the Medici mesh.	false	

Description

This statement can be used to read the results of a TSUPREM-4 simulation onto a Medici mesh. Both the topography and profile information from TSUPREM-4 will be used. The elements in the Medici mesh will be assigned a material type based on the corresponding TSUPREM-4 material at this location.

The parameters **X.LEFT**, **X.RIGHT**, **Y.TOP**, and **Y.BOT** define the portion of the Medici mesh to receive structure information from the TSUPREM-4 file. The parameters **X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX** define the portion of the TSUPREM-4 structure that contributes to the Medici structure.

Examples

Some examples illustrating the use of the **TSUPREM4** statement are shown in Figures 3-14 and 3-15. Figure 3-14 shows a simple TSUPREM-4 structure and a Medici mesh prior to importing the TSUPREM-4 structure. Figure 3-15 shows Medici structures that were obtained from importing the TSUPREM-4 structure in two different ways. In both cases, the **X.INTERF** parameter has been used to align the semiconductor/insulator interface in TSUPREM-4 (found at x=0) with the location y=0 in Medici. The figure on the right was generated by additionally specifying the **SYMM** parameter.

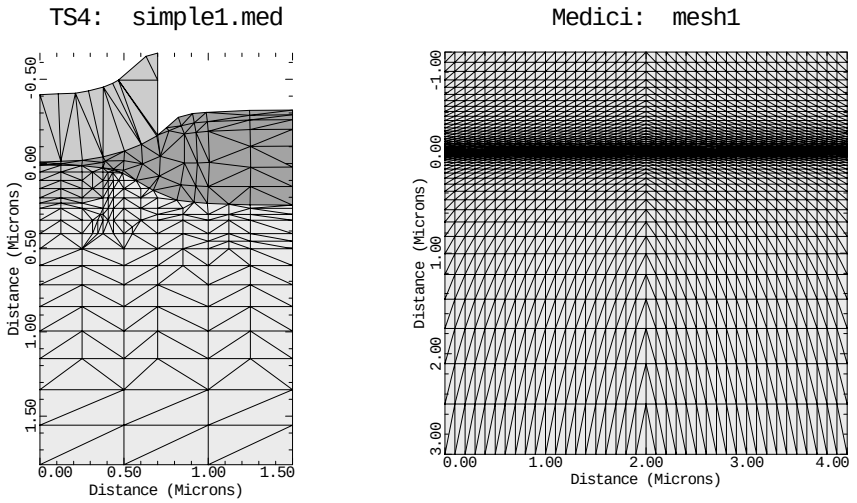
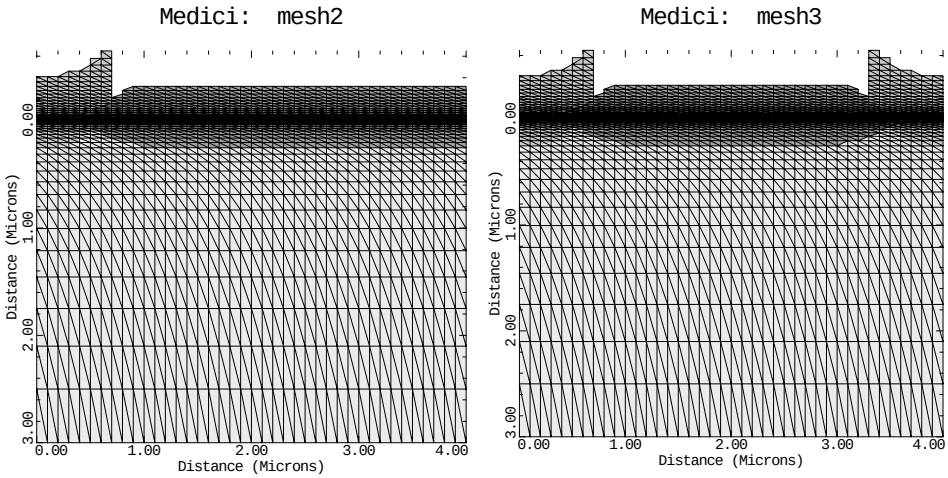


Figure 3-14 Simple TSUPREM-4 structure saved in Medici format and the original Medici mesh prior to importing the structure




```
mesh

x.mesh width=4 n.spaces=40
y.mesh  depth=1 h1=0.1  h2=0.01 y.min=-1
y.mesh depth=3 h1=0.01 h2=0.5

tsuprem4  in.file=simple1.med
+         x.interf=0.0

electr   name=bottom  y.min=2.9
plot.2d  grid  fill  scale
+         title="Medici:  mesh2"
```

```
mesh

x.mesh  width=4  n.spaces=40
y.mesh  depth=1 h1=0.1  h2=0.01 y.min=-1
y.mesh  depth=3 h1=0.01 h2=0.5

tsuprem4  in.file=simple1.med
+         x.interf=0.0  symm

electr   name=bottom  y.min=2.9
plot.2d  grid  fill  scale
+        title="Medici:  mesh3"
```

Figure 3-15 Two examples of Medici structures after importing the TSUPREM-4 structure

REGION

The **REGION** statement defines the location of materials in a rectangular mesh.

```
REGION

   NAME=<c>
    Semiconductor Materials
   {  ( {  SILICON | GAAS | POLYSILI | GERMANIU | SIC | SEMICOND
         | SIGE | ALGAAS | A-SILICO | DIAMOND | HGCDTE | INAS | INGAAS |
         | INP | S.OXIDE | ZNSE | ZNTE | ALINAS | GAASP | INGAP | INASP
        }

    Semiconductor Material Parameters
        [X.MOLE=<n>]
        [ {X.END=<n> | X.SLOPE=<n>} {X.LINEAR | Y.LINEAR} ]
      )

    Insulator Materials
    | OXIDE | NITRIDE | SAPPHIRE | OXYNITRI | INSULATO
   }

    Location
   {  ( [ {X.MIN=<n> | IX.MIN=<n>} ] [ {X.MAX=<n> | IX.MAX=<n>} ]
        [ {Y.MIN=<n> | IY.MIN=<n>} ] [ {Y.MAX=<n> | IY.MAX=<n>} ]
        [ { (ROTATE R.INNER=<n> R.OUTER=<n> X.CENTER=<n> Y.CENTER=<n> )
          | (POLYGON X.POLY=<a> Y.POLY=<a> )
          }
        ]
      )
    | [ X=<n> Y=<n> ]
   }

   [VOID]
```

Parameter	Type	Definition	Default	Units
NAME	char	The name identifying the region. A maximum of 200 regions are allowed in a device structure. Names may be up to 20 characters long, however, names of ten characters or less are recommended since the name may be truncated to ten characters in certain places in the output listing. **synonym: NUMBER**	The material name.	
Semiconductor Materials				
SILICON	logical	Specifies that the region is silicon.	false	
GAAS	logical	Specifies that the region is gallium arsenide.	false	
POLYSILI	logical	Specifies that the region is polysilicon. At the present time, the default material properties associated with regions specified as **POLYSILI** are the same as the default material properties associated with regions specified as **SILICON**.	false	

Parameter	Type	Definition	Default	Units
GERMANIU	logical	Specifies that the region is germanium.	false	
SIC	logical	Specifies that the region is silicon carbide.	false	
SEMICOND	logical	Specifies that the region is a generic semiconductor. The default material properties associated with regions specified as **SEMICOND** are the same as the default material properties associated with regions specified as **SILICON**.	false	
SIGE	logical	Specifies that the region is $Si_{1-x}Ge_x$. The default material properties associated with regions specified as **SIGE** are the same as the default material properties associated with regions specified as **SILICON**.	false	
ALGAAS	logical	Specifies that the region is $Al_xGa_{1-x}As$.	false	
A-SILICO	logical	Specifies that the region is amorphous silicon. The default material properties associated with regions specified as **A-SILICO** are the same as the default material properties associated with regions specified as **SILICON**.	false	
DIAMOND	logical	Specifies that the region is diamond.	false	
HGCDTE	logical	Specifies that the region is HgCdTe (mercury cadmium telluride).	false	
INAS	logical	Specifies that the region is indium arsenide. The default material properties associated with regions specified as **INAS** are the same as the default material properties associated with regions specified as **GAAS**.	false	
INGAAS	logical	Specifies that the region is $Inl_xGa_{1-x}As$.	false	
INP	logical	Specifies that the region is indium phosphide. The default material properties associated with regions specified as **INP** are the same as the default material properties associated with regions specified as **GAAS**.	false	
S.OXIDE	logical	Specifies that the region is silicon dioxide (treated as a wide bandgap semiconductor).	false	
ZNSE	logical	Specifies that the region is zinc selenide. The default material properties associated with regions specified as **ZNSE** are the same as the default material properties associated with regions specified as **GAAS**.	false	
ZNTE	logical	Specifies that the region is zinc telluride. The default material properties associated with regions specified as **ZNTE** are the same as the default material properties associated with regions specified as **GAAS**.	false	
ALINAS	logical	Specifies that the region is $Al_xIn_{1-x}As$.	false	
GAASP	logical	Specifies that the region is $GaAs_xP_{1-x}$.	false	
INGAP	logical	Specifies that the region is $In_xGa_{1-x}P$.	false	
INASP	logical	Specifies that the region is $InAs_{1-x}P_x$.	false	

Parameter	Type	Definition	Default	Units
Semiconductor Material Parameters				
X.MOLE	number	The mole fraction to use in the region for compound materials. For graded compounds, **X.MOLE** represents the initial mole fraction at the left, top, or front edge of the region depending on whether **X.LINEAR**, or **Y.LINEAR**, respectively, is specified. This parameter is only used with the Heterojunction Device AAM. **synonym: X.INITIA**	0.0	none
X.END	number	The mole fraction for graded compounds at the right, bottom, or backedge of the region depending on whether **X.LINEAR**, or **Y.LINEAR**, respectively, is specified. This parameter is only used with the Heterojunction Device AAM.	none	none
X.SLOPE	number	The slope of the mole fraction for graded compounds. If this parameter is used, the mole fraction has a value of **X.MOLE** at the left, top or front edge of the region and a value of **X.MOLE** + *width* * **X.SLOPE** at the right, bottom or back edge of the region, where *width* is the width or depth of the region. This parameter is only used with the Heterojunction Device AAM.	none	microns
X.LINEAR	logical	Specifies that the mole fraction grading is in the x-direction. This parameter is only used with the Heterojunction Device AAM.	false	
Y.LINEAR	logical	Specifies that the mole fraction grading is in the y direction. This parameter is only used with the Heterojunction Device AAM.	True if **X.END** or **X.SLOPE** is specified and **X.LINEAR** is not specified.	
Insulator Materials				
OXIDE	logical	Specifies that the region is silicon dioxide. **synonym: SIO2**	false	
NITRIDE	logical	Specifies that the region is silicon nitride. **synonym: SI3N4**	false	
SAPPHIRE	logical	Specifies that the region is sapphire.	false	
OXYNITRI	logical	Specifies that the region is oxynitride. The default material properties associated with regions specified as **OXYNITRI** are the same as the default material properties associated with regions specified as **OXIDE**.	false	
INSULATO	logical	Specifies that the region is an insulator. The default material properties associated with regions specified as **INSULATO** are the same as the default material properties associated with regions specified as **OXIDE**.	false	
Location				
X.MIN	number	The minimum x location of the region.	The minimum x location for the device structure.	microns
IX.MIN	number	The minimum x node index of the region. **synonym: IX.LOW**	1	none

Parameter	Type	Definition	Default	Units
X.MAX	number	The maximum x location of the region.	The maximum x location for the device structure.	microns
IX.MAX	number	The maximum x node index of the region. **synonym: IX.HIGH**	The maximum x node index for the device structure.	none
Y.MIN	number	The minimum y location of the region.	The minimum y location for the device structure.	microns
IY.MIN	number	The minimum y node index of the region. **synonym: IY.LOW**	1	none
Y.MAX	number	The maximum y location of the region.	The maximum y location for the device structure.	microns
IY.MAX	number	The maximum y node index of the region. **synonym: IY.HIGH**	The maximum y node index for the device structure.	none
X	number	The x coordinate of a point within a bounded region.	none	microns
Y	number	The y coordinate of a point within a bounded region.	none	microns
ROTATE	logical	Specifies that the region is rotated around a rotation center.	false	
X.CENTER	number	Specifies the x location of the rotation center.	none	microns
Y.CENTER	number	Specifies the y location of the rotation center.	none	microns
R.INNER	number	The inner radius (distance from the rotation center) of a circular region.	none	microns
R.OUTER	number	The outer radius (distance from the rotation center) of a circular region.	none	microns
POLYGON	logical	Specifies that the region is a polygon. The polygon vertices are defined by a pair of arrays: **X.POLY** and **Y.POLY**.	false	
X.POLY	array	Specifies an array of x coordinates of the polygon vertices. Number of the vertices should not exceed 40.	none	microns
Y.POLY	array	Specifies an array of y coordinates of the polygon vertices. Number of the vertices should not exceed 40.	none	microns
VOID	logical	Specifies that a region or portion of a region (as specified by **X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX)** is voided (i.e. grid points and mesh are removed from the region).	false	

Description

Every mesh element in the device structure must be defined to be some material. Multiple **REGION** statements which specify the same **NAME** are allowed providing that the material specified on each such statement is the same. This makes it possible to specify regions which have complicated shapes.

See Also... To further illustrate the **REGION** statement, refer to input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Device Regions" on page 4-5.

Material Types

The selection of a material type on the **REGION** statement causes the program to select default semiconductor or insulator parameters to associate with that region. These parameters may be modified from their default values on the **MATERIAL** and **MOBILITY** statements.

Multiple insulator and semiconductor material types can be specified for a simulation using **REGION** statements.

Boundaries

The boundaries of a region can be specified in one of two ways:

- A bounding box, polygon, or a circular segment can be specified.
- A point within an existing region read in from a file can be specified.

Bounding Box

A bounding box can be specified which defines bounds for the region boundaries. Either nodal indices or coordinate values may be used to define the region boundaries. Any unspecified bound will default to the edges of the device.

Polygon

A polygon can be specified which defines boundaries of a region. The following statement could be used to specify a triangular polysilicon region:

```
REGION  NAME=STRANGE  POLYGON  POLY
+ X.POLY=(1, 2, 3)
+ Y.POLY=(1, 2, 1)
```

Rotation

A circular region can be defined by specifying a **0** parameter. The following statement could be used to specify a donut-shaped nitride region with a center at x=0 and y=0, internal radius of 0.5 micron and external radius of 1 micron:

```
REGION  NAME=RING  ROTATE  NITRIDE
+ X.CENTER=0  Y.CENTER=0
+ R.INNER=0.5  R.OUTER=1
```

A zero internal radius would convert a donut-shaped region into a circular one.

Point Within an Existing Region

A coordinate pair (X,Y) can be used to specify a point within an existing region which is read in from a file.

The use of the parameters **X** and **Y** for identifying a region is only valid when boundaries have been read using a **BOUNDARY** statement.

Examples

The following **REGION** statements may be used to define the material regions for a MOSFET that has an interface between oxide and silicon at y=0. In this example, the silicon region was named "Body" and the oxide region was named "SiO2":

```
REGION    NAME=Body  SILICON
REGION    NAME=Si02  OXIDE  Y.MAX=0
```

In the following example, the entire device structure is defined to be GaAs (since no region boundaries were specified). Since the **NAME** parameter is not specified, the region name is the same as the material name (GaAs):

```
REGION    GAAS
```

ELECTRODE

The **ELECTRODE** statement specifies the placement of electrodes in a device structure.

```
ELECTRODE

  NAME=<c> [VOID]
   {  ( [ {TOP | BOTTOM | LEFT | RIGHT | INTERFAC | PERIMETE} ]
        [ {X.MIN=<n> | IX.MIN=<n>} ] [ {X.MAX=<n> | IX.MAX=<n>} ]
        [ {Y.MIN=<n> | IY.MIN=<n>} ] [ {Y.MAX=<n> | IY.MAX=<n>} ]
        [ { (ROTATE X.CENTER=<n> Y.CENTER=<n> R.INNER=<n> R.OUTER=<n>)
           | (POLYGON X.POLY=<a> Y.POLY=<a>)
          }
        ]
      )
    | [ X=<n> Y=<n> ]
    | [REGION=<c>]
   }
   [MAJORITY]

 Lattice Temperature AAM Parameters
   [THERMAL]
```

Parameter	Type	Definition	Default	Units
NAME	char	The name of the electrode. A maximum of 200 electrodes are allowed in a device structure. Names must be less than 20 characters long, however a length of ten characters or less is recommended since names may be truncated to ten characters at certain places in the output listing. The Circuit analysis AAM also requires names of ten characters or less since the electrode name used by the program is constructed by concatenating the device name and the user-specified electrode name. **synonym: NUMBER**	none	
VOID	logical	Specifies that interior nodes and elements of a non-zero-area electrode are removed from the simulation grid.	false	
TOP	logical	Specifies that the electrode lies along the top edge of the device structure.	false	
BOTTOM	logical	Specifies that the electrode lies along the bottom edge of the device structure.	false	
LEFT	logical	Specifies that the electrode lies along the left edge of the device structure.	false	
RIGHT	logical	Specifies that the electrode lies along the right edge of the device structure.	false	
INTERFAC	logical	Specifies that the electrode lies along an insulator-semiconductor interface.	false	

Parameter	Type	Definition	Default	Units
PERIMETE	logical	Specifies that the electrode lies along the perimeter of the device structure.	false	
X.MIN	number	The minimum x location of the electrode.	The minimum x location for the device structure.	microns
IX.MIN	number	The minimum x node index of the electrode. **synonym:IX.LOW**	1	none
X.MAX	number	The maximum x location of the electrode.	The maximum x location for the device structure.	microns
IX.MAX	number	The maximum x node index of the electrode. **synonym:IX.HIGH**	The maximum x node index for the device structure.	none
Y.MIN	number	The minimum y location of the electrode.	The minimum y location for the device structure.	microns
IY.MIN	number	The minimum y node index of the electrode. **synonym:IY.LOW**	1	none
Y.MAX	number	The maximum y location of the electrode.	The maximum y location for the device structure.	microns
IY.MAX	number	The maximum y node index of the electrode. **synonym:IY.HIGH**	The maximum y node index for the device structure.	none
ROTATE	logical	Specifies that the electrode is rotated around a rotation center.	false	
X.CENTER	number	Specifies the x location of the rotation center.	none	microns
Y.CENTER	number	Specifies the y location of the rotation center.	none	microns
R.INNER	number	The inner radius (distance from the rotation center) of a circular electrode.	none	microns
R.OUTER	number	The outer radius (distance from the rotation center) of a circular electrode.	none	microns
POLYGON	logical	Specifies that the electrode is a polygon. The polygon vertices are defined by a pair of arrays: **X.POLY** and **Y.POLY**.	false	
X.POLY	array	Specifies an array of x coordinates of the polygon vertices. Number of the vertices should not exceed 40.	none	microns
Y.POLY	array	Specifies an array of y coordinates of the polygon vertices. Number of the vertices should not exceed 40.	none	microns
X	number	The x coordinate of a point which locates a region that is converted to an electrode.	none	microns
Y	number	The y coordinate of a point which locates a region that is converted to an electrode.	none	microns
REGION	char	The name of a region that is converted to an electrode.	none	

Parameter	Type	Definition	Default	Units
MAJORITY	logical	Specifies that the electrode forms a majority carrier contact only. The majority carrier quasi-Fermi level is set equal to the electrode potential. Minority carriers are unaffected by this electrode.	false	
Lattice Temperature AAM Parameters				
THERMAL	logical	Specifies that this is a thermal electrode used to set the lattice temperature at this location. This parameter is only used with the Lattice Temperature AAM.	false	

Description

The **ELECTRODE** statement serves to define the boundaries of an electrode within the device structure. Electrodes may be specified in any order.

Multiple **ELECTRODE** statements may be used to define the boundaries of a single electrode. The same electrode name may also be assigned to electrodes that are not in contact with each other within the device structure. This may be desired, for instance, if two electrodes are to be biased in exactly the same way.

See Also...

To further illustrate the **ELECTRODE** statement, refer to input file *mdex1* N-Channel MOSFET Examples, Chapter 4, "Electrode Locations" on page 4-5.

Boundaries

The boundaries of an electrode can be specified in one of three ways:

- A bounding box, polygon, or a circular segment can be specified.
- A region name can be specified.
- A point within a region can be specified.

Bounding Box

A bounding box can be specified which defines bounds for the electrode boundaries. All nodes within these bounds become part of the electrode. The bounds may be defined using the following:

- Nodal indices
- Coordinate values
- Parameters **TOP**, **BOTTOM**, **LEFT**, **RIGHT**, **INTERFAC**, and **PERIMETE**.

Unspecified bounds default to the device edges. As an example, the following statements could be used to specify the electrodes for a MOSFET device:

```
ELECTROD  NAME=Drain  INTERFAC  X.MIN=2.5
ELECTROD  NAME=Gate  TOP       X.MIN=1.0  X.MAX=2.0
ELECTROD  NAME=Source INTERFAC  X.MAX=0.5
ELECTROD  NAME=Substrate  BOTTOM
```

Polygon

A polygon can be specified which defines boundaries of a electrode. The following statement could be used to specify a triangular electrode:

```
ELECTROD  NAME=STRANGE  POLYGON
+ X.POLY=(1, 2, 3)
+ Y.POLY=(1, 2, 1)
```

If both bounding box and polygon are provided in an **ELECTRODE** statement, then the electrode is defined only where the two overlap. Therefore, a bounding box can be used to truncate a polygonal electrode.

Rotation

A circular electrode can be defined by specifying a **ELECTRODE** parameter. The following statement could be used to specify a donut-shaped electrode with a center at x=0 and y=0, internal radius of 0.5 micron and external radius of 1 micron:

```
ELECTROD  NAME=RING  ROTATE
+ X.CENTER=0  Y.CENTER=0
+ R.INNER=0.5  R.OUTER=1
```

A zero internal radius would convert a donut-shaped electrode into a circular one.

If both bounding box and rotation are specified in an **ELECTRODE** statement, then the electrode is defined only where the two overlap. Therefore, a bounding box can be used to define segments of a circle or a donut.

Region Name

A region name may be specified that is converted to an electrode. In this case, every node in the specified region becomes part of the electrode. As an example, the following statement converts region named **Top_part** into an electrode. The electrode name in this example is arbitrarily chosen to be **Anode**.

```
ELECTROD  NAME=Anode  REGION=Top_part
```

Point Within a Region Specification

The method is very similar to the region name method. A coordinate pair (X,Y) is used to specify a point within a region. As in the previous case, every node in the region where the point (X,Y) is located becomes part of the electrode.

Electrode Nodes

The total number of nodes associated with all electrodes in the device structure must be less than

- 1000 for a 3200 node version of Medici
- 2500 for a 10000 node version of Medici
- 5000 for a 20000 node version of Medici.

The **VOID** parameter may be used to reduce the number of nodes associated with electrodes. When this parameter is specified, all interior nodes within a non-zero-thickness electrode are completely removed from the structure.

As an example, the following statement creates an electrode in the corner of a structure and then uses **VOID** to remove all the interior nodes.

```
ELECTROD  NAME=Corner  X.MAX=0.6  Y.MAX=0.6  VOID
```

Adding Electrodes to Existing Structures

An electrode can be added to an existing structure, or added to a structure created by TSUPREM-4 by following the **MESH** statement that reads in the structure with the appropriate **ELECTRODE** statements. For example, the following statements add a source and drain contact to a structure read from TSUPREM-4:

```
MESH      IN.FILE=TS4FILE  TSUPREM4
ELECTROD  NAME=Source  INTERFAC  X.MAX=0.5
ELECTROD  NAME=Drain  INTERFAC  X.MAX=2.5
```

RENAME

Changes the name of an electrode or region.

RENAME

(ELECTROD | REGION | T.ELECTR) OLDNAME=<c> NEWNAME=<c>

Parameter	Type	Definition	Default
ELECTROD	logical	Specifies that an electrical electrode is renamed.	false
REGION	logical	Specifies that a region is renamed.	false
T.ELECTR	logical	Specifies that a thermal electrode is renamed.	false
OLDNAME	char	Specifies the old name of the electrode or region.	none
NEWNAME	char	Specifies the new name for the electrode or region.	none

Description

Usually **RENAME** is used in order to provide meaningful names to electrodes read from a TSUPREM-4 file.

See Also... To further illustrate the **RENAME** statement, refer to input file *mdex9b* in Interface Examples, Chapter 10, "Medici Simulation" on page 10-16.

Examples

Suppose that an existing device structure (for example read in from TSUPREM-4 as a TIF file) contains an electrical electrode "ALUMINUM1" which you would like to change to "GATE". The following statements could be used:

```
MESH      IN.FILE=SOME_FILE.TIF  TIF
RENAME    ELECTROD  OLDNAME=ALUMINUM1  NEWNAME=GATE
SYMBOL    ......
SOLVE     V(GATE)=3 ......
```

PROFILE

The **PROFILE** statement defines profiles for impurities and other quantities to be used in the device structure.

```
PROFILE

   [REGION=<c>]
   [X.MIN=<n>] [ {WIDTH=<n> | X.MAX=<n>} ]
   [Y.MIN=<n>] [ {DEPTH=<n> | Y.MAX=<n>} ]

 Output Doping File
   [OUT.FILE=<c>]


 Uniform Profile
   {  ( UNIFORM {N-TYPE | P-TYPE | IMPURITY=<c> | OTHER=<c>} N.PEAK=<n> )

 Analytic Profiles
    | ( {N-TYPE | P-TYPE | IMPURITY=<c> | OTHER=<c>} {N.PEAK=<n> | DOSE=<n>}
        {Y.CHAR=<n> | Y.JUNCTI=<n>} {X.CHAR=<n> | XY.RATIO=<n>} [X.ERFC]
      )

 Analytic Polygonal Profiles
    | ( POLYGON {N-TYPE | P-TYPE | IMPURITY=<c> | OTHER=<c>} N.PEAK=<n>
        X.POLY=<a> Y.POLY=<a> N.CHAR=<n> [N.ERFC]
      )

 Analytic Rotated Profiles
    | ( ROTATE {N-TYPE | P-TYPE | IMPURITY=<c> | OTHER=<c>} N.PEAK=<n>
        X.CENTER=<n> Y.CENTER=<n> R.INNER=<n> R.OUTER=<n> R.CHAR=<n> [R.ERFC]
      )

 One-Dimensional Profiles from Data Files
    | ( IN.FILE=<c> [N.OFFSET=<n>] [Y.OFFSET=<n>]
        {  ( 1D.PROC [N-TYPE] [P-TYPE] )
         | ( SUPREM2 [N-TYPE] [P-TYPE] )
         | ( 1D.ASCII [Y.COLUMN=<n>]
             {  ( [N.COLUMN=<n>] [P.COLUMN=<n>] )
              | ( D.COLUMN=<n> {IMPURITY=<c> | OTHER=<c>} )
             }
           )
        }
        {X.CHAR=<n> | XY.RATIO=<n>} [X.ERFC]
      )
```

(**PROFILE** statement continued on next page)

(**PROFILE** statement continued from previous page)

```
Two-Dimensional Profiles from Data Files
   | ( IN.FILE=<c> [N.OFFSET=<n>] [X.OFFSET=<n>] [Y.OFFSET=<n>]
       {  ( 2D.PROC [N-TYPE] [P-TYPE] )
        | ( SUPRA [N-TYPE] [P-TYPE] )
        | ( TSUPREM4 [N-TYPE] [P-TYPE] )
        | ( 2D.ASCII [X.COLUMN=<n>] [Y.COLUMN=<n>]
            {  ( [N.COLUMN=<n>] [P.COLUMN=<n>] )
             | ( D.COLUMN=<n> {IMPURITY=<c> | OTHER=<c>} )
            }
          )
        | ( TIF [ {IMPURITY=<c> | ( [N-TYPE] [P-TYPE] )} ]
                [OTHER=<c> [INSULATO] ]
          )
        | ( MEDICI [ {IMPURITY=<c> | ( [N-TYPE] [P-TYPE] )} ] [OTHER=<c>] )
       }
       [X.CHAR=<n>] [X.ERFC] [Y.CHAR=<n>] [Y.ERFC]
     )
  }
```

Parameter	Type	Definition	Default	Units
REGION	char	The names of the regions to which the profile is to be added. If more than one name is given, they should be enclosed within parentheses and separated with commas (for example, "(silicon-1,silicon-2)").	All semiconductor regions.	
X.MIN	number	The minimum x location of the profile. **synonyms: X.LEFT, X.PEAK**	The minimum x location in the structure.	microns
WIDTH	number	The x extent of the profile.	The maximum x location in the structure minus **X.MIN**.	microns
X.MAX	number	The maximum x location of the profile. **synonym: X.RIGHT**	The maximum x location in the structure.	microns
Y.MIN	number	The minimum y location of the profile. **synonyms: Y.TOP, Y.PEAK**	The minimum y location in the structure for input from a file or if **UNIFORM** is specified; otherwise, 0.0.	microns
DEPTH	number	The y extent of the profile.	The maximum y location in the structure minus **Y.MIN** for input from a file or if **UNIFORM** is specified; else 0.0.	microns

Parameter	Type	Definition	Default	Units
Y.MAX	number	The maximum y location of the profile. **synonym: Y.BOTTOM**	The maximum y location in the structure for input from a file or if **UNIFORM** is specified; otherwise, **Y.MIN**.	microns
N-TYPE	logical	Specifies that the impurity profile is n-type (donors). If the **1D.PROC**, **SUPREM2**, **2D.PROC**, **SUPRA**, **TIF**, or **MEDICI** parameter is specified, then the donor impurity profile is read from the data file. If the **TSUPREM4** parameter is specified, the donor impurity profile is reconstructed from the doping in the data file using N(donor)=[N(total)+N(net)]/2. **synonym: N.TYPE**	false	
P-TYPE	logical	Specifies that the impurity profile is p-type (acceptors). If the **1D.PROC**, **SUPREM2**, **2D.PROC**, **SUPRA**, **TIF**, or **MEDICI** parameter is specified, then the acceptor impurity profile is read from the data file. If the **TSUPREM4** parameter is specified, the acceptor impurity profile is reconstructed from the doping in the data file using N(acceptor)=[N(total)-N(net)]/2. **synonym: P.TYPE**	false	
IMPURITY	char	The name of an impurity for which a profile is being defined or read. The chemical name of the impurity should be specified here (such as "B" or "As"). Medici assumes all impurities specified with this parameter are electrically active. If the **TIF** or **MEDICI** parameter is specified, multiple impurities may be read (separate their names with commas and enclose the entire list with parentheses).	none	
OTHER	char	The name of an arbitrary quantity for which a profile is being defined or read. If the **TIF** or **MEDICI** parameter is specified, multiple **OTHER** quantities may be read (separate their names with commas and enclose the entire list with parentheses).	none	
Output Doping File				
OUT.FILE	char	The identifier for the data file to which the profile information is to be written. This file is used only in conjunction with the **REGRID** statement. **synonym: OUTFILE**	none	
Uniform Profile				
UNIFORM	logical	Specifies that the profile has a uniform distribution.	false	
N.PEAK	number	The peak impurity concentration for an impurity profile or the peak value for **OTHER** profiles. **synonym: CONCENTR**	none	$\#/cm^3$ for impurities
Analytic Profiles				
DOSE	number	The dose of the impurity profile assuming a full Gaussian distribution.	none	$\#/cm^2$
Y.CHAR	number	The y characteristic length of the profile outside the range of **Y.MIN** < y < **Y.MAX**. **synonym: CHAR**	0.0002 for data read from a file; otherwise, none	microns

Parameter	Type	Definition	Default	Units
Y.JUNCTI	number	The y location under the center of the profile where the magnitude of the profile being added equals the magnitude of the background profile. **synonym: JUNCTION**	none	microns
X.CHAR	number	The horizontal characteristic length of the profile outside the range **X.MIN** < x < **X.MAX**. **synonym: LAT.CHAR**	0.0002 for data read from a file; otherwise, **XY.RATIO***Y.CHAR,	microns
XY.RATIO	number	The ratio of the horizontal characteristic length to the vertical characteristic length for an analytic profile. If the vertical profile is input from a data file, **XY.RATIO** is the factor which multiplies the extent of the profile when the profile is rotated to the horizontal direction. **synonym: RATIO.LA**	1.0	none
X.ERFC	logical	Specifies that the x variation of the profile is described by a complementary error function. If this parameter is false, then the horizontal variation is uniform from **X.MIN** to **X.MAX** with Gaussian tails outside of this region. **synonym:ERFC.LAT**	false	

Analytic Polygonal Profiles

POLYGON	logical	Specifies that the profile is a polygon. The polygon vertices are defined by a pair of arrays: **X.POLY** and **Y.POLY**.	false	
X.POLY	array	Specifies an array of x coordinates of the polygon vertices. The number of the vertices should not exceed 40.	none	microns
Y.POLY	array	Specifies an array of y coordinates of the polygon vertices. The number of the vertices should not exceed 40.	none	microns
N.CHAR	number	The characteristic length of the profile outside of the polygon. **synonym: R.CHAR**	none	microns
N.ERFC	logical	Specifies that the lateral variation of the polygonal profile outside of the polygon is described by a complimentary error function. **synonym: R.ERFC**	false	

Analytic Rotated Profiles

ROTATE	logical	Specifies that the profile is rotated around a rotation center.	false	
X.CENTER	number	Specifies the x location of the rotation center.	none	microns
Y.CENTER	number	Specifies the y location of the rotation center.	none	microns
R.INNER	number	The inner radius (distance from the rotation center) of a circular profile.	none	microns
R.OUTER	number	The outer radius (distance from the rotation center) of a circular profile.	none	microns
R.CHAR	number	The characteristic length of the profile outside of the **R.INNER** to **R.OUTER** rotation range. **synonym: N.CHAR**	none	microns

Parameter	Type	Definition	Default	Units
R.ERFC	logical	Specifies that the radial variation of the rotated profile outside of the **R.INNER** to **R.OUTER** range is described by a complimentary error function. **synonym: N.ERFC**	false	
One-Dimensional Profiles from Data Files				
IN.FILE	char	The identifier for the data file containing the profile. **synonyms: INFILE** and **FILE**	none	
N.OFFSET	number	The concentration to be subtracted from both the donor and acceptor impurity profiles obtained from a data file.	0.0	atoms/ cm^3
Y.OFFSET	number	The y direction shift of a profile obtained from a data file.	0.0	microns
1D.PROC	logical	Specifies that the data file containing an impurity profile was generated by TMA SUPREM-3.	false	
SUPREM2	logical	Specifies that the data file containing an impurity profile was generated by SUPREM-2.	false	
1D.ASCII	logical	Specifies that profile data is read from a formatted file. One column of the file should correspond to depth (in microns) and should be identified with **Y.COLUMN**. The file should also contain one or more columns of profile data that can be identified with the parameters **N.COLUMN**, **P.COLUMN**, or **D.COLUMN**.	false	
Y.COLUMN	number	The column of the formatted data file corresponding to the y coordinate.	1 if **1D.ASCII** is specified; 2 if **2D.ASCII** is specified.	none
N.COLUMN	number	The column of the formatted data file corresponding to net donor impurity concentration. Data with values less than zero are considered to correspond to net acceptor impurity concentration.	None if **P.COLUMN** is specified; otherwise, 2 if **1D.ASCII** is specified and 3 if **2D.ASCII** is specified.	none
P.COLUMN	number	The column of the formatted data file corresponding to net acceptor impurity concentration. Data with values less than zero are considered to correspond to net donor impurity concentration.	none	none
D.COLUMN	number	The column of the formatted data file corresponding to either **IMPURITY** or **OTHER** profile values.	none	none
Two-Dimensional Profiles from Data Files				
X.OFFSET	number	The x-direction shift of a two-dimensional profile obtained from a data file.	0.0	microns
2D.PROC	logical	Specifies that the data file is in a format generated by TMA SUPRA with a revision code of 8501 or later. This parameter is used to read a data that was saved by using TMA SUPRA's **SAVEFILE** statement with the **DEVICE** parameter specified.	false	
SUPRA	logical	Specifies that the data file is in a format generated by a version of TMA SUPRA with a revision code earlier than 8501.	false	

Parameter	Type	Definition	Default	Units
TSUPREM4	logical	Specifies that the data file is in a format generated by TSUPREM-4. This parameter is used to read a data that was saved by using TSUPREM-4's **SAVEFILE** statement with the **MEDICI** parameter specified.	false	
2D.ASCII	logical	Specifies that profile data is read from a formatted file. Two columns of the file should correspond to the x- and y coordinates (in microns) and should be identified with **X.COLUMN** and **Y.COLUMN**, respectively. The file should also contain one or more columns of profile data that can be identified with the parameters **N.COLUMN**, **P.COLUMN**, or **D.COLUMN**.	false	
X.COLUMN	number	The column of the formatted data file corresponding to the x coordinate.	1 if **2D.ASCII** is specified	none
TIF	logical	Specifies that the data file is in TIF.	false	
INSULATO	logical	Specifies that when an **OTHER** quantity is read from a TIF file at an interface node, the value associated with the insulator material should be read instead of the value associated with the semiconductor material.	false	
MEDICI	logical	Specifies that the data file is a standard Medici mesh file.	false	

Description

The **PROFILE** statement can be used to describe both impurity profiles and other arbitrary two-dimensional profiles for the structure. Profiles may be defined either analytically or through input from a data file. The data file can be a formatted file containing columns of data representing coordinates and values, the output generated by a process simulation program, or a Medici mesh file containing profile information.

See Also...

To further illustrate the **PROFILE** statement, refer to input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Impurity Profiles" on page 4-5.

Profile Types

The profile type is specified with one of the parameters **N-TYPE**, **P-TYPE**, **IMPURITY**, or **OTHER**.

The parameters **N-TYPE** and **P-TYPE** are used with impurity profiles and merely classify them as either donors or acceptors.

Specifying Individual Impurity Profiles

The **IMPURITY** parameter is used to specify individual impurities for the structure (such as B or As). This parameter can be used to specify the name of an impurity that is being defined analytically or the names of the impurities to be read from ASCII, Medici, or TIF files. The use of the **IMPURITY** parameter is illustrated in the following examples.

Define an analytic boron profile:

```
PROFILE  IMPURITY=B N.PEAK=1E18 Y.MIN=0 Y.MAX=0
+        Y.CHAR=.2 X.MIN=0 X.MAX=0.5 X.CHAR=0.2
```

Read an arsenic profile from a 2D ASCII file:

```
PROFILE  IMPURITY=As IN.FILE=cool_data 2D.ASCII
+        X.COLUMN=1 Y.COLUMN=2 D.COLUMN=3
```

Read multiple profiles from a TIF file:

```
PROFILE  IMPURITY=(B,As,P,In) IN.FILE=cool.tif TIF
```

Note:

When using the* IMPURITY *parameter, impurities should be specified with their chemical name. All impurities specified with* IMPURITY *are assumed to be electrically active and their names are stored internally with the letter "a" appended to their chemical name. When reading impurities from TIF files, specifying* IMPURITY=(B,As,P,...) *will cause the program to look for* Ba, Asa, Pa,... *in the TIF file.

Specifying Profiles for Other Quantities

The **OTHER** parameter is used to specify spatial distributions of arbitrary quantities other than electrically active impurities. As with impurities, **OTHER** quantities can be defined analytically or read from ASCII, Medici, or TIF files. Some uses of the **OTHER** parameter are illustrated in the following examples.

Define an analytic density of states for use with the **TRAP** statement:

```
PROFILE  OTHER=DOS N.PEAK=1E19 Y.MIN=0 Y.MAX=0
+        Y.CHAR=.1
```

Read a lifetime profile from a 2D ASCII file for use on **EXTRACT** or **TRAP** statements:

```
PROFILE  OTHER=Lifetime IN.FILE=life.dat 2D.ASCII
+        X.COLUMN=1 Y.COLUMN=2 D.COLUMN=3
```

Read the chemical boron concentration from a TIF file:

```
PROFILE  OTHER=B IN.FILE=stuff.tif TIF
```

Read the stress components from a TIF file:

```
PROFILE  OTHER=(Sxx,Syy,Sxy) IN.FILE=stuff.tif TIF
```

Analytic Profiles

Specifying **UNIFORM** creates a uniform profile of concentration **N.PEAK**. Otherwise, the profile has a constant concentration of **N.PEAK** only in the region **X.MIN** < x < **X.MAX** and **Y.MIN** < y < **Y.MAX**. Outside this region, the profile varies vertically as a Gaussian with a characteristic length of **Y.CHAR**.

The horizontal profile outside this region can either vary as a Gaussian (the default) or as the difference of two complementary error functions (if **X.ERFC** is

specified). In either case, a characteristic length of **X.CHAR** or **XY.RATIO***Y.CHAR is used in the x direction.

The mathematical description of an analytic profile is given by

$$N(x,y) = \mathbf{N.PEAK} \cdot a(x) \cdot b(y) \qquad \text{Equation 3-3}$$

Vertical Variation

The function $b(y)$ describes the vertical variation of the profile and is given by

$$b(y) = \begin{cases} \exp\left[-\left(\dfrac{y-\mathbf{Y.MIN}}{\mathbf{Y.CHAR}}\right)^2\right] & y < \mathbf{Y.MIN} \\ 1 & \mathbf{Y.MIN} \le y \le \mathbf{Y.MAX} \\ \exp\left[-\left(\dfrac{y-\mathbf{Y.MAX}}{\mathbf{Y.CHAR}}\right)^2\right] & y > \mathbf{Y.MAX} \end{cases} \qquad \text{Equation 3-4}$$

Lateral Direction Variation

The function $a(x)$ describes the lateral variation of the profile and is given by

$$a(x) = \begin{cases} a_1(x) & \mathbf{X.ERFC} \text{ not specified} \\ a_2(x) & \mathbf{X.ERFC} \text{ specified} \end{cases} \qquad \text{Equation 3-5}$$

where

$$a_1(x) = \begin{cases} \exp\left[-\left(\dfrac{x-\mathbf{X.MIN}}{\mathbf{X.CHAR}}\right)^2\right] & x < \mathbf{X.MIN} \\ 1 & \mathbf{X.MIN} \le x \le \mathbf{X.MAX} \\ \exp\left[-\left(\dfrac{x-\mathbf{X.MAX}}{\mathbf{X.CHAR}}\right)^2\right] & x > \mathbf{X.MAX} \end{cases} \qquad \text{Equation 3-6}$$

and

$$a_2(x) = \frac{\operatorname{erfc}\left(\dfrac{x-\mathbf{X.MAX}}{\mathbf{X.CHAR}}\right) - \operatorname{erfc}\left(\dfrac{x-\mathbf{X.MIN}}{\mathbf{X.CHAR}}\right)}{2} \qquad \text{Equation 3-7}$$

Junction Depth

The **Y.JUNCTI** parameter may be used to specify the p-n junction depth as a y location where the magnitude of the impurity profile equals the magnitude of the current background impurity concentration.

Y.JUNCTI is used to define the vertical characteristic length which otherwise must be specified with the **Y.CHAR** parameter.

Impurity Dose

The **DOSE** parameter may be used to specify the integral amount of doping atoms which relates to the peak concentration in terms of the vertical characteristic length as **N.PEAK** = **DOSE**$/(\sqrt{\pi} \cdot$ **Y.CHAR**$)$.

Polygonal Profiles

A polygon can be defined in the XY plane for describing an area where the profile has a constant value of **N.PEAK**. The polygon is defined by a pair of coordinate arrays **X.POLY** and **Y.POLY** representing the location of vertices of the polygon. The polygon can be arbitrary as long as it is non-self-intersecting. The last vertex may or may not coincide with the first one, either way will do.

Lateral profile extension outside of the polygon is determined by the parameters **N.CHAR** and **N.ERFC**. These parameters are applied to the lateral profile extension in the direction, normal to the nearest polygon edge.

Outside of the profile bounding box (**X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX**) there is no lateral profile extension for a polygonal profile.

Profile Rotation

A profile can also be described by a rotation in the XY plane around a rotation center, defined by the center location coordinates **X.CENTER** and **Y.CENTER**. When using a rotated profile, the profile has a constant value of **N.PEAK** within the range **R.INNER** to **R.OUTER**. If **R.INNER** is zero, then a circular profile is created, otherwise, a donut-shaped profile is created.

Lateral profile extension outside of the range **R.INNER** to **R.OUTER** is determined by the parameters **R.CHAR** and **R.ERFC**. These parameters are applied to the lateral profile extension in the radial direction.

Outside of the profile bounding box the lateral profile extension is defined by the conventional parameters **X.CHAR**, **X.ERFC**, and **Y.CHAR**.

A combination of the profile rotation and profile bounding box (**X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX**) can be used in order to form segments of a circular profile.

One-Dimensional Profiles from Data Files

The vertical impurity profile may be input from a data file by specifying the **IN.FILE** parameter and one parameter from the set **1D.PROC**, **SUPREM2**, or **1D.ASCII**. By default, the origin for the impurity profile is aligned with the vertical origin in Medici.

For one-dimensional impurity profiles input from data files created by a process simulation program, the origin occurs at the first point of the bottom semiconductor material (polysilicon and silicon are treated as the same material). The **Y.OFFSET** parameter may be used to shift the vertical profile relative to the Medici structure.

Vertical Range

The **Y.MIN** and **Y.MAX** parameters define the y interval within which the impurity profiles contribute to the total impurity distribution for the structure. The impurity profiles are considered to drop to zero concentration outside of this y interval.

When **Y.MIN** and **Y.MAX** are not specified and the Medici structure has a greater y extent than the input profile, the program extends the input profile. This is done by setting the concentration at points of the structure outside the vertical interval of the input profile to be equal to the concentration at the top and bottom points of the input profile.

Lateral Range

For a particular vertical location, the profile has a constant concentration in the horizontal direction in the range **X.MIN** < x < **X.MAX**. The horizontal variation outside of this range can vary as a Gaussian (the default) or as the difference of two complementary error functions (if **X.ERFC** is specified). In either case, the characteristic length is specified with **X.CHAR**.

Rotational Profile Extension

The **XY.RATIO** parameter may be used to rotate the vertical profile about axes perpendicular to the simulation plane and located at x coordinates of **X.MIN** and **X.MAX**. The y coordinate for each rotation axis corresponds to the top of the impurity profile input from the data file. This coordinate is regardless of what portion is actually used in Medici (**Y.MIN** and **Y.MAX** parameters) or how it is shifted (**Y.OFFSET** parameter). During the rotation, the extent of the profile in the horizontal direction is multiplied by the factor specified with **XY.RATIO**.

Choosing Profiles to Input

The choice of which profiles to input from the file is made by specifying either, both, or none of the parameters **N-TYPE** and **P-TYPE**. If neither parameter is specified, then both the donor and acceptor impurity profiles are read from the data file.

If the impurity profiles are taken from a formatted file, the parameters **N.COLUMN** and **P.COLUMN** are used to identify columns of data containing net donor and/or net acceptor impurity concentration. The **D.COLUMN** parameter can be used in conjunction with the **IMPURITY** or **OTHER** parameter to read individual impurity profiles or other spatial distributions from the file.

As an example, the following **PROFILE** statements read in impurity profiles generated by TMA SUPREM-3 in order to specify the doping for an N-channel MOSFET:

```
PROFILE   P-TYPE 1D.PROC IN.FILE=CHANNEL
PROFILE   N-TYPE 1D.PROC IN.FILE=SRCDRN X.MIN=0 WIDTH=1
+         XY.RATIO=.8
PROFILE   N-TYPE 1D.PROC IN.FILE=SRCDRN  X.MIN=3 WIDTH=1
+         XY.RATIO=.8
```

The first statement reads the p-type profile from a file named *CHANNEL* and by default places this over the entire width of the device. The second and third statements read the n-type profile from a file named *SRCDRN*. These statements define the source and drain for the device.

As a further example, the following statement reads both n-type and p-type impurity profiles from a formatted data file and places them over the entire device:

```
PROFILE  1D.ASCII  IN.FILE=MYDATA  Y.COL=1  N.COL=2  P.COL=3
```

Two-Dimensional Profiles From Data Files

The entire two-dimensional impurity profile may be input from a data file by specifying the **IN.FILE** parameter and one parameter from the set **2D.PROC**, **SUPRA**, **TSUPREM4**, **2D.ASCII**, **TIF**, or **MEDICI**.

By default the origin for the impurity profile is aligned with the origin in Medici. The **X.OFFSET** and **Y.OFFSET** parameters may be used to shift the two-dimensional impurity profile relative to Medici structure.

Vertical and Lateral Ranges

The **X.MIN** and **X.MAX** parameters and the **Y.MIN** and **Y.MAX** parameters define the x and y intervals, respectively, of the profile definition range within which the impurity profiles contribute to the impurity distribution for the structure. Outside of these intervals the profile decays as a Gaussian with characteristic lengths of **X.CHAR** and **Y.CHAR** for horizontal and vertical directions, respectively. If **X.ERFC** and/or **Y.ERFC** are specified, then an error function is used to extend the profiles outside the profile definition range in the horizontal and vertical directions, respectively.

When **X.MIN**, **X.MAX**, **Y.MIN**, or **Y.MAX** is not specified and the structure has a greater x or y extent than the input profile, the input profile is extended by setting the concentration at points of the structure outside the x or y intervals. The profile is extended by the input profile to be equal to the concentration at the top, bottom, left, and right edges of the input profile.

Choosing Profiles to Input

The choice of which impurity profiles to input from the file is made by specifying either, both, or neither of the parameters **N-TYPE** and **P-TYPE**. If neither parameter is specified, then both the donor and acceptor impurity profiles are read from the data file. For **2D.ASCII**, **TIF**, and **MEDICI** files, the choice of profiles to read can also be made by using the **IMPURITY** and **OTHER** parameters.

When the impurity profiles are taken from a formatted file, the parameters **N.COLUMN**, **P.COLUMN**, or **D.COLUMN** are used to identify columns of data containing net donor impurity concentration, net acceptor impurity concentration, or profiles identified with the **IMPURITY** or **OTHER** parameters.

As an example, the following **PROFILE** statement reads in an entire two-dimensional impurity profile generated by TSUPREM-4 from a file named *LDDSTRUC*:

```
PROFILE   TSUPREM4  IN.FILE=LDDSTRUC
```

REGRID

The **REGRID** statement allows refinement of a coarse mesh.

```
REGRID

 Regrid Criteria
   {  POTENTIA | ( E.FIELD [ {X.COMPON | Y.COMPON} ] ) | QFN | QFP
    | DOPING | ELECTRON | HOLES | NET.CHAR | NET.CARR
    | ( MIN.CARR [LOCALDOP] ) | II.GENER | BB.GENER | PHOTOGEN
    | ELE.TEMP | HOL.TEMP | TRUNC | ARRAY1 | ARRAY2 | ARRAY3
    | IMPURITY=<c> | OTHER=<c>

 Lattice Temperature AAM Parameters
    | LAT.TEMP
   }

 Regrid Controls
   (RATIO=<n> | FACTOR=<n>) [IN.FILE=<c>]
   [CHANGE] [ABSOLUTE] [LOGARITH] [MAX.LEVE=<n>] [SMOOTH.K=<n>]
   [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
   [REGION=<c>] [IGNORE=<c>] [COS.ANGL=<n>]
   [ OUT.FILE=<c> [NO.TTINF] [ASCII] ]

 Quadtree Regrid Controls
   [BOUNDARY]
   [MAXDEL=<n>] [MAXDEL.X=<n>] [MAXDEL.Y=<n>]
   [MINDEL=<n>] [MINDEL.X=<n>] [MINDEL.Y=<n>]
   [ASINH] [UNREFINE=<n>]
```

Parameter	Type	Definition	Default	Units
Regrid Criteria				
POTENTIA	logical	Specifies that the grid refinement is based on mid-gap potential.	false	
E.FIELD	logical	Specifies that the grid refinement is based on the magnitude of electric field.	false	
X.COMPON	logical	Specifies that the grid refinement is based on the x-component of electric field instead of the magnitude.	false	
Y.COMPON	logical	Specifies that the grid refinement is based on the y component of electric field instead of the magnitude.	false	
QFN	logical	Specifies that the grid refinement is based on electron quasi-Fermi potential.	false	
QFP	logical	Specifies that the grid refinement is based on hole quasi-Fermi potential.	false	
DOPING	logical	Specifies that the grid refinement is based on net impurity concentration.	false	

Parameter	Type	Definition	Default	Units
ELECTRON	logical	Specifies that the grid refinement is based on electron concentration.	false	
HOLES	logical	Specifies that the grid refinement is based on hole concentration.	false	
NET.CHAR	logical	Specifies that the grid refinement is based on net charge concentration.	false	
NET.CARR	logical	Specifies that the grid refinement is based on net carrier concentration.	false	
MIN.CARR	logical	Specifies that the grid refinement is based on minority carrier concentration.	false	
LOCALDOP	logical	Specifies that when **MIN.CARR** is specified, refinement takes place when the minority carrier concentration exceeds the local doping. This is the only case where **RATIO** does not have to be specified.	false	
II.GENER	logical	Specifies that the grid refinement is based on the total generation rate due to impact ionization.	false	
BB.GENER	logical	Specifies that the grid refinement is based on the total generation rate due to band-to-band tunneling.	false	
PHOTOGEN	logical	Specifies that the grid refinement is based on total photogeneration.	false	
ELE.TEMP	logical	Specifies that the grid refinement is based on the electron temperature.	false	
HOL.TEMP	logical	Specifies that the grid refinement is based on the hole temperature.	false	
TRUNC	logical	Specifies that grid refinement is based on the truncation error for the Poisson equation.	false	
ARRAY1	logical	Specifies that grid refinement is based on the quantity stored in the user calculated **ARRAY1**.	false	
ARRAY2	logical	Specifies that grid refinement is based on the quantity stored in the user calculated **ARRAY2**.	false	
ARRAY3	logical	Specifies that grid refinement is based on the quantity stored in the user calculated **ARRAY3**.	false	
IMPURITY	char	The name of an impurity to use as the basis for grid refinement.	none	
OTHER	char	The name of an **OTHER** quantity to use as the basis for grid refinement.	none	
Lattice Temperature AAM Parameters				
LAT.TEMP	logical	Specifies that the grid refinement is based on lattice temperature. This parameter is only used with the Lattice Temperature AAM.	false	

Parameter	Type	Definition	Default	Units
Regrid Controls				
RATIO	number	The numerical criterion for refining a triangle. If the specified quantity differs by more than **RATIO** at the nodes of a triangle, the triangle is refined. If **^CHANGE** is specified, the triangle is refined if the magnitude of the specified quantity exceeds **RATIO** at each node of the triangle. If **LOGARITH** is specified, then **RATIO** should be specified as a logarithmic value. **synonym: STEP**	none	None if **LOGARITH** is specified; otherwise volts, volts/cm, #/cm^3, or #/cm^3/sec.
FACTOR	number	Numerical factor by which to increase the number of grid points by. If **FACTOR** is specified the program automatically determines the refinement criteria (**RATIO**) required to produce a new grid with a total number of grid points equal to **FACTOR** multiplied by the present number of grid points	none	none
IN.FILE	char	The identifier for the file which contains the doping for the device. Specifying **IN.FILE** avoids interpolating doping values at any newly created grid points (the default), by using the initial doping specification to redope the structure. **synonym: DOPFILE**	none	
CHANGE	logical	Specifies that the difference of the specified quantity at the nodes of a triangle is used as the criterion for refinement. If **^CHANGE** is specified, the magnitude of the specified quantity at the nodes of a triangle is used as the criterion for refinement.	true	
ABSOLUTE	logical	Specifies that refinement is based on the absolute value of the specified quantity.	false	
LOGARITH	logical	Specifies that refinement is based on the logarithm of the specified quantity. Since many of the quantities may become negative, the program actually uses sign(x) * log(1 + |x|) to avoid overflow. To get refinement on the true logarithm of a quantity, specify **ABSOLUTE** and **LOGARITH**. Absolute value is taken first and there is no danger of negative arguments.	false	
MAX.LEVE	number	Specifies the maximum level of any refined triangle relative to the original mesh	One more than the maximum level of the grid.	none
SMOOTH.K	number	Specifies a method for mesh smoothing. **SMOOTH.K**=1 indicates triangular smoothing is used, maintaining all region boundaries fixed. **SMOOTH.K**=2 indicates triangular smoothing is used, maintaining only material boundaries	none	none
X.MIN	number	The minimum x coordinate of the area to be refined.	The minimum x location in the device structure.	microns
X.MAX	number	The maximum x coordinate of the area to be refined.	The maximum x location in the device structure.	microns
Y.MIN	number	The minimum y coordinate of the area to be refined.	The minimum y location in the device structure.	microns

Parameter	Type	Definition	Default	Units
Y.MAX	number	The maximum y coordinate of the area to be refined.	The maximum y location in the device structure.	microns
REGION	char	The names of the regions over which refinement takes place. Elements in other regions may be refined as a side effect to maintain well-shaped triangles. Multiple regions may be specified by enclosing the names in parentheses and separating them with commas (for example, "(Body,Oxide,Gate)").	all regions	
IGNORE	char	The names of the regions over which no refinement takes place. Ignored regions are not refined either according to the user criterion nor according to the "obtuse criterion" (see **COS.ANGL**). Ignored regions are also not smoothed after regrids. Multiple regions may be specified by enclosing the names in parentheses and separating them with commas (for example, "(Body,Oxide,Gate)").	none	
COS.ANGL	number	The "obtuse criterion" to limit the creation of obtuse triangles in the mesh. If a regrid would create a triangle with an angle whose cosine is less than **-COS.ANGL**, then nodes are added so that this does not occur. The test may be turned off locally by using the **IGNORE** parameter. It can be turned off everywhere by using a value of **COS.ANGL** greater than 1.	2.0	none
OUT.FILE	char	The identifier for the binary output file to store the refined mesh. **synonym: OUTFILE**	none	
NO.TTINF	logical	Specifies that triangle tree information is not written to the output mesh specified by **OUT.FILE**. This option is not recommended if further grid refinement is to be performed.	false	
ASCII	logical	Specifies that the refined mesh should be stored in a formatted file instead of a binary file.	false	

Quadtree Regrid Controls

BOUNDARY	char	Specifies the name of the region whose boundaries should be refined.	none	
MAXDEL	number	Specifies the maximum allowed grid spacing in any direction.	none	microns
MAXDEL.X	number	Specifies the maximum allowed grid spacing in the x direction.	none	microns
MAXDEL.Y	number	Specifies the maximum allowed grid spacing in the y direction.	none	microns
MINDEL	number	Specifies the minimum allowed grid spacing in any direction.	none	microns
MINDEL.X	number	Specifies the minimum allowed grid spacing in the x direction.	none	microns
MINDEL.Y	number	Specifies the minimum allowed grid spacing in the y direction.	none	microns
ASINH	logical	Specifies that regridding should occur on the asinh of the criterion	false	
UNREFINE	number	Specifies that unrefinement should occur with the given factor.	none	microns

Description

The **REGRID** statement allows coarse meshes to be refined based on where the physical properties of the device structure require it. A triangle is refined if the value of the chosen criterion differs by more than a specified tolerance (**RATIO**) at the nodes of the triangle. Alternatively, by selecting **^CHANGE**, the triangle can be refined if the magnitude of the chosen criterion exceeds a specified value.

See Also...

To further illustrate the **REGRID** statement, refer to input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Grid Refinement" on page 4-6.

Note:

If the value of the specified criteria ranges over several orders of magnitude, it is advisable to base the refinement on the logarithm of the value by selecting the* LOGARITH *parameter.

Grid Refinement

This section details the procedures used to control and manipulate the regrid process.

Refinement Levels

An initial mesh consists of all level 0 triangles: no regridding has been performed. When a triangle is found that satisfies the user-specified refinement criterion, it is subdivided into four congruent triangles. These are now level 1 triangles and the various grid quantities are interpolated onto the new nodes.

After all level 0 triangles have been examined and refined if needed, the same procedure is applied in turn to level 1 triangles. Refinement of level 1 triangles creates level 2 triangles, and so on.

When a **REGRID** statement is encountered, the default maximum level of refinement is one level higher than the highest existing level before refinement begins. It is often advisable to specify the maximum level of refinement with the **MAX.LEVE** parameter.

If a mesh has already been refined several times, and it is only desired to refine a coarse part of the mesh without regridding the finer regions, then **MAX.LEVE** should be set below the level of the finer regions.

When several levels of regrid are performed in immediate succession, the refinement decisions at the higher levels are made using interpolated data. Because of the nonlinearity of semiconductor problems, this can result in too many nodes in areas where they are not required, and not enough where they are needed.

Note:

It is recommended that only one level of refinement be performed with each* REGRID *statement. A* SOLVE *should be performed before the next* REGRID *statement.

Using the FACTOR Option

The **FACTOR** parameter gives a convenient way to increase the size of the mesh automatically without having to guess at the proper regridding criterion (**RATIO**). The user specifies the appropriate regrid quantity and the **FACTOR** by which to increase the grid, for example:

```
REGRID  POTENTIAL FACTOR=1.2 SMOOTH=1
```

Note:

***If the old grid contained 1000 grid points, the new grid produced by this regrid operation contains approximately 1200 grid points. Due to the complicated constraints on the gridding process, the exact size of the mesh cannot be predetermined so the actual increase in grid will differ slightly from the specified* FACTOR.**

Other Notes

Some other regrid considerations include:

- If performing a regrid that requires solution information, a **SOLVE** should be performed before the next regrid.
- If performing a **DOPING** regrid, the original doping specification should be read using the **IN.FILE** parameter to improve the accuracy of doping interpolation onto the refined mesh.
- The refined mesh can be smoothed using the **SMOOTH.K** parameter. Refer to the **MESH** statement for a discussion of smoothing.

Regridding a Quadtree Mesh

If a quadtree mesh has been created using the **QUADTREE** parameter on the **MESH** statement, anisotropic refinement can be performed on the mesh. The spacing limits in the x and y direction can be specified using the **MAXDEL.X**, **MAXDEL.Y**, **MINDEL.X**, and **MINDEL.Y** parameters. If equal spacing limits in the x and y directions are desired, the **MAXDEL** and **MINDEL** parameters can be used as a shorthand. As an alternative to regridding on the **LOG** of a quantity, regridding on the **ASINH** of a quantity can be specified. This is very useful for regridding on the doping. Regrids on a quadtree mesh are always on the change in a quantity, equivalent to specifying the **CHANGE** parameter. The current quantities that can serve as regridding criteria are: **DOPING**, **POTENTIAL**, **ELECTRONS**, **HOLES**, and **BOUNDARY**. The **BOUNDARY** parameter is used to specify the name of a region whose boundaries should be refined. When regridding a quadtree mesh, multiple regrids may be performed at the same time by specifying multiple **REGRID** statements. Unrefinement can be performed by specifying the unrefine factor, **UNREFINE**, to be greater than 2. The other **REGRID** parameters that can be used during quadtree regridding are: **X.MIN**, **X.MAX**, **Y.MIN**, **Y.MAX**, **REGION**, **IGNORE**, **COS.ANGL**, **OUT.FILE**, and **ASCII**. These parameters are described above.

See Also... To further illustrate the **REGRID** statement as applied to a quadtree mesh, refer to input file *mdex1qt* in N-Channel MOSFET Examples, Chapter 4, "Grid Refinement" on page 4-6.

STITCH

The **STITCH** statement reads a previously generated structure from a file and merges it with the currently initiated structure.

```
STITCH

   IN.FILE=<c> [ASCII.IN] [ {TIF | TSUPREM4} [POLY.ELE] ]
   { TOP | BOTTOM | LEFT | RIGHT } [X.OFFSET=<n>] [Y.OFFSET=<n>]
   [FLIP.X] [FLIP.Y] [ELEC.MER] [REG.MERG]
```

Parameter	Type	Definition	Default	Units
IN.FILE	char	The identifier for the file containing a previously-generated structure. Unless **TIF** or **TSUPREM4** is specified, the file is assumed to be in standard Medici format.	none	
ASCII.IN	logical	Specifies that the input data file is formatted.	false	
TIF	logical	Specifies that the input data file is in TIF format.	false	
TSUPREM4	logical	Specifies that the input data file was generated by TSUPREM-4 for Medici.	false	
POLY.ELE	logical	Specifies that polysilicon regions in the input data file are converted to electrodes.	false	
TOP	logical	Specifies that the appended structure is stitched to the top of the existing device structure (minimum y coordinate).	false	
BOTTOM	logical	Specifies the appended structure is stitched to the bottom of the existing device structure (maximum y coordinate).	false	
LEFT	logical	Specifies the appended structure is stitched to the left side of the existing device structure (minimum x coordinate).	false	
RIGHT	logical	Specifies the appended structure is stitched to the right side of the existing device structure (maximum coordinate).	false	
X.OFFSET	number	Specifies an additional horizontal shift of the input structure before stitching	0	microns
Y.OFFSET	number	Specifies an additional vertical shift of the input structure before stitching.	0	microns
FLIP.X	logical	Specifies that the input structure is flipped horizontally before stitching.	false	
FLIP.Y	logical	Specifies that the input structure is flipped vertically before stitching.	false	
ELEC.MER	logical	Specifies that electrodes with the same name in the existing structure and in the input structure are merged. If **ELEC.MER** is false, electrodes from the input structure will be given unique names.	false	
REG.MERG	logical	Specifies that regions with the same name and material type in the existing structure and in the input structure are merged. If **REG.MER** is false, regions from the input structure will be given unique names.	true	

STITCH Statement Usage

A **STITCH** statement specifies that a structure stored in a file should be appended to the existing device structure. You can include several **STITCH** statements in a

single input file to stitch multiple structures together. Stitched structures can come from Medici files, TSUPREM-4 files (stored in Medici format), and TIF files.

One of the parameters, **TOP**, **BOTTOM**, **LEFT**, or **RIGHT** must be specified to designate where the appended structure is to be stitched relative to the existing structure. The **X.OFFSET** and **Y.OFFSET** parameters are used to specify additional shifts of the appended structure prior to stitching. Stitching structures that overlap is not allowed.

When a structure is stitched to an existing structure, regions of the same material type that are in contact with each other will be merged into a single region. Similarly, electrodes that are in contact with each other will be merged into a single electrode. The names of the merged regions and electrodes will be those of the initial structure. The **ELEC.MER** and **REG.MERG** parameters are used to control the merging of electrodes and regions that are not in contact with each other.

As an example, Figure 3-16 shows an initial structure and a structure to be appended (stitched) to the initial structure. The stitching is accomplished by the statements:

```
MESH      IN.FILE=INITIAL.MSH
STITCH    IN.FILE=APPEND.MSH  BOTTOM  FLIP.Y
+         X.OFFSET=-0.5
```

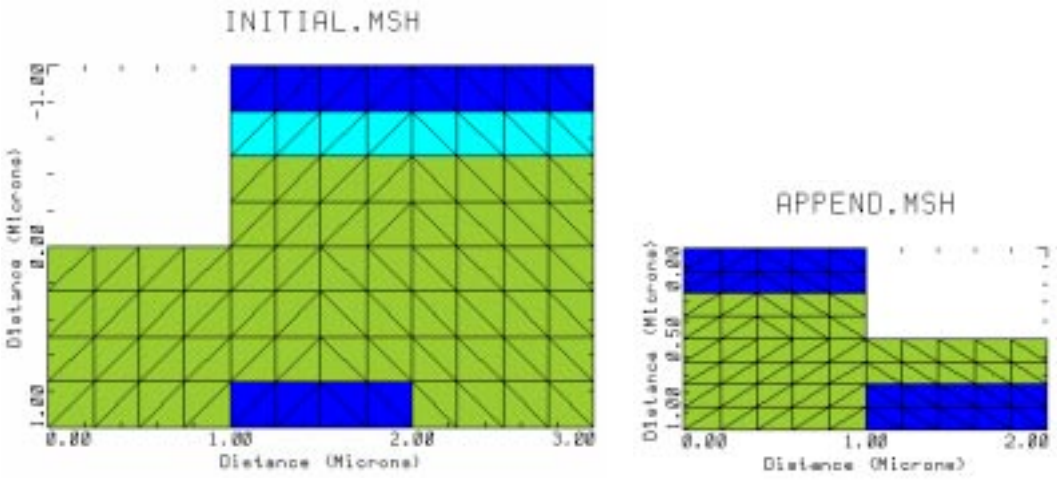


Figure 3-16 Initial structure and structure to be stitched to it

The appended structure is stored in the file *APPEND.MSH*. The **STITCH** statement says to vertically flip the structure stored in this file and attach it to the bottom of the existing structure. The appended structure is also shifted to the left by 0.5 micron by specifying **X.OFFSET=-0.5**. The final structure is shown in Figure 3-17.

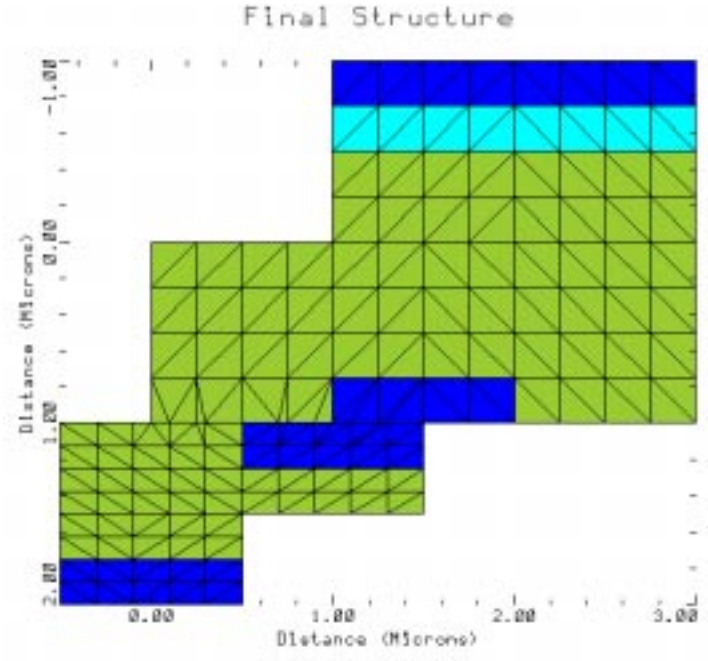


Figure 3-17 The final structure

3.2 Solution Specification

The following statements specify how solutions are generated by Medici:

Statement	Definition	Page
MODELS	Enables the use of physical models during solutions.	3-106
PHOTOGEN	Models the spatial and temporal generation of electron-hole pairs in a device.	3-114
TRAPS	Includes traps in the device materials.	3-130
SYMBOLIC	Performs a symbolic factorization.	3-137
METHOD	Sets parameters associated with solution algorithms.	3-139
SOLVE	Generates solutions for specified biases.	3-150

MODELS

The **MODELS** statement sets the temperature for the simulation and specifies model flags to indicate the inclusion of various physical mechanisms and models.

```
MODELS

   [ {SRH | CONSRH} [R.TUNNEL] ] [AUGER] [BGN]
   [ {FN.CUR | (DT.CUR [DT.METH=<n>] [DT.CBET] [DT.VBET])} ]
   [ {BOLTZMAN | FERMIDIR} ] [IMPACT.I] [II.NLOC] [II.TEMP]
   [ INCOMPLE [ENERGY.L] [HIGH.DOP] [IMPURITY=<c>] ]
   [ BTBT [BT.MODEL=<n>] [BT.LOCAL=<n>]
     [ BT.QUAD [BT.ATOL=<n>] [BT.RTOL=<n>] [BT.TINY=<n>] ]
   ]
   [ {  CONMOB | ANALYTIC | ARORA | CCSMOB | PHUMOB | LSMMOB
      | GMCMOB | SHIRAMOB | LUCMOB
     }
   ]
   [ {  SRFMOB | SRFMOB2 | UNIMOB | PRPMOB | LSMMOB
      | GMCMOB | SHIRAMOB | TFLDMOB | HPMOB | LUCMOB
     }
   ]
   [ {HPMOB | FLDMOB | LUCMOB | TMPMOB } [ {ND.MOB | C.ND.MOB=<n>} ] ]
   [E.EFFECT] [EJ.MOBIL] [EHSCAT] [STRMOB]
   [ QM.PHILI [QM.AC] [QM.OLD] [QM.METHO=<n>] [QM.NORP=<n>]
     [QM.EFIEL=<n>] [QM.EMIN=<n>] [QM.EXTEN]
   ]
   [ {GATE1 | ( GATE2 [GATE.SUR] )} [GATE.GEN=<n>] [GATE.TEM] ]
   [TMPDIFF] [ET.MODEL] [EF.TMP] [EFI.TMP] [COMP.ET] [EBLT.HT]
   [TMPTAUWN]  [TMPTAUWP] [EB.SRH.G] [TEMPERAT=<n>] [3KT.LT] [ECII.LAT]
   [ {HJSC2 | ( HJTEM [HJTUN] )} ] [STRESS] [Y.ORIENT=<n>] [PRINT]
```

Parameter	Type	Definition	Default	Units
SRH	logical	Specifies that Shockley-Read-Hall recombination with fixed lifetimes is used.	false	
CONSRH	logical	Specifies that Shockley-Read-Hall recombination with concentration dependent lifetimes is used.	false	
R.TUNNEL	logical	Specifies that trap-assisted and band-to-band tunneling are included in the Shockley-Read-Hall recombination.	false	
AUGER	logical	Specifies that Auger recombination is used.	false	
BGN	logical	Specifies that band-gap narrowing is used.	false	
FN.CUR	logical	Specifies that self-consistent Fowler-Nordheim tunneling is to be included in the simulation. Since a self consistent model requires that current flow in the oxide be modeled, the oxide layers must be modeled as **S.OXIDE**, i.e., as a wide band gap semiconductor. If **FN.CUR** is specified on the **SOLVE** statement instead of on the **MODEL** statement then a post processing Fowler-Nordheim tunneling model is used in place of the self consistent version.	false	

Parameter	Type	Definition	Default	Units
DT.CUR	logical	Specifies that self-consistent direct tunneling is to be included in the simulation. If **DT.CUR** is specified on the **SOLVE** statement instead of on the **MODELS** statement, then a post processing direct tunneling model is used in place of the self consistent version.	false	
DT.METH	number	The method for evaluating the direct tunneling current: 2=WKB, 3=Gundlach	2	
DT.CBET	logical	Specifies that conduction band electron tunneling should be included in direct tunneling.	true	
DT.VBET	logical	Specifies that valence band electron tunneling should be included in direct tunneling.	false	
BOLTZMAN	logical	Specifies that Boltzmann carrier statistics are used.	true	
FERMIDIR	logical	Specifies that Fermi-Dirac carrier statistics are used.	false	
IMPACT.I	logical	Specifies that carrier generation due to impact ionization is included in the solution self-consistently.	false	
II.NLOC	number	Specifies the model level for non-local impact ionization: -1=no, 0=post-processing only, (2,1)=self-consistent solution with(2) or without(1) derivatives	-1	
II.TEMP	logical	Specifies that a carrier temperature-based model is used for impact ionization instead of the electric field-dependent model whenever impact ionization is used.	false	
INCOMPLE	logical	Specifies that incomplete-ionization of impurities are accounted for.	false	
ENERGY.L	logical	Specifies that a doping and temperature dependent impurity activation energy model is used in conjunction with the **INCOMPLE** model.	false	
HIGH.DOP	logical	Specifies that the **INCOMPLE** model is modified to account for a transition from partial ionization to complete ionization at high doping levels.	false	
IMPURITY	char	The impurities for which the **INCOMPLE** model is applied. To specify more than one impurity, separate their names with commas and enclose the group within parentheses.	all impurities	
BTBT	logical	Specifies that carrier generation due to band-to-band tunneling is included in the solution.	false	
BT.MODEL	number	Specifies the tunneling field model for **BTBT**. A value of 1 indicates a local field, a value of 2 indicates an average field, and a value of 3 indicates a path integral.	1	
BT.LOCAL	number	Specifies the locality of **BTBT** generation. A value of 1 indicates that electron-hole pairs are generated at the beginning of the tunneling path, and a value of 0 indicates that electrons and holes are generated at opposite ends of the tunneling path.	1	
BT.QUAD	logical	Specifies that recursive refinement is to be applied in **BTBT** calculations.	false	
BT.ATOL	number	Specifies the threshold **BTBT** generation rate below which recursive refinement is not applied.	1.0e16	#/cm^3/s
BT.RTOL	number	Specifies the log10 relative change in **BTBT** generation rate above which recursive refinement is to be applied.	0.3	
BT.TINY	number	Specifies the size of the triangle below which recursive refinement is not applied.	5e-4	microns
CONMOB	logical	Specifies that mobility tables are used to model the dependence of carrier mobility on impurity concentration.	false	

Parameter	Type	Definition	Default	Units
ANALYTIC	logical	Specifies that concentration and temperature dependent mobility calculated from an analytic expression is used.	false	
ARORA	logical	Specifies that a concentration and temperature dependent mobility model based on the work of Arora, et al. is used.	false	
CCSMOB	logical	Specifies that the carrier-carrier scattering model is to be used. This model also includes the dependence of mobility on doping and temperature.	false	
PHUMOB	logical	Specifies that the Philips Unified mobility model is used. Note that **PHUMOB** models minority carrier mobility adequately, which is crucial for bipolar devices. However, to model minority carrier transport correctly, appropriate values of band-gap narrowing parameters should be used. See Chapter 2, "Philips Unified Mobility" on page 2-21.	false	
SHIRAMOB	logical	Specifies that the Shirahata mobility model is used. Note that **SHIRAMOB** uses all the parameters of **PHUMOB**.	false	
LUCMOB	logical	Specifies that the Lucent mobility model is used.	false	
LSMMOB	logical	Specifies that the Lombardi surface mobility model is used.	false	
SRFMOB	logical	Specifies that an effective mobility is calculated at semiconductor-insulator surfaces using an effective electric field.	false	
SRFMOB2	logical	Specifies that an enhanced surface mobility model is used along semiconductor-insulator surfaces which accounts for phonon scattering, surface roughness scattering, and charged impurity scattering.	false	
UNIMOB	logical	Specifies that the Universal Mobility model is used.	false	
PRPMOB	logical	Specifies that a mobility model using the perpendicular electric field component is used.	false	
GMCMOB	logical	Specifies that the Generalized Mobility Curve mobility model is used.	false	
TFLDMOB	logical	Specifies that a transverse field-dependent mobility model based on work done at the University of Texas, Austin, is used.	false	
HPMOB	logical	Specifies that a mobility model developed by Hewlett-Packard which takes into account both the transverse and parallel electric field components is used.	false	
FLDMOB	logical	Specifies that a mobility model using the parallel electric field component is used.	false	
TMPMOB	logical	Specifies that a mobility model using a carrier temperature-based effective field is used.	false	
ND.MOB	logical	Only relevant for GaAs-like semiconductors when **FLDMOB** or **TMPMOB** are used. If **ND.MOB** is true, negative values of mobility derivatives are included in the calculation of the Jacobian matrix. If **ND.MOB** is false, negative values of the mobility derivative are limited to the value of **C.ND.MOB**, which may improve convergence.	false	
C.ND.MOB	number	Only relevant for GaAs-like semiconductors when **FLDMOB** or **TMPMOB** are used. Specifies the minimum value of normalized mobility derivatives, $(E/\mu)\cdot(\partial\mu/\partial E)$, that are included in the calculation of the Jacobian matrix. The value specified for **C.ND.MOB** should be less than or equal to 0. Specifying values in the vicinity of -1 for this parameter sometimes helps difficult convergence problems. NOTE: **C.ND.MOB** will only be applied if **ND.MOB** is false.	0.0	none
E.EFFECT	logical	Specifies that an effective electric field is calculated at interfaces for use in the transverse field dependent mobility models.	true	

Parameter	Type	Definition	Default	Units
EJ.MOBIL	logical	Specifies that the electric field components used in the mobility calculations are parallel and perpendicular to current flow. If this parameter is not specified, the mobility reduction along a side of a triangular element is computed using the electric field components parallel and perpendicular to the side.	false	
EHSCAT	logical	Specifies that electron hole scattering is included in the electron and hole continuity equations.	false	
STRMOB	logical	Specifies that stress-induced changes to the electron and hole mobilities are included. This model must be used in conjunction with the **STRESS** model for bandgap change.	false	
QM.PHILI	logical	Specifies that quantum mechanical corrections for MOSFET inversion layers are included by invoking the Philip's band-gap widening model.	false	
QM.AC	logical	Specifies that a new implementation of QM correction model is applied. This allows the QM correction to be included in AC small-signal analysis. This implementation also reduces model evaluation time in normal simulation. **synonym: QM.NEW**	true	
QM.OLD	logical	Specifies that the old calculation of interface normal electric field used in versions of Medici prior to 1999.2 will be used with the **QM.PHILI** model.	false	
QM.METHO	number	An integer that specifies how quantum mechanical band-gap widening is applied as we move away from the interface. Specifying **QM.METHO**=1 uses a method suggested by van Dort. Specifying **QM.METHO**=2 uses a similar method suggested by researchers at Stanford University.	1	
QM.NORP	number	An integer that specifies whether quantum mechanical band-gap widening occurs in n-type regions (**QM.NORP**=1), p-type regions (**QM.NORP**=-1), or both (**QM.NORP**=0). Values of **QM.NORP**=±2 cause the program to behave like **QM.NORP**=±1 before inversion and like **QM.NORP**=0 as you approach inversion. If this parameter is specified here, it will apply to the entire device structure. To apply this parameter to specific regions, use the **QM.NORP** parameter on the **MATERIAL** statement.	0	
QM.EFIEL	number	An integer that specifies whether quantum mechanical band-gap widening occurs in regions where the normal electric field at the interface points into the semiconductor (**QM.EFIEL**=1), points into the insulator (**QM.EFIEL**=-1), or both (**QM.EFIEL**=0). If this parameter is specified here, it will apply to the entire device structure. To apply this parameter to specific regions, use the **QM.EFIEL** parameter on the **MATERIAL** statement.	0	
QM.EMIN	number	The minimum interface normal electric field that must exist before the **QM.PHILI** model is applied.	1e3	V/cm
QM.EXTEN	logical	Specifies that the **QM** model evaluation is extended such that flat band singularity is removed and **QM** correction in the accumulation regime is reduced at high doping levels.	true	
GATE1	logical	Specifies that the standard lucky-electron gate current model is to be used for performing gate current calculations. Gate current calculations are enabled by specifying **GATE.CUR** on the **SOLVE** statement.	true	
GATE2	logical	Specifies that a more complicated angle-dependent gate current model is used for performing gate current calculations. See "Carrier Temperature Dependent Gate Current Models" on page 2-103 for more details. Gate current calculations are enabled by specifying **GATE.CUR** on the **SOLVE** statement.	false	
GATE.SUR	logical	Specifies that gate current calculations are only performed along the insulator-semiconductor interface. This type of analysis may be more accurate with coarse grids. This parameter only effects the **GATE2** model.	false	

Parameter	Type	Definition	Default	Units
GATE.GEN	number	Selects the gate current hot carrier distribution function to be used during gate current calculations. A value of 1 selects a standard Maxwellian function. A value of 2 selects a more complex formula that was derived from Monte Carlo data. See Chapter 2, "Gate Current Analysis" on page 2-97 for more details.	1	none
GATE.TEM	logical	Specifies that the electron or hole temperatures calculated from the solution of the energy balance equations are used in place of the local electric field during gate current calculations.	false	
TMPDIFF	logical	Specifies that the thermal diffusion term in the current density definition is used. This parameter is only relevant when one or both of the carrier energy balance equations are included with the solution of the device equations.	true	
ET.MODEL	logical	Specifies that the Energy Transport Model is used when solving the energy balance equations. If this parameter is not selected, the conventional energy balance model is used.	false	
EF.TMP	logical	Specifies that the homogeneous energy balance equation is solved locally to determine the effective electric field used in the carrier temperature dependent mobility models. This parameter should normally be selected in simulations that include the solution of the carrier energy balance equations if the **TMPMOB** parameter is specified.	Si, Poly, Semi, SiGe, Ge, SiC: false; GaAs, AlGaAs: true	
EFI.TMP	logical	Specifies that the cooling term due to impact ionization is included in the homogeneous energy balance equation utilized to determine the local effective electric field used in the carrier temperature-dependent mobility and/or impact ionization models. Automatically turns **EF.TMP** on when selected.	false	
COMP.ET	logical	Only relevant for GaAs-like semiconductor regions. It enables an energy-balance model suitable for compound-semiconductors. No action is taken if neither **ELE.TEMP** nor **HOL.TEMP** is specified on the **SYMBOLIC** statement.	false	
EBLT.HT	logical	Enables a lattice heating model based on the energy transfer between the carrier bath and the phonon bath. Only relevant if **LAT.TEMP** and at least one of **ELE.TEMP**, **HOL.TEMP** is specified on the **SYMBOLIC** statement.	true	
TMPTAUWN	logical	Enables a carrier temperature-dependent electron energy relaxation time model. Separate models exist for silicon and GaAs materials. This model is only applicable if **ELE.TEMP** is specified on the **SYMBOLIC** statement.	false	
TMPTAUWP	logical	Enables a carrier temperature-dependent hole energy relaxation time model. This model is only applicable if **HOL.TEMP** is specified on the **SYMBOLIC** statement.	false	
EB.SRH.G	logical	Allows SRH recombination carrier cooling to be accounted for (active only at nodes where $U_{SRH} < 0$). Relevant only when at least one carrier temperature is calculated.	false	
TEMPERAT	number	The (initial) lattice temperature for the structure.	300	°Kelvin
3KT.LT	logical	Specifies that each recombining carrier pair transfers $3kT_{Lattice}$ Joules to the lattice in addition to the gap energy E_g. Relevant only if the lattice temperature is calculated and neither balance equation is solved.	true	
ECII.LAT	logical	Specifies that the critical electric field used in impact ionization is calculated from the default expression for all regions. This resets values previously specified with **ECN.II** or **ECP.II** on the **MATERIAL** statement.	false	
HJSC2	logical	Specifies that currents across virtual sides are such that fermi levels are continuous across heterojunctions. Only relevant with heterojunctions if **VIRTUAL** was turned *on* in the **MESH** statement.	true	

Parameter	Type	Definition	Default	Units
HJTEM	logical	Specifies that a thermionic emission current model will be used to calculate heterojunction currents. Automatically turns **HJSC2** off when stated. Only relevant with heterojunctions if **VIRTUAL** was turned *on* in the **MESH** statement.	false	
HJTUN	logical	Specifies that a tunneling current model will be used to calculate heterojunction currents. This model requires **HJTEM** to be on. The tunneling current and thermionic emission current will be combined. Only relevant with heterojunctions if **VIRTUAL** was turned *on* in the **MESH** statement.		
STRESS	logical	Specifies that stress-induced changes in the bandgap are included in the solution.	false	
Y.ORIENT	number	Specifies the crystallographic orientation of the substrate used for stress calculations. Must be one of 100, 110, or 111.	100	
PRINT	logical	Specifies that the status of all models, some constants, and some temperature dependent parameters are printed.	false	

Description

The **MODELS** statement allows you to specify various physical models to use in the simulation. The temperature for the simulation is also set here. A model selected with this statement remains in effect until another **MODELS** statement explicitly requests not to use it.

See Also…

To further illustrate the **MODELS** statement, refer to:

- Input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, on page 4-1
- Input file *mdex2* in NPN Bipolar Transistor Examples, Chapter 5, on page 5-1
- Most other examples where a solution is calculated.

Impact Ionization

Impact ionization can be requested in one of two ways:

- By specifying **IMPACT.I** on the **MODELS** statement, carrier generation due to impact ionization is included in the solution. In this case, the Newton solution method with two carriers is required.
- By specifying **IMPACT.I** on the **SOLVE** statement, a post-processing analysis of impact ionization is performed. This is based on the electric fields and current densities for the most recently solved for bias or time point (see the **SOLVE** statement).

Carrier Temperature

If **II.TEMP** is specified on the **MODELS** statement, impact ionization is based upon carrier temperature instead of the standard local electric field value as described in, Chapter 3, on page 3-146. This applies to both the self-consistent and post-processing simulation of impact ionization.

Fowler-Nordheim Tunneling

Fowler-Nordheim tunneling can be requested in one of two ways:

- By specifying **FN.CUR** on the **MODELS** statement, current due to tunneling is included in the solution. In this case, the Newton solution method with two carriers is required and the oxide through which tunneling is occurring must be modeled using **S.OXIDE**.
- By specifying **FN.CUR** on the **SOLVE** statement, a post-processing analysis is performed. This is based on the electric fields and current densities for the most recently solved for bias or time point (see the **SOLVE** statement).

Direct Tunneling

Direct tunneling can be requested in one of two ways:

- By specifying **DT.CUR** on the **MODELS** statement, current due to direct tunneling is included in the solution. In this case, the Newton solution method with electrons-only for CBET or electrons and holes for VBET is required. Unlike Fowler-Nordheim tunneling, normal oxide may be used as the material through which tunneling occurs.
- By specifying **DT.CUR** on the **SOLVE** statement, a post-processing analysis is performed. This is based on the potential, electron and hole concentrations for the most recently solved for bias or time point (see the **SOLVE** statement).

Quantum Mechanical Effects in MOSFET Inversion Layers

An approximate method to account for quantum mechanical effects in MOSFET inversion layers is invoked by specifying the **QM.PHILI** parameter. This invokes a band-gap widening approach suggested by van Dort at Philips (Reference [84] in Chapter 2). A description of this model and the parameters associated with it (**QM.METHO**, **QM.NORP**, **QM.EFIEL**, and **QM.EMIN)** is given in Chapter 2, "Quantum Mechanical Effects in MOSFET Inversion Layers" on page 2-14. The quantum mechanical effects can be accounted for when performing AC small-signal analysis by specifying the parameter **QM.AC** in addition to **QM.PHILI**.

Stress-Induced Effects in Silicon

Stress-induced effects on the bandgap and mobility in silicon can be included using the **STRESS** and **STRMOB** parameters, respectively. When using the **STRMOB** mobility model, the **STRESS** model must also be used. These models require that the device orientation relative to the crystallographic coordinate system be specified using the **Y.ORIENT** parameter. Currently, only three orienta-

tions are supported. Specification of the substrate orientation using **Y.ORIENT** fixes the orientation of the *x*-axis to the value given in the following table:

Y.ORIENT	*x*-axis
100	110
110	$1\bar{1}0$
111	$1\bar{1}0$

The stress models also require that the 2D stress tensor at each node location be read from a *TIF* file using the **PROFILE** statement. The three components of the stress tensor must be identified by the names Sxx, Syy, and Sxy and are read using the **OTHER** parameter on the **PROFILE** statement as follows:

```
PROFILE IN.FILE=DEVICE.TIF TIF OTHER=(SXX,SYY,SXY)
```

Models in Solution Files

When a solution file is read using the **LOAD** statement, the model flags and temperature stored in this file replace the corresponding model flags and temperature in the present setup. This makes it unnecessary to respecify models that were selected in a previous simulation when continuing the simulation from a saved solution.

PHOTOGEN

The **PHOTOGEN** statement allows the steady-state or time-dependent injection of electrons and holes into the device.

```
PHOTOGEN

   [X.START=<n>] [Y.START=<n>] X.END=<n> Y.END=<n>
   [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
   [ELECTRON] [HOLES]

 Spatial Terms
   [R.CHAR=<n>]
   [A1=<n>] [A2=<n>] [A3=<n>] [A4=<n>]
   [C1=<n>] [C2=<n>] [C3=<n>] [C4=<n>]
   [RECO=<n>] [ IN.FILE=<c> [RD.CHAR] ] [PC.UNITS] [CLEAR]
   [G.INTEG] [N.INTEG=<n>]

 Temporal Terms
   {  UNIFORM
    | ( GAUSSIAN TC=<n> [T0=<n>] )
    | ( DELTA [T0=<n>] )
    | ( PULSE TRS=<n> TPD=<n> TFS=<n> TPRD=<n> [T0=<n>] )
   }

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]

Optical Device AAM Parameters

 Incident Ray Quantities
   [ RAYTRACE X.ORG=<n> Y.ORG=<n> [ANGLE=<n>] [SPLIT.RA]
     {  ( WAVELENG=<n> {FLUX=<n> | INTENSIT=<n>} )
      | ( {SP.FILE=<c> | ( BB.RADIA [BB.TEMP=<n>] ) }
          WAVE.STA=<n> WAVE.END=<n> WAVE.NUM=<n>
        )
     }
     [RAY.WIDT=<n>] [RAY.NUM=<n>]
     [WAVE.SCA=<n>] [INT.SCAL=<n>]
     {  ( [POLARIZA=<n>] [PHASE.DI=<n>] )
      | ( [A.ELLIPS=<n>] [R.ELLIPS=<n>] )
     }
   ]

 Ray-Tracing Quantities
   [ {INT.RATI=<n> | INT.LIMI=<n>} ]
   [BOT.RFLT=<n>] [TOP.RFLT=<n>] [SID.RFLT=<n>] [SID.INCI] [TRANSPAR]
   [AMB.REFR=<n>] [WIDTH.CH] [QUAN.EFF=<n>] [PRINT.AB]

 Film Quantities
   [FILM.REG=<c>]
```

Parameter	Type	Definition	Default	Units
X.START	number	The x-coordinate of the start of the line segment along which carriers are injected.	Minimum x-coordinate in the device structure.	microns
Y.START	number	The y-coordinate of the start of the line segment along which carriers are injected.	Minimum y-coordinate in the device structure.	microns
X.END	number	The x-coordinate of the end of the line segment along which carriers are injected.	Minimum x-coordinate in the device structure.	microns
Y.END	number	The y-coordinate of the end of the line segment along which carriers are injected.	Minimum y-coordinate in the device structure.	microns
X.MIN	number	The minimum x-coordinate at which charge generation is allowed; generation is zero for x < **X.MIN**.	Minimum x-coordinate in the device structure.	microns
X.MAX	number	The maximum x-coordinate at which charge generation is allowed; generation is zero for x > **X.MAX**.	Maximum x-coordinate in the device structure.	microns
Y.MIN	number	The minimum y-coordinate at which charge generation is allowed; generation is zero for y < **Y.MIN**.	Minimum y-coordinate in the device structure.	microns
Y.MAX	number	The maximum y-coordinate at which charge generation is allowed; generation is zero for y > **Y.MAX**.	Maximum y-coordinate in the device structure.	microns
ELECTRON	logical	Specifies that electrons are generated.	true	
HOLES	logical	Specifies that holes are generated.	true	

Spatial Terms

R.CHAR	number	The characteristic radial distance for charge generation. **synonym: DCHR**	0.0 or read from **IN.FILE** if **RD.CHAR** is true	microns
A1	number	The constant depth dependence of charge generation.	0.0	carriers/cm^3
A2	number	The linear depth dependence of charge generation.	0.0	carriers/cm^3/micron
A3	number	The pre-exponential depth dependence of charge generation.	0.0	carriers/cm^3
A4	number	The exponential depth dependence of charge generation.	0.0	micron

Parameter	Type	Definition	Default	Units
C1	number	Curve-fit parameter for single-event upset.	0.0	carriers/cm^3 or picoCoul/ micron
C2	number	Curve-fit parameter for single-event upset.	1.0	none
C3	number	Curve-fit parameter for single-event upset.	0.0	1/micron
C4	number	Curve-fit parameter for single-event upset.	0.0	none
RECO	number	The Shockley-Read-Hall lifetime modifier.	0.0	1/second/ cm^3
IN.FILE	char	The formatted file from which to read Lf(l). The file must contain two columns: the first is the depth (distance l) in microns, the second is the generation in carriers/cm^3 or picoCoul/micron. If **IN.FILE** is not specified Lf(l) = 0 for all values of l. **synonym: LETFILE**	none	
RD.CHAR	logical	Specifies that the characteristic radial distance **R.CHAR** is read from the third column of **IN.FILE.** **synonym: RD.DCHR**	false	
PC.UNITS	logical	Specifies that the units for **C1** and the second column of the **IN.FILE** are picoCoul/micron instead of carriers/cm^3.	false	
CLEAR	logical	Causes Medici to zero the spatial photogeneration array before adding new generation terms.	true	
G.INTEG	logical	Causes Medici to integrate the generation function over each element when applying the generation at a node. This greatly reduces the discretization error if a coarse grid is used.	true	
N.INTEG	number	Specifies the number of integration steps to take within each element along each direction (X,Y). For example, the integration step size in the X direction is given by min(X_e/**N.INTEG**,**R.CHAR**/**N.INTEG**,**A4/N.INTEG**) where X_e is the length of the element in the x direction.	8	none

Temporal Terms

UNIFORM	logical	Specifies that the photogeneration is uniform (constant) in time.	true	
T0	number	The time offset for the generation pulse, (also peak of the time Gaussian).	0.0	seconds
GAUSSIAN	logical	Specifies that the time dependence of the photogeneration is Gaussian.	false	
DELTA	logical	Specifies that the time dependence of the photogeneration is a Dirac delta function i.e. all generation is applied at a single time point.	false	
PULSE	logical	Specifies that the time dependence of the photogeneration is a trapezoidal pulse.	false	
TC	number	The characteristic time of the generation pulse.	0.0	seconds
TRS	number	The rise time of the intensity of the pulse-type light source.	0	second
TFS	number	The fall time of the intensity of the pulse type light source.	0	second
TPD	number	The pulse width of the intensity of the pulse type light source.	1.0e9	second
TPRD	number	The repetition period of the intensity of the pulse type light source.	1.0e9	second

Parameter	Type	Definition	Default	Units
Circuit Analysis AAM Parameters				
STRUCTUR	char	The device in which generation specified by this statement occurs. This parameter is only used with the Circuit Analysis AAM. If this parameter is not specified, then generation is applied to all devices.	none	
Optical Device AAM Parameters				
Incident Ray Quantities				
RAYTRACE	logical	Specifies that ray-tracing and photogeneration by traced rays is performed. If not specified, the related statements below this are ignored.	false	
X.ORG	number	The x-coordinate of the origin of the ray. The origin must be located outside the device region.	none	microns
Y.ORG	number	The y-coordinate of the origin of the ray. The origin must be located outside the device region.	none	microns
ANGLE	number	The angle of the ray direction relative to the horizontal axis. The value is positive clockwise and negative counter-clockwise. **synonym: THETA**	90	degrees
SPLIT.RA	logical	Specifies that ray splitting is used for ray tracing. This method uses a minimum number of rays for determining where photogeneration occurs within the device. If **^SPLIT.RA** is specified, then specify enough incident rays (using the **RAY.NUM** parameter) to "cover" the device.	true	
FLUX	number	The photon flux of the incident monochromatic ray.	none	photons/cm^2-sec
INTENSIT	number	The intensity of the incident monochrome ray. The light intensity and flux are related as follows: **INTENSIT** = (hc/λ)× **FLUX**, where h is Plank's constant, c is speed of light (hc=1.986473 × 10^{-17} J-μm).	none	Watts/cm^2
WAVELENG	number	The wavelength of the incident monochrome ray.	none	microns
SP.FILE	char	The formatted spectral intensity file name. The file must contain two columns: the first is the wavelength in microns, the second is the spectral power density in Watts/cm^2/micron. The file must contain at least two lines and the maximum number of lines that the file can have is 100.	none	
BB.RADIA	logical	Specifies that the spectral intensity distribution is given by the black body radiation formula which is a function of temperature. The intensity is given in units of [Watts/cm^2/micron] for each wavelength.	false	
BB.TEMP	number	The temperature used to characterize the black body radiation. A temperature of 5800 °Kelvin can be used to approximate the AM0 solar spectrum, which is the case outside the atmosphere of the earth.	5800	°Kelvin
WAVE.STA	number	The starting wavelength of the source ray when multi-wavelength simulations are performed.	none	microns
WAVE.END	number	The ending wavelength of the source ray when multi-wavelength simulations are performed.	none	microns
WAVE.NUM	number	The number of wavelengths of the source ray when multi-wavelength simulations are performed.	none	none
RAY.WIDT	number	The width of incident ray. The illumination width is automatically clipped to the device domain. The default width is wide enough to cover the whole device surface. (See Figure 3-20) **synonym: A.WIDTH**	twice the diagonal distance of the device domain	microns

Parameter	Type	Definition	Default	Units
RAY.NUM	number	The number of rays in the incident beam. The rays are incident in a collimated way. **synonym: A.RAYS**	1	none
INT.SCAL	number	The scale factor for the relative light intensities in multi-wavelength simulations. Each of the intensities from the spectrum file is multiplied by this factor.	1.0	none
POLARIZA	number	The linear polarization angle of the incident ray which is the angle between the electric field vector and the incident plane of the ray, or the azimuthal angle of the vibration.	45.0	degrees
PHASE.DI	number	The phase difference between TM wave and TE wave components of the incident ray.	0.0	degrees
A.ELLIPS	number	The angle made by the major axis of the polarization ellipsis of the incident wave and the horizontal axis. The zero default value means there is no phase difference between the impinging TM and TE waves.	0.0	degrees
R.ELLIPS	number	The ratio between the lengths of the elliptical axes of polarization of the incident wave ($0.0 \leq$ **R.ELLIPS** ≤ 1.0). The unity default value means the ray is non-polarized or the amplitudes of electric field of the impinging TM and TE waves are equal.	1.0	none
		Ray-Tracing Quantities		
INT.RATI	number	The numerical criterion for deciding whether to continue tracing for one ray being traced (relative value). The absolute intensity value of the criterion is (intensity of ray segment at the starting point in one region)*(**INT.RATI**).	1.0e-3	none
INT.LIMI	number	The numerical criterion for decision to continue tracing or not for one ray being traced (absolute value). If the light intensity is below this value, the program stops tracing for this beam. This parameter is only used if specified.	none	Watts/cm^2
BOT.RFLT	number	Specifies the boundary conditions for rays at the bottom edge of the device structure. A value of 1 specifies the vacuum boundary, 2 specifies total reflection, and 0 specifies the continuous boundary.	0	none
TOP.RFLT	number	Specifies the boundary conditions for rays at the top edge of the device structure. A value of 1 specifies the vacuum boundary, 2 specifies total reflection, and 0 specifies the continuous boundary.	0	none
SID.RFLT	number	Specifies the boundary conditions for rays at the left and right edges of the device structure. A value of 1 specifies a vacuum boundary and 0 specifies a continuous boundary.	0	none
SID.INCI	logical	Specifies that the incident rays can enter through the left and right edges of the device.	false	
TRANSPAR	logical	Specifies that incident rays suffer no reflection or refraction at the device surface. The transmissivity is assumed to be 1.0 in this case.	false	
AMB.REFR	number	The refractive index of the ambient surrounding the device. This value is assumed constant over the whole wavelength range. The default value is that of vacuum.	1.0	none
WIDTH.CH	logical	Specifies that the characteristic radial distance **R.CHAR** is set to half the width of the ray travelling the region.	false	none
QUAN.EFF	number	The quantum efficiency of photogeneration.	1.0	none

Parameter	Type	Definition	Default	Units
PRINT.AB	logical	Causes a table of the optical absorption distance as a function of wavelength to be printed in the program output. A separate table is printed for each semiconductor region; the doping used in the calculation of each table is indicated. Note that the actual absorption distance used at each node may be different for the values indicated in the table since the doping is not necessarily constant within a region and the absorbing distance depends on doping. The absorbing distance is defined as the distance the light must travel for its intensity to fall by a factor of 1/e.	false	
	Film Quantities			
FILM.REG	char	The region names belonging to layers of stacked materials where interference effects are accounted for. A maximum of 10 regions can be specified.	none	

Description

The **PHOTOGEN** statement allows steady-state or time-dependent injection of charge (electrons and holes) into the device. This photogeneration mode was originally developed for Single-Event Upset (SEU) simulation, but can also be used to simulate transient upset by gamma rays, x-rays, or visible light.

See Also...

To further illustrate the **PHOTOGEN** statement, refer to input files:

- *mdex5* in Photogeneration Examples, Chapter 7, on page 7-2
- *mdex6* in Photogeneration Examples, Chapter 7, on page 7-10
- *mdex12b* in Circuit Analysis Examples, Chapter 12, on page 12-7
- *mdex19a* in Optical Device Examples, Chapter 16, on page 16-1
- *mdex20a* in Optical Device Examples, Chapter 16, on page 16-13

Model

You describe a path through the device by specifying the coordinates of the end points of a line segment using:

- **X.START**, **Y.START**
- **X.END**, **Y.END**

The location of any point in the device relative to this path is parameterized by a length *l*, which is defined as the distance from (**X.START**, **Y.START**), measured parallel to the line segment and by a radius *r* which is defined as the perpendicular distance from the segment (see Figure 3-18).

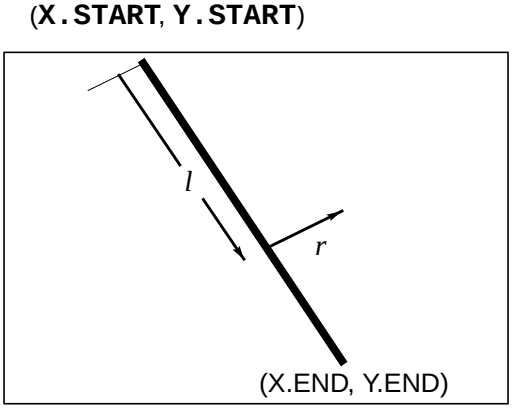


Figure 3-18 Specification of the photogeneration path

Photogeneration Equation

The generation term which is applied to the continuity equations has the following form:

$$G_n(l, r, t) \ , G_p(l, r, t) = L(l) \cdot R(r) \cdot T(t) \qquad \text{Equation 3-8}$$

where t is time.

Time-Dependent Term

The time-dependent term may be chosen as one of the four forms, **GAUSSIAN**, **DELTA**, **UNIFORM**, or **PULSE**:

Equation 3-9

$$T(t) = \begin{cases} \dfrac{2\exp\left[-\left(\dfrac{t-\mathbf{T0}}{\mathbf{TC}}\right)^2\right]}{\mathbf{TC}\sqrt{\pi}\ \mathrm{erfc}\left(-\dfrac{\mathbf{T0}}{\mathbf{TC}}\right)} & \mathbf{GAUSSIAN} \\ \delta(t-\mathbf{T0}) & \mathbf{DELTA} \\ 1 & \mathbf{UNIFORM} \\ f(\mathbf{T0, TRS, TPD, TFS, TPRD}) & \mathbf{PULSE} \end{cases}$$

Pulse Light Type Source

The parameters for a pulse type light source are defined as shown in Figure 3-19:

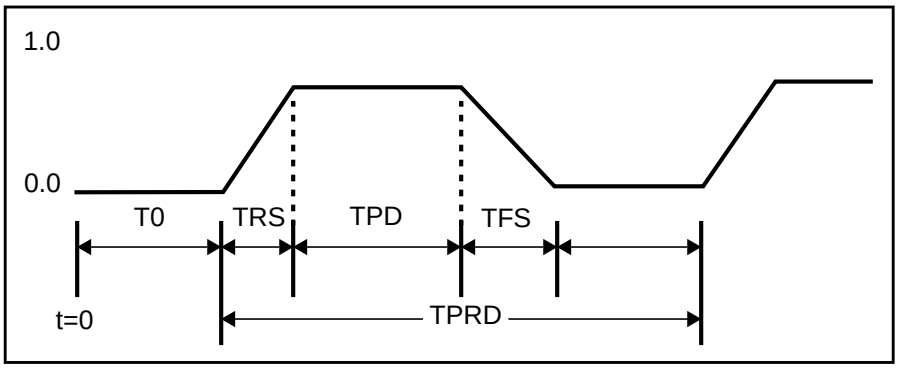


Figure 3-19 Parameter definition of pulse type light source

Note that for the **GAUSSIAN** and **DELTA** dependence, the integral of the above expression is 1.0, assuming the simulation starts at $t = 0$.

Radial Dependence

The radial dependence is:

$$R(r) = \begin{cases} \exp\left[-\left(\frac{r}{\mathbf{R.CHAR}}\right)^2\right] & \mathbf{R.CHAR} > 0 \\ 1 & \mathbf{R.CHAR}=0 \end{cases} \qquad \text{Equation 3-10}$$

Length Dependence

The length dependence has the following form:

$$L(l) = \mathbf{A1} + \mathbf{A2} \cdot l + \mathbf{A3} \cdot \exp(\mathbf{A4} \cdot l) + k[\mathbf{C1} \cdot (\mathbf{C2} + \mathbf{C3} \cdot l)^{\mathbf{C4}} + L_f(l)] \qquad \text{Equation 3-11}$$

In the above expression:

- Parameters **A1**, **A2**, **A3**, and **A4** can be used to describe the length dependence of the photogeneration rate.
- Parameters **C1**, **C2**, **C3**, and **C4** can be used to define the linear energy transfer (LET) for SEU simulations.
- $L_f(l)$ represents a table of LET values as a function of length l which is read from a formatted file using the **IN.FILE** parameter.
- If the parameter **RD.CHAR** is specified, then the characteristic distance **R.CHAR** are read from the third column of the file and used during the generation process.
- If **IN.FILE** is not specified, $L_f(l) = 0$ for all values of l.

The following defines k in Equation 3-11:

- If parameter **PC.UNITS** is specified as true, constant k is calculated by the program so that the generation rate integrated over the radial distance yields a number of carriers corresponding to the specified LET in picoCoul/micron.

- Therefore, if **PC.UNITS** is FALSE, $k = 1$ and if **PC.UNITS** is TRUE, constant k is numerically equal to:

$$k = \begin{cases} \dfrac{1.0}{1.6023\times10^{-19}\sqrt{\pi}\,\mathbf{R.CHAR}} & \text{Cartesian coordinates} \\ \dfrac{1.0}{1.6023\times10^{-19}\pi\ \mathbf{R.CHAR}^2} & \text{Cylindrical coordinates} \end{cases} \qquad \text{Equation 3-12}$$

Note:

100 MeV/(mg/cm^2) = 1.0 picoCoul/micron in silicon.

Spatially Dependent Terms

Medici stores the spatially dependent terms (the product $L(l)R(r)$ at each grid point) in two special arrays:

- One array for electrons
- One array for holes

Multiple **PHOTOGEN** statements can be used to build up complicated generation distributions since each **PHOTOGEN** statement adds its contribution to the arrays. The **CLEAR** parameter controls whether the array is cleared before the present **PHOTOGEN** statement is processed.

For example, to create a pattern of two intersecting charge tracks, the following pair of **PHOTOGEN** statements is used.

```
PHOTOGEN  X.START=0  X.END=5  Y.START=0  Y.END=5  PC.UNITS
+         C1=0.2  R.CHAR=0.2

PHOTOGEN  X.START=5  X.END=0  Y.START=0  Y.END=5  PC.UNITS
+         C1=0.3  R.CHAR=0.3  T0=4E-12  TC=2E-12  ^CLEAR

SOLVE     TSTEP=5.0E-13  TSTOP=2E-9
```

The first track has an LET of 0.2 picoCoul/micron, 1/e radius of 0.2 microns and starts at the upper left corner and ends at the lower right corner. The second track has an LET of 0.3 picoCoul/micron, 1/e radius of 0.3 microns and starts at the upper right corner and ends at the lower left corner.

The **T0** and **TC** parameters apply to *both* tracks, and both are generated simultaneously (with peak generation at 4ps).

Note:

Time parameters* T0 *and* TC *apply to both electrons and holes simultaneously.

Shockley-Read-Hall Lifetime Modifier

Since damage to the semiconductor crystal structure may occur as a result of the passage of a high energy ion, it may be appropriate to reduce the SRH lifetime along the ion's path. If **RECO** is specified, then the electron lifetime is defined by:

$$\frac{1}{T_n(x,y)} = \frac{1}{T_{n0}(x,y)} + \mathbf{RECO} \cdot G_n(x,y) \qquad \text{Equation 3-13}$$

where:

- T_n is the new SRH lifetime
- T_{n0} is the original SRH lifetime
- G_n is the electron generation function.

The evaluation of G_n in this case is not time-dependent; that is, $T(t)$ is evaluated as 1. The expression for hole lifetime is analogous.

Examples

This section provides examples of the **PHOTOGEN** statement. The following subjects are presented:

- SEU
- Gaussian X-Ray pulse
- Uniform X-ray pulse
- Visible light simulation

Single-Event Upset (SEU) #1

SEU simulation requires the generation of a thin line of charge over a period of tens of picoseconds. The line of charge is characterized by its diameter and charge density per unit length (an equivalent term is Linear Energy Transfer (LET)) in picoCoul/micron.

A typical transient SEU simulation might require the generation of a line of charge along the left edge of the device with a diameter of 1 micron and a charge density of 0.18 picoCoul/micron. The following parameters are used to specify the carrier generation:

```
PHOTOGEN  X.ST=0.  Y.ST=0.  X.EN=0.  Y.EN=5.
+         T0=20e-12  TC=10e-12  R.CHAR=0.5  C1=0.18  PC.UNITS
```

This statement creates a line of charge with a 1/e radius of 0.5 microns (diameter of 1 micron). The charge is generated gradually, over a period of about 40ps, with the peak generation occurring at 20ps. The total charge generated along the 5 micron path would be $5 \cdot (0.18) = 0.9e{\times}10^{-12}$ Coulombs. A detailed example of this type is presented in Photogeneration Examples, Chapter 7, on page 7-1.

Single-Event Upset (SEU) #2

This example simulates the impact of a 5.5 MeV alpha particle. From the TRIM program (J. F. Ziegler, J. P. Biersack, U. Littmark, "The Stopping and Ranges of

Ions in Matter," Pergamon Press, 1985), it is determined that the charge column has the following:

- A diameter of 1000 Angstroms
- An average charge density of 6.0e18 Electron-hole-pairs/cm^3
- A total length of 40 microns.

Assuming this charge is generated over a 10 picosecond time span and the particle passes down the left edge of the device, use the following:

```
PHOTOGEN  X.ST=0.  Y.ST=0.  X.EN=0.  Y.EN=40.0
+       T0=5e-12  TC=2.5e-12  R.CHAR=0.05  A1=6.0E18
```

Gaussian X-Ray Pulse

This example performs a transient simulation of a 50ns (full width), 10.0 RAD x-ray pulse. The example assumes spatially uniform generation throughout a 3x5x3 micron portion of the device.

1. One RAD of radiation generates 4.2e13 electron-hole-pairs/cm^3 (EHP/cm^3), therefore:

$$\begin{aligned} \mathbf{A1} &= 10 \ \text{RADS} \cdot 4.2 \times 10^{13} \\ &= 4.2 \times 10^{14} \ \text{EHP}/\text{cm}^3 \end{aligned}$$

2. To assure spatially uniform generation, set **R.CHAR**=0, which sets the radial dependence $R(r)$ to unity.
3. To define the 5 by 3 micron box, use the **MIN** and **MAX** parameters.
4. To define a segment along the left edge of the device, use the **START** and **END** parameters.

   Medici requires that the **START** and **END** parameters be defined, even though they have no effect here.
5. Specify the time dependence. Since the full pulse width is 50ns, the 1/e characteristic time for the Gaussian is 25ns.
6. Center the time Gaussian at 75ns to be sure to include the complete pulse.

The following statement is appropriate:

```
PHOTOGEN  A1=4.2e14.  R.CHAR=0.
+         X.MIN=1.  X.MAX=4. Y.MIN=0.  Y.MAX=5.0
+         X.ST=0.   Y.ST=0.  X.EN=0.  Y.EN=5.0
+         T0=75e-9  TC=25e-9

SOLVE     DT=1E-9  TSTOP=100E-9
```

The total charge generated (assuming the device is larger than 5 by 5 microns) would be:

Equation 3-14

$$Q = q \cdot (\mathbf{Y.MAX} - \mathbf{Y.MIN}) \cdot 10^{-4} \cdot (\mathbf{X.MAX} - \mathbf{X.MIN}) \cdot 10^{-4} \cdot \mathbf{A1}$$
$$= 1.6023 \times 10^{-19} \cdot (5.0 - 0.0) \cdot 10^{-4} \cdot (4.0 - 1.0) \cdot 10^{-4} \cdot 4.2 \times 10^{14}$$
$$= 1.01 \times 10^{-11} \text{ Coul/micron}$$

Note:

The factors of 10^{-4} convert from microns to cm.

Uniform X-Ray Pulse

This example repeats the x-ray pulse example with a uniform generation rate pulse instead of the Gaussian. This illustration assumes that the X-Ray pulse lasts for 50ns and generates 2e8 RADS/s (a total of 10 RADS as before). The electron-hole-pair generation rate G is:

$$G = 2 \times 10^{8} \text{ RADS} \cdot 4.2 \times 10^{13}$$
$$\mathbf{A1} = G = 8.21 \times 10^{21} \text{ EHP}/s$$

Equation 3-15

To assure that the pulse only lasts for 50ns, two **PHOTOGEN** statements are used as follows:

```
PHOTOGEN  A1=8.21e21  R.CHAR=0. UNIFORM
+         X.MIN=1.  X.MAX=4.  Y.MIN=0.  Y.MAX=5.0
+         X.ST=0.  Y.ST=0.  X.EN=0.  Y.EN=5.0

SOLVE     DT=1E-9  TSTOP=50E-9

PHOTOGEN  A1=0.0  R.CHAR=0. UNIFORM
+         X.MIN=1.  X.MAX=4. Y.MIN=0.  Y.MAX=5.0
+         X.ST=0.  Y.ST=0.  X.EN=0.  Y.EN=5.0

SOLVE     TSTOP=100E-9
```

The second **PHOTOGEN** statement has a generation rate of zero and stops all generation at 50ns.

Visible Light Simulation

Consider steady-state simulation of visible light at 1.0e12 photons/cm^2/s, with a characteristic 1/e absorption distance of 0.5 micron. Note that since **A3** has units of EHP/cm^3, it is necessary to divide the photon flux by the absorption distance (in cm) to calculate **A3**. Appropriate coefficient values would be:

```
PHOTOGEN  X.ST=0.  Y.ST=0.  X.EN=0.  Y.EN=2.  X.MIN=0.
+         X.MAX=3.  A3=2.0e16  A4=-2.
```

In this case the total charge generated is:

Equation 3-16

$$Q = q \cdot \#\text{photons/cm}^2 \cdot (\texttt{X.MAX} - \texttt{X.MIN}) \cdot 10^{-4} \cdot 1 \text{ micron}$$
$$= 1.6023\times10^{-19} \cdot 10^{12} \cdot (3.0 - 0.0) \cdot 10^{-4} \cdot 10^{-4}$$
$$= 4.8\times10^{-15} \text{ Coul}$$

A detailed example of this type is presented in Chapter 7, "" on page 7-1.

Ray Tracing

Note:

These features are available only if the OD-AAM has been licensed.

This section describes the parameters associated with using ray tracing for determining photogeneration rates within a device structure.

Incident Ray Quantities

The following parameters describe the geometrical properties of the incident radiation:

- **X.ORG** and **Y.ORG** specify the origin of the light source.
- **ANGLE** specifies the beam direction. The angle has positive value when the ray direction vector is rotated clockwise and negative value counter-clockwise with reference to the horizontal axis (x axis).
- **RAY.WIDT** specifies the width of the incident beam. The default value is twice the diagonal distance of the device structure. Any part of the beam width that bypasses the device is automatically clipped.
- **RAY.NUM** (default = 1) specifies the number of initial rays incident on the device. Each ray has a width associated with it such that the sum of widths covers the entire illuminated surface.

The above quantities are illustrated in Figure 3-20.

TM and TE Wave

The relationship between the angles and the coefficients of incidence, reflection and transmission at the interface between two media is calculated using the Fresnel Formulae for the TM and TE wave for each wavelength in the chosen spectrum.

These values are calculated as a function of the complex refractive indices of the two different materials. Total reflection occurs when a beam proceeds from an optically more dense media to a less dense media ($n_1 > n_2$) and the incidence angle is larger than the critical value, which is given as $\arcsin(n_2/n_1)$.

Boundary Conditions

On the first interface at which the ray meets the device, the vacuum/device boundary is assumed unless **AMB.REFR** is specified. By default, when a ray hits an electrode other than a polysilicon or transparent electrode, total reflection is assumed.

Once within the device, each beam suffers optical absorption within the media, and refraction and reflections at the material interfaces and at the device edges.

Boundary conditions for rays within the device that hit the top, bottom, or side are specified with **TOP.RFLT**, **BOT.RFLT**, and **SID.RFLT** parameters:

TOP.RFLT
BOT.RFLT
SID.RFLT

= 0 → continuous boundary: rays simply leave the device (default)
1 → vacuum boundary: rays refract/reflect at boundary as usual
2 → total reflection boundary: rays reflect with no energy loss

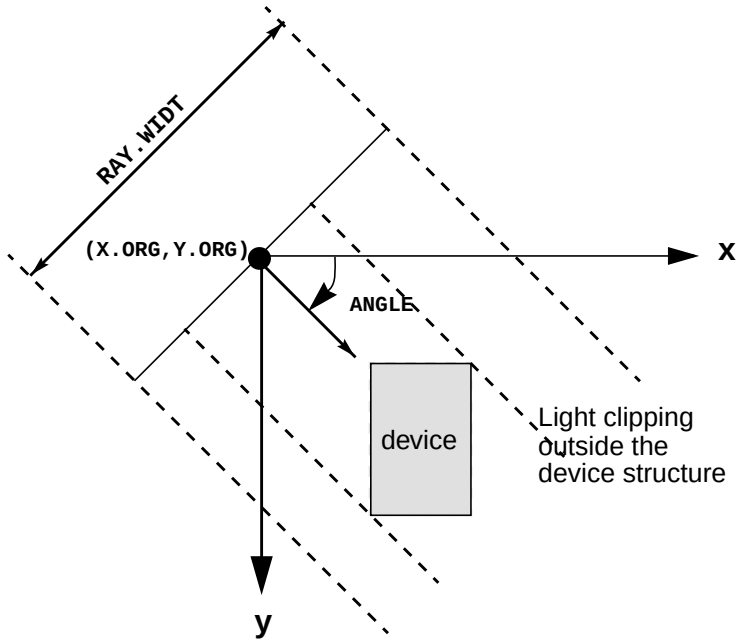


Figure 3-20 Optical ray geometry

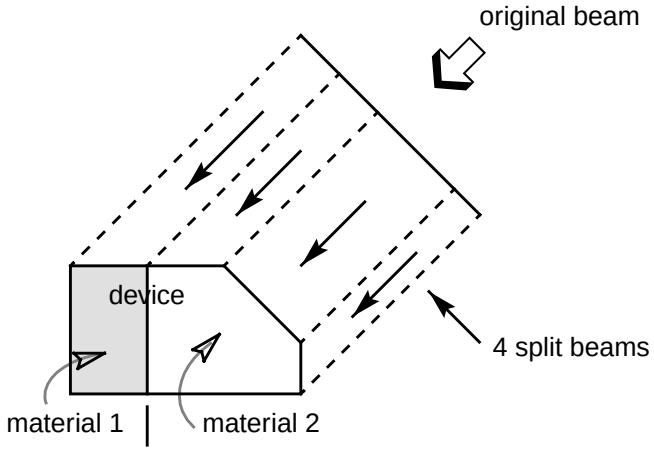


Figure 3-21 Automatic splitting of ray at discontinuous points

Ray Termination

A ray is discarded when its intensity becomes smaller than **INT.LIMI**, which is a user-specified absolute criteria for intensity, or smaller than **INTENSIT*INT.RATI**, where **INT.RATI** is also user-specifiable. Tracing is also stopped when the number of ray segments originated from one initial beam exceeds 50.

Spectral Intensity File

The optical source can be either monochromatic or spectral.

For a monochromatic ray, use the **WAVELENG** parameter to specify the wavelength in microns.

In case of a spectral light source, a spectral intensity file should be given, specified by the **SP.FILE** parameter. The format of the file is as follows:

```
wavelength_1  spectral_intensity_1
wavelength_2  spectral_intensity_2
. . .
. . .
wavelength_n  spectral_intensity_n
```

Additional considerations when using a spectral intensity file include:

- The maximum number of records is 100.
- Wavelength should be specified in units of microns and the spectral intensity in units of Watts/cm^2/μm.
- The parameters **WAVE.STA**, **WAVE.END**, and **WAVE.NUM** should be specified to sample wavelengths from the file. The maximum value of **WAVE.NUM** is 30.
- **WAVE.SCA** and **INT.SCAL** are used when the spectral intensity file has data with units other than the default units and one needs to scale them uniformly.

Transmission Through a Stack of Material Layers

Multiple layers can be modeled as a single film to take into account interference effects (constructive and destructive). This is accomplished by using the **FILM.REG** parameter to identify the regions that make up the material layers.

One should be careful in defining **FILM.REG** statements since there are some prerequisites:

- Every layer should be rectangular-shaped and run parallel with the neighboring layer.
- Electrodes can not be part of the **FILM.REG** specification.

  The exception is for polysilicon or transparent electrodes with zero thickness, since transmission is not allowed electrodes.

When a ray enters or leaves the device through the side edges of the stack defined by **FILM.REG** or meets a non-film material during propagation within the stack, the tracing of that ray is terminated. This means the stack should be wide and thin enough as well as planar and parallel to prevent such errors.

Example Statements

The following statements calculate the transmittance of an anti-reflecting coating structure of a solar cell device:

```
REGION NAME=1 Y.MAX=0 OXIDE Y.MIN=-5E-3
REGION NAME=2 Y.MIN=0 Y.MAX=150e-3  INSULATOR
REGION NAME=3 Y.MIN=150e-3 Y.MAX=300e-3  INSULATOR
REGION NAME=4 Y.MIN=300e-3 SILICON
. . .
MATERIAL REG=2 WAVE.RE=(0.31,1) INDEX.RE=(1.35,1.35)
MATERIAL REG=3 WAVE.RE=(0.31,1) INDEX.RE=(2.25,2.25)
. . .
PHOTOGEN RAYTRACE SP.FILE=AM0.DAT WAVE.ST=0.3 WAVE.EN=1
+  WAVE.NUM=10
+  X.ORG=0.5 Y.ORG=-4 ANGLE=90 INT.RATI=1e-2 N.INTEG=10
+  RAY.N=1 FILM.REG=(1,2,3)
```

- The first four statements define the structure of an AR coating which is a stack of a 5nm oxide, a 150nm MgF_2 and a 150nm ZnS layers.
- The next two **MATERIAL** statements assign the appropriate refractive indices (real terms) to MgF_2 and ZnS materials. The index of oxide is given by the default table.
- The **FILM.REG**=(1,2,3) term specifies that the stack consists of the materials with the region names "1", "2", and "3" and that the stack is analyzed as a single film structure so that the interference within the stack is taken into account.
- The *am0.dat* file gives the solar spectrum outside the atmosphere of earth and is to be read in for each wavelength.

Black-Body Radiation

The black-body radiation spectrum can be specified with the **BB.RADIA** parameter. Particular wavelengths of the spectrum can be sampled by using the parameters **WAVE.STA**, **WAVE.END** and **WAVE.NUM**. The intensity (in Watts/μm/cm^2) is defined by

$$\Delta I = \left(\frac{R_S}{R_{SE}}\right)^2 2\pi hc^2 \frac{\lambda^{-5}}{\exp\left(\frac{hc/\lambda}{k\,\mathbf{BB.TEMP}}\right) - 1}\Delta\lambda \qquad \text{Equation 3-17}$$

where:

- $R_S = 7\text{x}10^8$ m is the radius of the sun
- $R_{SE} = 1.5\text{x}10^{11}$ m is the distance between the sun and the earth
- **BB.TEMP** is the black-body temperature

The extra-terrestrial solar spectrum (AM0) can be approximated by that of a black body at 5800K.

TRAPS

The **TRAPS** statement instructs Medici to create trap states within semiconductor regions. This statement is available only if the Trapped Charge Advanced Application Module (TC-AAM) has been authorized for use.

```
TRAPS

 Energy Level Creation
  {  ( DISTRIB [N.LEVEL=<n>] [ OUT.FILE=<c> X.PLOT=<n> Y.PLOT=<n> ] )
   | ( [E1=<n>] ... [E50=<n>] )
  }
  [MIDGAP] [CHARGE1] ... [CHARGE50] [ALL.CHAR] [DGEN1=<n>] ... [DGEN50=<n>]

 Trap Parameters
  [ [TAUN=<c>] [TAUP=<c>] [N.TOTAL=<c>] [Q.FIX=<c>] [CONDITIO=<c>]
    [FREEZE]
  ]

 Transient Parameter
  [TIME.DEP]
```

Parameter	Type	Definition	Default	Units
Energy Level Creation				
DISTRIB	logical	Specifies that a distribution of **N.LEVEL** trap levels, equally spaced within the forbidden band, is created with spacing: $\Delta E_t = E_g/($**N.LEVEL**$+1)$. By default, the trap levels are centered around the intrinsic Fermi level. If the **MIDGAP** parameter is specified, then the trap levels are centered around the middle of the forbidden band.	false	
N.LEVEL	number	Specifies the number of energy states created if an uniform distribution is specified. Note that the total number of states must be less than 51.	20	none
OUT.FILE	char	Specifies that a file is created containing the trap density (**N.TOTAL**), electron and hole lifetimes (**TAUN**, **TAUP**) and fixed charge density (**QFIX**) as a function of energy at the location specified by the **X.PLOT** and **Y.PLOT** parameters. The file created is of the TIF *.ivl* format and may be viewed using Taurus Visual.	none	
X.PLOT	number	Specifies the location of the point at which the output file specified by **OUT.FILE** is created. Refer to the **OUT.FILE** parameter for more information.	0	
Y.PLOT	number	Specifies the location of the point at which the output file specified by **OUT.FILE** is created. Refer to the **OUT.FILE** parameter for more information.	0	
E<i>	number	Specifies the energy value of the *ith* energy state. Note that *i* must be in the range of 1 through 50. By default, **E<i>** is taken relative to the instrinsic Fermi level. If **MIDGAP** is specified, **E<i>** is taken relative to the middle of the forbidden band.	None, unless **DISTRIB** is specified	eV
MIDGAP	logical	Specifies that the middle of the forbidden band should be used as the energy reference when specifying the trap energy level(s). By default, the intrinsic Fermi level is used as the energy reference.	false	

Parameter	Type	Definition	Default	Units
CHARGE<i>	logical	Specifies that the state *i* is electrically charged when empty. Note that *i* must be in the range of 1 through 50. A charged electron trap is a donor state. A charged hole trap is an acceptor state.	false	
ALL.CHAR	logical	Specifies that all states are electrically charged when empty. A charged electron trap is a donor state. A charged hole trap is an acceptor state.	false	
DGEN<i>	number	Specifies the degeneracy factor for *ith* energy state.	1.0	none
Trap Parameters				
TAUN	char	Specifies the electron lifetime for the state (or states). The argument of this parameter is a numeric expression which may be a function of the state energy *fener*, the state number *fnener*, or any other predefined Medici variable (refer to "EXTRACT" on page 3-166).	none	Seconds
TAUP	char	Specifies the hole lifetime for the state (or states). The argument of this parameter is a numeric expression which may be a function of the state energy *fener*, the state number *fnener*, or any other predefined Medici variable (refer to "EXTRACT" on page 3-166).	none	Seconds
N.TOTAL	char	Specifies the number of traps for the state (or states). The argument of this parameter is a numeric expression which may be a function of the state energy *fener*, the state number *fnener*, or any other predefined Medici variable (refer to "EXTRACT" on page 3-166). If **N.TOTAL** > 0 electron traps are created. If **N.TOTAL** < 0 hole traps are created. If **N.TOTAL** = 0, only recombination occurs from the trap (no charge trapping occurs).	none	traps/cm^3/eV if **DISTRIB** is specified, otherwise traps/cm^3
Q.FIX	char	Specifies a density for fixed charge which is placed within the device. The argument of this parameter is a numeric expression which may be a function of any predefined Medici variable (refer to "EXTRACT" on page 3-166).	none	#/cm^3
CONDITIO	char	Specifies the condition that must be satisfied for the above four parameter expressions (**TAUN**, **TAUP**, **N.TOT**, **QFIX**) to be applied. The argument of this parameter is a logical expression which may be a function of the state energy *fener*, the state number *fnener*, or any other predefined Medici variable (refer to "EXTRACT" on page 3-166).	none	
FREEZE	logical	Specifies that the present trap occupation levels are "frozen" and are not allowed to change during subsequent **SOLVE** statements.	false	
Transient Parameter				
TIME.DEP	logical	Specifies that the traps are to be time dependent. This parameter applies to all trap states.	false	

Description

The **TRAPS** statement provides a powerful method of modeling the effects of defects within the bandgap of a semiconductor.

See Also...

To further illustrate the **TRAPS** statement, refer to input files:

- *mdex18a* in "Thin Film Transistor Example" on page 15-1
- *mdex18b* in "Bipolar Junction Transistor Turn-Off with Traps Simulation" on page 15-7
- *mdex18c* in "Breakdown Walk-Out in Power MOS Device" on page 15-10

Note:

Special Medici ***assigned variables for this statement only:***

fener—Represents the energy value within the forbidden gap

fnener—Represents the energy state number i

When discrete trap levels are specified using multiple **TRAPS** ***statements, fnener must be used in*** **CONDITIO** ***for all the statements after the first to assure that the trap parameters are assigned properly. Otherwise, all the trap levels specified thus far will be assigned the parameter values that are specified on the current*** **TRAPS** ***statement.***

Trap Model

The trap model consists of two parts:

- A recombination model (which applies only to the continuity equations)
- A charge trapping model (which applies to the Poisson equation)

The charge trapping model is activated by specifying **N.TOTAL** on the **TRAPS** statement.

Trap States

There are five possibilities for a given state:

N.TOTAL	**CHARGE**<i>	**Result**
0	Not Applicable	Recombination only
>0	TRUE	Donor state
>0	FALSE	Neutral electron trap
<0	TRUE	Acceptor state
<0	FALSE	Neutral hole trap

Note:

Parameters **N.TOTAL** ***and*** **CHARGE<i>** ***do not affect the recombination process.***

Saved Information

Trap information is stored within the mesh file and the solution file, and does not need to be respecified each time a simulation is performed. For use with the CA-AAM, each numerical device model *must* have the same number of total energy levels.

Trap Statement Examples

This section presents examples of trap statements. Each example includes a discussion of the created trap, and its necessary parameters and statements.

Single Electron Trap

The following statement creates a single electron trap level positioned 0.1eV above mid-gap with an electron lifetime of 1e-5 sec, a hole lifetime of 1e-6 sec and a trap density of 1e13/cm^3:

```
TRAPS  E1=0.1 MIDGAP TAUN="1E-5" TAUP="1E-6"
+      N.TOT="1E13"
```

Time Dependent Traps

The trap statements below create two time dependent trap states, one for electrons and one for holes, with parameters listed in the following table. The trap energy levels are specified relative to the intrinsic Fermi level.

Parameter	State 1	State 2
Type	Electron	Hole
Tn	1e-5	3e-5
Tp	1e-6	5e-5
Density	1e14	1e13
Energy	0.2	-0.3

```
TRAP  E1=0.2 TAUN="1E-5" TAUP="1E-6"
+     N.TOT="1E14" COND="@FNENER=1"
TRAP  E2="-0.3" TAUN="3E-5" TAUP="5E-6"
+     N.TOT="-1E13" COND="@FNENER=2"  TIMEDEP
```

Note:

By default, energy level E1 is assigned the normal SRH recombination parameters. Normally these parameters are discarded by specifying energy levels E1, E2, etc. Starting the energy level sequence with E2 will retain the normal SRH recombination parameters for energy level E1. Any gap in the numeric sequence of energy levels will create an unintentional normal SRH recombination level. For example, specifying E1, E2, and E4 will create an E3 level for normal SRH recombination.

TFT Simulation

To create the trap distribution of Figure 3-22 (which is of the type used in TFT simulation), it is easiest to use two **TRAPS** statements.

- The first statement creates the hole traps in the lower 1/2 of the forbidden band (*fener<0).*
- The second statement creates the electron traps (*fener*>0).

For the electron and hole lifetimes, constant values of 1e-5 and 1e-6 is used, respectively. For this example the trap density is greatest at the surface of the device (y=0) and decays linearly to zero at y=0.1 micron.

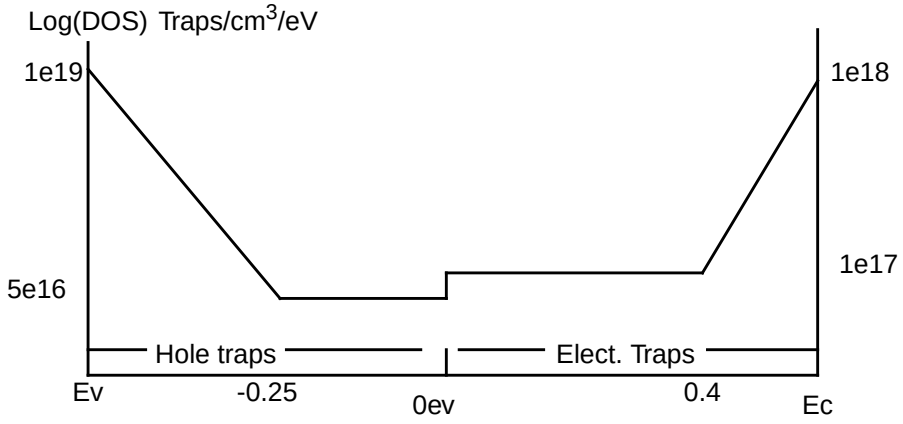


Figure 3-22 Trap density states function to be modeled

Electron Traps

For the electron traps, the density of state must have the following form:

$$G(x, y, E) = (1 - y/0.1)(1e17 + 1e18\exp(E - E_c)/L_n) \quad \text{Equation 3-18}$$

In the above equation:

- E is the energy within the band
- E_c is the conduction band energy
- L_n is the characteristic energy for the exponential portion of the electron DOS profile

  L_n is calculated from $L_n = (E_c - 0.4)\ \log(1e18/1e17)$.

This expression is the result of forcing the exponential portion Equation 3-18 to equal 5e16 at an energy of 0.4 eV. In the input file, L_n is represented by the assigned variable NCHR.

Hole Traps

The expression for the hole density of states function is:

$$G(x, y, E) = (1 - y/0.1)(5e16 + 1e19\exp(-(E - E_v)/L_p) \quad \text{Equation 3-19}$$

In a similar manner, the characteristic length for the exponential portion of the hole trap distribution is $L_p = (-0.25 - E_v)\ \log(1e19/1e16)$. In the input file L_p is

represented by the assigned variable PCHR. The input statements to generate the traps are as follows:

```
$
ASSIGN NAME=EV N.VALUE=-1.08/2
ASSIGN NAME=EC N.VALUE=1.08/2
$
$ Find the characteristic length for the hole traps
ASSIGN NAME=PCHAR N.VALUE=(-0.25-@EV)/log(1e19/5e16)
$ Create hole traps
TRAP  DISTR TAUN="1e-5"  TAUP="1e-6"
+     N.TOT="-(1.0-@Y/0.1)*(5e16+1e19/exp(-(@FENER-@EV)/@PCHR))"
+     MIDGAP COND="(@FENER<0)&(@Y<0.1)"
$
$ Find the characteristic length for the electron traps.
ASSIGN NAME=NCHAR N.VALUE=(@EC-0.4)/log(1e18/1e17)
$Create electron traps
TRAP  TAUN="1e-5"  TAUP="1e-6"
+     N.TOT="(1.0-@Y/0.1)*(1e17+1e18*exp((@FENER-@EC)/@NCHR))"
+     MIDGAP COND="(@FENER>0)&(@Y<0.1)"
```

Interface Traps

Interface traps are a special case of the general trap model described above, and deserve special discussion.

Interface traps are created only at interface nodes. An *interface node* is defined as a semiconductor node which is on the boundary between any two regions. The interface need not be between semiconductor and oxide.

Identifying Interface Nodes

Medici provides several special predefined variables for identifying these nodes (see Figure 3-23):

- **INTERFAC** evaluates to 1 if the node is at an interface, and 0 otherwise.
- **NX.INT** evaluates to 1 for nodes which are next to, but not on interfaces and 0 otherwise.
- **PRP.DIST** gives the thickness of the grid section at an interface node measured perpendicular to the interface.

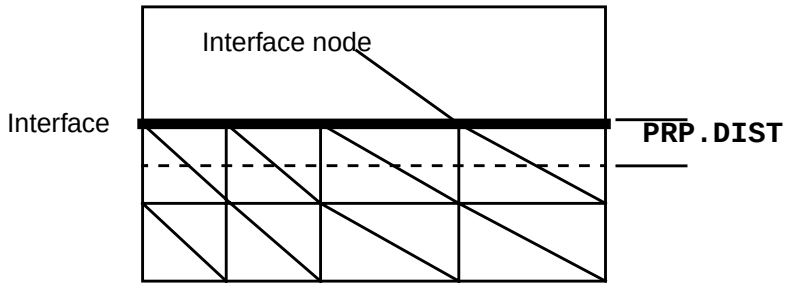


Figure 3-23 Definition of Interface nodes and **PRP.DIST**

Specifying Interface States

Interface trap densities are normally measured on the basis of #/cm^2 rather than #/cm^3 (as with the **TRAPS** statement). It is necessary to perform this conversion if the **TRAPS** statement is to be used for interface traps. The **PRP.DIST** predefined variable provides a means of doing this.

Referring to Figure 3-23, a section of grid along the interface is shown. To convert from #/cm^2 to #/cm^3, dividing by the thickness of the grid section associated with the interface nodes and **PRP.DIST** is precisely this distance.

The following example shows how this is done to create donor and acceptor states distributed uniformly across the band gap. The following may be used to replace the **INTERFACE** statement in example *mdex1f* (see Chapter 4, "" on page 4-1), with the advantage that arbitrary trap distributions as a function of energy may be specified.

```
LOOP      STEPS=3
  COMMENT   Assign the number of donors/acceptors in #/cm^2/eV
  ASSIGN    NAME=NDON N.VALUE=(0.0, 5e11, -5e11)
  COMMENT   Create the traps
  TRAPS     DISTRIB ALL.CHARGE N.TOTAL="@NDON/@PRP.DIST"
+           MIDGAP COND="@INTERFAC=1"
  SOLVE .....
 ......
L.END
```

Trap Placement

The **CONDITIO** parameter can be expanded to allow more precise placement of the traps as well. For example, to restrict interface trap generation to be between X equal 2 and 3 microns, the following is used:

```
COND="@INTERFAC=1&@X>2&@X<3"
```

Heterojunction Limitations

Some difficulties have been encountered when placing traps along heterojunction interfaces. This is mainly a problem when using the wide bandgap semiconductor **S.OXIDE** to model charge trapping within the oxide layers.

Avant! TCAD recommends that the traps be distributed throughout the oxide or placed close to the oxide rather than on the interface itself. To aid in this process, the predefined variable **NX.INT** is available which evaluates to true at those nodes adjacent to the interface. A fine grid should be specified in oxide.

A predefined variable equivalent to **PRP.DIST** is not available, and you need to evaluate this distance manually, based on the grid being used. Example *mdex18c* (see Chapter 15, "" on page 15-1) provides an illustration of how to create oxide traps. These restrictions are to be removed in a future version of the program.

SYMBOLIC

The **SYMBOLIC** statement performs a symbolic factorization in preparation for the LU decompositions in the solution phase of the program.

```
SYMBOLIC

  {NEWTON | GUMMEL}
  CARRIERS=<N> [ {ELECTRON | HOLES} ]
  [ ELE.TEMP [COUP.ELE] ] [ HOL.TEMP [COUP.HOL] ] [EB.POST]
  [ LAT.TEMP [COUP.LAT] ]
  [MIN.DEGR] [ {[ILUCGS] | [BICGS]} ] [STRIP] [VIRTUAL]
  [BLOCK.MA] [PRINT]
```

Parameter	Type	Definition	Default	Units
NEWTON	logical	Specifies that the Newton solution method is used.	true	
GUMMEL	logical	Specifies that the Gummel solution method is used.	false.	
CARRIERS	number	The number of carriers to be simulated.	1	none
ELECTRON	logical	Specifies that the simulation is for electrons if a solution for one carrier is being obtained.	true	
HOLES	logical	Specifies that the simulation is for holes if a solution for one carrier is being obtained.	false	
ELE.TEMP	logical	Specifies that the simulation is done for Poisson, continuity and electron temperature.	false	
COUP.ELE	logical	Specifies that the electron temperature equation is fully coupled (i.e. solved simultaneously) with the carrier continuity equation(s) and the Poisson equation.	false	
HOL.TEMP	logical	Specifies that the simulation is done for Poisson, continuity and hole temperature.	false	
COUP.HOL	logical	Specifies that the hole temperature equation is fully coupled (i.e. solved simultaneously) with the carrier continuity equation(s) and the Poisson equation.	false	
EB.POST	logical	Specifies that post-processing energy balance analysis is used. Post-processing energy balance ignores the electron/hole temperature dependence of the continuity equations giving an approximate solution, but with great savings in CPU time over the complete energy balance model.	false	
LAT.TEMP	logical	Specifies that the simulation is done for Poisson, continuity and lattice temperature. This parameter is only used with the Lattice Temperature AAM.	false	
COUP.LAT	logical	Specifies that the lattice temperature equation is fully coupled with the Poisson and continuity equations. This parameter is only used with the Lattice Temperature AAM.	false	

Parameter	Type	Definition	Default	Units
MIN.DEGR	logical	Specifies that a minimum degree ordering of the pivots for decomposition in order to reduce the size of the generated L and U matrices. In the case that a solution method using a complete LU factorization is used, this reduces the solution time for solving the linear system. If an iterative method relying on incomplete factorization is selected, specifying this parameter may inhibit convergence.	true	
ILUCGS	logical	Specifies that an incomplete symbolic factorization is performed instead of direct LU decomposition in order to conserve memory and ILUCGS is used to solve the linear system. **synonym: INCOMPLE**	false	
BICGS	logical	Specifies that an incomplete symbolic factorization is performed instead of direct LU decomposition in order to conserve memory and Bi-CGSTAB is used to solve the linear system.	false	
STRIP	logical	Specifies that if two adjoining elements form a rectangle, the coupling between the nodes along the diagonal are removed from the symbolic map.	False if **NEWTON** is specified and **IMPACT.I**, **PRP-MOB**, or **EJ.MOBIL** has been selected on the **MODELS** statement; otherwise, true.	
VIRTUAL	logical	Generates virtual nodes at heterojunctions	false	
BLOCK.MA	logical	Specifies that the block structure in Jacobian matrix is exploited to speed up the linear solver	false	
PRINT	logical	Specifies that memory allocation information for the run should be printed to the standard output file.	false	

Description

The **SYMBOLIC** statement performs a symbolic factorization in preparation for the LU decompositions in the solution phase of the program.

See Also...

To further illustrate the **SYMBOLIC** statement, refer to:

- Input file *mdex1* in "Saving Zero Bias Solution" on page 4-9
- Input file *mdex2* in "Generation of the Simulation Structure" on page 5-1
- Most other examples where a solution is calculated

Usage

Because each of the available numerical solution techniques used by the program may result in entirely different linear systems, the method used and the number of carriers to be simulated must be specified at this time. A symbolic factorization should be performed whenever the solution technique is changed, and before the next **SOLVE** statement if the mesh has been refined as the result of a **REGRID** statement

METHOD

The **METHOD** statement sets parameters associated with a particular solution algorithm chosen in the **SYMBOLIC** statement.

```
METHOD

  [ITLIMIT=<n>] [XNORM] [RHSNORM] [XRNORM [NODE.ERR=<n>] ]
  [PX.TOLER=<n>] [CX.TOLER=<n>] [PR.TOLER=<n>] [CR.TOLER=<n>]
  [NO.CARR=<c>] [LIMIT] [PRINT] [FIX.QF] [ITER.TTY] [ASMB.OLD]

 ILUCGS Solver Parameters
  [ILU.ITER=<n>] [ILU.TOL=<n>] [ILU.XTOL=<n>]

 Gummel's Method Parameters
  {  ( [ {  DVLIMIT=<n>
          | ( DAMPED [DELTA=<n>] [DAMPLOOP=<n>] [DFACTOR=<n>] )
         }
       ]
       [ ICCG [LU1CRIT=<n>] [LU2CRIT=<n>] [MAXINNER=<n>] ]
       [ SINGLEP
         [ ACCELERA [ACCSTART=<n>] [ACCSTOP=<n>] [ACCSTEP=<n>] ]
       ]
     )

 Newton's Method Parameters
    | ( [ AUTONR [NRCRITER=<n>] [ERR.RAT=<n>] ]
       [ CONT.RHS [ITRHS=<n>] ] [CONT.PIV] [ {CONT.ITL | STOP.ITL} ]
       [CONT.STK] [STACK=<n>] [ACONTINU=<n>]
       [TAUTO] [2NDORDER] [TOL.TIME=<n>] [L2NORM] [DT.MIN=<n>]
       [EXTRAPOL]
       [CARR.MIN=<n>] [CARR.FAC=<n>] [N.DVLIM=<n>] [N.DVMAX] [N.DAMP]
     )
  }

 Energy Balance Parameters
  [ETX.TOLE=<n>] [ETR.TOLE=<n>] [N.MAXBL=<n>] [N.MAXEB=<n>]

 Lattice Temperature AAM Parameters
  [LTX.TOLE=<n>] [LTR.TOLE=<n>] [LTX.FACT=<n>] [LTR.FACT=<n>]
  [MAX.TEMP=<n>]

 Direct Tunneling Parameter
  [DT.JACOB]
```

Parameter	Type	Definition	Default	Units
ITLIMIT	number	The maximum number of Newton iterations or Gummel continuity iterations. This parameter is also used for the maximum number of iterations for the global algorithm to solve four equations: Poisson, continuity (electrons and holes), and one of carrier temperature equations (electrons or holes) or lattice temperature equation.	20	none
XNORM	logical	Specifies that the size of the updates to the device variables at each iteration are used to determine convergence. Potential updates are measured in units of kT/q, carrier concentration updates are measured relative to the local carrier concentration and carrier temperature updates are measured relative to the ambient temperature. The solution is considered converged when the potential updates are smaller than the value specified by **PX.TOLER**, the carrier concentration updates are smaller than the value specified by **CX.TOLER**, the carrier temperature updates are smaller than the value specified by **ETX.TOLE** and the lattice temperature updates are smaller than the value specified by **LTX.TOLE** at every node in the device.	false	
RHSNORM	logical	Specifies that the differences between the left- and right-hand sides of the device equations are used to determine convergence. The Poisson equation error is measured in C/micron. The continuity equation errors are measured in A/micron. The solution is considered converged when the Poisson equation errors are smaller than the value specified by **PR.TOLER**, the continuity equation errors are smaller than the value specified by **CR.TOLER**, the carrier temperature equation errors are smaller than the value specified by **ETR.TOLE** and the lattice temperature equation errors are smaller than the value specified by **LTR.TOLE** at every node in the device.	false	
XRNORM	logical	Specifies that both **XNORM** and **RHSNORM** are used to determine convergence. A solution is considered converged when *either* the **XNORM** tolerances or the **RHSNORM** tolerances are satisfied at every node in the device.	true	
NODE.ERR	number	Specifies that node numbers are printed that correspond to the locations in the device where the maximum **XRNORM** and **RHSNORM** errors occur. **NODE.ERR**=1 prints node numbers corresponding to the maximum error. Higher values of **NODE.ERR** cause node numbers and error values to be printed for the top **NODE.ERR** errors. The maximum value allowed for **NODE.ERR** is 10.	0	none
PX.TOLER	number	The potential update tolerance. **synonym: P.TOLER**	1e-5	kT/q
CX.TOLER	number	The carrier concentration update tolerance. **synonym: C.TOLER**	1e-5	none
PR.TOLER	number	The Poisson equation error tolerance.	1e-26	C/micron
CR.TOLER	number	The continuity equation error tolerance.	5e-18	A/micron
NO.CARR	char	The region names for which steady-state carrier equations are not solved are specified. If more than one region name is specified, separate their names with commas and enclose the entire group in parentheses (for example, "(polygate,substrate)").	none	
LIMIT	logical	Specifies that the convergence criterion should be ignored, and iterations are to proceed until **ITLIMIT** is reached.	false	

Parameter	Type	Definition	Default	Units
PRINT	logical	Specifies that the terminal fluxes and currents are printed after each continuity iteration. If this parameter is not set, the terminal fluxes and currents are only printed after the solution converges.	false	
FIX.QF	logical	Specifies that the quasi-Fermi potential of each unsolved for carrier is fixed to a single value, instead of picking a value based on local bias. If electrons are not being solved for, the electron quasi-Fermi potential throughout the device is set to the maximum applied bias in the structure. If holes are not being solved for, the hole quasi-Fermi potential throughout the device is set to the minimum applied bias in the structure (also see the **P.BIAS** and **N.BIAS** parameters on the **SOLVE** statement).	false	
ITER.TTY	logical	Causes the error norms for each Newton iteration to be displayed on your terminal.	true	
ASMB.OLD	logical	Specifies that default routines in previous versions of Medici are used in equation assembly. The use of this parameter will likely result in longer CPU time.	false	

ILUCGS Solver Parameters

ILU.ITER	number	The maximum number of ILUCGS iterations to perform. The default iteration count is determined dynamically according to the number of carriers and nodes in the simulation by the following equation: $max\left(35, \sqrt{(1+carriers)\times nodes}\right)$. NOTE: The default value of **ILU.ITER** is recalulated using this expression whenever a **SYMBOLIC** statement is encountered.	The default iteration count is determined dynamically. See definition.	none
ILU.TOL	number	The ILUCGS error tolerance where the error norm is defined as $\|\|(LU)^{-1}r\|\|/\|\|(LU)^{-1}b\|\|$. For a more complete description please refer to the discussion below and Chapter 2. The majority carrier contact is implemented by injecting a majority carrier current I_m at the nodes of the electrode. The majority carrier current is calculated from $I_m = G(\phi_n - V_a)$ for N-type material and $I_m = G(\phi_p - V_a)$ for P-type material. The conductivity G is calculated by the program and corresponds to the resistivity of 0.1 micron of silicon. V_a is the voltage applied to the contact.	1e-7	none
ILU.XTOL	number	The ILUCGS x-vector error tolerance where the error norm is defined as $\|1.0-\|\|x\|\|/\|\|x'\|\|\|$. For a more complete description please refer to Chapter 2, "ILUCGS Solver" on page 2-72.	1e-3	none

Gummel's Method Parameters

DVLIMIT	number	The maximum potential update allowed for a single Gummel loop.	1.0 if **DAMPED** is true and 0.1 otherwise.	volts
DAMPED	logical	Specifies that a sophisticated damping scheme proposed by Bank and Rose is used when using Gummel's method for the solution.	true	
DELTA	number	The threshold for determining the damping factor for the potential update when **DAMPED** is specified. **DELTA** must be between 0 and 1.	0.5	none
DAMPLOOP	number	The maximum number of damping loops allowed to find a suitable damping coefficient when **DAMPED** is specified.	10	none
DFACTOR	number	The factor which serves to increase the initial damping coefficient for the next damping loop when **DAMPED** is specified.	10.0	none

Parameter	Type	Definition	Default	Units
ICCG	logical	Specifies that the iterative method of Incomplete Cholesky Conjugate Gradients is used to solve each linearized Poisson equation in the multi-Poisson loop portion of Gummel's method.	false	
LU1CRIT	number	The amount that the inner norm is required to decrease by in each Poisson loop before returning when using ICCG.	3e-3	none
LU2CRIT	number	The factor below the projected Newton error that the inner norm must reach in each Poisson loop before returning when using ICCG.	3e-2	none
MAXINNER	number	The maximum number of ICCG iterations to perform.	25	none
SINGLEP	logical	Specifies that only a single Poisson iteration is to be performed per Gummel loop. If this parameter is not specified, the standard Gummel iterative procedure is carried out where the continuity equation is only treated after the Poisson equation has fully converged (i.e., multi-Poisson).	false	
ACCELERA	logical	Specifies that acceleration is used in the **SINGLEP** mode.	false	
ACCSTART	number	The starting value of the acceleration parameter.	0.3	none
ACCSTOP	number	The final (limiting) value of the acceleration parameter.	0.6	none
ACCSTEP	number	The step to be added to the value of the acceleration parameter after each iteration.	0.04	none

Newton's Method Parameters

AUTONR	logical	Specifies that an automated Newton-Richardson procedure is used to reduce the number of LU decompositions per bias point when using Newton's method to obtain the solution.	False if **EJ.MOBIL** has been selected on the **MODELS** statement; otherwise, true.	
NRCRITER	number	The value which the ratio of the error norms from the two most recent Newton iterations must be smaller than in order to use the same Jacobian (LU decomposition) for the next Newton iteration.	0.1	none
ERR.RAT	number	The value that the ratio of the maximum **RHSNORM** continuity error to the maximum terminal current must be smaller than before the program stops on an iteration for which an LU decomposition was not performed.	1e-5	none
CONT.RHS	logical	Specifies that if the solution process starts to diverge, the electrode bias steps taken from the previous solution are reduced by the multiplicative factor **ACONTINU**. The solution process is considered to be diverging if the **RHSNORM** errors for Poisson's equation and both continuity equations all increase for **ITRHS** iterations. **synonym: CONTINUA**	false	
ITRHS	number	Number of iterations to be made with an increasing **RHSNORM** (see **CONT.RHS**).	1	none
CONT.PIV	logical	Specifies that if the numerical value of a pivot is approximately equal to zero, the electrode bias steps taken from the previous solution are reduced by the multiplicative factor **ACONTINU**.	true	
CONT.ITL	logical	Specifies that if a solution does not converge within **ITLIMIT** iterations, the electrode bias steps taken from the previous solution are reduced by the multiplicative factor **ACONTINU**.	true	

Parameter	Type	Definition	Default	Units
STOP.ITL	logical	Specifies that if a solution does not converge within **ITLIMIT** iterations, execution of the program is terminated.	false	
CONT.STK	logical	Specifies that if a stack overflow occurs (see **STACK**), that program execution should continue with the next non-**SOLVE** statement.		
STACK	number	The size of the stack which holds the bias conditions that have not converged when using a continuation method (**CONT.RHS**, **CONT.PIV**, or **CONT.ITL**). When this value is exceeded, execution of the program terminates if **CONT.STK** is not specified. If **CONT.STK** is specified, the program attempts to continue execution with the next non-**SOLVE** statement (Maximum is 10).	4	none
ACONTINU	number	The multiplicative factor used to reduce the electrode bias steps when **CONT.RHS**, **CONT.PIV**, or **CONT.ITL** is specified.	0.5	none
TAUTO	logical	Specifies that time steps are selected automatically by Medici from the local truncation error estimates. This is the default for both the first- and second-order schemes.	true	
2NDORDER	logical	Specifies that a variable order time discretization is used (maximum order is second). If this parameter is false, then a first-order time discretization is used.	true	
TOL.TIME	number	The maximum allowed local truncation error. To use dynamic error tolerance, set **TOL.TIME** to zero.	1e-2	none
L2NORM	logical	Specifies the error norms are L2 as opposed to infinity norms for calculating the time steps.	true	
DT.MIN	number	The minimum time step allowed.	1e-25	seconds
EXTRAPOL	logical	Specifies that a second-order extrapolation is used to compute initial guesses for successive time steps.	false	
CARR.MIN	number	Specifies the minimum number of carriers considered to be significant when calculating the **XNORM** for the continuity equation. If the number of electrons or holes at a node is less than **CARR.MIN**, the electron or hole continuity equation error, respectively, at the node is not included in the **XNORM** calculation.	0	$\#/cm^3$
CARR.FAC	number	The factor which is used to reduce the carrier concentration at a node instead of the actual update if the update would cause the concentration to be negative.	0.5	none
N.DVLIM	number	Specifies the maximum change in potential at any node in the device during a single Newton iteration.	2	volts
N.DVMAX	number	Sets the termination threshold for Newton's iteration. If the maximum potential update exceeds this threshold, the Newton iteration terminates according to the applied bias. If **N.DVMAX** is not set, the program calculates the termination threshold according to the applied bias.	none	volts
N.DAMP	logical	Specifies that a sophisticated Rose-Bank damping scheme be used during Newton iteration. Use of this parameter may help convergence when applying large bias steps or increasing the number of carriers.	false	

Energy Balance Parameters

ETX.TOLE	number	The carrier temperature update tolerance.	1e-2	none
ETR.TOLE	number	The carrier temperature equation error tolerance.	1e-18	W/micron

Parameter	Type	Definition	Default	Units
N.MAXBL	number	Maximum number of energy balance block iteration plug-ins.	25	none
N.MAXEB	number	Maximum number of energy balance iterations.	15	none

Lattice Temperature AAM Parameters

LTX.TOLE	number	The lattice temperature update tolerance. This parameter is only used with the Lattice Temperature AAM.	1e-3	none
LTR.TOLE	number	The lattice temperature equation error tolerance. This parameter is only used with the Lattice Temperature AAM.	le-11	W/micron
LTX.FACT	number	Factor by which **CX.TOLER** and **LTX.TOLE** are scaled for the intermediate step of the global method when solving the lattice temperature equation. This parameter is only used with the Lattice Temperature AAM.	100	none
LTR.FACT	number	Factor by which **CR.TOLER** and **LTR.TOLE** are scaled for the intermediate step of the global method when solving the lattice temperature equation. This parameter is only used with the Lattice Temperature AAM.	100	none
MAX.TEMP	number	Maximum lattice temperature allowed during a simulation. If the lattice temperature at any point within the device exceeds **MAX.TEMP** the program terminates the simulation. This parameter is only used with the Lattice Temperature AAM.	2000	degrees K

Direct Tunneling Parameters

DT.JACOB	logical	Specifies whether or not direct tunneling should be included in the jacobian.	true	none

Description

The **METHOD** statement sets parameters associated with a particular solution algorithm chosen in the **SYMBOLIC** statement. If no **METHOD** statement is specified in a simulation input file, the default values of all parameters are used. Whenever a **METHOD** statement is encountered, only the parameters which are actually specified are modified. All other parameters retain their previous values.

See Also... To further illustrate the **METHOD** statement, refer to:

- Input file *mdex1* in "Generation of the Simulation Structure" on page 4-2
- Input file *mdex2* in "Generation of the Simulation Structure" on page 5-1
- Most other examples where a solution is calculated

Solution Methods

Four linear solution methods are used in Medici, listed here in the order of increasing memory and CPU effort.

- ICCG—An iterative technique used only to solve Poisson's equation during the Gummel iteration. Convergence is not guaranteed.

- ILUCGS—An iterative technique used to solve linear equations in Newton's method or Poisson's or the continuity equations in Gummel's method. This method requires that a partial (incomplete) LU factorization be performed as a pre-conditioner prior to iteration. Convergence is not guaranteed.
- Bi-CGSTAB—An iterative method used to solve the augmented Jacobian matrix as a whole. Although slower than SOR at low frequencies, it is able to converge at higher frequencies. In most cases, bipolar transistors and MOSFETs can be analyzed well above their cutoff frequencies.
- DIRECT—Conventional Gaussian elimination. May be used to solve any of the linear equations. Requires that a full (complete) LU factorization be performed. This method always finds a solution but is expensive in CPU and memory requirements.

Note:

***ILUCGS** and **BICGS** are available as parameters on the **SYMBOLIC** statement.*

Convergence Criteria

This section details common convergence criteria. In addition, specific convergence criteria is discussed for the following:

- Energy balance solution
- Lattice temperature solution
- ILUCGS and Bi-CGSTAB
- Gummel's method
- Newton's method
- Continuation methods

Default

The convergence criterion established with **XRNORM** is used as the default. This implies that a solution is considered converged when *either* the **XNORM** tolerances or the **RHSNORM** tolerances are satisfied at every node in the device.

The iteration errors that are printed to the output file are the maximum potential and carrier concentration updates in the device (**XNORM**) excluding those nodes where the **RHSNORM** tolerances are already satisfied. A printed value of "0.0000E+00" indicates that the **RHSNORM** tolerances are satisfied at every node in the device.

The **XRNORM** criterion remains in effect until it is explicitly turned off by specifying **^XRNORM**. For example, to use the **RHSNORM** criterion instead of the **XRNORM** criterion, the following statement should be specified:

```
METHOD     ^XRNORM   RHSNORM
```

As a further example, the following statement causes the program to print the **XNORM** and **RHSNORM** errors to the output file, but the **XRNORM** errors are still used (by default) to determine convergence:

```
METHOD     XNORM   RHSNORM
```

Error Tolerances

The **XNORM** and **RHSNORM** error tolerances can be adjusted with the parameters **PX.TOLER**, **CX.TOLER**, **ETX.TOLE**, **LTX.TOLE**, **PR.TOLER**, **CR.TOLER**, **ETR.TOLE** and **LTR.TOLE**.

Note:

Specifying larger values for these parameters may decrease the number of iterations required for convergence, but may also decrease the accuracy of the resulting solution.

Energy Balance Solution

The accuracy and convergence of the energy balance solution is controlled by the following parameters.

- The relative accuracy of the carrier temperature is specified with **ETX.TOLE**.
- The right-hand-side norm tolerance for the energy balance equation is specified with **ETR.TOLE**.
- If neither **COUP.ELE** nor **COUP.HOL** is specified on the **SYMBOLIC** statement, a decoupled solution is performed.

  In this case, the convergence criterion specified with **ETX.TOLE** has to be satisfied both inside the energy balance iteration and in the outer loop of the block iteration (see Chapter 2, "Energy Balance Equations" on page 2-112).
- The number of decoupled energy balance solutions is limited by **N.MAXEB**.
- The maximum number of block iteration plug-ins is limited by **N.MAXBL**.

Lattice Temperature Solution

A further set of parameters used with the Lattice Temperature AAM controls the solution of the lattice temperature equation.

- The tolerance of the temperature update and the right-hand-side norm tolerance for the lattice heat flow equation is specified by **LTX.TOLE** and **LTR.TOLE**.
- If a decoupled solution is performed, i.e., **^COUP.LAT** is specified, the parameters **LTX.FACT** and **LTR.FACT** can be used to relax the convergence criteria for the inner loop lattice heat equation iteration.
- Only one of **COUP.LAT**, **COUP.ELE** and **COUP.HOL** can be specified.

ILUCGS and Bi-CGSTAB

The ILUCGS or Bi-CGSTAB method is considered to have converged if:

$$\frac{\left\|(LU)^{-1}r\right\|}{\left\|(LU)^{-1}b\right\|} < \texttt{ILU.TOL} \qquad \text{Equation 3-20}$$

where:

- b is the right hand side of $Ax = b$

- r is the residual defined by $r = b - Ax$

ILU.XTOL

An additional convergence criterion is placed on ILUCGS or Bi-CGSTAB by the **ILU.XTOL** parameter in which case ILUCGS or Bi-CGSTAB is considered to have converged if:

$$\left| 1 - \frac{\|x\|}{\|x'\|} \right| < \textbf{ILU.XTOL} \qquad \text{Equation 3-21}$$

where:

- x is the present solution found by ILUCGS or Bi-CGSTAB.
- x' is the solution from the previous ILUCGS or Bi-CGSTAB iteration.

In general, it is not necessary to specify this extra criterion and in any case, specifying **ILU.XTOL** < 1.0e-3 results in an increased number of ILUCGS or Bi-CGSTAB iterations on non-convergence.

Gummel's Method

Various methods are available to try to speed convergence when using Gummel's method for the solution. Among these are:

- ICCG
- The single-Poisson method (**SINGLEP**) with acceleration (**ACCELERA**)

For many cases ICCG provides the fastest convergence and should routinely be chosen.

Damping

When Gummel's method is being used, or when performing a Poisson-only solution (zero carriers), damping is used to stabilize the convergence by limiting the maximum potential change per iteration inside the device with **DVLIMIT**.

The default value of **DVLIMIT** is 1.0 V unless **^DAMPED** is specified, in which case it is 0.1 V.

Increasing **DVLIMIT** from its default value may result in faster simulations by allowing the potential at each node to be adjusted by a larger amount. However, if **DVLIMIT** is set too high, it may cause the convergence to be unstable.

In general, **DVLIMIT** should not be set higher than 10-20% of the bias step applied to an electrode. Larger values of **DVLIMIT** may be possible at high voltages.

A more sophisticated damping scheme than that described above is available by specifying **DAMPED**. This option is recommended for most situations.

Newton's Method

This section details the special parameters used to control the various convergence criteria for Newton's method. Use of Newton's method also allows the selection of various continuation methods (see "Continuation Methods" on page 3-148).

Speeding Convergence with AUTONR

It is possible to speed up convergence significantly by using the automated Newton-Richardson method parameter **AUTONR**. This attempts to reduce the number of times that the Jacobian matrix is refactored. When using Newton's method, **AUTONR** is used by default in all cases except when **EJ.MOBIL** has been selected on the **MODELS** statement.

Without **AUTONR**, the Jacobian matrix is refactored at every iteration. With **AUTONR**, the Jacobian matrix is not refactored if the error norms fall by more than the amount specified by **NRCRITER**. This is indicated by an asterisk (*) next to the iteration number in the output file.

Maximum Potential Update with N.DVLIM

The parameter **N.DVLIM** controls the maximum potential update during a single newton iteration. In many cases, reducing the size of **N.DVLIM** to about one third of the present bias step can aid convergence by damping large oscillations in the solution.

Selecting too small a value for **N.DVLIM**, however, can harm convergence since (assuming the previous solution is used as the initial guess) the number of Newton iterations required will be greater than the bias step specified divided by **N.DVLIM**.

Rose-Bank Damping

The parameter **N.DAMP** causes the program to use Rose-Bank damping during Newton iteration, which can help improve convergence in certain cases (particularly when using large bias steps or when increasing the number of carriers). For example, when going from a 0 carrier solution to a 1 or 2 carrier solution at the same bias.

Rare cases have been observed when damping can harm the process and cause a bias point to fail to converge. In most cases, when convergence fails while using **N.DAMP**, it also fails without **N.DAMP**. Convergence fails sooner with damping because the program stops iterating once the damping factor is less than 1e-4, rather than when 20 iterations are reached. This results in less total CPU time. The overhead for the damping is very slight, and in most cases it does not cause harm.

Continuation Methods

Medici allows continuation methods to be chosen when using the Newton solution technique. These methods may be used for the following:

- To adjust for bias steps that are too large by automatically reducing the step if convergence problems occur
- To adjust time steps in transient simulations if automatic time step selection is not being used

The criteria for performing a reduction of the bias or time step can be:

- An increase in the **RHSNORM** errors (**CONT.RHS**)
- An occurrence of a numerical pivot which is approximately zero (**CONT.PIV**)
- Failure to converge within **ITLIMIT** iterations (**CONT.ITL**)

In each case the multiplicative factor **ACONTINU** is used to reduce the step. The continuation methods specified with **CONT.ITL** and **CONT.PIV** are used by default. If **STOP.ITL** is selected, program execution terminates if the solution does not converge within **ITLIMIT** iterations.

Transient Simulation

The **METHOD** statement is also used to select parameters affecting transient simulations. Two time discretization schemes are available:

- A first-order backward difference scheme
- A sophisticated variable order method with a maximum order of 2 (the default)

When using the either scheme, time steps are selected automatically by Medici unless this capability is disabled by specifying **^TAUTO**.

SOLVE

The **SOLVE** statement instructs Medici to perform a solution for one or more specified bias points.

```
SOLVE

 Initial Guesses, Biasing, and Fermi Potentials
   [ {INITIAL | PREVIOUS | PROJECT | LOCAL | P.LOCAL} ]
   [ {V(name1)=<n> | I(name1)=<n> | T(name1)=<n> | Q(name1)=<n>} ]....
   [ {V(name200)=<n> | I(name200)=<n> | T(name200)=<n> | Q(name200)=<n>} ]
   [N.REGION=<c>] [N.BIAS=<a>]
   [P.REGION=<c>] [P.BIAS=<a>]

 Steady State Analysis Parameters
   [ {  ( ELECTROD=<c> {VSTEP=<n> | ISTEP=<n>} NSTEPS=<n> )

 Continuation Method Parameters
      | ( CONTINUE ELECTROD=<c> C.VSTEP=<n> [ C.AUTO [C.TOLER=<n>] ]
          [C.VMIN=<n>] [C.VMAX=<n>] [C.IMIN=<n>] [C.IMAX=<n>]
          [C.DVMAX=<n>] [C.RMAX]
        )
 Transient Analysis Parameters
      | ( TSTEP=<n> {TSTOP=<n> | NSTEPS=<n>} [TMULT=<n>]
          [ {RAMPTIME=<n> | ENDRAMP=<n>} ] [DT.MAX=<n>]
        )
     }
   ]
 Hot Carrier and Parasitic Analysis Parameters
   [IMPACT.I] [GATE.CUR] [DQDV]

 Programmable Device AAM Parameters
   [FN.CUR]

 Direct Tunneling Analysis Parameters
   [DT.CUR] [DT.METH=<n>] [DT.CBET] [DT.VBET]

 AC Small-Signal Analysis Parameters
   [ AC.ANALY FREQUENC=<n> [ FSTEP=<n> NFSTEP=<n> [MULT.FRE] ]
     [VSS=<n>] [TERMINAL=<c>]
     [S.OMEGA=<n>] [MAX.INNE=<n>] [TOLERANC=<n>] [HI.FREQ]
     [ S.PARAM [R.SPARA=<n>] ]
   ]
 AC Charge-Partition Analysis Parameters
   [ AC.CHARG [TERMINAL=<c>] ]
 Circuit Analysis AAM Parameters
   [ ELEMENT=<c> V.ELEMEN=<n> [VSTEP=<n> NSTEPS=<n>] ] [UIC]

 AC Analysis with a Circuit
   [ AC.ANALY FREQUENC=<n> AC.SOURC=<c>
     [ FSTEP=<n> NFSTEP=<n> [MULT.FRE] ]
   ]
```

(**SOLVE**, continued next page)

(**SOLVE**, continued from previous page)

```
 Output Choices
   [ OUT.FILE=<c> [SAVE.BIA]
     [ {  ( TIF [ALL] [BANDS] [CURRENTS] [GENERATI] [COMPONEN] )
        | ( [CURRENTS] [ASCII] [STRUCTUR=<c>] )
       }
     ]
   ]
 Optical Device AAM Parameters
   [ {  ( {  WAVE=<n>
           | ( [WAVE.STA=<n>] [WAVE.END=<n>] )
           | SPECTR
         }
       )
      | ( [FLUX=<n>]
         { [LAMBDA=<n>]
          | ( LAMBDA.S=<n> LAMBDA.E=<n> LAMBDA.N=<n> )
         }
       )
      | ( INTENSIT=<n> [INT.STEP=<n>] )
     }
   ]
   [L.MODULA  LSS=<n>]
```

Parameter	Type	Definition	Default	Units
Initial Guesses, Biasing, and Fermi Potentials				
INITIAL	logical	Specifies that the charge neutral assumption is used to compute the initial guess. This is the starting point for all device simulations.	True if no solution is available to use as an initial guess; otherwise, false.	
PREVIOUS	logical	Specifies that the previous solution is used as the initial guess. The previous solution is modified by setting the applied bias at the contacts. This is the default when a previous solution is available.	false	
PROJECT	logical	Specifies that a projection is used as the initial guess. This is the default when two previous solutions are available and the ratio of two successive bias steps is the same for each electrode where the bias has changed.	false	
LOCAL	logical	Specifies that local values of the quasi-Fermi potentials are used to compute an initial guess. This type of initial guess takes the previous solution, sets the applied bias, and then sets the majority carrier quasi-Fermi potentials to the applied bias throughout each region connected to an electrode by material of the same doping type.	false	
P.LOCAL	logical	Specifies that local quasi-Fermi potentials are used to compute the initial guess in heavily doped regions attached to electrodes, while the previous solution is used for the initial guess elsewhere. This type of initial guess may be helpful as an alternative to "previous" particularly when the heat equation is being solved.	False unless lattice temperature is being solved for.	

Parameter	Type	Definition	Default	Units
V(*name*)	number	The applied bias at the electrode specified by *name.* Any number of biases corresponding to electrodes can be specified on the same **SOLVE** statement.	Previous bias at electrode *name*	volts
I(*name*)	number	The terminal current at the electrode specified by *name*, if a current boundary condition was specified for this contact. Any number of terminal currents corresponding to electrodes can be specified on the same **SOLVE** statement.	Previous current at electrode *name*	amps/ micron
T(*name*)	number	The temperature at the thermal electrode specified by *name*. Values of temperature for multiple thermal electrodes can be specified on the same statement. This parameter is only used with the Lattice Temperature AAM.	Previous temperature at thermal electrode *name*	Kelvins
Q(*name*)	number	The charge at the electrode number specified by *name*, if a charge boundary condition was specified for this contact. The charge on multiple electrodes can be specified on the same statement. This parameter is only used with the Programmable Device AAM.	Previous charge at electrode *name*	Coulombs/ micron
N.REGION	char	The region names for which electron quasi-Fermi potentials are specified. If more than one region name is specified, separate their names with commas and enclose the entire group in parentheses (for example, "(substrate,drain)").	none	
N.BIAS	array	The electron quasi-Fermi potentials to use in the regions identified with **N.REGION**, if electrons are not being solved for. If more than one value is specified, separate each value with commas and enclose the entire group in parentheses (for example, "(5.0,3.0)"). Values specified here override any value established as a result of specifying **FIX.QF** on the **METHOD** statement.	Local quasi-Fermi potential based on bias and doping type.	volts
P.REGION	char	The region names for which hole quasi-Fermi potentials are specified. If more than one region name is specified, separate their names with commas and enclose the entire group in parentheses (for example, "(substrate,drain)").	none	
P.BIAS	array	The hole quasi-Fermi potentials to use in the regions identified with **P.REGION**, if holes are not being solved for. If more than one value is specified, separate each value with commas and enclose the entire group in parentheses (for example, "(5.0,3.0)"). Values specified here override any value established as a result of specifying **FIX.QF** on the **METHOD** statement.	Local quasi-Fermi potential based on bias and doping type.	volts

Steady State Analysis Parameters

ELECTROD	char	The name of an electrode for which the applied bias or terminal current is stepped. To step more than one electrode, separate their names with commas and enclose the entire group in parentheses (for example, "(drain,gate,source)").	none	
VSTEP	number	The increment for the bias applied to one or more electrodes as specified by **ELECTROD**. If this parameter is specified, the bias at the specified electrode(s) is incremented **NSTEPS** times.	0.0	volts
ISTEP	number	The increment for the terminal current at one or more electrodes as specified by **ELECTROD**. If this parameter is specified, the current at the specified electrode(s) is incremented **NSTEPS** times.	0.0	amp/ micron
NSTEPS	number	The number of bias steps, current steps, or time steps to be performed.	0	none

Continuation Method Parameters

CONTINUE	logical	Specifies that an automatic continuation procedure is used to trace I-V curves. This procedure automatically selects the bias step and switches from voltage to current boundary conditions as appropriate.	false	

Parameter	Type	Definition	Default	Units
C.VSTEP	number	Specifies the initial voltage step for the continuation method. If the value is > 0 the initial step is positive. If the value is less than 0 the initial steps in the negative voltage direction. **synonym: C.LENGTH**	none	volts
C.AUTO	logical	Specifies that automatic bias step selection is performed with the continuation method.	true	
C.TOLER	number	Local truncation error tolerance for continuation method. A smaller value causes the program to use more bias steps and produce a finer curve at the expense of CPU time. This parameter *does not* affect the accuracy of the computed points, only the spacing between them.	0.05	none
C.VMIN	number	Minimum terminating voltage for the continuation method. If the continuation method is tracing an I-V curve and the bias voltage at the electrode becomes less than this value, the tracing process is considered complete and the continuation terminates.	-5.0	volts
C.VMAX	number	Maximum terminating voltage for the continuation method. If the continuation method is tracing an I-V curve and the bias voltage at the electrode becomes greater than this value, the tracing process is considered complete and the continuation terminates.	5.0	volts
C.IMIN	number	Minimum terminating current for the continuation method. If the continuation method is tracing an I-V curve and the bias current at the electrode becomes less than this value, the tracing process is considered complete and the continuation terminates.	-1.0e-4	amps/ micron
C.IMAX	number	Maximum terminating current for the continuation method. If the continuation method is tracing an I-V curve and the bias current at the electrode becomes greater than this value, the tracing process is considered complete and the continuation terminates.	1.0e-4	amps/ micron
C.DVMAX	number	Maximum potential update allowed during continuation method. If the potential update exceeds this limit, the bias step is immediately reduced and the program tries again. This is useful because it stops the program from wasting time trying to solve for bias points that are not likely to converge. Since projection is used to find the initial guess during continuation, the potential updates are normally quite small and large updates indicate a possible problem.	50.0	kT/q
C.RMAX	number	Maximum value of external resistance allowed during continuation. If not specified, it is dynamically determined by the program.		ohms

Transient Analysis Parameters

TSTEP	number	The time step between solutions. For simulations using automatic time step selection (see the **METHOD** statement), **TSTEP** is used to select the first time step only. All other time steps are chosen automatically by Medici.	none	seconds
TSTOP	number	The end of the time interval to be simulated.	none	seconds
TMULT	number	The multiplicative factor used to vary the size of successive time steps during a transient simulation when automatic time step selection is not used.	1.0	none
RAMPTIME	number	A time interval over which any bias change is applied as a linear ramp. If the ramp begins at time $t = t_0$, it ends at $t = t_0 +$ **RAMPTIME**.	0.0	seconds
ENDRAMP	number	The ending time for a period over which any bias change is applied as a linear ramp. If the ramp begins at time $t = t_0$, it ends at $t =$ **ENDRAMP**.	0.0	seconds
DT.MAX	number	Maximum time step as a ratio of the total simulation interval.	0.25	none

Parameter	Type	Definition	Default	Units
Hot Carrier and Parasitic Analysis Parameters				
IMPACT.I	logical	Specifies that an impact ionization analysis is performed after each bias or time point is solved for.	false	
GATE.CUR	logical	Specifies that a gate current analysis is performed after each bias or time point is solved for.	false	
DQDV	logical	Causes Medici to calculate the capacitance at the electrodes. The capacitance is calculated by dividing the change in terminal charge by the change in voltage. This procedure gives one column of the capacitance matrix. This method can only be used if the voltage at one electrode is changed at a time, and if a previous solution resides in memory.	false	
Programmable Device AAM Parameters				
FN.CUR	logical	Specifies that Fowler-Nordheim tunneling current is be calculated. During a transient analysis, the charge on any floating regions affected by this current is updated automatically. This parameter is only used with the Programmable Device AAM.	false	
Direct Tunneling Analysis Parameters				
DT.CUR	logical	Specifies that the direct tunneling current through insulating layers is calculated after the solution is found.	false	
DT.METH	number	The method for evaluating the direct tunneling current. 1=analytical, 2=WKB, 3=Gundlach.	2	
DT.CBET	logical	Specifies that conduction band electron tunneling should be included in the direct tunneling.	true	
DT.VBET	logical	Specifies that valence band electron tunneling should be included in the direct tunneling.	false	
AC Small-Signal Analysis Parameters				
AC.ANALY	logical	Specifies that AC sinusoidal small-signal analysis is performed after the DC condition is solved for.	false	
FREQUENC	number	The frequency at which the AC small-signal analysis is performed.	none	Hz
FSTEP	number	The increment for frequency when performing an AC small-signal analysis at multiple frequencies. If **MULT.FRE** is not specified, the frequency for each analysis is obtained by adding **FSTEP** to the previous value of frequency. If **MULT.FRE** is specified, the frequency for each analysis is obtained by multiplying the previous value of frequency by **FSTEP**.	0.0	None if **MULT.FRE** is specified; otherwise, Hz.
NFSTEP	number	The number of additional frequencies at which an AC small-signal analysis is performed.	0	none
MULT.FRE	logical	Specifies that **FSTEP** is a multiplicative factor for incrementing frequency.	false	
VSS	number	The magnitude of the applied small-signal bias.	$0.1 \cdot KT/q$	volts
TERMINAL	char	The electrode(s) to which the AC bias is applied. More than one electrode may be specified, but each case is solved separately. To specify more than one electrode, separate them with commas, and enclose the entire group within parentheses (for example, "`(drain,gate,source)`").	all electrodes	

Parameter	Type	Definition	Default	Units
S.OMEGA	number	The SOR relaxation parameter used when solving the linear system during an AC small-signal analysis. This parameter is only used if **HI.FREQ** is false.	1.0	none
MAX.INNE	number	The maximum number of iterations when performing an AC small-signal analysis. **Default:** 25 or $2 \times max(35, \sqrt{3 \times nodes})$ if using **HI.FREQ**	See Definition	none
TOLERANC	number	The SOR convergence criterion when performing an AC small-signal analysis. If **HI.FREQ** is true, then this parameter specifies the Bi-CGSTAB convergence criterion when performing an AC small-signal analysis. The error norm is calculated as $\left\|(X_i - X_i^{'})/(X_i^{'})\right\|$ and represents the component-wise relative error in the solution vector. For a complete description please refer to Chapter 2, "ILUCGS Solver" on page 2-72.	1e-5	none
HI.FREQ	logical	Specifies that high frequency AC analysis mode is used. This option allows AC analysis at higher frequencies than the standard method of SOR iteration but it is much slower. When **HI.FREQ** is true, the AC system is solved using either Bi-CGSTAB or a direct method with iterative correction depending on whether a direct method or iterative method, respectively, is used for the DC solution. If **HI.FREQ** is not specified, the program tries SOR first, and if this method fails it automatically switches to the **HI.FREQ** method.	false	
S.PARAM	logical	Specifies that S-parameters should be calculated. If S-parameters are to be calculated then **TERMINAL** should be used to specify which electrodes are terminal "1" and terminal "2" (see below).	false	
R.SPARA	number	Transmission line impedance used in S-parameter calculation. Input the actual transmission line impedance multiplied by the device width.	50	Ohms × Microns
AC Charge Analysis Parameter				
AC.CHARG	logical	Specifies that AC charge partition analysis is performed after a DC solution is obtained. This analysis avoids a problem of the standard AC analysis that produces abnormal capacitance values at the terminals of significant DC current.	false	
Circuit Analysis AAM Parameters				
ELEMENT	char	A voltage or current source that has its voltage or current set to the value specified with the **V.ELEMEN** parameter. This parameter is only used with the Circuit Analysis AAM.	none	
V.ELEMEN	number	Specifies the voltage or current to be applied to the circuit element specified by the **ELEMENT** parameter. This value remains in effect for the remainder of the simulation. This parameter is only used with the Circuit Analysis AAM.	none	volts or amps
UIC	logical	Specifies that the voltages on the circuit nodes is forced to remain at the values specified on the **.IC** statement. This parameter is only used with the Circuit Analysis AAM.	false	
AC Analysis with a Circuit				
AC.SOURC	char	Specifies the name of the AC small signal voltage or current source when AC analysis is performed with a circuit.	none	
Output Choices				
OUT.FILE	char	The identifier for the file in which the solution information is saved. If multiple solutions are obtained from a single **SOLVE** statement, the last non-blank character of the supplied identifier is incremented in succession, resulting in a unique file for each solution. If necessary, the incrementing of the file identifiers extended to the characters prior to the last. **synonym: OUTFILE**	solution is not saved	

Parameter	Type	Definition	Default	Units
CURRENTS	logical	Specifies that electron, hole, and displacement current densities are written to the output file. For TIF files, the output also includes carrier velocities. For non-TIF files, the stored information, which also includes electric field and recombination, is only used with the **DIFFEREN** option on the **LOAD** statement.	false	
ASCII	logical	Specifies that the solution file is written as a formatted file.	false	
SAVE.BIA	logical	Specifies that solution files are only saved for the biases that **OUT.FILE** is specified for. That is, if a bias step is reduced because of convergence problems, solution files are not saved at the these additional bias points.	false	
STRUCTUR	char	Specifies which device data to write to the solution file. This parameter is only used with the Circuit Analysis AAM.	all devices	
TIF	logical	Specifies that the TIF format is used for the output file. The output automatically includes basic physical quantities such as doping, potential, carrier concentrations, carrier and lattice temperatures, electric field and total current density.	false	
ALL	logical	Specifies that all available physical quantities are written to the TIF file.	false	
BANDS	logical	Specifies that physical quantities associated with the band structure of the device are written to the TIF file. These include electron and hole quasi-Fermi potentials, and valence band, conduction band, and vacuum level potentials.	false	
GENERATI	logical	Specifies that impact ionization generation, band-to-band tunneling generation, photogeneration, and recombination are written to the TIF file.	false	
COMPONEN	logical	Specifies that components of vector quantities are written to the TIF file in addition to the magnitudes of these quantities.	false	

Optical Device Parameters

WAVE	number	Specifies the wavelength number from the sampling of wavelengths specified with the **WAVE.STA**, **WAVE.END**, and **WAVE.NUM** parameters on the **PHOTOGEN** statement. The charge generated within the device structure is only due to the selected wavelength number. If this parameter is not specified, the generation is due to the entire spectrum.	none	
WAVE.STA	number	Specifies the first of a series of wavelength numbers taken from the sampling of wavelengths specified with the **WAVE.STA**, **WAVE.END**, and **WAVE.NUM** parameters on the **PHOTOGEN** statement. A solution is performed for each wavelength number from **WAV.STA** through **WAV.END**. The charge generated within the device structure is only due to the specific wavelength number under consideration.	1	
WAVE.END	number	Specifies the last of a series of wavelength numbers taken from the sampling of wavelengths specified with the **WAVE.STA**, **WAVE.END**, and **WAVE.NUM** parameters on the **PHOTOGEN** statement. A solution is performed for each wavelength number from **WAV.STA** through **WAV.END.** The charge generated within the device structure is only due to the specific wavelength number under consideration.	**WAVE.NUM** from the **PHOTOGEN** statement	
SPECTR	logical	Specifies that a solution is to be performed for each of the wavelength numbers taken from the sampling of wavelengths specified with the **WAVE.STA**, **WAVE.END**, and **WAVE.NUM** parameters on the **PHOTOGEN** statement. The charge generated within the device structure is only due to the specific wavelength number under consideration.	false	
FLUX	number	Specifies that **PHOTOGEN** and **RAYTRACE** are to be reinitiated with the given light **FLUX** while the other parameters remain unchanged. The solutions are to be obtained for the photogeneration associated with this new wavelength. This is only for the monochrome ray.	none	photons/ cm^2-sec

Parameter	Type	Definition	Default	Units
LAMBDA	number	Specifies the wavelength for which ray tracing and photogeneration are reinitiated while the other parameters remain unchanged. The solutions are obtained using the photogeneration associated with this new wavelength.	none	micron
LAMBDA.S	number	Specifies the minimum wavelength for which ray tracing and photogeneration are reinitiated while the other parameters remain unchanged. Spectral simulations are reperformed.	none	micron
LAMBDA.E	number	Specifies the maximum wavelength for which ray tracing and photogeneration are reinitiated while the other parameters remain unchanged. Spectral simulations are reperformed.	none	micron
LAMBDA.N	number	Specifies the number of wavelengths between **LAMBDA.S** and **LAMBDA.E** for which ray tracing and photogeneration are reinitiated while the other parameters remain unchanged. Spectral simulations are reperformed.	none	none
INTENSIT	number	The light intensity to use for a new calculation of ray tracing and photogeneration. Unspecified ray tracing quantities retain the values specified on the **PHOTOGEN** statement. Solutions are to be obtained for the photogeneration associated with this new intensity.	none	Watts/ cm^2
INT.STEP	number	The light intensity step to use for new calculations of ray tracing and photogeneration. After an initial solution with the intensity given by **INTENSIT**, the intensity will be incremented by **INT.STEP** for each successive calculation for a total of **NSTEPS** additional steps.	none	Watts/ cm^2
L.MODULA	logical	Specifies that light modulation analysis is performed after the DC condition is solved for.	false	
LSS	logical	The magnitude of the applied small signal light intensity to be used with light modulation analysis.	none	Watts/ cm^2

Description

The **SOLVE** statement performs a solution for one or more specified bias points. The following sections discuss the various elements of the **SOLVE** statement including:

- Initial guesses
- Bias selection
- Circuit analysis
- Continuation method
- Transient simulations
- Post-Processing impact ionization
- Gate current
- AC small-signal analysis
- Optical analysis

See Also...

To further illustrate the **SOLVE** statement, refer to:

- Input file *mdex1* in "Simulation of Gate Characteristics" on page 4-11
- Input file *mdex1g* in "Simulation of Gate Characteristics" on page 4-11
- Input file *mdex1d* in "Simulation of Drain Characteristics" on page 4-12

- Every other example where a solution is calculated

Initial Guesses

Medici automatically uses **INITIAL** as the initial guess for the first bias point for a given structure. For this bias point, 0 volts is assumed for any electrode where voltage is not specified.

If a previous solution exists, Medici uses it as the initial guess for the next solution. If two previous solutions are present and equivalent bias steps were taken for any electrode biases that were changed, a projection is used to obtain an improved initial guess for the next solution.

Bias Selection

The boundary conditions for the simulation to be performed are set by specifying the electrode applied biases:

- **V(DRAIN)**, **V(GATE)**, ..., **V(WHAT_EVER)**

or the terminal currents:

- **I(DRAIN)**, **I(SOURCE)**, ..., **I(ENOUGH)** at the contacts.

If terminal currents are specified at the contact, you should have previously specified this contact as one where current boundary conditions are to apply (see the **CONTACT** statement). If an electrode boundary condition is not specified, the previous bias or current is used by default.

Multiple Solutions

Multiple solutions with one **SOLVE** statement can be accomplished by specifying:

- Either the voltage or current step (**VSTEP** or **ISTEP**)
- The electrode(s) to be stepped (**ELECTROD**)
- The number of additional solutions to be performed (**NSTEPS**)

This is particularly convenient for obtaining I-V characteristics.

Synchronization in Saving Solutions

The **SAVE.BIA** parameter is useful in order to ensure synchronization of the reading and writing of solution files. This parameter forces writing of solution files only at bias, current, or time points actually specified. Solutions generated

during a step cutback (due to nonconvergence) are not saved. For example, the following code fragment attempts to sweep multiple drain curves:

```
SOLVE  V(gate)=0  V(source)=0  V(drain)=0

$ Bias up the gate.
SOLVE  ELECTROD=gate  VSTEP=0.1  NSTEP=10  OUT.FILE=SOL01

$ Drain curves.
LOOP   STEPS=10
  ASSIGN  NAME=SFX  C.VAL=01  DELTA=1
  LOAD    IN.FILE="SOL"@SFX
  SOLVE  ELECTROD=DRAIN  VSTEP=0.5 NSTEP=20
L.END
```

If any bias point from the second **SOLVE** statement failed, then the filenames of solutions written would lose their intended correspondence to voltage (i.e., **SOLO1** $\rightarrow 0.1\text{v}$, **SOLO5** $\rightarrow 0.5\text{v}$). In order to enforce the correspondence, the **SAVE.BIA** parameter should be added to the second **SOLVE** statement:

```
SOLVE ELECT=gate VSTEP=0.1 NSTEP=10 OUT.F=SOL01 SAV.BIA
```

Circuit Analysis Parameters

Steady state or transient analysis may also be performed on circuits. If steady state analysis is performed then the **ELEMENT** parameter is used to specify the element which is to be altered. This element may be a voltage source, current source or resistor. The value to be used for the element is specified with the **V.ELEMEN** parameter.

Multiple steps may be taken using **VSTEP** to determine the step size and **NSTEPS** to determine the number of steps.

Note:

The final value specified on the solve replaces the original value of the element.

For example, the following two **SOLVE** statements step source VCC and VDD from 0 to 5 and from 0 to 15 volts respectively. At the end of the simulation, VCC=5 and VDD=15 volts regardless of their original values of 7 and 3 volts.

```
VDD 1 0 7
VCC 2 0 3
SOLVE ELEMENT=VCC V.ELEMEN=0 VSTEP=1 NSTEP=5
SOLVE ELEMENT=VDD V.ELEMEN=0 VSTEP=3 NSTEP=5
```

Transient analysis with a circuit is very straightforward and is the same as simulation without circuit (see below). Voltage and current sources take on their time dependent values (see the V and I element).

Continuation Method

The continuation method can be used to trace difficult I-V characteristics such as those due to snap-back or latch-up. The continuation method automatically selects voltage or current boundary conditions based on the relative slope of the I-V characteristics and automatically selects bias points to resolve interesting features (refer to Chapter 2, "Continuation Method" on page 2-66 for more details).

User-Input

You are required to supply the following:

- The electrode (only one at a time) for which voltage and current is incremented
- The initial bias step to use (subsequent ones are selected by the program)

It is also wise to specify the terminating values for the voltage or current, although the defaults may be sufficient in some cases.

For example to step the drain from 1V to 10V with a maximum current of 1e-2 and an initial bias step of 0.5V, you would specify:

```
SOLVE   V(drain)=1 CONTINU ELECT=drain C.VSTEP=0.5 C.VMAX=10
+       C.IMAX=1E-2
```

Likewise, to start at 2V and to step to -5V with an initial voltage step of 0.3V with minimum and maximum currents of -3e-4 and 1e-5, respectively, the following statement could be used:

```
SOLVE   V(drain)=2  CONTINU  ELECT=drain  C.VSTEP=-0.3
+       C.VMIN=-5   C.VMAX=2  C.IMIN=-3e-4  C.IMAX=1e-5
```

Trace Back Problems

Occasionally the continuation method becomes confused at a sharp bend in an IV curve and traces back along the same path which it came up (see Figure 3-24). This problem can often be cured simply by reducing the continuation tolerance **C.TOLER** to a smaller value such as 0.01.

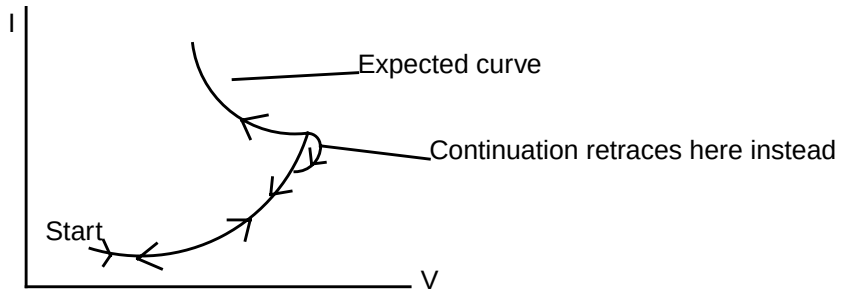


Figure 3-24 Continuation method becomes confused and retraces same curve

Transient Simulations

The Newton solution method must be used when performing a transient analysis. The use of the parameters relating to a transient analysis depend on the type of transient analysis which is being performed (see the **METHOD** statement).

A transient analysis is performed by specifying:

- The size of the initial time step with **TSTEP**
- The stopping time with **TSTOP**.

All intermediate time steps are calculated automatically, based on the size of the local truncation error. When automatic time step selection is not used, **TMULT** can be specified to increase or decrease the time step size for all successive steps.

Ramped Voltage or Current

A ramped voltage or current can be applied using one of two parameters:

- **RAMPTIME** specifies a time interval over which any bias change specified on the **SOLVE** statement is applied as a linear ramp.
- **ENDRAMP** causes the linear ramp to begin at the simulation time when the **SOLVE** statement is encountered, and ends at time t = **ENDRAMP**.

Post-Processing Impact Ionization

An impact ionization analysis is performed after each solution for which the parameter **IMPACT.I** is specified. This analysis calculates the generation rate at each node of the simulation mesh based on the electric field and current densities at the most recently solved for bias or time point.

The generation rate is integrated over the entire device to arrive at a value for the total impact ionization current. For a MOS device, this may be interpreted as the substrate current due to impact ionization. This analysis also gives the location in the device structure where the generation rate is maximum, including the magnitude of electric field and current density at that location.

Gate Current

A gate current analysis is performed after each solution for which the parameter **GATE.CUR** is specified. This analysis calculates the electron, hole, and total current injected into all insulators present in the structure due to carriers capable of surmounting the insulator potential barrier.

Results of the gate current analysis are reported in the output file if the gate current is not negligible. An example of the output is:

```
Hot carrier injection analysis:
------------------------------

Electrode   Elec. Curr.     Hole Curr.    Tot. Current
              (A/um)          (A/um)         (A/um)
---------   ------------   ------------   ------------
Float_Gate  -1.5315E-11     0.0000E+00    -1.5315E-11
```

The hot carrier injection analysis reports that electrons are injected onto the floating gate with the current density of 1.5315E-11.

Fowler-Nordheim Tunneling Analysis

In addition to the hot-carrier contribution to the gate current turned on by the parameter **GATE.CUR**, a tunneling current analysis based on the Fowler-Nordheim model is performed after each solution for which **FN.CUR** is specified.

Results of the simulation are reported in the output file if the predicted current is not negligible. An example of the output is:

```
Fowler-Nordheim Tunneling Analysis:
-----------------------------------

Electrode        Tun. Current
                   (A/um)
---------        ------------
Float_Gate         7.6131E-12
S: Source         -7.6131E-12
```

In the example shown, tunneling occurs between the floating gate and a semiconductor region identified by "s:" before the number of the electrode attached to this region (in this case the source of the flash EEPROM). If the region is attached to more than one electrode, only the first two electrodes are reported.

Direct Tunneling Analysis

As a substitute for Fowler-Nordheim tunneling analysis for very thin insulating layers, **DT.CUR** can be specified instead of **FN.CUR** to activate the direct tunneling model. As for the **FN.CUR** model, after each solution is found, the direct tunneling current injected into each electrode is calculated and written to the output file. For an example of using the direct tunneling mode, see the input file *mdex1dt* in "Direct Tunneling Analysis in a N-MOSCAP" on page 4-39

AC Small-Signal Analysis

An AC sinusoidal small-signal analysis is performed after each DC solution whenever the parameter **AC.ANALY** is specified. The following parameters are used to delineate this process:

- The frequency at which to perform the analysis must be specified with **FREQUENC**.
- The analysis can be performed at a number of different frequencies (using the same DC solution) by using the parameters **FSTEP**, **NFSTEP**.

  Optionally, use **MULT.FRE** to increment the initial frequency by a multiplicative factor.
- Specify the magnitude of applied small-signal bias with the **VSS** parameter.

  This bias is applied separately to all contacts (the default) but may be applied only to selected contacts using the **TERMINAL** parameter.
- For high frequencies (approaching cutoff), if it is necessary to use a value of the SOR relaxation parameter less than unity, use the **S.OMEGA** parameter.
- To increase the value for the maximum number of SOR iterations allowed, use **MAX.INNE**.

Note:

The Newton solution method is required when performing AC small-signal analysis.

S-Parameters

If S-parameters are requested, the program calculates these from the Y-parameters calculated by AC analysis.

- **TERMINAL** is used to specify which of the device electrodes are used for the S-parameter analysis (maximum of 2).
- **R.SPARA** is used to specify the characteristic transmission line impedance.

In order to obtain the correct S-parameters it is important to multiply the characteristic transmission line impedance by the actual device width (one micron). Note that Medici's default device width is one micron. As an example, if S-parameters for a BJT with width of 20 microns (in the z direction) are to be calculated at 100Mhz in the common emitter configuration with a characteristic transmission line impedance of 50 ohms:

```
SOLVE AC.ANAL FREQ=1e8 TERM=(BASE,COLLECT) S.PARA R.SPAR=50*20
```

AC Analysis with Circuit Simulation

If a simulation is performed with a circuit, then AC analysis may also be performed. The frequency and the source (either voltage or current) to which the AC voltage are applied are specified using the **FREQUENC** and **AC.SOURC** parameters.

The program then calculates the small signal voltages at all nodes in the circuit and the small-signal currents in all inductors and voltage sources. For example, to apply an AC voltage at source VIN at frequencies of 1e6, 1e7, 1e8, and 1e9 Hz:

```
SOLVE AC.ANAL FREQ=1e6 AC.SOURC=VIN FSTEP=10 MULT NFSTEP=3
```

Optical Analysis

This section provides two examples of using the **SOLVE** statement in conjunction with ray tracing.

- Monochrome light DC response
- Multi-spectral light DC response

The light modulation analysis capability is also briefly described.

Monochrome Light DC Response

The following four **SOLVE** statements specify that solutions are to be obtained for the specified optical parameters:

```
SOLVE LAMBDA=0.7
SOLVE INTENSI=10
SOLVE FLUX=1e17
SOLVE LAMBDA=0.6 FLUX=1e20
```

For the first three **SOLVE** statements, ray tracing and the photogeneration calculation are repeated with the new wavelength (**LAMBDA**), intensity (**INTENSIT**) or photon flux (**FLUX**) as specified. The final **SOLVE** statement requests both a new wavelength and a new photon flux. For all four of the statements, unspecified ray tracing quantities retain their values from the previous **PHOTOGEN** statement.

Multi-Spectral Light DC Response

This example is similar to the monochrome example except that a spectral intensity file *AM0.DAT* (representing the AM0 spectrum of sunlight outside the atmosphere) is used:

```
PHOTOGEN  RAYTRACE  SP.FILE=AM0.DAT  WAVE.SCA=1E-3
+         WAVE.ST=0.2  WAVE.EN=0.8  WAVE.NUM=6
+         X.ORG=5  Y.ORG=190  ANGLE=-90
+         RAY.WIDT=10  RAY.NUM=1  TRANSPAR
+         INT.RATI=1E-2
SYMBOLIC  NEWTON  CARRIERS=2
SOLVE
SOLVE     WAVE=5
SOLVE     LAMBDA=0.55
SOLVE     LAMBDA.S=0.4  LAMBDA.E=0.6  LAMBDA.N=4
```

The above statements first perform photogeneration, ray tracing, and a solution with parameters as given on the **PHOTOGEN** statement. A solution is then obtained using only the carriers generated by the fifth wavelength from the sampling specified on the **PHOTOGEN** statement. Ray tracing is not repeated for this case, since the information is already available from the previous calculation.

The next **SOLVE** statement specifies **LAMBDA**=0.55. For this solution, ray tracing using the new wavelength is repeated, interpolating from the information in the spectral intensity file, *AM0.DATA,* if necessary. The calculation is repeated as if there were a new **PHOTOGEN** statement.

The last **SOLVE** statement is similar to the previous one, except that new calculations are requested for four different wavelengths between 0.4 μm and 0.6 μm.

Light Modulation Analysis

Light modulation analysis is requested by specifying the parameter **L.MODULA**. After a DC solution with photogeneration is obtained, light modulation analysis induces a sinusoidal optical generation at each node i such that:

$$G_{opt,i} = G_{opt,i0} + \tilde{G}_{opt,i}\exp(jwt)$$

where $G_{opt,i0}$ is the existing optical generation rate and $\tilde{G}_{opt,i}$ is the magnitude of the sinusoidal generation that depends on the light modulation magnitude specified with the parameter **LSS**. The frequency at which to perform the analysis, **FREQUENC**, must be specified. The analysis can be performed at different frequencies by using the parameters **FSTEP**, **NFSTEP**, and **MULT.FRE**.

3.3 Input/Output

The following statements print and plot results, write results to files, or read solutions from files.

Statement	Definition	Page
EXTRACT	Extracts selected data over device cross-sections.	3-166
PRINT	Prints values of a quantity over a device cross-section.	3-179
PLOT.1D	Plots a quantity along a line through the structure; plots terminal characteristics from data in a log file.	3-182
PLOT.2D	Initializes graphics display device; plots device boundaries, junctions, and depletion edges in two dimensions.	3-195
PLOT.3D	Initiates three-dimensional plots.	3-201
3D.SURFACE	Performs a three-dimensional surface projection plot.	3-209
CONTOUR	Plots two-dimensional contours of a quantity.	3-211
VECTOR	Plots current and field vectors over a device cross-section.	3-218
FILL	Fills areas of a two-dimensional plot.	3-223
E.LINE	Locates and plots potential gradient paths as part of a 2D plot; calculates and plots quantities along potential gradient paths as part of a 1D plot.	3-226
LABEL	Plots a line or character string on a 1D or 2D plot.	3-232
LOG	Specifies files for storing terminal and user-defined data.	3-237
LOAD	Reads a solution stored in a file.	3-243
SAVE	Writes solution or mesh information to a file.	3-246

EXTRACT

The **EXTRACT** statement extracts selected data from the solution over a specified cross-section of the device.

```
EXTRACT

 Extraction Using Names and Expressions
   {  ( EXPRESSI=<c> NAME=<c> [UNITS=<c>] [CONDITIO=<c>] [INITIAL=<n>]
        [ {AT.BIAS | NOW} ] [OUT.FILE=<c> TWB] [CLEAR] [PRINT]

 Optimization Using Targets and Expressions
        [ TARGET=<n> [WEIGHT=<n>] [MIN.REL=<n>] [MIN.ABS=<n>]
          [TARTOL=<n>] [TARREL=<n>]
        ]
      )

 Extract Physical Quantities from Solution
    | ( {  NET.CHAR | NET.CARR | ELECTRON | HOLE | RECOMBIN | IONIZATI
         | RESISTAN | N.RESIST | P.RESIST | ( METAL.CH CONTACT=<c> )
         | ( {N.CURREN | P.CURREN} {CONTACT=<c> | REGIONS=<c>} )
         | II.GENER | (SHEET.RE X.POINT=<n>)
        }
        [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
        [OUT.FILE=<c>]

 Device to Extract (Circuit Analysis AAM)
        [STRUCTUR=<c>]
      )

 Extract MOS Device Parameters
    | ( MOS.PARA [DRAIN=<c>] [GATE=<c>] [IN.FILE=<c>] [I.DRAIN=<n>]
        [OUT.FILE=<c> [TWB] ] [CONDITIO=<c>] [APPLIED]
      )
   }
```

Parameter	Type	Definition	Default	Units
Extraction Using Names and Expressions				
NAME	char	Name of the assigned variable where the numeric value calculated by **EXPRESSI** is to be stored.	none	
EXPRESSI	char	Specifies the numeric character expression which is to be evaluated. This expression may contain assigned variables or solution variables.	none	
UNITS	char	Units to be used for the extracted quantity. These are used during plotting.	none	
CONDITIO	char	Specifies a numeric character expression which must evaluate to "TRUE" before **EXPRESSI** is evaluated.	TRUE	

Parameter	Type	Definition	Default	Units
INITIAL	number	Specifies the initial value to use during evaluation of **EXPRESSI**. This parameter is useful mainly for accumulator type expression such as: **EXTRACT NAME=DOG EXPRESS=MAX(@DOG;@TL) INITIAL=0**. Here the variable DOG must be assigned some value at the start of evaluation, its value comes from **INITIAL=0**. **synonym: INITIAL.V**.	0	none
AT.BIAS	logical	Specifies that the expressions **CONDITIO** and **EXPRESSI** are evaluated at the conclusion of each bias / time point.	TRUE, unless **NOW**=TRUE	
NOW	logical	Specifies that the expressions **CONDITIO** and **EXPRESSI** are evaluated when the **EXTRACT** statement is encountered. If this parameter is not specified, the extraction is performed after each **SOLVE** step or when a **PLOT.1D** statement is encountered.	false	
OUT.FILE	char	Specifies a file to which **TARGET** information is written to in **TWB** format.	none	
TWB	logical	Specifies that the result calculated by **EXPRESSI** is to be stored in the file specified by **OUT.FILE** in the format used by **TWB**. **synonym: CAESAR**.	false	
CLEAR	logical	Specifies that all previous **EXTRACT** expressions are deleted.	false	
PRINT	logical	Specifies that the value of the extracted expression is displayed on the user's terminal and printed to the standard output file.	false	
Optimization Using Targets and Expressions				
TARGET	number	The desired value of the target being defined for an optimization loop. The optimization attempts to match the extracted value with the value of this parameter.	none	Defined by the extracted value.
WEIGHT	number	The weighting factor applied to the target being defined for optimization. The weights are used to control the relative importance of the individual targets in calculation of error during optimization.	1.0	none
MIN.REL	number	The minimum target ratio for which relative error is used to calculate error during optimization.	1e-2	none
MIN.ABS	number	The minimum target value for which relative error is used to calculate the error during optimization.	1e-10	none
TARTOL	number	The target tolerance used during optimization.	5	%
TARREL	number	The relative target tolerance used during optimization (see below).	0.1	%
Extract Physical Quantities from Solution				
NET.CHAR	logical	Specifies that the integrated net charge is extracted.	false	
NET.CARR	logical	Specifies that the integrated net carrier concentration is extracted.	false	
ELECTRON	logical	Specifies that the integrated electron concentration is extracted.	false	
HOLE	logical	Specifies that the integrated hole concentration is extracted.	false	
RECOMBIN	logical	Specifies that the integrated recombination rate is extracted.	false	
IONIZATI	logical	Specifies that the maximum electron and hole ionization integrals are extracted.	false	

Parameter	Type	Definition	Default	Units
RESISTAN	logical	Specifies that the resistance (including both electrons and holes) of a cross-section is extracted. The current flow is assumed to be perpendicular to the xy plane. The result is given in Ohms per micron of distance perpendicular to the xy plane.	false	
N.RESIST	logical	Specifies that the n-resistance of a cross-section is extracted. The current flow is assumed to be perpendicular to the xy plane. The result is given in Ohms per micron of distance perpendicular to the xy plane.	false	
P.RESIST	logical	Specifies that the p-resistance of a cross-section is extracted. The current flow is assumed to be perpendicular to the xy plane. The result is given in Ohms per micron of distance perpendicular to the xy plane.	false	
METAL.CH	logical	Specifies that the integrated charge on an electrode is extracted.	false	
CONTACT	char	The electrode name over which the integration takes place when **METAL.CH**, **N.CURREN**, or **P.CURREN** are specified. Only nodes falling within the specified bounds and belonging to the electrode are included in the integration.	none	
N.CURREN	logical	Specifies that the electron current through an electrode or the electron current across the boundary between two regions is extracted.	false	
P.CURREN	logical	Specifies that the hole current through an electrode or the hole current across the boundary between two regions is extracted.	false	
REGIONS	char	The names of two adjacent regions. Electron or hole current flow across the boundary between these regions is extracted if **N.CURREN** or **P.CURREN**, respectively, is specified. The two region names should be enclosed in parentheses and separated by a comma.	none	
II.GENER	logical	Specifies that the number of electron-hole pairs generated by impact ionization in the entire device or, if specified, in the area identified by **X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX**, is extracted.	false	
SHEET.RE	logical	Extracts the sheet resistance of the layers intersected by the vertical line at x=**X.POINT** producing a table giving the resistance of each doped layer. Current flow is assumed to be perpendicular to xy plane.	false	
X.POINT	number	The x location of the line along which to extract sheet resistance.	none	microns
X.MIN	number	The minimum x coordinate of the region over which the specified quantity is extracted.	The minimum x location in the device structure.	microns
X.MAX	number	The maximum x coordinate of the region over which the specified quantity is extracted.	The maximum x location in the device structure.	microns
Y.MIN	number	The minimum y coordinate of the region over which the specified quantity is extracted.	The minimum y location in the device structure.	microns
Y.MAX	number	The maximum y coordinate of the region over which the specified quantity is extracted.	The maximum y location in the device structure.	microns
OUT.FILE	char	The identifier of a formatted output file for storing the extracted information. **synonym: OUTFILE**	none	

Parameter	Type	Definition	Default	Units
Device to Extract (Circuit Analysis AAM)				
STRUCTUR	char	Specifies the device in which the extraction is performed. This parameter is only used with the Circuit Analysis AAM.	first element	
Extract MOS Parameters				
MOS.PARA	logical	Specifies that various parameters associated with MOS devices are extracted. If an I-V log file is available, or if one is specified with **IN.FILE**, the program attempts to extract information such as threshold voltage and subthreshold slope. If a mesh file is available, the program attempts to extract the channel length for the structure.	false	
DRAIN	char	The electrode name associated with the drain in an open log file, or one that is read in using **IN.FILE**.	none	
GATE	char	The electrode name associated with the gate in an open log file, or one that is read in using **IN.FILE**.	none	
IN.FILE	char	The identifier of an I-V log file to use for extracting MOS parameters.	Currently open I-V log file.	
I.DRAIN	number	The drain current for which the corresponding voltage is extracted. The extraction is performed using both linear and logarithmic interpolation.	none	A/micron
CONDITIO	char	Specifies a numeric expression that must evaluate to "TRUE" before data will be used for the MOS parameter extraction.	none	
APPLIED	logical	Specifies that applied biases are used for the MOS parameter extraction. Specifying **^APPLIED** will cause the contact voltages to be used.	true	

Description

The **EXTRACT** statement extracts selected data from the solution over a specified cross-section of the device.

See Also…

To further illustrate the **EXTRACT** statement, refer to input files:

- *mdex1m* in N-Channel MOSFET Examples, Chapter 4, "Analysis Including Band-to-Band Tunneling" on page 4-25
- *mdex2fp* in NPN Bipolar Transistor Examples, Chapter 5, "Simulation of Forward Characteristics" on page 5-5
- *mdex2m* in NPN Bipolar Transistor Examples, Chapter 5, "Simulation of a One-Dimensional Bipolar Transistor" on page 5-19

Extraction with Expressions and Names

This capability allows access to internal (predefined) variables through an algebraic expression. The expression is evaluated (at each device node if required) and stored in the variable given by **NAME**. The **NAME** variable may then be used for plotting, to control program execution, or in other expressions.

If two extract statements use the same **NAME**, the second extract statement replaces the first, thus the following two statements:

```
EXTRACT   NAME=DOG  EXP="@X*@X"
EXTRACT   NAME=DOG  EXP="@X*@X*@X"
```

would assign x^3 to the variable dog. All extract definitions may be deleted by use of the **CLEAR** parameter.

Four examples of **EXTRACT** capabilities are presented below.

Extraction of Maximum Electric Field

Calculate the maximum value of the electric field within the portion of the device x>5 and y<3. Next on the **EXTRACT** statement set **EMAX=MAX(@EMAX;E(i))** where **E(i)** represents the magnitude of the electric field at each grid point. The **CONDITIO** parameter is used to limit the points to be checked to those within the portion of the device where x>5 and y<3. The parameter **INITIAL** is used to set the target EMAX to zero at the start of the calculation.

```
EXTRACT  EXPRESS="MAX(@EMAX;@EM)"  COND="@X>5&@Y<3"  NAME=EMAX
+        INITIAL=0
```

$\vec{J} \cdot \vec{E}$ Calculation

Plot the heating term $\vec{J} \cdot \vec{E}$. The predefined array **ARRAY1** is used to hold the result of the calculation. The program multiplies J_n and E at each grid point and stores the result in the array **ARRAY1**. **ARRAY1** may then be plotted in the normal way.

```
EXTRACT  EXPRESS="(@JNX+@JPX)*@EX+(@JNY+@JPY)*@EY"  NAME=ARRAY1
PLOT.2D  BOUND
CONTOUR  FILL  ARRAY1
```

Defining Mobility and Lifetime Values

Note that lifetime and mobility are predefined quantities that are "writable". This means they can be used to define the mobility and lifetime to be used during the solution process. For example, the following statement can be used to define the low-field electron mobility in the structure based on total donor concentration and lattice temperature:

```
EXTRACT  NAME=mobn NOW ^AT.BIAS
+        EXPRESS=1000*(1e18/@Nd)*(300/@tl)**2
```

CAUTION

When defining mobility or lifetime using an EXTRACT expression, models should not be specified that will cause the defined values to be overwritten. For example, if "MODELS CONMOB FLDMOB" is specified, then during the iterative solution process, low-field mobility will be calculated using CONMOB rather than the defined expression on the EXTRACT statement.

Binary Search

Use a binary search to find the gate voltage needed to make the drain current equal to 1mA. Search the interval Vg=1V to Vg=5V. The **EXTRACT** statement is used to check whether or not the drain current is greater than 1e-3A. If it is, the assigned variable **TEST** is set to 1. Next, check **TEST**; if **TEST**=1, decrease the bias.

This example illustrates a powerful method of extraction or optimization. This case varied a bias voltage, but a doping profile or model parameter could have been varied instead.

```
MESH .....
SYMBOLIC .....
ASSIGN  NAME=LOW N.VAL=1
ASSIGN  NAME=HIGH N.VAL=5
ASSIGN  NAME=STEP  N.VAL=(@HIGH-@LOW)/2
ASSIGN  NAME=VA  N.VAL=@LOW+@STEP
EXTRACT EXPRESS="@I(Drain)>1E-3"  NAME=TEST
LOOP STEPS=10
   SOLVE V(Gate)=@VA
   ASSIGN NAME=STEP  N.VAL=@STEP/2
   IF COND=@TEST
      ASSIGN NAME=VA N.VAL=@VA-@STEP
   ELSE
      ASSIGN NAME=VA N.VAL=@VA+@STEP
   IF.END
L.END
ECHO "THE REQUIRED VOLTAGE IS: "@VA" +/-"@STEP
```

Creating a Ring of Constant Doping

The following example creates a ring of dopant with a constant density of le17 cm^{-3}, with an inner radius of 5 microns and an outer radius of 8 microns.

```
EXTRACT  EXPRESS="@NET+1E17"  NAME=NET
         COND="(@X*@X+@Y*@Y)>25&(@X*@X+@Y*@Y)<64"
```

Optimization

The optimization functions built into the input parser can be used for optimizing a wide variety of parameters such as:

- Bias voltages
- Doping profiles
- Model parameters

The following example optimizes two doping profiles in a simple 1D diode to give specific IV results:

```
LOOP  OPTIMIZE
      ASSIGN NAME=DOP1 UP=1E20 LOW=2E17 N.VAL=3E18 OPTIMIZE
      ASSIGN NAME=DOP2 UP=1E17 LOW=1E14 N.VAL=1E16 OPTIMIZE
      MESH
      X.MESH   WIDTH=1  N.SPACES=1
      Y.MESH   WIDTH=1  N.SPACES=50
      ELECT    NUM=ANODE  TOP
      ELECT    NUM=CATHODE  BOTTOM
      PROFILE  N-TYPE  N.PEAK=@DOP2  UNIF
      PROFILE  P-TYPE  N.PEAK=@DOP1  Y.JUNC=0.2
      SYMB     NEWT  CARR=2
      EXTRACT  NAME=P1  EXP="@I(ANODE)"  TARG=2.2E-8
+             COND="@V(ANODE)=.5"
      EXTRACT  NAME=P2  EXP=@I(ANODE)  TARG=4.6E-7
+             COND="@V(ANODE)=.6"
      SOLVE    V(ANODE)=0.5
      SOLVE    V(ANODE)=0.6
L.END
```

where:

- The **LOOP** statement defines the start of the optimization loop.
- The statements between the **LOOP** and the **L.END** are repeated until the optimization is complete.
- The **OPTIMIZE** parameter on the **LOOP** statement tells the program to perform the optimization.
- The two **ASSIGN** statements set up the following:
  - The initial values for the variables DOP1 and DOP2
  - The **LOWER** and **UPPER** bounds to constrain the optimization
  - Tells the program to actually optimize these variables via the **OPTIMIZE** parameter
- The two **PROFILE** statements make use of the optimized doping values via the DOP1 and DOP2 assigned used as the values for **CONC**.
- The two **EXTRACT** statements are used to extract the anode currents into variables P1 and P2.

  At the same time the **EXTRACT** statement defines the **TARGET** values for the optimization.
- The program then tries to adjust DOP1 and DOP2 so that the extracted anode currents match the targets (2.2e-8 and 4.6e-7).

Optimization Guidelines

Device simulation and optimization can be a lengthy process. Since both device simulation and optimization are highly non-linear processes there is no guarantee

that the optimization will succeed. To get the most from the optimization, *Avant!* TCAD recommends the following guidelines.

Parameter Selection

- Optimize as few parameters as possible.
- While the **EXTRACT** statement can optimize up to 20 parameters at once, a practical upper limit is about three or four.
- If possible, break a complex optimization up into several smaller optimizations.
- For example, for a MOSFET, you may want to first optimize mobility parameters which effect the forward region and then separately optimize band-to-band tunneling parameters in the reverse region.

Initial Guess and Limits

- Carefully choose the initial guess and limits.
- Use the **UPPER** and **LOWER** parameters to constrain the solution to values which should be close to the true solution.

The optimizer can become easily confused if the initial guess is far from the true solution. The reason is that the optimizer only finds local minima. If a local minimum lies between the initial guess point and the true solution the optimizer finds the local minimum, not the true solution. In a solution space of several variables, there may be a great many local minima, but only one true solution.

In general, picking many parameters, using randomly chosen initial values and letting the simulator "sort them out" does not work.

Examining the Results

- Carefully examine the results of the optimization.

The minimum found is not guaranteed to give a good fit and even if it does, the parameter set that was found may not be unique or the best set. The parameters which give the best fit may not be the best choices from the standpoint of producing predictive simulations. This is particularly true if a large number of parameters are optimized.

Terminating the Process

There are two parameters which determine when the optimization process terminates.

- **TARTOL** is the absolute tolerance and the optimization stops as soon as the error becomes less than **TARTOL** percent.
- **TARREL** stops the optimization when the error between two consecutive optimization steps becomes less than **TARREL**.

  The second limit is needed since in general the bottom of a minima is not at zero. At the bottom of a minima, however, the error stops changing and **TARREL** terminates the process.

Simple Extraction

The **EXTRACT** statement can be used for the following "simple" processes:

- To integrate concentrations over a specified cross-section of a device for net carrier, net charge, electron, and hole.
- To extract the charge on part of an electrode, as well as the current through that part.

  This is useful for capacitance studies, in conjunction with the difference mode on the **LOAD** statement.
- The current flow across the boundary between two adjacent regions can be extracted by using the **REGIONS** parameter.

  The resistance of a cross-sectional structure, for instance a diffused line, can also be extracted.
- Ionization integrals for electrons and holes are computed by using the **IONIZATI** parameter.

  Ionization integrals are computed along potential gradient paths initiated at nodes lying in the rectangular region defined by the parameters **X.MIN**, **X.MAX**, **Y.MIN**, and **Y.MAX**.

  The output consists of the maximum ionization integral for both electron and holes, the peak field along the path that produced the maximum ionization integral, and the location of the peak field along this path.

Extraction of MOS Device Parameters

Various parameters associated with MOS devices can be extracted by selecting **MOS.PARA**. If an I-V log file is available, or if one is specified with **IN.FILE**, the program attempts to extract information such as threshold voltage and sub-threshold slope. The program will look for electrodes that have names "gate" and "drain." If these names are not available for the device structure, then you must specify the electrode names that correspond to the gate and drain using the **GATE** and **DRAIN** parameters, respectively.

If a valid Medici device structure is available, the program attempts to extract the MOS channel length from it. For this purpose, the channel length is defined as the distance between the source-channel and drain-channel metallurgical junctions at the semiconductor-insulator interface.

Example:

```
EXTRACT  MOS.PARA  IN.FILE=GATE.IVL
```

An example of the output produced by this statement follows:

```
 MOS Parameter Extraction
------------------------
Lchan:          1.50238      microns
  x(start) =   0.748808       y(start) =   6.914763E-04
  x(end)   =    2.25119       y(end)   =   6.914803E-04

file: GATE.IVL
number of data values:   11
first point: Vg=      0.    , Vd=  0.1000    , Id=  8.8634E-12
last point:  Vg=   2.000    , Vd=  0.1000    , Id=  8.2236E-06
 S_lin (linear slope):    5.5002E-06 (A/um-V)   at Vg=  0.8000     (V)
      Vth (intercept):    0.3621     (V)
 S_sub (subthreshold slope):   81.76    (mV/dec) at Vg=  0.2000     (V)
```

Predefined Quantities in Expressions

Expressions follow the general rules set forth in "Numerical Expressions" on page 3-5. The following predefined quantities are available for use in calculations. These quantities allow access to the simulation structure and the solution process.

The special variables are accessed in parameter expressions by prefixing the variable name with an at sign "@" in the same way as normal assigned variables. Some of the quantities are writable meaning that the value currently in use by the simulator can be altered. For example, you can create new initial guesses by altering *n*, *p*, or *pot*. Footnotes are at the end of the table.

Note:

In addition to the predefined quantities listed below, quantities that were originally defined on the* `PROFILE` *statement using the* `IMPURITY` *or* `OTHER` *parameters can also be used in expressions by prefixing their names with "@". The* `IMPURITY` *and* `OTHER` *quantities are also writable.

Table 3-1 Predefined Quantities Used in Calculations

Name	Quantity	Units	Writable?
x	Node x coordinate	Microns	No
y	Node y coordinate	Microns	No
net	Net node doping	#/cm^3	Yes
total	Total node doping	#/cm^3	Yes
Na	Total acceptor concentration	#/cm^3	No
Nd	Total donor concentration	#/cm^3	No
region	Node region #	-	No
node.num	Node number	-	No
interfac	At interface?	-	No
electrod	Node electrode #	-	No
taun	Electron lifetime	sec^{-1}	Yes

Table 3-1 Predefined Quantities Used in Calculations

Name	Quantity	Units	Writable?
taup	Hole lifetime	sec^{-1}	Yes
mobn	Low-field electron mobility	$cm^2/V\text{-}sec$	Yes
mobp	Low-field hole mobility	$cm^2/V\text{-}sec$	Yes
photogen	Photogeneration rate (1)	$Ehp/cm^3/sec$	Yes
time	Time	Seconds	No
delt	Current time step	Seconds	No
iterat	Number of Newton iterations	-	No
nx.int	At node next to interface	-	No
sem.area	Semiconductor area of a node	cm^2	No
ins.area	Insulator area of a node	cm^2	No
prp.dist	Perpendicular interface dist. in semi. (2)	cm	No
ins.dist	Perpendicular interface dist. in insulator	cm	No
tot.dist	Total perpendicular dist. at interface	cm	No
pot	Node potential	Volts	Yes
n	Node electron conc	$\#/cm^3$	Yes
p	Node hole conc	$\#/cm^3$	Yes
tn	Node electron temp.	Kelvin	Yes
tp	Node hole temp	Kelvin	Yes
tl	Node Lattice temp	Kelvin	Yes
jnm	Electron current density	$Amps/cm^2$	No
jpm	Hole current density	$Amps/cm^2$	No
jdm	Displacment current density	$Amps/cm^2$	No
jtm	Total current density	$Amps/cm^2$	No
ii.gener	Impact Ionization rate	$\#/cm^3/sec$	No
recomb	Recombination rate (for unequal electron and hole recombination, **recombin** is the same as **n.recomb**)	$\#/cm^3/sec$	No
n.recomb	Electron recombination rate	$\#/cm^3/sec$	No
p.recomb	Hole recombination rate	$\#/cm^3/sec$	No
em	Electric field	Volts/cm	No
bb.gener	Band-to-band generation rate	$\#/cm^3/sec$	No
trap.occ	Trap occupation	$\#/cm^3$	No
qfn	Electron quasi Femi level	eV	No
qfp	Hole quasi Fermi level	eV	No

Table 3-1 Predefined Quantities Used in Calculations

Name	Quantity	Units	Writable?
conduc.b	Conduction band energy	eV	No
valenc.b	Valence band energy	eV	No
vacuum	Vacuum level	eV	No
i(i)	Terminal current (3)	Amps/micron	No
va(i)	Applied voltage at the terminal	Volts	No
v(i)	Terminal Voltage	Volts	No
he(i)	Hot electron gate current at term.	Amps/micron	No
fe(i)	Fowler Nordheim or Direct Tunneling current at term.	Amps/micron	No
q(i)	Terminal charge	Coul./micron	No
freq	AC analysis frequency	Hz	No
g(i,j)	AC small signal conductance	Mhos/micron	No
y(i,j)	Y parameter at terminal	Mhos/micron	No
c(i,j)	AC small signal capacitance	Farads/micron	No
sr(i,j)	Real S parameter (4)	none	No
si(i,j)	Imaginary S parameter (4)	none	No
vc(i)	Circuit node voltage	Volts	No
ic(i)	Circuit inductor/voltage src. current	Amps	No
vcr(i)	AC node voltage, real part	Volts	No
icr(i)	AC ind/voltage src. current (real)	Amps	No
vci(i)	AC node voltage imaginary part	Volts	No
ici(i)	AC ind/voltage src. current (Imag)	Amps	No
array1	Scratch array 1	-	Yes
array2	Scratch array 2	-	Yes
array3	Scratch array 3	-	Yes
ii.n.int	Electron ionization integral (**synonym: ii.integ**)	-	No
ii.p.int	Hole ionization integral	-	No
jnx	x-component elect. curr density	A/cm^2	No
jny	y-component elect. curr density	A/cm^2	No
jpx	x-component hole curr density	A/cm^2	No
jpy	y-component hole curr density	A/cm^2	No
ex	x-component Electric field	V/cm	No
ey	y-component Electric field	V/cm	No

Table 3-1 Predefined Quantities Used in Calculations

Name	Quantity	Units	Writable?
nie	Intrinsic Carrier concentration	#/cm^3	No
net.carr	net carriers	#/cm^3	No
net.char	net charge	#/cm^3	No
gin	Hot electron injection current	A/micron	No
gip	Hot hole injection current	A/micron	No
x.mole	mole Fraction	none	No
Lchan	Channel length	microns	No
Vth	Threshold voltage (5)	Volts	No
S_lin	Linear region slope (5)	A/micron-V	No
S_sub	Subthreshold region slope (5)	mV/decade	No
Vg_lin	V_g where I_d = user specified value (using linear interpolation) (6)	Volts	No
Vg_log	V_g where I_d = user specified value (using logarithmic interpolation) (6)	Volts	No
Vd_lin	V_d where I_d = user specified value (using linear interpolation) (7)	Volts	No
Vd_log	V_d where I_d = user specified value (using logarithmic interpolation) (7)	Volts	No
Vth_sat	Saturation region intercept. (8)	Volts	No

Table Notes:

(1) This is the case for steady state simulations. In the case of time-dependent simulations, the rate is integrated over the duration of the simulation and a rate in #/cm^3 is used.

(2) This quantity gives the perpendicular distance to the interface associated with each interface node. If the node is not an interface node, 0 is returned. Refer to the section on the **TRAPS** statement for use.

(3) The "i" and "j" arguments used with the names of terminal characteristics represent valid electrode names. If "drain" and "gate" are valid electrode names in a simulation, then "c(gate,drain)" could be used in an expression.

(4) Only the real and imaginary parts of S(1,1), S(1,2), S(2,1), S(2,2) may be used in expressions.

(5) An appropriate set of I_d-V_g data must be available to extract this quantity. Vth is defined as the intercept with the V_g axis from the point of maximum slope.

(6) This quantity is only available if an appropriate set of I_d-V_g data is available and if a previous **EXTRACT** statement specified "**MOS.PARA I.DRAIN**=<n>"

(7) This quantity is only available if an appropriate set of I_d-V_d data is available and if a previous **EXTRACT** statement specified "**MOS.PARA I.DRAIN**=<n>"

(8) This quantity is only available if I_d-V_g data is available for which V_d=V_g.

Vth_sat is defined as the intercept with the V_g axis from the point of maximum slope of the $\sqrt{I_d}$ - V_g curve.

PRINT

The **PRINT** statement prints specific quantities at points within a defined area of a device.

```
PRINT

   [ {  ( [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>] )
      | ( [IX.MIN=<n>] [IX.MAX=<n>] [IY.MIN=<n>] [IY.MAX=<n>] )
     }
   ]
   [POINTS] [ELEMENTS] [GEOMETRY] [INTERFAC] [IMPURITY] [OTHER]
   [SOLUTION] [ CURRENT [ {X.COMPON | Y.COMPON} ] ] [E.FIELD]
   [NET.CHAR] [RECOMBIN] [II.GENER] [II.EJG] [CONC.DEP]
   [BB.GENER] [BB.EG] [TEMPERAT] [BAND.STR]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]
```

Parameter	Type	Definition	Default	Units
X.MIN	number	The minimum x coordinate of the area which defines the print region.	The minimum x location in the device structure.	microns
X.MAX	number	The maximum x coordinate of the area which defines the print region.	The maximum x location in the device structure.	microns
Y.MIN	number	The minimum y coordinate of the area which defines the print region.	The minimum y location in the device structure.	microns
Y.MAX	number	The maximum y coordinate of the area which defines the print region.	The maximum y location in the device structure.	microns
IX.MIN	number	The minimum x node index of the area which defines the print region.	1	none
IX.MAX	number	The maximum x node index of the area which defines the print region.	The maximum x node number in the device structure.	none
IY.MIN	number	The minimum y node index of the area which defines the print region.	1	none
IY.MAX	number	The maximum y node index of the area which defines the print region.	The maximum y node number in the device structure.	none

Parameter	Type	Definition	Default	Units
POINTS	logical	Specifies that node information is printed. This includes the node coordinates, impurity concentration, interface charge, region number, material type index, and electrode connections.	false	
ELEMENTS	logical	Specifies that element information is printed. This includes element number, node numbers of the element's vertices, and material number.	false	
GEOMETRY	logical	Specifies that geometrical information for each element is printed. This includes both Poisson and current-continuity coupling coefficients and the area associated with each node of the element.	false	
INTERFAC	logical	Specifies that information for interface nodes is printed. This includes the amount of fixed charge, densities of fast interface traps, and insulator charge (at insulator nodes).	false	
IMPURITY	logical	Specifies that nodal values of impurity concentration for all impurities present in the structure are printed. Net and total concentration is also printed.	false	
OTHER	logical	Specifies that nodal values of all **OTHER** quantities present in the structure are printed.	false	
SOLUTION	logical	Specifies that information for the present solution is printed. This includes potential, carrier concentrations, and quasi-Fermi potentials.	false	
CURRENT	logical	Specifies that the magnitude of electron, hole, conduction, displacement, and total current density for the present solution is printed.	false	
X.COMPON	logical	Specifies that the x component of current density is printed instead of the magnitude.	false	
Y.COMPON	logical	Specifies that the y component of current density is printed instead of the magnitude.	false	
E.FIELD	logical	Specifies that the electric field for the present solution is printed. This includes location, magnitude, and x and y components.	false	
NET.CHAR	logical	Specifies that net charge for the present solution is printed. **synonym: QU**	false	
RECOMBIN	logical	Specifies that recombination for the present solution is printed.	false	
II.GENER	logical	Specifies that the total generation rate due to impact ionization is printed at each node of the simulation mesh.	false	
II.EJG	logical	Specifies that information pertaining to the calculation of impact ionization generation for each element is printed. This information includes the electric field parallel to the current flow for the element, the current density magnitude for the element, and the impact ionization generation rate for the element.	false	
CONC.DEP	logical	Specifies that concentration-dependent mobility and lifetime values at each node are printed.	false	

Parameter	Type	Definition	Default	Units
BB.GENER	logical	Specifies that the total generation rate due to band-to-band tunneling is printed at each node of the simulation mesh.	false	
BB.EG	logical	Specifies that information pertaining to the calculation of band-to-band tunneling generation for each element is printed. This information includes the electric field parallel to the current flow for the element, and the band-to-band tunneling generation rate for the element.	false	
TEMPERAT	logical	Specifies that the lattice, electron and hole temperature at each grid point is printed.	false	
BAND.STR	logical	Specifies that various parameters related to the band structure are printed. These include node number, region name, electron affinity, the band structure parameter, the energy gap, the conduction and valence band densities, and the intrinsic carrier concentration.	false	
Circuit Analysis Parameters				
STRUCTUR	char	Selects the device for which the information is printed. This parameter is only used with the Circuit Analysis AAM.	first element	

Description

The **PRINT** statement prints specific quantities at points within a defined area of a device. Information is printed for all nodes (for nodal quantities) or for all elements (for elemental quantities) within the selected area of the device.

The region of interest may be selected by specifying either the bounding device coordinates or by specifying the bounding nodal indices. If nodal indices are used the mesh must be an unrefined rectangular mesh. The default area is the entire device.

PLOT.1D

The **PLOT.1D** statement plots a specific quantity along a line segment through the device, or plots terminal characteristics from data accumulated in a log file or read in from a previous log file.

```
PLOT.1D

 Distance Plot Quantities
   {  ( {  POTENTIA | QFN | QFP | VALENC.B | CONDUC.B | VACUUM
         | E.FIELD | ARRAY1 | ARRAY2 | ARRAY3
         | ( {TRAPS | TRAP.OCC} [LEVEL=<n>] )
         | DOPING | ELECTRON | HOLES | NIE | NET.CHAR | NET.CARR
         | J.CONDUC | J.ELECTR | J.HOLE | J.DISPLA | J.TOTAL
         | RECOMBIN | N.RECOMB | P.RECOMB | II.GENER | BB.GENER
         | ( PHOTOGEN [WAVE.NUM=<n>] ) | N.MOBILI | P.MOBILI | SIGMA
         | ELE.TEMP | HOL.TEMP | ELE.VEL | HOL.VEL | J.EFIELD
         | G.GAMN | G.GAMP | G.GAMT | G.IN | G.IP | G.IT
         | IMPURITY=<c> | OTHER=<c>

 Lattice Temperature AAM Parameters
         | LAT.TEMP

 Heterojunction Device AAM Parameters
         | X.MOLE
        }

 AC Small-Signal Analysis Quantity Parameters
        [ {AC.REAL | AC.IMAG | AC.MAGN | AC.PHAS} ]

 Distance Plot Parameters
        [X.COMPON] [Y.COMPON]
        X.START=<n> Y.START=<n> X.END=<n> Y.END=<n> [HORZ.STA=<n>]
        [ {FIND.MIN | FIND.MAX} [SEMICOND] [INSULATO] [FIND.DIS=<n>] ]
      )

 Terminal Characteristics Plot Parameters
    | ( X.AXIS=<c> Y.AXIS=<c> [ORDER] [IN.FILE=<c>]
        [X.MIN=<n>] [X.MAX=<n>] [CONDITIO=<c>]
      )
   } Plot Controls
   [ SPLINE [NSPLINE=<n>] ]
   [LEFT=<n>] [RIGHT=<n>] [BOTTOM=<n>] [TOP=<n>] [UNCHANGE]
   [ {Y.LOGARI | S.LOGARI | INTEGRAL} ] [ABSOLUTE] [NEGATIVE]
   [CLEAR] [AXES] [LABELS] [MARKS] [TITLE=<c>] [T.SIZE=<n>]
   [X.OFFSET=<n>] [X.LENGTH=<n>] [X.SIZE=<n>] [X.LOGARI]
   [Y.OFFSET=<n>] [Y.LENGTH=<n>] [Y.SIZE=<n>]
   [CURVE] [ {SYMBOL=<n> | POINTS} ] [C.SIZE=<n>]
   [LINE.TYP=<n>] [COLOR=<n>] [DEVICE=<c>] [PAUSE]
   [PLOT.OUT=<c>] [PLOT.BIN=<c>] [PRINT] [OUT.FILE=<c>]
   [TIMESTAM [TIME.SIZ=<n>] ]
```

(**PLOT.1D**, continued next page)

(**PLOT.1D**, continued from previous page)

Circuit Analysis AAM Parameters
[STRUCTUR=<c>]

Parameter	Type	Definition	Default	Units
Distance Plot Quantities				
POTENTIA	logical	Specifies that midgap potential in volts is plotted versus distance along the specified line through the device.	false	
QFN	logical	Specifies that the electron quasi-Fermi potential in volts is plotted versus distance along the specified line through the device.	false	
QFP	logical	Specifies that the hole quasi-Fermi potential in volts is plotted versus distance along the specified line through the device.	false	
VALENC.B	logical	Specifies that the valence band potential in volts is plotted versus distance along the specified line through the device.	false	
CONDUC.B	logical	Specifies that the conduction band potential in volts is plotted versus distance along the specified line through the device.	false	
VACUUM	logical	Specifies that the vacuum potential in volts is plotted versus distance along the specified line through the device.	false	
E.FIELD	logical	Specifies that the magnitude of electric field in volts per centimeter is plotted versus distance along the specified line through the device.	false	
ARRAY1	logical	Specifies that the user generated array number 1 is to be plotted along the specified line. Refer to the **EXTRACT** statement for more information.	false	
ARRAY2	logical	Specifies that the user generated array number 2 is to be plotted along the specified line. Refer to the **EXTRACT** statement for more information.	false	
ARRAY3	logical	Specifies that the user generated array number 3 is to be plotted along the specified line. Refer to the **EXTRACT** statement for more information.	false	
TRAPS	logical	Specifies that the trap density in number per cubic centimeter is to be plotted along the specified line.	false	
TRAP.OCC	logical	Specifies that the filled trap density in number per cubic centimeter is to be plotted along the specified line.	false	
LEVEL	number	The specific trap level to plot. If this parameter is not specified, all trap levels are summed.	none	none
DOPING	logical	Specifies that the net impurity concentration in number per cubic centimeter is plotted versus distance along the specified line through the device. The net impurity concentration is the donor impurity concentration minus the acceptor impurity concentration.	false	
ELECTRON	logical	Specifies that electron concentration in number per cubic centimeter is plotted versus distance along the specified line through the device.	false	
HOLES	logical	Specifies that hole concentration in number per cubic centimeter is plotted versus distance along the specified line through the device.	false	

Parameter	Type	Definition	Default	Units
NIE	logical	Specifies that effective intrinsic carrier concentration in number per cubic centimeter is plotted versus distance along the specified line through the device.	false	
NET.CHAR	logical	Specifies that the net charge concentration in number per cubic centimeter is plotted versus distance along the specified line through the device. The net charge concentration is the sum of the donor impurity concentration and hole concentration minus the sum of the acceptor impurity concentration and electron concentration plus the concentration of any trapped charge.	false	
NET.CARR	logical	Specifies that the net carrier concentration in number per cubic centimeter is plotted versus distance along the specified line through the device. The net carrier concentration is the hole concentration minus the electron concentration.	false	
J.CONDUC	logical	Specifies that conduction current in amps per square centimeter is plotted versus distance along the specified line through the device.	false	
J.ELECTR	logical	Specifies that electron current in amps per square centimeter is plotted versus distance along the specified line through the device.	false	
J.HOLE	logical	Specifies that hole current in amps per square centimeter is plotted versus distance along the specified line through the device.	false	
J.DISPLA	logical	Specifies that displacement current in amps per square centimeter is plotted versus distance along the specified line through the device.	false	
J.TOTAL	logical	Specifies that total current in amps per square centimeter is plotted versus distance along the specified line through the device.	false	
RECOMBIN	logical	Specifies that net recombination in number per cubic centimeter per second is plotted versus distance along the specified line through the device. For unequal electron and hole recombination, **RECOMBIN** is the same as **N.RECOMB**.	false	
N.RECOMB	logical	Specifies that net electron recombination in number per cubic centimeter per second is plotted versus distance along the specified line through the device.	false	
P.RECOMB	logical	Specifies that net hole recombination in number per cubic centimeter per second is plotted versus distance along the specified line through the device.	false	
II.GENER	logical	Specifies that the total generation rate due to impact ionization in pairs per cubic centimeter per second is plotted versus distance along the specified line through the device.	false	
BB.GENER	logical	Specifies that the total generation rate due to band-to-band tunneling in pairs per cubic centimeter per second is plotted versus distance along the specified line through the device.	false	
PHOTOGEN	logical	Specifies that total photogeneration in pairs per cubic centimeter per second is plotted versus distance along the specified line through the device.	false	
N.MOBILI	logical	Specifies that the electron mobility in cm^2/V-s is plotted versus distance along the specified line through the device.	false	
P.MOBILI	logical	Specifies that the hole mobility cm^2/V-s is plotted versus distance along the specified line through the device.	false	
SIGMA	logical	Specifies that the conductivity in $(Ohm\text{-}cm)^{-1}$ is plotted versus distance along the specified line through the device.	false	

Parameter	Type	Definition	Default	Units
ELE.TEMP	logical	Specifies that the electron temperature in Kelvins is plotted versus distance along the specified line through the device.	false	
HOL.TEMP	logical	Specifies that the hole temperature in Kelvins is plotted versus distance along the specified line through the device.	false	
ELE.VEL	logical	Specifies that the electron mean velocity in cm/s is plotted versus distance along the specified line through the device.	false	
HOL.VEL	logical	Specifies that the hole mean velocity in cm/s is plotted versus distance along the specified line through the device.	false	
J.EFIELD	logical	Specifies that the component of the electric field in V/cm in the direction of the total current density is plotted versus distance along the specified line through the device.	false	
G.GAMN	logical	Specifies that the probability per unit length that an electron is injected into the oxide is plotted versus distance along the specified line through the device.	false	
G.GAMP	logical	Specifies that the probability per unit length that a hole is injected into the oxide is plotted versus distance along the specified line through the device.	false	
G.GAMT	logical	Specifies that the probability per unit length that an electron or hole (the sum of the electron and hole probabilities) is injected into the oxide is plotted versus distance along the specified line through the device.	false	
G.IN	logical	Specifies that hot electron injection current initiated from each point in amps/micron is plotted versus distance along the specified line through the device.	false	
G.IP	logical	Specifies that hot hole injection current initiated from each point in amps/micron is plotted versus distance along the specified line through the device.	false	
G.IT	logical	Specifies that total hot carrier injection current initiated from each point in amps/micron is plotted versus distance along the specified line through the device.	false	
IMPURITY	char	The name of an impurity to plot in number per cubic centimeter as a function of distance along the specified line through the device.	none	
OTHER	char	The name of an **OTHER** quantity to plot as a function of distance along the specified line through the device.	none	
Lattice Temperature AAM Parameters				
LAT.TEMP	logical	Specifies that the lattice temperature in Kelvins is plotted versus distance along the specified line through the device. This parameter is only used with the Lattice Temperature AAM.	false	
Heterojunction Device AAM Parameters				
X.MOLE	logical	Specifies that the mole fraction for the material is plotted versus distance along the specified line through the device. This parameter is only used with the Heterojunction Device AAM.	false	
AC Small-Signal Analysis Quantity Parameters				
AC.REAL	logical	Specifies that the real part of the quantity obtained from AC analysis is plotted.	false	
AC.IMAG	logical	Specifies that the imaginary part of the quantity obtained from AC analysis is plotted	false	

Parameter	Type	Definition	Default	Units
AC.MAGN	logical	Specifies that the magnitude of the quantity obtained from AC analysis is plotted.	false	
AC.PHAS	logical	Specifies that the phase of the quantity obtained from AC analysis is plotted. Phase is defined as $\text{atan}(\text{imag}(X)/\text{real}(X))$, where X represents the quantity to be plotted.	false	
Distance Plot Parameters				
X.COMPON	logical	Specifies that the x component of a vector quantity is plotted as opposed to the default magnitude.	false	
Y.COMPON	logical	Specifies that the y component of a vector quantity is plotted as opposed to the default magnitude.	false	
X.START	number	The x location of the initial point of the line segment along which the specified quantity is plotted. synonyms: A.X	none	microns
Y.START	number	The y location of the initial point of the line segment along which the specified quantity is plotted. **synonym: A.Y**	none	microns
X.END	number	The x location of the final point of the line segment along which the specified quantity is plotted. **synonym: B.X**	none	microns
Y.END	number	The y location of the final point of the line segment along which the specified quantity is plotted. **synonym: B.Y**	none	microns
HORZ.STA	number	The value along the horizontal plot axis associated with the starting point of the line. This value establishes the reference for horizontal distance along the line.	0.0	microns
FIND.MIN	logical	Specifies that the minimum value of the specified quantity is plotted versus distance along the specified line through the device. For each point along the specified line, the program finds the minimum value by searching along a line that passes through the point and is perpendicular to the specified line.	false	
FIND.MAX	logical	Specifies that the maximum value of the specified quantity is plotted versus distance along the specified line through the device. For each point along the specified line, the program finds the maximum value by searching along a line that passes through the point and is perpendicular to the specified line.	false	
SEMICOND	logical	Specifies that when **FIND.MIN** or **FIND.MAX** is used, the search area should include semiconductor materials.	true	
INSULATO	logical	Specifies that when **FIND.MIN** or **FIND.MAX** is used, the search area should include insulator materials.	false	
FIND.DIS	number	The maximum distance to either side of the specified line over which the search takes place when **FIND.MIN** or **FIND.MAX** is specified.	The search line spans the device.	microns
Terminal Characteristics Plot Parameters				
X.AXIS	char	The quantity used for the horizontal axis when plotting data stored in a log file. If a log file is available or read using the **IN.FILE** parameter, the choices include the following: **I-V Terminal Characteristics** *VA(<name>)* applied bias for electrode *<name>*	none	

Parameter	Type	Definition	Default	Units
		V(<name>) contact bias for electrode *<name>*		
		I(<name>) total terminal current for electrode *<name>*		
		IE(<name>) electron terminal current for electrode *<name>*		
		IH(<name>) hole terminal current for electrode *<name>*		
		ID(<name>) displacement terminal current for electrode *<name>*		
		QE(<name>) total charge on electrode *<name>*		
		HE(<name>) hot carrier injection current into electrode *<name>*		
		TIME simulation time (transient simulations)		
		II impact ionization current (integrated I.I. gener. rate)		
		<name> a quantity defined with the **EXTRACT** statement		
		AC Analysis Quantities Note: Quotes are required.		
		"C(<ni>,<nj>)" AC capacitance component for elect. *<ni>,<nj>*		
		"G(<ni>,<nj>)" AC conductance component for elect. *<ni>,<nj>*		
		"Y(<ni>,<nj>)" AC admittance component for elect. *<ni>,<nj>*		
		"SR(<ni>,<nj>)" Real component of S-param. for elect. *<ni>,<nj>*		
		"SI(<ni>,<nj>)" Imag. component of S-param. for elect. *<ni>,<nj>*		
		FREQ AC frequency		
		Programmable Device AAM Quantities		
		FE(<name>) FN tunneling current into electrode *<name>*		
		Circuit Analysis AAM Quantities		
		VC(<name>) voltage at circuit node *<name>*		
		IC(<name>) current in voltage source *<name>*		
		V(<dn>.<tn>) voltage at electrode *<tn>* of device *<dn>*		
		I(<dn>.<tn>) *total* current at electrode *<tn>* of device *<dn>*		
		IE(<dn>.<tn>) *electron* current at electrode *<tn>* of device *<dn>*		
		IH(<dn>.<tn>) *hole* current at electrode *<tn>* of device *<dn>*		
		ID(<dn>.<tn>) *displacement* current at electrode *<tn>* of device *<dn>*		
		Circuit Analysis AAM AC Quantities Note: Quotes are required.		
		"VCR(<name>)" AC real voltage component for node *<name>*		
		"VCI(<name>)" AC imaginary voltage comp. for node *<name>*		
		"ICR(<name>)" AC real current component for source *<name>*		
		"ICI(<name>)" AC imaginary current comp. for source *<name>*		
		Optical Device AAM Quantities		
		WA Wavelength (microns)		
		IT Intensity (W/cm^2)		
		LF Frequency of light modulation (Hz)		

Parameter	Type	Definition	Default	Units
		IP Internal photo current (Amps/micron)		
		EP External photo current (Amps/micron)		
		TR Transmittance at the illumination surface		
		RF Reflectance at the illumination surface		
		CE(name) External collection efficiency at electrode <*name*>		
		CI(name) Internal collection efficiency at electrode <*name*>		
		IS(name) Light modulation small signal current at elec <*name*>		
Y.AXIS	char	The quantity used for the vertical axis when plotting data stored in a log file. The choices are the same as given previously for the **X.AXIS** parameter.	none	
ORDER	logical	Specifies that the data points in log files are sorted by abscissa value before plotting. If **^ORDER** is specified, the data points in the log file are plotted as they occur.	true	
IN.FILE	char	The identifier for a log file containing either I-V data or AC data to be used when plotting terminal characteristics. If **IN.FILE** is not specified, data accumulated in the most recent log file during the present run is used. **synonym: INFILE**	none	
X.MIN	number	The minimum abscissa value plotted. Data points in the log file with smaller abscissa values are ignored.	The minimum value available.	abscissa dependent
X.MAX	number	The maximum abscissa value plotted. Data points in the log file with higher abscissa values are ignored.	The maximum value available.	abscissa dependent
CONDITIO	char	Specifies a numeric expression that must evaluate to "TRUE" before data will be plotted.	none	
Plot Controls				
SPLINE	logical	Specifies that spline smoothing is performed on the data.	false	
NSPLINE	number	The number of interpolated points to use when spline smoothing is specified. The maximum allowed is 1000.	100	none
LEFT	number	The value associated with the left end of the horizontal axis.	The minimum value available.	abscissa dependent
RIGHT	number	The value associated with the right end of the horizontal axis.	The maximum value available.	abscissa dependent
BOTTOM	number	The value associated with the bottom end of the vertical axis. **synonym: MIN.VALU**	The minimum value available.	ordinate dependent
TOP	number	The value associated with the top end of the vertical axis. **synonym: MAX.VALU**	The maximum value available.	ordinate dependent
UNCHANGE	logical	Specifies that the data is added to the previous plot. **UNCHANGE** has the effect of disabling **CLEAR** and **AXES**, and forces the previous axis bounds to be used for scaling. **UNCHANGE** can be used to plot more than one curve on the same plot.	false	

Parameter	Type	Definition	Default	Units
Y.LOGARI	logical	Specifies that a logarithmic vertical axis is used. **synonym: LOGARITH**	false	
S.LOGARI	logical	Specifies that a signed logarithmic vertical axis is used. To avoid overflow, the actual quantity plotted is given by sign(y) * log(1 + |y|).	false	
INTEGRAL	logical	Specifies that the integral of the ordinate is plotted.	false	
ABSOLUTE	logical	Specifies that the absolute value of the ordinate is plotted.	false	
NEGATIVE	logical	Specifies that the negative of the ordinate is plotted.	false	
CLEAR	logical	Specifies that the graphics display area is cleared before beginning the plot.	true	
AXES	logical	Specifies that the horizontal and vertical axes, axis labels, distance marks, and title are plotted.	true	
LABELS	logical	Specifies that axis labels are to be plotted.	true	
MARKS	logical	Specifies that distance marks are to be plotted along the plot axes.	true	
TITLE	char	The character string to be used as the title of the plot.	The character string in the most recent **TITLE** statement.	
T.SIZE	number	The height of the characters in the character string used as the plot title.	0.4	cm
X.OFFSET	number	The distance by which the left end of the horizontal axis is offset from the left edge of the graphics display area.	2.0	cm
X.LENGTH	number	The length of the horizontal axis.	screen width-**X.OFFSET**-1.25	cm
X.SIZE	number	The height of the characters used to label the horizontal axis.	0.25	cm
X.LOGARI	logical	Specifies that the horizontal axis is logarithmic.	false	
Y.OFFSET	number	The distance by which the bottom end of the vertical axis is offset from the bottom edge of the graphics display area.	2.0	cm
Y.LENGTH	number	The length of the vertical axis.	screen height-**Y.OFFSET**-1.25	cm
Y.SIZE	number	The height of the characters used to label the vertical axis at the left edge of the plot.	0.25	cm
CURVE	logical	Specifies that solid or dashed line curves are to be plotted connecting the data points in the plot.	true	
SYMBOL	number	The type of centered symbol plotted at the data points in the plot. The value of this parameter may lie in the range 1 to 15. If this parameter is not specified, the plot will not contain centered symbols. Values of this parameter are associated with the following symbol: 1 Square 2 Circle 3 Triangle 4 Plus 5 Upper case X 6 Diamond	none	none

Parameter	Type	Definition	Default	Units
		7 Up-arrow 8 Roofed upper case X 9 Upper case Z 10 Upper case Y 11 Curved square 12 Asterisk 13 Hourglass 14 Bar 15 Star		
POINTS	logical	Specifies that centered squares are plotted at the data points in the plot. This parameter has the same effect as specifying **SYMBOL**=1.	false	
C.SIZE	number	The size of the centered symbol used for the plot.	0.25	cm
LINE.TYP	number	The type of line used for the plot. A line type value of 1 generates a solid line plot. Line type values greater than 1 generate dashed line plots, with the dash size increasing with the value of line type.	1	none
COLOR	number	The index of the color used for the plot. The color associated with each color index is dependent upon the color graphics device that is used. This parameter has no effect if a color graphics device is not used.	1	none
DEVICE	char	The name of the graphics output device. Valid names are defined by the file *mdpdev* (see Chapter 1, "Plot Device Definition File—mdpdev" on page 1-14 and Appendix B). If the value of this parameter is "DEFAULT", the first entry in *mdpdev* preceded by "*" is chosen.	The last value specified or "DEFAULT"	
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. The user must enter a space followed by a carriage return to continue execution.	false	
PLOT.OUT	char	The identifier for the file in which the character sequences controlling the graphics device are saved. This file may be output to the graphics device to reproduce the graphical output. This output is only available for the direct device drivers such as those used when the **DEVICE** parameter is **HP2648**, **HP2623**, **HP7550**, **TEK4010**, **TEK4100**, **REGIS**, or **POSTSCRIPT**.	*<base>.dplt* if the **DF** entry is "T" in the file *mdp-dev*	
PLOT.BIN	char	The identifier for the file in which the binary information describing the graphical output is saved.	*<base>.bplt* if the **BF** entry is "T" in the file *mdp-dev*	
PRINT	logical	Specifies that the data points which are plotted are also printed to the standard output file.	false	
OUT.FILE	char	The identifier for a formatted file to store the values of the plotted data points. **synonym: OUTFILE**	none	
TIMESTAM	logical	Specifies that the date and time are to be plotted in the lower right corner of the plot. This option is not available on all computer systems.	false	
TIME.SIZ	number	The height of the characters used to plot the date and time.	0.25	cm
Circuit Analysis AAM Parameters				
STRUCTUR	char	Specifies the device to plot. This parameter is only used with the Circuit Analysis AAM.	first element	

Parameter	Type	Definition	Default	Units
Optical Device AAM Parameters				
WAVE.NUM	number	The identifying index for a wavelength to use for plotting **PHOTOGEN**. The specified index must correspond to one of the wavelengths actually considered during the photogeneration ray tracing analysis. This parameter is only used with the Optical Device AAM.	Contribution due to all wavelengths are included	none

Description

The **PLOT.1D** statement plots the following:

- A specific quantity along a line segments through the device (distance plots)

or

- Terminal characteristics from data accumulated in a log file or read in from a previous log file, I-V, AC, or user-defined (arbitrary) data

See Also...

To further illustrate the **PLOT.1D** statement, refer to:

- Input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Impurity Distribution Plots" on page 4-9
- Input file *mdex1g* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Gate Characteristics" on page 4-11
- Input file *mdex1d* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Drain Characteristics" on page 4-12
- Most other examples that have graphical output

Distance Plots

Plots of quantities along a specified line segment through the device require that a device structure be previously defined. This may be accomplished with a structure definition initiated by a **MESH** statement or by using the **IN.FILE** parameter on the **MESH** statement to input a structure file generated previously.

Plots of all quantities except impurity concentration (**DOPING**), require that a solution be present. This may be accomplished with a solution initiated by a **SOLVE** statement or by using a **LOAD** statement to input a data file generated previously by a **SOLVE** statement.

A distance plot requires specifying the endpoints of the line segment through the device along which the specified quantity is plotted. As an example, the following statement plots the potential horizontally through a device:

```
PLOT.1D   POTENTIAL  X.START=0  X.END=3  Y.START=0  Y.END=0
```

Minimum or Maximum Quantity and Location

The parameters **FIND.MIN** and **FIND.MAX** can be used to find and plot the minimum or maximum of the specified quantity as a function of distance along the specified line segment through the device.

For each point along the line segment, the program searches for the minimum or maximum value along a line that passes through the point and is perpendicular to the line segment.

The actual locations of the minimum or maximum can be found by specifying the **PRINT** parameter and examining the standard output listing. The search area can be configured to the following:

- Confined to semiconductor materials only (the default)
- Confined to insulator materials only
- Include all materials by using the parameters **SEMICOND** and **INSULATO**
- Confined to occur within a specified distance of the line segment by using the parameter **FIND.DIS**

Internal Plots of AC Quantities

Distance plots of quantities obtained from the results of an AC small-signal analysis can be obtained by specifying one of the parameters **AC.REAL**, **AC.IMAG**, **AC.MAGN**, or **AC.PHAS** in addition to the desired quantity (**POTENTIA**, **ELECTRON**, **HOLE**, **ELE.TEMP**, **HOL.TEMP**, **LAT.TEMP**, **J.CONDUC**, **J.ELECT**, **J.HOLE**, **J.DISPLA**, or **J.TOTAL**). For the current vectors, the quantity may be further qualified by specifying **X.COMPON** or **Y.COMPON**.

Plots of Log File Data

Plots of data contained in log files include:

- I-V and transient data
- Results of AC analysis
- Optical quantities
- User-defined or arbitrary quantities.

The **IN.FILE** parameter is used to specify the name of a log file containing the data of interest. If **IN.FILE** is not specified, the program attempts to use data from the most recently opened log file.

Log files are opened, named, and initiated using the **LOG** statement or, if no **LOG** statement is specified, a default log file with the name <*base*>.*ivl* is created.

Axis Quantities

A plot of log file data requires specifying the quantity to plot along each axis. As an example, the following statements plot the drain current of a MOSFET as a

function of the drain voltage and the drain current as a function of the gate voltage:

```
PLOT.1D   IN.FILE=IV.DAT  Y.AXIS=I(DRAIN)  X.AXIS=V(DRAIN)
PLOT.1D   IN.FILE=MOS.IVL Y.AXIS=I(DRAIN)  X.AXIS=V(GATE)
```

In the above examples, the I-V data was read from log files named *IV.DAT* and *MOS.IVL*.

Circuit Analysis AAM

With the Circuit Analysis AAM, node voltages and the currents flowing in voltage sources are also available as plot options for the x- or y-axis.

- Circuit node voltages are selected using *VC(<name>)*, where *<name>* is the name of the node of interest.
- Currents in voltage sources or inductors can be selected using *IC(<name>)*, where *<name>* is the name of the voltage source of interest.
- To plot the voltage or current at the terminal of a numerical device, use *V(*<d*name>.<tname>)* or *I(*<d*name>.<tname>)*, where *P<dname>* is the name of the Medici device and *<tname>* is the electrode name.

The following examples plot the voltage at circuit node 1, the current in source *VDD*, and the current at the drain terminal of device *P4* respectively.

```
PLOT.1D   X.AXIS=TIME  Y.AXIS=VC(1)
PLOT.1D   X.AXIS=TIME  Y.AXIS=IC(VDD)
PLOT.1D   X.AXIS=TIME  Y.AXIS=I(P4.drain)
```

Other Information

This section contains additional information important to using **PLOT.1D**. It includes the following:

- Disabling the clear operation
- Plotting more than one curve
- Integrating the abscissa
- Labeling
- Electric field lines

Specify **^CLEAR** to disable the clear operation, and leave the previous plot intact. This allows two or more independent plots to be displayed simultaneously by overriding clearing the display device when a plot is initialized.

The **UNCHANGE** parameter can be used to plot more than one curve on the same plot. **UNCHANGE** has the same effect as specifying **^CLEAR** and **^AXES**, and additionally forces the previous axis bounds to be used for scaling.

INTEGRAL can be used to integrate the specified function over the abscissa coordinate. If **INTEGRAL** is specified, the parameters **X.LOGARI**, **Y.LOGARI**, and

S.LOGARI should not be specified. If **INTEGRAL** is specified, and either of the parameters **ABSOLUTE** or **NEGATIVE** are also specified, then the absolute value or negative, respectively, of the specified function is taken before the integration is performed.

The **PLOT.1D** statement may be followed by any number of **LABEL** statements to facilitate placing labels in the graphical output.

E.LINE statements for plotting quantities along electric field lines may also follow **PLOT.1D** statements.

PLOT.2D

The **PLOT.2D** statement initializes the graphical display device for two-dimensional plots of device characteristics and plots device boundaries, metallurgical junctions, and depletion region edges.

```
PLOT.2D

   [BOUNDARY [REGION] ] [JUNCTION] [DEPLETIO] [LUMPED] [CON.RESI]
   [ GRID [ELEM.NUM] [NODE.NUM] [REG.NUM] [N.SIZE=<n>] [OBTUSE] ]
   [CROSSES] [FILL] [SCALE]
   [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
   [CLEAR] [LABELS] [MARKS] [TOP.MARK] [TITLE=<c>] [T.SIZE=<n>]
   [L.BOUND=<n>] [L.JUNCT=<n>] [L.DEPLE=<n>] [L.GRID=<n>] [L.ELECT=<n>]
   [C.BOUND=<n>] [C.JUNCT=<n>] [C.DEPLE=<n>] [C.GRID=<n>] [C.ELECT=<n>]
   [X.OFFSET=<n>] [X.LENGTH=<n>] [X.SIZE=<n>]
   [Y.OFFSET=<n>] [Y.LENGTH=<n>] [Y.SIZE=<n>]
   [DEVICE=<c>] [PLOT.OUT=<c>] [PLOT.BIN=<c>] [PAUSE]
   [TIMESTAM [TIME.SIZ=<n>] ]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]

 Optical Device AAM Parameter
   [RAYPLOT [WAVE.NUM=<n>] ]
```

Parameter	Type	Definition	Default	Units
BOUNDARY	logical	Specifies that the boundary of the device is to be plotted. This includes the boundary of all device regions and electrodes.	false	
REGION	logical	Specifies that the boundary between regions of the same material is plotted.	True if **BOUNDARY** is specified.	
JUNCTION	logical	Specifies that the locations of metallurgical junctions are to be plotted.	false	
DEPLETIO	logical	Specifies that the locations of the edges of depletion regions are to be plotted.	false	
LUMPED	logical	Specifies that schematic representations for lumped resistances and capacitances are plotted at each contact for which a lumped resistance or capacitance was specified.	false	
CON.RESI	logical	Specifies that a schematic representation for contact resistance is plotted at each contact for which contact resistance was specified.	false	
GRID	logical	Specifies that the simulation grid showing the boundaries of each mesh element is plotted. **synonym: MESH**	false	
ELEM.NUM	logical	Specifies that numbers which identify the elements are plotted.	false	
NODE.NUM	logical	Specifies that numbers which identify the nodes are plotted.	false	

Parameter	Type	Definition	Default	Units
REG.NUM	logical	Specifies that the region index for each element is plotted.	false	
N.SIZE	number	The size of the numbers which are plotted if **ELEM.NUM**, **NODE.NUM**, or **MAT.NUM** is specified.	0.25	cm
OBTUSE	logical	Specifies that obtuse triangles in the grid structure are color filled on output devices that support this operation.	false	
CROSSES	logical	Specifies that crosses at the locations of the grid points are plotted.	false	
FILL	logical	Specifies that material regions are color filled. The colors used to fill the various material regions can be specified on the **FILL** statement.	false	
SCALE	logical	Specifies that the size of the plot is reduced from the specified size in either the x or y direction so that the same scale factor is used in both the x and y directions. This parameter facilitates visualization of the device in its proper aspect ratio.	false	
X.MIN	number	The x coordinate in the device coordinate system of the left edge of the display area.	The minimum x location in the device structure.	microns
X.MAX	number	The x coordinate in the device coordinate system of the right edge of the display area.	The maximum x location in the device structure.	microns
Y.MIN	number	The y coordinate in the device coordinate system of the top edge of the display area.	The minimum y location in the device structure.	microns
Y.MAX	number	The y coordinate in the device coordinate system of the bottom edge of the display area.	The maximum y location in the device structure.	microns
CLEAR	logical	Specifies that the graphics display area is cleared before beginning the plot.	true	
LABELS	logical	Specifies that axis and distance labels are plotted along the left and bottom sides of the plot.	true	
MARKS	logical	Specifies that distance marks are plotted around the plotted area.	true	
TOP.MARK	logical	Specifies that distance marks are plotted along the top boundary of the plotted area.	true	
TITLE	char	The character string to be used as the title of the plot.	The character string in the most recent **TITLE** statement.	
T.SIZE	number	The height of the characters in the character string used as the plot title.	0.4	cm
L.BOUND	number	The index of the line type used to plot the boundaries of the device.	1	none
L.JUNCT	number	The index of the line type used to plot metallurgical junctions.	3	none
L.DEPLE	number	The index of the line type used to plot depletion edges.	2	none

Parameter	Type	Definition	Default	Units
L.GRID	number	The index of the line type used to plot the simulation grid.	1	none
L.ELECT	number	The index of the line type used to plot electrodes. If a negative value is specified, only the boundary of the electrode is plotted. In this case, the electrode boundary is plotted with a line type equal to |**L.ELECT**|.	4	none
C.BOUND	number	The index of the color used to plot the boundaries of the device.	1	none
C.JUNCT	number	The index of the color used to plot metallurgical junctions.	1	none
C.DEPLE	number	The index of the color used to plot depletion edges.	2	none
C.GRID	number	The index of the color used to plot the simulation grid.	1	none
C.ELECT	number	The index of the color used to plot electrodes.	4	none
X.OFFSET	number	The distance by which the left edge of the boundary is offset from the left edge of the graphics display area.	2.0	cm
X.LENGTH	number	The horizontal length of the plot.	screen width - **X.OFFSET** - 1.25	cm
X.SIZE	number	The height of the characters used to label the horizontal boundary at the bottom of the plot.	0.25	cm
Y.OFFSET	number	The distance by which the bottom boundary is offset from the bottom edge of the graphics display area.	2.0	cm
Y.LENGTH	number	The vertical height of the plot.	screen height - **Y.OFFSET** - 1.25	cm
Y.SIZE	number	The height of the characters used to label the vertical boundary at the left edge of the plot.	0.25	cm
DEVICE	char	The name of the graphics output device. Valid names are defined by the file *prdvmdpdev* (see Chapter 1, "Plot Device Definition File—mdpdev" on page 1-14 and Appendix B). If the value of this parameter is "DEFAULT", the first entry in *prdvmdpdev* preceded by "*" is chosen.	The last value specified or "DEFAULT".	
PLOT.OUT	char	The identifier for the file in which the character sequences controlling the graphics device are saved. This file may be output to the graphics device to reproduce the graphical output. This output is only available for the direct device drivers such as those used when the **DEVICE** parameter is **HP2648**, **HP2623**, **HP7550**, **TEK4010**, **TEK4100**, **REGIS**, or **POSTSCRIPT**.	<*base*>.*dplt* if the **DF** entry is "T" in the file *prdvmdpdev*	
PLOT.BIN	char	The identifier for the file in which the binary information describing the graphical output is saved.	<*base*>.*bplt* if the **BF** entry is "T" in the file *prdvmdpdev*	
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. The user must enter a space followed by a carriage return to continue execution.	false	
TIMESTAM	logical	Specifies that the date and time are to be plotted in the lower right corner of the plot. This option is not available on all computer systems.	false	

Parameter	Type	Definition	Default	Units
TIME.SIZ	number	The height of the characters used to plot the date and time.	0.25	cm
Circuit Analysis AAM Parameters				
STRUCTUR	char	Specifies the device to plot. This parameter is only used with the Circuit Analysis AAM.	first element	
Optical Device AAM Parameters				
RAYPLOT	logical	Resizes the plot window for plotting optical rays. This parameter is only used with the Optical Device AAM.	false	
WAVE.NUM	number	The identifying index for a wavelength to use when plotting the ray tracing results. The specified index must correspond to one of the wavelengths actually considered during the photogeneration ray tracing analysis. This parameter is only used with the Optical Device AAM.	All wavelengths are included.	none

Description

The **PLOT.2D** statement initializes the graphical display device for two-dimensional plots of the following:

- Device characteristics and plots device boundaries
- Metallurgical junctions
- Depletion region edges

The plot sequence may include **CONTOUR**, **VECTOR**, **E.LINE**, **FILL**, and **LABEL** statements to generate various plotted output. **TITLE** and **COMMENT** statements may also appear within the plot sequence.

See Also...

To further illustrate the **PLOT.2D** statement, refer to:

- Input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Generation of the Simulation Structure" on page 4-2
- Input file *mdex1d* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Drain Characteristics" on page 4-12
- Most other examples that have graphical output

Disabling Clear

Specify **^CLEAR** to disable the clear operation, and leave the previous plot intact. This allows two or more independent plots to be displayed simultaneously by overriding clearing the display device when a plot is initialized. This facilitates the comparison of results for different device structures or bias conditions.

Physical Device Boundaries

A plot of the physical device boundaries is obtained by specifying the **BOUNDARY** parameter. The locations plotted include:

- Outer extremities of the device structure
- Boundaries between different regions
- Electrode locations

Only those boundaries occurring within the display area are plotted.

Junctions, Depletion Regions, and Grid

The locations of metallurgical junctions are plotted if the **JUNCTION** parameter is specified. A metallurgical junction occurs where the net impurity concentration is zero.

The locations of the edges of depletion regions are plotted if the **DEPLETIO** parameter is specified. The edge of a depletion region is defined as the boundary where the majority carrier concentration equals half of the net impurity concentration.

The discretization grid used for the solutions may be plotted by using the **GRID** parameter.

Schematic Representations

The parameters **LUMPED** and **CON.RESI** can be used to plot schematic representations of lumped elements and contact resistance, respectively, at contacts where these quantities were specified.

The lumped element representation consists of a single resistor symbol, single capacitor symbol, or the parallel combination of a resistor and capacitor, drawn at the center of the outer boundary of the contact. The contact resistance representation consists of a single resistor symbol drawn at each node of the outer boundary of the contact.

Device Structure

Plots of the device boundaries, metallurgical junctions, or discretization grid, require that a device structure be previously defined. This may be done with a structure definition initiated by a **MESH** statement or by using the **IN.FILE** parameter on the **MESH** statement to input a data file generated previously.

Required Solutions

Plots of depletion edge locations or lumped element and contact resistance representations additionally require that a solution be present. This may be accomplished with a solution initiated by a **SOLVE** statement or by using a **LOAD** statement to input a data file generated previously by a **SOLVE** statement.

PLOT.3D

The **PLOT.3D** statement is used to initialize the graphical display device for three-dimensional plots of physical quantities and defines the placement, size, and rotation of the plot axes.

```
PLOT.3D

 Plot Quantities
   {  POTENTIA | QFN | QFP | VALENC.B | CONDUC.B | VACUUM | E.FIELD
    | DOPING | ELECTRON | HOLES | NIE | NET.CHAR | NET.CARR
    | J.CONDUC | J.ELECTR | J.HOLE | J.DISPLA | J.TOTAL
    | RECOMBIN | N.RECOMB | P.RECOMB | II.GENER | BB.GENER | PHOTOGEN
    | ELE.TEMP | HOL.TEMP | ELE.VEL | HOL.VEL | J.EFIELD
    | G.GAMN | G.GAMP | G.GAMT | G.IN | G.IP | G.IT
    | ARRAY1 | ARRAY2 | ARRAY3 | ( {TRAPS | TRAP.OCC} [LEVEL=<n>] )
    | N.MOBILI | P.MOBILI | SIGMA | LAT.TEMP | X.MOLE
    | IMPURITY=<c> | OTHER=<c>
  }

 AC Small-Signal Analysis Quantity Parameters
  [ {AC.REAL | AC.IMAG | AC.MAGN | AC.PHAS} ]

 Plot Controls
   [X.COMPON] [Y.COMPON] [Z.MIN=<n>] [Z.MAX=<n>] [ABSOLUTE] [LOGARITH]

 Device Bounds
   [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]

 Viewing Parameters
   [THETA=<n>] [PHI=<n>] [X.LINES=<n>] [Y.LINES=<n>] [EQUIDIST]

 Axis and Labels
   [AXES] [LABELS] [MARKS] [TITLE=<c>] [T.SIZE=<n>]
   [X.LENGTH=<n>] [Y.LENGTH=<n>] [Z.LENGTH=<n>]
   [X.LABEL=<c>]  [Y.LABEL=<c>]  [Z.LABEL=<c>]
   [X.SIZE=<n>]   [Y.SIZE=<n>]   [Z.SIZE=<n>]

 Display Parameters
   [CLEAR] [FRAME] [CENTER] [FILL.VIE]
   [XV.LENGT=<n>] [XV.OFFSE=<n>] [YV.LENGT=<n>] [YV.OFFSE=<n>]
   [X.OFFSET=<n>] [Y.OFFSET=<n>]
   [DEVICE=<c>] [L.BOX=<n>] [C.BOX=<n>] [PAUSE]
   [ TIMESTAM [TIME.SIZ=<n>] ]

 File Output
   [PLOT.OUT=<c>] [PLOT.BIN=<c>]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]
```

Parameter	Type	Definition	Default	Units
Plot Quantities				
POTENTIA	logical	Specifies that mid-gap potential in volts is plotted over a specified cross-section of the device.	false	
QFN	logical	Specifies that the electron quasi-Fermi potential in volts is plotted over a specified cross-section of the device.	false	
QFP	logical	Specifies that the hole quasi-Fermi potential in volts is plotted over a specified cross-section of the device.	false	
VALENC.B	logical	Specifies that the valence band potential in volts is plotted over a specified cross-section of the device.	false	
CONDUC.B	logical	Specifies that the conduction band potential in volts is plotted over a specified cross-section of the device.	false	
VACUUM	logical	Specifies that the vacuum potential in volts is plotted over a specified cross-section of the device.	false	
E.FIELD	logical	Specifies that the magnitude of electric field in volts per centimeter is plotted over a specified cross-section of the device.	false	
DOPING	logical	Specifies that the net impurity concentration in number per cubic centimeter is plotted over a specified cross-section of the device. The net impurity concentration is the donor impurity concentration minus the acceptor impurity concentration.	false	
ELECTRON	logical	Specifies that electron concentration in number per cubic centimeter is plotted over a specified cross-section of the device.	false	
HOLES	logical	Specifies that hole concentration in number per cubic centimeter is plotted over a specified cross-section of the device.	false	
NIE	logical	Specifies that effective intrinsic carrier concentration in number per cubic centimeter is plotted over a specified cross-section of the device.	false	
NET.CHAR	logical	Specifies that the net charge concentration in number per cubic centimeter is plotted over a specified cross-section of the device. The net charge concentration is the sum of the donor impurity concentration and hole concentration minus the sum of the acceptor impurity concentration and electron concentration plus the concentration of any trapped charge.	false	
NET.CARR	logical	Specifies that the net carrier concentration in number per cubic centimeter is plotted over a specified cross-section of the device. The net carrier concentration is the hole concentration minus the electron concentration.	false	
J.CONDUC	logical	Specifies that conduction current in amps per square centimeter is plotted over a specified cross-section of the device.	false	
J.ELECTR	logical	Specifies that electron current in amps per square centimeter is plotted over a specified cross-section of the device.	false	
J.HOLE	logical	Specifies that hole current in amps per square centimeter is plotted over a specified cross-section of the device.	false	
J.DISPLA	logical	Specifies that displacement current in amps per square centimeter is plotted over a specified cross-section of the device.	false	

Parameter	Type	Definition	Default	Units
J.TOTAL	logical	Specifies that total current in amps per square centimeter is plotted over a specified cross-section of the device.	false	
RECOMBIN	logical	Specifies that net recombination in number per cubic centimeter per second is plotted over a specified cross-section of the device. For unequal electron and hole recombination, **RECOMBIN** is the same as **N.RECOMB**.	false	
N.RECOMB	logical	Specifies that net electron recombination in number per cubic centimeter per second is plotted over a specified cross-section of the device.	false	
P.RECOMB	logical	Specifies that net hole recombination in number per cubic centimeter per second is plotted over a specified cross-section of the device.	false	
II.GENER	logical	Specifies that the total generation rate due to impact ionization in pairs per cubic centimeter per second is plotted over a specified cross-section of the device.	false	
BB.GENER	logical	Specifies that the total generation rate due to band-to-band tunneling in pairs per cubic centimeter per second is plotted over a specified cross-section of the device.	false	
PHOTOGEN	logical	Specifies that total photogeneration in pairs per cubic centimeter per second is plotted over a specified cross-section of the device.	false	
ELE.TEMP	logical	Specifies that the electron temperature in Kelvins is plotted over a specified cross-section of the device.	false	
HOL.TEMP	logical	Specifies that the hole temperature in Kelvins is plotted over a specified cross-section of the device.	false	
ELE.VEL	logical	Specifies that the magnitude of the electron velocity in cm/s is plotted over a specified cross-section of the device.	false	
HOL.VEL	logical	Specifies that the magnitude of the hole velocity in cm/s is plotted over a specified cross-section of the device.	false	
J.EFIELD	logical	Specifies that the component of the electric field in the direction of current density in volts per centimeter is plotted over a specified cross-section of the device.	false	
G.GAMN	logical	Specifies that the probability that an electron is injected into the oxide is plotted over a specified cross-section of the device.	false	
G.GAMP	logical	Specifies that the probability that a hole is injected into the oxide is plotted over a specified cross-section of the device.	false	
G.GAMT	logical	Specifies that the probability that an electron or hole (that is, the sum of the hole and electron probabilities) is injected into the oxide is plotted over a specified cross-section of the device.	false	
G.IN	logical	Specifies that hot electron injection current initiated from each point in amps/micron is plotted over a specified cross-section of the device.	false	
G.IP	logical	Specifies that hot hole injection current initiated from each point in amps/micron is plotted over a specified cross-section of the device.	false	
G.IT	logical	Specifies that total hot carrier injection current initiated from each point in amps/micron is plotted over a specified cross-section of the device.	false	
ARRAY1	logical	Specifies that the user generated array number 1 is to be plotted. Refer to the **EXTRACT** statement for more information.	false	

Parameter	Type	Definition	Default	Units
ARRAY2	logical	Specifies that the user generated array number 2 is to be plotted. Refer to the **EXTRACT** statement for more information.	false	
ARRAY3	logical	Specifies that the user generated array number 3 is to be plotted. Refer to the **EXTRACT** statement for more information.	false	
TRAPS	logical	Specifies that the trap density in number per cubic centimeter is to be plotted.	false	
TRAP.OCC	logical	Specifies that the filled trap density in number per cubic centimeter is to be plotted.	false	
LEVEL	number	The specific trap level to plot. If this parameter is not specified, all trap levels are summed.	none	none
N.MOBILI	logical	Specifies that the electron mobility in cm^2/V-s is plotted over a specified cross-section of the device.	false	
P.MOBILI	logical	Specifies that the electron mobility in cm^2/V-s is plotted over a specified cross-section of the device.	false	
SIGMA	logical	Specifies that the conductivity in $(Ohm\text{-}cm)^{-1}$is plotted over a specified cross-section of the device.	false	
LAT.TEMP	logical	Specifies that the lattice temperature in Kelvins is plotted over a specified cross-section of the device. This parameter is only used with the Lattice Temperature AAM.	false	
X.MOLE	logical	Specifies that mole fraction is plotted over a specified cross-section of the device. This parameter is only used with the Heterojunction Device AAM.	false	
IMPURITY	char	The name of an impurity to plot in number per cubic centimeter over a specified cross-section of the device.	none	
OTHER	char	The name of an **OTHER** quantity to plot over a specified cross-section of the device.	none	
AC Small-Signal Analysis Quantity Parameters				
AC.REAL	logical	Specifies that the real part of the quantity obtained from AC analysis is plotted	false	
AC.IMAG	logical	Specifies that the imaginary part of the quantity obtained from AC analysis is plotted.	false	
AC.MAGN	logical	Specifies that the magnitude of the quantity obtained from AC analysis is plotted.	false	
AC.PHAS	logical	Specifies that the phase of the quantity obtained from AC analysis is plotted. Phase is defined as $\text{atan}(\text{imag}(X)/\text{real}(X))$, where X represents the quantity to be plotted.	false	
Plot Controls				
X.COMPON	logical	Specifies that the x component of a vector quantity is plotted as opposed to the default magnitude.	false	
Y.COMPON	logical	Specifies that the y component of a vector quantity is plotted as opposed to the default magnitude.	false	

Parameter	Type	Definition	Default	Units
Z.MIN	number	The value of the specified quantity associated with the minimum extent of the z axis. **synonym: MINIMUM**	minimum of data	Dependent on plotted quantity
Z.MAX	number	The value of the specified quantity associated with the maximum extent of the z axis. **synonym: MAXIMUM**	maximum of data	Dependent on plotted quantity
ABSOLUTE	logical	Specifies that the absolute value of the specified quantity is to be plotted.	false	
LOGARITH	logical	Specifies that the data axis is logarithmic. To avoid errors, the actual quantity plotted is given by log(|z|). **synonym: Z.LOGARI**	false	
Device Bounds				
X.MIN	number	The horizontal distance in the device coordinate system associated with the minimum extent of the x-axis.	Minimum x location in structure.	microns
X.MAX	number	The horizontal distance in the device coordinate system associated with the maximum extent of the x-axis.	Maximum x location in structure.	microns
Y.MIN	number	The vertical distance in the device coordinate system associated with the minimum extent of the y-axis.	Minimum y location in structure.	microns
Y.MAX	number	The vertical distance in the device coordinate system associated with the maximum extent of the y-axis.	Maximum y location in structure.	microns
Viewing Parameters				
THETA	number	The angle of rotation of the plot axes about the x-axis (z into y).	50.0	degrees
PHI	number	The angle of rotation of the plot axes about the z-axis (y into x).	50.0	degrees
X.LINES	number	Specifies the number of lines to use in the x direction when interpolating from the Medici grid to the rectangular plotting grid.	automatic	none
Y.LINES	number	Specifies the number of lines to use in the y direction when interpolating from the Medici grid to the rectangular plotting grid.	automatic	none
EQUIDIST	logical	Specifies that the rectangular plotting grid lines are equally spaced in the x and y directions. If false, then the number of lines specified by **X.LINES** and **Y.LINES** is distributed throughout the rectangular plotting grid according to the density of nodes in the Medici grid.	true	
Axes and Labels				
AXES	logical	Specifies that the x-, y-, and z-axes are to be plotted.	true	
LABELS	logical	Specifies that labels are to be plotted along the axes.	true	
MARKS	logical	Specifies that marks are to be plotted along the axes.	true	

Parameter	Type	Definition	Default	Units
TITLE	char	The character string to be used as the plot title.	Character string in most recent **TITLE** statement.	
T.SIZE	number	The height of the characters in the character string used as the plot title.	0.4	cm
X.LENGTH	number	The length of the x-axis when **THETA**=0, **PHI**=0.	0.5 * min(**XV.LENGT**, **YV.LENGT**)	cm
Y.LENGTH	number	The length of the y-axis when this axis lies vertically in the viewport plane (**THETA**=270, **PHI**=0).	0.5 * min(**XV.LENGT**, **YV.LENGT**)	cm
Z.LENGTH	number	The length of the z-axis when **THETA**=0,**PHI**=0.	0.5 * min(**XV.LENGT**, **YV.LENGT**)	cm
X.LABEL	char	Label for the x-axis.	"X (um)"	
Y.LABEL	char	Label for the y-axis.	"Y (um)"	
Z.LABEL	char	Label for the z-axis.	Label representing the plotted quantity.	
X.SIZE	number	The height of characters used to label the x-axis.	0.25	cm
Y.SIZE	number	The height of the characters used to label the y-axis.	0.25	cm
Z.SIZE	number	The height of the characters used to label the z-axis.	0.25	cm

Display Parameters

CLEAR	logical	Specifies that the graphics display area is to be cleared before beginning the plot.	true	
FRAME	logical	Specifies that the viewport window frame is to be plotted around the plot area. **synonym**: **VIEWPORT**	true	
CENTER	logical	Specifies that the plot is to be centered in the viewport window. If the value of this parameter is false, then the top vertex of the plot is placed at the top of the viewport.	true	
FILL.VIE	logical	Specifies that the plot is to be scaled to the maximum size that fits inside the viewport window.	true	
XV.LENGT	number	The width of the viewport.	screen width - **XV.OFFSE**	cm
XV.OFFSE	number	The horizontal distance by which the left edge of the viewport is offset from the left edge of the graphics display area.	0.0	cm
YV.LENGT	number	The height of the viewport.	screen height - **YV.OFFSE** - 1.25	cm
YV.OFFSE	number	The vertical distance by which the bottom edge of the viewport is offset from the bottom edge of the graphics display area.	0.0	cm

Parameter	Type	Definition	Default	Units
X.OFFSET	number	The horizontal distance by which the plot is shifted from its default location in the viewport.	0.0	cm
Y.OFFSET	number	The vertical distance by which the plot is shifted from its default location in the viewport.	0.0	cm
DEVICE	char	The name of the graphics output device. Valid names are defined by the file *prdvmdpdev* (see Chapter 1, "Plot Device Definition File—mdpdev" on page 1-14 and Appendix B). If the value of this parameter is "DEFAULT", the first entry in *prdvmdpdev* preceded by "*" is chosen.	The last value specified or "DEFAULT".	
L.BOX	number	The type of line used to plot the viewport window. A line type value of 1 generates a solid line plot. Line type values greater than 1 generate dashed line plots, with the dash size increasing with the value of the line type.	1	none
C.BOX	number	The index of the color used to plot the viewport window and axes. The color associated with each color index is dependent upon the color graphics device that is used. This parameter has no effect if a color graphics device is not used.	1	none
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. The user must enter a space followed by a carriage return to continue execution.	false	
TIMESTAM	logical	Specifies that the date and time are to be plotted in the lower right corner of the plot. This option is not available on all computer systems.	false	
TIME.SIZ	number	The height of the characters used to plot the date and time.	0.25	cm
File Output				
PLOT.OUT	char	The identifier for the file in which the character sequences controlling the graphics device are saved. This file may be output to the graphics device to reproduce the graphical output. This output is only available for the direct device drivers such as those used when the **DEVICE** parameter is **HP2648**, **HP2623**, **HP7550**, **TEK4010**, **TEK4100**, **REGIS**, or **POSTSCRIPT**.	<*base*>.*dplt* if the **DF** entry is "T" in the file *prdvmdpdev*	
PLOT.BIN	char	The identifier for the file in which the binary information describing the graphical output is saved.	<*base*>.*bplt* if the **BF** entry is "T" in the file *prdvmdpdev*	
Circuit Analysis AAM Parameters				
STRUCTUR	char	Specifies the device to plot. This parameter is only used with the Circuit Analysis AAM.	first element	

Description

The **PLOT.3D** statement is used to initialize the graphical display device (such as a graphics terminal or a pen plotter) for three-dimensional plots of physical quantities. **PLOT.3D** defines the placement, size, and rotation of the plot axes.

This sequence is initiated with a **PLOT.3D** statement and may include one or more **3D.SURFACE** statements. **TITLE** and **COMMENT** statements can also appear within the three-dimensional plot sequence.

See Also... To further illustrate the **PLOT.3D** statement, refer to input files:

- *mdex7b* in Template Examples, Chapter 8, "NPN Bipolar Junction Transistor Example" on page 8-11
- *mdex14s* in Lattice Temperature Examples, Chapter 13, "SOI Drain Characteristics Example" on page 13-1

Viewport, Plot Axes, and Rotation Angles

This section details the functions of the viewport, plot axes, and rotation angles. See Figure 3-25.

Viewport

The viewport serves as a reference frame for the placement of the plot axes. The x and y axes correspond to the two-dimensional device coordinate system axes. The physical quantity specified on the **PLOT.3D** statement is plotted along the z-axis.

Plot Axes and Rotation Angle

The orientation of the plot axes with respect to the viewport plane is specified by the angles **THETA** and **PHI**. Values of zero for **THETA** and **PHI** produce a plot with the x and z axes lying in the viewport plane and the y-axis extending out of the plane, directly toward the observer. **THETA** and **PHI** can be specified to have any positive or negative values. They are reduced modulo 360 to values between -360 and +360 degrees.

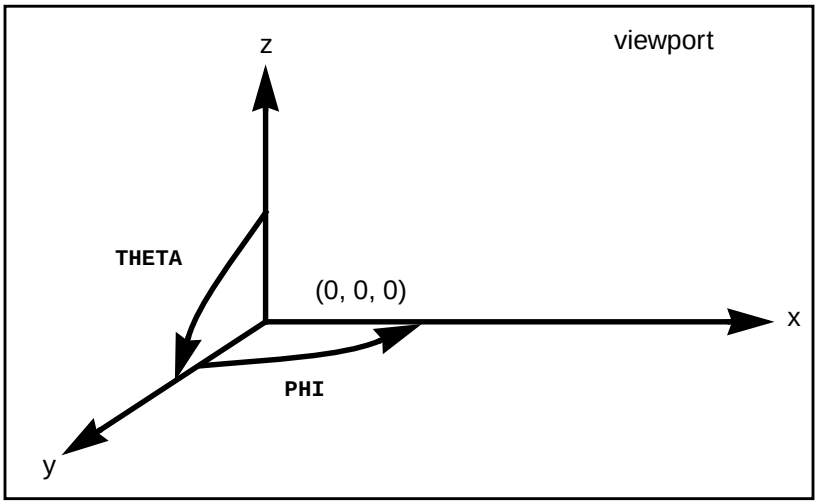


Figure 3-25 Viewport, plot axes, and rotation angles

3D.SURFACE

The **3D.SURFACE** statement plots the projection of a three-dimensional view of the specified data onto a two-dimensional viewport.

3D.SURFACE

[HIDDEN] [VISIBLE] [LOWER] [UPPER] [X.LINE] [Y.LINE] [MASK]
[Z.MIN=<n>] [Z.MAX=<n>] [C.AUTO] [LINE.TYP=<n>] [COLOR=<n>] [PAUSE]

Parameter	Type	Definition	Default	Units
HIDDEN	logical	Specifies that the surface lines which are hidden to the viewer by other parts of the surface to be plotted.	false	
VISIBLE	logical	Specifies that the surface lines which are visible to the viewer to be plotted.	true	
LOWER	logical	Specifies that the surface lines associated with the bottom side of the surface to be plotted.	true	
UPPER	logical	Specifies that the surface lines associated with the top side of the surface to be plotted.	true	
X.LINE	logical	Specifies that surface lines parallel to the x-axis to be plotted.	true	
Y.LINE	logical	Specifies that surface lines parallel to the y-axis to be plotted.	true	
MASK	logical	Specifies that the surface lines which are above the maximum clipping level or below the minimum clipping level are removed and do not affect the visibility of any of the other plotted surface lines.	false	
Z.MIN	number	The value defining the minimum clipping plane. **synonym: MINIMUM**	minimum of data	Dependent on plotted quantity
Z.MAX	number	The value defining the maximum clipping plane. **synonym: MAXIMUM**	maximum of data	Dependent on plotted quantity
C.AUTO	logical	Causes the color of the surface to vary with the data. The colors used follow the spectrum with violet indicating the largest z value and red the smallest z value.	false	
LINE.TYP	number	The type of line used to plot the data. A line type value of 1 generates a solid line plot. Line type values greater than 1 generate dashed line plots, with the dash size increasing with the value of the line type.	1	none
COLOR	number	The index of the color used for the plot. The color associated with each color index is dependent upon the color graphics device that is used. This parameter has no effect if a color graphics device is not used.	1	none
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. The user must enter a space followed by a carriage return to continue execution.	false	

Description

The **3D.SURFACE** statement plots the projection of a three-dimensional view of the specified data onto a two-dimensional viewport.

See Also... To further illustrate the **3D.SURFACE** statement, refer to input files:

- *mdex7b* in Template Examples, Chapter 8, "NPN Bipolar Junction Transistor Example" on page 8-11
- *mdex14s* in Lattice Temperature Examples, Chapter 13, "SOI Drain Characteristics Example" on page 13-1

Limitations and Plotting Combinations

The **3D.SURFACE** statement may only be used within plot sequences initiated by the **PLOT.3D** statement. Any combination of the following may be plotted:

- Visible and hidden lines
- Upper and lower surface lines,
- x and y grid lines
- Lines lying between specified minimum and maximum clipping planes

The axes, if specified to be plotted on the preceding **PLOT.3D** statement, are plotted immediately following the first specified surface to be plotted. The axes are clipped to conform with the view of this first surface.

Quantity Ranges

Various ranges of the quantity being plotted can be distinguished by using multiple **3D.SURFACE** statements. This is accomplished with different clipping plane values and distinct **COLOR** and/or **LINE.TYP** values for each range. Alternatively, **C.AUTO** can be used to obtain this type of plot automatically.

CONTOUR

The **CONTOUR** statement plots contours of various physical quantities on a two-dimensional area of the device as defined on the most recent **PLOT.2D** statement.

```
CONTOUR

 Contour Quantities
   {  POTENTIA | QFN | QFP | VALENC.B | CONDUC.B | VACUUM | E.FIELD
    | DOPING | ELECTRON | HOLES | NIE | NET.CHAR | NET.CARR
    | J.CONDUC | J.ELECTR | J.HOLE | J.DISPLA | J.TOTAL | FLOWLINE
    | RECOMBIN | N.RECOMB | P.RECOMB | II.GENER | BB.GENER | PHOTOGEN
    | ELE.TEMP | HOL.TEMP | ELE.VEL | HOL.VEL | J.EFIELD
    | G.GAMN | G.GAMP | G.GAMT | G.IN | G.IP | G.IT
    | ARRAY1 | ARRAY2 | ARRAY3 | ( {TRAPS | TRAP.OCC} [LEVEL=<n>] )
    | N.MOBILI | P.MOBILI | SIGMA
    | IMPURITY=<c> | OTHER=<c>

 Lattice Temperature AAM Parameters
    | LAT.TEMP

 Heterojunction Device AAM Parameters
    | X.MOLE
   }

 AC Small-Signal Analysis Quantity Parameters
   [ {AC.REAL | AC.IMAG | AC.MAGN | AC.PHAS} ]

 Contour Controls
   [MIN.VALU=<n>] [MAX.VALU=<n>] [WINDOW] [DEL.VALU=<n>] [NCONTOUR=<n>]
   [ FILL [C.START=<n>] [C.INCREM=<n>] ]
   [ABSOLUTE] [LOGARITH] [X.COMPON] [Y.COMPON]
   [LINE.TYP=<n>] [COLOR=<n>] [PAUSE]
```

Parameter	Type	Definition	Default	Units
Contour Quantities				
POTENTIA	logical	Specifies that contours of constant mid-gap potential in volts are plotted.	false	
QFN	logical	Specifies that contours of constant electron quasi-Fermi potential in volts are plotted.	false	
QFP	logical	Specifies that contours of constant hole quasi-Fermi potential in volts are plotted.	false	
VALENC.B	logical	Specifies that contours of constant valence band potential in volts are plotted.	false	
CONDUC.B	logical	Specifies that contours of constant conduction band potential in volts are plotted.	false	
VACUUM	logical	Specifies that contours of constant vacuum potential in volts are plotted.	false	

Parameter	Type	Definition	Default	Units
E.FIELD	logical	Specifies that contours of constant electric field magnitude in volts per centimeter are plotted.	false	
DOPING	logical	Specifies that contours of constant net impurity concentration in number per cubic centimeter are plotted.	false	
ELECTRON	logical	Specifies that contours of constant electron concentration in number per cubic centimeter are plotted.	false	
HOLES	logical	Specifies that contours of constant hole concentration in number per cubic centimeter are plotted.	false	
NIE	logical	Specifies that contours of constant effective intrinsic carrier concentration in number per cubic centimeter are plotted.	false	
NET.CHAR	logical	Specifies that contours of constant net charge concentration in number per cubic centimeter are plotted.	false	
NET.CARR	logical	Specifies that contours of constant net carrier concentration in number per cubic centimeter are plotted.	false	
J.CONDUC	logical	Specifies that contours of constant conduction current in amps per square centimeter are plotted.	false	
J.ELECTR	logical	Specifies that contours of constant electron current in amps per square centimeter are plotted.	false	
J.HOLE	logical	Specifies that contours of constant hole current in amps per square centimeter are plotted.	false	
J.DISPLA	logical	Specifies that contours of constant displacement current in amps per square centimeter are plotted.	false	
J.TOTAL	logical	Specifies that contours of constant total current in amps per square centimeter are plotted.	false	
FLOWLINE	logical	Specifies that current flow lines are plotted.	false	
RECOMBIN	logical	Specifies that contours of constant net recombination in number per cubic centimeter per second are plotted. For unequal electron and hole recombination, **RECOMBIN** is the same as **N.RECOMB**.	false	
N.RECOMB	logical	Specifies that contours of constant net electron recombination in number per cubic centimeter per second are plotted.	false	
P.RECOMB	logical	Specifies that contours of constant net hole recombination in number per cubic centimeter per second are plotted.	false	
II.GENER	logical	Specifies that contours of constant total generation rate due to impact ionization in pairs per cubic centimeter per second are plotted.	false	
BB.GENER	logical	Specifies that contours of constant band-to-band tunneling generation rate in pairs per cubic centimeter per second are plotted.	false	
PHOTOGEN	logical	Specifies that contours of constant total photogeneration in pairs per cubic centimeter per second are plotted.	false	
ELE.TEMP	logical	Specifies that contours of constant electron temperature in Kelvins are plotted.	false	
HOL.TEMP	logical	Specifies that contours of constant hole temperature in Kelvins are plotted.	false	

Parameter	Type	Definition	Default	Units
ELE.VEL	logical	Specifies that contours of constant electron mean velocity in cm/s are plotted.	false	
HOL.VEL	logical	Specifies that contours of constant hole mean velocity in cm/s are plotted.	false	
J.EFIELD	logical	Specifies that the contours where the component of the electric field in V/cm in the direction of the total current density vector is constant are plotted.	false	
G.GAMN	logical	Specifies that contours of constant probability that an electron will be injected into the oxide are plotted.	false	
G.GAMP	logical	Specifies that contours of constant probability that a hole will be injected into the oxide are plotted.	false	
G.GAMT	logical	Specifies that contours of constant probability that a hole or electron (that is, the sum of the hole and electron probabilities) will be injected into the oxide are plotted.	false	
G.IN	logical	Specifies that contours of constant hot electron injection current initiated from each point in amps/micron are plotted.	false	
G.IP	logical	Specifies that contours of constant hot hole injection current initiated from each point in amps/micron are plotted.	false	
G.IT	logical	Specifies that contours of constant total hot carrier injection current initiated from each point in amps/micron are plotted.	false	
ARRAY1	logical	Specifies that the user-generated array number 1 is to be plotted. Refer to the statement "EXTRACT" on page 3-166 for more information.	false	
ARRAY2	logical	Specifies that the user-generated array number 2 is to be plotted. Refer to the statement "EXTRACT" on page 3-166 for more information.	false	
ARRAY3	logical	Specifies that the user-generated array number 3 is to be plotted. Refer to the statement "EXTRACT" on page 3-166 for more information.	false	
TRAPS	logical	Specifies that constant trap density contours in number per cubic centimeter are to be plotted.	false	
TRAP.OCC	logical	Specifies that constant filled trap density contours in number per cubic centimeter are to be plotted.	false	
LEVEL	number	The specific trap level to plot. If this parameter is not specified, all trap levels are summed.	none	none
N.MOBILI	logical	Specifies that contours of constant electron mobility in cm^2/V-s are to be plotted.	false	
P.MOBILI	logical	Specifies that contours of constant hole mobility in cm^2/V-s are to be plotted.	false	
SIGMA	logical	Specifies that contours of constant conductivity in $(Ohm\text{-}cm)^{-1}$ are to be plotted.	false	
IMPURITY	char	The name of an impurity to plot contours of in number per cubic centimeter.	none	
OTHER	char	The name of an **OTHER** quantity to plot contours of.	none	

Lattice Temperature AAM Parameters

Parameter	Type	Definition	Default	Units
LAT.TEMP	logical	Specifies that contours of constant lattice temperature in Kelvins are plotted. This parameter is only used with the Lattice Temperature AAM.	false	

Heterojunction Device AAM Parameters

X.MOLE	logical	Specifies that contours of mole fraction are plotted. This parameter is only used with the Heterojunction Device AAM.	false	

AC Small-Signal Analysis Quantity Parameters

AC.REAL	logical	Specifies that contours of the real part of the quantity obtained from AC analysis are plotted.	false	
AC.IMAG	logical	Specifies that contours of the imaginary part of the quantity obtained from AC analysis are plotted	false	
AC.MAGN	logical	Specifies that contours of the magnitude of the quantity obtained from AC analysis are plotted.	false	
AC.PHAS	logical	Specifies that contours of the phase of the quantity obtained from AC analysis is plotted. Phase is defined as $\text{atan}(\text{imag}(X)/\text{real}(X))$, where X represents the quantity to be plotted.	false	

Contour Controls

MIN.VALU	number	The minimum contour value. If **LOGARITH** is specified, **MIN.VALU** should be a logarithmic value.	Minimum value of the specified quantity within the structure.	Dependent on plotted quantity
MAX.VALU	number	The maximum contour value. If **LOGARITH** is specified, **MAX.VALU** should be a logarithmic value.	Maximum value of the specified quantity within the structure.	Dependent on plotted quantity
WINDOW	logical	Specifies that the default values for **MIN.VALU** and **MAX.VALU** should be found from the portion of the structure that lies within the two-dimensional plot bounds.	false	
DEL.VALU	number	The difference between successive contour values.	(**MAX.VALU** - **MIN.VALU**)/(**NCONTOUR** - 1) for **NCONTOUR** > 1	Dependent on plotted quantity
NCONTOUR	number	The number of contours to plot.	1 + (**MAX.VALU** - **MIN.VALU**)/|**DEL.VALU**|, if **DEL.VALU** is specified; otherwise, 11	none
FILL	logical	Specifies that regions between constant contour values are color filled on devices that support this operation. The contour values defining the edges of the regions to be filled can be specified in the usual way using combinations of **MIN.VALU**, **MAX.VALU**, **DEL.VALU**, and **NCONTOUR**. If none of these parameters are specified, the entire plot area is filled, with the minimum to the maximum of the specified quantity being spanned by the available colors for the output device that is being used.	false	

Parameter	Type	Definition	Default	Units
C.START	number	The starting color index for the first region which is filled when the **FILL** parameter is specified.	8	none
C.INCREM	number	The color index increment for successive regions which are filled when the **FILL** parameter is specified. If **C.START** corresponds to a primary color, then **C.INCREM** causes the color index to cycle through the available primary colors. If **C.START** corresponds to a secondary color, then **C.INCREM** causes the color index to cycle through the available secondary colors. This parameter has no effect if a color graphics device is not used.	1	none
ABSOLUTE	logical	Specifies that contours of the absolute value of the specified quantity are plotted.	false	
LOGARITH	logical	Specifies that contours of the logarithm of the specified quantity are plotted. Since many of the quantities may become negative, the program actually uses sign(x) * log(1 + |x|) to avoid overflow. To get the true logarithm of a quantity, specify **ABSOLUTE** and **LOGARITH**. The absolute value is taken first and there is no danger of negative arguments.	false	
X.COMPON	logical	Specifies that if the selected quantity is a vector, then constant contours of its x component are plotted.	false	
Y.COMPON	logical	Specifies that if the selected quantity is a vector, then constant contours of its y component are plotted.	false	
LINE.TYP	number	The type of line used for the contour plot. A line type value of 1 generates a solid line plot. Line type values greater than 1 generate dashed line plots, with the dash size increasing with the value of line type.	1	none
COLOR	number	The index of the color used for the plot. The color associated with each color index is dependent upon the color graphics device that is used. This parameter has no effect if a color graphics device is not used.	1	none
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. The user must enter a space followed by a carriage return to continue execution.	false	

Description

The **CONTOUR** statement plots contours of various physical quantities on a two-dimensional area of the device as defined on the most recent **PLOT.2D** statement.

A **CONTOUR** statement must be preceded by a **PLOT.2D** statement in order to define the two-dimensional plot bounds.

See Also... To further illustrate the **CONTOUR** statement, refer to:

- *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Potential Contours and E-Line Plots" on page 4-32
- *mdex1d* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Drain Characteristics" on page 4-12
- Most other examples that have 2D graphical output

Minimum, Maximum, and Interval Values

The minimum and maximum contour values can be specified with the **MIN.VALU** and **MAX.VALU** parameters. The interval between contours can be specified with the **DEL.VALU** parameters.

The units of these parameters are determined by the specific quantity that is being plotted. If **LOGARITH** is specified:

- **MIN.VALU** and **MAX.VALU** are logarithmic values (base 10)
- **DEL.VALU** is a logarithmic interval between contours

The **NCONTOUR** parameter may also be specified to request a specific number of contours.

When a **CONTOUR** statement is encountered, the program attempts to plot contours corresponding to the following values:

```
MIN.VALU + (i-1)*DEL.VALU, i=1,NCONTOUR, if DEL.VALU > 0

MAX.VALU + (i-1)*DEL.VALU, i=1,NCONTOUR, if DEL.VALU < 0
```

Contour values which are actually plotted are printed to the standard output file.

If **MIN.VALU** and **MAX.VALU** are not specified:

- Default values are determined from the minimum and maximum values of the specified quantity over the entire device structure.

If **WINDOW** is specified:

- Default values for **MIN.VALU** and **MAX.VALU** are determined from the minimum and maximum values of the specified quantity over the portion of the device structure that is within the two-dimensional plot bounds.

Examples

The following statements plot contours of impurity concentration:

```
CONTOUR   DOPING  LOG  MIN=16   MAX=18   DEL=1  LINE=1
CONTOUR   DOPING  LOG  MIN=-18  MAX=-16  DEL=1  LINE=2
```

The first statement above plots n-type contours of concentration 1e16, 1e17, and 1e18/cm^3. The second statement above plots p-type contours of concentration 1e16, 1e17, and 1e18/cm^3.

The following statement plots contours of potential:

```
CONTOUR   POTENTIAL  NCONTOUR=11
```

Since neither **MIN.VALU** nor **MAX.VALU** are specified, the plotted contours span the entire range of potential values for the structure.

VECTOR

The **VECTOR** statement plots vector quantities over an area of the device defined by the previous **PLOT.2D** statement.

```
VECTOR

  {  J.CONDUC | J.ELECTR | J.HOLE | J.DISPLA | J.TOTAL | E.FIELD
   | AC.POTEN | AC.CN | AC.CP | AC.TN | AC.TP | AC.TL

 Optical Device AAM Parameters
   | ( RAYTRACE [INCIDENT] [INTERNAL] [EXITING] )
  }

 AC Small-Signal Analysis Vector Quantity Parameters
  [ {  AC.VECT | AC.XCOMP | AC.YCOMP | AC.REAL | AC.IMAG
     | AC.MAGN | AC.PHAS
    }
  ]

 Plot Control Parameters
  [LOGARITH [NORM.LOG=<n>] ] [V.SIZE=<n>] [CLIPFACT=<n>]
  [MINIMUM=<n>] [MAXIMUM=<n>] [LINE.TYP=<n>] [COLOR=<n>] [PAUSE]
```

Parameter	Type	Definition	Default	Units
J.CONDUC	logical	Specifies that a vector plot of conduction current is plotted.	false	
J.ELECTR	logical	Specifies that a vector plot of electron current is plotted.	false	
J.HOLE	logical	Specifies that a vector plot of hole current is plotted.	false	
J.DISPLA	logical	Specifies that a vector plot of displacement current is plotted.	false	
J.TOTAL	logical	Specifies that a vector plot of total current is plotted.	false	
E.FIELD	logical	Specifies that a vector plot of electric field is plotted.	false	
AC.POTEN	logical	Specifies that a complex vector plot of AC potential is plotted.	false	
AC.CN	logical	Specifies that a complex vector plot of AC electron concentration is plotted.	false	
AC.CP	logical	Specifies that a complex vector plot of AC hole concentration is plotted.	false	
AC.TN	logical	Specifies that a complex vector plot of AC electron temperature is plotted.	false	
AC.TP	logical	Specifies that a complex vector plot of AC hole temperature is plotted.	false	
AC.TL	logical	Specifies that a complex vector plot of AC lattice temperature is plotted.	false	

Parameter	Type	Definition	Default	Units
Optical Device AAM Parameters				
RAYTRACE	logical	Specifies that the optical rays generated by the OD-AAM are plotted.	false	
INCIDENT	logical	Specifies that incident optical rays generated by the OD-AAM are plotted.	true, if **RAYTRACE** is specified	
INTERNAL	logical	Specifies that internal optical rays generated by the OD-AAM are plotted.	true, if **RAYTRACE** is specified	
EXITING	logical	Specifies that exiting optical rays generated by the OD-AAM are plotted.	true, if **RAYTRACE** is specified	
AC Small-Signal Analysis Vector Quantity Parameters				
AC.VECT	logical	Specifies that a complex vector plot of the spatial magnitude of the vector quantity obtained from AC analysis is plotted.	false	
AC.XCOMP	logical	Specifies that a complex vector plot of the x component of the vector quantity obtained from AC analysis is plotted.	false	
AC.YCOMP	logical	Specifies that a complex vector plot of the y component of the vector quantity obtained from AC analysis is plotted.	false	
AC.REAL	logical	Specifies that a spatial vector plot of the real part of the vector quantity obtained from AC analysis is plotted.	false	
AC.IMAG	logical	Specifies that a spatial vector plot of the imaginary part of the vector quantity obtained from AC analysis is plotted.	false	
AC.MAGN	logical	Specifies that a spatial vector plot of the complex magnitude of the vector quantity obtained from AC analysis is plotted.	false	
AC.PHAS	logical	Specifies that a spatial vector plot of the phase of the vector quantity obtained from AC analysis is plotted.	false	
Plot Control Parameters				
LOGARITH	logical	Specifies that the vector magnitudes are logarithmically scaled. If this parameter is not specified, all vectors are scaled linearly.	false	
NORM.LOG	number	The number of orders of magnitude of the specified quantity that are plotted when vectors are plotted logarithmically.	all orders	none
V.SIZE	number	Specifies the size of the largest vector plotted. The vector with the largest magnitude in the display area has a length equal to **V.SIZE**. All other vectors are scaled appropriately.	1.0	cm
CLIPFACT	number	The threshold factor for plotting vectors. No vector whose length is smaller than the length of the largest vector multiplied by **CLIPFACT** is plotted.	0.01	none

Parameter	Type	Definition	Default	Units
MINIMUM	number	The minimum magnitude plotted.	The minimum magnitude of the specified quantity.	volts/cm for electric field, amps/cm^2 for currents
MAXIMUM	number	The maximum magnitude plotted.	The maximum magnitude of the specified quantity.	volts/cm for electric field, amps/cm^2 for currents
LINE.TYP	number	The type of line used to plot vectors. A line type value of 1 generates a solid line. **LINE.TYP** values greater than 1 generate dashed line vectors, with the dash size increasing with the value of **LINE.TYP**.	1	none
COLOR	number	The index of the color used to plot vectors. The color associated with each color index is dependent upon the color graphics device that is used. This parameter has no effect if a color graphics device is not used.	1	none
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. You must enter a space followed by a carriage return to continue execution.	false	

Description

The **VECTOR** statement is used to plot one of various vector quantities over the device cross-section specified on the **PLOT.2D** statement. One vector is plotted at each node of the simulation grid. The vectors are oriented to point in the direction of the specified vector quantity. When a complex vector from AC analysis is plotted, the *x* direction represents the real part of the vector while the *y* direction represents its imaginary part. By default, vectors are plotted linearly.

See Also...

To further illustrate the **VECTOR** statement, refer to input files:

- *mdex2fp* in NPN Bipolar Transistor Examples, Chapter 5, "Post-Processing of Forward Bias Results" on page 5-6
- *mdex2pp* in NPN Bipolar Transistor Examples, Chapter 5, "Post-Processing of Device with Modified Emitter" on page 5-14

Vector Scaling

For linearly plotted vector involving current, the length of each vector is proportional to the current density at the node. For electric field, the length of each vector is proportional to the magnitude of the electric field at the node.

The following parameters affect the magnitude of plotted vectors.

- **LOGARITH** specifies that the magnitude of the specified quantity at each node is first normalized by the minimum magnitude. **NORM.LOG** specifies that the magnitude of the specified quantity at each node is first normalized by 10**(-**NORM.LOG**) times the maximum magnitude.

The length of each plotted vector is then proportional to the logarithm of these normalized values.

- **NORM.LOG** may be used to limit the number of orders of magnitude of the specified quantity that is plotted.
- **MINIMUM** and **MAXIMUM** may used to arbitrarily set the minimum and maximum magnitudes.

  This makes it possible to plot two bias conditions or devices with the same scaling. Both parameters are printed during the execution of a plot.

Vector Size

V.SIZE may be used to increase or decrease the size of the plotted vectors. **CLIPFACT** may be used to prevent very small vectors from being plotted. The smallest plotted vector has a length equal to the length of the largest vector multiplied by **CLIPFACT**.

AC Vectors

The **VECTOR** statement can also be used to plot AC quantities as vectors at each node of the device structure. Complex vector plots (AC real component in the horizontal direction and AC imaginary component in the vertical direction) of the scalar quantities potential, electron concentration, hole concentration, electron temperature, hole temperature and lattice temperature can be plotted by specifying **AC.POTEN**, **AC.CN**, **AC.CP**, **AC.TN**, **AC.TP**, and **AC.TL**, respectively.

For the current densities available on the **VECTOR** statement (**J.CONDUC**, **J.ELECTR**, **J.HOLE**, **J.DISPLA**, and **J.HOLE**), the choice of plotting either complex vector plots or spatial vector plots is available. If $\vec{J}_{AC}$ represents one of the complex vector current densities, it can be written as

Equation 3-22

$$\vec{J}_{AC}(x,y) = [J_{R,x}(x,y) + i\, J_{I,x}(x,y)]\hat{x} + [J_{R,y}(x,y) + i\, J_{I,y}(x,y)]\hat{y}$$

The choices on the **VECTOR** statement for plotting this are given in Table 3-2.

Table 3-2

Parameter	Horizontal Component of Vector	Vertical Component of Vector
AC.VECT	$\sqrt{J_{R,x}^2 + J_{R,y}^2}$	$\sqrt{J_{I,x}^2 + J_{I,y}^2}$
AC.XCOMP	$J_{R,x}$	$J_{I,x}$
AC.YCOMP	$J_{R,y}$	$J_{I,y}$
AC.REAL	$J_{R,x}$	$J_{R,y}$
AC.IMAG	$J_{I,x}$	$J_{I,y}$
AC.MAGN	$\sqrt{J_{R,x}^2 + J_{I,x}^2}$	$\sqrt{J_{R,y}^2 + J_{I,y}^2}$
AC.PHAS	$\mathrm{atan}(J_{I,x}/J_{R,x})$	$\mathrm{atan}(J_{I,y}/J_{R,y})$

FILL

The **FILL** statement causes all material regions to be filled with the specified colors. The **FILL** statement is also used to associate colors with material regions for subsequent **FILL** statements. The colors established by the **FILL** statement are also used when the **FILL** parameter is encountered on the **PLOT.2D** statement.

```
FILL

   [REGION=<c>] [SET.COLO] [N-TYPE] [P-TYPE] [PAUSE]
   {  [COLOR=<n>]
    | ( [C.SILIC=<n>] [C.GAAS=<n>] [C.POLYSI=<n>] [C.GERMAN=<n>]
        [C.SIC=<n>] [C.SEMI=<n>] [C.SIGE=<n>] [C.ALGAAS=<n>]
        [C.A-SILI=<n>] [C.DIAMON=<n>] [C.HGCDTE=<n>] [C.INAS=<n>]
        [C.INGAAS=<n>] [C.INP=<n>] [C.S.OXID=<n>] [C.ZNSE=<n>]
        [C.ZNTE=<n>]
        [C.OXIDE=<n>] [C.NITRID=<n>] [C.SAPPHI=<n>] [C.OXYNIT=<n>]
        [C.INSUL=<n>] [C.ELECTR=<n>]
        [NP.COLOR] [C.NTYPE=<n>] [C.PTYPE=<n>]
      )
   }
```

Parameter	Type	Definition	Default	Units
REGION	char	Color fills only the named region.	All regions are filled with the appropriate color.	
SET.COLO	logical	Specifies that the colors associated with the various materials are set to the values indicated by any **C.**<*material*> parameter that is specified or to the values that are established in the key file. If this parameter is specified, no color fill takes place.	false	
N-TYPE	logical	Specifies that the n-type portion of the specified region is color filled.	True except when **P-TYPE** alone is specified.	
P-TYPE	logical	Specifies that the p-type portion of the specified region is color filled.	True except when **N-TYPE** alone is specified.	
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. The user must enter a space followed by a carriage return to continue execution.	false	
COLOR	number	The color index for the color used to fill the specified regions.	none	none
C.SILIC	number	The color index for the color used to fill **SILICON** regions.	-1	none
C.GAAS	number	The color index for the color used to fill **GAAS** regions.	-1	none
C.POLYSI	number	The color index for the color used to fill **POLYSILI** regions.	2	none
C.GERMAN	number	The color index for the color used to fill **GERMANIU** regions.	-1	none

Parameter	Type	Definition	Default	Units
C.SIC	number	The color index for the color used to fill **SIC** regions.	-1	none
C.SEMI	number	The color index for the color used to fill **SEMICOND** regions.	-1	none
C.SIGE	number	The color index for the color used to fill **SIGE** regions.	-1	none
C.ALGAAS	number	The color index for the color used to fill **ALGAAS** regions.	-1	none
C.A-SILI	number	The color index for the color used to fill **A-SILICO** regions.	-1	none
C.DIAMON	number	The color index for the color used to fill **DIAMOND** regions.	-1	none
C.HGCDTE	number	The color index for the color used to fill **HGCDTE** regions.	-1	none
C.INAS	number	The color index for the color used to fill **INAS** regions.	-1	none
C.INGAAS	number	The color index for the color used to fill **INGAAS** regions.	-1	none
C.INP	number	The color index for the color used to fill **INP** regions.	-1	none
C.S.OXID	number	The color index for the color used to fill **S.OXIDE** regions.	-1	none
C.ZNSE	number	The color index for the color used to fill **ZNSE** regions.	-1	none
C.ZNTE	number	The color index for the color used to fill **ZNTE** regions.	-1	none
C.OXIDE	number	The color index for the color used to fill **OXIDE** regions.	5	none
C.NITRID	number	The color index for the color used to fill **NITRIDE** regions.	6	none
C.SAPPHI	number	The color index for the color used to fill **SAPPHIRE** regions.	6	none
C.OXYNIT	number	The color index for the color used to fill **OXYNITRI** regions.	7	none
C.INSUL	number	The color index for the color used to fill **INSULATO** regions.	5	none
C.ELECTR	number	The index of the color used to fill finite thickness electrodes.	4	none
NP.COLOR	logical	Color fills n- and p-type regions using the colors **C.NTYPE** and **C.PTYPE**. If **^NP.COLOR** is specified, n- and p-type regions are color-filled using the color corresponding to their material type (**C.SILIC**, **C.GAAS**, **C.SEMI**, or **C.POLYSI**).	true	
C.NTYPE	number	The index of the color used to fill n-type regions.	3	none
C.PTYPE	number	The index of the color used to fill p-type regions.	7	none

Description

The **FILL** statement causes all material regions to be filled with the specified colors. This statement provides a means of allowing material regions to have interesting and informative colors.

See Also...

To further illustrate the **FILL** statement, refer to input files:

- *mdex16* in Heterojunction Examples, Chapter 14, "SiGe Heterojunction Bipolar Transistor Simulation" on page 14-1

- *mdex17* in Heterojunction Examples, Chapter 14, "High Electron Mobility Transistor Simulation" on page 14-7
- *mdex18c* in Trapped Charge Examples, Chapter 15, "Breakdown Walk-Out in Power MOS Device" on page 15-10

Properties and Parameters

The **FILL** statement has the following properties and parameters:

- By default, the **FILL** statement fills all material regions with their corresponding colors.
- A particular region can be filled with a particular color by using the **REGION** and **COLOR** parameters.
- All specified values for **C.**<*material*> remain in effect for subsequent occurrences of the **FILL** statement.
- Colors specified by **FILL** are used to fill material regions when the **PLOT.2D** statement is encountered.
- A negative value for one of the material colors disables color fill for that material.

E.LINE

The **E.LINE** statement locates potential gradient paths and calculates the ionization integrals for electron and hole initiated multiplication along these paths. This statement also either plots the paths as part of a plot associated with the **PLOT.2D** statement or extracts and plots various physical quantities along the paths as part of a plot associated with the **PLOT.1D** statement.

```
E.LINE

  {  ( [POTENTIA] [QFN] [QFP] [VALENC.B] [CONDUC.B] [VACUUM]
       [ARRAY1] [ARRAY2] [ARRAY3]
     )
   | [ E.FIELD [ANGLE=<n>] ]
   | ( [DOPING] [ELECTRON] [HOLES] [NET.CHAR] [NET.CARR] )
   | ( [J.CONDUC] [J.ELECTR] [J.HOLE] [J.DISPLA] [J.TOTAL]
       [ANGLE=<n>]
     )
   | [RECOMBIN] | [II.GENER] | [BB.GENER] | [PHOTOGEN]
   | [ELE.TEMP] | [HOL.TEMP] | [ELE.VEL] | [HOL.VEL] | [J.EFIELD]
   | [G.GAMN] | [G.GAMP] | [G.GAMT] | [G.IN] | [G.IP] | [G.IT]

 Lattice Temperature AAM Parameters
    | [LAT.TEMP]

 Heterojunction Device AAM Parameters
    | [X.MOLE]
  }

 Control Parameters
   X.START=<n> Y.START=<n> [S.DELTA=<n>] [N.LINES=<n>] [HORZ.STA=<n>]
   [I.ELECTR] [I.HOLES] [E.MARK=<n>] [M.SIZE=<n>] [INSULATO]
   [LINE.TYP=<n>] [COLOR=<n>] [FILE=<c>] [SUMMARY] [PLOT] [PAUSE]
```

Parameter	Type	Definition	Default	Units
POTENTIA	logical	Specifies that the mid-gap potential in volts is to be plotted.	false	
QFN	logical	Specifies that the electron quasi-Fermi potential in volts is to be plotted.	false	
QFP	logical	Specifies that the hole quasi-Fermi potential in volts is to be plotted.	false	
VALENC.B	logical	Specifies that the valence band potential in volts is to be plotted.	false	
CONDUC.B	logical	Specifies that the conduction band potential in volts is to be plotted.	false	
VACUUM	logical	Specifies that the vacuum potential in volts is to be plotted.	false	
ARRAY1	logical	Specifies that the user generated ARRAY # 1 is to be plotted. See the **EXTRACT** statement for more information.	false	

Parameter	Type	Definition	Default	Units
ARRAY2	logical	Specifies that the user generated ARRAY # 2 is to be plotted. See the **EXTRACT** statement for more information.	false	
ARRAY3	logical	Specifies that the user generated ARRAY # 3 is to be plotted. See the **EXTRACT** statement for more information.	false	
E.FIELD	logical	Specifies that the electric field magnitude in volts per centimeter be plotted. If the **ANGLE** parameter is specified, the electric field component along the specified angular direction is plotted. Otherwise, the electric field magnitude is plotted.	True, if **E.FIELD** specified in **PLOT.1D**; otherwise, false	
ANGLE	number	The angle relative to the horizontal of the electric field or current density component being plotted.	none	degree
DOPING	logical	Specifies that the net impurity concentration in number per cubic centimeter is to be plotted.	false	
ELECTRON	logical	Specifies that the electron concentration in number per cubic centimeter is to be plotted.	false	
HOLES	logical	Specifies that the hole concentration in number per cubic centimeter is to be plotted.	false	
NET.CHAR	logical	Specifies that the net charge concentration in number per cubic centimeter is to be plotted.	false	
NET.CARR	logical	Specifies that the net carrier concentration in number per cubic centimeter is to be plotted.	false	
J.CONDUC	logical	Specifies that the conduction current density in amps per square centimeter is to be plotted. If the **ANGLE** parameter is specified, the current density component along the specified angular direction is plotted. Otherwise, the current density magnitude is plotted.	false	
J.ELECTR	logical	Specifies that the electron current density in amps per square centimeter is to be plotted. If the **ANGLE** parameter is specified, the current density component along the specified angular direction is plotted. Otherwise, the current density magnitude is plotted.	false	
J.HOLE	logical	Specifies that the hole current density in amps per square centimeter is to be plotted. If the **ANGLE** parameter is specified, the current density component along the specified angular direction is plotted. Otherwise, the current density magnitude is plotted.	false	
J.DISPLA	logical	Specifies that the displacement current density in amps per square centimeter is to be plotted. If the **ANGLE** parameter is specified, the current density component along the specified angular direction is plotted. Otherwise, the current density magnitude is plotted.	false	
J.TOTAL	logical	Specifies that the total current density in amps per square centimeter is to be plotted. If the **ANGLE** parameter is specified, the current density component along the specified angular direction is plotted. Otherwise, the current density magnitude is plotted.	false	
RECOMBIN	logical	Specifies that the net recombination in number per cubic centimeter per second is to be plotted.	True, if **RECOMBIN** specified in **PLOT.1D**; otherwise, false.	

Parameter	Type	Definition	Default	Units
II.GENER	logical	Specifies that the total generation rate due to impact ionization in pairs per cubic centimeter per second is to be plotted.	True, if **II.GENER** specified in **PLOT.1D**; otherwise, false.	
BB.GENER	logical	Specifies that the generation rate due to band-to-band tunneling in pairs per cubic centimeter per second is to be plotted.	True, if **BB.GENER** specified in **PLOT.1D**; otherwise, false.	
PHOTOGEN	logical	Specifies that total photogeneration in pairs per cubic centimeter per second is to be plotted.	True, if **PHOTOGEN** specified in **PLOT.1D**; otherwise, false.	
ELE.TEMP	logical	Specifies that the electron temperature in Kelvins is to be plotted.	True, if **ELE.TEMP** specified in **PLOT.1D**; otherwise, false.	
HOL.TEMP	logical	Specifies that the hole temperature in Kelvins is to be plotted.	True, if **HOL.TEMP** specified in **PLOT.1D**; otherwise, false.	
ELE.VEL	logical	Specifies that the electron mean velocity in cm/s be plotted.	True, if **ELE.VEL** specified in **PLOT.1D**; otherwise, false.	
HOL.VEL	logical	Specifies that the hole mean velocity in cm/s is to be plotted.	True, if **HOL.VEL** specified in **PLOT.1D**; otherwise, false.	
J.EFIELD	logical	Specifies that the component of the electric field in the direction of the total current density in V/cm is to be plotted.	True, if **J.EFIELD** specified in **PLOT.1D**; otherwise, false.	
G.GAMN	logical	Specifies that the probability per unit length that an electron will be injected into the gate insulator in number per centimeter is to be plotted.	True, if **G.GAMN** specified in **PLOT.1D**; otherwise, false.	
G.GAMP	logical	Specifies that the probability per unit length that a hole will be injected into the gate insulator in number per centimeter is to be plotted.	True, if **G.GAMP** specified in **PLOT.1D**; otherwise, false.	
G.GAMT	logical	Specifies that the probability per unit length that an electron or hole will be injected into the gate insulator in number per centimeter is to be plotted.	True, if **G.GAMT** specified in **PLOT.1D**; otherwise, false.	
G.IN	logical	Specifies that the hot electron injection current in amps/micron is to be plotted.	True, if **G.IN** specified in **PLOT.1D**; otherwise, false.	
G.IP	logical	Specifies that the hot hole injection current in amps/micron is to be plotted.	True, if **G.IP** specified in **PLOT.1D**; otherwise, false.	
G.IT	logical	Specifies that the total hot carrier injection current in amps/micron is to be plotted.	True, if **G.IT** specified in **PLOT.1D**; otherwise, false.	

Parameter	Type	Definition	Default	Units
Lattice Temperature AAM Parameters				
LAT.TEMP	logical	Specifies that the lattice temperature in Kelvins is to be plotted. This parameter is only used with the Lattice Temperature AAM.	True if **LAT.TEMP** specified in **PLOT.1D**; otherwise, false.	
Heterojunction Device AAM Parameters				
X.MOLE	logical	Specifies that the mole fraction is to be plotted. This parameter is only used with the Heterojunction Device AAM.	True if **X.MOLE** specified in **PLOT.1D**; otherwise, false.	
Control Parameters				
X.START	number	The horizontal coordinate in the device coordinate system of the starting point for the first potential gradient path.	none	microns
Y.START	number	The vertical coordinate in the device coordinate system of the starting point for the first potential gradient path.	none	microns
S.DELTA	number	The distance increment between the starting points of successive potential gradient paths. A positive (negative) value for this parameter causes successive starting points to be located to the right (left) of the last potential gradient path, where the forward direction is in the direction of decreasing potential along the path.	none	microns
N.LINES	number	The number of potential gradient paths.	1	none
HORZ.STA	number	The value along the horizontal plot axis associated with the starting point of the path. This value establishes the reference for horizontal distance along the path.	0.0	microns
I.ELECTR	logical	Specifies that the potential gradient path in the direction against the gradient direction terminates when the path intersects an insulator-semiconductor interface. If this parameter is false, the path continues parallel to the interface after an intersection occurs, potentially increasing the electron ionization integral.	false	
I.HOLES	logical	Specifies that the potential gradient path in the direction along the gradient direction terminates when the path intersects an insulator-semiconductor interface. If this parameter is false, the path continues parallel to the interface after an intersection occurs, potentially increasing the hole ionization integral.	false	
E.MARK	number	The electric field magnitude defining the locations of marks placed along the plots of the potential gradient paths or the quantities extracted along those paths. The marks are placed at locations along the paths where the magnitude of the electric field equals **E.MARK**. The marks point in the direction of increasing magnitude of electric field.	none	volts/ cm
M.SIZE	number	The height of the marks located with the **E.MARK** parameter.	0.25	cm
INSULATO	logical	Specifies that the electric field lines associated with a two-dimensional plot be plotted in insulating materials as well as semiconductor materials. If this parameter is true, ionization integrals along the electric field lines are not calculated.	false	

Parameter	Type	Definition	Default	Units
LINE.TYP	number	The type of line used for the plot. A line type value of 1 generates a solid line plot. Line type values greater than 1 generate dashed line plots, with the dash size increasing with the value of line type.	1	none
COLOR	number	The index of the color used for the plot. The color associated with each color index is dependent upon the color graphics device that is used. This parameter has no effect if a color graphics device is not used.	1	none
FILE	char	The identifier for the formatted output data file in which the distance and extracted quantities at every point along the path are to be stored.	none	
SUMMARY	logical	Specifies that the ionization integral information for each potential gradient path is to be printed on the standard output device.	true	
PLOT	logical	Specifies that the potential gradient paths or extracted quantities are to be plotted.	true	
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. The user must enter a space followed by a carriage return to continue execution.	false	

Description

The **E.LINE** statement locates potential gradient paths and calculates the ionization integrals for electron and hole initiated multiplication along these paths. This statement either:

- Plots the paths as part of a plot associated with the **PLOT.2D** statement

or

- Extracts and plots various physical quantities along the paths as part of a plot associated with the **PLOT.1D** statement

See Also... To further illustrate the **E.LINE** statement, refer to input file *mdex1a* in N-Channel MOSFET Examples, Chapter 4, "Potential Contours and E-Line Plots" on page 4-32.

Identifying Output and Input

The printed output associated with the **E.LINE** statement is labeled with two lines of information identifying the type of output and the input line number responsible for the output.

The **E.LINE** statement may either appear alone or within plot sequences initiated by the **PLOT.2D** and **PLOT.1D** statements. Within a two-dimensional plot sequence, the locations of the potential gradient paths are plotted. Within a one-

dimensional plot sequence, the requested physical quantities are plotted along the potential gradient paths.

First Potential Gradient Path

The location of the starting point for the first potential gradient path must be specified with the **X.START** and **Y.START** parameters. This point must lie within the semiconductor region of the device structure. The default starting point of the potential gradient path is associated with the value of zero on the horizontal axis when extracted quantities are being plotted.

The extracted data can be shifted on the horizontal axis by specification of the **HORZ.STA** parameter. **HORZ.STA** allows for arbitrary shifting of data along the horizontal axis, facilitating the comparison of profiles along different lines.

Previously Defined Solutions

The use of the **E.LINE** statement requires that solutions be previously defined. This may be accomplished with a solution initiated by a **SOLVE** statement or by using a **LOAD** statement to input a data file generated previously by a **SOLVE** statement.

Line Type

The type of line used during the plotting of the data may be changed from the default solid line to a variety of dotted and dashed lines. This allows easier distinction between various plots.

LABEL

The **LABEL** statement plots character strings, centered symbols, and lines as part of a plot associated with the **PLOT.1D** and **PLOT.2D** statements.

LABEL

```
[LABEL=<c>] [SYMBOL=<n>] [X=<n>] [Y=<n>] [ANGLE=<n>]
[ {START.LE | START.CE | START.RI} ] [ARROW]
[LX.START=<n>] [LY.START=<n>]
[LX.FINIS=<n>] [LY.FINIS=<n>]
[CM] [C.SIZE=<n>] [LINE.TYP=<n>] [COLOR=<n>] [PAUSE]
```

Parameter	Type	Definition	Default	Units
LABEL	char	The character string to be used to label the plot.	none	
SYMBOL	number	The type of centered symbol plotted at the data points in the plot. The value of this parameter may lie in the range 1 to 15. If this parameter is not specified, the plot will not contain centered symbols. Values of this parameter are associated with the following symbols: 1 Square 2 Circle 3 Triangle 4 Plus 5 Upper case X 6 Diamond 7 Up-arrow 8 Roofed upper case X 9 Upper case Z 10 Upper case Y 11 Curved square 12 Asterisk 13 Hourglass 14 Bar 15 Star	none	none
X	number	The x location associated with the lower left corner of the first character in the character string or the center of the centered symbol. If the **CM** parameter is specified, then this parameter specifies the location in cm relative to the left edge of the graphics display area. Otherwise, this parameter specifies the location in axis units along the x-axis.	The left side of the plot for the first **LABEL** statement after a plot; otherwise determined by the previous **LABEL** statement.	cm or x axis units
Y	number	The y location associated with the lower left corner of the first character in the character string or the center of the centered symbol. If the **CM** parameter is specified, then this parameter specifies the location in cm relative to the bottom edge of the graphics display area. Otherwise, this parameter specifies the location in axis units along the y-axis.	The top side of the plot for the first **LABEL** statement after a plot; otherwise determined by the previous **LABEL** statement.	cm or y-axis units
ANGLE	number	The angle (increasing counterclockwise) relative to the horizontal of the bottom of the character string.	0.0	degrees

Parameter	Type	Definition	Default	Units
START.LE	logical	Specifies the line is to start at the left side of the character string or centered symbol.	false	
START.CE	logical	Specifies the line is to start at the center of the character string or centered symbol.	false	
START.RI	logical	Specifies the line is to start at the right side of the character string or centered symbol.	false	
ARROW	logical	Specifies that an arrowhead is to be plotted at the end of the line.	false	
LX.START	number	The starting x coordinate of the line. If the **CM** parameter is specified, then this parameter specifies the location in cm relative to the left edge of the graphics display area. Otherwise, this parameter specifies the location in axis units along the x-axis.	The left side of the plot for the first **LABEL** statement after a plot statement; otherwise the previous value of **LX.FINIS**.	cm or x-axis units
LY.START	number	The starting y coordinate of the line. If the **CM** parameter is specified, then this parameter specifies the location in cm relative to the bottom edge of the graphics display area. Otherwise, this parameter specifies the location in axis units along the y-axis.	The top side of the plot for the first **LABEL** statement after a plot statement; otherwise the previous value of **LY.FINIS**.	cm or y-axis units
LX.FINIS	number	The ending x coordinate of the line. If the **CM** parameter is specified, then this parameter specifies the location in cm relative to the left edge of the graphics display area. Otherwise, this parameter specifies the location in axis units along the x-axis.	The horizontal starting point of the line.	cm or x-axis units
LY.FINIS	number	The ending y coordinate of the line. If the **CM** parameter is specified, then this parameter specifies the location in cm relative to the bottom edge of the graphics display area. Otherwise, this parameter specifies the location in axis units along the y-axis.	The vertical starting point of the line.	cm or y-axis units
CM	logical	Specifies that the **X**, **Y**, **LX.START**, **LY.START**, **LX.FINIS**, and **LY.FINIS** parameters specify locations in cm relative to the left and bottom edges of the graphics display area.	false	
C.SIZE	number	The height of the characters in the character string or centered symbol.	0.25	cm
LINE.TYP	number	The type of line used to plot the line. A line type value of 1 generates a solid line plot. Line type values greater than 1 generate dashed line plots, with the dash size increasing with the value of the line type.	1	none
COLOR	number	The index of the color used to plot the character string, centered symbol, and line. The color associated with each color index is dependent upon the color graphics device that is used. This parameter has no effect if a color graphics device is not used.	1	none
PAUSE	logical	Specifies that program execution pauses after the completion of all graphical output associated with this statement. The user must enter a space followed by a carriage return to continue execution.	false	

Description

The **LABEL** statement plots character strings, centered symbols, lines, and arrows as part of a plot associated with the **PLOT.1D** and **PLOT.2D** statements.

See Also...

To further illustrate the **LABEL** statement, refer to:

- Input file *mdex1g* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Gate Characteristics" on page 4-11
- Input file *mdex1d* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Drain Characteristics" on page 4-12
- Most other examples

Example

LABEL statements may appear at any point in the input file after the first **PLOT.1D** or **PLOT.2D** statement. For example, the following statement plots a label:

```
LABEL  LABEL="This is a label"
```

Parameters

The **LABEL** statement uses a variety of parameters to plot an assortment of functions.

Note:

The default settings for many of the* LABEL *parameters are dependent on the previous use of other statements and parameters.

Sizing

The sizes of characters and centered symbols plotted by the **LABEL** statement are specified by the **C.SIZE** parameter. Characters have a height of **C.SIZE** and a width of 0.5713***C.SIZE**.

The spacing between the left sides of two successive characters in a label is **C.SIZE**. The length of a label containing n characters is (n-0.4286)***C.SIZE**. Centered symbols have a height of **C.SIZE** and a width of **C.SIZE**.

Location

The **X** and **Y** parameters specify the location of the lower left corner of the first character in the character string or the center of the centered symbol. Default values are used for the **X** and **Y** parameters if they are not specified.

Default Selection

The default for **LABEL** statement is dependent on whether **LABEL** statements have appeared since the last **PLOT.1D** or **PLOT.2D** statement.

No Previous LABEL Statement

If no previous **LABEL** statements have appeared since the last **PLOT.1D** or **PLOT.2D** statement, the default values for **X** and **Y** are selected as follows:

1. A default value is used for **X** which places the start of the character string or centered symbol at the left side of the current plot.
2. A default value is used for **Y** which places the start of the character string or centered symbol at the top side of the current plot. For example, the following statements plot axes with a label in the top left corner of the plot:

```
PLOT.2D  BOUND
LABEL    LABEL="This label appears in the top left corner"
```

Previous LABEL Statement

If previous **LABEL** statements have appeared since the last **PLOT.1D** or **PLOT.2D** statement, the default values for **X** and **Y** are selected based on the following conditions:

1. If neither **X** nor **Y** is specified, default values are used for **X** and **Y** which place the character string or centered symbol below the previously plotted string or symbol.
2. If only **X** is specified, a default value is used for **Y** which places the start of the character string or centered symbol at the same vertical location as the previously plotted string or symbol.
3. If only **Y** is specified, a default value is used for **X** which places the start of the character string or centered symbol at the same horizontal location as the previously plotted string or symbol. For example, the following statements plot a label and a centered square symbol:

```
LABEL  LABEL="A centered square is under this label"
+      X=.1  Y=1E20
LABEL  SYMBOL=1
```

The start of the label is placed at coordinates (0.1,1E20) on the plot. The centered square is plotted under the first character in the label.

Line and Arrowhead

A line is plotted if either the **LX.FINIS** or **LY.FINIS** parameters is specified. An arrowhead is plotted at the end of the line if the **ARROW** parameter is specified. For example, the following statement plots a line between the coordinates (0.1,1E19) and (0.2,1E20) with an arrowhead at the end of the line:

```
LABEL  LX.START=.1  LY.START=1E19
+      LX.FINIS=.2  LY.FINIS=1E20  ARROW
```

Starting Coordinates

Starting coordinates are determined by the **LX.START**, **LY.START**, **START.LE**, **START.CE**, and **START.RI** parameters.

Default Selection

If either the **LX.START** or **LY.START** parameters is not specified, default starting coordinates are selected based on the following conditions:

1. If **START.LE**, **START.CE**, or **START.RI** is specified, default starting coordinates are selected which place the starting point of the line at the specified edge or center of the character string or centered symbol.

   For example, the following statement plots a label with a line from the right edge of the label to the coordinates (0.1,1E20):

   ```
   LABEL  LABEL="The line starts at the right"  START.RI
   +      LX.FINIS=.1  LY.FINIS=1E20
   ```

2. If neither **START.LE**, **START.CE**, nor **START.RI** is specified and a line has been plotted previously since the last **PLOT.1D** or **PLOT.2D** statement, default starting coordinates are selected which place the starting point of the line at the end of the previous line.

   For example, the following statements plot a line between the coordinates (0.1,1E19), (0.2,1E20), and (0.3,1E21):

   ```
   LABEL  LX.START=.1  LY.START=1E19
   +      LX.FINIS=.2  LY.FINIS=1E20
   LABEL  LX.FINIS=.3  LY.FINIS=1E21
   ```

   The coordinates of the final point of the first line are used as the coordinates for the starting point of the second line.

3. If neither **START.LE**, **START.CE**, nor **START.RI** is specified and a line has not been plotted previously since the last **PLOT.1D** or **PLOT.2D** statement, default starting coordinates are selected which place the starting point of the line at the top left corner of the plot.

LOG

The **LOG** statement allows the I-V and/or AC terminal data for a simulation to be written to a file. This statement also provides for the conversion of Medici log files to data files that can be used by common parameter extraction programs.

```
LOG

   {  ( [OUT.FILE=<c> [TIF] ] [CLOSE] )
    | ( [ {  ( AURORA [LENGTH=<n>] [WIDTH=<n>] [DEVID=<n>] [REGION=<n>]
                      [P.CHANNE]
             )
           | ( ICCAP [MDM] [I.MIN=<n>] [V.MIN=<n>]
               [INP1=<c> [LIST1] ] [INP2=<c> [LIST2] ]
               [INP3=<c> [LIST3] ] [INP4=<c> [LIST4] ]
               [INP5=<c> [LIST5] ]
               {  ( [OUT1=<c>] [OUT2=<c>] [OUT3=<c>] [OUT4=<c>] )
                | ( [S.PARAM TERMINAL=<c>] )
               }
             )
            | STANFORD
          }
          IN.FILE=<c> OUT.FILE=<c>
          [GATE=<c>] [SOURCE=<c>] [DRAIN=<c>] [SUBSTRAT=<c>]
          [BASE=<c>] [EMITTER=<c>] [COLLECT=<c>] [EXTRA=<c>]
      ]
    )
   }
```

Parameter	Type	Definition	Default	Units
OUT.FILE	char	The identifier for the file to store data obtained at each bias or time point. In the case where Medici data is being converted to a new format, this parameter specifies the name of the new file. **synonym: IVFILE**	none	
TIF	logical	Specifies that the log file is to be written in the TIF format.	true	
CLOSE	logical	Specifies that the presently open log file is to be closed so that no additional data is written to the file.	false	
AURORA	logical	Specifies that a data file for *Avant!* TCAD's Aurora program is created. **synonym: TOPEX**	none	
LENGTH	number	The device channel length to be written to output files created for Aurora.	none	microns
WIDTH	number	The device channel width to be written to output files created for Aurora. This value also multiplies the values of current stored in the input I-V file before they are written to the output file.	none	microns
DEVID	number	An optional device identification parameter written to output files created for Aurora.	none	none
REGION	number	An optional device behavior region parameter written to output files created for Aurora.	none	none

Parameter	Type	Definition	Default	Units
P.CHANNE	logical	Specifies that the sign of all voltages and currents are changed before writing the data to the Aurora output file.	false	
ICCAP	logical	Specifies that a data file for Hewlett-Packard's IC-CAP program is created.	none	
MDM	logical	Specifies that the data file created for Hewlett-Packard's IC-CAP program is written using the measured data management file format. The file extension for these data files should be ".mdm". This is the recommended format.	false	
I.MIN	number	The minimum recognizable current value when examining currents in the Medici log file. Currents with absolute values smaller than this are considered to be zero.	1.e-17	A/micron
V.MIN	number	The minimum recognizable voltage value when examining voltages in the Medici log file. Voltages with absolute values smaller than this are considered to be zero.	1.e-5	volts
INP1	char	The first input variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, terminal current *I(name)*, and frequency *freq*.	none	
LIST1	logical	Specifies that the sweep type of the first IC-CAP input variable is LIST. If this parameter is not specified, the sweep type is LIN. This parameter is valid only when **MDM** is specified.	false	
INP2	char	The second input variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, terminal current *I(name)*, and frequency *freq*.	none	
LIST2	logical	Specifies that the sweep type of the second IC-CAP input variable is LIST. If this parameter is not specified, the sweep type is LIN. This parameter is valid only when **MDM** is specified.	false	
INP3	char	The third input variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, terminal current *I(name)*, and frequency *freq*.	none	
LIST3	logical	Specifies that the sweep type of the third IC-CAP input variable is LIST. If this parameter is not specified, the sweep type is LIN. This parameter is valid only when **MDM** is specified.	false	
INP4	char	The fourth input variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, terminal current *I(name)*, and frequency *freq*.	none	
LIST4	logical	Specifies that the sweep type of the forth IC-CAP input variable is LIST. If this parameter is not specified, the sweep type is LIN. This parameter is valid only when **MDM** is specified.	false	
INP5	char	The fifth input variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, terminal current *I(name)*, and frequency *freq*.	none	
LIST5	logical	Specifies that the sweep type of the fifth IC-CAP input variable is LIST. If this parameter is not specified, the sweep type is LIN. This parameter is valid only when **MDM** is specified.	false	
OUT1	char	The first output variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, and terminal current *I(name)*. If **MDM** is specified, the choices also include capacitance *C(name1,name2)*, conductance *G(name1,name2)*, and admittance *Y(name1,name2)*.	none	

Parameter	Type	Definition	Default	Units
OUT2	char	The second output variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, and terminal current *I(name)*. If **MDM** is specified, the choices also include capacitance *C(name1,name2)*, conductance *G(name1,name2)*, and admittance *Y(name1,name2)*.	none	
OUT3	char	The third output variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, and terminal current *I(name)*. If **MDM** is specified, the choices also include capacitance *C(name1,name2)*, conductance *G(name1,name2)*, and admittance *Y(name1,name2)*.	none	
OUT4	char	The fourth output variable written to the data file created for IC-CAP. Valid choices include applied bias *VA(name)*, contact bias *V(name)*, and terminal current *I(name)*. If **MDM** is specified, the choices also include capacitance *C(name1,name2)*, conductance *G(name1,name2)*, and admittance *Y(name1,name2)*.	none	
S.PARAM	logical	Specifies that S-parameters found in the Medici log file are written to a data file for IC-CAP. This parameter is valid only when **MDM** is specified.	none	
TERMINAL	char	The names of the two terminals used during the creation of S-parameters in Medici.	none	
STANFORD	logical	Specifies that the Medici log file identified with the **IN.FILE** parameter is converted to a Stanford PISCES-IIB log file format and stored in the file identified with **OUT.FILE**.	none	
IN.FILE	char	The identifier for a Medici I-V log file that is used to create a data file for a parameter extraction program. **synonym: INFILE**	none	
GATE	char	The electrode name associated with the gate for an MOS device.	none	
SOURCE	char	The electrode name associated with the source for an MOS device.	none	
DRAIN	char	The electrode name associated with the drain for an MOS device.	none	
SUBSTRAT	char	The electrode name associated with the substrate for either an MOS or bipolar device.	none	
BASE	char	The electrode name associated with the base for a bipolar device.	none	
EMITTER	char	The electrode name associated with the emitter for a bipolar device.	none	
COLLECT	char	The electrode name associated with the collector for a bipolar device.	none	
EXTRA	char	The electrode name associated with the an arbitrary contact.	none	

Description

The **LOG** statement allows the I-V and/or AC terminal data for a simulation to be written to a file. This statement also provides for the conversion of Medici I-V log files to data files that can be used by common parameter extraction programs. This section describes the **LOG** statement as it functions with the following:

- Medici log files
- Aurora data files
- IC-CAP data files

See Also... To further illustrate the **LOG** statement, refer to:

- Input file *mdex1g* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Gate Characteristics" on page 4-11
- Input file *mdex1d* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Drain Characteristics" on page 4-12
- Most other examples when I-V characteristics are saved for later plotting.

Medici Log Files

A **LOG** statement causes I-V or AC data obtained as a result of all subsequent **SOLVE** statements to be saved. When a **LOG** statement is encountered, any open log file is closed and a new file, as specified by **OUT.FILE**, is opened.

Data stored in log files is used by **PLOT.1D** statements to generate one-dimensional parametric plots. Data is generated for log files whenever a **SOLVE** statement is used to create steady state, transient, or AC solutions. The data stored in a log file can consist of the following:

- Frequency
- Terminal currents
- AC capacitances
- Optical data
- User-defined parameters
- Transient simulation times
- Applied voltages
- AC conductances
- Contact voltages
- Electrode charge
- Impact ionization current
- AC admittances
- Hot electron injection and tunneling currents associated with each electrode

Aurora Data Files

The **LOG** statement can also be used to create a data file that can be read directly by *Avant!* TCAD's parameter extraction program Aurora. To accomplish this:

- The **AURORA** parameter should be specified
- A previously created Medici log file containing the data of interest should be specified with the **IN.FILE** parameter
- The **OUT.FILE** parameter is used to specify the identifier for the output data file for Aurora.
- The parameters **GATE**, **DRAIN**, etc., identify the electrodes for the I-V data in **IN.FILE** that correspond to the gate contact, drain contact, etc., of the simulated structure.

  For each such parameter that is specified, the program writes both the applied voltage and terminal current corresponding to this contact to the data file created for Aurora.

IC-CAP Data Files

The **LOG** statement can also be used to create a data file that can be read directly by Hewlett-Packard's IC-CAP program. To accomplish this:

- The **ICCAP** parameter should be specified. To create a file in the Measured Data Management file format (recommended), the parameter **MDM** should also be specified.
- A previously created Medici log file containing the data of interest should be specified with the **IN.FILE** parameter.
- The **OUT.FILE** parameter should be specified to identify the output data file for IC-CAP.

Input Variables

Up to five input variables can be specified using the parameters **INP1**, **INP2**, **INP3**, **INP4**, and **INP5**. The quantities that can be specified for these parameters include any of the available voltages and currents contained in the Medici log file, or frequency (if an AC small-signal analysis was performed).

When specifying input variables, **INP1** should be used for the most rapidly varying quantity, **INP2** should be used for the next most rapidly varying quantity, and so on. For input variables that are constant, the order is unimportant as long as they are specified after quantities that vary.

By default, Medici will assume that the input variable sweep type is LIN and will look for *start*, *stop*, and *step* values consistent with the data in the log file. The user may, however, indicate that the sweep type is LIST for one or more of the input variables by specifying **LIST1**, **LIST2**, **LIST3**, **LIST4**, and/or **LIST5**.

Note:

The LIST parameters can only be used when writing IC-CAP data files using the* MDM *format.

Output Variables

Up to four output variables can be specified using the parameters **OUT1**, **OUT2**, **OUT3**, and **OUT4**. The quantities that can be specified for these parameters are any of the available voltages and currents contained in the Medici log file. If the **MDM** parameter is specified, the choices also include any capacitance, conductance, and admittance values contained in the Medici log file. The latter quantities are available if an AC small-signal analysis is performed.

S-Parameters

S-parameters can also be written to IC-CAP data files, provided that S-parameters were calculated by Medici during an AC small-signal analysis (the **S.PARAM** parameter on the **SOLVE** statement) and are contained in a Medici log file. This is accomplished by specifying the **S.PARAM** parameter instead of the output variables described above. In this case, the **TERMINAL** parameter is also required to specify which electrodes are terminal "1" and terminal "2". S-parameters are only allowed when writing data using the **MDM** format.

Electrode Names

The parameters **GATE**, **DRAIN**, etc., identify the electrode names for the data in **IN.FILE** that correspond to the gate contact, drain contact, etc., of the simulated structure. If these parameters are specified, then appropriate mnemonics are substituted for the electrode names when the input and output variables are written to the IC-CAP data file.

IC-CAP Example

Consider a case where Medici is used to simulate the gate characteristics for a MOS device (*Id* vs. *Vg*) for several substrate biases (*Vb*). In Medici, electrode names "Drain", "Gate", "Src" and "Subst" correspond to the drain, gate, source, and substrate contacts, respectively, and that the I-V data is stored in the file *medici.log*.

The following statement can be used to create a data file for IC-CAP using the **MDM** format:

```
LOG    ICCAP  MDM  IN.FILE=medici.log  OUT.FILE=iccap.mdm
+      DRAIN=Drain  GATE=Gate  SOURCE=Src  SUBSTRAT=Subst
+      OUT1=I(Drain)  INP1=V(Gate)  INP2=V(Subst)
+      INP3=V(Drain)  INP4=V(Src)
```

In this example,

- Drain current is identified as the output variable with the parameter **OUT1**.
- Gate voltage is identified as the most rapidly varying input variable with the parameter **INP1**.
- The parameter **INP2** identifies the substrate voltage as the next most rapidly varying input variable.
- Both the drain voltage and the source voltage are constant and are identified with the input parameters **INP3** and **INP4**.
- The parameters **DRAIN**, **GATE**, **SOURCE**, and **SUBSTRAT** cause the name *id* to be used for the output variable and the names *vg*, *vb*, *vd*, and *vs* to be used for the four input variables in the IC-CAP data file.

Data File Names

To be compatible with IC-CAP nomenclature, the data file for IC-CAP should end with the suffix "*.set*" (for example, *iccap.set*) when **MDM** is not specified, or with the suffix ".mdm" if the **MDM** format is used (for example, *iccap.mdm* in the above example).

LOAD

The **LOAD** statement reads previous solutions from files to use as initial guesses for continued simulations or for post-processing.

```
LOAD

   [ IN.FILE=<c> [ASCII.IN] ] [ IN.PREV=<c> [DIFFEREN] ]
   [ OUT.FILE=<c> [ASCII.OU] ] [CHECK.ER] [TIF]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]
```

Parameter	Type	Definition	Default
IN.FILE	char	The identifier for a file containing a solution to read in. **synonym: INFILE**	none
ASCII.IN	logical	Specifies that **IN.FILE** is a formatted file.	false
IN.PREV	char	The identifier for a file containing a second solution to read in. **synonym: IN2FILE**	none
DIFFEREN	logical	Specifies that the difference between the solutions in **IN.FILE** and **IN.PREV** is calculated.	false
OUT.FILE	char	The identifier for a file in which to store a solution. **synonym: OUTFILE**	none
ASCII.OU	logical	Specifies that **OUT.FILE** is a formatted file.	false
CHECK.ER	logical	Specifies that an inconsistency between the solution read in and the presently available device structure flags an error and terminates execution. If this parameter is false, a warning is given and an attempt to continue execution is made.	true
TIF	logical	Specifies that the **TIF** format is assumed for the input file.	false
Circuit Analysis AAM Parameters			
STRUCTUR	char	Specifies the device to load. This parameter is only used with the Circuit Analysis AAM.	all devices

Description

The **LOAD** statement reads previous solutions from files to use as initial guesses for continued simulations or for post-processing.

See Also...

To further illustrate the **LOAD** statement, refer to:

- Input file *mdex1d* in N-Channel MOSFET Examples, Chapter 4, "Initialization" on page 4-11

- Input file *mdex1g* in N-Channel MOSFET Examples, Chapter 4, "Simulation of Gate Characteristics" on page 4-11
- Input file *mdex2d* in NPN Bipolar Transistor Examples, Chapter 5, "Generation of the Simulation Structure" on page 5-1
- Several other examples

Parameters

The **LOAD** statement uses a variety of parameters to read, store, and analyze differences between stored solutions.

Single Solution

A previous solution which is stored in a file can be read using the **IN.FILE** parameter. By default, the program assumes the solution is stored in a binary file. If the solution is stored in a formatted file, the **ASCII.IN** parameter should be specified.

Two Solutions

Two previous solutions stored in files can be read using the **IN.FILE** and **IN.PREV** parameters. This may be desired, for example, so that a projection can be used to obtain the initial guess for the next solution.

Since the program only stores the two most recent solutions, the solution read in with **IN.PREV** is the first to be lost when new solutions are obtained.

Analyzing Differences

The difference between two solutions can be analyzed by reading in both and specifying the **DIFFEREN** parameter. The difference is stored as a solution but can only be used for plotting or extracting data.

The difference solution can not be used as an initial guess for subsequent solutions.

Storing a Solution

The **LOAD** statement can also be used to store a solution. This is accomplished by specifying the **OUT.FILE** parameter. By default, the solution is stored in a binary file. To store the solution in a formatted file, the parameter **ASCII.OU** should be specified.

Solution File Data

The data stored in a solution file includes:

- The potential at each node
- The carrier concentrations at each node
- The carrier temperature at each node
- The lattice temperature at each node

In addition, information describing the setup used to obtain the solution is stored. This includes:

- Physical parameters for the materials in the device structure
- Special boundary conditions that may have been specified
- Flags indicating what physical models were selected.

Setup Information

When a solution file is read using the **LOAD** statement, the setup information contained in the file is used by default for all subsequent solutions. The setup may be modified using **MATERIAL**, **MOBILITY**, **CONTACT**, **INTERFACE**, and **MODELS** statements.

SAVE

Specifies files in which to save data.

```
SAVE

   OUT.FILE=<c>

   {  ( SOLUTION [STRUCTUR=<c>] [ASCII] )
    | ( MESH [W.MODELS] [ASCII] )

  Technology Interchange Format
     | ( TIF [ALL] [BANDS] [CURRENTS] [GENERATI] [COMPONEN]
         [AC.POTEN] [AC.CN] [AC.CP] [AC.TN] [AC.TP] [AC.TL]
         [AC.JDISP] [AC.JN] [AC.JP] [AC.JCOND] [AC.JTOT]
         [AC.COMP] [AC.SCOMP] )

  Structure Parameters
         [MESH] [BOUND]

  Scalar Quantities
         [POTENTIA] [QFN] [QFP] [VALENC.B] [CONDUC.B] [VACUUM]
         [DOPING] [ELECTRON] [HOLES] [NET.CHAR] [NET.CARR]
         [RECOMBIN] [II.GENER] [BB.GENER] [PHOTOGEN]
         [ELE.TEMP] [HOL.TEMP] [ELE.VEL] [HOL.VEL] [J.EFIELD]
         [G.GAMN] [G.GAMP] [G.GAMT] [G.IN] [G.IP] [G.IT]

  Vector Quantities
         [J.CONDUC] [J.ELECTR] [J.HOLE] [J.DISPLA] [J.TOTAL] [E.FIELD]

  Lattice Temperature AAM Parameters
         [LAT.TEMP]

  Heterojunction Device AAM Parameters
         [X.MOLE]
       )
   }
```

Parameter	Type	Definition	Default
OUT.FILE	char	The identifier for the file in which the information is written.	none
SOLUTION	logical	Specifies that the solution information shall be written to the file.	false
STRUCTUR	char	Specifies the device for which the solution information is written. This parameter is only used with the Circuit Analysis AAM.	all devices
ASCII	logical	Indicates that the specified file is a formatted file.	false
MESH	logical	Specifies that the mesh information (node and element information) is written to the file. Circuit information is also written if the Circuit Analysis AAM is available.	false

Parameter	Type	Definition	Default
W.MODELS	logical	Specifies that the information written contains model and coefficient information.	false

Technology Interface Format

TIF	logical	Specifies that the TIF format is used for the output file. The output automatically includes structure information and some basic physical quantities such as doping, potential, carrier concentrations, carrier and lattice temperatures, electric field and total current density. If the Circuit Analysis AAM is being used, the circuit (topology, element values), circuit node voltages, and currents are also written to the TIF file.	false
ALL	logical	Specifies that all available physical quantities are written to the TIF file.	false
BANDS	logical	Specifies that physical quantities associated with the band structure of the device are written to the TIF file. These include electron and hole quasi-Fermi potentials, and valence band, conduction band, and vacuum level potentials.	false
CURRENTS	logical	Specifies that electron, hole, and displacement current densities and carrier velocities are written to the output file.	false
GENERATI	logical	Specifies that impact ionization generation, band-to-band tunneling generation, photogeneration, and recombination are written to the TIF file.	false
COMPONEN	logical	Specifies that components of vector quantities are written to the TIF file in addition to the magnitudes of these quantities.	false
AC.POTEN	logical	Specifies that values of the magnitude of AC potential in volts are written to the TIF file.	false
AC.CN	logical	Specifies that values of the magnitude of AC electron concentration in number per cubic centimeter are written to the TIF file.	false
AC.CP	logical	Specifies that values of the magnitude of AC hole concentration in number per cubic centimeter are written to the TIF file.	false
AC.TN	logical	Specifies that values of the magnitude of AC electron temperature in Kelvins are written to the TIF file.	false
AC.TP	logical	Specifies that values of the magnitude of AC hole temperature in Kelvins are written to the TIF file.	false
AC.TL	logical	Specifies that values of the magnitude of AC lattice temperature in Kelvins are written to the TIF file.	false
AJDISP	logical	Specifies that values of the magnitude of AC displacement current in amps per square centimeter are written to the TIF file.	false
AC.JN	logical	Specifies that values of the magnitude of AC electron current in amps per square centimeter are written to the TIF file.	false
AC.JP	logical	Specifies that values of the magnitude of AC hole current in amps per square centimeter are written to the TIF file.	false
AC.JCOND	logical	Specifies that values of the magnitude of AC conduction current in amps per square centimeter are written to the TIF file.	false
AC.JTOT	logical	Specifies that values of the magnitude of AC total current in amps per square centimeter are written to the TIF file.	false

Parameter	Type	Definition	Default
AC.COMP	logical	Specifies that complex components of AC quantities are written to the TIF file in addition to the magnitudes of these quantities.	false
AC.SCOMP	logical	Specifies that spatial components of AC vector quantities are written to the TIF file in addition to the magnitudes of these quantities.	false
Structure Parameters			
BOUND	logical	Specifies that boundary information is written to the data file.	false
Scalar Quantities			
POTENTIA	logical	Specifies that values of mid-gap potential in volts are written to the data file.	false
QFN	logical	Specifies that values of electron quasi-Fermi potential in volts are written to the data file.	false
QFP	logical	Specifies that values of hole quasi-Fermi potential in volts are written to the data file.	false
VALENC.B	logical	Specifies that values of valence band potential in volts are written to the data file.	false
CONDUC.B	logical	Specifies that values of conduction band potential in volts are written to the data file.	false
VACUUM	logical	Specifies that values of vacuum potential in volts are written to the data file.	false
DOPING	logical	Specifies that values of net impurity concentration in number per cubic centimeter are written to the data file.	false
ELECTRON	logical	Specifies that values of electron concentration in number per cubic centimeter are written to the data file.	false
HOLES	logical	Specifies that values of hole concentration in number per cubic centimeter are written to the data file.	false
NET.CHAR	logical	Specifies that values of net charge concentration in number per cubic centimeter are written to the data file.	false
NET.CARR	logical	Specifies that values of net carrier concentration in number per cubic centimeter are written to the data file.	false
RECOMBIN	logical	Specifies that values of net recombination in number per cubic centimeter per second are written to the data file.	false
II.GENER	logical	Specifies that values of the total generation rate due to impact ionization in pairs per cubic centimeter per second are written to the data file.	false
BB.GENER	logical	Specifies that values of band-to-band tunneling generation rate in pairs per cubic centimeter per second are written to the data file.	false
PHOTOGEN	logical	Specifies that total photogeneration in pairs per cubic centimeter per second are written to the data file.	false
ELE.TEMP	logical	Specifies that values of electron temperature in Kelvins are written to the data file.	false
HOL.TEMP	logical	Specifies that values of hole temperature in Kelvins are written to the data file.	false

Parameter	Type	Definition	Default
ELE.VEL	logical	Specifies that values of electron mean velocity in cm/s are written to the data file.	false
HOL.VEL	logical	Specifies that values of hole mean velocity in cm/s are written to the data file.	false
J.EFIELD	logical	Specifies that values of J dot E (the dot product of the current density with the electric field) in watts per cubic centimeter are written to the data file.	false
G.GAMN	logical	Specifies that values of probability that an electron will be injected into the oxide are written to the data file.	false
G.GAMP	logical	Specifies that values of probability that a hole will be injected into the oxide are written to the data file.	false
G.GAMT	logical	Specifies that values of probability that a hole or electron (that is, the sum of the hole and electron probabilities) will be injected into the oxide are written to the data file.	false
G.IN	logical	Specifies that values of hot electron injection current initiated from each point in amps/micron are written to the data file.	false
G.IP	logical	Specifies that values of hot hole injection current initiated from each point in amps/micron are written to the data file.	false
G.IT	logical	Specifies that values of total hot carrier injection current initiated from each point in amps/micron are written to the data file.	false
Vector Quantities			
J.CONDUC	logical	Specifies that the magnitude and vectors of conduction current density in amps per square centimeter are written to the data file.	false
J.ELECTR	logical	Specifies that the magnitude and vectors of electron current density in amps per square centimeter are written to the data file.	false
J.HOLE	logical	Specifies that the magnitude and vectors of hole current density in amps per square centimeter are written to the data file.	false
J.DISPLA	logical	Specifies that the magnitude and vectors of displacement current density in amps per square centimeter are written to the data file.	false
J.TOTAL	logical	Specifies that the magnitude and vectors of total current in amps per square centimeter are written to the data file.	false
E.FIELD	logical	Specifies that the magnitude and vectors of electric field in volts per centimeter are written to the data file.	false
Lattice Temperature AAM Parameters			
LAT.TEMP	logical	Specifies that values of lattice temperature in Kelvins are written to the data file. This parameter is only used with the Lattice Temperature AAM.	false
Heterojunction Device AAM Parameters			
X.MOLE	logical	Specifies that values of mole fraction are written to the data file. This parameter is only used with the Heterojunction Device AAM.	false

Description

The **SAVE** statement specifies files in which to save data, and provides a convenient way to save simulation results at any point during the simulation.

See Also...

To further illustrate the **SAVE** statement, refer to:

- *mdex9b* in Interface Examples, Chapter 10, "Interface to TSUPREM-4" on page 10-11
- *mdex11* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-2
- *mdex17* in Heterojunction Examples, Chapter 14, "High Electron Mobility Transistor Simulation" on page 14-7
- Any other examples with the **SAVE** statement.

File Formats

The **TIF** format should be used for communication between the Medici program and other *Avant!* TCAD programs such as Taurus Visual and Taurus WorkBench.

When specified with the **DATEX** parameter, the value of the **OUT.FILE** parameter is the root name of up to three data files to be written (each named with a separate suffix), and should therefore *not* have a suffix. The following is an example of the correct syntax.

```
SAVE  TIF   OUT.FILE=SOL11.TIF  ALL
SAVE  MESH  OUT.FILE=NMOS.000  W.MODELS
SAVE  DATEX  OUT.FILE=nmos1  MESH  BOUND  ELECTRON  HOLE
+                          POTENTIA
```

Saving AC Quantities in TIF Files

The internal solution distributions obtained from an AC small-signal analysis can be saved in a TIF file for visualization with Taurus Visual. This is accomplished on the **SAVE** statement by selecting among the parameters **AC.POTEN**, **AC.CN**, **AC.CP**, **AC.TN**, **AC.TP**, **AC.TL** **AC.JDISP**, **AC.JN**, **AC.JP**, **AC.JCOND**, and **AC.JTOT**.

The magnitude of a requested AC quantity is always written to the TIF file. If **AC.COMP** is specified, then the real and imaginary parts of the AC quantities are also written to the TIF file. If **AC.SCOMP** is specified, then the spatial components of requested AC vector quantities are also written to the TIF file.

Note:

Because of the large amount of data involved, specifying the* ALL *parameter will not cause the AC internal distributions to be saved. To save AC quantities in TIF files, they must be specified explicitly.

3.4 Coefficients and Material Parameters

The following statements specify the material parameters and coefficients used by Medici:

Statement	Definition	Page
MATERIAL	Specifies material properties.	3-252
MOBILITY	Specifies parameters associated with mobility models.	3-275
IMPURITY	Specifies parameters associated with impurities.	3-298
CONTACT	Specifies parameters associated with electrodes; specifies special boundary conditions.	3-301
INTERFACE	Specifies interface parameters for the structure.	3-306
ANISOTROPIC	Specifies anisotropic material coefficients.	3-309

MATERIAL

The **MATERIAL** statement associates physical parameters with the materials in the device structure.

```
MATERIAL

   [PRINT]

 Semiconductor Parameters
   [ {  ( {  SILICON | GAAS | POLYSILI | SEMICOND | SIGE | ALGAAS
           | GERMANIU | SIC | S.OXIDE | HGCDTE | INGAAS | INP | INAS
           | DIAMOND | ZNSE | ZNTE | A-SILICO | ALINAS | GAASP | INGAP
           | INASP | REGION=<c>
          }
          [PERMITTI=<n>] [EG.MODEL=<n>] [AFFINITY=<n>] [EG300=<n>]
          [EGALPH=<n>] [EGBETA=<n>] [EGGAMM=<n>]
          [NC300=<n>] [NC.F=<n>] [NV300=<n>] [NV.F=<n>]
          [GCB=<n>] [GVB=<n>] [EDB=<n>] [EAB=<n>]
          [TAUN0=<n>] [NSRHN=<n>] [AN=<n>] [BN=<n>] [CN=<n>] [EN=<n>]
          [TAUP0=<n>] [NSRHP=<n>] [AP=<n>] [BP=<n>] [CP=<n>] [EP=<n>]
          [EXN.TAU=<n>] [EXP.TAU=<n>]
          [ETRAP=<n>] [M.RTUN=<n>] [B.RTUN=<n>] [S.RTUN=<n>]
          [E.RTUN=<n>] [C.DIRECT=<n>]
          [AUGN=<n>] [AUGP=<n>] [ARICHN=<n>] [ARICHP=<n>]
          [N0.BGN=<n>] [V0.BGN=<n>] [CON.BGN=<n>]
          [A.EHS=<n>] [B.EHS=<n>] [C.EHS=<n>]
         [N.IONIZA=<n>] [N.ION.1=<n>] [N.ION.2=<n>] [ECN.II=<n>] [EXN.II=<n>]
         [P.IONIZA=<n>] [P.ION.1=<n>] [P.ION.2=<n>] [ECP.II=<n>] [EXP.II=<n>]
          [CN.IIGAP=<n>] [CP.IIGAP=<n>]
          [LAMHN=<n>] [LAMRN=<n>] [LAMHP=<n>] [LAMRP=<n>]
          [A.BTBT=<n>] [B.BTBT=<n>] [A.FN=<n>] [B.FN=<n>]
          {[KAPPA.QM=<n>] | ( [KAPPA.N=<n>] [KAPPA.P=<n>] )}
          [N.ACCUM=<n>] [P.ACCUM=<n>] [DREF.QM=<n>]
          [QM.NORP=<n>] [QM.EFIEL=<n>]
          [U.STRESS=<n>] [D.STRESS=<n>] [A.STRESS=<n>]
          [B.STRESS=<n>] [C.STRESS=<n>]
          [ME.DT=<n>] [MHH.DT=<n>][MLH.DT=<n>]

 Energy Balance Equation Parameters
          [ELE.CQ=<n>] [ELE.TAUW=<n>] [WTN0=<n>] [WTN1=<n>]
          [WTN2=<n>] [WTN3=<n>] [WTN4=<n>] [WTN5=<n>] [WTNL=<n>] [TNL=<n>]
          [HOL.CQ=<n>] [HOL.TAUW=<n>] [WTP0=<n>] [WTP1=<n>]
          [WTP2=<n>] [WTP3=<n>] [WTP4=<n>] [WTP5=<n>] [WTPL=<n>] [TPL=<n>]

 Lattice Temperature AAM Parameters for Semiconductors
          [DENSITY=<n>] [DN.LAT=<n>] [DP.LAT=<n>]
          [A.SP.HEA=<n>] [B.SP.HEA=<n>] [C.SP.HEA=<n>] [D.SP.HEA=<n>]
          [F.SP.HEA=<n>] [G.SP.HEA=<n>]
```

(**MATERIAL** statement continued on next page)

(**MATERIAL** statement continued from previous page)

```
          [A.TH.CON=<n>] [B.TH.CON=<n>] [C.TH.CON=<n>] [D.TH.CON=<n>]
          [E.TH.CON=<n>]
          [OP.PH.EN=<n>] [LAN300=<n>] [LAP300=<n>]

 Heterojunction Device AAM Parameters
          [ {X.MOLE=<n> | X.OTHER=<c>} ] [EPS.X1=<n>] [EPS.X2=<n>]
          [NC.0=<n>] [NC.E=<n>] [NV.0=<n>] [NV.E=<n>] [EM.MODEL=<n>]
          [EG.X0=<n>] [EG.X1=<n>] [EG.X2=<n>] [EG.X3=<n>] [EG.X4=<n>]
          [EG.X5=<n>] [EG.X6=<n>] [EG.X7=<n>] [EG.X8=<n>] [EG.X9=<n>]
          [EG.X10=<n>] [EG.X11=<n>] [EG.X12=<n>] [EG.X13=<n>] [EG.X14=<n>]
          [AF.X0=<n>] [AF.X1=<n>] [AF.X2=<n>] [AF.X3=<n>] [AF.X4=<n>]
          [AF.X5=<n>] [AF.XL=<n>] [EGALX=<n>] [EGBEX=<n>] [EGGAX=<n>]
          [EGALL=<n>] [EGBEL=<n>] [EGGAL=<n>]
          [MEG=<n>] [MEG.X1=<n>] [MEX=<n>] [MEX.X1=<n>] [MEL=<n>]
          [MEL.X1=<n>] [MH0=<n>] [MH0.X1=<n>] [ML0=<n>] [ML0.X1=<n>]
        )

 Insulator Parameters
      | ( {OXIDE | NITRIDE | SAPPHIRE | OXYNITRI | INSULATO | REGION=<c>}
          [PERMITTI=<n>] [AFFINITY=<n>] [EG300=<n>]
          [ECN.GC=<n>] [BARLN=<n>] [TUNLN=<n>]
          [ECP.GC=<n>] [BARLP=<n>] [TUNLP=<n>]
          [ME.DT=<n>]

 Lattice Temperature AAM Parameters for Insulators
          [DENSITY=<n>]
          [A.SP.HEA=<n>] [B.SP.HEA=<n>] [C.SP.HEA=<n>] [D.SP.HEA=<n>]
          [F.SP.HEA=<n>] [G.SP.HEA=<n>]
          [A.TH.CON=<n>] [B.TH.CON=<n>] [C.TH.CON=<n>] [D.TH.CON=<n>]
          [E.TH.CON=<n>]
        )

 Electrode Parameters
      | ( ELECTROD=<c> [A.FN=<n>] [B.FN=<n>] [ME.DT=<n>] [BARR.DT=<n>] )
     }
  ]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]
```

(**MATERIAL** statement continued on next page)

(**MATERIAL** statement continued from previous page)

```
 Optical Device AAM Parameters
   [PR.TABLE]

  Optical Energy Gap
   [EGO300=<n>] [EGOALPH=<n>] [EGOBETA=<n>]

   Real Refractive Index
   [ {( WAVE.RE=<a> INDEX.RE=<a> ) | RRI.FILE=<c>} [FIRST] [LAST] ]

   Imaginary Refractive Index or Absorption Coefficient
   [ {  ( {  ( WAVE.IM=<a> {INDEX.IM=<a> | ABSORPTI=<a>} )
           | ( {IRI.FILE=<c> | ABS.FILE=<c>} )
         }
         [FIRST] [LAST]
       )
     | ( BTBT.AB B.BB=<a> E.PHONON=<a> E1.BB=<a> EXP.BB=<a> )
    }
   ]
   [ BATA.AB [E1.BT=<n>] {E.URBACH=<n> | G.BT=<n>} ]
   [ FRCA.AB [EL.EMAS=<n>] [HO.EMAS=<n>] ]
```

Parameter	Type	Definition	Default	Units
PRINT	logical	Specifies that semiconductor and insulator material parameters for the structure are printed to the standard output.	false	

Semiconductor Parameters

SILICON	logical	Specifies that the material parameters apply to all regions that were specified as **SILICON** with **REGION** statements.	false	
GAAS	logical	Specifies that the material parameters apply to all regions that were specified as **GAAS** with **REGION** statements.	false	
POLYSILI	logical	Specifies that the material parameters apply to all regions that were specified as **POLYSILI** with **REGION** statements.	false	
SEMICOND	logical	Specifies that the material parameters apply to all regions that were specified as **SEMICOND** with **REGION** statements.	false	
SIGE	logical	Specifies that the material parameters apply to all regions that were specified as **SIGE** with **REGION** statements.	false	
ALGAAS	logical	Specifies that the material parameters apply to all regions that were specified as **ALGAAS** with **REGION** statements.	false	
GERMANIU	logical	Specifies that the material parameters apply to all regions that were specified as **GERMANIU** with **REGION** statements.	false	
SIC	logical	Specifies that the material parameters apply to all regions that were specified as **SIC** with **REGION** statements.	false	
S.OXIDE	logical	Specifies that the material parameters apply to all regions that were specified as **S.OXIDE** with **REGION** statements.	false	

Parameter	Type	Definition	Default	Units
HGCDTE	logical	Specifies that the material parameters apply to all regions that were specified as **HGCDTE** with **REGION** statements.	false	
INGAAS	logical	Specifies that the material parameters apply to all regions that were specified as **INGAAS** with **REGION** statements.	false	
INP	logical	Specifies that the material parameters apply to all regions that were specified as **INP** with **REGION** statements.	false	
INAS	logical	Specifies that the material parameters apply to all regions that were specified as **INAS** with **REGION** statements.	false	
DIAMOND	logical	Specifies that the material parameters apply to all regions that were specified as **DIAMOND** with **REGION** statements.	false	
ZNSE	logical	Specifies that the material parameters apply to all regions that were specified as **ZNSE** with **REGION** statements.	false	
ZNTE	logical	Specifies that the material parameters apply to all regions that were specified as **ZNTE** with **REGION** statements.	false	
A-SILICO	logical	Specifies that the material parameters apply to all regions that were specified as **A-SILICO** with **REGION** statements.	false	
ALINAS	logical	Specifies that the material parameters apply to all regions that were specified as **ALINAS** with **REGION** statements	false	
GAASP	logical	Specifies that the material parameters apply to all regions that were specified as **GAASP** with **REGION** statements	false	
INGAP	logical	Specifies that the material parameters apply to all regions that were specified as **INGAP** with **REGION** statements	false	
INASP	logical	Specifies that the material parameters apply to all regions that were specified as **INASP** with **REGION** statements	false	
REGION	char	The name(s) of the regions for which the material parameters apply. If more than one name is given, the entire group should be surrounded by parentheses and the individual names should be separated with commas.	none	
PERMITTI	number	The relative dielectric permittivity of the material.	See tables	none
EG.MODEL	number	The energy bandgap model to use for the specified material or region: = 0 ==> use $E_g(300)$ = 1 ==> calculate dependence on lattice temperature only = 2 ==> use model for strained $Si_{1-x}Ge_x$ = 3 ==> use model for unstrained $Si_{1-x}Ge_x$ = 4 ==> use model for ternary III-V materials Specifying **EG.MODEL**=2, 3, or 4 requires that the Heterojunction Device AAM be enabled.	See tables	none
AFFINITY	number	The electron affinity for the material.	See tables	volts
EG300	number	The energy bandgap of the material at 300 K.	See tables	eV
EGALPH	number	The value of alpha used in calculating the energy bandgap as a function of temperature (the energy gap of band Γ if **EM.MODEL** set to 1).	See tables	eV/Kelvin

Parameter	Type	Definition	Default	Units
EGBETA	number	The value of beta used in calculating the energy bandgap as a function of temperature (the energy gap of band Γ if **EM.MODEL** set to 1).	See tables	Kelvins
EGGAMM	number	The value of gamma used in calculating the energy bandgap as a function of temperature (the energy gap of band Γ if **EM.MODEL** set to 1).	See tables	eV/Kelvins
NC300	number	The effective density of states in the conduction band of semiconductor at 300 K.	See tables	$\#/cm^3$
NC.F	number	The exponent of temperature for describing the effective density of states in the conduction band.	See tables	none
NV300	number	The effective density of states in the valence band of semiconductor at 300 K.	See tables	$\#/cm^3$
NV.F	number	The exponent of temperature for describing the effective density of states in the valence band.	See tables	none
GCB	number	The conduction band degeneracy factor.	See tables	none
GVB	number	The valence band degeneracy factor.	See tables	none
EDB	number	The donor ionization energy referenced to the conduction band energy. That is, **EDB** = $E_C - E_D$.	See tables	eV
EAB	number	The acceptor ionization energy referenced to the valence band energy. That is, **EAB** = $E_A - E_V$.	See tables	eV
TAUN0	number	The Shockley-Read-Hall electron lifetime.	See tables	s
NSRHN	number	The Shockley-Read-Hall concentration parameter for electrons.	See tables	$\#/cm^3$
AN	number	The constant term in the concentration-dependent expression for electron lifetime.	See tables	none
BN	number	The linear term coefficient in the concentration-dependent expression for electron lifetime.	See tables	none
CN	number	The exponential term coefficient in the concentration-dependent expression for electron lifetime.	See tables	none
EN	number	The exponent in the concentration-dependent expression for electron lifetime.	See tables	none
TAUP0	number	The Shockley-Read-Hall hole lifetime.	See tables	s
NSRHP	number	The Shockley-Read-Hall concentration parameter for holes.	See tables	$\#/cm^3$
AP	number	The constant term in the concentration-dependent expression for hole lifetime.	See tables	none
BP	number	The linear term coefficient in the concentration-dependent expression for hole lifetime.	See tables	none
CP	number	The exponential term coefficient in the concentration-dependent expression for hole lifetime.	See tables	none
EP	number	The exponent in the concentration-dependent expression for hole lifetime.	See tables	none

Parameter	Type	Definition	Default	Units
EXN.TAU	number	The exponent of temperature for describing the electron lifetime dependence on lattice temperature. Specifying a non-zero value for this parameter invokes the lattice temperature-dependent electron lifetime model.	See tables	none
EXP.TAU	number	The exponent of temperature for describing the hole lifetime dependence on lattice temperature. Specifying a non-zero value for this parameter invokes the lattice temperature-dependent hole lifetime model.	See tables	none
ETRAP	number	The trap level (E_t - E_i) used in determining the Shockley-Read-Hall recombination rate.	See tables	eV
M.RTUN	number	The trap-assisted tunneling effective mass.	See tables	free electron rest mass m_0
B.RTUN	number	Band-to-band tunneling rate proportionality factor.	See tables	depends on **S.RTUN**
S.RTUN	number	Band-to-band field power ratio.	See tables	none
E.RTUN	number	Band-to-band reference electric field.	See tables	V/cm
C.DIRECT	number	The band-to-band recombination coefficient. The band-to-band recombination model is activated by specifying **C.DIRECT** > 0.0.	See tables	cm^3/s
AUGN	number	The Auger coefficient for electrons.	See tables	cm^6/s
AUGP	number	The Auger coefficient for holes.	See tables	cm^6/s
ARICHN	number	The effective Richardson constant for electrons.	See tables	amp/(Kelvins-cm)^2
ARICHP	number	The effective Richardson constant for holes.	See tables	amp/(Kelvins-cm)^2
N0.BGN	number	The concentration parameter used in band-gap narrowing model.	See tables	$1/cm^3$
V0.BGN	number	The voltage parameter used in the band-gap narrowing model.	See tables	volts
CON.BGN	number	The constant parameter used in the band-gap narrowing model.	See tables	none
A.EHS	number	First electron-hole scattering parameter.	See tables	V-cm-s
B.EHS	number	Second electron-hole scattering parameter.	See tables	$1/cm^3$
C.EHS	number	Third electron-hole scattering parameter.	See tables	$1/cm^3$
N.IONIZA	number	The constant term in the multiplicative prefactor of the electron ionization coefficient. **synonym: N.ION.0**	See tables	1/cm
N.ION.1	number	The coefficient multiplying T_{lat} in the multiplicative prefactor of the electron ionization coefficient.	See tables	1/cm-K
N.ION.2	number	The coefficient multiplying $(T_{lat})^2$ in the multiplicative prefactor of the electron ionization coefficient.	See tables	$1/cm\text{-}K^2$
ECN.II	number	The critical electric field used in the exponential factor of the electron ionization coefficient.	See tables	volts/cm

Parameter	Type	Definition	Default	Units
EXN.II	number	The exponent of the ratio of the critical electrical field to the local electric field used in the exponential factor of the electron ionization coefficient.	See tables	none
P.IONIZA	number	The constant term in the multiplicative prefactor of the hole ionization coefficient. **synonym: P.ION.0**	See tables	1/cm
P.ION.1	number	The coefficient multiplying T_{lat} in the multiplicative prefactor of the hole ionization coefficient.	See tables	1/cm-k
P.ION.2	number	The coefficient multiplying $(T_{lat})^2$ in the multiplicative prefactor of the hole ionization coefficient.	See tables	1/cm-k^2
ECP.II	number	The critical electric field used in the exponential factor of the hole ionization coefficient.	See tables	volts/cm
EXP.II	number	The exponent of the ratio of the critical electrical field to the local electric field used in the exponential factor of the hole ionization coefficient.	See tables	none
CN.IIGAP	number	Coefficient of electron impact ionization threshold energy	1.0	none
CP.IIGAP	number	Coefficient of hole impact ionization threshold energy	1.0	none
LAMHN	number	The hot-electron scattering mean-free path length in the semiconductor used in the gate current model.	See tables	cm
LAMRN	number	The electron re-direction scattering mean free path length used in the gate current model.	See tables	cm
LAMHP	number	The hot-hole scattering mean-free path length in the semiconductor used in the gate current model.	See tables	cm
LAMRP	number	The hole re-direction scattering mean free path length used in the gate current model.	See tables	cm
A.BTBT	number	Coefficient of the pre-exponential term for the band-to-band tunneling model.	See tables	$eV^{0.5}$/cm-s-V^2
B.BTBT	number	Coefficient of the exponential term for the band-to-band tunneling model.	See tables	V/cm-$eV^{1.5}$
A.FN	number	Coefficient of the pre-exponential term for the Fowler-Nordheim tunneling model.	See tables	A/V^2
B.FN	number	Coefficient of the exponential term for the Fowler-Nordheim tunneling model.	See tables	V/cm
KAPPA.QM	number	Multiplication factor for the band-gap widening expression when **QM.PHILI** model is used	See tables	none
KAPPA.N	number	Multiplication factor for the band-gap widening expression with the electron-inducing field when **QM.PHILI** model is used.	See tables	none
KAPPA.P	number	Multiplication factor for the band-gap widening expression with the hole-inducing field when **QM.PHILI** model is used.	See tables	none
N.ACCUM	number	Threshold concentration for the band-gap widening expression in electron accumulation regime when **QM.PHILI** model is used.	See tables	$1/cm^3$
P.ACCUM	number	Threshold concentration for the band-gap widening expression in hole accumulation regime when **QM.PHILI** model is used.	See tables	$1/cm^3$

Parameter	Type	Definition	Default	Units
DREF.QM	number	Reference distance used with the **QM.PHILI** model.	See tables	microns
QM.NORP	number	An integer that specifies whether quantum mechanical band-gap widening occurs in n-type regions (**QM.NORP**=1), p-type regions (**QM.NORP**=-1), or both (**QM.NORP**=0).	0	none
QM.EFIEL	number	An integer that specifies whether quantum mechanical band-gap widening occurs in regions where the normal electric field at the interface points into the semiconductor (**QM.EFIEL**=1), points into the insulator (**QM.EFIEL**=-1), or both (**QM.EFIEL**=0).	0	none
U.STRESS	number	One of the deformation potential coefficients used in calculating the change in the conduction band edge with the **STRESS** model. For silicon only.	See tables	eV
D.STRESS	number	One of the deformation potential coefficients used in calculating the change in the conduction band edge with the **STRESS** model. For silicon only.	See tables	eV
A.STRESS	number	One of the deformation potential coefficients used in calculating the change in the valence band edge with the **STRESS** model. For silicon only.	See tables	eV
B.STRESS	number	One of the deformation potential coefficients used in calculating the change in the valence band edge with the **STRESS** model. For silicon only.	See tables	eV
C.STRESS	number	One of the deformation potential coefficients used in calculating the change in the valence band edge with the **STRESS** model. For silicon only.	See tables	eV
ME.DT	number	The effective tunneling mass of electrons used in the direct tunneling model.	See tables	
MHH.DT	number	The effective tunneling mass of electrons in the heavy-hole band used in the VBET direct tunneling model.	See tables	
MLH.DT	number	The effective tunneling mass of electrons in the light-hole band used in the VBET direct tunneling model.	See tables	

Energy Balance Equation Parameters

ELE.CQ	number	Electron thermal conductivity coefficient.	See tables	none
ELE.TAUW	number	Electron energy relaxation time.	See tables	s
WTN0	number	Coefficient appearing in the electron relaxation time model.	See tables	s
WTN1	number	Coefficient appearing in the electron relaxation time model.	See tables	s
WTN2	number	Coefficient appearing in the electron relaxation time model.	See tables	s
WTN3	number	Coefficient appearing in the electron relaxation time model.	See tables	s
WTN4	number	Coefficient appearing in the electron relaxation time model.	See tables	s
WTN5	number	Coefficient appearing in the electron relaxation time model.	See tables	s
WTNL	number	Coefficient appearing in the electron relaxation time model.	See tables	s
TNL	number	Coefficient appearing in the electron relaxation time model.	See tables	K
HOL.CQ	number	Hole thermal conductivity coefficient.	See tables	none

Parameter	Type	Definition	Default	Units
HOL.TAUW	number	Hole energy relaxation time.	See tables	s
WTP0	number	Coefficient appearing in the hole relaxation time model.	See tables	s
WTP1	number	Coefficient appearing in the hole relaxation time model.	See tables	s
WTP2	number	Coefficient appearing in the hole relaxation time model.	See tables	s
WTP3	number	Coefficient appearing in the hole relaxation time model.	See tables	s
WTP4	number	Coefficient appearing in the hole relaxation time model.	See tables	s
WTP5	number	Coefficient appearing in the hole relaxation time model.	See tables	s
WTPL	number	Coefficient appearing in the hole relaxation time model.	See tables	s
TPL	number	Coefficient appearing in the hole relaxation time model.	See tables	K

Lattice Temperature AAM Parameters for Semiconductors

DENSITY	number	Specific mass density for the material.	See tables	Kg/cm^3
DN.LAT	number	Multiplication factor for the thermal diffusion term in the electron current density relation.	See tables	none
DP.LAT	number	Multiplication factor for the thermal diffusion term in the hole current density relation.	See tables	none
A.SP.HEA	number	First parameter for the specific heat model of the material.	See tables	J/Kg/K
B.SP.HEA	number	Second parameter for the specific heat model of the material.	See tables	$J/Kg/K^2$
C.SP.HEA	number	Third parameter for the specific heat model of the material.	See tables	$J/Kg/K^3$
D.SP.HEA	number	Fourth parameter for the specific heat model of the material.	See tables	((J/Kg)K))
F.SP.HEA	number	Fifth parameter for the specific heat model of the material.	See tables	$J/Kg/K^4$
G.SP.HEA	number	Sixth parameter for the specific heat model of the material.	See tables	$J/Kg/K^5$
A.TH.CON	number	First parameter for the thermal conductivity model of the material.	See tables	(cm-K/W)
B.TH.CON	number	Second parameter for the thermal conductivity model of the material.	See tables	(cm/W)
C.TH.CON	number	Third parameter for the thermal conductivity model of the material.	See tables	(cm/W/K)
D.TH.CON	number	Fourth parameter for the thermal conductivity model of the material. **Units**: $\left(\text{cm/W/K}^{(\texttt{E.TH.CON}-1)}\right)$	See tables	See description
E.TH.CON	number	Fifth parameter for the thermal conductivity model of the material.	See tables	none
OP.PH.EN	number	Mean optical phonon energy used for the impact ionization model depending on lattice temperature.	See tables	eV
LAN300	number	Energy free path for electrons at 300 K, used for the impact ionization model depending on lattice temperature.	See tables	cm
LAP300	number	Energy free path for holes at 300 K, used for the impact ionization model depending on lattice temperature.	See tables	cm

Parameter	Type	Definition	Default	Units
Heterojunction AAM Parameters				
X.MOLE	number	The mole fraction to use for compound materials in the specified region.	See tables	none
X.OTHER	char	The name of an **OTHER** quantity that was originally defined on the **PROFILE** statement that contains a two-dimensional mole fraction distribution to use for compound materials in the specified region.	none	none
EPS.X1	number	Parameter used in the mole fraction dependent expression for permittivity.	See tables	none
EPS.X2	number	Parameter used in the mole fraction dependent expression for permittivity.	See tables	none
NC.0	number	Parameter used in the mole fraction dependent expression for the conduction band density of states.	See tables	none
NC.E	number	Parameter used in the mole fraction dependent expression for the conduction band density of states.	See tables	eV
NV.0	number	Parameter used in the mole fraction dependent expression for the valence band density of states.	See tables	none
NV.E	number	Parameter used in the mole fraction dependent expression for the valence band density of states.	See tables	eV
EM.MODEL	number	The effective mass model to use for the specified material or region: = 0 ==> use **EL.EMAS**, **HO.EMAS** = 1 ==> use model for ternary III-V materials Specifying **EM.MODEL**=1 requires that the Heterojunction Device AAM be enabled.	See tables	none
EG.X0 to **EG.X14**	numbers	The factors multiplying mole fraction raised to various powers in the expressions for energy bandgaps for three electron bands (see Chapter 2).	See tables	eV
AF.X0 to **AF.X5**	number	The factors multiplying mole fraction raised to various powers in the expression for electron affinity (see Chapter 2).	See tables	eV
AF.XL	number	The value of the mole fraction at which Medici will switch from one affinity model to another (see Chapter 2).	See tables	eV
EGALX	number	The value of alpha used in calculating the energy bandgap of band X as a function of temperature.	See tables	eV/Kelvin
EGBEX	number	The value of beta used in calculating the energy bandgap of band X as a function of temperature.	See tables	Kelvins
EGGAX	number	The value of gamma used in calculating the energy bandgap of band X as a function of temperature.	See tables	eV/Kelvins
EGALL	number	The value of alpha used in calculating the energy bandgap of band L as a function of temperature.	See tables	eV/Kelvin
EGBEL	number	The value of beta used in calculating the energy bandgap of band L as a function of temperature.	See tables	Kelvins
EGGAL	number	The value of gamma used in calculating the energy bandgap of band L as a function of temperature.	See tables	eV/Kelvins

Parameter	Type	Definition	Default	Units
MEG	number	Coefficient used in calculating the effective mass of electrons from band Γ.	See tables	none
MEG.X1	number	Coefficient used in calculating the effective mass of electrons from band Γ.	See tables	none
MEX	number	Coefficient used in calculating the effective mass of electrons from band X.	See tables	none
MEX.X1	number	Coefficient used in calculating the effective mass of electrons from band X.	See tables	none
MEL	number	Coefficient used in calculating the effective mass of electrons from band L.	See tables	none
MEL.X1	number	Coefficient used in calculating the effective mass of electrons from band L.	See tables	none
MH0	number	Coefficient used in calculating the effective mass of heavy holes.	See tables	none
MH0.X1	number	Coefficient used in calculating the effective mass of heavy holes.	See tables	none
ML0	number	Coefficient used in calculating the effective mass of light holes.	See tables	none
ML0.X1	number	Coefficient used in calculating the effective mass of light holes.	See tables	none

Insulator Parameters

OXIDE	logical	Specifies that the material parameters are to apply to all regions that were specified as **OXIDE** with **REGION** statements.	false	
NITRIDE	logical	Specifies that the material parameters are to apply to all regions that were specified as **NITRIDE** with **REGION** statements.	false	
SAPPHIRE	logical	Specifies that the material parameters are to apply to all regions that were specified as **SAPPHIRE** with **REGION** statements.	false	
OXYNITRI	logical	Specifies that the material parameters are to apply to all regions that were specified as **OXYNITRI** with **REGION** statements.	false	
INSULATO	logical	Specifies that the material parameters are to apply to all regions that were specified as **INSULATO** with **REGION** statements.	false	
S.OXIDE	logical	Specifies that the material parameters are to apply to all regions that were specified as **S.OXIDE**.	false	
REGION	char	The name(s) of the regions for which the material parameters are to apply. If more than one name is given, the entire group should be surrounded by parentheses and the individual names should be separated with commas.	none	
PERMITTI	number	The relative dielectric permittivity of the material.	See tables	none
AFFINITY	number	The electron affinity for the material.	See tables	volts
EG300	number	The energy bandgap of the material at 300 K.	See tables	eV
ECN.GC	number	The critical electric field for electron scattering in the insulator used in the gate current model.	See tables	volts/cm
BARLN	number	The coefficient for the potential barrier lowering term for electrons used in the gate current model.	See tables	$(\text{volts-cm})^{(1/2)}$

Parameter	Type	Definition	Default	Units
TUNLN	number	The coefficient for the tunneling term for electrons used in the gate current model.	See tables	$(volts\text{-}cm^2)^{(1/3)}$
ECP.GC	number	The critical electric field for hole scattering in the insulator used in the gate current model.	See tables	volts/cm
BARLP	number	The coefficient for the potential barrier lowering term for holes used in the gate current model.	See tables	$(volts\text{-}cm)^{(1/2)}$
TUNLP	number	The coefficient for the tunneling term for holes used in the gate current model.	See tables	$(volts\text{-}cm^2)^{(1/3)}$
ME.DT	number	The effective tunneling mass of electrons used in the direct tunneling model.	See tables	

Lattice Temperature AAM Parameters for Insulators

DENSITY	number	Specific mass density for the material.	See tables	Kg/cm^3
A.SP.HEA	number	First parameter for the specific heat model of the material.	See tables	J/Kg/K
B.SP.HEA	number	Second parameter for the specific heat model of the material.	See tables	$J/Kg/K^2$
C.SP.HEA	number	Third parameter for the specific heat model of the material.	See tables	$J/Kg/K^3$
D.SP.HEA	number	Fourth parameter for the specific heat model of the material.	See tables	((J/Kg)K))
F.SP.HEA	number	Fifth parameter for the specific heat model of the material.	See tables	$J/Kg/K^4$
G.SP.HEA	number	Sixth parameter for the specific heat model of the material.	See tables	$J/Kg/K^5$
A.TH.CON	number	First parameter for the thermal conductivity model of the material.	See tables	(cm-K/W)
B.TH.CON	number	Second parameter for the thermal conductivity model of the material.	See tables	(cm/W)
C.TH.CON	number	Third parameter for the thermal conductivity model of the material.	See tables	(cm/W/K)
D.TH.CON	number	Fourth parameter for the thermal conductivity model of the material. **Units:** $\left(cm/W/K^{(\mathbf{E.TH.CON}-1)}\right)$	See tables	See description
E.TH.CON	number	Fifth parameter for the thermal conductivity model of the material.	See tables	none

Electrode Parameters

ELECTROD	char	The name(s) of the electrodes for which the Fowler-Nordheim parameters are to apply. If more than one name is given, the entire group should be surrounded by parentheses and the individual names should be separated with commas.	none	
A.FN	number	Coefficient of the pre-exponential term for the Fowler-Nordheim tunneling model.	See tables	A/V^2
B.FN	number	Coefficient of the exponential term for the Fowler-Nordheim tunneling model.	See tables	V/cm
ME.DT	number	The effective tunneling mass of electrons used in the direct tunneling model.	See tables	

Parameter	Type	Definition	Default	Units
BARR.DT	number	The barrier height at the electrode/insulator interface used in the direct tunneling model.	See tables	eV
Circuit Analysis AAM Parameters				
STRUCTUR	char	Selects the device in which the material parameters are altered. This parameter is only used with the Circuit Analysis AAM.	all devices	
Optical Device AAM Parameters				
PR.TABLE	logical	Print the wavelength dependent tables for refractive index and absorption coefficient for each region.	false	
Optical Energy Gap				
EGO300	number	The optical energy gap at 300 K.	See tables	eV
EGOALPH	number	The value of alpha used in calculating the temperature-dependent optical energy gap.	See tables	eV/Kelvin
EGOBETA	number	The value of beta used in calculating the temperature-dependent optical energy gap.	See tables	Kelvin
Real Refractive Index				
WAVE.RE	array	The wavelength values for which real refractive index values are specified. At most thirty values may be defined with this parameter.	none	microns
INDEX.RE	array	The values of the real refractive index corresponding to the wavelength values given by **WAVE.RE**. At most 30 values may be defined with this parameter.	none	none
RRI.FILE	char	The name of a formatted file containing real refractive index versus wavelength data. The first column should be the wavelength in microns and the second column should be the real refractive indices. The file must contain at least two lines and the maximum number of lines is 40.	none	
FIRST	logical	Specifies that the minimum wavelength given by **WAVE.RE** or **WAVE.IM**, or the minimum wavelength found in the files specified with **RRI.FILE**, **IRI.FILE**, or **ABS.FILE** are used as the first row in the corresponding data table. Any entries in the relevant table with smaller values of wavelength are removed.	false	
LAST	logical	Specifies that the maximum wavelength given by **WAVE.RE** or **WAVE.IM**, or the maximum wavelength found in the files specified with **RRI.FILE**, **IRI.FILE**, or **ABS.FILE** are used as the last row in the corresponding data table. Any entries in the relevant table with larger values of wavelength are removed.	false	
Imaginary Refractive Index or Absorption Coefficient				
WAVE.IM	array	The wavelength values for which imaginary refractive index values or absorption coefficient values are specified. At most thirty values may be defined with this parameter.	none	microns
INDEX.IM	array	The values of the imaginary refractive index corresponding to the wavelength values given by **WAVE.IM**. At most 30 values may be defined with this parameter.	none	none

Parameter	Type	Definition	Default	Units
ABSORPTI	array	The values of the absorption coefficient corresponding to the wavelength values given by **WAVE.IM**. At most 30 values may be defined with this parameter.	none	cm^{-1}
IRI.FILE	char	The name of a formatted file containing imaginary refractive index versus wavelength data. The first column should be the wavelength in microns and the second column should be the imaginary refractive indices. The file must contain at least two lines and the maximum number of lines is 40.	none	
ABS.FILE	char	The name of a formatted file containing absorption coefficient versus wavelength data. The first column should be the wavelength in microns and the second column should be the absorption coefficient. The file must contain at least two lines and the maximum number of lines is 40.	none	
BTBT.AB	logical	Specifies that an absorption model accounting for band-to-band transitions is used instead of the empirical tables for absorption coefficient.	false	
B.BB	array	An array of coefficients used in the calculation of the band-to-band portion of the absorption coefficient. A maximum of ten values can be specified.	none	transition dependent
E.PHONON	array	An array of phonon energies for indirect transitions in the calculation of the band-to-band portion of the absorption coefficient. A maximum of ten values can be specified.	none	eV
E1.BB	array	An array of energy gaps for indirect transitions used in the calculation of the band-to-band portion of the absorption coefficient. A maximum of ten values can be specified.	none	eV
EXP.BB	array	An array of exponents used in the calculation of the band-to-band portion of the absorption coefficient. A maximum of ten values can be specified.	none	none
BATA.AB	logical	Specifies that an absorption model accounting for band-to-tail transitions is included in the calculation of the absorption coefficient.	false	
E.URBACH	number	The Urbach energy used in the calculation of the band-to-tail portion of the absorption coefficient.	none	eV
E1.BT	number	The ending energy value used in the calculation of the band-to-tail portion of the absorption coefficient.	$E_{g,opt}(T)$	eV
G.BT	number	A parameter used in the calculation of the Urbach tail in the band-to-tail portion of the absorption coefficient. If **E.URBACH** is specified, this parameter is ignored.	none	none
FRCA.AB	logical	Specifies that an absorption model accounting for free carrier transitions is included in the calculation of the absorption coefficient.	false	
EL.EMAS	number	The relative electron effective mass.	See tables	none
HO.EMAS	number	The relative hole effective mass.	See tables	none

Description

The **MATERIAL** statement is used to modify the physical parameters associated with the materials in the device structure. If no **MATERIAL** statement is specified, the default parameter values are used.

See Also...

To further illustrate the **MATERIAL** statement, refer to:

- Input file *mdex2p* in NPN Bipolar Transistor Examples, Chapter 5, "Simulation with Modified Emitter Region" on page 5-12
- Input file *mdex5* in Photogeneration Examples, Chapter 7, "Generation of the Simulation Structure and Solutions" on page 7-2
- Several other examples

Applicable Regions

If a material name is specified (such as **SILICON** or **OXIDE**), then parameters specified on the **MATERIAL** statement will apply to all regions in the device structure that consist of that material. Otherwise, the **REGION** parameter can be used

Parameters from Solution Files

When a solution file is read in using the **LOAD** statement, material parameters stored in the file will replace the corresponding material parameters in the present setup. This makes it unnecessary to re-specify parameters that were modified in a previous simulation when continuing the simulation from a saved solution.

This section provides information and examples on how the **MATERIAL** statement is used with the Optical Device Advanced Application Module (OD-AAM).

Optical Device AAM Parameters

This section provides information and examples on how the **MATERIAL** statement is used with the Optical Device Advanced Application Module (OD-AAM).

Changing Table Values

Complex indices of refraction for various materials are supplied as built-in wavelength dependent tables. These tables can be modified for your own particular needs. For example:

```
MATERIAL  SILICON  WAVE.RE=0.5  INDEX.RE=2.5
```

The above example adds (or replaces) an entry in the real refractive index table for all silicon regions at a wavelength of 0.5 microns. In this case, the real refractive index is set to a value of 2.5. The entries for wavelengths other than 0.5 micron are unchanged from their previous values.

Data for absorption coefficients can be added by specifying either the imaginary refractive index or the absorption coefficient itself. For example:

```
MATERIAL  REGION=Silicon1  WAVE.IM=0.6  ABSORPT=2500
```

The above statement adds (or replaces) an entry in the absorption coefficient table for the region named "Silicon1" at a wavelength of 0.6 microns. In this case, the absorption coefficient is set to a value of 2500 cm^{-1}. The entries for wavelengths other than 0.6 micron are unchanged from their previous values.

It is also possible to specify a range of entries as illustrated in the following example:

```
MATERIAL  GAAS  WAVE.RE=(0.4,0.5,0.6)  INDEX.RE=(4.5,5.1,4.3)
+  WAVE.IM=(0.5,0.6,0.7)  INDEX.IM=(1.8,1.0,0.6)
```

The above statement adds or replaces entries in both the real and imaginary refractive index tables for all GaAs regions.

- For the real refractive index, the table entries in the wavelength range 0.4 microns to 0.6 microns are replaced by the specified values.
- For the imaginary refractive index, the table entries in the wavelength range 0.5 microns to 0.7 microns are replaced by the specified values.

Coefficient Files

In addition to specifying table entries directly as illustrated in the previous examples, it is possible to specify files that contain this information. The parameters **RRI.FILE**, **IRI.FILE**, or **ABS.FILE**. can be used to specify files that contain data for the real refractive index, the imaginary refractive index, or the absorption coefficient, respectively.

If such a data file is used, it should contain two columns as shown in the following example:

```
0.4  3.33
0.5  3.56
0.6  4.66
 ..   ...
 ..   ...
```

- The first column is for wavelength values.
- The second column is for the real or imaginary indices of refraction or the absorption coefficients.

The file should contain at least two entries, but no more than 40.

Physical Absorption Models

The **MATERIAL** statement can be used to specify region dependent physical absorption models and the parameters associated with these models. For example:

```
MATERIAL  REG=Silicon2  BTBT.AB
+         B.BB=(7.5859e3,7.5859e3,7.5859e3,7.5859e3)
+         E.PHONON=(55.3e-3,55.3e-3,55.3e-3,55.3e-3)
+         E1.BB=(3.4,4.4,5.2,5.4)
+         EXP.BB=(2,2,2,2)
```

In this case,

- **BTBT.AB** specifies that the band-to-band absorption mechanism is used for the region named **Silicon2**.
- Nonzero **E.PHONON** values represent indirect transitions.
- Values of 2 for **EXP.BB** are characteristic of *allowed* transitions.
- The range of **E1.BB** values used above allows for a wide spectral response, while most semiconductors have peak absorption coefficients in the range 3.4 to 5.2 eV.

More than one absorption mechanism can be included in the analysis. For example:

```
MATERIAL  A-SI  BTBT.AB  BATA.AB
+         B.BB=4.225E5  E.PHONON=0  E1.BB=0  EXP.BB=2
+         EGO300=2.2  E.URBACH=0.06
```

In this case,

- **BTBT.AB** and **BATA.AB** specify that both band-to-band and band-tail absorption are considered for all amorphous-silicon regions.
- **E.PHONON**=0 indicates a direct transition.
- **EXP.BB**=2 indicates an *allowed* transition.
- **EGO300** is the optical energy gap at 300K.
- Below the optical gap, the absorption coefficient decreases exponentially, with a characteristic energy given by **E.URBACH**.

Default Parameters

The following tables contain the default material parameters available in Medici.

- Table 3-3, "Default Semiconductor Material Parameters," on page 3-268
- Table 3-4, "Default Semiconductor Material Parameters," on page 3-272
- Table 3-5, "Default Insulator Material Parameters," on page 3-273
- Table 3-6, "Default Electrode Region Material Parameters Used by Medici," on page 3-274
- For Parameters not included in the tables see "Other Semiconductor Parameters" on page 3-274.

Table 3-3 Default Semiconductor Material Parameters

Parameter	Units	SILICON	GAAS	POLYSILI	S.OXIDE	GERMANIU
PERMITTI		11.8	13.1	Si	3.9	Si
EPS.X1		0.0	0.0	0.0	0.0	0.0
EPS.X2		0.0	0.0	0.0	0.0	0.0
EG.MODEL		1	1	1	1	1
AFFINITY	V	4.17	4.07	Si	0.97	4

Table 3-3 Default Semiconductor Material Parameters

Parameter	Units	SILICON	GAAS	POLYSILI	S.OXIDE	GERMANIU
AF.X0 to **AF.X5** and **AF.XL**	V	0.0	0.0	Si	0.0	0.0
EG300	eV	1.08	1.424	Si	9.0	0.66
EG.X0 to **EG.X14**	eV	0.0	0.0	Si	0.0	0.0
EGALPH	eV/K	4.73e-4	5.405e-4	Si	Si	4.77e-4
EGBETA	K	636	204	Si	Si	235
EGGAMM	eV/K	0	0	Si	Si	0
EGALX	eV/K	4.73e-4	4.6e-4	0	0	0
EGBEX	K	636	204	0	0	0
EGGAX	eV/K	0	0	0	0	0
EGALL	eV/K	0	6.05e-4	0	0	0
EGBEL	K	0	204	0	0	0
EGGAL	eV/K	0	0	0	0	0
NC300	cm^{-3}	2.8e19	4.7e17	Si	Si	1.04e19
NC.F		1.5	1.5	Si	Si	Si
NC.0		0.0	0.0	0.0	0.0	0.0
NC.E	eV	0.0	0.0	0.0	0.0	0.0
NV300	cm^{-3}	1.04e19	7e18	Si	Si	6e18
NV.F		1.5	1.5	Si	Si	Si
NV.0		0.0	0.0	0.0	0.0	0.0
NV.E	eV	0.0	0.0	0.0	0.0	0.0
GCB		2	2	Si	Si	Si
GVB		4	2	Si	Si	Si
EDB	eV	0.044	0.005	Si	Si	Si
EAB	eV	0.045	0.005	Si	Si	Si
TAUN0	s	1e-7	1e-9	Si	2e-5	Si
NSRHN	cm^{-3}	5e16	1	Si	Si	Si
AN		1	1	1	1	1
BN		1	0	1	1	1
CN		0	0	0	0	0
EN		2	0	2	2	2
TAUP0	s	1e-7	1e-9	Si	2e-5	Si
NSRHP	cm^{-3}	5e16	1	Si	Si	Si
AP		1	1	1	1	1
BP		1	0	1	1	1
CP		0	0	0	0	0
EP		2	0	2	2	2
EXN.TAU		0	0	0	0	0
EXP.TAU		0	0	0	0	0
ETRAP	eV	0	0	0	0	0

Table 3-3 Default Semiconductor Material Parameters

Parameter	Units	SILICON	GAAS	POLYSILI	S.OXIDE	GERMANIU
M.RTUN	m_0	0.25	0.25	Si	Si	Si
B.RTUN	$(cm/V)^{S.RTUN}/cm^3/s$	4.0e14	0	Si	Si	Si
S.RTUN		2.5	2	Si	Si	Si
E.RTUN	V/cm	1.9e7	0	Si	Si	Si
C.DIRECT	cm^3/s	0	0	0	0	0
AUGN	cm^6/s	2.8e-31	0	Si	Si	Si
AUGP	cm^6/s	9.9e-32	0	Si	Si	Si
ARICHN	$A/(K\text{-}cm)^2$	110	6.2857	Si	Si	Si
ARICHP	$A/(K\text{-}cm)^2$	30	105	Si	Si	Si
N0.BGN	cm^{-3}	1e17	1e17	Si	Si	Si
V0.BGN	V	9e-3	0	Si	Si	Si
CON.BGN		0.5	0	Si	Si	Si
A.EHS	V-cm-s	1.35e-20	0	Si	Si	Si
B.EHS	cm^{-3}	3.58e-18	0	Si	Si	Si
C.EHS	cm^{-3}	2.86e-17	0	Si	Si	Si
N.IONIZA	cm^{-1}	7.03e5	2.994e5	Si	7.03e2	1.55e7
N.ION.1	$(cm\text{-}K)^{-1}$	0.0	0.0	0.0	0.0	0.0
N.ION.2	$(cm\text{-}K^2)^{-1}$	0.0	0.0	0.0	0.0	0.0
ECN.II	V/cm	1.231e6	6.848e5	Si	1.11e8	1.56e6
EXN.II		1	1.6	Si	Si	Si
P.IONIZA	cm^{-1}	1.528e6	2.215e5	Si	1.528e3	1.0e7
P.ION.1	$(cm\text{-}K)^{-1}$	0.0	0.0	0.0	0.0	0.0
P.ION.2	$(cm\text{-}K^2)^{-1}$	0.0	0.0	0.0	0.0	0.0
ECP.II	V/cm	2.036e6	6.57e5	Si	1.83e8	1.28e6
EXP.II		1	1.75	Si	Si	Si
LAMHN	cm	9.2e-7	1	Si	Si	Si
LAMRN	cm	61.6e-7	1	Si	Si	Si
LAMHP	cm	1e-7	1	Si	Si	Si
LAMRP	cm	LAMRN	1	Si	Si	Si
A.BTBT	$eV^{1/2}/cm/s/V^2$	3.5e21	0	Si	Si	Si
B.BTBT	$V/cm/eV^{3/2}$	22.5e6	0	Si	Si	Si
A.FN	A/V^2	6.32e-7	0	Si	Si	Si
B.FN	V/cm	2.21e8	0	Si	Si	Si
KAPPA.QM	none	1.0	0.0	0.0	0.0	0.0
KAPPA.N	none	1.0	0.0	0.0	0.0	0.0
KAPPA.P	none	1.0	0.0	0.0	0.0	0.0
N.ACCUM	cm^{-3}	2.85e19	0	0	0	0
P.ACCUM	cm^{-3}	1.2e19	0	0	0	0
DREF.QM	microns	0.0025	0.0025	0.0025	0.0025	0.0025
QM.NORP	none	0	0	0	0	0

Table 3-3 Default Semiconductor Material Parameters

Parameter	Units	SILICON	GAAS	POLYSILI	S.OXIDE	GERMANIU
QM.EFIEL	none	0	0	0	0	0
U.STRESS	eV	10.5	0	Si	Si	0
D.STRESS	eV	1.1	0	Si	Si	0
A.STRESS	eV	2.1	0	Si	Si	0
B.STRESS	eV	-2.33	0	Si	Si	0
C.STRESS	eV	-4.75	0	Si	Si	0
ME.DT		0.35	Si	Si	Si	Si
MHH.DT		0.49	Si	Si	Si	Si
MLH.DT		0.16	Si	Si	Si	Si
ELE.CQ		1	1	1	1	1
ELE.TAUW	s	2e-13	1e-12	Si	Si	Si
WTN0	s	1.6852e-13	2.4e-12	ELE.TAUW	ELE.TAUW	ELE.TAUW
WTN1	s	1.0299e-13	4e-13	0	0	0
WTN2	s	-5.1845e-15	0	0	0	0
WTN3	s	0	0	0	0	0
WTN4	s	0	0	0	0	0
WTN5	s	0	0	0	0	0
WTNL	s	6.8e-13	0	0	0	0
TNL	K	2979.8	1866.27	0	0	0
HOL.CQ		1	1	1	1	1
HOL.TAUW	s	ELE.TAUW	ELE.TAUW	Si	Si	Si
WTP0	s	-1.56e-14	0.0	0.0	0.0	0.0
WTP1	s	1.38e-13	0	0	0	0
WTP2	s	-2.5e-14	0	0	0	0
WTP3	s	2.31e-15	0	0	0	0
WTP4	s	-1.05e-16	0	0	0	0
WTP5	s	1.82e-18	0	0	0	0
WTPL	s	HOL.TAUW	HOL.TAUW	HOL.TAUW	HOL.TAUW	HOL.TAUW
TPL	K	1e5	0	0	0	0
DENSITY	kg/cm^3	2.32e-3	5.3176e-3	Si	2.26e-3	5.33e-3
DN.LAT		1	1	1	1	1
DP.LAT		1	1	1	1	1
A.SP.HEA	J/kg/K	850.9	181.5	737	3000	310
B.SP.HEA	$J/kg/K^2$	152.2e-3	0.8225	442e-3	0	0
C.SP.HEA	$J/kg/K^3$	0	-0.00135	0	0	0
D.SP.HEA	J-K/kg	-158.2e5	0	0	0	0
F.SP.HEA	$J/kg/K^4$	0	7.5e-7	0	0	0
G.SP.HEA	$J/kg/K^5$	0	0	0	0	0
A.TH.CON	cm-K/W	0.03	2.9644	0.437	71.4	1.6667
B.TH.CON	cm/W	1.56e-3	-0.0139255	0	0	0
C.TH.CON	cm/W/K	1.65e-6	5.10475e-5	0	0	0
D.TH.CON	$cm/W/K^{E.TH-1}$	0	-3.8075e-8	0	0	0

Table 3-3 Default Semiconductor Material Parameters

Parameter	Units	SILICON	GAAS	POLYSILI	S.OXIDE	GERMANIU
E.TH.CON		0	0	0	0	0
OP.PH.EN	eV	0.063	0.035	Si	Si	0.037
LAN300	cm	10.4542e-7	3.5272e-6	Si	9.66151e-8	6.88825e-7
LAP300	cm	6.32079e-7	3.6765e-6	Si	5.86026e-8	8.39505e-7
X.MOLE	0	0	0	0	0	0
EGO300	eV	EG300	1.424	Si	Si	Si
EGOALPH	eV/K	EGALPH	5.405e-4	Si	Si	Si
EGOBETA	K	EGBETA	204	Si	Si	Si
B.BB		None	None	None	None	None
E.PHONON	eV	None	None	None	None	None
E1.BB	eV	None	None	None	None	None
EXP.BB		None	None	None	None	None
E1.BT	eV	EG300	Si	Si	Si	Si
E.URBACH	eV	None	None	None	None	None
G.BT		None	None	None	None	None
EL.EMAS		0.31	0.067	Si	Si	Si
HO.EMAS		0.5	0.6915	Si	Si	Si
EM.MODEL		0	0	0	0	0
MEG		0	0.067	0	0	0
MEG.X1		0	0	0	0	0
MEL		0.3282	0.85	0	0	0
MEL.X1		0	0	0	0	0
MEX		0	0.56	0	0	0
MEX.X1		0	0	0	0	0
MH0		0.53	0.62	0	0	0
MH0.X1		0	0	0	0	0
ML0		0.16	0.087	0	0	0
ML0.X1		0	0	0	0	0

Table 3-4 Default Semiconductor Material Parameters

Parameter	Units	SIC	SIGE	ALGAAS
PERMITTI		9.72	Si	GaAs
EG.MODEL		1	2	4
EG300	eV	2.86	Si	GaAs
ECN.II	V/cm	1.77e7	Si	GaAs
ECP.II	V/cm	1.4e7	Si	GaAs
DENSITY	kg/cm^3	3.21e-3	Si	GaAs
OP.PH.EN	eV	0.1042	Si	GaAs
LAN300	cm	1.67425e-7	Si	GaAs
LAP300	cm	2.11673e-7	Si	GaAs
N.IONIZA	cm	2.5e5	Si	GaAs
P.IONIZA	cm	5.18e6	Si	GaAs

Table 3-4 Default Semiconductor Material Parameters

Parameter	Units	SIC	SIGE	ALGAAS
EL.EMAS		0.59	Si	GaAs
HO.EMAS		1	Si	GaAs
AFFINITY	V	3.8	Si	GaAs
NC300	cm^{-3}	1.23e19	Si	GaAs
NV300	cm^{-3}	4.58e18	Si	GaAs
KAPPA.QM		0.0	Si	0.0
KAPPA.N		0.0	Si	0.0
KAPPA.P		0.0	Si	0.0
N.ACCUM	cm^{-3}	0	Si	0
P.ACCUM	cm^{-3}	0	Si	0
DREF.QM	microns	0.0025	Si	0.0025
All Others		Si	Si	GaAs

Table 3-5 Default Insulator Material Parameters

Parameter	Units	OXIDE	NITRIDE	SAPPHIRE	INSULATO	OXYNITRI
PERMITTI		3.9	7.5	12	Oxide	Oxide
AFFINITY	V	0.97	Oxide	Oxide	Oxide	Oxide
EG300	eV	9.0	4.7	4.7	Oxide	Oxide
ECN.GC	V/cm	1.65e5	8.79e4	Oxide	Oxide	Oxide
BARLN	$(V\text{-}cm)^{1/2}$	2.59e-4	Oxide	Oxide	Oxide	Oxide
TUNLN	$(V\text{-}cm^{1/2})^{1/3}$	4.0e-5	Oxide	Oxide	Oxide	Oxide
ECP.GC	V/cm	**ECN.GC**	Oxide	Oxide	Oxide	Oxide
BARLP	$(V\text{-}cm)^{1/2}$	**BARLN**	Oxide	Oxide	Oxide	Oxide
TUNLP	$(V\text{-}cm^{1/2})^{1/3}$	**TUNLN**	Oxide	Oxide	Oxide	Oxide
ME.DT		0.35	Oxide	Oxide	Oxide	Oxide
DENSITY	kg/cm^3	2.26e-3	Oxide	Oxide	Oxide	Oxide
A.SP.HEA	J/kg/K	3000	170	Oxide	Oxide	Oxide
B.SP.HEA	$J/kg/K^2$	0	0	0	Oxide	Oxide
C.SP.HEA	$J/kg/K^3$	0	0	0	Oxide	Oxide
D.SP.HEA	J-K/kg	0	0	0	Oxide	Oxide
F.SP.HEA	$J/kg/K^4$	0	0	0	Oxide	Oxide
G.SP.HEA	$J/kg/K^5$	0	0	0	Oxide	Oxide
A.TH.CON	cm-K/W	71.4	5.4054	Oxide	Oxide	Oxide
B.TH.CON	cm/W	0	0	0	Oxide	Oxide
C.TH.CON	cm/W/K	0	0	0	Oxide	Oxide
D.TH.CON	$cm/W/K^{E.TH-1}$	0	0	0	Oxide	Oxide
E.TH.CON		0	0	0	Oxide	Oxide

Table 3-6 Default Electrode Region Material Parameters Used by Medici

Parameter	Units	ELECTROD
DENSITY	kg/cm^3	2.69e-3
A.SP.HEA	J/kg/K	737.0
B.SP.HEA	$J/kg/K^2$	0.442
C.SP.HEA	$J/kg/K^3$	0
D.SP.HEA	J-K/kg	0
F.SP.HEA	$J/kg/K^4$	
G.SP.HEA	$J/kg//K^5$	0.44
A.TH.CON	cm-K/W	0
B.TH.CON	cm/W	0
C.TH.CON	cm/W/K	0
D.TH.CON	$cm/W/K/^{E.TH-1}$	0
E.TH.CON		0
ME.DT		0.35
BARR.DT	eV	14.15

Other Semiconductor Parameters

For the remaining semiconductor materials:

- Materials **HGCDTE**, **INGAAS**, **INP**, **INAS**, **ZNSE**, **ZNTE** use the same material parameters as **GAAS**.
- Materials **A.SILICO**, **SEMICOND** use the same material parameters as **SILICON**.

Material **DIAMOND** has the same material parameters as **SIC**.

MOBILITY

The **MOBILITY** statement is used to modify parameters associated with the various carrier mobility models available in the program.

```
MOBILITY

   [SILICON] [GAAS] [POLYSILI] [SEMICOND] [SIGE] [ALGAAS]
   [GERMANIU] [SIC] [S.OXIDE] [HGCDTE] [INGAAS] [INP] [INAS]
   [DIAMOND] [ZNSE] [ZNTE] [A-SILICO] [REGION=<c>] [PRINT]
   [ALINAS] [GAASP] [INGAP] [INASP]

 Constant Mobility Parameters
   [MUN0=<n>] [MUP0=<n>]

 Mobility Table Parameters
   [ CONCENTR=<a> [ELECTRON=<a>] [HOLE=<a>] [FIRST] [LAST] ] [PR.TABLE]

 Analytic Mobility Model Parameters
   [MUN.MIN=<n>] [MUN.MAX=<n>] [NREFN=<n>]
   [NUN=<n>] [XIN=<n>] [ALPHAN=<n>]
   [MUP.MIN=<n>] [MUP.MAX=<n>] [NREFP=<n>]
   [NUP=<n>] [XIP=<n>] [ALPHAP=<n>]

 III-V Compound Semiconductor Analytic Mobility Model Parameters
   [MIN.X1=<n>] [MIN.X2=<n>] [MAN.X1=<n>] [MAN.X2=<n>] [NREFN2=<n>]
   [MIP.X1=<n>] [MIP.X2=<n>] [MAP.X1=<n>] [MAP.X2=<n>] [NREFP2=<n>]

 Arora Mobility Model Parameters
   [MUN1.ARO=<n>] [MUN2.ARO=<n>] [CN.ARORA=<n>] [AN.ARORA=<n>]
   [EXN1.ARO=<n>] [EXN2.ARO=<n>] [EXN3.ARO=<n>] [EXN4.ARO=<n>]
   [MUP1.ARO=<n>] [MUP2.ARO=<n>] [CP.ARORA=<n>] [AP.ARORA=<n>]
   [EXP1.ARO=<n>] [EXP2.ARO=<n>] [EXP3.ARO=<n>] [EXP4.ARO=<n>]

 Carrier-Carrier Scattering Mobility Model Parameters
   [A.CCS=<n>] [B.CCS=<n>] [A.LIC=<n>] [B.LIC=<n>]
   [C.LIC=<n>] [EX.LIC=<n>]
   [MUN0.LAT=<n>] [EXN.LAT=<n>] [AN.IIS=<n>] [BN.IIS=<n>]
   [MUP0.LAT=<n>] [EXP.LAT=<n>] [AP.IIS=<n>] [BP.IIS=<n>]

 Philips Unified Mobility Model Parameters
   [MMNN.UM=<n>] [MMXN.UM=<n>] [NRFN.UM=<n>] [ALPN.UM=<n>]
   [TETN.UM=<n>] [NRFD.UM=<n>] [CRFD.UM=<n>]
   [MMNP.UM=<n>] [MMXP.UM=<n>] [NRFP.UM=<n>] [ALPP.UM=<n>]
   [TETP.UM=<n>] [NRFA.UM=<n>] [CRFA.UM=<n>]

 Effective Field Parameters
   [ETAN=<n>]  [ZETAN=<n>] [ETAP=<n>] [ZETAP=<n>]

 Surface Degradation Factors
   [GSURFN=<n>] [GSURFP=<n>]
```

(**MOBILITY** statement continued on next page)

(**MOBILITY** statement continued from previous page)

```
Lombardi Surface Mobility Model Parameters
  [MUN0.LSM=<n>] [MUN1.LSM=<n>] [MUN2.LSM=<n>]
  [CRN.LSM=<n>] [CSN.LSM=<n>]
  [BN.LSM=<n>] [CN.LSM=<n>] [DN.LSM=<n>]
  [EXN1.LSM=<n>] [EXN2.LSM=<n>] [EXN3.LSM=<n>] [EXN4.LSM=<n>]
  [EXN8.LSM=<n>
  [MUP0.LSM=<n>] [MUP1.LSM=<n>] [MUP2.LSM=<n>]
  [CRP.LSM=<n>] [CSP.LSM=<n>]
  [BP.LSM=<n>] [CP.LSM=<n>] [DP.LSM=<n>]
  [EXP1.LSM=<n>] [EXP2.LSM=<n>] [EXP3.LSM=<n>] [EXP4.LSM=<n>] [EXP8.LSM=<n>]
  [PC.LSM=<n>]

Generalized Mobility Curve Model
  [BN.GMC=<n>] [CN.GMC=<n>] [DN.GMC=<n>]
  [D1N.GMC=<n>] [D2N.GMC=<n>]
  [EXN4.GMC=<n>] EXN5.GMC=<n>] EXN6.GMC=<n>] [EXN7.GMC=<n>]
  [EXN8.GMC=<n>]
  [BP.GMC=<n>] [CP.GMC=<n>] [DP.GMC=<n>]
  [D1P.GMC=<n>] [D2P.GMC=<n>]
  [EXP4.GMC=<n>] EXP5.GMC=<n>] EXP6.GMC=<n>] [EXP7.GMC=<n>]
  [EXP8.GMC=<n>]

Shirahata Mobility Model Parameters
  [E1N.SHI=<n>] [EX1N.SHI=<n>] [E2N.SHI=<n>] [EX2N.SHI=<n>]
  [E1P.SHI=<n>] [EX1P.SHI=<n>] [E2P.SHI=<n>] [RX2P.SHI=<n>]

Surface Mobility Model Parameters
  [EREFN=<n>] [EXN.SM=<n>] [MUREFN=<n>]
  [EREFP=<n>] [EXP.SM=<n>] [MUREFP=<n>]

Enhanced Surface Mobility Model Parameters
  [MUN1.SM=<n>] [MUN2.SM=<n>] [MUN3.SM=<n>]
  [EXN1.SM=<n>] [EXN2.SM=<n>] [EXN3.SM=<n>]
  [MUP1.SM=<n>] [MUP2.SM=<n>] [MUP3.SM=<n>]
  [EXP1.SM=<n>] [EXP2.SM=<n>] [EXP3.SM=<n>]

Universal Mobility Model Parameters
  [MUN.UNI=<n>] [ECN.UNI=<n>] [EXN.UNI=<n>]
  [MUP.UNI=<n>] [ECP.UNI=<n>] [EXP.UNI=<n>]

Perpendicular Field Mobility Parameters
  [ECN.MU=<n>] [ECP.MU=<n>]

Hewlett-Packard Mobility Model Parameters
  [MUN0.HP=<n>] [ECN.HP=<n>] [VSN.HP=<n>] [VCN.HP=<n>] [GN.HP=<n>]
  [NRFN.HP=<n>]
  [MUP0.HP=<n>] [ECP.HP=<n>] [VSP.HP=<n>] [VCP.HP=<n>] [GP.HP=<n>]
  [NRFP.HP=<n>]

Field Dependent Mobility Model Parameters
  [VSATN=<n>] [BETAN=<n>] [E0N=<n>]
  [VSATP=<n>] [BETAP=<n>] [E0P=<n>]
  [FLDMOB=<n>]
```

(**MOBILITY** statement continued on next page)

Lucent Mobility Model Parameters
[AN.LUC=<n>] [AP.LUC=<n>] [BN.LUC=<n>] [BP.LUC=<n>]
[CN.LUC=<n>] [CP.LUC=<n>] [DN.LUC=<n>] [DP.LUC=<n>]
[FN.LUC=<n>] [FP.LUC=<n>] [KN.LUC=<n>] [KP.LUC=<n>]
[EXN4.LUC=<n>] [EXP4.LUC=<n>] [EXN9.LUC=<n>] [EXP9.LUC=<n>]

III-V Compound Semiconductor Field Dependent Mobility Model Parameters
[VSN.X1=<n>] [VSN.X2=<n>] [EN.X1=<n>] [EN.X2=<n>]

Tranverse Field Dependent Mobility Model Parameters
[TEMPN.UT=<n>] [PHONN.UT=<n>] [SURFN.UT=<n>] [COULN.UT=<n>]
[TEMPP.UT=<n>] [PHONP.UT=<n>] [SURFP.UT=<n>] [COULP.UT=<n>]
[ACC.N.UT=<n>] [ACC.P.UT=<n>] [INV.N.UT=<n>] [INV.P.UT=<n>]

Stress-Induced Mobility Model Parameters
[MLT.STR=<n>] [MUL0.STR=<n>]

Device Selection (Circuit Analysis AAM)
[STRUCTUR=<c>]

Parameter	Type	Definition	Default	Units
SILICON	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **SILICON** with **REGION** statements.	false	
GAAS	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **GAAS** with **REGION** statements.	false	
POLYSILI	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **POLYSILI** with **REGION** statements.	false	
SEMICOND	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **SEMICOND** with **REGION** statements.	false	
SIGE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **SIGE** with **REGION** statements.	false	
ALGAAS	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **ALGAAS** with **REGION** statements.	false	
GERMANIU	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **GERMANIU** with **REGION** statements.	false	
SIC	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **SIC** with **REGION** statements.	false	
S.OXIDE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **S.OXIDE** with **REGION** statements.	false	
HGCDTE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **HGCDTE** with **REGION** statements.	false	
INGAAS	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **INGAAS** with **REGION** statements.	false	
INP	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **INP** with **REGION** statements.	false	

Parameter	Type	Definition	Default	Units
INAS	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **INAS** with **REGION** statements.	false	
DIAMOND	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **DIAMOND** with **REGION** statements.	false	
ZNSE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **ZNSE** with **REGION** statements.	false	
ZNTE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **ZNTE** with **REGION** statements.	false	
A-SILICO	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **A-SILICO** with **REGION** statements.	false	
REGION	char	The name(s) of the regions for which the mobility parameters are to apply. If more than one name is given, the entire group should be surrounded by parentheses and the individual names should be separated with commas.	none	
PRINT	logical	Specifies that mobility parameters for the specified materials or regions are printed to the standard output.	false	
ALINAS	logical	Specifies that the spacing parameters should be applied to all regions of material AlInAs.	false	
GAASP	logical	Specifies that the spacing parameters should be applied to all regions of material GaAsP.	false	
INGAP	logical	Specifies that the spacing parameters should be applied to all regions of material InGaP.	false	
INASP	logical	Specifies that the spacing parameters should be applied to all regions of material InAsP.	false	
Constant Mobility Parameters				
MUN0	number	The low-field electron mobility. This value is only used when a concentration-dependent mobility model is not selected on the **MODELS** statement.	See Table 3-7	$cm^2/V\text{-}s$
MUP0	number	The low-field hole mobility. This value is only used when a concentration-dependent mobility model is not selected on the **MODELS** statement.	See Table 3-7	$cm^2/V\text{-}s$
Mobility Table Parameters				
CONCENTR	array	The impurity concentration values for which electron and/or hole mobility values are specified. At most 40 values may be defined with this parameter.	none	$\#/cm^3$
ELECTRON	array	The values of electron mobility corresponding to the impurity concentration values given by **CONCENTR**. At most 40 values may be defined with this parameter.	none	$cm^2/V\text{-}s$
HOLE	array	The values of hole mobility corresponding to the impurity concentration values given by **CONCENTR**. At most 40 values may be defined with this parameter.	none	$cm^2/V\text{-}s$

Parameter	Type	Definition	Default	Units
FIRST	logical	Specifies that the minimum impurity concentration given by **CONCENTR** that corresponds to a mobility value specified by **ELECTRON** or **HOLE** is the first row in the mobility table. Any entries in existing mobility tables with smaller values of impurity concentration are removed.	false	
LAST	logical	Specifies that the maximum impurity concentration given by **CONCENTR** that corresponds to a mobility value specified by **ELECTRON** or **HOLE** is the last row in the mobility table. Any entries in existing mobility tables with larger values of impurity concentration are removed.	false	
PR.TABLE	logical	Specifies that the concentration-dependent mobility tables for the specified material or regions is printed to the standard output.	false	
Analytic Mobility Model Parameters			See Table 3-8	
MUN.MIN	number	The minimum electron mobility in the analytic mobility model.		cm^2/V-s
MUN.MAX	number	The maximum electron mobility in the analytic mobility model.		cm^2/V-s
NREFN	number	The reference impurity concentration used in the analytic mobility model for electrons.		#/cm^3
NUN	number	The exponent of normalized temperature used in the numerator of the analytic mobility model for electrons.		none
XIN	number	The exponent of normalized temperature used in the denominator of the analytic mobility model for electrons.		none
ALPHAN	number	The exponent of the ratio of the total impurity concentration to **NREFN** used in the analytic mobility model for electrons.		none
MUP.MIN	number	The minimum hole mobility in the analytic mobility model.		cm^2/V-s
MUP.MAX	number	The maximum hole mobility in the analytic mobility model.		cm^2/V-s
NREFP	number	The reference impurity concentration used in the analytic mobility model for holes.		#/cm^3
NUP	number	The exponent of normalized temperature used in the numerator of the analytic mobility model for holes.		none
XIP	number	The exponent of normalized temperature used in the denominator of the analytic mobility model for holes.		none
ALPHAP	number	The exponent of the ratio of the total impurity concentration to **NREFP** used in the analytic mobility model for holes.		none
III-V Compound Semiconductor Analytic Mobility Model Parameters			See Table 3-9	
MIN.X1	number	The linear term appearing in the expression of $\mu_n^{Min}(X)$ used in the analytic mobility model.		none
MIN.X2	number	The quadratic term appearing in the expression of $\mu_n^{Min}(X)$ used in the analytic mobility model.		none
MAN.X1	number	The linear term appearing in the expression of $\mu_n^{Max}(X)$ used in the analytic mobility model.		none
MAN.X2	number	The quadratic term appearing in the expression of $\mu_n^{Max}(X)$ used in the analytic mobility model.		none

Parameter	Type	Definition	Default	Units
NREFN2	number	The reference impurity concentration used in the quadratic term of the analytic mobility model for electrons.		#/cm3
MIP.X1	number	The linear term appearing in the expression of $\mu_p^{Min}(X)$ used in the analytic mobility model.		none
MIP.X2	number	The quadratic term appearing in the expression of $\mu_p^{Min}(X)$ used in the analytic mobility model.		none
MAP.X1	number	The linear term appearing in the expression of $\mu_p^{Max}(X)$ used in the analytic mobility model.		none
MAP.X2	number	The quadratic term appearing in the expression of $\mu_p^{Max}(X)$ used in the analytic mobility model.		none
NREFP2	number	The reference impurity concentration used in the quadratic term of the analytic mobility model for holes.		#/cm3
Arora Mobility Model Parameters			See Table 3-10	
MUN1.ARO	number	The minimum electron mobility used in the Arora mobility model.		cm^2/V-s
MUN2.ARO	number	The maximum electron mobility used in the Arora mobility model.		cm^2/V-s
CN.ARORA	number	The reference impurity concentration used in the Arora mobility model for electrons.		#/cm^3
AN.ARORA	number	Parameter used in the exponent of normalized impurity concentration in the Arora mobility model for electrons.		none
EXN1.ARO	number	Exponent of normalized temperature used in the Arora mobility model for electrons.		none
EXN2.ARO	number	Exponent of normalized temperature used in the Arora mobility model for electrons.		none
EXN3.ARO	number	Exponent of normalized temperature used in the Arora mobility model for electrons.		none
EXN4.ARO	number	Exponent of normalized temperature used in the Arora mobility model for electrons.		none
MUP1.ARO	number	The minimum hole mobility used in the Arora mobility model.		cm^2/V-s
MUP2.ARO	number	The maximum hole mobility used in the Arora mobility model.		cm^2/V-s
CP.ARORA	number	The reference impurity concentration used in the Arora mobility model for holes.		#/cm^3
AP.ARORA	number	Parameter used in the exponent of normalized impurity concentration in the Arora mobility model for holes.		none
EXP1.ARO	number	Exponent of normalized temperature used in the Arora mobility model for holes.		none
EXP2.ARO	number	Exponent of normalized temperature used in the Arora mobility model for holes.		none
EXP3.ARO	number	Exponent of normalized temperature used in the Arora mobility model for holes.		none
EXP4.ARO	number	Exponent of normalized temperature used in the Arora mobility model for holes.		none

Parameter	Type	Definition	Default	Units
Carrier-Carrier Scattering Mobility Model Parameters			See Table 3-11	
A.CCS	number	Parameter used in the carrier-carrier scattering term of the mobility model **CCSMOB**.		1/cm-V-s
B.CCS	number	Parameter used in the carrier-carrier scattering term of the mobility model **CCSMOB**.		$1/cm^2$
A.LIC	number	Parameter used in the mobility model **CCSMOB**.		none
B.LIC	number	Parameter used in the mobility model **CCSMOB**.		none
C.LIC	number	Parameter used in the mobility model **CCSMOB**.		none
EX.LIC	number	Exponent used in the mobility model **CCSMOB**.		none
MUN0.LAT	number	The room temperature mobility for electrons used in the lattice scattering term of the mobility model **CCSMOB**.		$cm^2/V-s$
EXN.LAT	number	The exponent used in the lattice scattering term for electrons in the mobility model **CCSMOB**.		none
AN.IIS	number	Parameter used in the ionized impurity scattering term for electrons of the mobility model **CCSMOB**.		1/cm-V-s
BN.IIS	number	Parameter used in the ionized impurity scattering term for electrons of the mobility model **CCSMOB**.		$1/cm^3$
MUP0.LAT	number	The room temperature mobility for holes used in the lattice scattering term of the mobility model **CCSMOB**.		$cm^2/V-s$
EXP.LAT	number	The exponent used in the lattice scattering term for holes in the mobility model **CCSMOB**.		none
AP.IIS	number	Parameter used in the ionized impurity scattering term for holes of the mobility model **CCSMOB**.		1/cm-V-s
BP.IIS	number	Parameter used in the ionized impurity scattering term for holes of the mobility model **CCSMOB**.		$1/cm^3$
Philips Unified Mobility Model Parameters			See Table 3-12	
MMNN.UM	number	Mobility parameter used in the Philips Unified mobility model to determine electron mobility at high dopant and/or carrier levels.		$cm^2/V-s$
MMXN.UM	number	Maximum electron mobility in the Philips Unified mobility model.		$cm^2/V-s$
NRFN.UM	number	Reference impurity concentration for electron mobility, used with the Philips Unified mobility model.		$\#/cm^3$
ALPN.UM	number	Exponent used for electron mobility in the Philips Unified mobility model.		none
TETN.UM	number	Exponent used for temperature dependence of lattice scattering for electrons in the Philips Unified mobility model.		none
NRFD.UM	number	Reference impurity concentration for donors to model ultra-high doping effects in the Philips Unified mobility model.		$\#/cm^3$
CRFD.UM	number	Factor determining the ultra-high doping effects for donors in the Philips Unified mobility model.		none

Parameter	Type	Definition	Default	Units
MMNP.UM	number	Mobility parameter used in the Philips Unified mobility model to determine hole mobility at high dopant and/or carrier levels.		cm^2/V-s
MMXP.UM	number	Maximum hole mobility in the Philips Unified mobility model.		cm^2/V-s
NRFP.UM	number	Reference impurity concentration for hole mobility, used with the Philips Unified mobility model.		#/cm^3
ALPP.UM	number	Exponent used for hole mobility in the Philips Unified mobility model.		none
TETP.UM	number	Exponent used for temperature dependence of lattice scattering for holes in the Philips Unified mobility model.		none
NRFA.UM	number	Reference impurity concentration for acceptors to model ultra-high doping effects in the Philips Unified mobility model.		#/cm^3
CRFA.UM	number	Factor determining the ultra-high doping effects for acceptors in the Philips Unified mobility model.		none
Effective Field Parameters			See Table 3-13	
ETAN	number	A factor used in determining the effective electric field at interfaces used in the field-dependent mobility models for electrons.		none
ZETAN	number	A factor used in determining the effective electric field at interfaces used in the field-dependent mobility models for electrons.		none
ETAP	number	A factor used in determining the effective electric field at interfaces used in field-dependent mobility models for holes.		none
ZETAP	number	A factor used in determining the effective electric field at interfaces used in field-dependent mobility models for holes.		none
Surface Degradation Factors			See Table 3-14	
GSURFN	number	The low-field surface reduction factor for electron mobility.		none
GSURFP	number	The low-field surface reduction factor for hole mobility.		none
Lombardi Surface Mobility Model Parameters			See Table 3-15	
MUN0.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for electrons.		cm^2/V-s
MUN1.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for electrons.		cm^2/V-s
MUN2.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for electrons.		cm^2/V-s
CRN.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for electrons.		#/cm^3
CSN.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for electrons.		#/cm^3
BN.LSM	number	Mobility parameter used in the acoustic term of the Lombardi surface mobility model for electrons.		cm/s
CN.LSM	number	Mobility parameter used in the acoustic term of the Lombardi surface mobility model for electrons.		(K-cm/s)(V/cm)$^{-2/3}$

Parameter	Type	Definition	Default	Units
DN.LSM	number	Mobility parameter used in the acoustic term of the Lombardi surface mobility model for electrons.		$(cm^2/V\text{-}s)(V/cm)^{EXN8.LSM}$
EXN1.LSM	number	Exponent used in the bulk term of the Lombardi surface mobility model for electrons.		none
EXN2.LSM	number	Exponent used in the bulk term of the Lombardi surface mobility model for electrons.		none
EXN3.LSM	number	Exponent used in the bulk term of the Lombardi surface mobility model for electrons.		none
EXN4.LSM	number	Exponent used in the acoustic term of the Lombardi surface mobility model for electrons.		none
EXN8.LSM	number	Exponent used in surface roughness term of the Lombardi surface mobility model for electrons.		none
MUP0.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for holes.		$cm^2/V\text{-}s$
MUP1.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for holes.		$cm^2/V\text{-}s$
MUP2.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for holes.		$cm^2/V\text{-}s$
CRP.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for holes.		$\#/cm^3$
CSP.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for holes.		$\#/cm^3$
BP.LSM	number	Mobility parameter used in the acoustic term of the Lombardi surface mobility model for holes.		cm/s
CP.LSM	number	Mobility parameter used in the acoustic term of the Lombardi surface mobility model for holes.		$(K\text{-}cm/s)(V/cm)^{-2/3}$
DP.LSM	number	Mobility parameter used in the acoustic term of the Lombardi surface mobility model for holes.		$(cm^2/V\text{-}s)(V/cm)^{EXP8.LSM}$
EXP1.LSM	number	Exponent used in the bulk term of the Lombardi surface mobility model for holes.		none
EXP2.LSM	number	Exponent used in the bulk term of the Lombardi surface mobility model for holes.		none
EXP3.LSM	number	Exponent used in the bulk term of the Lombardi surface mobility model for holes.		none
EXP4.LSM	number	Exponent used in the acoustic term of the Lombardi surface mobility model for holes.		none
EXP8.LSM	number	Exponent used in surface roughness term of the Lombardi surface mobility model for holes.		none
PC.LSM	number	Mobility parameter used in the bulk term of the Lombardi surface mobility model for holes.		$\#/cm^3$

Generalized Mobility Curve Model Parameters

See Table 3-16

Parameter	Type	Definition	Default	Units
BN.GMC	number	Mobility parameter used in the acoustic term of the Generalized Mobility curve model for electrons.		cm/s
CN.GMC	number	Mobility parameter used in the acoustic term of the Generalized Mobility curve model for electrons.		$(K\text{-}cm/s)(V/cm)^{-2/3}$
DN.GMC	number	Mobility parameter used in the acoustic term of the Generalized Mobility curve model for electrons.		$(cm^2/V\text{-}s)(V/cm)^{\text{EXN8.GMC}}$
D1N.GMC	number	Mobility parameter used in the "screened" term of the Generalized Mobility curve model for electrons.		
D2N.GMC	number	Mobility parameter used in the "unscreened" term of the Generalized Mobility curve model for electrons.		
EXN4.GMC	number	Exponent used in the Generalized Mobility curve model for electrons.		none
EXN5.GMC	number	Exponent used in the Generalized Mobility curve model for electrons.		none
EXN6.GMC	number	Exponent used in the Generalized Mobility curve model for electrons.		none
EXN7.GMC	number	Exponent used in the Generalized Mobility curve model for electrons.		none
EXN8.GMC	number	Exponent used in the Generalized Mobility curve model for electrons.		none
BP.GMC	number	Mobility parameter used in the acoustic term of the Generalized Mobility curve model for holes.		cm/s
CP.GMC	number	Mobility parameter used in the acoustic term of the Generalized Mobility curve model for holes.		$(K\text{-}cm/s)(V/cm)^{-2/3}$
DP.GMC	number	Mobility parameter used in the acoustic term of the Generalized Mobility curve model for holes.		$(cm^2/V\text{-}s)(V/cm)^{\text{EXP8.GMC}}$
D1P.GMC	number	Mobility parameter used in the "screened" term of the Generalized Mobility curve model for holes.		
D2P.GMC	number	Mobility parameter used in the "unscreened" term of the Generalized Mobility curve model for holes.		
EXP4.GMC	number	Exponent used in the Generalized Mobility curve model for holes.		none
EXP5.GMC	number	Exponent used in the Generalized Mobility curve model for holes.		none
EXP6.GMC	number	Exponent used in the Generalized Mobility curve model for holes.		none
EXP7.GMC	number	Exponent used in the Generalized Mobility curve model for holes.		none
EXP8.GMC	number	Exponent used in the Generalized Mobility curve model for holes.		none
Shirahata Mobility Model Parameters			See Table 3-17	
E1N.SHI	number	The reference electric field in the weak surface field dependent term of Shirahata mobility model for electrons.		
EX1N.SHI	number	The exponent in the weak surface field dependent term of Shirahara mobility model for electrons.		
E2N.SHI	number	The reference electric field in the strong surface field dependent term of Shirahata mobility model for electrons.		

Parameter	Type	Definition	Default	Units
EX2N.SHI	number	The exponent in the strong surface field dependent term of Shirahara mobility model for electrons.		
E1P.SHI	number	The reference electric field in the weak surface field dependent term of Shirahata mobility model for holes.		
EX1P.SHI	number	The exponent in the weak surface field dependent term of Shirahara mobility model for holes.		
E2P.SHI	number	The reference electric field in the strong surface field dependent term of Shirahata mobility model for holes.		
EX2P.SHI	number	The exponent in the strong surface field dependent term of Shirahara mobility model for holes.		
Surface Mobility Model Parameters			See Table 3-18	
EREFN	number	The reference electric field used in the surface mobility model for electrons.		V/cm
EXN.SM	number	The exponent used in the surface mobility model for electrons.		none
MUREFN	number	The reference mobility used in the surface mobility model for electrons.		cm^2/V-s
EREFP	number	The reference electric field used in the surface mobility model for holes.		V/cm
EXP.SM	number	The exponent used in the surface mobility model for holes.		none
MUREFP	number	The reference mobility used in the surface mobility model for holes.		cm^2/V-s
Enhanced Surface Mobility Model Parameters			See Table 3-19	
MUN1.SM	number	The first reference mobility used in the enhanced surface mobility model for electrons.		cm^2/V-s
MUN2.SM	number	The second reference mobility used in the enhanced surface mobility model for electrons.		cm^2/V-s
MUN3.SM	number	The third reference mobility used in the enhanced surface mobility model for electrons.		cm^2/V-s
EXN1.SM	number	The first exponent used in the enhanced surface mobility model for electrons.		none
EXN2.SM	number	The second exponent used in the enhanced surface mobility model for electrons.		none
EXN3.SM	number	The third exponent used in the enhanced surface mobility model for electrons.		none
MUP1.SM	number	The first reference mobility used in the enhanced surface mobility model for holes.		cm^2/V-s
MUP2.SM	number	The second reference mobility used in the enhanced surface mobility model for holes.		cm^2/V-s
MUP3.SM	number	The third reference mobility used in the enhanced surface mobility model for holes.		cm^2/V-s
EXP1.SM	number	The first exponent used in the enhanced surface mobility model for holes.		none

Parameter	Type	Definition	Default	Units
EXP2.SM	number	The second exponent used in the enhanced surface mobility model for holes.		none
EXP3.SM	number	The third exponent used in the enhanced surface mobility model for holes.		none
Universal Mobility Model Parameters			See Table 3-20	
MUN.UNI	number	The low field mobility for the Universal Mobility model for electrons.		$cm^2/V\text{-}s$
ECN.UNI	number	Critical field for the Universal Mobility model for electrons.		V/cm
EXN.UNI	number	Exponent of field for the Universal Mobility model for electrons.		none
MUP.UNI	number	The low field mobility for the Universal Mobility model for holes.		$cm^2/V\text{-}s$
ECP.UNI	number	Critical field for the Universal Mobility model for holes.		V/cm
EXP.UNI	number	Exponent of field for the Universal Mobility model for holes.		none
Perpendicular Field Mobility Parameters			See Table 3-21	
ECN.MU	number	The critical electric field used in the perpendicular electric field mobility model for electrons.		V/cm
ECP.MU	number	The critical electric field used in the perpendicular electric field mobility model for holes.		V/cm
Hewlett-Packard Mobility Model Parameters			See Table 3-22	
MUN0.HP	number	Parameter used in the transverse field dependent portion of the Hewlett-Packard mobility model for electrons.		$cm^2/V\text{-}s$
ECN.HP	number	Critical electric field used in the transverse field dependent portion of the Hewlett-Packard mobility model for electrons.		V/cm
VSN.HP	number	Saturation velocity used in the Hewlett-Packard mobility model for electrons.		cm/s
VCN.HP	number	Velocity parameter used in the Hewlett-Packard mobility model for electrons.		cm/s
GN.HP	number	Parameter used in the Hewlett-Packard mobility model for electrons.		none
NRFN.HP	number	Reference concentration used in the Hewlett-Packard mobility model for electrons.		$\#/cm^3$
MUP0.HP	number	Parameter used in the transverse field dependent portion of the Hewlett-Packard mobility model for holes.		$cm^2/V\text{-}s$
ECP.HP	number	Critical electric field used in the transverse field dependent portion of the Hewlett-Packard mobility model for holes.		V/cm
VSP.HP	number	Saturation velocity used in the Hewlett-Packard mobility model for holes.		cm/s
VCP.HP	number	Velocity parameter used in the Hewlett-Packard mobility model for holes.		cm/s
GP.HP	number	Parameter used in the Hewlett-Packard mobility model for holes.		none
NRFP.HP	number	Reference concentration used in the Hewlett-Packard mobility model for holes.		$\#/cm^3$

Parameter	Type	Definition	Default	Units
Field-Dependent Mobility Model Parameters			See Table 3-23	
VSATN	number	The electron saturation velocity used in the calculation of field dependent mobilities.		cm/s
BETAN	number	The exponent used in the field-dependent mobility model for electrons in silicon.		none
E0N	number	The critical electric field used in the field-dependent mobility model for electrons in gallium arsenide.		V/cm
VSATP	number	The hole saturation velocity used in the calculation of field dependent mobilities.		cm/s
BETAP	number	The exponent used in the field-dependent mobility model for holes in silicon.		none
E0P	number	The critical electric field used in the field-dependent mobility model for holes in gallium arsenide.		V/cm
FLDMOB	number	Selects a parallel field-dependent mobility model. A value of 1 selects the Caughey-Thomas expression. A value of 2 selects a GaAs-like mobility model that produces a negative slope in the velocity vs. field curve for high electric field. A value of 3 selects a model that is similar to the Caughey-Thomas expression (see "Alternative Parallel Field-Dependent Expression" on page 2-37). Any other value disables the parallel field-dependent mobility model for the specified materials or regions.		none
Lucent Mobility Model Parameters			See Table 3-24	
AN.LUC	number	The constant part of the exponent used in the surface roughness term of the Lucent mobility model for electrons.		none
AP.LUC	number	The constant part of the exponent used in the surface roughness term of the Lucent mobility model for holes.		none
BN.LUC	number	Mobility parameter used in the acoustic term of the Lucent mobility model for electrons.		cm/s
BP.LUC	number	Mobility parameter used in the acoustic term of the Lucent mobility model for holes.		cm/s
CN.LUC	number	Mobility parameter used in the acoustic term of the Lucent mobility model for electrons.		$(cm^2/V\text{-}s)(V/cm)^{1/3} cm^{3\,\mathbf{EXN4.LUC}}$
CP.LUC	number	Mobility parameter used in the acoustic term of the Lucent mobility model for holes.		$(cm^2/V\text{-}s)(V/cm)^{1/3} cm^{3\,\mathbf{EXNP.LUC}}$
DN.LUC	number	Mobility parameter used in the acoustic term of the Lucent mobility model for electrons.		$(cm^2/V\text{-}s)(V/cm)^{\gamma_n}$
DP.LUC	number	Mobility parameter used in the acoustic term of the Lucent mobility model for holes.		$(cm^2/V\text{-}s)(V/cm)^{\gamma_p}$
FN.LUC	number	The factor multiplying total carrier concentration in the exponent used in the surface roughness term of the Lucent mobility model for electrons.		$cm^{3(1-\mathbf{EXN9.LUC})}$

Parameter	Type	Definition	Default	Units
FP.LUC	number	The factor multiplying total carrier concentration in the exponent used in the surface roughness term of the Lucent mobility model for holes.		$cm^{3(1-\text{EXP9.LUC})}$
KN.LUC	number	The exponent of temperature used in the acoustic term of the Lucent mobility model for electrons.		none
KP.LUC	number	The exponent of temperature used in the acoustic term of the Lucent mobility model for holes.		none
EXN4.LUC	number	The exponent of total impurity concentration used in the acoustic term of the Lucent mobility model for electrons.		none
EXP4.LUC	number	The exponent of total impurity concentration used in the acoustic term of the Lucent mobility model for holes.		none
EXN9.LUC	number	The exponent of total impurity concentration used in the surface roughness term of the Lucent mobility model for electrons.		none
EXP9.LUC	number	The exponent of total impurity concentration used in the surface roughness term of the Lucent mobility model for holes.		none
III-V Compound Semiconductor Field-Dependent Mobility Model Parameters			See Table 3-25	
VSN.X1	number	Linear term used to calculate the mole fraction dependent electron saturation velocity, $v_n^{sat}(X)$.		none
VSN.X2	number	Quadratic term used to calculate the mole fraction dependent electron saturation velocity, $v_n^{sat}(X)$.		none
EN.X1	number	Linear term used to calculate the critical field for electrons, $E_0(X)$.		none
EN.X2	number	Quadratic term used to calculate the critical field for electrons, $E_0(X)$.		none
Transverse Field-Dependent Mobility Model Parameters			See Table 3-26	
TEMPN.UT	number	The prefactor used in the temperature term of the phonon mobility for electrons in silicon.		cm/(V*s)
PHONN.UT	number	The prefactor used in the field term of the phonon mobility for electrons in silicon.		V*s/cm
SURFN.UT	number	The prefactor used in the surface roughness mobility term for electrons in silicon.		$cm^3/(V^3*s)$
COULN.UT	number	The prefactor used in the Coulomb mobility term for electrons in silicon.		1/(cm*V*s)
TEMPP.UT	number	The prefactor used in the temperature term of the phonon mobility for holes in silicon.		cm/(V*s)
PHONP.UT	number	The prefactor used in the field term of the phonon mobility for holes in silicon.		V*s/cm
SURFP.UT	number	The prefactor used in the surface roughness mobility term for holes in silicon.		$cm^3/(V^3*s)$
COULP.UT	number	The prefactor used in the Coulomb mobility term for holes in silicon.		1/(cm*V*s)
ACC.N.UT	number	Surface mobility reduction factor for accumulation layers. Applies only to the semiconductor/insulator interface for electrons in n-type silicon (i.e. source and drain of an N-channel MOSFET).		none

Parameter	Type	Definition	Default	Units
ACC.P.UT	number	Surface mobility reduction factor for accumulation layers. Applies only to the semiconductor/insulator interface for holes in p-type silicon (i.e. source and drain of a P-channel MOSFET).		none
INV.N.UT	number	Surface mobility reduction factor for inversion layers. Applies only to the semiconductor/insulator interface for electrons in p-type silicon (i.e. channel of an N-channel MOSFET) at weak inversion.		none
INV.P.UT	number	Surface mobility reduction factor for inversion layers. Applies only to the semiconductor/insulator interface for holes in n-type silicon (i.e. channel of a P-channel MOSFET) at weak inversion.		none
Stress-Induced Mobility				
MLT.STR	number	Ratio of the longitudinal to transverse effective mass in the ellipsoidal conduction band minima in silicon.		
MUL0.STR	number	Ratio of the unstressed light hole mobility to the total hole mobility in silicon.		
Device Selection (Circuit Analysis AAM)				
STRUCTUR	char	Selects the device where the mobility parameters is modified. This parameter is only used with the Circuit Analysis AAM.	all devices	

Description

The **MOBILITY** statement is used to modify parameters associated with the various carrier mobility models which are available in the program. The actual selection of one or more mobility models is accomplished on the **MODELS** statement.

By default, the specified mobility parameters apply to all semiconductor regions of the device. However, the mobility parameters can be requested to apply to only particular semiconductor regions by using the **REGION** parameter.

See Also...

To further illustrate the **MOBILITY** statement, refer to input files:

- *mdex2p* in NPN Bipolar Transistor Examples, Chapter 5, "Simulation with Modified Emitter Region" on page 5-12
- *mdex17* in Heterojunction Examples, Chapter 14, "Material and Mobility Parameters" on page 14-8

Impurity Dependent Mobility Tables

A table of values is used to define the dependence of electron and hole mobility on impurity concentration for the semiconductor. Interpolation is used to obtain values for impurity concentrations between the values in the table.

During the interpolation, the impurity concentration is assumed to vary exponentially and the mobility is assumed to vary linearly. For impurity concentrations outside of the range of values present in the table, the last value of mobility at the appropriate end of the table is used.

Default mobility tables for both silicon and gallium arsenide are given in Chapter 2, Mobility Models on page 2-16. These tables may be modified by using the parameters **CONCENTR**, **ELECTRON**, and **HOLE**.

If a table entry for a single impurity concentration is given, this entry is inserted directly into existing tables of values. If two or more entries are given, then all entries of the existing tables that have impurity concentration values that fall within the range of values specified with **CONCENTR** are replaced by the new entries. For example, the statement

```
MOBILITY  CONCENTR=(1E15,  1E16,  1E17)
+         ELECTRON=(1305., 1000., 700.)
```

causes all entries for existing electron mobility tables with impurity concentration values within the range 1e15 through 1e17 to be replaced by the three values given above. There may be at most 60 rows in a table and each row corresponds to one impurity concentration.

Mobility Parameters from Solution Files

Solution files that are read in using the **LOAD** statement replace the values of the mobility material parameters in the present setup, with the values of the corresponding mobility parameters stored in the file. This makes it unnecessary to respecify parameters that were modified in a previous simulation when continuing the simulation from a saved solution.

Default Mobility Parameters

The tables below list default values for the various mobility parameters for each material. In these tables, the following notations are used in the column headings:

SL "Silicon-like" materials. Parameters for these materials currently have the same values as parameters for **SILICON** (unless otherwise noted). Materials included in this group include **POLYSILI**, **SEMICOND**, **SIGE**, **GERMANIU**, **SIC**, **DIAMOND**, **A-SILICO**, and **S.OXIDE**.

GL "Gallium arsenide-like" materials. Parameters for these materials currently have the same default values as parameters for **GAAS** (unless otherwise noted). Materials in this group include: **HGCDTE**, **INP**, **INAS**, **ZNSE**, and **ZNTE**.

Note:

When appropriate default parameters for materials defined as* GAAS *are unknown,* SILICON *values are often used instead.

Table 3-7 Constant Mobility Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	S.OXIDE	SIC	SL	GL
MUN0	1e3	8.5e3	9.89e3	2.73e4	2.4e4	2e2	2e2	2.4e4	20	300	Si	GaAs
MUP0	5e2	4e2	4e2	4.8e2	4.8e2	1.5e2	1.5e2	4.8e2	2e-5	50	Si	GaAs

Table 3-8 Analytic Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	S.OXIDE	SL	GL
MUN.MIN	55.24	0.0	2.37e3	4e3	4.97e2	9.5e1	9.5e1	4.97e2	1e1	Si	GaAs
MUN.MAX	1429.23	8.5e3	9.89e3	2.73e4	2.41e4	2e2	2e2	2.41e4	2e1	Si	GaAs
NREFN	1.07e17	1.69e17	3.63e17	3.63e17	1e17	3.63e17	1e17	1e17	Si	Si	GaAs
NUN	-2.3	-1.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
XIN	-3.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
ALPHAN	0.733	4.36e-1	1.0	1.0	1.0	1.0	1.0	1.0	Si	Si	GaAs
MUP.MIN	49.705	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1e-5	Si	GaAs
MUP.MAX	479.37	4e2	4e2	4.8e2	4.8e2	1.5e2	1.5e2	4.8e2	1e-5	Si	GaAs
NREFP	1.61e17	2.75e17	2.75e17	1e30	1e30	1e30	1e30	1e30	Si	Si	GaAs
NUP	-2.2	-2.1	0.0	0.0	0.0	0.0	0.0	-2.2	Si	Si	GaAs
XIP	-3.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
ALPHAP	0.70	3.95e-1	3.95e-1	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs

Table 3-9 III-V Compound Semiconductor Analytic Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
MIN.X1	0.0	0.0	-9.83e1	-2.23e1	-9.2e-1	2.39e1	1.76e1	2.6	Si	GaAs
MIN.X2	0.0	0.0	0.0	-1.86e-1	0.0	0.0	0.0	-4.5e2	Si	GaAs
MAN.X1	0.0	0.0	-9.5e-1	-1.76	-9.7e-1	4.85e1	2.14e1	-9.31e-1	Si	GaAs
MAN.X2	0.0	0.0	0.0	1.12	0.0	0.0	0.0	1.16e-1	Si	GaAs
NREFN2	1e30	1e30	1.75e18	1.75e18	2.51e17	1.75e18	2.51e17	2.51e17	Si	GaAs
MIP.X1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MIP.X2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MAP.X1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MAP.X2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
NREFP2	1e30	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs

Table 3-10 Arora Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	S.OXIDE	SL	GL
MUN1.ARO	88.0	8.5e3	9.89e3	2.73e4	2.41e4	2e2	2e2	2.41e4	1e1	Si	GaAs
MUN2.ARO	1252.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2e1	Si	GaAs
CN.ARORA	1.26e17	1.26e17	1e20	1e20	1e20	1e20	1e20	1e20	Si	Si	GaAs
AN.ARORA	0.88	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
EXN1.ARO	-0.57	-5.7e-1	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
EXN2.ARO	-2.33	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
EXN3.ARO	2.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
EXN4.ARO	-0.146	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
MUP1.ARO	54.3	4e2	4e2	4.8e2	4.8e2	1.5e2	1.5e2	4.8e2	1e-5	Si	GaAs
MUP2.ARO	407.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2e-5	Si	GaAs

Table 3-10 Arora Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	S.OXIDE	SL	GL
CP.ARORA	2.35e17	2.35e17	1e20	1e20	1e20	1e20	1e20	1e20	Si	Si	GaAs
AP.ARORA	0.88	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
EXP1.ARO	-0.57	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
EXP2.ARO	-2.23	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
EXP3.ARO	2.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs
EXP4.ARO	-0.146	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	Si	GaAs

Table 3-11 Carrier-Carrier Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
A.CCS	1.04e21	1.04e21	1e22	1e22	1e22	1e22	1e22	1e22	Si	GaAs
B.CCS	7.45e13	7.45e13	1e22	1e22	1e22	1e22	1e22	1e22	Si	GaAs
A.LIC	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
B.LIC	2.126	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
C.LIC	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EX.LIC	0.715	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUN0.LAT	1430	8.50e3	9.89e3	2.73e4	2.414e4	2e2	2e2	2.414e4	Si	GaAs
EXN.LAT	2.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
AN.IIS	2.4e21	2.4e21	1e22	1e22	1e22	1e22	1e22	1e22	Si	GaAs
BN.IIS	1.37e20	1.37e20	1e22	1e22	1e22	1e22	1e22	1e22	Si	GaAs
MUP0.LAT	495	4e2	4e2	4.8e2	4.8e2	1.5e2	1.5e2	4.8e2	Si	GaAs
EXP.LAT	2.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
AP.IIS	5.2e20	5.2e20	1e22	1e22	1e22	1e22	1e22	1e22	Si	GaAs
BP.IIS	5.63e19	5.63e19	1e22	1e22	1e22	1e22	1e22	1e22	Si	GaAs

Table 3-12 Philips Unified Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
MMNN.UM	5.22e1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MMXN.UM	1.417e3	8.5e3	9.892e3	2.725e4	2.414e4	2e2	2e2	2.414e4	Si	GaAs
NRFN.UM	9.68e16	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
ALPN.UM	6.8e-1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
TETN.UM	2.285	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
NRFD.UM	4e20	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
CRFD.UM	2.1e-1	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
MMNP.UM	4.49e1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MMXP.UM	4.705e2	4e2	4e2	4e2	4.8e2	1.5e2	1.5e2	4.8e2	Si	GaAs
NRFP.UM	2.23e17	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
ALPP.UM	7.19-1	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
TETP.UM	2.247	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
NRFA.UM	7.2e20	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
CRFA.UM	5e-1	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs

Table 3-13 Effective Field Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
ETAN	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
ZETAN	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
ETAP	0.333	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
ZETAP	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs

Table 3-14 Surface Degradation Factors

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
GSURFN	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
GSURFP	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs

Table 3-15 Lombardi Surface Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
MUN0.LSM	52.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUN1.LSM	43.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUN2.LSM	1417.0	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
CRN.LSM	9.68e16	9.68e16	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
CSN.LSM	3.43e20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
BN.LSM	4.75e7	1e10	1e10	1e10	1e10	1e10	1e10	1e10	Si	GaAs
CN.LSM	1.74e5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
DN.LSM	5.82e14	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
EXN1.LSM	0.680	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN2.LSM	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN3.LSM	2.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN4.LSM	0.125	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN8.LSM	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUP0.LSM	44.9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUP1.LSM	29.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUP2.LSM	470.5	1.0	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
CRP.LSM	2.23e17	2.23e17	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
CSP.LSM	6.1e20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
BP.LSM	9.93e6	1e10	1e10	1e10	1e10	1e10	1e10	1e10	Si	GaAs
CP.LSM	8.84e5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
DP.LSM	2.05e14	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
EXP1.LSM	0.719	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP2.LSM	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP3.LSM	2.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP4.LSM	0.0317	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP8.LSM	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs

Table 3-15 Lombardi Surface Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
PC.LSM	9.23e16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs

Table 3-16 Generalized Mobility Curve Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
BN.GMC	8.95e5	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
CN.GMC	3.23e6	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
DN.GMC	8.29e14	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
D1N.GMC	1.35e11	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
D2N.GMC	2.89e8	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
EXN4.GMC	2.84e-2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN5.GMC	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN6.GMC	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN7.GMC	3.5e-1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN8.GMC	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
BP.GMC	8.95e5	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
CP.GMC	3.23e6	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
DP.GMC	8.29e14	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
D1P.GMC	1.35e11	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
D2P.GMC	2.89e8	1e20	1e20	1e20	1e20	1e20	1e20	1e20	Si	GaAs
EXP4.GMC	2.84e-2	1e20	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP5.GMC	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP6.GMC	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP7.GMC	3.5e-1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP8.GMC	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs

Table 3-17 Shirahata Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
E1N.SHI	6.3e3	6.3e3	6.3e3	6.3e3	6.3e3	6.3e3	6.3e3	6.3e3	Si	GaAs
EX1N.SHI	0.28	0.28	0.28	0.0	0.0	0.0	0.0	0.0	Si	GaAs
E2N.SHI	7.7e5	7.7e5	7.7e5	7.7e5	7.7e5	7.7e5	7.7e5	7.7e5	Si	GaAs
EX2N.SHI	2.9	2.9	2.9	0.0	0.0	0.0	0.0	0.0	Si	GaAs
E1P.SHI	6.3e3	6.3e3	6.3e3	6.3e3	6.3e3	6.3e3	6.3e3	6.3e3	Si	GaAs
EX1P.SHI	0.28	0.28	0.28	0.0	0.0	0.0	0.0	0.0	Si	GaAs
E2P.SHI	7.7e5	7.7e5	7.7e5	7.7e5	7.7e5	7.7e5	7.7e5	7.7e5	Si	GaAs
EX2P.SHI	2.9	2.9	2.9	0.0	0.0	0.0	0.0	0.0	Si	GaAs

Table 3-18 Surface Mobility Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
EREFN	1e6	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
EXN.SM	0.285	2.85e-1	2.85e-1	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUREFN	389.0	2e3	2e3	2.73e4	2.73e4	2e2Surf	2e2	2.414e4	Si	GaAs
EREFP	1e6	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
EXP.SM	0.315	3.5e-1	3.15-1	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUREFP	84.5	6.76e1	6.76e1	4.8e2	4.8e2	1.5e2	1.5e2	4.8e2	Si	GaAs

Table 3-19 Enhanced Surface Mobility Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
MUN1.SM	481.0	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
MUN2.SM	591.0	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
MUN3.SM	1270.0	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
EXN1.SM	0.160	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN2.SM	2.170	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXN3.SM	1.070	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUP1.SM	92.8	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
MUP2.SM	124.0	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
MUP3.SM	534.0	1e6	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
EXP1.SM	0.296	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP2.SM	1.620	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
EXP3.SM	1.020	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs

Table 3-20 Universal Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
MUN.UNI	7.835e2	8.5e3	9.892e3	2.725e4	2.414e4	2e2	2e2	2.414e4	Si	GaAs
ECN.UNI	2.57e5	1e10	1e10	1e10	1e10	1e10	1e10	1e10	Si	GaAs
EXN.UNI	1.02	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUP.UNI	2.474e2	4e2	4e2	4.8e2	4.8e2	1.5e2	1.5e2	4.8e2	Si	GaAs
ECP.UNI	5.75e5	1e10	1e10	1e10	1e10	1e10	1e10	1e10	Si	GaAs
EXP.UNI	0.95	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs

Table 3-21 Perpendicular Field Mobility Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
ECN.MU	6.49e4	1e8	1e8	1e8	1e8	1e8	1e8	1e8	Si	GaAs
ECP.MU	1.87e4	1e8	1e8	1e8	1e8	1e8	1e8	1e8	Si	GaAs

Table 3-22 Hewlett-Packard Mobility Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
MUN0.HP	774.0	8.5e3	9.89e3	2.73e4	2.41e4	2e2	2e2	2e2	Si	GaAs
ECN.HP	5.5e5	1e8	1e8	1e8	1e8	1e8	1e8	1e8	Si	GaAs
VSN.HP	1.036e7	1e10	1e10	1e10	1e10	1e10	1e10	1e10	Si	GaAs
VCN.HP	4.9e6	1e10	1e10	1e10	1e10	1e10	1e10	1e10	Si	GaAs
GN.HP	8.8	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
NRFN.HP	5e17	5e17	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
MUP0.HP	250	4e2	4e2	4.8e2	4.8e2	1.5e2	1.5e2	4.8e2	Si	GaAs
ECP.HP	2.78e5	1e8	1e8	1e8	1e8	1e8	1e8	1e8	Si	GaAs
VSP.HP	1.2e17	1e10	1e10	1e10	1e10	1e10	1e10	1e10	Si	GaAs
VCP.HP	2.928e6	1e10	1e10	1e10	1e10	1e10	1e10	1e10	Si	GaAs
GP.HP	1.6	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
NRFP.HP	5e17	5e17	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs

Table 3-23 Field-Dependent Mobility Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
VSATN	1.035e7	6.5e6	6.35e6	1e7	1e7	6.92e6	6.92e6	1e7	Si	GaAs
BETAN	2.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
E0N	4e3	4e3	5.418e3	6e3	6e3	1.63e4	1.63e4	6e3	Si	GaAs
VSATP	1.035e7	7.7e6	6.352e6	1e7	1e7	6.92e6	6.92e6	1e7	Si	GaAs
BETAP	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
E0P	4e3	4e3	1e6	1e6	1e6	1e6	1e6	1e6	Si	GaAs
FLDMOB	1.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	Si	GaAs

Table 3-24 Lucent Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
AN.LUC	2.58	2.58	2.58	2.58	2.58	2.58	2.58	2.58	Si	GaAs
AP.LUC	2.18	2.18	2.18	2.18	2.18	2.18	2.18	2.18	Si	GaAs
BN.LUC	3.61e7	3.61e7	3.61e7	3.61e7	3.61e7	3.61e7	3.61e7	3.61e7	Si	GaAs
BP.LUC	1.51e7	1.51e7	1.51e7	1.51e7	1.51e7	1.51e7	1.51e7	1.51e7	Si	GaAs
CN.LUC	1.70e4	1.70e4	1.70e4	1.70e4	1.70e4	1.70e4	1.70e4	1.70e4	Si	GaAs
CP.LUC	4.18e3	4.18e3	4.18e3	4.18e3	4.18e3	4.18e3	4.18e3	4.18e3	Si	GaAs
DN.LUC	3.58e18	3.58e18	3.58e18	3.58e18	3.58e18	3.58e18	3.58e18	3.58e18	Si	GaAs
DP.LUC	4.10e15	4.10e15	4.10e15	4.10e15	4.10e15	4.10e15	4.10e15	4.10e15	Si	GaAs
FN.LUC	6.85e-21	6.85e-21	6.85e-21	6.85e-21	6.85e-21	6.85e-21	6.85e-21	6.85e-21	Si	GaAs
FP.LUC	7.82e-21	7.82e-21	7.82e-21	7.82e-21	7.82e-21	7.82e-21	7.82e-21	7.82e-21	Si	GaAs
KN.LUC	1.7	1.7	1.7	1.7	1.7	1.7	1.7	1.7	Si	GaAs
KP.LUC	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	Si	GaAs
EXN4.LUC	0.0233	0.0233	0.0233	0.0233	0.0233	0.0233	0.0233	0.0233	Si	GaAs

Table 3-24 Lucent Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
EXP4.LUC	0.0119	0.0119	0.0119	0.0119	0.0119	0.0119	0.0119	0.0119	Si	GaAs
EXN9.LUC	0.0767	0.0767	0.0767	0.0767	0.0767	0.0767	0.0767	0.0767	Si	GaAs
EXP9.LUC	0.123	0.123	0.123	0.123	0.123	0.123	0.123	0.123	Si	GaAs

Table 3-25 III-V Compound Semiconductor Field-Dependent Mobility Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
VSN.X1	0.0	0.0	-5.304e-1	-1.019	-7.493e-1	-8.212e-2	-3.41-2	-3.32e-1	Si	GaAs
VSN.X2	0.0	0.0	-748e-2	-7.09e-1	0.0	0.0	0.0	0.0	Si	GaAs
EN.X1	0.0	0.0	-2.471	-7.956e-1	4.169	-6.677e-1	1.533	5.883	Si	GaAs
EN.X2	0.0	0.0	7.194	6.3e-1	0.0	0.0	0.0	0.0	Si	GaAs

Table 3-26 Transverse Field-Dependent Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
TEMPN.UT	1.15e3	8.5e3	9.892e3	2.725e3	2.414e4	2e2	2e2	2.414e4	Si	GaAs
PHONN.UT	3.20e-9	1e-20	1e-20	1e-20	1e-20	1e-20	1e-20	1e-20	Si	GaAs
SURFN.UT	6e14	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
COULN.UT	1.1e21	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
TEMPP.UT	2.7e2	4e2	4e2	4.8e2	4.8e2	1.5e2	1.5e2	4.8e2	Si	GaAs
PHONP.UT	2.35e-9	1e-20	1e-20	1e-20	1e-20	1e-20	1e-20	1e-20	Si	GaAs
SURFP.UT	1e8	1e8	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
COULP.UT	1.4e18	1e30	1e30	1e30	1e30	1e30	1e30	1e30	Si	GaAs
ACC.N.UT	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
ACC.P.UT	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
INV.N.UT	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs
INV.P.UT	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	Si	GaAs

Table 3-27 Stress-Induced Mobility Model Parameters

Parameter	SILICON	GAAS	ALGAAS	INGAAS	ALINAS	GAASP	INGAP	INASP	SL	GL
MLT.STR	5.158	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs
MUL0.STR	2.79	0.0	0.0	0.0	0.0	0.0	0.0	0.0	Si	GaAs

IMPURITY

The **IMPURITY** statement defines the physical parameters associated with impurities used in the structure.

```
IMPURITY

   NAME=<c> [PRINT]

 Material or Region Name
   [ {  SILICON | GAAS | POLYSILI | SEMICOND | SIGE | ALGAAS
      | GERMANIU | SIC | S.OXIDE | HGCDTE | INGAAS | INP | INAS
      | DIAMOND | ZNSE | ZNTE | A-SILICO | ALINAS | GAASP | INGAP
      | INASP | REGION=<c>
     }

 Incomplete Ionization Parameters
     [GB=<n>] [EB0=<n>] [ALPHA=<n>] [BETA=<n>] [GAMMA=<n>]
     [HDT.MIN=<n>] [HDT.MAX=<n>]
   ]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]
```

Parameter	Type	Definition	Default	Units
NAME	char	The name of the impurity.	none	
PRINT	logical	Specifies that the impurity parameters for the structure are printed to the standard output.	false	
Material or Region Name				
SILICON	logical	Specifies that the impurity parameters apply to all regions that were specified as **SILICON**.	false	
GAAS	logical	Specifies that the impurity parameters apply to all regions that were specified as **GAAS**.	false	
POLYSILI	logical	Specifies that the impurity parameters apply to all regions that were specified as **POLYSILI**.	false	
SEMICOND	logical	Specifies that the impurity parameters apply to all regions that were specified as **SEMICOND**.	false	
SIGE	logical	Specifies that the impurity parameters apply to all regions that were specified as **SIGE**.	false	
ALGAAS	logical	Specifies that the impurity parameters apply to all regions that were specified as **ALGAAS**.	false	
GERMANIU	logical	Specifies that the impurity parameters apply to all regions that were specified as **GERMANIU**.	false	
SIC	logical	Specifies that the impurity parameters apply to all regions that were specified as **SIC**.	false	

Parameter	Type	Definition	Default	Units
S.OXIDE	logical	Specifies that the impurity parameters apply to all regions that were specified as **S.OXIDE**.	false	
HGCDTE	logical	Specifies that the impurity parameters apply to all regions that were specified as **HGCDTE**.	false	
INGAAS	logical	Specifies that the impurity parameters apply to all regions that were specified as **INGAAS**.	false	
INP	logical	Specifies that the impurity parameters apply to all regions that were specified as **INP**.	false	
INAS	logical	Specifies that the impurity parameters apply to all regions that were specified as **INAS**.	false	
DIAMOND	logical	Specifies that the impurity parameters apply to all regions that were specified as **DIAMOND**.	false	
ZNSE	logical	Specifies that the impurity parameters apply to all regions that were specified as **ZNSE**.	false	
ZNTE	logical	Specifies that the impurity parameters apply to all regions that were specified as **ZNTE**.	false	
A-SILICO	logical	Specifies that the impurity parameters apply to all regions that were specified as **A-SILICO**.	false	
ALINAS	logical	Specifies that the impurity parameters apply to all regions that were specified as **ALINAS**.	false	
GAASP	logical	Specifies that the impurity parameters apply to all regions that were specified as **GAASP**.	false	
INGAP	logical	Specifies that the impurity parameters apply to all regions that were specified as **INGAP**.	false	
INASP	logical	Specifies that the impurity parameters apply to all regions that were specified as **INASP**.	false	
REGION	char	The name(s) of the regions for which the impurity parameters apply. If more than one name is given, the entire group should be surrounded by parentheses and the individual names should be separated with commas.	none	

Incomplete Ionization Parameters

GB	number	The band degeneracy factor.	See tables	none
EB0	number	The constant term used in the calculation of the band ionization energy.	See tables	eV
ALPHA	number	The prefactor for the doping dependent term used in the calculation of the band ionization energy.	See tables	eV/cm
BETA	number	The prefactor the temperature dependent term used in the calculation of the band ionization energy.	See tables	none
GAMMA	number	The exponent of temperature used in the calculation of the band ionization energy.	See tables	none
HDT.MIN	number	The minimum impurity concentration for which the high doping transition from incomplete ionization to complete ionization applies.	See tables	$1/cm^3$

Parameter	Type	Definition	Default	Units
HDT.MAX	number	The maximum impurity concentration for which the high doping transition from incomplete ionization to complete ionization applies.	See tables	$1/cm^3$

Circuit Analysis AAM Parameters

STRUCTUR	char	Selects the device in which the material parameters are altered. This parameter is only used with the Circuit Analysis AAM.	all devices	

Description

The **IMPURITY** statement defines the physical parameters associated with impurities used in the structure. The recognized impurity names and the default parameter values associated with them are shown in the tables below.

Table 3-28 Impurity Parameters in Silicon and "Silicon-Like" Materials

Parameter	N-Type	P-Type	B	P	As	Sb	In	Al	Ga
Type	Donor	Acceptor	Acceptor	Donor	Donor	Donor	Acceptor	Acceptor	Acceptor
GB	2.0	4.0	4.0	2.0	2.0	2.0	4.0	4.0	4.0
EB0	0.044	0.045	0.045	0.045	0.054	0.039	0.160	0.067	0.072
ALPHA	3.100e-8	3.037e-8	3.037e-8	3.100e-8	3.100e-8	3.100e-8	3.037e-8	3.037e-8	3.037e-8
BETA	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0	200.0
GAMMA	1.000	0.950	0.950	1.000	1.000	1.000	0.950	0.950	0.950
HDT.MIN	1.0e18	1.0e18	1.0e18	1.0e18	1.0e18	1.0e18	1.0e18	1.0e18	1.0e18
HDT.MAX	1.0e19	1.0e19	1.0e19	1.0e19	1.0e19	1.0e19	1.0e19	1.0e19	1.0e19

Table 3-29 Impurity Parameters in GaAs and "GaAs-Like" Materials

Parameter	N-Type	P-Type
Type	Donor	Acceptor
GB	2.0	2.0
EB0	0.005	0.005
ALPHA	3.100e-8	3.037e-8
BETA	200.0	200.0
GAMMA	1.000	0.950
HDT.MIN	1.0e18	1.0e18
HDT.MAX	1.0e19	1.0e19

CONTACT

The **CONTACT** statement defines the physical parameters associated with an electrode. Lumped resistances and capacitances, distributed contact resistance, and current boundary conditions are also specified with the **CONTACT** statement.

```
CONTACT

  {NAME=<c> | ALL} [PRINT]
  [ {  NEUTRAL | ALUMINUM | P.POLYSI | N.POLYSI | MOLYBDEN | TUNGSTEN
     | MO.DISIL | TU.DISIL | WORKFUNC=<n>
    }
    [PIN]
  ]

The following set may NOT presently be used with the Circuit Analysis
AAM if the terminal is attached to the circuit:
  [ {  CURRENT
     | CON.RESI=<n>
     | ( CHARGE [CAPACITA=<n>] [V.CAPAC=<n>] )
     | ( RESISTAN=<n>] [CAPACITA=<n>] [INDUCTAN=<n>] )
     | ( SURF.REC [VSURFN=<n>] [VSURFP=<n>] [ BARRIERL [ALPHA=<n>] ] )
     | VOLTAGE
    }
  ]

Lattice Temperature AAM PArameters
  [ R.THERMA=<n>] [C.THERMA=<n>] ]

Circuit Analysis AAM Parameters
  [STRUCTUR=<c>]

Optical Device AAM Parameters
  [ {TRANSELE | REFLECTI=<n>} ]
```

Parameter	Type	Definition	Default	Units
NAME	char	The name of a previously defined electrode for which the specified properties apply. The same properties can be assigned to more than one electrode by separating them with commas, and enclosing the entire set in parentheses. **synonym: NUMBER**	none	
ALL	logical	Specifies that the same properties apply to all electrodes.	false	
PRINT	logical	Specifies that the contact parameters for the structure are printed to the standard output.	false	
NEUTRAL	logical	Specifies that the work function potential for the electrode is calculated from the doping.	true	
ALUMINUM	logical	Specifies that the work function potential of aluminum (4.10 volts) is used for the electrode.	false	

Parameter	Type	Definition	Default	Units
P.POLYSI	logical	Specifies that the work function potential of p+ polysilicon (5.25 volts) is used for the electrode.	false	
N.POLYSI	logical	Specifies that the work function potential of n+ polysilicon (4.17 volts) is used for the electrode.	false	
MOLYBDEN	logical	Specifies that the work function potential of molybdenum (4.53 volts) is used for the electrode.	false	
TUNGSTEN	logical	Specifies that the work function potential of tungsten (4.63 volts) is used for the electrode.	false	
MO.DISIL	logical	Specifies that the work function potential of molybdenum disilicide (4.80 volts) is used for the electrode.	false	
TU.DISIL	logical	Specifies that the work function potential of tungsten disilicide (4.80 volts) is used for the electrode.	false	
WORKFUNC	number	The value of the work function potential assigned to the specified electrode.	none	volts
PIN	logical	Specifies that a work function value that is less than the electron affinity, χ, is pinned to χ, and that a work function value that is greater than $\chi + E_g$ is pinned to $\chi + E_g$. If **^PIN** is specified, the program accepts any non-negative value for work function.	true	

The following set may NOT presently be used with the Circuit Analysis AAM if the terminal is attached to the circuit

CURRENT	logical	Specifies that a current boundary condition is used at this contact.	false	
CON.RESI	number	The value used for the distributed contact resistance of the electrode. The specified value should represent the resistance of the contact in Ohms multiplied by the contact area in cm^2.	0.0	Ohm-cm^2
CHARGE	logical	Specifies that a charge boundary condition is used at this contact. This parameter is only used with the Programmable Device AAM.	false	
CAPACITA	number	The value of a lumped capacitance attached to the contact. The specified value should represent the true capacitance in Farads divided by the device width (direction perpendicular to the simulated cross-section) in microns.	0.0	Farad/micron
V.CAPAC	number	The value of voltage applied to the terminal of the capacitor which is *not* attached to the device. This parameter is used only with the Programmable Device AAM.	0.0	volts
RESISTAN	number	The value of a lumped resistance attached to the contact. The specified value should represent the true resistance in Ohms multiplied by the device width (direction perpendicular to the simulated cross-section) in microns.	0.0	Ohm-microns
INDUCTAN	number	The value of a lumped inductance attached to the contact. The specified value should represent the true inductance in Henrys multiplied by the device width (direction perpendicular to the simulated cross-section) in microns.	0.0	Henry-microns

Parameter	Type	Definition	Default	Units
SURF.REC	logical	Specifies that finite surface recombination velocities are used at the specified electrode. **synonym: SCHOTTKY**	false	
VSURFN	number	The surface recombination velocity for electrons when **SURF.REC** is specified.	**ARICHN***TEMPERAT^2 */q/Nc(T)*	cm/s
VSURFP	number	The surface recombination velocity for holes when **SURF.REC** is specified.	**ARICHP***TEMPERAT^2 */q/Nv(T)*	cm/s
BARRIERL	logical	Specifies that the barrier lowering mechanism is used.	false	
ALPHA	number	The linear dipole barrier lowering coefficient.	0.0	cm
VOLTAGE	logical	Specifies that voltage boundary conditions are used at the contact. This parameter acts as a reset button for the electrode, removing any special boundary conditions that were previously specified.	false	
Lattice Temperature AAM Parameters				
R.THERMA	number	The value of the lumped thermal resistance which is placed in series with the thermal electrode.	0.0	K*micron/W
C.THERMA	number	The value of the lumped thermal capacitance which is connected to the thermal electrode.	none	J/K/micron
Circuit Analysis AAM Parameters				
STRUCTUR	character	Selects which device the **CONTACT** statement effects. This parameter is only used with the Circuit Analysis AAM.	all devices	
Optical Device AAM Parameters				
TRANSELE	logical	Specifies that the designated electrode is optically transparent during ray tracing analysis. This parameter has no effect on the electrode's electrical properties.	false	
REFLECTI	number	The reflectivity of optical power at the electrode/semiconductor or electrode/insulator interface.	1.0	none

Description

The **CONTACT** statement defines the physical parameters associated with an electrode. The **CONTACT** statement also specifies the following:

- Lumped resistances, capacitances, and inductances
- Distributed contact resistance
- Current boundary conditions

See Also...

To further illustrate the **CONTACT** statement, refer to:

- *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Potential Regrid" on page 4-7
- *mdex3* in Diode and Lumped Element Examples, Chapter 6, "Lumped Resistance Solution" on page 6-14
- Several other examples

Note:

The Newton solution method is required when using resistive, capacitive, or inductive elements, contact resistance, current boundary conditions, or charge boundary conditions.

Work Function Potentials

Work function potentials can be assigned to electrodes by specifying the electrode material, or alternatively, by indicating the desired value using the **WORKFUNC** parameter. If no **CONTACT** statement is supplied for an electrode, it is assumed to be **NEUTRAL**.

Cautions

CAUTION

This section details areas in which the CONTACT statement should be used with caution, to ensure accurate results.

Contacts to P-type Regions

Care should be taken when specifying a contact material to a p-type region in Medici. If **ALUMINUM** (or material with a similar work function potential) is specified, severe band bending occurs at the contact, causing conduction to be effectively blocked. This is because of the large difference between the metal work function potential (approximately 4.10v) and the p-type semiconductor work function potential (approximately 5.25v).

It is recommended that either **NEUTRAL** contacts or **P.POLYSI** contacts be made to p-type materials in Medici.

Insulator Contacts

Care should be taken to ensure that all insulator contacts have a work function potential specified. Failure to do so results in, for example, unexpected threshold voltage shifts for MOS devices.

N^+ Poly Gate MOS Simulations

In the case of MOS simulations involving heavily-doped n^+ polysilicon gate structures, the material of the polycrystalline gate is known to behave in a way similar to that of a conductor.

For simulations involving n^+ polysilicon gate structures it is recommended that the work function value for the contact be set to 4.35V:

```
CONTACT NAME=Gate WORKFUNC=4.35
```

Boundary Conditions

The **CONTACT** statement can also be used to specify special boundary conditions at the contact.

- **CURRENT** indicates that current boundary conditions are used at the contact. The actual value of current to use is specified at solve time on the **SOLVE** statement.
- A value of **CON.RESI** may be specified to take into account the finite resistivity of contacts to the semiconductor.
- Lumped resistances and/or capacitances and/or inductances between applied biases and semiconductor device contacts can be specified by assigning values to the parameters **RESISTAN**, **CAPACITA**, and/or **INDUCTAN** respectively.

  Note that the specified resistance, capacitance and inductance are connected in parallel.
- If the electrode is a thermal electrode, then lumped thermal resistance and capacitance can be specified using the **R.THERMA** and **C.THERMA** parameters.

  The temperature at the device contact T_d is given by

$$T_d = T_0 + P*\mathbf{R.THERMA} - \mathbf{R.THERMA}*\mathbf{C.THERMA}\frac{dT_d}{dt}$$

where:

- T_0 is the applied contact temperature
- P is the power flowing into the contact

Schottky Contacts

Schottky contacts can be specified by indicating that finite surface recombination velocities are to be used with the **SURF.REC** parameter. Default values for the recombination velocities are calculated using the following:

- Effective Richardson constant for electrons or holes
- Effective density of states in the conduction or valence bands if values are not specified here

The Schottky model also takes into account field-dependent barrier-lowering mechanisms if **BARRIERL** is specified. The coefficient of the linear dipole term, **ALPHA**, may optionally be specified.

INTERFACE

The **INTERFACE** statement allows the specification of interface parameters (recombination velocities and fixed and trapped charges) at any interface in the structure. It also allows fixed charges to be placed inside insulator regions.

```
INTERFACE

   [ {MATERIAL=<c> | REGION=<c>} ]
   [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
   {  ( [S.N=<n>] [S.P=<n>] [QF=<n>]
        [N.ACCEPT=<n>] [P.ACCEPT=<n>] [N.DONOR=<n>] [P.DONOR=<n>]
      )
    |
      ( Q.INSULA=<n> [D.CHAR=<n>] )
   }
   [CLEAR] [ALL.CLEA]
```

Parameter	Type	Definition	Default	Units
MATERIAL	char	The material names of two adjacent materials. The interface parameters specified with this statement applies to all interfaces between these two materials. The two material names should be enclosed in parentheses and separated by a comma.	none	
REGION	char	The region names of two adjacent regions. The interface parameters specified with this statement applies to all interfaces between these two regions. The two region names should be enclosed in parentheses and separated by a comma.	none	
X.MIN	number	The minimum x coordinate of the bounding box that encloses the portion of the interface that the parameters specified on this statement apply to.	The minimum x location in the device structure.	microns
X.MAX	number	The maximum x coordinate of the bounding box that encloses the portion of the interface that the parameters specified on this statement apply to.	The maximum x location in the device structure.	microns
Y.MIN	number	The minimum y coordinate of the bounding box that encloses the portion of the interface that the parameters specified on this statement apply to.	The minimum y location in the device structure.	microns
Y.MAX	number	The maximum y coordinate of the bounding box that encloses the portion of the interface that the parameters specified on this statement apply to.	The maximum y location in the device structure.	microns
S.N	number	The electron surface recombination velocity.	0.0	cm/s
S.P	number	The hole surface recombination velocity.	0.0	cm/s
QF	number	The interface fixed charge density.	0.0	cm^{-2}
N.ACCEPT	number	The interface trapped charge density for the electron acceptors.	0.0	cm^{-2}/eV
P.ACCEPT	number	The interface trapped charge density for the hole acceptors.	0.0	cm^{-2}/eV
N.DONOR	number	The interface trapped charge density for the electron donors.	0.0	cm^{-2}/eV

Parameter	Type	Definition	Default	Units
P.DONOR	number	The interface trapped charge density for the hole donors.	0.0	cm^{-2}/eV
Q.INSULA	number	The insulator fixed charge density. **synonyms: QINS**	0.0	cm^{-3}
D.CHAR	number	Parameter that describes the placement of insulator fixed charged within the insulator. The value of **D.CHAR** has the following meaning: =0 : **Q.INSULA** is assigned to all the insulator nodes inside the bounding box. >0 : **Q.INSULA** is assigned to all the insulator nodes inside the bounding box at a distance less than **D.CHAR** from the interface. <0 : The charge density assigned to the insulator nodes inside the bounding box is given by **Q.INSULA***exp(-(d/**D.CHAR**)^2) where d is the distance from the interface.	0.0	microns
CLEAR	logical	Specifies that the value of all interface parameters for all interfaces enclosed within the bounding box are cleared (set to zero) prior to processing any new parameters specified on this statement. If **^CLEAR** is specified, then the values of parameters specified on this statement are added to any previous values specified for interfaces enclosed within the bounding box.	true	
ALL.CLEA	logical	Specifies that all preceding interface statements or statements read as part of a mesh or solution are to be completely removed.	false	

Description

The **INTERFACE** statement allows the specification of the following:

- Recombination velocities at interfaces
- Fixed and trapped charges at interfaces
- Fixed charges inside insulator regions

See Also...

To further illustrate the **INTERFACE** statement, refer to input files:

- *mdex1* in N-Channel MOSFET Examples, Chapter 4, "Generation of the Simulation Structure" on page 4-2
- *mdex1f* in N-Channel MOSFET Examples, Chapter 4, "Analysis Including Fast Interface States" on page 4-19

Interface Selection

Interface parameters can be defined at interfaces between two materials or two regions. Specific interfaces can be selected by using the **MATERIAL** or **REGION** parameters. If neither **MATERIAL** nor **REGION** are specified, interfaces between semiconductor and insulator regions will be used.

Fixed and trapped charge and surface recombination velocities are defined at interfaces found within a box bounded by the device coordinates **X.MIN**, **X.MAX**, **Y.MIN**, **Y.MAX**. The default for the bounding box is the entire device.

Insulator Charge

If **Q.INSULA** is specified, charges will be placed at insulator nodes within the bounding box. The exact placement is dependent on the value of **D.CHAR** (see the description of the **D.CHAR** parameter).

Saving Interface Parameters in Files

Parameters specified on the **INTERFACE** statement are saved in solution files, but not mesh files. An exception is the mesh files in Circuit Analysis AAM.

To use the interface parameters in continued simulations without respecifying them, it is necessary to obtain and save a solution at some point after the **INTERFACE** statement is specified. When the saved solution is loaded during a continued simulation, the previously specified **INTERFACE** parameters are used automatically.

For the Circuit Analysis AAM, the **INTERFACE**, **MODEL**, **MATERIAL**, and **MOBILITY** data can be written to and read from mesh files as well as solution files.

To save this data in a mesh file, the **W.MODELS** parameter should be specified on a **SAVE** statement. Only data that has been specified prior to the **SAVE** statement in the input file is saved.

ANISOTROPIC

The **ANISOTROPIC** statement allows you to specify anisotropic components for various physical models associated with materials.

This statement is also used to specify advanced band structure parameters that account for non-parabolicity and multiple bands in the calculation of the density of states.

```
ANISOTROPIC

   [PRINT]

 Semiconductor Parameters
   {  ( {  SILICON | GAAS | POLYSILI | SEMICOND | SIGE | ALGAAS
         | GERMANIU | SIC | S.OXIDE | HGCDTE | INGAAS | INP | INAS
         | DIAMOND | ZNSE | ZNTE | A-SILICO | REGION=<c>
        }

 Anisotropic Component Factors
        [PERMITTI=<a>] [MU.N=<a>] [MU.P=<a>] [II.N=<a>] [II.P=<a>]
        [TH.COND=<a>]

 General Anisotropic Electron Impact Ionization
        [ ANIIN [N.ION.O=<a>] [N.ION.1=<a>] [N.ION.2=<a>] [ECN.II=<a>]
           [EXN.II=<a>]
        ]

 General Anisotropic Hole Impact Ionization
        [ ANIIP [P.ION.O=<a>] [P.ION.1=<a>] [P.ION.2=<a>] [ECP.II=<a>]
           [EXP.II=<a>]
        ]

 General Anisotropic Thermal Conductivity
       [ ANTHCON [A.TH.CON=<a>] [B.TH.CON=<a>] [C.TH.CON=<a>] [D.TH.CON=<a>]
           [E.TH.CON=<a>]
        ]

 Advanced Band Structure Parameters
        [ALPH0.N=<n>] [ALPHJ.N=<a>] [MJ.N=<a>] [EJ.N=<a>]
        [ALPH0.P=<n>] [ALPHJ.P=<a>] [MJ.P=<a>] [EJ.P=<a>]
      )

 Insulator Parameters
    { ( {  OXIDE | NITRIDE | SAPPHIRE | OXYNITRI | INSULATO
         | REGION=<c>
        }
        [PERMITTI=<a>] [TH.COND=<a>]
      )
   }

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]
```

Parameter	Type	Definition	Default	Units
PRINT	logical	Specifies that the anisotropic tensor components for each region are printed to the standard output.	false	

Semiconductor Parameters

SILICON	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **SILICON** with **REGION** statements.	false	
GAAS	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **GAAS** with **REGION** statements.	false	
POLYSILI	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **POLYSILI** with **REGION** statements.	false	
SEMICOND	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **SEMICOND** with **REGION** statements.	false	
SIGE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **SIGE** with **REGION** statements.	false	
ALGAAS	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **ALGAAS** with **REGION** statements.	false	
GERMANIU	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **GERMANIU** with **REGION** statements.	false	
SIC	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **SIC** with **REGION** statements.	false	
S.OXIDE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **S.OXIDE** with **REGION** statements.	false	
HGCDTE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **HGCDTE** with **REGION** statements.	false	
INGAAS	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **INGAAS** with **REGION** statements.	false	
INP	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **INP** with **REGION** statements.	false	

Parameter	Type	Definition	Default	Units
INAS	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **INAS** with **REGION** statements.	false	
DIAMOND	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **DIAMOND** with **REGION** statements.	false	
ZNSE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **ZNSE** with **REGION** statements.	false	
ZNTE	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **ZNTE** with **REGION** statements.	false	
A-SILICO	logical	Specifies that the mobility parameters are to apply to all regions that were specified as **A-SILICO** with **REGION** statements.	false	
REGION	char	The region name for which the anisotropic parameters are to apply.	none	
PRINT	logical	Specifies that the anisotropic tensor components for each region are printed to the standard output.	false	

Anisotropic Component Factors

PERMITTI	array	The diagonal components of the permittivity tensor. These components should be normalized by the isotropic permittivity for the region.	1.0, 1.0, 1.0	none
MU.N	array	The diagonal components of the electron mobility tensor. These components should be normalized by the isotropic electron mobility for the region.	1.0, 1.0, 1.0	none
MU.P	array	The diagonal components of the hole mobility tensor. These components should be normalized by the isotropic hole mobility for the region.	1.0, 1.0, 1.0	none
II.N	array	The diagonal components of the electron impact ionization coefficient tensor. These components should be normalized by the isotropic electron impact ionization coefficient for the region.	1.0, 1.0, 1.0	none
II.P	array	The diagonal components of the hole impact ionization coefficient tensor. These components should be normalized by the isotropic hole impact ionization coefficient for the region.	1.0, 1.0, 1.0	none
TH.COND	array	The diagonal components of the thermal conductivity tensor. These components should be normalized by the isotropic thermal conductivity for the region. This parameter is only used with the Lattice Temperature AAM.	1.0, 1.0, 1.0	none

Parameter	Type	Definition	Default	Units
General Anisotropic Electron Impact Ionization				
ANIIN	logical	Specifies that impact ionization generation due to electrons is treated anisotropically. If this parameter is specified, the array parameters which follow are used instead of the corresponding parameters available on the **MATERIAL** statement.	false	
N.ION.0	array	Coefficients used for impact ionization generation due to electrons when **ANIIN** is specified. These have the same meaning as **N.IONIZA** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **N.IONIZA** on the **MATERIAL** statement for each direction	cm^{-1}
N.ION.1	array	Coefficients used for impact ionization generation due to electrons when **ANIIN** is specified. These have the same meaning as **N.ION.1** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **N.ION.1** on the **MATERIAL** statement for each direction	1/cm-K
N.ION.2	array	Coefficients used for impact ionization generation due to electrons when **ANIIN** is specified. These have the same meaning as **N.ION.2** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **N.ION.2** on the **MATERIAL** statement for each direction	$1/cm\text{-}K^2$
ECN.II	array	Coefficients used for impact ionization generation due to electrons when **ANIIN** is specified. These have the same meaning as **ECN.II** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **ECN.II** on the **MATERIAL** statement for each direction	volts/cm
EXN.II	array	Coefficients used for impact ionization generation due to electrons when **ANIIN** is specified. These have the same meaning as **EXN.II** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **EXN.II** on the **MATERIAL** statement for each direction	none
General Anisotropic Hole Impact Ionization				
ANIIP	logical	Specifies that impact ionization generation due to holes is treated anisotropically. If this parameter is specified, the array parameters which follow are used instead of the corresponding parameters available on the **MATERIAL** statement.	false	
P.ION.0	array	Coefficients used for impact ionization generation due to holes when **ANIIP** is specified. These have the same meaning as **P.IONIZA** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **P.IONIZA** on the **MATERIAL** statement for each direction	cm^{-1}

Parameter	Type	Definition	Default	Units
P.ION.1	array	Coefficients used for impact ionization generation due to holes when **ANIIP** is specified. These have the same meaning as **P.ION.1** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **P.ION.1** on the **MATERIAL** statement for each direction	1/cm-K
P.ION.2	array	Coefficients used for impact ionization generation due to holes when **ANIIP** is specified. These have the same meaning as **P.ION.2** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **P.ION.2** on the **MATERIAL** statement for each direction	$1/cm\text{-}K^2$
ECP.II	array	Coefficients used for impact ionization generation due to holes when **ANIIP** is specified. These have the same meaning as **ECP.II** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **ECP.II** on the **MATERIAL** statement for each direction	volts/cm
EXP.II	array	Coefficients used for impact ionization generation due to holes when **ANIIP** is specified. These have the same meaning as **EXP.II** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **EXP.II** on the **MATERIAL** statement for each direction	none

General Anisotropic Thermal Conductivity

ANTHCON	logical	Specifies that thermal conductivity is treated anisotropically. If this parameter is specified, the array parameters which follow are used instead of the corresponding parameters available on the **MATERIAL** statement.	false	
A.TH.CON	array	Coefficients used for the anisotropic thermal conductivity model when **ANTHCON** is specified. These have the same meaning as **A.TH.CON** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **A.TH.CON** on the **MATERIAL** statement for each direction	(cm-K/W)
B.TH.CON	array	Coefficients used for the anisotropic thermal conductivity model when **ANTHCON** is specified. These have the same meaning as **B.TH.CON** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **B.TH.CON** on the **MATERIAL** statement for each direction	(cm/W)

Parameter	Type	Definition	Default	Units
C.TH.CON	array	Coefficients used for the anisotropic thermal conductivity model when **ANTHCON** is specified. These have the same meaning as **C.TH.CON** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **C.TH.CON** on the **MATERIAL** statement for each direction	(cm/W/K)
D.TH.CON	array	Coefficients used for the anisotropic thermal conductivity model when **ANTHCON** is specified. These have the same meaning as **D.TH.CON** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **D.TH.CON** on the **MATERIAL** statement for each direction	(cm/W/K$^{(\mathbf{E.TH.CON}-1)}$)
E.TH.CON	array	Coefficients used for the anisotropic thermal conductivity model when **ANTHCON** is specified. These have the same meaning as **E.TH.CON** on the **MATERIAL** statement except that a separate component can be specified for each direction.	same as **E.TH.CON** on the **MATERIAL** statement for each direction	none

Advanced Band Structure Parameters

ALPH0.N	number	First band structure parameter for electrons.	1.0	none
ALPHJ.N	array	Second band structure parameter for electrons. Up to 5 different values may be specified as an array.	0	1/eV
MJ.N	array	Third band structure parameter for electrons. Up to 5 different values may be specified as an array.	0	none
EJ.N	array	Fourth band structure parameter for electrons. Up to 5 different values may be specified as an array.	0	eV
ALPH0.P	number	First band structure parameter for holes.	1.0	none
ALPHJ.P	array	Second band structure parameter for holes. Up to 5 different values may be specified as an array.	0	1/eV
MJ.P	array	Third band structure parameter for holes. Up to 5 different values may be specified as an array.	0	none
EJ.P	array	Fourth band structure parameter for holes. Up to 5 different values may be specified as an array.	0	eV

Insulator Parameters

OXIDE	logical	Specifies that the anisotropic parameters are to apply to all regions that were specified as **OXIDE**.	false	

Parameter	Type	Definition	Default	Units
NITRIDE	logical	Specifies that the anisotropic parameters are to apply to all regions that were specified as **NITRIDE**.	false	
SAPPHIRE	logical	Specifies that the anisotropic parameters are to apply to all regions that were specified as **SAPPHIRE**.	false	
OXYNITRI	logical	Specifies that the anisotropic parameters are to apply to all regions that were specified as **OXYNITRI**.	false	
INSULATO	logical	Specifies that the anisotropic parameters are to apply to all regions that were specified as **INSULATO**.	false	
REGION	char	The region name for which the anisotropic parameters are to apply.	none	
PERMITTI	array	The diagonal components of the permittivity tensor. These components should be normalized by the isotropic permittivity for the region.	1.0, 1.0, 1.0	none
TH.COND	array	The diagonal components of the thermal conductivity tensor. These components should be normalized by the isotropic thermal conductivity for the region. This parameter is only used with the Lattice Temperature AAM.	1.0, 1.0, 1.0	none
Circuit Analysis AAM Parameters				
STRUCTUR	char	Selects the device in which the anisotropic parameters are altered. This parameter is only used with the Circuit Analysis AAM.	all devices	

Description

The **ANISOTROPIC** statement can be used to specify the diagonal components of tensors associated with physical models for materials that are to be treated as anisotropic. These tensors can either be simple multiplicative factors for the corresponding isotropic model. In the case of impact ionization and thermal conductivity, completely separate models can be specified for each direction.

The **ANISOTROPIC** statement should be specified before the first **SOLVE** statement where it is desired to account for the anisotropic nature of the material. The specified parameters remain in effect until one of the following occurs:

- They are changed by you on another **ANISOTROPIC** statement
- A **LOAD** statement is used to read in a saved solution that may contain different values for the anisotropic parameters
- A **MESH** statement is encountered, which initializes all parameters back to their default values.

This statement is also used to specify advanced band structure parameters that account for non-parabolicity and multiple bands in the calculation of the density of states.

See Also... To further illustrate the **ANISOTROPIC** statement, refer to input file *mdex22* in Anisotropic Material Examples, Chapter 17, "Anisotropic Block Example" on page 17-1.

Simple Anisotropic Models

The components that are specified should be normalized by the corresponding isotropic quantity. For example, the permittivity tensor can be expressed as:

$$\underline{\underline{\varepsilon}} = \begin{pmatrix} \varepsilon_{xx} & 0 & 0 \\ 0 & \varepsilon_{yy} & 0 \\ 0 & 0 & \varepsilon_{zz} \end{pmatrix} = \varepsilon_{mat} \cdot \begin{pmatrix} \left(\frac{\varepsilon_{xx}}{\varepsilon_{mat}}\right) & 0 & 0 \\ 0 & \left(\frac{\varepsilon_{yy}}{\varepsilon_{mat}}\right) & 0 \\ 0 & 0 & \left(\frac{\varepsilon_{zz}}{\varepsilon_{mat}}\right) \end{pmatrix} \quad \text{Equation 3-23}$$

$$= \varepsilon_{mat} \cdot \begin{pmatrix} \mathbf{PERM(1)} & 0 & 0 \\ 0 & \mathbf{PERM(2)} & 0 \\ 0 & 0 & \mathbf{PERM(3)} \end{pmatrix}$$

where ε_{mat} is the permittivity that is used by the program if the material is isotropic (that is, if **PERM(1)** = **PERM(2)** = **PERM(3)** = 1)

The value of ε_{mat} can be specified for each material using the **PERMITTI** parameter on the **MATERIAL** statement.

Anisotropic Properties of Materials

In addition to the anisotropic permittivity, the program provides for anisotropic properties for the following. Electron mobility, hole mobility, electron impact ionization coefficient, hole impact ionization coefficient, and thermal conductivity

The following equations illustrate these functions.

Equation 3-24

$$\underline{\underline{\mu_n}} = \begin{pmatrix} \mu_{n_{xx}} & 0 & 0 \\ 0 & \mu_{n_{yy}} & 0 \\ 0 & 0 & \mu_{n_{zz}} \end{pmatrix} = \mu_{n_{mat}} \cdot \begin{pmatrix} \mathbf{MU.N(1)} & 0 & 0 \\ 0 & \mathbf{MU.N(2)} & 0 \\ 0 & 0 & \mathbf{MU.N(3)} \end{pmatrix}$$

Equation 3-25

$$\underline{\underline{\mu_p}} = \begin{pmatrix} \mu_{p_{xx}} & 0 & 0 \\ 0 & \mu_{p_{yy}} & 0 \\ 0 & 0 & \mu_{p_{zz}} \end{pmatrix} = \mu_{p_{mat}} \cdot \begin{pmatrix} \mathbf{MU.P(1)} & 0 & 0 \\ 0 & \mathbf{MU.P(2)} & 0 \\ 0 & 0 & \mathbf{MU.P(3)} \end{pmatrix}$$

Equation 3-26

$$\underline{\underline{\alpha_n}} = \begin{pmatrix} \alpha_{n_{xx}} & 0 & 0 \\ 0 & \alpha_{n_{yy}} & 0 \\ 0 & 0 & \alpha_{n_{zz}} \end{pmatrix} = \alpha_{n_{mat}} \cdot \begin{pmatrix} \mathbf{II.N(1)} & 0 & 0 \\ 0 & \mathbf{II.N(2)} & 0 \\ 0 & 0 & \mathbf{II.N(3)} \end{pmatrix}$$

Equation 3-27

$$\underline{\underline{\alpha_p}} = \begin{pmatrix} \alpha_{p_{xx}} & 0 & 0 \\ 0 & \alpha_{p_{yy}} & 0 \\ 0 & 0 & \alpha_{p_{zz}} \end{pmatrix} = \alpha_{p_{mat}} \cdot \begin{pmatrix} \mathbf{II.P(1)} & 0 & 0 \\ 0 & \mathbf{II.P(2)} & 0 \\ 0 & 0 & \mathbf{II.P(3)} \end{pmatrix}$$

Equation 3-28

$$\underline{\underline{\kappa}} = \begin{pmatrix} \kappa_{xx} & 0 & 0 \\ 0 & \kappa_{yy} & 0 \\ 0 & 0 & \kappa_{zz} \end{pmatrix} = \kappa_{mat} \cdot \begin{pmatrix} \mathbf{TH.COND(1)} & 0 & 0 \\ 0 & \mathbf{TH.COND(2)} & 0 \\ 0 & 0 & \mathbf{TH.COND(3)} \end{pmatrix}$$

The specific models or parameters used to describe $\mu_{n_{mat}}$, $\mu_{p_{mat}}$, $\alpha_{n_{mat}}$, $\alpha_{p_{mat}}$, and κ_{mat} can be specified on the **MODELS**, **MATERIAL**, and **MOBILITY** statements.

Carrier Thermal Diffusivities

The Medici program accounts for anisotropic carrier thermal diffusion coefficients through the carrier mobility. That is, in the current density relations,

$$\vec{J_n} = qn\mu_n\vec{E} + q\left(\frac{k_B T}{q}\right)\mu_n\vec{\nabla}n + qnD_n^T\vec{\nabla}T$$ Equation 3-29

$$\vec{J_p} = qn\mu_p\vec{E} - q\left(\frac{k_B T}{q}\right)\mu_p\vec{\nabla}p - qpD_p^T\vec{\nabla}T \quad \text{Equation 3-30}$$

the thermal diffusion terms are assumed to be proportional to mobility and are given by

$$D_n^T = \mathbf{DN.LAT}\left(\frac{k_B}{q}\right)\mu_n \quad \text{Equation 3-31}$$

$$D_p^T = \mathbf{DP.LAT}\left(\frac{k_B}{q}\right)\mu_p \quad \text{Equation 3-32}$$

The factors **DN.LAT** and **DP.LAT** have been provided as user adjustable parameters and can be specified on the **MATERIAL** statement. The default values for these parameters are 1.

Example

For a previously created diode structure, specify an anisotropic electron mobility for all material regions that are defined as **SIC** such that $\mu_{N_{xx}} = 0.5\mu_n$ and $\mu_{n_{yy}} = \mu_n$:

```
MESH       IN.FILE=DIODE.MESH
MODELS     ANALYTIC  FLDMOB  CONSRH  AUGER  BGN
ANISOTRO   SIC  MU.N=(0.5,1.0,1.0)

SYMBOLIC   CARRIER=2  NEWTON
SOLVE      V1=0  ELEC=1  VSTEP=0.1  NSTEP=10
```

General Anisotropic Models

There are general anisotropic models for impact ionization and thermal conductivity in Medici.

Impact Ionization

General anisotropic models are available for both electron and hole impact ionization coefficients. Although the form of the impact ionization model is the same for each direction, all parameters used in the model expressions can have different values for each direction. The expressions for the electron impact ionization coefficients in *x* direction are

$$\alpha_{n,\,xx} = \mathbf{II.N(1)} \cdot \alpha_{n,\,xx}^{\infty}(T) \cdot \exp\left[-\left(\frac{E_{n,\,xx}^{crit}(T)}{E_{n,\,\|}}\right)^{\mathbf{EXN.II(1)}}\right] \quad \text{Equation 3-33}$$

where

Equation 3-34

$$E_{n,xx}^{crit}(T) = \frac{E_g(T)}{q\ \lambda_n(T)} \text{(default calculation), or } \mathbf{ECN.II(1)} \text{(if specified)}$$

and

Equation 3-35

$$\alpha_{n,xx}^{\infty} = \mathbf{N.ION.0(1)} + \mathbf{N.ION.1(1)} \cdot T + \mathbf{N.ION.2(1)} \cdot T^2$$

For $\alpha_{n,yy}^{\infty}$ similar expressions apply but with the index of all of the array parameters used in the above expressions changed from "1" to "2." Analogous expressions hold for hole impact ionization coefficients.

Thermal Conductivity

General anisotropic models are available for thermal conductivity. The same form of the thermal conductivity model is used for each direction. All parameters used in each direction can have different values. The expressions for thermal conductivity are

$$\underline{\underline{\kappa}} = \begin{bmatrix} \kappa_{xx} & 0 & 0 \\ 0 & \kappa_{yy} & 0 \\ 0 & 0 & \kappa_{zz} \end{bmatrix}$$

Equation 3-36

and

Equation 3-37

$$\kappa_{xx} = [\mathbf{A.TH.CON(1)} + \mathbf{B.TH.CON(1)} \cdot T + \mathbf{C.TH.CON(1)} \cdot T^2 + \mathbf{D.TH.CON(1)} \cdot T^{\mathbf{E.TH.CON(1)}}]^{-1}$$

For κ_{yy} and κ_{zz} similar expressions apply but with the index of all of the array parameters used in the above expressions changed from "1" to "2" and "3," respectively.

Examples

The following statement specifies that general anisotropic impact ionization due to electrons should be used. Some parameters associated with the $\alpha_{n,yy}$ component of the electron impact ionization coefficient have been changed from their default values. The **PRINT** parameter requests that the values for all anisotropic parameters be printed to the standard output file.

```
ANISOTRO SIC ANIIN PRINT
+       N.ION.0(2)=8.0E5
+       N.ION.1(2)=2.0E3
+       ECN.II(2)=2.0E7
```

The following statement specifies the general anisotropic thermal conductivity in silicon carbide. Some parameters associated with the y-direction have been changed from their default values.

```
ANISOTRO SIC ANTHCON PRINT
+            A.TH.CON(2)=0.027
+            B.TH.CON(2)=-1.9E-6
+            C.TH.CON=(1.65E-6)
```

Advanced Band Structure Parameters

The advanced band structure model is described by the following set of equations for electrons and holes. These parameters allow for non-parabolicity and multiple bands in the calculation of the density of states. The array parameters **ALPHJ.N**, **MJ.N**, **EJ.N**, **ALPHJ.P**, **MJ.P** and **EJ.P**, take on different values for each value of the summation index i. Up to five values may be specified for each array.

Equation 3-38

$$N_c = N_c \,\mathbf{ALPH0.N} + \sum_j \left(\mathbf{MJ.N} \left(1 + \mathbf{ALPHJ.N} \frac{15 k_b T}{4q} \right) \exp\left(\frac{\mathbf{EJ.N}\ q}{k_b T} \right) \right)$$

Equation 3-39

$$N_v = N_v \,\mathbf{ALPH0.P} + \sum_j \left(\mathbf{MJ.P} \left(1 + \mathbf{ALPHJ.P} \frac{15 k_b T}{4q} \right) \exp\left(\frac{\mathbf{EJ.P}\ q}{k_b T} \right) \right)$$

3.5 Circuit Analysis

The following statements create and simulate a circuit:

Statement	Definition	Page
START	Enters circuit mode.	3-328
C<*name*>	Creates a capacitor.	3-329
D<*name*>	Creates a diode.	3-329
E<*name*>	Creates a voltage controlled voltage source.	3-330
F<*name*>	Creates a current controlled current source.	3-331
G<*name*>	Creates a voltage controlled current source.	3-332
H<*name*>	Creates a current controlled voltage source.	3-332
I<*name*>	Creates an independent current source.	3-333
K<*name*>	Creates a coupling between inductors.	3-336
L<*name*>	Creates an inductor.	3-336
M<*name*>	Creates a MOS transistor.	3-337
P<*name*>	Creates a Medici numerical element.	3-338
Q<*name*>	Creates a bipolar transistor.	3-339
R<*name*>	Creates a resistor.	3-340
V<*name*>	Creates an independent voltage source.	3-341
W<*name*>	Creates a multiplier type voltage controlled current source.	3-344
.MODEL	Specifies parameters for the active models (**D**, **M**, **Q**).	3-346
.NODESET	Specifies an initial guess for circuit voltages.	3-353
.IC	Specifies fixed node voltages during DC analysis of an initial guess during transient analysis.	3-354
.OPTIONS	Specifies solution and output control options.	3-355
.LOAD	Loads a solution or mesh.	3-357
FINISH	Exits circuit mode.	3-358

Note:

The* .DC, .SAVE, *and* .TRAN *statements are documented in Section 3.7 Old Statements on page 3-407. Although these statements can still be specified, their use is not recommended.

It is recommended that the circuit analysis statements be used for constructing while the actual simulation is done in the regular* Medici *mode. See Chapter 12 for examples.

Circuit Mode Overview

The program enters circuit mode whenever a **`START CIRCUIT`** statement is encountered and returns to Medici mode when the **`FINISH CIRCUIT`** statement is read.

A circuit may contain the following:

- Maximum of 20 Medici elements (devices)
- Maximum of 500 circuit nodes
- Maximum of 500 circuit elements
- Maximum of 50 circuit models
- Maximum of 200 model parameters
- Maximum of 200 Medici terminals
- Maximum of 200 Medici device regions

The total number of nodes for all devices in the circuit must be less than or equal to the maximum number of nodes allowed for the version of Medici that is being used.

This section details the following:

- Syntax
- Statements
- Parameters
- Differences between Medici mode and circuit mode
- Elements

Note:

Medici ***requires that the Newton method be specified when using circuit mode.***

Syntax and Use

Circuit mode syntax is different from standard Medici syntax. This section details the differences and provides instructive examples.

Note:

The syntax in circuit mode is similar to that of SPICE programs. Except for some minor differences, users of SPICE will recognize the circuit mode syntax.

Control Statements

Control statements, such as **.IC**, **.DC**, and **.OPTIONS** are distinguished since they start with a period.

Order of Execution

Statements in the file are executed in the order given, so nodes must be defined before their voltages can be set. Once a simulation has been performed, no new elements can be added to the circuit. This differs from standard SPICE syntax, where **.TRAN** and **.DC** statements may be placed before the circuit elements.

Circuit Elements

- Circuit element types are determined by the first character of the element name.

  For example, **R1** is a resistor because it begins with an "R" and **CAXFGR** is a a capacitor because it begins with a "C."
- Element names must be eight characters or less and must be unique.
- The names of all allowed elements are contained in the circuit analysis formatted key file *mdfky1*. Active elements (**D**, **Q**, and **M** for diode, BJT, and MOSFET elements, respectively) have model parameters associated with them. These model parameters and their default values are also found in the file *mdfky1*. This file is extensible. New elements and model parameters can be added to the program or the default values can be altered.

Punctuation and Case

The program does not distinguish between upper and lower case, however the case is preserved. For example, text is *not* converted to upper case as in some versions of SPICE.

- A space, comma, tab, left parenthesis, or right parenthesis may serve as a delimiter between statement names, element names, node names, or parameters.
- The dollar sign (**$**) or asterisk (*****) at the start of a line indicates that the line is a comment and the line is not processed.

Circuit Nodes

- Node names are character parameters and can be any alphanumeric string less than or equal to eight characters in length.
- Node zero is always the datum node (or ground) and all circuits must contain node zero.

- All nodes must have a DC path to ground and every node must have at least two connections.

Current Nodes

The names of voltage sources (dependent and independent) and inductors are in effect "current nodes". The current flowing in these elements can be plotted on a **`PLOT.1D`** statement, or set to initial values using a **`.IC`** or **`.NODESET`** statement.

Medici Devices

Unused terminals in a Medici device are connected to a ground (all terminals do not need to be connected to circuit nodes). Only Dirichlet contacts may be attached to circuit nodes. Schottky contacts, resistive contacts, lumped element, and charge and current boundary contacts may not be connected to circuit nodes, but they may be connected to other device terminals.

Parameters

Circuit-mode parameters obey slightly different syntax rules than Medici parameters. The following table summarizes the differences in syntax rules between circuit-mode and Medici parameters:

Table 3-30 Circuit Mode Parameters

Parameter Type	Mode	Use Assigned Names?	Use Unit Identifier?	Type	Usage
`num`	Medici	YES	NO	Numeric	name=value
`num`	Circuit	YES	YES	Numeric	name=value
`n`	Circuit	YES	YES	Numeric	name
`log`	Medici	YES	NO	Logical	name
`log`	Circuit	NO	NO	Logical	name
`char`	Medici	YES	NO	Character	name=value
`char`	Circuit	NO	NO	Character	name=value
`c`	Circuit	NO	NO	Character	name

Numerical Parameters

The values of resistors and capacitors can be either of the following types:

- num—Type num parameters are similar to Medici numerical parameters. They are specified with the parameter name and value, separated by an "=".
- n—Type n parameters are position-dependent and have names surrounded by "< >". They are specified by replacing the parameter name in the statement syntax with the parameter value.

Unit Identifiers

The values of all circuit-mode numerical parameters can include the following unit identifiers as in SPICE:

```
T=1e12   G=1e9   MEG=1e6   K=1e3
M=1e-3   U=1e-6  N=1e-9    P=1e-12   F=1e-15
```

Therefore the number 1000 can be represented in any of the following ways:

```
1000   1K    1e3   .001MEG  1000.00
```

Numerical Expressions

Numerical expressions and assigned names can be used to specify numerical values. The following statements produce a 1000 ohm resistor connected to a 20 volt DC source:

```
ASSIGN   NAME=RV1  N.VALUE=0.5
ASSIGN   NAME=VV1  N.VALUE=20.0
START CIRCUIT
   V1    dog   cat  2*[@RV1*@VV1]
   R1    dog   cat  [0.5+@RV1]k
FINISH CIRCUIT
```

Note:

Left and right parentheses cannot be used as part of numerical expressions because they serve as delimiters in the SPICE syntax. Group them by using square brackets. See the example in Numerical Expressions on page 3-325.

Logical Parameters

Circuit-mode logical parameters may not be specified by assigning a value to the parameter name with an "=", and the name cannot be specified with an assigned name.

Character Parameters

Circuit-mode character parameters can be either of the following types:

- char—Type char parameters are similar to Medici character parameters and are specified with the parameter name and value, separated by an "=".
- c—Type c parameters are position-dependent and have names surrounded by "< >". They are specified by replacing the parameter name in the statement syntax with the parameter value.

The values of circuit-mode character parameters may not include assigned names.

Regional Parameters

All regional parameters, such as material coefficients, and mobility parameters are local to the Medici devices. That is, each Medici device can have its own parameters. The models used in the simulation and the temperature, however, are global. For example, if one Medici device uses **CONSRH**, all Medici devices will use **CONSRH**.

Active Circuit Elements

Every active circuit element (**D**, **Q**, **M**) must have a **.MODEL** statement to define its parameters. However, every **.MODEL** statement need not be utilized and several active circuit elements may refer to the same **.MODEL** statement.

The **.MODEL** statement has the form

```
.MODEL <model-name> <model-type> <parameters>
```

where **<model-name>** is the user-selected name to associate with the current model specification, **<model-type>** is the identifier for a set of default model parameters to use from the circuit analysis formatted key file *mdfky1*, and **<parameters>** represents adjustments to the default parameter values for the particular model at hand.

Model Types

Medici currently supports eight different model types that include **d** (diode model), **npn** and **pnp** (BJT models), **nmos** and **pmos** (MOSFET models), and **hspice**, **hspice28**, and **bsim3v3** (Star-Hspice-specific MOSFET models). Parameters associated with these model types can be found in the file *mdfky1*.

The file *mdfky1* can be extended to include new model types and their parameters. The maximum number of model types currently allowed is 10.

Model Parameters

Before Medici will recognize a parameter that is specified on the **.MODEL** statement, it must be included in the file *mdfky1*. If there are model parameters that are not presently included in this file that are needed for your simulations, they can be added to the end of the list for the appropriate model type. The maximum number of parameters for a given model type is 200, and the maximum number of parameters for all model types is 1000.

Every parameter in the file *mdfky1* must include a default value, a minimum value, and a maximum value. If a parameter value specified on the **.MODEL** statement does not fall within the range of the minimum and maximum values, Medici will issue a warning, but program execution will continue. Parameters which are not specified on the **.MODEL** statement will use the default values from *mdfky1*.

Note:

For model types* nmos *and* pmos*, some parameters in the file* *mdfky1* *have "*def=-999.*" specified. This indicates that the default values are calculated from other parameters. For the* Star-Hspice*-specific model types (*hspice*,* hspice28*,* bsim3v3*), "*def=-999.*" indicates that the* Star-Hspice *default value for this parameter should be used.

Using Your Own *mdfky1* File

In some cases, you may want to modify the parameters in the circuit analysis formatted key file, *mdfky1*, for your own particular needs. This is easy to do with Medici. Simply copy *mdfky1* to the directory where you want to keep or use it, make your desired changes, and then set the environment variable *MDFKY1* so that Medici knows where to find it.

```
setenv MDFKY1 /my-directory/mdfky1
```

See Chapter 1 for more information regarding the use of environment variables and initially assigned names with Medici.

MOSFET Models

Medici supports a number of different compact MOSFET model options. These include built-in SPICE models for levels 1, 2, and 3, and various Star-Hspice MOSFET models that are evaluated using *Avant!*'s Common Model Interface (CMI) functions. Table 3-31 lists the available choices.

Note:

If level 1, 2, or 3 is specified, by default the Medici ***built-in models will be used. To use the*** Star-Hspice ***version of level 1, 2, or 3, the*** **`HSPICE`** ***parameter on the*** **`.OPTION`** ***statement should be specified.***

Table 3-31 MOSFET Models Available in Medici

Level	MOSFET Model Description	Model specific parameters available in *mdfky1*	Model type selected with **`nmos`**/**`pmos`** specification
1	Schichman-Hodges model	yes	nmos/pmos or hspice (if .OPTION HSPICE is specified
2	MOS2 Grove-Frohman model (SPICE2G)	yes	nmos/pmos or hspice (if .OPTION HSPICE is specified
3	MOS3 empirical model (SPICE2G)	yes	nmos/pmos or hspice (if .OPTION HSPICE is specified
13	BSIM model	no	hspice
28	BSIM derivative; Avant! proprietary model	yes	hspice28
47	BSIM3 Version 2.0	no	hspice
49	BSIM3 Version 3 (Enhanced)	yes	bsim3v3
50	Philips MOS9	no	hspice
53	BSIM3 Version 3 (Berkeley)	yes	bsim3v3
55	EPFL-EKV Model Ver 2.6, R 11	no	hspice

START

Causes Medici to enter circuit mode.

START

CIRCUIT [INITIAL]

Name	Type	Definition	Default	Units
CIRCUIT	logical	Puts the program into circuit mode for circuit simulation.	true	
INITIAL	logical	Initializes the program for a new simulation.	false	

Example

```
START CIRCUIT
```

See Also... To further illustrate the **START** statement, refer to input files:

- *mdex11* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-2
- *mdex12b* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-7
- *mdex13c* in Circuit Analysis Examples, Chapter 12, "Structure" on page 12-26
- *mdex13d* and *mdex13e* in Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26

C<*name*>

Causes Medici to create a capacitive element.

C<*name*>

<*node+*> <*node-*> <*value*>

Name	Type	Definition	Default	Units
<*name*>	c	The user chosen identifying name.	none	
<*node+*>	c	The positive circuit node where the capacitor is attached.	none	
<*node-*>	c	The negative circuit node where the capacitor is attached.	none	
<*value*>	n	The value of the capacitor, which must not be zero.	none	farads

Example

```
C1  2 4 1p
```

See Also... To further illustrate the **C** statement, refer to input file *mdex13c* in Circuit Analysis Examples, Chapter 12, "Structure" on page 12-26.

D<*name*>

Causes Medici to create a PN junction diode.

D<*name*>

<*node+*> <*node-*> <*mname*> **[AREA=<n>]**

Name	Type	Definition	Default	Units
<*name*>	c	The user chosen identifying name.	none	
<*node+*>	c	The positive (P-type) node of the diode.	none	
<*node-*>	c	The negative (N-type) node of the diode.	none	
<*mname*>	c	The name of the model used for the diode.	none	
AREA	num	An area factor that multiplies the currents of the diode.	1.0	none

Example

```
D1 1 2 modela area=5
```

E*<name>*

Causes Medici to create a voltage controlled voltage source (VCVS).

E*<name>*

<node+> <node-> <cnode+> <cnode-> <value>

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<node+>	c	The positive circuit node where the VCVS is attached.	none	
<node->	c	The negative circuit node where the VCVS is attached.	none	
<cnode+>	c	The positive control node where the VCVS is attached.	none	
<cnode->	c	The negative control node where the VCVS is attached.	none	
<value>	n	The value of the VCVS.	none	none

Example

```
E1 a b c d 2
```

To further illustrate the **E** statement, refer to input file *mdex14b* in Lattice Temperature Examples, Chapter 13, "Amplifier" on page 13-11.

F*<name>*

Causes Medici to create a current controlled current source (CCCS).

F*<name>*

<node+> <node-> <vname> <value>

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<node+>	c	The positive circuit node where the CCCS is attached.	none	
<node->	c	The negative circuit node where the CCCS is attached.	none	
<vname>	c	The voltage source whose current controls the CCCS.	none	
<value>	n	The value of the CCCS.	none	none

Example

```
F1 a b Vdd 2.0
```

G*<name>*

Causes Medici to create a voltage controlled current source (VCCS).

G*<name>*

<node+> <node-> <cnode+> <cnode-> <value>

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<node+>	c	The positive circuit node where the VCCS is attached.	none	
<node->	c	The negative circuit node where the VCCS is attached.	none	
<cnode+>	c	The positive control node where the VCCS is attached.	none	
<cnode->	c	The negative control node where the VCCS is attached.	none	
<value>	n	The value of the VCCS.	none	mhos

Example

```
Gm a b c d 1m
```

H*<name>*

Causes Medici to create a current controlled voltage source (CCVS).

H*<name>*

<node+> <node-> <vname> <value>

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<node+>	c	The positive circuit node where the CCVS is attached.	none	
<node->	c	The negative circuit node where the CCVS is attached.	none	
<vname>	c	The voltage source whose current controls the CCVS.	none	
<value>	n	The value of the CCVS.	none	ohms

Example

```
H1 a b Vdd 2
```

I<*name*>

Causes Medici to create an independent current source.

I<*name*>

<*node+*> <*node-*>
{ <*value*>
| (PULSE <*i0*> <*ia*> <*td*> <*tr*> <*tf*> <*tp*> <*per*> **)**
| (EXP <*i0*> <*ia*> <*td1*> <*tau1*> <*td2*> <*tau2*> **)**
| (SIN <*i0*> <*ia*> <*freq*> <*tds*> <*theta*> **)**
| (SFFM <*i0*> <*ia*> <*fc*> <*mdi*> <*fs*> **)**
}

Name	Type	Definition	Default	Units
<*name*>	c	The user chosen identifying name.	none	
<*node+*>	c	The positive circuit node to which the current source is attached.	none	
<*node-*>	c	The negative circuit node to which the current source is attached.	none	
<*value*>	n	The value of the current source, if fixed.	none	amps
EXP	logical	Specifies that an exponential type current source is used.	false	
PULSE	logical	Specifies that a pulse type current source is used.	false	
SIN	logical	Specifies that a sinusoidal type current source is used.	false	
SFFM	logical	Specifies that a single frequency FM type current source is used.	false	
<*i0*>	n	Initial value for the current source.	none	amps
<*ia*>	n	Applied value for the current source.	none	amps
<*td*>	n	Delay time for the pulse type current source.	none	seconds
<*tr*>	n	Rise time for the pulse type current source.	none	seconds
<*tf*>	n	Fall time for the pulse type current source.	none	seconds
<*tp*>	n	Pulse width for the pulse type current source.	none	seconds
<*per*>	n	Period for the pulse type current source.	none	seconds
<*td1*>	n	Delay time for the rising portion of the exponential waveform.	none	seconds
<*tau1*>	n	Time constant for the rising portion of the exponential waveform.	none	seconds
<*td2*>	n	Delay time for the falling portion of the exponential waveform.	none	seconds
<*tau2*>	n	Time constant for the falling portion of the exponential waveform.	none	seconds
<*freq*>	n	Frequency of the sinusoidal source.	none	Hz
<*tds*>	n	Time delay for the sinusoidal source.	none	seconds
<*theta*>	n	Damping factor for the sinusoidal source.	none	1/seconds
<*fc*>	n	Carrier frequency for the single frequency FM source.	none	Hz
<*mdi*>	n	Modulation index for the single frequency FM source.	none	none
<*fs*>	n	Signal frequency for the single frequency FM source.	none	Hz

The value of an exponential type current source is given by the following equation:

Equation 3-40

$$= \begin{cases} <i0> & 0.0 < t < <td1> \\ <i0> + (<ia> - <i0>)\left[1 - \exp\left(-\dfrac{t - <td1>}{<taul>}\right)\right] & <td1> < t < <td2> \\ <i0> + (<ia> - <i0>) \cdot & \\ \left[\exp\left(-\dfrac{t - <td2>}{<tau2>}\right) - \exp\left(-\dfrac{t - <td1>}{<tau1>}\right)\right] & t > <td2> \end{cases}$$

The parameters of a pulse type current source are defined as follows. The value of a sinusoidal type current source is given by the following equation:

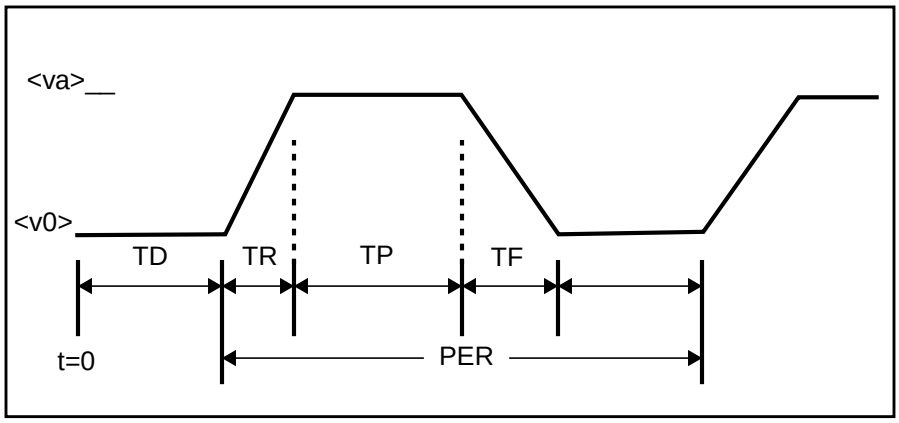


Figure 3-26 Pulse Type Current Source Parameters

$$I = \begin{cases} <i0> & 0.0 < t < <tds> \\ <i0> + <ia> \cdot \exp[-(t - <tds>)<theta>] \cdot & \\ \sin[2\pi <freq>(t - <tds>)] & t > <tds> \end{cases}$$

Equation 3-41

The value of a single frequency FM type current source is given by the following equation:

$$I = <i0> + <ia> \cdot \sin[2\pi <fc> t + <mdi> \cdot \sin(2\pi <fs> t)]$$

Equation 3-42

Example

```
Ia 1 2 pulse 1 5 1n 1n 1n 10n 30n
Ii 3 4 5
```

See Also... To further illustrate the **I** statement, refer to input files:

- *mdex11* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-2
- *mdex14b* in Lattice Temperature Examples, Chapter 13, "Current Source" on page 13-10

K*<name>*

Causes Medici to create a mutual inductive coupling element.

K*<name>*

<lname1> <lname2> <value>

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<lname1>	c	The name of the first inductor of the coupled pair.	none	
<lname2>	c	The name of the second inductor of the coupled pair.	none	
<value>	n	The value of the mutual inductance. This value must be between zero and one.	none	none

Example

```
K1 L1  L2 1.0
```

L*<name>*

Causes Medici to create an inductive element.

L*<name>*

<node+> <node-> <value>

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<node+>	c	The positive circuit node where the inductor is attached.	none	
<node->	c	The negative circuit node where the inductor is attached.	none	
<value>	n	The value of the inductor. This value must not be zero.	none	henrys

Example

```
Lload a b 1n
```

M*<name>*

Causes Medici to create an MOS transistor.

M*<name>*

<noded> <nodeg> <nodes> <nodeb> <mname>
[W=<n>] [L=<n>] [AS=<n>] [AD=<n>] [PS=<n>] [PD=<n>]

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<noded>	c	The drain node of the transistor.	none	
<nodeg>	c	The gate node of the transistor.	none	
<nodes>	c	The source node of the transistor.	none	
<nodeb>	c	The bulk node of the transistor.	none	
<mname>	c	The name of the model to be used for the transistor.	none	
W	num	The channel width of the transistor.	100.0	microns
L	num	The channel length of the transistor.	100.0	microns
AS	num	The area of the source diffusion of the transistor.	0.0	square microns
AD	num	The area of the drain diffusion of the transistor.	0.0	square microns
PS	num	The perimeter of the source diffusion of the transistor.	0.0	microns
PD	num	The perimeter of the drain diffusion of the transistor.	0.0	microns

Example

```
M1 1 2 3 4 modela L=5 W=1 AS=5 AD=5 PS=12 PD=12
```

See Also... To further illustrate the **M** statement, refer to input file *mdex13c* Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26.

P*<name>*

Causes Medici to create a numerical Medici-type circuit element.

P*<name>*

<node1>=*<term1>* *<node2>*=*<term2>* **.........**
FILE=<c> WIDTH=<n>

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<nodex>	c	The circuit node where the terminal is connected. This name should not start with the letter "f" or the letter "w".	none	
<termx>	char	The device terminal (electrode) which is connected to the circuit node.	none	integer
FILE	char	The file identifier for the mesh file that describes the Medici device. If this file is created using the **SAVE** statement with the **W.MODELS** parameter set, the device models, interface, material and mobility parameters are also transferred to Medici. Keep in mind that the models apply globally. For example, if any one device has **CONSRH** enabled, **CONSRH** is enabled for all devices in the circuit. All other parameters, such as mobility, interface, and material, are local and apply only to the one device. To use TIF files, specify **TIF** on the **.OPTIONS** statement.	none	
WIDTH	num	The width of this device. This factor multiplies the terminal current.	1	microns

Example

```
P1 1=2 5=1 8=3 FILE=BJT.MSH WIDTH=2.0
```

In this example, the *P1* device has:

- Electrode 2 connected to circuit node 1
- Electrode 1 connected to node 5
- Electrode 3 connected to node 8

It is not necessary to connect a circuit node to every terminal of the device. Unused terminals are automatically connected to ground.

See Also...

To further illustrate the **P** statement, refer to input files:

- *mdex13c* in Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26
- *mdex14b* in Lattice Temperature Examples, Chapter 13, "Bipolar Transistor Thermal Run-Away Analysis" on page 13-7

Q*<name>*

The **Q** statement instructs Medici to create a bipolar junction transistor.

Q*<name>*

<nodec> <nodeb> <nodee> <mname> **[AREA=<n>]**

Name	Type	Definition	Default	Units
<name>	c	The user chosen identifying name.	none	
<nodec>	c	The collector node of the transistor.	none	
<nodeb>	c	The base node of the transistor.	none	
<nodee>	c	The emitter node of the transistor.	none	
<mname>	c	The name of the model used for the transistor.	none	
AREA	num	An emitter area factor that multiplies the currents of the transistor.	1.0	none

Example

```
Q1 1 2 3 modela area=5
```

R<*name*>

Causes Medici to create a resistive element.

R<*name*>

<*node+*> <*node-*> <*value*> **[T1=<n>] [T2=<n>]**

Name	Type	Definition	Default	Units
<*name*>	c	The user chosen identifying name.	none	
<*node+*>	c	The positive circuit nodewhere the resistor is attached.	none	
<*node-*>	c	The negative circuit node the resistor is attached.	none	
<*value*>	n	The value of the resistor. This value must not be zero.	none	ohms
T1	num	Specifies the first temperature coefficient.	0.0	ohms/K
T2	num	Specifies the second temperature coefficient.	0.0	ohms/K^2

The value of the resistor is determined by the following equation:

$$R = \langle value \rangle [1 + \mathbf{T1} \cdot (T - \mathbf{T.NOM}) + \mathbf{T2} \cdot (T - \mathbf{T.NOM})^2] \quad \text{Equation 3-43}$$

where T is the analysis temperature and **T.NOM** is the nominal temperature set on the **.OPTIONS** statement.

Example

```
Rpwr  1 2 500 T1=.01 T2=.001
```

See Also... To further illustrate the **R** statement, refer to input files:

- *mdex11* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-2
- *mdex12b* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-7
- *mdex13c* in Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26

V<*name*>

Causes Medici to create an independent voltage source.

V<*name*>

<*node+*> <*node-*>
{ <*value*>
| **(** **PULSE** <*v0*> <*va*> <*td*> <*tr*> <*tf*> <*tp*> <*per*> **)**
| **(** **EXP** <*v0*> <*va*> <*td1*> <*tau1*> <*td2*> <*tau2*> **)**
| **(** **SIN** <*v0*> <*va*> <*freq*> <*tds*> <*theta*> **)**
| **(** **SFFM** <*v0*> <*va*> <*fc*> <*mdi*> <*fs*> **)**
}

Name	Type	Definition	Default	Units
<*name*>	c	The user chosen identifying name.	none	
<*node+*>	c	The positive circuit node the voltage source is attached.	none	
<*node-*>	c	The negative circuit node the voltage source is attached.	none	
<*value*>	n	The value of the voltage source, if fixed.	none	volts
EXP	logical	Specifies that an exponential type voltage source be used.	false	
PULSE	logical	Specifies that a pulse type voltage source is used.	false	
SIN	logical	Specifies that a sinusoidal type voltage source is used.	false	
SFFM	logical	Specifies that a single frequency FM type voltage source is used.	false	
<*v0*>	n	Initial value for the voltage source.	none	volts
<*va*>	n	Applied value for the voltage source.	none	volts
<*td*>	n	Delay time for the pulse type voltage source.	none	seconds
<*tr*>	n	Rise time for the pulse type voltage source.	none	seconds
<*tf*>	n	Fall time for the pulse type voltage source.	none	seconds
<*tp*>	n	Pulse width for the pulse type voltage source.	none	seconds
<*per*>	n	Period for the pulse type voltage source.	none	seconds
<*td1*>	n	Delay time for the rising portion of the exponential waveform.	none	seconds
<*tau1*>	n	Time constant for the rising portion of the exponential waveform.	none	seconds
<*td2*>	n	Delay time for the falling portion of the exponential waveform.	none	seconds
<*tau2*>	n	Time constant for the falling portion of the exponential waveform.	none	seconds
<*freq*>	n	Frequency of the sinusoidal source.	none	Hz
<*tds*>	n	Time delay for the sinusoidal source.	none	seconds
<*theta*>	n	Damping factor for the sinusoidal source.	none	1/seconds
<*fc*>	n	Carrier frequency for the single frequency FM source.	none	Hz

Name	Type	Definition	Default	Units
<*mdi*>	n	Modulation index for the single frequency FM source.	none	none
<*fs*>	n	Signal frequency for the single frequency FM source.	none	Hz

The value of an exponential type voltage source is given by the following equation:

Equation 3-44

$$V = \begin{cases} <v0> & 0.0 < t < <td1> \\ <v0> + (<va> - <v0>)\left[1 - \exp\left(-\dfrac{t - <td1>}{<tau1>}\right)\right] & <td1> < t < <td2> \\ <v0> + (<va> - <v0>) \cdot & \\ \left[\exp\left(-\dfrac{t - <td2>}{<tau2>}\right) - \exp\left(-\dfrac{t - <td1>}{<tau1>}\right)\right] & t > <td2> \end{cases}$$

The parameters of a pulse type voltage source are defined as follows:

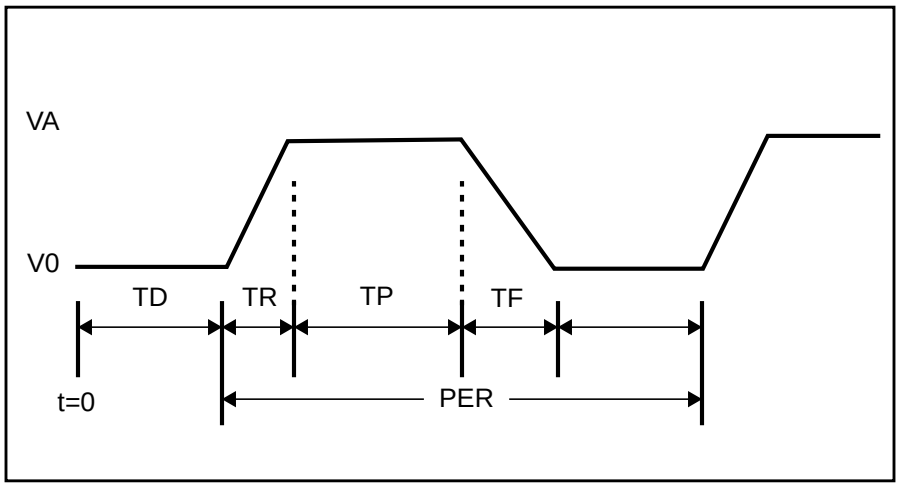


Figure 3-27 Pulse type voltage source

The value of a sinusoidal type voltage source is given by the following equation:

Equation 3-45

$$V = \begin{cases} <v0> & 0.0 < t < <tds> \\ <v0> + <va> \cdot \exp[-(t - <tds>)<theta>] \cdot & \\ \sin[2\pi <freq>(t + <tds>)] & t > <tds> \end{cases}$$

The value of a single frequency FM type voltage source is given by the equation:

Equation 3-46

$$V = \langle v0\rangle + \langle va\rangle \cdot \sin[2\pi\langle fct\rangle + \langle mdi\rangle \cdot \sin(2\pi\langle fs\rangle t)]$$

Example

```
Va 1 2 pulse 1 5 1n 1n 1n 10n 30n
Vi 3 4 5
```

See Also...

To further illustrate the **V** statement, refer to input files:

- *mdex11* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-2
- *mdex12b* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-7
- *mdex13c* in Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26

W<*name*>

Causes Medici to create a multiplier type voltage controlled current source.

W<*name*>

<*node+*> <*node-*> <*nodec1+*> <*nodec1-*> <*nodec2+*> <*nodec2-*> <*value*>

Name	Type	Definition	Default	Units
<*name*>	c	The user chosen identifying name.	none	
<*node+*>	c	The name of the positive output node.	none	
<*node-*>	c	The name of the negative output node.	none	
<*nodec1+*>	c	The name of the first positive control node.	none	
<*nodec1-*>	c	The name of the first negative control node.	none	
<*nodec2+*>	c	The name of the second positive control node.	none	
<*nodec2-*>	c	The name of the second negative control node.	none	
<*value*>	n	Multiplier value.	none	Amps/Volt^2

Example

```
W1 1 2    3 4  5 6 1.0
```

The W element generates a current equal to the product of two control voltages:

I = <value> * (<nodec1+> - <nodec1->) * (<nodec2+> - <nodec2->)

To see the general use of the **W** statement, refer to Circuit Analysis Examples, Figure 12-22 on page 12-23.

One of the most common uses of the W element is to create a voltage controlled conductance (resistor). To create a voltage controlled conductance, simply make one of the control voltages the same as the W element output nodes.

In the following example, the voltage controlled conductance is connected between terminals 1 and 2 while the controlling voltage is between nodes 3 and 4.

```
W1 1 2  1 2  3 4 1.0
```

If a time dependent voltage source is connected to nodes 3 and 4 in the above example, than a time dependent conductance (resistance) will be created.

For example to create a resistor (connected between nodes 1 and 2) which has a value of 2 ohms for 1ms and 1e6 ohms the rest of the time (with rise, fall and delay times of 1ns and period 1.0 sec) use:

```
W1 1 2   1 2   3 4  1.0
Rdummy 3 4 1e6
V1 3 4  pulse 1e-6 0.5 1e-9 1e-9 1e-9 1e-3 1.0
```

.MODEL

Specifies models to be used for the active lumped elements.

```
.MODEL

   <mod.nam>
   {  Diode Parameters
      ( D
        [IS=<n>] [CJO=<n>] [M=<n>] [VJ=<n>] [N=<n>] [FC=<n>]
        [BV=<n>] [EG=<n>] [XTI=<n>] [TT=<n>]
      )

      Bipolar Junction Transistor Parameters
    | ( {NPN | PNP}
        [IS=<n>] [BF=<n>] [BR=<n>] [TF=<n>] [TR=<n>] [CJC=<n>]
        [CJE=<n>] [VJC=<n>] [VJE=<n>] [MJC=<n>] [MJE=<n>]
        [IKF=<n>] [IKR=<n>] [NE=<n>] [NC=<n>] [NF=<n>] [NR=<n>]
        [VAF=<n>] [VAR=<n>] [ISC=<n>] [ISE=<n>] [FC=<n>]  [FE=<n>]
        [VTF=<n>] [ITF=<n>] [XTF=<n>] [RB=<n>] [RBM=<n>]
        [XTI=<n>] [XTB=<n>] [EG=<n>]
      )

      MOS Transistor Parameters (Medici Built-in Models)
    | ( {NMOS | PMOS}

       Common and Level 1 parameters
        [LEVEL=<n>] [LD=<n>] [TOX=<n>] [NSUB=<n>] [NSS=<n>] [UO=<n>]
        [PHI=<n>] [GAMMA=<n>] [TPG=<n>] [VTO=<n>] [KP=<n>] [JS=<n>]
        [XJ=<n>] [LAMBDA=<n>]

       Level 2 adds the following to the common set
        [UCRIT=<n>] [UEXP=<n>] [VMAX=<n>] [NFS=<n>] [NEFF=<n>]
        [DELTA=<n>]

       Level 3 adds the following to the common set
        [KAPPA=<n>] [DELTA=<n>] [THETA=<n>] [VMAX=<n>] [ETA=<n>]
        [NFS=<n>]

       Capacitance Parameters
        [CGSO=<n>] [CGDO=<n>] [CJ=<n>] [CJSW=<n>][MJ=<n>] [MJSW=<n>]
        [MCAP=<n>] [FC=<n>] [PB=<n>] [XQC=<n>] [K1=<n>]
      )

      MOS Transistor Parameters (Star-Hspice Levels 1, 2, and 3)
    | ( { ( HSPICE TYPE=<n> ) | NMOS | PMOS}
        LEVEL=<n> [COX=<n>] [KP=<n>] [LAMBDA=<n>] [TOX=<n>] [UO=<n>]
        [DEL=<n>] [LD=<n>] [LDAC=<n>] [LMLT=<n>] [WD=<n>] [WDAC=<n>]
        [WMLT=<n>] [XJ=<n>] [XL=<n>] [XW=<n>] [GAMMA=<n>] [NFS=<n>]
        [NSUB=<n>] [PHI=<n>] [VTO=<n>] [ECRIT=<n>] [NEFF=<n>] [VMAX=<n>]
        [LREF=<n>] [WREF=<n>] [DELTA=<n>] [LND=<n>] [LN0=<n>] [ND=<n>]
```

(**.MODEL** statement continued on next page)

(**.MODEL** statement continued from previous page)

```
    [N0=<n>] [WIC=<n>] [WND=<n>] [WN0=<n>] [MOB=<n>] [THETA=<n>]
    [UCRIT=<n>] [UEXP=<n>] [UTRA=<n>] [DERIV=<n>] [KAPPA=<n>]
    [ETA=<n>] [JS=<n>] [CAPOP=<n>] [CJ=<n>] [CJSW=<n>] [MJ=<n>]
    [MJSW=<n>] [FC=<n>] [PB=<n>] [CGBO=<n>] [CGSO=<n>] [CGDO=<n>]
    [XQC=<n>] [K1=<n>]
  )

  MOS Transistor Parameters (Star-Hspice Level 28)
| ( { ( HSPICE28 TYPE=<n> ) | NMOS | PMOS}
    LEVEL=<n> [LD=<n>] [LDAC=<n>] [LMLT=<n>] [LREF=<n>] [XLREF=<n>]
    [WD=<n>] [WDAC=<n>] [WMLT=<n>] [XL=<n>] [XW=<n>] [WREF=<n>]
    [XWREF=<n>] [BEX=<n>] [FEX=<n>] [TCV=<n>] [B1=<n>] [LB1=<n>]
    [WB1=<n>] [B2=<n>] [LB2=<n>] [WB2=<n>] [CGBO=<n>] [CGDO=<n>]
    [CGSO=<n>] [ETA0=<n>] [LETA=<n>] [WETA=<n>] [ETAMN=<n>]
    [LETAMN=<n>] [WETAMN=<n>] [GAMMN=<n>] [LGAMN=<n>] [WGAMN=<n>]
    [K1=<n>] [LK1=<n>] [WK1=<n>] [K2=<n>] [LK2=<n>] [WK2=<n>]
    [MUZ=<n>] [LMUZ=<n>] [WMUZ=<n>] [N0=<n>] [LN0=<n>] [WN0=<n>]
    [NB0=<n>] [LNB=<n>] [WNB=<n>] [ND0=<n>] [LND=<n>] [WND=<n>]
    [PHI0=<n>] [LPHI=<n>] [WPHI=<n>] [TOXM=<n>] [U00=<n>] [LU0=<n>]
    [WU0=<n>] [U1=<n>] [LU1=<n>] [WU1=<n>] [VDDM=<n>] [VFB0=<n>]
    [LVFB=<n>] [WVFB=<n>] [WFAC=<n>] [LWFAC=<n>] [WWFAC=<n>]
    [WFACU=<n>] [LWFACU=<n>] [WWFACU=<n>] [X2E=<n>] [LX2E=<n>]
    [WX2E=<n>] [X2M=<n>] [LX2M=<n>] [WX2M=<n>] [X2U0=<n>] [LX2U0=<n>]
    [WX2U0=<n>] [X2U1=<n>] [LX2U1=<n>] [WX2U1=<n>] [X33M=<n>]
    [LX33M=<n>] [WX33M=<n>] [X3E=<n>] [LX3E=<n>] [WX3E=<n>] [X3MS=<n>]
    [LX3MS=<n>] [WX3MS=<n>] [X3U1=<n>] [LX3U1=<n>] [WX3U1=<n>]
    [XPART=<n>]
  )

  MOS Transistor Parameters (Star-Hspice Levels 49 and 53)
| ( { ( BSIM3V3 TYPE=<n> ) | NMOS | PMOS}
    LEVEL=<n> [VERSION=<n>] [HSPVER=<n>] [PARAMCHK=<n>] [APWARN=<n>]
    [BINFLAG=<n>] [MOBMOD=<n>] [CAPMOD<n>] [NOIMOD=<n>] [NLEV=<n>]
    [NQSMOD=<n>] [SFVTFLAG=<n>] [VFBFLAG=<n>] [VGSLIM=<n>] [TOX=<n>]
    [XJ=<n>] [NGATE=<n>] [VTH0=<n>] [NSUB=<n>] [NCH=<n>] [NLX=<n>]
    [K1=<n>] [K2=<n>] [K3=<n>] [K3B=<n>] [W0=<n>] [DVT0W=<n>]
    [DVT1W=<n>] [DVT2W=<n>] [DVT0=<n>] [DVT1=<n>] [DVT2=<n>]
    [ETA0=<n>] [ETAB=<n>] [DSUB=<n>] [VBM=<n>] [U0=<n>] [UA=<n>]
    [UB=<n>] [UC=<n>] [A0=<n>] [AGS=<n>] [B0=<n>] [B1=<n>] [KETA=<n>]
    [VOFF=<n>] [VSAT=<n>] [A1=<n>] [A2=<n>] [RDSW=<n>] [PRWG=<n>]
    [PRWB=<n>] [WR=<n>] [NFACTOR=<n>] [CIT=<n>] [CDSC=<n>] [CDSCD=<n>]
    [CDSCB=<n>] [PCLM=<n>] [PDIBLC1=<n>] [PDIBLC2=<n>] [PDIBLCB=<n>]
    [DROUT=<n>] [PSCBE1=<n>] [PSCBE2=<n>] [PVAG=<n>] [DELTA=<n>]
    [ALPHA0=<n>] [BETA0=<n>] [RSH=<n>] [XPART=<n>] [CGSO=<n>]
    [CGDO=<n>] [CGBO=<n>] [CGS1=<n>] [CGD1=<n>] [CKAPPA=<n>] [CF=<n>]
    [CLC=<n>] [CLE=<n>] [VFBCV=<n>] [WINT=<n>] [WLN=<n>] [WW=<n>]
    [WWN=<n>] [WWL=<n>] [DWG=<n>] [DWB=<n>] [LINT=<n>] [LL=<n>]
    [LLN=<n>] [LW=<n>] [LWN=<n>] [LWL=<n>] [DLC=<n>] [DWC=<n>]
    [KT1=<n>] [KT1L=<n>] [KT2=<n>] [UTE=<n>] [UA1=<n>] [UB1=<n>]
    [UC1=<n>] [AT=<n>] [PRT=<n>] [XTI=<n>] [LMIN=<n>] [LMAX=<n>]
```

(**.MODEL** statement continued on next page)

(**.MODEL** statement continued from previous page)

```
        [WMIN=<n>] [WMAX=<n>] [BINUNIT=<n>] [GAMMA1=<n>] [GAMMA2=<n>]
        [VBX=<n>] [XT=<n>] [NOIA=<n>] [NOIB=<n>] [NOIC=<n>] [EM=<n>]
        [AF=<n>] [KF=<n>] [EF=<n>] [ACM=<n>] [JS=<n>] [JSW=<n>] [NJ=<n>]
        [N=<n>] [CJ=<n>] [CJSW=<n>] [CJSWG=<n>] [CJGATE=<n>] [PB=<n>]
        [PBSW=<n>] [PHP=<n>] [PBSWG=<n>] [MJ=<n>] [MJSW=<n>] [MJSWG=<n>]
        [ELM=<n>] [TOXM=<n>] [VFB=<n>] [NOFF=<n>] [VOFFCV=<n>] [IJTH=<n>]
        [ALPHA1=<n>] [ACDE=<n>] [MOIN=<n>] [TPB=<n>] [TPBSW=<n>]
        [TPBSWG=<n>] [TCJ=<n>] [TCJSW=<n>] [TCJSWG=<n>] [LLC=<n>]
        [LWC=<n>] [LWLC=<n>] [WLC=<n>] [WWC=<n>] [WWLC=<n>]
    )
}
```

Name	Type	Definition	Default	Units
<*mod.nam*>	c	The model name identifier.	none	
D	logical	Indicates this model is for a diode.	false	
NPN	logical	Indicates this model is for a NPN type bipolar transistor.	false	
PNP	logical	Indicates this model is for a PNP type bipolar transistor.	false	
NMOS	logical	Indicates this model is for a N-channel MOS transistor.	false	
PMOS	logical	Indicates this model is for a P-channel MOS transistor.	false	
HSPICE	logical	Indicates this model is for a MOS transistor. This selection indicates that the parameter set is taken from the model type **hspice** in the file *mdfky1*, which is appropriate for Star-Hspice levels 1, 2, or 3. If this parameter is specified instead of **NMOS** or **PMOS**, then the **TYPE** parameter should also be specified.	false	
HSPICE28	logical	Indicates this model is for a MOS transistor. This selection indicates that the parameter set is taken from the model type **hspice28** in the file *mdfky1*, which is appropriate for Star-Hspice level 28. If this parameter is specified instead of **NMOS** or **PMOS**, then the **TYPE** parameter should also be specified.	false	
BSIM3V3	logical	Indicates this model is for a MOS transistor. This selection indicates that the parameter set is taken from the model type **bsim3v3** in the file *mdfky1*, which is appropriate for Star-Hspice levels 49 or 53. If this parameter is specified instead of **NMOS** or **PMOS**, then the **TYPE** parameter should also be specified.	false	
TYPE	num	+1 for NMOS, -1 for PMOS	1.0	
IS	num	The diode saturation current.	le-14	amps
CJO	num	The diode zero bias capacitance.	0.0	farads
M	num	The diode junction grading coefficient.	0.5	none
VJ	num	The diode built in potential.	1.0	volts
N	num	Emission coefficient for the diode junction.	1.0	none

Name	Type	Definition	Default	Units
FC	num	The diode coefficient for forward biased junction depletion capacitance.	0.5	none
BV	num	The diode breakdown voltage.	infinity	volts
EG	num	The diode energy gap.	1.11	volts
XTI	num	The diode saturation current temperature exponent.	3.0	none
TT	num	The diode transit time.	0.0	seconds

Bipolar Junction Transistor Parameters

IS	num	The BJT saturation current.	le-16	amps
BF	num	The BJT forward current gain.	100.0	none
BR	num	The BJT reverse current gain.	1.0	none
TF	num	The BJT forward transit time.	0.0	seconds
TR	num	The BJT reverse transit time.	0.0	seconds
CJC	num	The BJT base collector zero bias junction capacitance.	0.0	farads
CJE	num	The BJT base emitter zero bias junction capacitance.	0.0	farads
VJC	num	The BJT base collector junction built in potential.	0.75	volts
VJE	num	The BJT base emitter junction built in potential.	0.75	volts
MJC	num	The BJT base collector junction grading coefficient.	0.33	none
MJE	num	The BJT base emitter junction grading coefficient.	0.33	none
IKF	num	The BJT forward knee current.	le6	amps
IKR	num	The BJT reverse knee current.	le6	amps
NE	num	The BJT base emitter junction leakage current emission coefficient.	1.5	none
NC	num	The BJT base collector junction leakage current emission coefficient.	1.5	none
NF	num	The BJT forward emission coefficient.	1.0	none
NR	num	The BJT reverse emission coefficient.	1.0	none
VAF	num	The BJT forward early voltage.	le6	volts
VAR	num	The BJT reverse early voltage.	le6	volts
ISC	num	The BJT base collector junction leakage saturation current.	0.0	amps
ISE	num	The BJT base emitter junction leakage saturation current.	0.0	amps
FC	num	The BJT base collector forward biased junction depletion capacitance coefficient.	0.5	none
FE	num	The BJT base emitter forward biased junction depletion capacitance coefficient.	0.5	none
VTF	num	The BJT parameter describing the dependence of parameter **TF** on the base collector voltage.	le6	volts

Name	Type	Definition	Default	Units
ITF	num	The BJT parameter describing the dependence of parameter **TF** on the collector current.	le6	amps
XTF	num	The BJT parameter describing the overall dependence of parameter **TF** on bias.	0.0	none
RB	num	The BJT zero bias base resistance.	0.0	ohms
RBM	num	The BJT minimum base resistance under bias.	0.0	ohms
XTI	num	The BJT saturation current temperature exponent.	3.0	none
XTB	num	The BJT temperature factor for parameter **BF**.	0.0	none
EG	num	The BJT energy gap.	1.11	volts
MOS Common and Level 1 Model Parameters				
LEVEL	num	The MOS model to use.	1	none
LD	num	The MOS lateral source and drain out diffusion.	0.0	microns
TOX	num	The MOS oxide thickness.	700	Angstroms
NSUB	num	The MOS substrate doping.	le15	/cm^3
NSS	num	The MOS surface state density.	0.0	/cm^2
UO	num	The MOS low field mobility.	400	cm^2/V-sec
PHI	num	The MOS surface potential.	2*Vt*log(**NSUB**/Ni)	volts
GAMMA	num	The MOS bulk threshold parameter. **Default: TOX**/Eox*sqrt(2*Esi*q*__NSUB__)	See Definition	volts
TPG	num	The MOS gate type parameter. 1 for N-type silicon gate with NMOS transistor or P-type silicon gate with PMOS transistor. 0 for aluminum gate. -1 for P-type silicon gate with NMOS transistor or N-type silicon gate with PMOS transistor.	1	none
VTO	num	The MOS zero bias threshold voltage.	calculated	volts
KP	num	The MOS transconductance parameter.	**UO***Eox/**TOX**	amps/volt^2
JS	num	The MOS saturation current for source and drain junctions.	0.0	amps/cm^2
XJ	num	The MOS source and drain junction depth.	0.0	microns
LAMBDA	num	The MOS channel modulation factor.	0.0	1/volt
Level 2 MOS Parameters				
UCRIT	num	The MOS vertical field mobility degradation factor.	le4	V/cm
UEXP	num	The MOS vertical field mobility degradation exponent.	0.0	none
VMAX	num	The MOS maximum carrier velocity.	0.0	cm/sec
NFS	num	The MOS fast surface state density.	0.0	1/cm^2
NEFF	num	The MOS total channel charge coefficient.	1.0	none
DELTA	num	Width effect on threshold voltage.	0.0	1/volt

Name	Type	Definition	Default	Units
MOS Level 3 Model Parameters				
KAPPA	num	The MOS saturation field factor.	0.2	none
THETA	num	The MOS mobility modulation factor.	0.0	1/volt
ETA	num	The MOS static feedback factor.	0.0	none
Common MOS Capacitance Parameters				
CGSO	num	The MOS gate source overlap capacitance parameter.	0.0	farads/cm
CGDO	num	The MOS gate drain overlap capacitance parameter.	0.0	farads/cm
CJ	num	The MOS source and drain area zero bias depletion capacitance parameter.	sqrt(q*Esi***NSUB**/**PB**)	farads/cm^2
CJSW	num	The MOS source and drain perimeter zero bias depletion capacitance parameter.	**CJ***__XJ__	farads/cm
MJ	num	The MOS source and drain bottom capacitance grading coefficient.	0.5	none
MJSW	num	The MOS source and drain perimeter capacitance grading coefficient.	0.333	none
MCAP	num	The MOS capacitance model to use: 1=BSIM, 2=MOS level 2.	1	none
FC	num	The MOS source and drain capacitance forward biased junction depletion capacitance parameter.	0.5	none
PB	num	The MOS source and drain junction built in potential.	1.0	none
XQC	num	The MOS coefficient of channel charge attributed to the drain.	0.5	none
K1	num	The MOS BSIM capacitance body effect coefficient.	0.63	none
Star-Hspice MOSFET Model Parameters				
<param>	num	Descriptions, default values, and units of all Star-Hspice parameters can be found in the *Star-Hspice User's Manual, Volume III - MOSFET Models*	Star-Hspice default	Star-Hspice units

Description

Medici contains built-in models for diodes, BJTs, and MOSFETs (levels 1, 2, and 3). For MOSFETs, various Star-Hspice models can also be selected (levels 1, 2, 3, 13, 28, 47, 49, 50, 53, and 55). Star-Hspice models are evaluated using *Avant!*'s Common Model Interface (CMI) functions.

If a MOSFET model is selected with **LEVEL** > 3, the appropriate Star-Hspice model is automatically selected. If **LEVEL** = 1, 2, or 3, the Medici built-in SPICE models are used by default. To use the Star-Hspice versions for levels 1, 2, and 3, include a **.OPTION** statement in the circuit mode input with the **HSPICE** parameter specified.

See Also... To further illustrate the **.MODEL** statement, refer to input files:

- *mdex12b* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-7
- *mdex13c* in Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26

Examples

Create a diode model named "MD":

```
.MODEL MD D IS=1e-14 CJO=1e-12
```

Create a n-channel MOSFET model named "NMD2" using Medici's built-in level 2 SPICE model:

```
.MODEL NMD2 NMOS LEVEL=2 TOX=250.
```

Create a n-channel MOSFET model named "NHS2" using Star-Hspice's level 2:

```
.OPTION HSPICE
.MODEL NHS2 NMOS LEVEL=2 TOX=250E-10
```

Create a n-channel MODEL model named "NHS28" using Star-Hspice's level 28:

```
.MODEL NHS28 NMOS LEVEL=28 TOXM=0.0250
```

SPICE Capacitance Models

The built-in MOSFET model differs from the one in standard SPICE 2G.6 in that the BSIM capacitance model is used as the default. This model was the most reliable and gives the best convergence.

The Meyer model was not included because it requires trapezoidal time integration and does not conserve charge, which can harm convergence.

The level 2 MOSFET capacitance model (Ward-Dutton) conserves charge but has discontinuous capacitance derivatives. This frequently produces the "Time-Step Too Small" error. This model is available in Medici, but is not recommended.

The SPICE LEVEL 3 MOSFET capacitance model, upon examination of the code, is nonfunctional and is not included.

.NODESET

The **.NODESET** statement specifies voltages at nodes used as an initial guess for a DC solution.

.NODESET

V(<*node1*>**)=<n> V(<***node2***>)=<n> .....**

Name	Type	Definition	Default	Units
<*nodex*>	c	The node where the voltage is specified.	none	

Example

```
.NODESET V(A)=5.0 V(B)=0.0 V(D)=1.0
```

See Also... To further illustrate the **.NODESET** statement, refer to input files:

- *mdex11* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-2
- *mdex12b* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-7
- *mdex13c* in Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26

.IC

Specifies voltages at nodes that remain fixed during DC solutions or used as a starting point for a transient solution.

.IC

V(<*node1*>**)=<n> V(**<*node2*>**)=<n> .....**

Name	Type	Definition	Default	Units
<*nodex*>	c	The node where the voltage is to be specified.	none	

Example

```
.IC V(1)=5.0 V(2)=0.0 V(3)=1.0
```

.OPTIONS

Sets options for the solution process.

```
.OPTIONS
   [T.TOL=<n>] [T.MIN=<n>] [P.TOL=<n>] [C.TOL=<n>] [ITLIM=<n>]
   [DELVMAX=<n>] [G.FORCE=<n>] [T.NOM=<n>] [V.MIN=<n>] [V.MAX=<n>]
   [2ND] [AUTO] [ALT] [ILUCGS] [LAT.TEMP] [COUP.LAT] [ELE.TEMP]
   [HOL.TEMP] [TIF] [HSPICE]
```

Parameter	Type	Definition	Default	Units
T.TOL	num	The target local truncation error for the automatic time step control. This is the same as the **TOL.TIME** parameter on the **METHOD** statement.	0.01	none
T.MIN	num	The minimum allowed time step. If the time step is reduced to less than **T.MIN**, the program halts.	le-15	seconds
P.TOL	num	The potential update tolerance. This is the same as the **PX.TOLER** parameter on the **METHOD** statement.	le-5	kT/q
C.TOL	num	The carrier concentration update tolerance. This is the same as the **CX.TOLER** parameter on the **METHOD** statement.	le-5	kT/q
ITLIM	num	The maximum number of Newton or Gummel iterations. This is the same as the **ITLIMIT** parameter on the **METHOD** statement.	20	none
DELVMAX	num	The maximum potential update allowed during one Newton iteration. This parameter can aid convergence by damping the Newton iteration process.	0.5	volts
G.FORCE	num	The value of the conductance used to force circuit nodes to a fixed potential.	10	mhos
T.NOM	num	The nominal temperature used in the resistor, MOSFET, diode and BJT models.	300	degrees K
VMIN	num	The minimum voltage allowed at a circuit node during Newton iteration or projection. Specifying an appropriate value for **VMIN** can aid convergence by constraining the node voltages to reasonable values.	-9999	volts
VMAX	num	The maximum voltage allowed at a circuit node during Newton iteration or projection. Specifying an appropriate value for **VMAX** can aid convergence by constraining the node voltages to reasonable values.	9999	volts
2ND	logical	Causes Medici to use the first/second variable order time discretization method rather than the fixed first order method.	true	
AUTO	logical	Enables Newton-Richardson iterations rather than standard Newton iterations.	true	
ALT	logical	Causes Medici to use alternating solution file names rather incrementing the file names. If this option is true, the last character of the filename alternates between zero and one. For example, if the original file name was *data*, the filenames generated for saving the solution alternates between *dat0* and *dat1*.	false	
ILUCGS	logical	Causes Medici to use ILUCGS as the linear solver rather than the direct method. ILUCGS may be faster on very large problems.	false	
LAT.TEMP	logical	Causes Medici to solve the lattice temperature equation, as well as the Poisson and continuity equations.	false	
COUP.LAT	logical	Causes Medici to solve all four equations (Poisson, electron continuity, hole continuity, and lattice temperature) as a coupled set rather than using the decoupled block approach.	false	
ELE.TEMP	logical	Causes Medici to solve the electron energy balance equation, as well as the Poisson and continuity equations.	false	

Parameter	Type	Definition	Default	Units
HOL.TEMP	logical	Causes Medici to solve the hole energy balance equation, as well as the Poisson and continuity equations.	false	
TIF	logical	Specifies that all mesh files and solution files will be read and written in the TIF format. This also applies to mesh files read by "p" elements.	true	
HSPICE	logical	Specifies that a Star-Hspice model will be used for MOS levels 1, 2, and 3.	false	

Example

```
.OPTIONS ILUCGS ^AUTO ^2ND P.TOL=.01 T.TOL=.01
```

See Also... To further illustrate the **.OPTIONS** refer to input files:

- *mdex11* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-2
- *mdex14b* in Lattice Temperature Examples, Chapter 13, "Bipolar Transistor Thermal Run-Away Analysis" on page 13-7

.LOAD

Specifies files from which to load data.

.LOAD

[MESH=<c>] [[SOLUTION=<c>] [STRUCTUR=<c>]] [ASCII]

Name	Type	Definition	Default	Units
MESH	char	The identifier for the mesh file to be read.	none	
SOLUTION	char	The identifier for the solution file to be read.	none	
STRUCTUR	char	The name of the device into which the solution information is read. The use of this parameter implies that the solution file being read contains information for only one device.	none	
ASCII	logical	Indicates that the specified file(s) are formatted files.	false	

Example

```
.LOAD  MESH=NMOS.MSH  SOLUTION=NMOS.SOL
```

See Also... To further illustrate the **.LOAD** statement, refer to input files:

- *mdex13d* and *mdex13e* in Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26
- *mdex14b* in Lattice Temperature Examples, Chapter 13, "Bipolar Transistor Thermal Run-Away Analysis" on page 13-7

FINISH

Causes Medici to exit circuit mode.

```
FINISH
   CIRCUIT
```

Name	Type	Definition	Default	Units
CIRCUIT	logical	Puts the program back into Medici mode.	true	

Example

```
FINISH CIRCUIT
```

See Also... To further illustrate the **FINISH** statement, refer to input files:

- *mdex11* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-2
- *mdex12b* in Circuit Analysis Examples, Chapter 12, "Generation of the Simulation Structure and Solutions" on page 12-7
- *mdex13c* in Circuit Analysis Examples, Chapter 12, "Structure" on page 12-26
- *mdex13d* and *mdex13e* in Circuit Analysis Examples, Chapter 12, "Transient Simulation of CMOS Pair with Compact Load" on page 12-26

3.6 Documentation and Control

The following statements document the input file and control execution of Medici:

Statement	Definition	Page
TITLE	Documents the input file and program output.	3-365
COMMENT	Documents the input file.	3-365
OPTION	Controls program output and function.	3-367
HELP	Prints information describing statements and parameters.	3-368
CALL	Enters statements into the input from files.	3-369
INTERACTIVE	Starts interactive input mode.	3-374
BATCH	Terminates interactive input mode.	3-376
I.PRINT	Prints input statements.	3-377
I.SAVE	Saves input statements in a file.	3-379
IF	Begins a sequence of one or more conditionally processed input statement blocks.	3-382
ELSE	Terminates a conditionally processed input statement block and begins an alternative conditionally processed input statement block.	3-384
IF.END	Terminates a sequence of one or more conditionally processed input statement blocks.	3-385
LOOP	Begins an input statement loop.	3-386
L.MODIFY	Modifies processing of an input statement loop.	3-392
L.END	Terminates input statement loops.	3-394
ASSIGN	Assigns values to an assigned name.	3-396
ECHO	Outputs text to your terminal.	3-404
RETURN	Terminates further processing of input statements in a file.	3-405
STOP	Terminates program execution.	3-405
IGNORE	Prevents processing of subsequent input statements in a file.	3-406

Control Statements

The control statements provide extended capabilities for controlling program execution including:

- Interactive operation
- Insertion of input statements from other files
- Saving of interactively entered input statements in output files

Obtaining Help

The **HELP** statement prints information describing a statement and its associated parameters. It is typically used during interactive input mode to determine valid parameter names and combinations of parameters.

The available information consists of the following:

- Statement syntax
- Default parameter values
- Units used for numerical and array parameters
- Synonyms for parameter names

Interactive and Batch Input Modes

The program runs in either of the following modes:

- Batch input mode—In batch mode the program reads input statements from the batch input file.
- Interactive input mode—In interactive input mode the program reads input statements entered interactively from your terminal.

At the beginning of program execution you must provide a file specification for the command input file, also referred to as the batch input file. If this file specification is blank, the program immediately enters interactive input mode and input statements must be entered from your terminal. Otherwise, the file specification is used to read the command input file.

The **INTERACTIVE** and **BATCH** statements provide the ability to switch between batch and interactive input modes.

Statement Line Numbers

Input statements are numbered with sequentially increasing line numbers. Input statements obtained from the batch input file are numbered using integers that start with 1 for the first statement. For each group of input statements entered interactively, the statements are numbered using the format "xxx/yyy".

- "xxx" is the line number of the statement preceding the first statement in the group.
- "yyy" starts with "001" for the first statement in the group and increases with successive statements in the group.

Processing Order

Input statements are processed in the order of increasing line number. The statement with line number "xxx/001" is processed immediately after the statement with line number "xxx". For a group of "yyy" input statements entered interactively, the statement with line number "xxx/yyy" is processed immediately before the statement with line number "xxx"+1.

Example

The following example illustrates the statements and line numbers printed on the standard output for a case including interactive input:

```
1... STMT1  PARM1=1
2... INTERACTIVE
3... STMT3  PARM3=3

2/001... STMT2  PARM2=2
2/002... BATCH
```

where:

- **STMT1**, **INTERACTIVE**, and **STMT3** statements are obtained from the batch input file
- **STMT2** and **BATCH** statements are entered interactively

The statements obtained from the batch input file are printed first, followed by the statements entered interactively. The statements are processed in the following order:

```
STMT1, INTERACTIVE, STMT2, BATCH, AND STMT3
```

Currently Available Input Statements

The program may obtain input statements by:

- Reading the batch input file
- Reading interactive input from your terminal
- Processing **CALL** statements

CALL statements read input statements from files or from the set of previously obtained input statements. At the beginning of program execution the batch input file is read completely and, where possible, **CALL** statements are processed. Interactive input is read when interactive input mode is entered.

At any time during program execution, all input statements that have been read are considered *currently available*. These statements are available for processing by

the **CALL**, **I.PRINT**, and **I.SAVE** statements. Statements are *currently available* even if they follow the input statement currently being processed.

Output of Statement Information

The **I.PRINT** statement prints a range of the currently available input statements. This statement is used during interactive input mode to determine which statements have been previously entered.

The **I.SAVE** statement saves input statements, including those entered interactively, in output files. These files are suitable for use as command input files during subsequent program execution.

Output to Terminal

The **ECHO** statement outputs text to your terminal. This statement is used with the **PROMPT** parameter on the **ASSIGN** statement to provide interactive terminal input and output.

Controlling Program Execution

This section describes the statements that control various aspects of program execution. The following statements are discussed:

- **CALL**
- **IF**, **ELSE**, and **IF.END**
- **LOOP**, **L.MODIFY**, and **L.END**
- **ASSIGN**
- **RETURN**
- **STOP**
- **IGNORE**

CALL Statement

The **CALL** statement reads input statements from a file or copies them from the *currently available* input statements. It can be used to repeatedly input groups of statements.

CALL can also be combined with **ASSIGN** statements to produce template files containing variable input values in the form of assigned names.

Conditional Statements

The **IF**, **ELSE**, and **IF.END** statements control conditional processing of input statement blocks. These statements are used to choose one block of statements for processing from a sequence of alternative statement blocks.

The choice of which statement block to process depends on the values of the following:

- Assigned names
- Numerical expressions
- Character expressions

LOOP Statements

The **LOOP**, **L.MODIFY**, and **L.END** statements control repeated processing of input statements in loops. These statements are used to efficiently specify a variety of different conditions for program execution.

Statement loops contain arbitrary combinations of input statements and control the variation of numerical and array parameters and assigned names.

ASSIGN Statement

The **ASSIGN** statement assigns numerical and character values to assigned names. The values of assigned names can be varied during statement looping. Assigned names can appear in numerical expressions which define the values of numerical and array parameters.

They can also be used in character expressions to define the following:

- Titles
- Labels
- File specifications

RETURN Statement

The **RETURN** statement terminates further processing of input statements in a file. This statement is used to prevent processing of statements at the end of the command input file or a file read with a **CALL** statement.

STOP Statement

The **STOP** statement terminates execution of the program. It is used to terminate program execution from the command input file or a file read with a **CALL** statement.

IGNORE Statement

The **IGNORE** statement prevents processing of subsequent input statements in a file. This statement is used to ignore statements at the end of the command input file or a file read with a **CALL** statement.

Optimization

The following statements specify the optimization process:

- The **LOOP** statement is used to perform optimization by specifying the **OPTIMIZE** parameter.
- The assigned names which are varied, are defined by using the **OPTIMIZE** parameter on the **ASSIGN** statement.
- The targets are defined with the **EXTRACT** statement.

Optimization Process

An optimization process does the following:

- The statements in the optimization loop are processed repeatedly, varying the values of assigned names until the error in the specified targets is minimized.
- The state of the simulation is saved before the first pass through an optimization loop.

  The simulation is restored to this saved state at the beginning of each pass through the loop.
- When the optimization is complete, the following is printed:
  - The number of function evaluations
  - The total RMS error
  - The final values of assigned names
  - The final values and RMS error for each target
  - The derivatives of each target with respect to each assigned name

Sensitivity Analysis

The following statements are used to specify sensitivity analysis:

- The **LOOP** statement is used to perform sensitivity analysis by specifying the **SENSITIV** parameter.
- The assigned names which are varied, are defined by using the **SENSITIV** parameter on the **ASSIGN** statement.
- The targets are defined with the **EXTRACT** statement.

Sensitivity Analysis Process

A sensitivity analysis does the following:

- The statements in the sensitivity analysis loop are processed repeatedly, varying the value of each assigned name to calculate the variation in each target value.
- The state of the simulation is saved before the first pass through a sensitivity analysis loop.

  The simulation is restored to this saved state at the beginning of each pass through the loop.
- When the sensitivity analysis is complete, the following is printed:
  - The number of function evaluations
  - The values of assigned names and targets
  - The derivatives of each target with respect to each assigned name

TITLE

The **TITLE** statement specifies character strings which document your input and the program output.

```
TITLE
   [<c>]
```

Description

The character string associated with the first line of the **TITLE** statement is used in documenting printed output. The character string is also used as the default title string for output requested in subsequent **PLOT.1D**, **PLOT.2D**, or **PLOT.3D** statements.

There may be any number of **TITLE** statements present, and they may be located at any point in the input file.

See Also...

To further illustrate the **TITLE** statement, refer to:

- Input file *mdex1* in N-Channel MOSFET Examples, Chapter 4, Generation of the Simulation Structure on page 4-2
- Input file *mdex2* in NPN Bipolar Transistor Examples, Chapter 5, Generation of the Simulation Structure on page 5-1
- Several other examples

COMMENT

The **COMMENT** statement is used to specify character strings which document your input and the program output.

```
COMMENT
  [<c>]
OR
$
  [<c>]
```

Description

The character strings associated with **COMMENT** statements serve only to document the input file and are ignored by the program.

There can be any number of **COMMENT** statements present, and they can be located at any point in the input file. Note that blank lines can also be used to improve readability of the input.

See Also...

To further illustrate the **COMMENT** statement, refer to:

- *mdex1* in N-Channel MOSFET Examples, Chapter 4, Generation of the Simulation Structure on page 4-2
- *mdex2* inNPN Bipolar Transistor Examples, Chapter 5, Generation of the Simulation Structure on page 5-1
- Most other examples

OPTION

The **OPTION** statement allows you to set flags to obtain debugging information and CPU statistics, and to determine how the Medici version in use is configured.

```
OPTION
[G.DEBUG] [N.DEBUG] [ CPU.STAT [CPU.FILE=<c>] ] [I.ERROR]
[MAXNODES] [ SAVE.SOL [SOL.FILE=<c>] ]
```

Parameter	Type	Definition	Default
G.DEBUG	logical	Specifies that general debugging information is printed to the standard output.	The current value; initially false.
N.DEBUG	logical	Specifies that numerical parameter debugging information is printed to the standard output.	The current value; initially false.
CPU.STAT	logical	Specifies that a CPU profile of the solution process is being printed to the file specified by **CPU.FILE**.	The current value; initially false.
CPU.FILE	char	Specifies the identifier for the file to receive printed CPU information when the **CPU.STAT** parameter is specified.	*<base>.cpu* See Chapter 1, Execution Time Files on page 1-13.
I.ERROR	logical	Specifies that interactive input mode is entered if a fatal error occurs during program execution. This option allows diagnosis of the simulation to determine the cause of the error. The simulation should not be continued because the state of the simulation may not be reliable.	The current value; initially false.
MAXNODES	logical	Specifies that information is printed on the configuration of the program and the limits this configuration places on the number of nodes that can be simulated.	false
SAVE.SOL	logical	Specifies that the two most recent solutions are automatically saved in files, whose identifiers are specified by **SOL.FILE**. However, **SOLVE** statements with **OUT.FILE** specified save only the solutions identified by **OUT.FILE**.	The current value; initially false.
SOL.FILE	char	The identifier for the files that are saved when the **SAVE.SOL** parameter is specified. The characters "1" and "2" are alternately appended to this identifier whenever a new solution is available.	*<base>.sav*

Description

Specifying **MAXNODES** causes the program to write the following information to the standard output:

- The maximum nodes available
- How the program is configured
- The maximum number of nodes available for various types of analysis based on this configuration

HELP

The **HELP** statement prints information describing a statement and its associated parameters. A question mark (**?**) is a synonym for the **HELP** statement.

```
HELP
   [NAME=<c>] [ {PARAMETE=<c> | VERBOSE} ]
```

Parameter	Type	Definition	Default
NAME	char	Specifies the name of the statement for which information is printed.	none
PARAMETE	char	Specifies the name of the parameter for which information is printed describing the units, default values, and synonyms.	none
VERBOSE	logical	Specifies that information is printed describing the units, default values, and synonyms for all parameters in the statement.	false

Description

A **HELP** statement without parameters prints general information describing the **HELP** statement and the statements for which help is available.

If the **NAME** parameter is specified, information is printed describing the specified statement and its associated parameters. For example, the following statement prints help information describing the **ASSIGN** statement:

```
HELP   NAME=ASSIGN
```

Either the **PARAMETE** or **VERBOSE** parameter can be specified to print information describing the units, default values, and synonyms for parameters. If the **PARAMETE** parameter is specified, information is printed for the specified parameter. For example, the following statement prints help information for the **NAME** parameter on the **ASSIGN** statement:

```
HELP   NAME=ASSIGN   PARAMETE=NAME
```

If the **VERBOSE** parameter is specified, information is printed for all parameters in the statement. For example, the following statement prints help information describing the **ASSIGN** statement and all of its parameters:

```
HELP   NAME=ASSIGN   PARAMETE=VERBOSE
```

If the **HELP** statement is entered interactively, the help information is printed on your terminal. If the **HELP** statement is entered through the batch input file or through a **CALL** statement, the help information is printed on the standard output.

For more information see Obtaining Help on page 3-360.

CALL

The **CALL** statement enters additional statements into the input by either reading them from a file or copying them from the *currently available* input statements. These additional statements are called CALL contents.

```
CALL
{FILE=<c> | ( [FIRST=<c>] [LAST=<c>] [EXPAND] )} [ONCE] [PRINT]
```

Parameter	Type	Definition	Default
FILE	char	Specifies the identifier for the formatted input file from which the input statements are read.	none
FIRST	char	Specifies the line number of the first input statement being copied.	The current **CALL** statement.
LAST	char	Specifies the line number of the last input statement being copied.	The current **CALL** statement.
EXPAND	logical	Specifies that **CALL** statements being copied are converted to comments and the CALL contents associated with these **CALL** statements are copied. If the value of this parameter is false, **CALL** statements are copied in their original form and the CALL contents associated with these **CALL** statements are not copied.	true
ONCE	logical	Specifies that input statements are only entered the first time the **CALL** statement is processed during statement looping and remain unchanged afterward. If the value of this parameter is false, input statements are reentered each time the **CALL** statement is processed. This allows the input statements entered during a statement loop to be changed by varying the identifier or contents of the input file in the loop. This parameter has no effect if the **CALL** statement is not in a statement loop.	true
PRINT	logical	Specifies that the input statements entered by the **CALL** statement are printed on the standard output as part of the list of input statements. If the value of this parameter is false, only the **CALL** statement itself is printed.	true

Reading a CALL File

Specifying the **FILE** parameter causes the CALL contents to be read from the file identified by this parameter. For example, the following **CALL** statement reads input statements from the file *FILE1* and enters them after the **CALL** statement:

```
CALL  FILE=FILE1
```

Range Parameters

Specifying either or both of the **FIRST** and **LAST** parameters causes the CALL contents to consist of a range of statements. These are the *currently available* input statements between and including the statements identified by these parameters.

These statements must be located entirely before or after the current **CALL** statement. The current **CALL** statement cannot be included in the CALL contents.

The **I.PRINT** statement can be used to print the *currently available* input statements with their associated line numbers.

The **FIRST** and **LAST** parameters are intended primarily for use when the **CALL** statement is entered interactively. This allows previously entered statements to be easily repeated. For example, the following **CALL** statement copies the input statements from line 1/005 through line 1/008 and enters them after the **CALL** statement:

```
CALL   FIRST=1/5   LAST=1/8
```

Default Values

The default values for the **FIRST** and **LAST** parameters are chosen to simplify copying groups of statements immediately preceding or following the current **CALL** statement. Only the **FIRST** parameter is necessary to copy a group of statements immediately preceding the current **CALL** statement.

The following **CALL** statement copies the input statements from line 1/005 through the statement immediately preceding the **CALL** statement. It then enters them after the **CALL** statement:

```
CALL   FIRST=1/5
```

Only the **LAST** parameter is necessary to copy a group of statements immediately following the current **CALL** statement. If you do not specify either the **FIRST** or **LAST** parameter, the **FILE** parameter must be specified. If the value of the **FIRST** parameter is greater than the value of the **LAST** parameter, the values of these parameters are interchanged.

Statement Modification

Using the **FIRST** and **LAST** parameters to specify a range of statements to be copied causes some of these statements to be modified or removed in the following ways:

- **INTERACTIVE** and **BATCH** statements are converted to comments

  They help identify which statements in the CALL contents were entered interactively, however they are only processed the first time they are encountered.

- **HELP** and **I.PRINT** statements are removed

  They do not serve a useful purpose in the CALL contents and are only processed the first time they are encountered.

- **CALL** statements are converted to comments and the CALL contents are copied if *any* of the following conditions is satisfied:
  - The value of the **EXPAND** parameter is true.
  - The statement range does not include the **CALL** statement.
  - The statement range includes a portion, but not all, of the CALL contents.

- The CALL contents are not copied and the **CALL** statement is copied in its original form if *all* of the following conditions are satisfied:
  - The value of the **EXPAND** parameter is false.
  - The statement range includes the **CALL** statement.
  - The statement range includes either all or none of the CALL contents.

Nesting CALL Statements

The CALL contents entered **CALL** can contain other **CALL** statements. They may be nested in this way to a maximum of 15 levels, as long as the available input storage is not exceeded.

Any of these **CALL** statements can obtain input statements by reading them from files or by copying them from the *currently available* input statements.

The **CALL** statement can be used to simplify the repetition of groups of statements. A group of statements placed in a file can be entered through the **CALL** statement multiple times in a single input file or repeatedly in different input files.

The following is an example of a file named *FILE1*:

```
STMT1  PARM1=1
STMT2  PARM2=2
```

The following input statements enter the statements in *FILE1* at two locations in the same input file:

```
STMT3  PARM3=3
  CALL   FILE=FILE1
STMT4  PARM4=4
  CALL   FILE=FILE1
STMT5  PARM5=5
```

The statements in the above example are equivalent to the following input statements:

```
STMT3  PARM3=3
  STMT1  PARM1=1
  STMT2  PARM2=2
STMT4  PARM4=4
  STMT1  PARM1=1
  STMT2  PARM2=2
STMT5  PARM5=5
```

Generating Template Files

CALL can be combined with the **ASSIGN** statement and assigned names to generate template files with variable input values.

Construction Methods

Template files are constructed by replacing portions of the input with assigned names and numerical expressions including assigned names.

Examples of input that may be replaced are:

- Character strings
- Parameter names
- Parameter values

Example

The following is an example of a template file named *FILE2*:

```
STMT1  PARM1=@VAL1
```

The following input statements assign values to the assigned name *VAL1* and call the template file **FILE2**:

```
ASSIGN  NAME=VAL1  N.VALUE=5
CALL  FILE=FILE2
ASSIGN  NAME=VAL1  N.VALUE=10
CALL  FILE=FILE2
```

The preceding statements are equivalent to the following input statements:

```
STMT1  PARM1=5
STMT1  PARM1=10
```

Using Template Files

An input file that uses the template file must first include **ASSIGN** statements. The values of assigned names in the template file are established by the **ASSIGN** statements preceding the **CALL** statement in the input file.

The **ASSIGN** statement sets the values of assigned names appearing in the template file, and includes a **CALL** statement. **CALL** then enters the statements from the template file.

See Also...

To further illustrate the **CALL** statement, refer to input files:

- *mdex7b* in Template Examples, Chapter 8, Bipolar Structure Template on page 8-12
- *mdex7n* in Template Examples, Chapter 8, N-Channel LDD MOSFET Example on page 8-1
- *mdex7p* in Template Examples, Chapter 8, P-Channel MOSFET Example on page 8-7
- *mdex8a* in Energy Balance Examples, Chapter 9, Substrate Current Simulation in an LDD MOSFET on page 9-1
- *mdex8b* in Energy Balance Examples, Chapter 9, Bipolar Junction Transistor Example on page 9-7
- *mdex11* in Circuit Analysis Examples, Chapter 12, Generation of the Simulation Structure and Solutions on page 12-2

- *mdex14b* in Lattice Temperature Examples, Chapter 13, Bipolar Transistor Thermal Run-Away Analysis on page 13-7

INTERACTIVE

The **INTERACTIVE** statement starts interactive input mode, allowing statements to be entered interactively from your terminal. For more information see Interactive and Batch Input Modes on page 3-360.

```
INTERACTIVE
   [ONCE]
```

Parameter	Type	Definition	Default	Units
ONCE	logical	Specifies that interactive input mode is only started the first time the **INTERACTIVE** statement is processed during statement looping. This parameter has no effect if the **INTERACTIVE** statement is not in a statement loop.	False	

Starting Interactive Mode

Interactive input mode may be started through any of the following methods:

- An **INTERACTIVE** statement is entered through the batch input file.

  In this case, the program resumes processing statements from the batch input file when interactive input is terminated.
- A blank file specification is given for the command input file when prompted at the beginning of program execution.

  In this case, an **INTERACTIVE** statement is automatically added as the first input statement. All statements are entered interactively, and the program terminates when interactive input is terminated.
- A program execution error is encountered subsequent to the occurrence of an **OPTION** statement that specifies a true value for the **I.ERROR** parameter.

  In this case, the program resumes processing statements from the batch input file when interactive input is terminated.

The program indicates when interactive input mode is started by doing the following:

- Printing a message on the terminal
- Printing a three-character interactive input prompt identifying the program
- Awaiting input of statements

Terminating Interactive Mode

Interactive input mode can be terminated either by entering a **BATCH** statement or by entering an end-of-file during interactive input from your terminal. Typical end-of-file characters are CONTROL-*D* (*EOT*) and CONTROL-*Z* (*SUB*).

Entering Long Statements

A statement can be continued on a subsequent line by ending the current input line with a plus character (+). Continuation can be used repeatedly to generate input statements consisting of any number of input lines. The program indicates that continuation lines are expected by changing the interactive input prompt to "+>" until the statement is complete. A continued statement can be completed by not ending the last line with a plus or by the input of a blank line.

Entering Statements in Loops

When an **INTERACTIVE** statement is processed during statement looping, statements can be entered in either of the following two modes:

1. If the **ONCE** parameter is specified—One set of statements can be entered interactively the first time the **INTERACTIVE** statement is processed. These same statements are processed during subsequent passes through the loop.
2. If the **ONCE** parameter is not specified—A new set of statements can be entered interactively each time the **INTERACTIVE** statement is processed. The input statements entered interactively during the previous pass through the loop are replaced with the new set of interactively entered statements.

BATCH

The **BATCH** statement terminates interactive input mode.

```
BATCH
   [<c>]
```

Description

The **BATCH** statement can only be entered by direct interactive input. It cannot be entered through the batch input file or interactively through a **CALL** statement.

A **BATCH** statement is automatically added to the input when an end-of-file is encountered during interactive input from the terminal. Typical end-of-file characters are CONTROL-*D* (*EOT*) and CONTROL-*Z* (*SUB*). Interactive input mode can be terminated either with a **BATCH** statement or an end-of-file character.

The character strings associated with the **BATCH** statement are ignored by the program and serve only to document your input.

For more information see Interactive and Batch Input Modes on page 3-360.

I.PRINT

The **I.PRINT** statement prints the currently available input statements with their associated line numbers.

```
I.PRINT
   {( [FIRST=<c>] [LAST=<c>] ) | [ALL]} [EXPAND]
```

Parameter	Type	Definition	Default
FIRST	char	Specifies the line number of the first input statement being printed.	10 statements before the current **I.PRINT** statement.
LAST	char	Specifies the line number of the last input statement being printed.	10 statements after the current **I.PRINT** statement.
ALL	logical	Specifies that all input statements are printed.	false
EXPAND	logical	Specifies that **CALL** statements being printed are converted to comments and the CALL contents associated with these **CALL** statements are printed. If the value of this parameter is false, **CALL** statements are printed in their original form and the CALL contents associated with these **CALL** statements are not printed.	true

Range Parameters

The **FIRST**, **LAST**, and **ALL** parameters specify the range of statements printed by the **I.PRINT** statement. For example, the following statement prints the input statements from line 1/005 through line 1/008:

```
I.PRINT  FIRST=1/5  LAST=1/8
```

For more information see Output of Statement Information on page 3-362.

CALL Statements

CALL statements are converted to comments and the CALL contents are printed if *any* of the following conditions is satisfied:

- The value of the **EXPAND** parameter is true.
- The statement range does not include the **CALL** statement.
- The statement range includes a portion, but not all, of the CALL contents.

The CALL contents are not printed and the **CALL** statement is printed in its original form if *all* of the following conditions are satisfied:

- The value of the **EXPAND** parameter is false.
- The statement range includes the **CALL** statement.
- The statement range includes either all or none of the CALL contents.

Output

The output from the **I.PRINT** statement consists of the following:

- Current line number
- Loop counters
- Input statements with their associated line numbers.

The output may be printed on the terminal or the standard output. This is determined by which method the **I.PRINT** statement is entered.

- Terminal output—The **I.PRINT** statement is entered interactively.
- Standard output—The **I.PRINT** statement is entered through the batch input file or through a **CALL** statement.

I.SAVE

The **I.SAVE** statement saves input statements in a file.

```
I.SAVE
   FILE=<c> [NOW] [FIRST=<c>] [LAST=<c>] [EXPAND]
```

Parameter	Type	Definition	Default
FILE	char	Specifies the identifier for the formatted output file in which the input statements are saved.	none
NOW	logical	Specifies that the input statements are saved immediately. If the value of this parameter is false, the input statements are saved when program execution terminates.	True if **FIRST** or **LAST** is specified; otherwise, false.
FIRST	char	Specifies the line number of the first input statement being saved.	The first available statement.
LAST	char	The line number of the last input statement being saved.	The last available statement.
EXPAND	logical	Specifies that **CALL** statements being saved are converted to comments and the CALL contents associated with these **CALL** statements are saved. If the value of this parameter is false, **CALL** statements are saved in their original form and the CALL contents associated with these **CALL** statements are not saved.	true

Description

The primary use for the **I.SAVE** statement is to save input statements that are entered interactively.

The **I.SAVE** output file can be used later as a batch input file or can be input using the **CALL** statement. These files can be used in the same program execution or in a subsequent execution.

For more information see Output of Statement Information on page 3-362.

Saving Input Statements

When the **I.SAVE** statement is entered with a false value for the **NOW** parameter, the **FILE** parameter specifies the identifier of a single file in which part or all of the input statements are saved when program execution terminates.

For example, the following statement saves all input statements in the *FILE1* file when program execution terminates:

```
I.SAVE  FILE=FILE1
```

When the **I.SAVE** statement is entered with a true value for the **NOW** parameter, part or all of the currently available input statements are saved in the file identified by the **FILE** parameter.

For example, the following statement immediately saves the input statements from line 1/005 through line 1/010 in the *FILE1* file:

```
I.SAVE  FILE=FILE1  FIRST=1/5  LAST=1/10
```

Saving a Range of Statements

The default values of **FIRST** and **LAST** parameters save all the input statements. They can be modified to a range of statements to be saved.

Repeated I.SAVE

Repeated **I.SAVE** statements with a false value for the **NOW** parameter merely replace the file identifier and the statement range. At the termination of program execution, the specified range of input statements are saved in the file identified by 356

the last **I.SAVE** statement.

For example, the following statement saves the input statements from line 1/005 through line 1/010 in the *FILE1* file when program execution terminates:

```
I.SAVE  FILE=FILE1  FIRST=1/5  LAST=1/10  ^NOW
```

Statement Modification

Some statements are modified or removed before being saved in the following ways:

- **INTERACTIVE** and **BATCH** statements are converted to comments.

  They help to identify which statements in the saved output were entered interactively, but they are only processed when they are encountered in the original input.

- **HELP** and **I.PRINT** statements are removed.

  They do not serve a useful purpose in the saved output and are only processed when they are encountered in the original input.

- **CALL** statements are converted to comments and the CALL contents are saved if *any* of the following conditions is satisfied:

- The value of the **EXPAND** parameter is true.
- The statement range does not include the **CALL** statement.
- The statement range includes a portion, but not all, of the CALL contents.

- The CALL contents are not saved and the **CALL** statement is saved in its original form if *all* of the following conditions are satisfied:
  - The value of the **EXPAND** parameter is false.
  - The statement range includes the **CALL** statement.
  - The statement range includes either all or none of the CALL contents.

IF

The **IF** statement begins a sequence of one or more conditionally processed input statement blocks and its first block.

```
IF
   [COND]
```

Parameter	Type	Definition	Default
COND	logical	Specifies that the block of input statements between the **IF** statement and the next **ELSE** statement or **IF.END** statement are being processed. If the value of this parameter is false, the block of input statements is not processed.	true

Specifying Block Sequence

- The **IF** statement defines the beginning of a sequence of conditionally processed blocks of statements.
- An **IF.END** statement is used to indicate the end of the sequence of statement blocks.
- The first statement block in the sequence begins with the **IF** statement, while subsequent statement blocks begin with **ELSE** statements.
- Each statement block ends with either an **ELSE** statement or the **IF.END** statement.

Matching and Nesting

- Each **IF** statement must be paired with a matching **IF.END** statement, with possibly intervening **ELSE** statements.
- **IF** and **IF.END** statements must independently match in statement loops, outside of statement loops, in input entered while in interactive input mode, and in input entered through the batch input file.
- Pairs of **IF** and **IF.END** statements can be nested to a maximum depth of 20 levels.

Processing

At most, one statement block in a sequence of blocks is processed. The statement block processed is the first in the sequence with a true value for the **COND** parameter on the **IF** or **ELSE** statement that begins the block. None of the statement blocks in a sequence is processed if the **IF** statement and all **ELSE** statements in the sequence have a false value for the **COND** parameter.

Example

The following example illustrates the use of the **IF**, **ELSE**, and **IF.END** statements to enter the name of an object, test for recognized names, and output information regarding the object:

```
ECHO "Specify the object shape"
ASSIGN NAME=SHAPE  C.VALUE="none"  PROMPT="shape="

IF    COND=(@SHAPE="triangle")
  ECHO  "3 sides"
ELSE  COND=(@SHAPE="none")
  ECHO  "shape not specified"
ELSE
  ECHO  "invalid shape"
IF.END
```

The following output is generated for various inputs provided to the **ASSIGN** statement:

```
shape=triangle
3 sides

shape=
shape not specified

shape=rectangle
invalid shape
```

See Also...

To further illustrate the **IF** statement, refer to input files:

- *mdex1f* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Fast Interface States on page 4-19
- *mdexlt* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Band-to-Band Tunneling on page 4-25
- *mdex8a* in Energy Balance Examples, Chapter 9, Substrate Current Simulation in an LDD MOSFET on page 9-1
- *mdex8b* in Energy Balance Examples, Chapter 9, Bipolar Junction Transistor Example on page 9-7

ELSE

The **ELSE** statement terminates a conditionally processed input statement block began with a previous **IF** or a previous **ELSE** statement and begins a new conditionally processed input statement block.

```
ELSE
   [COND]
```

Parameter	Type	Definition	Default
COND	logical	Specifies that the block of input statements between the **ELSE** statement, and the next **ELSE** statement or **IF.END** statement are being processed. If the value of this parameter is false, the block of input statements is not processed.	true

Description

The **ELSE** statement defines the beginning of one statement block in a sequence of conditionally processed blocks of statements begun by an **IF** statement. The statement block is terminated either by another **ELSE** statement or by the **IF.END** statement that terminates the sequence of statement blocks.

The block of statements is processed if the value of the **COND** parameter is true and no previous statement blocks in the sequence have been processed.

See Also...

To further illustrate the **ELSE** statement, refer to input files:

- *mdex1f* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Fast Interface States on page 4-19
- *mdex1t* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Band-to-Band Tunneling on page 4-25
- *mdex8a* in Energy Balance Examples, Chapter 9, Substrate Current Simulation in an LDD MOSFET on page 9-1
- *mdex8b* in Energy Balance Examples, Chapter 9, Bipolar Junction Transistor Example on page 9-7

IF.END

The **IF.END** statement terminates sequences of conditionally processed input statement blocks associated with the **IF** statement.

IF.END
[<c>]

The character strings associated with the **IF.END** statement are ignored by the program and serve only to document your input.

See Also…

To illustrate the **IF.END** statement, refer to input files:

- *mdex1f* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Fast Interface States on page 4-19
- *mdex1t* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Band-to-Band Tunneling on page 4-25
- *mdex8a* in Energy Balance Examples, Chapter 9, Substrate Current Simulation in an LDD MOSFET on page 9-1
- *mdex8b* in Energy Balance Examples, Chapter 9, Bipolar Junction Transistor Example on page 9-7

LOOP

The **LOOP** statement begins an input statement loop and specifies the number of times to process the statements in the loop. Optimization or sensitivity analysis can be performed by a single loop. The values of numerical and array parameters and assigned names can be varied on statements in loops.

```
LOOP
   [STEPS=<n>] [PRINT] [ {OPTIMIZE | SENSITIV} ]
```

Parameter	Type	Definition	Default
STEPS	number	Specifies the maximum number of passes through the loop. The statements between the **LOOP** statement and its matching **L.END** statement are processed once during each pass through the loop. The loop terminates when the number of passes equals the value of the **STEPS** parameter. If **OPTIMIZE** or **SENSITIV** is specified, the loop also terminates when the optimization or sensitivity analysis is completed. The value of the **STEPS** parameter must be a positive integer.	100 if **OPTIMIZE** or **SENSITIV** are specified, otherwise 1
PRINT	logical	Specifies that the values of parameters and assigned names that vary under control of this loop level are printed each time they are varied. For numerical and array parameters, the statement name, line number, loop level, loop counter, parameter name, and parameter value are printed before processing the statement containing the varied parameter value. For assigned names, the assigned name, line number, loop level, loop counter, and assigned value are printed after processing the **ASSIGN** statement defining the varied assigned name value.	false
OPTIMIZE	logical	Specifies that this loop performs optimization of assigned name values that are defined by **ASSIGN** statements specifying the **OPTIMIZE** parameter. Only one loop in a nest of loops can specify the **OPTIMIZE** or **SENSITIV** parameters.	false
SENSITIV	logical	Specifies that this loop performs sensitivity analysis for assigned name values that are defined by **ASSIGN** statements specifying the **SENSITIV** parameter. Only one loop in a nest of loops can specify the **OPTIMIZE** or **SENSITIV** parameters.	false

Statement Looping

The **LOOP** statement defines the beginning of a sequence of statements to be processed repeatedly. An **L.END** statement is used to indicate the end of the statement sequence

Loop Processing

The statement sequence is processed the number of times specified by one of the following:

- The **STEPS** parameter is specified

The **L.MODIFY** statement can be used to modify the original values of the **STEPS** and **PRINT** parameters specified in the **LOOP** statement.

- If either **OPTIMIZE** or **SENSITIV** is specified

  In this case, the processing continues until the optimization or sensitivity analysis is completed.

Repeated processing of a statement sequence is similar to when the sequence of statements is explicitly repeated multiple times in the program input. The difference is that during statement looping a statement in the sequence is referenced by the same input line number during each pass through the loop.

Each **LOOP** statement must be paired with a matching **L.END** statement. **LOOP** and **L.END** statements must match input entered while in interactive input mode and also must independently match input entered through the batch input file.

Matching and Nesting

Nested statements have the following properties:

- Pairs of **LOOP** and **L.END** statements can be nested to a maximum depth of ten levels.
- Nested loop levels are numbered starting with 1 at the outer loop and increasing to a value less than or equal to 10 at the inner loop.

The current loop level associated with a statement is the level of the innermost loop which contains that statement. Only one loop in a nest of loops may specify the **OPTIMIZE** or **SENSITIV** parameters.

Loop Counters

Each loop uses a unique counter which varies from 1 to the value specified by the **STEPS** parameter for that loop. The loop counters are processed by Medici through the following steps:

1. Before the first **LOOP** statement is processed, the loop level is initialized to zero.
2. When a **LOOP** statement is processed, the loop level is incremented by one and the counter associated with the loop is initialized to one.

   If optimization or sensitivity analysis is being performed, the state of the simulation is saved.
3. The counter remains constant during processing of all statements following the **LOOP** statement until the matching **L.END** statement is encountered.
4. When the matching **L.END** statement is processed, the counter is incremented by one and compared to the **STEPS** parameter specified on the matching **LOOP** statement.
5. If the counter exceeds **STEPS** or if optimization or sensitivity analysis is completed, the following occurs:
   a. The loop terminates.
   b. The loop level is decremented by one.

c. The next statement processed is that following the **L.END** statement.

If the loop does not terminate, refer to Step 6.

6. If the loop does not terminate, the following occurs:
   a. The loop level remains unchanged
   b. The next statement processed is that following the matching **LOOP** statement.
   c. If optimization or sensitivity analysis is being performed, the previously saved state of the simulation is restored.

Parameter Values in Loops

The values of numerical and array parameters on statements in loops may be varied by either a constant difference or a constant ratio between successive passes through a loop. This is specified by using a more general form for a parameter value as follows:

<start>:<increment>:<level>

where:

- *<start>* is the initial value of the parameter for the first pass through the loop
- *<increment>* is the difference or ratio between the parameter values for successive passes
- *<level>* identifies the loop level which controls variation of the parameter.

Syntax

The colon (:) is used to separate portions of the value specification and can be preceded or followed by any number of spaces. The first colon can only be used if the *<increment>* is specified and the second colon can only be used if the *<level>* is specified.

Parameter values which are varied in this manner cannot be controlled by a loop level that is performing optimization or sensitivity analysis.

Parameter Varying Loop

The variation of a parameter value may be controlled by any loop at a level less than or equal to the current loop level. The counter associated with the controlling loop determines the value of the parameter.

For example, in the following input, the **PARM1**, **PARM2**, and **PARM3** parameters are controlled by loop levels 1, 2, and 3, respectively:

```
LOOP  STEPS=2
  LOOP  STEPS=2
    LOOP  STEPS=2
      STMT  PARM1=0:1:1  PARM2=0:1:2  PARM3=0:1:3
    L.END
  L.END
L.END
```

Parameters in Value Specification

The <*start*> is the only required portion of the parameter value specification and can consist of any valid numerical expression. If the <*increment*> is not specified, the parameter value remains constant and the <*level*> cannot be specified. The <*increment*> is ignored if the parameter value occurs outside of loops. In this case, the <*level*> can only be specified if its value is zero or negative.

Value <*increment*>

The <*increment*> can consist of any valid numerical expression. If the parameter value is to vary by a constant ratio, the <*increment*> must be nonzero and its first character must be an asterisk (*). In this case, the parameter value is determined by:

value = <*start*> ***** <*increment*>****(count-1)**

where count is the counter for the loop level specified by <*level*>. If the asterisk is not present, the parameter value varies by a constant difference and the <*increment*> can be any value. In this case, the parameter value is determined by:

value = <*start*> **+** <*increment*>***(count-1).**

Value <*level*>

The <*level*> can consist of any valid numerical expression. The <*level*> is truncated to an integer, after which it must be less than or equal to the current loop level. The following table illustrates the values which may be used for <*level*>, and the effects those values have on the process.

.

Table 3-32 Values and Effects for *<level>*

Value	Effect
Not specified	The value defaults to the current loop level, causing the parameter value to be varied each time the statement is processed.
0 (Zero)	The parameter variation is disabled and the parameter remains constant. This allows the variation of a parameter to be disabled without requiring deletion of the <*increment*>.
Positive	This value directly specifies the loop level which controls the parameter value.
Negative	The loop level which controls the parameter value is the sum of the current loop level and <*level*>. If this sum is not positive the parameter variation is disabled as if the <*level*> were zero. This feature allows a parameter to be controlled by a loop one or more levels lower than the current loop level without explicitly specifying the loop level.

Single Loop Example

The following example illustrates the use of statement looping for a single loop and a statement containing the numerical parameters **PARM1** and **PARM2**:

```
LOOP  STEPS=3
  STMT  PARM1=0:5  PARM2=1:*-1
L.END
```

where under control of the loop:

- The parameter **PARM1** assumes values of 0, 5, and 10.
- The parameter **PARM2** assumes values of 1, -1, and 1.

Complex Example

The following example illustrates a more complex use of statement looping for a statement containing the numerical parameter **PARM1** and the array parameter **PARM2**:

```
LOOP  STEPS=3
  LOOP  STEPS=2
   ASSIGN  NAME=BASE  N.VALUE=10  RATIO=2
   STMT  PARM1=0:5  PARM2=(10:*2:1 , @BASE+10:10:-1)
  L.END
L.END
```

where:

- The assigned name *BASE* starts with the value of 10 and varies by a constant ratio of 2 under control of the inner loop.
- The numerical parameter **PARM1** starts with the value of 0 and varies by a constant difference of 5 under control of the inner loop.
- Element 1 of the array parameter **PARM2** starts with the value of 10 and varies by a constant ratio of 2 under control of the outer loop. (The loop level is specified explicitly as 1.)

- Element 2 of the array parameter **PARM2** starts with the value of *BASE*+10 and varies by a constant difference of 10 under control of the outer loop. (The loop level is specified as 1 less than the current level of 2.)
- The variation of the value of the assigned name *BASE* under control of the inner loop causes the starting value for element 2 of the array parameter **PARM2** to vary between 20 and 30.

The values of the loop counters and varied parameter values, and assigned names during the statement processing are as follows:

count #1	count #2	PARM1	PARM2 (1)	BASE	PARM2 (2)
1	1	0	10	10	20
1	2	5	10	20	30
2	1	0	20	10	30
2	2	5	20	20	40
3	1	0	40	10	40
3	2	5	40	20	50

See Also...

To further illustrate the **LOOP** statement refer to input files:

- *mdex1f* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Fast Interface States on page 4-19
- *mdex1t* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Band-to-Band Tunneling on page 4-25
- *mdex8a* in Energy Balance Examples, Chapter 9, Substrate Current Simulation in an LDD MOSFET on page 9-1
- *mdex8b* in Energy Balance Examples, Chapter 9, Bipolar Junction Transistor Example on page 9-7

L.MODIFY

The **L.MODIFY** statement modifies the processing of a currently active statement loop associated with a **LOOP** statement.

```
L.MODIFY
   [LEVEL=<n>] [STEPS=<n>] [ {NEXT | BREAK} ] [PRINT]
```

Parameter	Type	Definition	Default
LEVEL	number	Specifies the loop level associated with the **LOOP** statement for which processing is being modified. The value of this parameter must be less than or equal to the current loop level. If the value of this parameter is zero, the modification of statement processing is disabled. If the value of this parameter is negative, the loop level used is the sum of this parameter value and the current loop level.	The current loop level.
STEPS	number	Specifies the number of times the statements between the **LOOP** statement and its matching **L.END** statement are processed for the specified loop level. The value of this parameter must be a positive integer and may be less than or equal to the current value of the loop counter for the specified loop level. A value of 1 prevents any subsequent passes through the loop from being performed.	The current value for the specified loop level.
NEXT	logical	Specifies that the next statement processed is the **L.END** statement for the specified loop level. The statements between the **L.MODIFY** statement and the **L.END** statement for the specified loop level are not processed during this pass through the loop.	false
BREAK	logical	Specifies that the next statement processed is the statement following the **L.END** statement for the specified loop level. No subsequent passes through the loop are performed. The statements between the **L.MODIFY** statement and the **L.END** statement for the specified loop level are not processed during this pass through the loop.	false
PRINT	logical	Specifies that the values of parameters and assigned names which vary under control of the specified loop level are printed each time they are varied. For numerical and array parameters, the statement name, line number, loop level, loop counter, parameter name, and parameter value are printed before processing the statement containing the varied parameter value. For assigned names, the assigned name, line number, loop level, loop counter, and assigned value are printed after processing the **ASSIGN** statement defining the varied assigned name value.	The current value for the specified loop level.

Description

The **L.MODIFY** statement modifies the processing of a currently active statement loop associated with a **LOOP** statement.The **L.MODIFY** statement can be used to modify the number of passes through a loop and whether values of parameters and assigned names which vary under control of a loop are printed. The **L.MODIFY**

statement can also be used to skip the remaining statements (**NEXT**) or terminate the specified loop (**BREAK**).

Modifying Passes Through the Loop

Based on the results of previous statements, you may choose to modify the number of subsequent passes through the loop.

An **L.MODIFY** statement specifying the **STEPS** parameter can be used to increase or decrease the total number of passes through the loop. If the value of the **STEPS** parameter is modified so it is less than or equal to the current value of the loop counter, no subsequent passes through the loop are performed.

For example, the following statement sets the number of passes through the current loop level to 5:

```
L.MODIFY  STEPS=5
```

L.END

The **L.END** statement terminates input statement loops associated with the **LOOP** statement.

```
L.END
   [BREAK] [ALL]
```

Parameter	Type	Definition	Default
BREAK	logical	Specifies that the loop terminates after the first pass.	false
ALL	logical	Specifies that all loops currently available for termination are terminated.	false

Preventing Subsequent Passes

The **BREAK** parameter can be used to disable multiple passes through a loop. It is intended primarily for use when the **L.END** statement and statements in the loop are entered interactively.

Based on the results of the first time the statements in the loop are processed, you can choose to prevent subsequent passes through the loop by specifying the **BREAK** parameter when the **L.END** statement is entered.

Terminating All Loops

The **ALL** parameter can be used to terminate all loops currently in effect with a single **L.END** statement.

LOOP and **L.END** statements must independently match in input entered while in the following modes:

- Interactive input mode

  L.END statements specifying the **ALL** parameter entered while in interactive input mode terminate only loops entered while in interactive input mode.
- Batch input file

  L.END statement specifying the **ALL** parameter entered through the batch input file, cause all loops currently in effect to be terminated.

The following example illustrates the termination of three loop levels with a single **L.END** statement:

```
LOOP  STEPS=2
  LOOP  STEPS=2
    LOOP  STEPS=2
      STMT
    L.END  ALL
```

If the **ALL** parameter is not specified, three consecutive **L.END** statements are required.

See Also...

To further illustrate the **L.END** statement, refer to input files:

- *mdex1f* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Fast Interface States on page 4-19
- *mdex1t* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Band-to-Band Tunneling on page 4-25
- *mdex8a* in Energy Balance Examples, Chapter 9, Substrate Current Simulation in an LDD MOSFET on page 9-1
- *mdex8b* in Energy Balance Examples, Chapter 9, Bipolar Junction Transistor Example on page 9-7

ASSIGN

The **ASSIGN** statement assigns values to an assigned name.

```
ASSIGN

   {  ( NAME=<c> [PRINT]
       {  ( N.VALUE=<a> [ {DELTA=<n> | RATIO=<n>} ] )
         | ( N.VALUE=<a> {OPTIMIZE | SENSITIV} LOWER=<n> UPPER=<n> )
         | ( L.VALUE=<a> )
         | ( C.VALUE=<c> [DELTA=<n>] )
         | ( [C1=<c>] [C2=<c>] [C3=<c>] [C4=<c>] [C5=<c>]
             [C6=<c>] [C7=<c>] [C8=<c>] [C9=<c>] [C10=<c>]
           )
        }
        [E.NAME=<c>] [PROMPT=<c>] [LEVEL=<n>]
     )
    | ( PRINT [INITIAL] [NAME=<c>] )
   }
```

Parameter	Type	Definition	Default
NAME	char	Specifies the assigned name to which a value is being assigned or for which the current value is printed. The name must consist only of letters, digits, and periods (.), and cannot exceed eight characters.	none
PRINT	logical	Specifies that the current values of assigned names are printed. If the **NAME** parameter is specified, only the value of the specified assigned name is printed. If the **INITIAL** parameter is specified, only the values of initially assigned names are printed.	false
N.VALUE	array	Specifies the numerical value(s) assigned to the assigned name. If a single value is specified and neither **OPTIMIZE** nor **SENSITIV** is specified, the **DELTA** or **RATIO** parameters can be specified to vary the value of the assigned name. If multiple values are specified, the value of the assigned name is varied by choosing successive values from the list of values specified with this parameter. Only a single value can be specified if **OPTIMIZE** or **SENSITIV** is specified. At most 100 values can be defined with this parameter. The value(s) specified with this parameter can be replaced by one or more values specified with the **PROMPT** or **E.NAME** parameters.	none
DELTA	number	Specifies the constant difference by which the value of the assigned name is varied. This parameter is only allowed if the **C.VALUE** parameter is specified or if a single value is specified with the **N.VALUE** parameter and neither **OPTIMIZE** nor **SENSITIV** is specified.	none
RATIO	number	Specifies the constant ratio by which the value of the assigned name is varied. The value of this parameter must be nonzero. This parameter is only allowed if a single value is specified with the **N.VALUE** parameter and neither **OPTIMIZE** nor **SENSITIV** is specified.	none

Parameter	Type	Definition	Default
OPTIMIZE	logical	Specifies that the value of the assigned name is controlled by an optimization loop.	false
SENSITIV	logical	Specifies that the value of the assigned name is controlled by a sensitivity analysis loop.	false
LOWER	number	Specifies the lower bound for the value of the assigned name during optimization or sensitivity analysis.	none
UPPER	number	Specifies the upper bound for the value of the assigned name during optimization or sensitivity analysis.	none
L.VALUE	array	Specifies the logical value or values assigned to the assigned name. If multiple values are specified, the value of the assigned name is varied by choosing successive values from the list of values specified with this parameter. At most 100 values may be defined with this parameter. The value(s) specified with this parameter may be replaced by one or more values specified with the **PROMPT** or **E.NAME** parameters.	none
C.VALUE	char	Specifies the character value assigned to the assigned name. The value specified with this parameter may be replaced by a value specified with the **PROMPT** or **E.NAME** parameters.	none
C1	char	Specifies the first in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none
C2	char	Specifies the second in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none
C3	char	Specifies the third in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none
C4	char	Specifies the fourth in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none
C5	char	Specifies the fifth in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none
C6	char	Specifies the sixth in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none
C7	char	Specifies the seventh in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none

Parameter	Type	Definition	Default
C8	char	Specifies the eighth in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none
C9	char	Specifies the ninth in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	none
C10	char	Specifies the tenth in a list of character values assigned to the assigned name. The value of the assigned name is varied by choosing successive values from the list of values specified with the parameters **C1** through **C10**.	
E.NAME	char	Specifies the name of an environment variable containing an alternative to the value specified by the **N.VALUE**, **L.VALUE**, or **C.VALUE** parameter. If the environment variable is not set or its value is blank, the value specified by the **N.VALUE**, **L.VALUE**, or **C.VALUE** parameter is used. This parameter is only allowed with the **N.VALUE**, **L.VALUE**, and **C.VALUE** parameters. The value(s) specified with this parameter can be replaced by one or more values specified with the **PROMPT** parameter.	none
PROMPT	char	Specifies the character string used to prompt you for interactive input of an alternative to the value specified by the **N.VALUE**, **L.VALUE**, **C.VALUE**, or **E.NAME** parameter. If this character string is blank, ">" is used instead. This character string is output on your terminal and the alternative value is read from the terminal input. If the character string read from the terminal input is blank, the value specified by the **N.VALUE**, **L.VALUE**, **C.VALUE**, or **E.NAME** parameter is used. This parameter is only allowed with the **N.VALUE**, **L.VALUE**, and **C.VALUE** parameters.	none
LEVEL	number	Specifies the loop level which controls variation of the value of the assigned name. The value of this parameter must be less than or equal to the current loop level. If the value of this parameter is zero the variation of the assigned name is disabled. If the value of this parameter is negative the loop level used is the sum of this parameter value and the current loop level.	The current loop level.
INITIAL	logical	Specifies that the values of initially assigned names are printed.	false

Description

The **ASSIGN** statement defines an assigned name and assigns values to an assigned name. A maximum of 200 assigned names can be defined.

See Also...

To further illustrate the **ASSIGN** statement, refer to:

- Input file *mdex1a* in N-Channel MOSFET Examples, Chapter 4, Generation of the Simulation Structure on page 4-2
- Input file *mdex1f* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Fast Interface States on page 4-19
- Input file *mdex1t* in N-Channel MOSFET Examples, Chapter 4, Analysis Including Band-to-Band Tunneling on page 4-25

- Several other examples

Definitions

The definition of an assigned name can be repeatedly changed using successive **ASSIGN** statements. The definition established by execution of an **ASSIGN** statement remains in effect until the one of the following occurs:

- The definition is changed by execution of another **ASSIGN** statement.

or

- By execution of the same **ASSIGN** statement during a subsequent pass through a statement loop

Values

An assigned name can be given one of the following type of values:

- Numerical
- Logical
- Character

These values can be constant or can vary in statement loops. If the value of an assigned name is varied during statement looping, it is given a new value during each pass through the loop that controls its variation.

Mixed Type Usage

Assigned names with different types of values can be used in the following ways:

- Assigned names with numerical and logical values can be used interchangeably in numerical and character expressions.
- Assigned names with character values can be used in numerical expressions as arguments to relational operators, logical functions, and conversion functions.
- Assigned names with character values can also be used to specify statement names and one or more complete parameter name/value pairs.

  This is illustrated in the last example at the end of this section.

Numerical Values

The **N.VALUE** parameter is used to assign numerical values to the assigned name. There are two methods of specifying numerical values, each of which have different effects on the program. Numerical values may be specified in the following ways:

- If a single value is specified—The value of the assigned name can be varied during statement looping either by a constant increment by specifying the **DELTA** parameter or by a constant ratio by specifying the **RATIO** parameter.
- If multiple values are specified—The **DELTA** and **RATIO** parameters can not be specified. In this case the **N.VALUE** parameter specifies a list of values

from which successive values are taken during each pass through a statement loop. After the last value in the list is taken, the sequence begins again with the first value in the list.

Logical Values

The **L.VALUE** parameter is used to assign logical values to the assigned name.

- If multiple values are specified—The **L.VALUE** parameter specifies a list of values from which successive values are taken during each pass through a statement loop. After the last value in the list is taken, the sequence begins again with the first value in the list.

Character Values

The **C.VALUE** parameter is used to assign a character value to the assigned name.

- If **DELTA** parameter is not specified—The value of the assigned name is constant.
- If **DELTA** parameter is specified—Its value is truncated to an integer and used to increment the value of the assigned name during statement looping.

Incrementing

The incrementing of character values is primarily useful for varying file identifiers and title strings during statement looping. The increment processes are described in the following:

- The increment value is specified by the **DELTA** parameter.
- When a character value is incremented, characters in any of the character sequences "0-9", "a-z", and "A-Z" can be changed, while other characters are left unchanged.
- A character is always changed to another character in the same character sequence.
- A character value is incremented by starting with the rightmost character in the value and moving forward or backward through the character sequence containing it.
- Each time either end of the sequence is passed, the next character to the left in the value is changed by moving forward or backward by one character.

Increment Example

For example, the following input statements assign character values using an increment of 4 to the *NAME1* assigned name:

```
LOOP  STEPS=6
  ASSIGN  NAME=NAME1  C.VALUE=aa.0  DELTA=4
L.END
```

The above input statements result in the *NAME1* assigned name assuming the following sequence of character values:

```
aa.0
aa.4
aa.8
ab.2
ab.6
ac.0
```

The parameters **C1** through **C10** are used to assign one of a list of character values to the assigned name. These parameters specify a list of from 1 to 10 values from which successive values are taken during each pass through a statement loop. After the last value in the list is taken, the sequence begins again with the first value in the list.

Assigned Names in Optimization

A maximum of 20 assigned names can be defined using the **OPTIMIZE** o **SENSITIV** parameters in one optimization or sensitivity analysis loop.

- If **OPTIMIZE** or **SENSITIV** is specified—The initial value of the assigned name is specified with the **N.VALUE** parameter. The optimization or sensitivity analysis loop determines appropriate values for the assigned name for all passes through the loop except the first.
- **LOWER** and **UPPER** parameters specify the range of allowed values for the assigned name

  It is important to choose values for these parameters that are as close as possible to the value specified for **N.VALUE**. This maximizes the efficiency and accuracy of the optimization and sensitivity analysis.

Overriding Assigned Values

The program provides two ways to override specified values.

Environment Variable

The **E.NAME** parameter specifies the name of an environment variable. If the specified environment variable is set, its value is used instead of the value specified by the following parameters:

- **N.VALUE**, **L.VALUE**, or **C.VALUE**

For example, the following input statement assigns the value "original" to the *NAME1* assigned name if the *NEWNAME* environment variable is not set:

```
ASSIGN  NAME=NAME1  C.VALUE="original"  E.NAME=NEWNAME
```

Assigning the *NEWNAME* environment variable to "new" before executing Medici, causes the above input statement to assign the value "new" to the assigned name *NAME1*.

Interactive Input Prompt

The **PROMPT** parameter specifies a character string being used to prompt you for interactive input of the assigned name value from your terminal. If the input string is not blank, its value is used instead of the value specified by the following parameters:

- **N.VALUE**, **L.VALUE**, **C.VALUE**, or **E.NAME**

For example, the following input statement uses the "INPUT>" character string to prompt for the value of the *NAME1* assigned name:

```
ASSIGN  NAME=NAME1  C.VALUE="original"  PROMPT="INPUT>"
```

if:

- The input provided in response to the prompt is blank, the statement assigns the value "original" to the *NAME1* assigned name
- The response to the prompt is "new", then the above statement assigns the value "new" to the *NAME1* assigned name

Examples

A maximum of 200 assigned names can be defined. An assigned name is referenced in an input statement by preceding the name with "@."

Simple Example

The following input statements assign values to the *NAME1* and *NAME2* assigned names and use these assigned names to define the values of the parameters **PARM1** and **PARM2**:

```
ASSIGN  NAME=NAME1  C.VALUE="String"
ASSIGN  NAME=NAME2  N.VALUE=5
STMT  PARM1=@NAME1  PARM2=@NAME2
```

The above input statements are equivalent to the following input statement:

```
STMT  PARM1="String"  PARM2=5
```

Complex Example

The following input statements illustrate the use of **ASSIGN** statements in a loop:

```
ASSIGN  NAME=NAME2  C.VALUE="PARM2=2"
LOOP  STEPS=2
  ASSIGN  NAME=NAME1  C1="STMT1  PARM1=1"  C2="STMT2 PARM1=2"
  LOOP  STEPS=3
    ASSIGN  NAME=NAME3  C.VALUE="String0"  DELTA=1
    ASSIGN  NAME=NAME4  N.VALUE=5  RATIO=2
    ASSIGN  NAME=NAME5  N.VALUE=(10,20)
    @NAME1  @NAME2  PARM3=@NAME3  PARM4=@NAME4  PARM5=@NAME5
  L.END
L.END
```

where:

- *NAME2* assigned name is given a character value which specifies the parameter name and value for the numerical parameter **PARM2**.
- *NAME1* assigned name is given a list of 2 character values from which successive values are taken during each pass through the outer loop to specify the value for the statement name and the parameter name and value for the **PARM1** numerical parameter.
- *NAME3* assigned name is given a character value which is incremented by 1 during each pass through the inner loop to specify the value for the **PARM3** character parameter.
- *NAME4* assigned name is given a single numerical value which is multiplied by 2 during each pass through the inner loop to specify the value for the **PARM4** numerical parameter.
- *NAME5* assigned name is given a list of 2 numerical values from which successive values are taken during each pass through the inner loop to specify the value for the **PARM5** numerical parameter.

The above statement loops are equivalent to the following input statements:

```
STMT1  PARM1=1  PARM2=2  PARM3="String0"  PARM4=5   PARM5=10
STMT1  PARM1=1  PARM2=2  PARM3="String1"  PARM4=10  PARM5=20
STMT1  PARM1=1  PARM2=2  PARM3="String2"  PARM4=20  PARM5=10

STMT2  PARM1=2  PARM2=2  PARM3="String0"  PARM4=5   PARM5=10
STMT2  PARM1=2  PARM2=2  PARM3="String1"  PARM4=10  PARM5=20
STMT2  PARM1=2  PARM2=2  PARM3="String2"  PARM4=20  PARM5=10
```

ECHO

The **ECHO** statement outputs text to your terminal.

```
ECHO
   [<c>]
```

Description

The character strings associated with the first 15 lines of the **ECHO** statement are output to your terminal. The **ECHO** statement is used with the **PROMPT** parameter on the **ASSIGN** statement to provide interactive terminal input and output.

Example

The following statements prompt you for the number of loop steps and output the specified value:

```
ECHO  Input the number of loop steps
+     (default:10)
ASSIGN  NAME=TYPE  N.VALUE=10  PROMPT="steps="
ECHO  " "
+     @TYPE" steps were requested"
```

Steps Not Specified

These statements produce the following output if you do not specify a number of steps:

```
Input the number of loop steps
(default:10)
steps=

10 steps were requested
```

Steps Specified

If you specify 20 steps, the following output is produced:

```
Input the number of loop steps
(default:10)
steps=20

20 steps were requested
```

RETURN

The **RETURN** statement terminates further processing of input statements in a file.

```
RETURN
   [<c>]
```

Description

Input statements following a **RETURN** statement are not executed. This statement is used to prevent processing of statements at the end of the command input file or a file read with a **CALL** statement.

The **RETURN** statement is equivalent to a **STOP** statement when it occurs in the command input file, because no further statement processing occurs, causing program execution to terminate.

The character strings associated with the **RETURN** statement are ignored by the program and serve only to document your input.

STOP

The **STOP** statement terminates the execution of the program. **EXIT** and **QUIT** are synonyms for this statement.

```
STOP
   [<c>]
```

Description

Input statements following a **STOP** statement are not executed. This statement is used to terminate program execution from the command input file or a file read with a **CALL** statement.

The character strings associated with the **STOP** statement are ignored by the program and serve only to document your input. A **STOP** statement is not necessary to terminate program execution.

IGNORE

The **IGNORE** statement prevents processing of subsequent input statements in a file.

```
IGNORE
   [<c>]
```

Description

Any input statements following a **IGNORE** statement are printed, but are not checked for proper syntax and are not executed. This statement is used to ignore statements at the end of the command input file or a file read with a **CALL** statement.

The **IGNORE** statement is equivalent to a **STOP** statement when it occurs in the command input file, because no further statement processing occurs, causing program execution to terminate.

The character strings associated with the **IGNORE** statement are ignored by the program and serve only to document your input.

3.7 Old Statements

The functions performed by the following statements have been superseded by new additions to the program.

CAUTION

The statements in this section are presently supported, but their use is discouraged since they are likely to be eliminated from further versions of the program.

Statement	Definition	Page
CALCULATE	Calculates derived quantities from terminal data. Replaced by more general capabilities on the **EXTRACT** statement.	3-408
CHECK	Compares specified solutions. Replaced by more general capabilities on the **EXTRACT** statement.	3-411
DEPOSITION	Deposits carriers within a specified device cross-section. Use the **PHOTOGEN** statement instead.	3-412
.DC	Calculates steady state solution in circuit mode. Use parameters on the normal **SOLVE** statement: **ELEMENT**, **V.ELEMENT**, **VSTEP**, **NSTEP**.	3-414
.SAVE	Saves solutions in circuit mode. Use the normal **SAVE** statement instead.	3-419
.TRAN	Calculates transient solution in circuit mode. Use parameters on the normal **SOLVE** statement, i.e., **TSTEP** and **TSTOP**.	3-420

CALCULATE

The **CALCULATE** statement is used to define and calculate derived quantities involving I-V and AC terminal data.

CALCULATE

```
NAME=<c> ( A=<c> [SQRTA] ) [ {B=<c> | C=<n>} [SQRTB] ]
[ {SUM | DIFFEREN | RATIO | PRODUCT} ] [UNITS=<c>]
```

Parameter	Type	Definition	Default	Units
NAME	char	The name assigned to the derived quantity.	none	
A	char	The name of the **A**-quantity used in calculating a value for **NAME**. The choices include any of the available I-V or AC quantities (such as *I2*, *Y31*, *Time*, etc.) or any **NAME** defined on a previous **CALCULATE** statement.	none	
SQRTA	logical	Specifies that the square root of the **A**-quantity is calculated prior to performing any specified operation. To avoid taking the square root of a negative number, the absolute value of the **A**-quantity is used as the argument to the square root function.	false	
B	char	The name of the **B**-quantity used in calculating a value for **NAME**. The choices include any of the available I-V or AC quantities (such as *I2*, *Y31*, *Time*, etc.) or any **NAME** defined on a previous **CALCULATE** statement.	none	
C	number	A constant value that is used in place of the **B**-quantity.	none	arbitrary
SQRTB	logical	Specifies that the square root of the **B**-quantity or **C**-value is calculated prior to performing any specified operation. To avoid taking the square root of a negative number, the absolute value of the **B**-quantity or **C**-value is used as the argument to the square root function. **synonyms: SQRTC**	false	
SUM	logical	Specifies that the value assigned to **NAME** is calculated from **A**+**B** or **A**+**C**.	false	
DIFFEREN	logical	Specifies that the value assigned to **NAME** is calculated from **A**-**B** or **A**-**C**.	false	
RATIO	logical	Specifies that the value assigned to **NAME** is calculated from **A**/**B** or **A**/**C**.	false	

Parameter	Type	Definition	Default	Units
PRODUCT	logical	Specifies that the value assigned to **NAME** is calculated from **A*****B** or **A*****C**.	false	
UNITS	char	The units to assign to **NAME**. These are used to label axes when **NAME** is plotted.	Same as the units of the **A**-quantity if **SUM**, **DIFFEREN**, or no operation is specified; otherwise, none.	

Description

The **CALCULATE** statement allows derived quantities to be calculated from I-V and AC terminal data. At the conclusion of each bias or time point in a simulation, the value of all quantities defined to this point is calculated and printed to the standard output. The quantities defined on the **CALCULATE** statement can also be plotted in subsequent **PLOT.1D** statements.

Specifying Quantities

Quantities to be calculated are defined by specifying **NAME** and at least the **A**-quantity.

A Quantity

If no **B**-quantity or **C**-value is specified, the statement merely serves as a redefinition. For example, the statement

```
CALCULATE  NAME=Idrain  A=I4
```

where:

- *I4* is redefined to be Idrain.

  In subsequent **PLOT.1D** statements, *Idrain* may be used instead of *I4*.

B Quantity

If the **B**-quantity and one of the operations **SUM**, **DIFFEREN**, **RATIO**, or **PRODUCT** is specified, then **NAME** is calculated from **A**+**B**, **A**-**B**, **A**/**B**, or **A*****B**, respectively.

C Quantity

If a **C**-value is given instead of the **B**-quantity, then **NAME** is calculated from **A**+**C**, **A**-**C**, **A**/**C**, or **A*****C** for the four operations.

Calculating Square Roots

Square roots are calculated prior to performing any operation which may be specified. The following parameters are used to calculate square roots:

A quantity**—SQRTA**

B quantity or **C** value—**SQRTB**

Previous Definitions

Quantities can also be defined in terms of previous definitions as shown in the following example:

```
CALCULATE   NAME=Icoll   A=I1     B=I2     SUM
CALCULATE   NAME=Ibase   A=I3     B=I4     SUM
CALCULATE   NAME=Beta    A=Icoll  B=Ibase  RATIO
```

There may be up to 20 quantities defined with the **CALCULATE** statement.

CHECK

The **CHECK** statement compares a specified solution against the current solution, returning the maximum and average difference in electrostatic and quasi-Fermi potentials.

```
CHECK

   IN.FILE=<c> [IN.MESH=<c>]
```

Parameter	Type	Definition	Default	Units
IN.FILE	char	The identifier for the binary solution file to compare against the current solution.	none	
IN.MESH	char	The identifier for the binary file containing the mesh corresponding to the solution specified by **IN.FILE**.	current mesh	

Description

The **CHECK** statement is useful for comparing solutions that have been obtained on different generations of regrids. The comparison is made only for those grid points that are common to both meshes.

DEPOSITION

The **DEPOSITION** statement is used to deposit excess electrons and holes over a specified cross-section of the device.

```
DEPOSITION

   X.START=<n> Y.START=<n> ANGLE=<n>
   CONCENTR=<a> LENGTH=<a> WIDTH=<a> [ELECTRON] [HOLE]
```

Parameter	Type	Definition	Default	Units
X.START	number	The x coordinate of the beginning of the line along which excess carriers are deposited. **synonyms: X.0**	0.0	microns
Y.START	number	The y coordinate of the beginning of the line along which excess carriers are deposited. **synonyms: Y.0**	0.0	microns
ANGLE	number	The angle of the line along which excess carriers are deposited. A value of 0 is parallel to the positive horizontal axis. Angles increase in the clockwise direction.	90.	degrees
CONCENTR	array	The carrier concentrations to deposit in up to 20 contiguous boxes located along the line described with **X.START**, **Y.START**, and **ANGLE**. **CONCENTR** refers to the concentrations of both electrons and holes unless indicated otherwise by specifying **^ELECTRON** or **^HOLE**.	none	$\#/cm^3$
LENGTH	array	The lengths of up to 20 contiguous boxes in which excess carriers are deposited.	none	microns
WIDTH	array	The widths of up to 20 contiguous boxes in which excess carriers are deposited.	none	microns
ELECTRON	logical	Specifies that the deposited carriers are electrons.	true	
HOLE	logical	Specifies that the deposited carriers are holes.	true	

Description

The **DEPOSITION** statement is used to deposit excess carriers over a specified cross-section of the device. This statement is used, for example, to model the presence of excess carriers due to the passage of ionizing particles through the device. A transient simulation is performed after the deposition to study the effect of the deposited carriers on the device behavior.

Before a **DEPOSITION** statement is used, a valid solution must be available. The specified carrier concentrations are added to those of the available solution before the next solution is performed.

Specifying Density of Carrier Boxes

The excess carriers are specified by giving the density of carriers (**CONCENTR**) in up to 20 contiguous boxes along a specified line through the device. The line and box dimensions are defined by the following parameters:

- The line begins at the location specified with **X.START** and **Y.START**.
- The line is oriented in the direction specified with **ANGLE**.

  Both the length and width of the boxes may vary along the line.
- The box lengths are specified with **LENGTH**, and are measured parallel to the line.
- The box widths are specified with **WIDTH**, and are measured perpendicular to this line and are bisected by it.

Examples

The following statement specifies a region for deposited carriers that deposits 1e18/cm^3 electrons and holes in a box 0.5 microns wide and 3.0 microns long beginning at x=5 microns and y=0 microns. By default, the box is directed vertically downward (positive y-direction):

```
DEPOSIT   X.START=5  Y.START=0  CONCEN=1E18  LENGTH=3
          WIDTH=.5
```

The following example has the same overall size and orientation as the example above, but the concentration of electrons and holes is varied along the length of the region:

```
DEPOSIT   X.START=5  Y.START=0  CONCEN=(1E18,8E17,6E17)
+                               LENGTH=( 0.5, 1.0, 1.5)
+                                WIDTH=( 0.5, 0.5, 0.5)
```

.DC

Causes Medici to perform a steady-state analysis.

.DC

[<*srcname*> **VSTART=<n> VSTOP=<n> VINCR=<n> [UIC] [MULT=<n>] [RESET]]**

Parameter	Type	Definition	Default	Units
<*srcname*>	c	The voltage or current source which is to be stepped.	none	
VSTART	number	The starting voltage or current.	none	volts or amps
VSTOP	number	The ending voltage or current.	none	volts or amps
VINCR	number	The voltage or current increment to use.	none	volts or amps
UIC	logical	Causes Medici to force the voltages at certain nodes to specified values. These nodes and their values are specified using the **.IC** statement.	false	
MULT	number	Multiplicative factor for **VINCR**. After each bias step, **VINCR** is increased by the factor **MULT**. This parameter is useful when a range of bias points covering several orders of magnitude must be simulated (for example from 1e-9 Amps to 1e-2 Amps).	1.0	none
RESET	logical	Causes the value of the voltage or current source being stepped to return to its original value after the present **.DC** statement finishes. If this parameter is false, the DC value remains at its final value reached on the present **.DC** statement.	true	

SPICE Compatibility

To preserve SPICE compatibility, **.OP** is a synonym for **.DC**.

Example

```
.OP
.DC Vin VSTART=-1.0 VSTOP=1.0 VINCR=0.1 UIC
```

See Also... To further illustrate the **.DC** statement refer to input file *mdex14b* in Lattice Temperature Examples, Chapter 13, Bipolar Transistor Thermal Run-Away Analysis on page 13-7.

Different Bias Point Densities

Several **.DC** statements may follow one another to create different densities of bias points. For example, the following sequence

```
.DC Vin VSTART=0.0 VSTOP=2.0 VINCR=.2
.DC Vin VSTART=2.0 VSTOP=3.0 VINCR=.1
.DC Vin VSTART=3.0 VSTOP=5.0 VINCR=.2
```

places a higher density of bias points between 2.0 and 3.0 volts. Such an arrangement of bias points might be used to examine the switching behavior of a 5V CMOS inverter.

Fixed Voltages and Nodes

The following parameters and statements are used to fix voltages at specified nodes:

- **UIC** parameter—Forces the voltages at certain nodes to remain fixed regardless of the input voltage.
- **.IC** statement—Specifies which nodes are fixed and the voltage to use.

  The program effectively connects a small resistor between the node to be fixed and the desired voltage.

Procedures

The **.DC** statement is a composite statement which performs a series of steps to obtain a solution. In terms of Medici statements, the procedure is as follows:

```
Vin=VSTART
if ( no solution in memory )
  {
    SYMBOLIC GUMMEL CARRIERS=0
    SOLVE INIT V1=Va V2=Vb V3=Vc ......
  }

SYMBOLIC NEWTON CARRIERS=2
while (Vin <= VSTOP)
  {
    SOLVE  OUTFILE=Outf
    Vin=Vin + VINCR
    Outf=Outf + 1
  }
```

- The voltages *Va, Vb, Vc* etc. are obtained from the user-specified guess on the **.NODESET** statement. These are applied to the Medici element terminals during the zero carrier solution.
- During the two carrier solution, the circuit voltages determine the Medici terminal voltages. This procedure of using a zero carrier solution as an initial guess works in a wide range of problems, with a wide range of bias conditions.
- When the first bias point has been obtained, the program uses either **PREVIOUS** or **PROJECT** to obtain an initial guess for subsequent bias points. If a bias point fails, the program halves the bias step and continues automatically.

Troubleshooting

This section details possible solutions to the following problems:

- First bias point failure
- Failure to converge

The recombination models **AUGER** and **CONSRH** help convergence in general and should always be turned on if appropriate.

First Bias Point Failure

When continuation is used, most convergence problems which occur on later bias points are dealt with automatically. If the first bias point fails, however, the program aborts since the bias step cannot be reduced. If a "Pivot Approximately Equal to Zero" error occurs check the following:

1. Double check the circuit connections and Medici terminals.

   Often a simple coding error generates a circuit very different than what you expected.

2. Check for voltage source / inductor loops.

   In this condition a loop is formed containing voltage sources and or inductors and no other elements. Since in general, no solution exists for such a connection of elements, the circuit matrix is singular and a pivot equal to zero error results.

   EXAMPLE: (**V1, V2**, and **L1** form a loop)

   ```
   V1 1 0 5
   V2 2 0 3
   L1 1 2 1
   R1 1 0 3
   ```

3. Check for isolated nodes or nodes without a path to ground.

   This condition occurs if a node is connected only to capacitors (or capacitive elements like the gate of a MOSFET). Or if a portion of the circuit has no path

to ground (Node 0). Since the voltage at such a node (in steady state) may be assigned any value, the circuit matrix is singular.

EXAMPLE 1: (Node 2 is floating)

```
V1 1 0 5
R1 1 0 100
C1 1 2 1u
C2 2 0 2u
```

EXAMPLE 2: (Nodes 2 and 3 have no path to ground)

```
V1 1 0 5
R1 1 0 100
V2 2 3 5
R2 2 3 100
```

Failure to Converge: Too Many Iterations

If the simulation fails to converge (too many iterations) try the following:

1. Double check the circuit connections and Medici terminals.

   Often a simple coding error generates a circuit very different than what you expected.

2. Make sure that an initial guess is specified for all nodes with non-zero voltage and that the values specified are reasonable.

   The values don't have to be exact but they should be within 0.5V on a 5V circuit.

3. Adjust the **DELVMAX** parameter on the **.OPTIONS** statement (see Chapter 3, .OPTIONS on page 3-355).

   a. If you are sure that your initial guess is correct use a smaller value for **DELVMAX** to increase the stability of the solution.

      Limiting the change in voltage stabilizes the solution process by preventing wild fluctuations in voltage which can cause the solution to oscillate or generate physically unrealistic voltages.

   b. If you are unsure of the initial guess you might try a larger **DELVMAX**.

      If the initial guess is too far off, many iterations may be needed to reach the correct solution. For example, if an initial guess of 1V is specified, **DELVMAX** is 0.5 and the correct voltage at a node is 20V a minimum of (20 - 1)/0.5 = 38 Newton iterations is needed.

4. Solve the devices separately and load their solutions.

5. Ramp the different voltage or current sources up from zero by:

   a. Setting all the voltage or current sources to zero in the circuit

   b. Use **.DC** statements to ramp them to their final values one at a time.

      If the **RESET** parameter on the **.DC** statements is set false (**^RESET**) then each source remains at its final ramped value after the **.DC** statement

finishes. Similar results can be obtained by exiting circuit mode and using **SOLVE** statements.

6. If the lumped MOS models are being used, connect a large resistor (1e8 ohms) from the drain to ground.

   This is especially true of CMOS stages when VDD<VTH.

7. Increase the maximum number of iterations allowed using the **ITLIM** parameter on the **.OPTIONS** statement.

   This helps if the iterations are converging (i.e. if the error norms, particularly the Poisson error, is going down). If the error norms are not decreasing this will probably not help.

8. Turn off some of the physical models and solve. If convergence is obtained, turn the models back on and solve again.

   To turn the models off, you must exit circuit mode.

Failure to Converge: Impact Ionization

Impact ionization in particular causes convergence problems. These problems are observed at voltages well below the breakdown voltage for the device since at low voltages, what little current there is may be dominated by the impact ionization current.

It is normally impossible to jump directly to a voltage above breakdown. You must gradually ramp the applied voltage into the breakdown region.

.SAVE

Specifies files in which to save data.

.SAVE

[MESH=<c>] [SOLUTION=<c>] [IVFILE=<c>] [ASCII]

Parameter	Type	Definition	Default	Units
MESH	char	The identifier for the file where the mesh information is written.	none	
SOLUTION	char	The identifier for the file where the solution information is written.	none	
IVFILE	char	The identifier for the file where the terminal currents, voltages, and circuit variables are written. This file is always written as a formatted file.	*<base>.ivl*	
ASCII	logical	Indicates that the specified file(s) are formatted files.	false	

Example

```
.SAVE  MESH=NMOS.MSH  SOLUTION=NMOS.000  IVFILE=NMOS.IVF
```

.TRAN

Causes Medici to perform a transient (time dependent) simulation.

```
.TRAN
   DT=<n> TSTOP=<n> TMAX=<n> UIC
```

Parameter	Type	Definition	Default	Units
DT	number	The initial time step to use. Subsequent steps are selected automatically.	none	seconds
TSTOP	number	The length of the simulated interval.	none	seconds
TMAX	number	The maximum timestep to use.	**TSTOP**	seconds
UIC	logical	Causes Medici to force the voltages at certain nodes to specified values during the DC solution which is used as the initial guess for the transient solution. These nodes and their values are specified using the **.IC** statement.	false	

Example

```
.TRAN DT=1p TSTOP=1n
```

Failed Convergence

The **.TRAN** statement is in effect a composite statement. If no solution is present, **.TRAN** first performs a DC solution to obtain the initial conditions.

If convergence problems occur, they are most likely in the DC phase of the solution (refer to the comments for the **.DC** statement).

Failed Time Points

Time points may occasionally fail during the time dependent portion of the simulation. Medici handles these by halving the times step and resolving.

Switching Waveforms

Failed time points are particularly common near the edges of switching waveforms but do not present any real problems. To reduce wasting CPU time on failed time points it may help to reduce the number of Newton iterations the program tries before failing.

This is controlled by the **ITLIM** parameter on the **.OPTIONS** statement (or on the **METHOD** statement in Medici mode).

MOSFET Models

In circuits with compact MOSFET models, a series of time points may occasionally fail in succession causing the time step to be reduced to a very small value. At this point the program usually terminates with the message "Time step too small" or "Too Many Time Point Reductions". The cause of this problem is usually the discontinuity in the MOS capacitance and I-V curves where the MOSFET switches from triode to saturation.

To correct this type of convergence problem try another MOS capacitance model (the default BSIM capacitance model used is usually the best).

3.8 Statement Summary

This section summarizes the input statements recognized by Medici. The format used for the parameter list associated with a statement is identical to that used in the detailed statement descriptions.

The special characters < >, **[ ]**, |, **{ }**, and **( )** are used to indicate parameter types, optional parameters, and valid parameter combinations. See Chapter 3, Input Statements on page 3-1 for detailed descriptions of parameter syntax.

The summaries are organized alphabetically by statement name and include references to the chapter and page number of the manual where a detailed description of the statement can be found.

3D.SURFACE - Page 3-209

```
[HIDDEN] [VISIBLE] [LOWER] [UPPER] [X.LINE] [Y.LINE] [MASK]
[Z.MIN=<n>] [Z.MAX=<n>] [C.AUTO] [LINE.TYP=<n>] [COLOR=<n>] [PAUSE]
```

ABC.MESH - Page 3-38

```
Local Spacing Control
 { ( [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
     {  (ALIGN REGION1=<c> REGION2=<c> REGION3=<c> REGION4=<c>)
      | ( [ {  REGION=<c> | (BOUNDARY REGION1=<c> REGION2=<c>)
              | {  SILICON | GAAS | POLYSILI | GERMANIU | SIC | SIGE
                 | ALGAAS | A-SILICO | DIAMOND | HGCDTE | INAS | INGAAS
                 | INP | S.OXIDE | ZNSE | ZNTE | ALINAS | GAASP | INGAP
                 | INASP | OXIDE | NITRIDE | SAPPHIRE | OXYNITRI
                }
            }
         ]
         [CRITICAL=<n>] [H1=<n>] [H2=<n>]
         [{GRDRIGHT | GRDLEFT | GRDUP | GRDDOWN | GRDCNTR}]
         [NEIGHBOR=<c>] [{NORMAL=<n> | (NORMAL1=<n> NORMAL2=<n>)}]
         [NORMGROW=<n>] [LATERAL=<n>]
       )
     }
   )
Automatic MOSFET Meshing
 | ( MOSFET
      [N.CHANN=<n>] [N.JUNC=<n>] [N.GATEOX=<n>] [N.POLY=<n>]
      [MAX.CHAN]=<n> [MAX.SD=<n>] [SI.NFACT=<n>] [GATE.NFA=<n>]
      [RATIO=<n>]
   )
 }
```

ANISOTROPIC - Page 3-309

```
  [PRINT]

Semiconductor Parameters
  {  ( {  SILICON | GAAS | POLYSILI | SEMICOND | SIGE | ALGAAS
        | GERMANIU | SIC | S.OXIDE | HGCDTE | INGAAS | INP | INAS
        | DIAMOND | ZNSE | ZNTE | A-SILICO | REGION=<c>
       }

Anisotropic Component Factors
       [PERMITTI=<a>] [MU.N=<a>] [MU.P=<a>] [II.N=<a>] [II.P=<a>]
       [TH.COND=<a>]

General Anisotropic Electron Impact Ionization
       [ ANIIN [N.ION.O=<a>] [N.ION.1=<a>] [N.ION.2=<a>] [ECN.II=<a>]
          [EXN.II=<a>]
       ]

General Anisotropic Hole Impact Ionization
       [ ANIIP [P.ION.O=<a>] [P.ION.1=<a>] [P.ION.2=<a>] [ECP.II=<a>]
          [EXP.II=<a>]
       ]

General Anisotropic Thermal Conductivity
       [ ANTHCON [A.TH.CON=<a>] [B.TH.CON=<a>] [C.TH.CON=<a>] [D.TH.CON=<a>]
          [E.TH.CON=<a>]
       ]

Advanced Band Structure Parameters
       [ALPH0.N=<n>] [ALPHJ.N=<a>] [MJ.N=<a>] [EJ.N=<a>]
       [ALPH0.P=<n>] [ALPHJ.P=<a>] [MJ.P=<a>] [EJ.P=<a>]
     )

Insulator Parameters
   { ( {  OXIDE | NITRIDE | SAPPHIRE | OXYNITRI | INSULATO
        | REGION=<c>
       }
       [PERMITTI=<a>] [TH.COND=<a>]
     )
  }

Circuit Analysis AAM Parameters
  [STRUCTUR=<c>]
```

ASSIGN - Page 3-396

```
{  ( NAME=<c> [PRINT]
     {  ( N.VALUE=<a> [ {DELTA=<n> | RATIO=<n>} ] )
      | ( N.VALUE=<a> {OPTIMIZE | SENSITIV} LOWER=<n> UPPER=<n> )
      | ( L.VALUE=<a> )
      | ( C.VALUE=<c> [DELTA=<n>] )
      | ( [C1=<c>] [C2=<c>] [C3=<c>] [C4=<c>] [C5=<c>]
          [C6=<c>] [C7=<c>] [C8=<c>] [C9=<c>] [C10=<c>]
        )
     }
     [E.NAME=<c>] [PROMPT=<c>] [LEVEL=<n>]
   )
 | ( PRINT [INITIAL] [NAME=<c>] )
}
```

BATCH - Page 3-376

```
[<c>]
```

BOUNDARY - Page 3-55

```
IN.FILE=<c> [ASCII.IN] [ {2D.PROC | TSUPREM4} ]
[ OUT.FILE=<c> [ASCII.OU] ]
[X.SCALE=<n>] [Y.SCALE=<n>] [X.OFFSET=<n>] [Y.OFFSET=<n>]
[X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
[X.TOLER=<n>] [Y.TOLER=<n>] [X.AVERAG] [Y.AVERAG]
[ SHARP [ANGLE.CR=<n>] ] [ ASPECT [LOW.ASPE=<n>] [HIGH.ASP=<n>] ]
```

CALCULATE - Page 3-408

```
NAME=<c> ( A=<c> [SQRTA] ) [ {B=<c> | C=<n>} [SQRTB] ]
[ {SUM | DIFFEREN | RATIO | PRODUCT} ] [UNITS=<c>]
```

CALL - Page 3-369

```
{FILE=<c> | ( [FIRST=<c>] [LAST=<c>] [EXPAND] )} [ONCE] [PRINT]
```

C<*name*> - Page 3-329

<*node+*> <*node-*> <*value*>

CHECK - Page 3-411

```
IN.FILE=<c> [IN.MESH=<c>]
```

COMMENT - Page 3-365

```
   [<c>]

or

$

   [<c>]
```

CONTACT - Page 3-301

```
   {NAME=<c> | ALL} [PRINT]
   [ {  NEUTRAL | ALUMINUM | P.POLYSI | N.POLYSI | MOLYBDEN | TUNGSTEN
      | MO.DISIL | TU.DISIL | WORKFUNC=<n>
     }
     [PIN]
   ]

 The following set may NOT presently be used with the Circuit Analysis
 AAM if the terminal is attached to the circuit:
   [ {  CURRENT
      | CON.RESI=<n>
      | ( CHARGE [CAPACITA=<n>] [V.CAPAC=<n>] )
      | ( RESISTAN=<n>] [CAPACITA=<n>] [INDUCTAN=<n>] )
      | ( SURF.REC [VSURFN=<n>] [VSURFP=<n>] [ BARRIERL [ALPHA=<n>] ] )
      | VOLTAGE
     }
   ]

 Lattice Temperature AAM PArameters
   [ R.THERMA=<n>] [C.THERMA=<n>] ]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]

 Optical Device AAM Parameters
   [ {TRANSELE | REFLECTI=<n>} ]
```

CONTOUR - Page 3-211

```
Contour Quantities
  {  POTENTIA | QFN | QFP | VALENC.B | CONDUC.B | VACUUM | E.FIELD
   | DOPING | ELECTRON | HOLES | NIE | NET.CHAR | NET.CARR
   | J.CONDUC | J.ELECTR | J.HOLE | J.DISPLA | J.TOTAL | FLOWLINE
   | RECOMBIN | N.RECOMB | P.RECOMB | II.GENER | BB.GENER | PHOTOGEN
   | ELE.TEMP | HOL.TEMP | ELE.VEL | HOL.VEL | J.EFIELD
   | G.GAMN | G.GAMP | G.GAMT | G.IN | G.IP | G.IT
   | ARRAY1 | ARRAY2 | ARRAY3 | ( {TRAPS | TRAP.OCC} [LEVEL=<n>] )
   | N.MOBILI | P.MOBILI | SIGMA
   | IMPURITY=<c> | OTHER=<c>

Lattice Temperature AAM Parameters
   | LAT.TEMP

Heterojunction Device AAM Parameters
   | X.MOLE
  }

AC Small-Signal Analysis Quantity Parameters
  [ {AC.REAL | AC.IMAG | AC.MAGN | AC.PHAS} ]

Contour Controls
  [MIN.VALU=<n>] [MAX.VALU=<n>] [WINDOW] [DEL.VALU=<n>] [NCONTOUR=<n>]
  [ FILL [C.START=<n>] [C.INCREM=<n>] ]
  [ABSOLUTE] [LOGARITH] [X.COMPON] [Y.COMPON]
  [LINE.TYP=<n>] [COLOR=<n>] [PAUSE]
```

.DC - Page 3-414

[<*srcname*> **VSTART=<n> VSTOP=<n> VINCR=<n> [UIC] [MULT=<n>][RESET]]**

DEPOSITION - Page 3-412

```
X.START=<n> Y.START=<n> ANGLE=<n>
CONCENTR=<a> LENGTH=<a> WIDTH=<a> [ELECTRON] [HOLE]
```

D<*name*> - Page 3-329

<*node+*> <*node-*> <*mname*> **[AREA=<n>]**

ECHO - Page 3-404

[<c>]

ELECTRODE - Page 3-76

```
  NAME=<c> [VOID]
   {  ( [ {TOP | BOTTOM | LEFT | RIGHT | INTERFAC | PERIMETE} ]
        [ {X.MIN=<n> | IX.MIN=<n>} ] [ {X.MAX=<n> | IX.MAX=<n>} ]
        [ {Y.MIN=<n> | IY.MIN=<n>} ] [ {Y.MAX=<n> | IY.MAX=<n>} ]
        [ { (ROTATE X.CENTER=<n> Y.CENTER=<n> R.INNER=<n> R.OUTER=<n>)
           | (POLYGON X.POLY=<a> Y.POLY=<a>)
          }
        ]
      )
    | [ X=<n> Y=<n> ]
    | [REGION=<c>]
   }
   [MAJORITY]

 Lattice Temperature AAM Parameters
   [THERMAL]
```

ELIMINATE - Page 3-50

```
{ROWS | COLUMNS}
[ {X.MIN=<n> | IX.MIN=<n>} ] [ {X.MAX=<n> | IX.MAX=<n>} ]
[ {Y.MIN=<n> | IY.MIN=<n>} ] [ {Y.MAX=<n> | IY.MAX=<n>} ]
```

E.LINE - Page 3-226

```
{  ( [POTENTIA] [QFN] [QFP] [VALENC.B] [CONDUC.B] [VACUUM]
     [ARRAY1] [ARRAY2] [ARRAY3]
   )
 | [ E.FIELD [ANGLE=<n>] ]
 | ( [DOPING] [ELECTRON] [HOLES] [NET.CHAR] [NET.CARR] )
 | ( [J.CONDUC] [J.ELECTR] [J.HOLE] [J.DISPLA] [J.TOTAL]
     [ANGLE=<n>]
   )
 | [RECOMBIN] | [II.GENER] | [BB.GENER] | [PHOTOGEN]
 | [ELE.TEMP] | [HOL.TEMP] | [ELE.VEL] | [HOL.VEL] | [J.EFIELD]
 | [G.GAMN] | [G.GAMP] | [G.GAMT] | [G.IN] | [G.IP] | [G.IT]

Lattice Temperature AAM Parameters
   | [LAT.TEMP]

Heterojunction Device AAM Parameters
   | [X.MOLE]
  }

Control Parameters
  X.START=<n> Y.START=<n> [S.DELTA=<n>] [N.LINES=<n>] [HORZ.STA=<n>]
  [I.ELECTR] [I.HOLES] [E.MARK=<n>] [M.SIZE=<n>] [INSULATO]
  [LINE.TYP=<n>] [COLOR=<n>] [FILE=<c>] [SUMMARY] [PLOT] [PAUSE]
```

ELSE - Page 3-384

```
[COND]
```

E<*name*> - Page 3-330

<*node+*> <*node-*> <*cnode+*> <*cnode-*> <*value*>

EXTRACT - Page 3-166

```
Extraction Using Names and Expressions
  {  ( EXPRESSI=<c> NAME=<c> [UNITS=<c>] [CONDITIO=<c>] [INITIAL=<n>]
       [ {AT.BIAS | NOW} ] [OUT.FILE=<c> TWB] [CLEAR] [PRINT]

Optimization Using Targets and Expressions
       [ TARGET=<n> [WEIGHT=<n>] [MIN.REL=<n>] [MIN.ABS=<n>]
         [TARTOL=<n>] [TARREL=<n>]
       ]
     )

Extract Physical Quantities from Solution
   | ( {  NET.CHAR | NET.CARR | ELECTRON | HOLE | RECOMBIN | IONIZATI
        | RESISTAN | N.RESIST | P.RESIST | ( METAL.CH CONTACT=<c> )
        | ( {N.CURREN | P.CURREN} {CONTACT=<c> | REGIONS=<c>} )
        | II.GENER | (SHEET.RE X.POINT=<n>)
       }
       [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
       [OUT.FILE=<c>]

Device to Extract (Circuit Analysis AAM)
       [STRUCTUR=<c>]
     )

Extract MOS Device Parameters
   | ( MOS.PARA [DRAIN=<c>] [GATE=<c>] [IN.FILE=<c>] [I.DRAIN=<n>]
       [OUT.FILE=<c> [TWB] ] [CONDITIO=<c>] [APPLIED]
     )
  }
```

FILL - Page 3-223

```
  [REGION=<c>] [SET.COLO] [N-TYPE] [P-TYPE] [PAUSE]
  {  [COLOR=<n>]
   | ( [C.SILIC=<n>] [C.GAAS=<n>] [C.POLYSI=<n>] [C.GERMAN=<n>]
       [C.SIC=<n>] [C.SEMI=<n>] [C.SIGE=<n>] [C.ALGAAS=<n>]
       [C.A-SILI=<n>] [C.DIAMON=<n>] [C.HGCDTE=<n>] [C.INAS=<n>]
       [C.INGAAS=<n>] [C.INP=<n>] [C.S.OXID=<n>] [C.ZNSE=<n>]
       [C.ZNTE=<n>]
       [C.OXIDE=<n>] [C.NITRID=<n>] [C.SAPPHI=<n>] [C.OXYNIT=<n>]
       [C.INSUL=<n>] [C.ELECTR=<n>]
       [NP.COLOR] [C.NTYPE=<n>] [C.PTYPE=<n>]
     )
  }
```

FINISH - PAGE 3-358

```
  CIRCUIT
```

F<*name*> - Page 3-331

<*node+*> <*node-*> <*vname*> <*value*>

G<*name*> - Page 3-332

<*node+*> <*node-*> <*cnode+*> <*cnode-*> <*value*>

HELP - Page 3-368

[NAME=<c>] [{PARAMETE=<c> | VERBOSE}]

H<*name*> - Page 3-332

<*node+*> <*node-*> <*vname*> <*value*>

.IC - Page 3-354

V(<node1>)=<n> V(<node2>)=<n> .....

IF - Page 3-382

[COND]

IF.END - Page 3-385

[<c>]

IGNORE - Page 3-406

[<c>]

IMPURITY - Page 3-298

```
  NAME=<c> [PRINT]

 Material or Region Name
   [ {  SILICON | GAAS | POLYSILI | SEMICOND | SIGE | ALGAAS
      | GERMANIU | SIC | S.OXIDE | HGCDTE | INGAAS | INP | INAS
      | DIAMOND | ZNSE | ZNTE | A-SILICO | ALINAS | GAASP | INGAP
      | INASP | REGION=<c>
     }

 Incomplete Ionization Parameters
     [GB=<n>] [EB0=<n>] [ALPHA=<n>] [BETA=<n>] [GAMMA=<n>]
     [HDT.MIN=<n>] [HDT.MAX=<n>]
   ]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]
```

I<*name*> - Page 3-333

```
   <node+> <node->
 {  <value>
     | ( PULSE <i0> <ia> <td> <tr> <tf> <tp> <per> )
     | ( EXP   <i0> <ia> <td1> <tau1> <td2> <tau2> )
     | ( SIN   <i0> <ia> <freq> <tds> <theta> )
     | ( SFFM  <i0> <ia> <fc> <mdi> <fs> )
   }
```

INTERACTIVE - Page 3-374

```
   [ONCE]
```

INTERFACE - Page 3-306

```
   [ {MATERIAL=<c> | REGION=<c>} ]
   [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
   {  ( [S.N=<n>] [S.P=<n>] [QF=<n>]
        [N.ACCEPT=<n>] [P.ACCEPT=<n>] [N.DONOR=<n>] [P.DONOR=<n>]
      )
    |
      ( Q.INSULA=<n> [D.CHAR=<n>] )
   }
   [CLEAR] [ALL.CLEA]
```

I.PRINT - Page 3-377

```
{( [FIRST=<c>] [LAST=<c>] ) | [ALL]} [EXPAND]
```

I.SAVE - Page 3-379

```
FILE=<c> [NOW] [FIRST=<c>] [LAST=<c>] [EXPAND]
```

K*<name>* - Page 3-336

<lname1> <lname2> <value>

LABEL - Page 3-232

```
[LABEL=<c>] [SYMBOL=<n>] [X=<n>] [Y=<n>] [ANGLE=<n>]
[ {START.LE | START.CE | START.RI} ] [ARROW]
[LX.START=<n>] [LY.START=<n>]
[LX.FINIS=<n>] [LY.FINIS=<n>]
[CM] [C.SIZE=<n>] [LINE.TYP=<n>] [COLOR=<n>] [PAUSE]
```

L.END - Page 3-394

```
[BREAK] [ALL]
```

L.MODIFY - Page 3-392

```
[LEVEL=<n>] [STEPS=<n>] [ {NEXT | BREAK} ] [PRINT]
```

L*<name>* - Page 3-336

<node+> <node-> <value>

LOAD - Page 3-243

```
[ IN.FILE=<c> [ASCII.IN] ] [ IN.PREV=<c> [DIFFEREN] ]
[ OUT.FILE=<c> [ASCII.OU] ] [CHECK.ER] [TIF]

Circuit Analysis AAM Parameters
  [STRUCTUR=<c>]
```

.LOAD - Page 3-357

```
[MESH=<c>] [[SOLUTION=<c>] [STRUCTUR=<c>]] [ASCII]
```

LOG - Page 3-237

```
{  ( [OUT.FILE=<c> [TIF] ] [CLOSE] )
 | ( [ {  ( AURORA [LENGTH=<n>] [WIDTH=<n>] [DEVID=<n>] [REGION=<n>]
                  [P.CHANNE]
           )
         | ( ICCAP [OUT1=<c>] [OUT2=<c>] [OUT3=<c>]
                   [INP1=<c>] [INP2=<c>] [INP3=<c>] [INP4=<c>]
           )
         | STANFORD
       }
       IN.FILE=<c> OUT.FILE=<c>
       [GATE=<c>] [SOURCE=<c>] [DRAIN=<c>] [SUBSTRAT=<c>]
       [BASE=<c>] [EMITTER=<c>] [COLLECT=<c>] [EXTRA=<c>]
     ]
   )
}
```

LOOP - Page 3-386

```
[STEPS=<n>] [PRINT] [ {OPTIMIZE | SENSITIV} ]
```

MATERIAL - Page 3-252

```
  [PRINT]

Semiconductor Parameters
  [ {  ( {  SILICON | GAAS | POLYSILI | SEMICOND | SIGE | ALGAAS
          | GERMANIU | SIC | S.OXIDE | HGCDTE | INGAAS | INP | INAS
          | DIAMOND | ZNSE | ZNTE | A-SILICO | ALINAS | GAASP | INGAP
          | INASP | REGION=<c>
        }
        [PERMITTI=<n>] [EG.MODEL=<n>] [AFFINITY=<n>] [EG300=<n>]
        [EGALPH=<n>] [EGBETA=<n>] [EGGAMM=<n>]
        [NC300=<n>] [NC.F=<n>] [NV300=<n>] [NV.F=<n>]
        [GCB=<n>] [GVB=<n>] [EDB=<n>] [EAB=<n>]
        [TAUN0=<n>] [NSRHN=<n>] [AN=<n>] [BN=<n>] [CN=<n>] [EN=<n>]
        [TAUP0=<n>] [NSRHP=<n>] [AP=<n>] [BP=<n>] [CP=<n>] [EP=<n>]
        [EXN.TAU=<n>] [EXP.TAU=<n>]
        [ETRAP=<n>] [M.RTUN=<n>] [B.RTUN=<n>] [S.RTUN=<n>]
        [E.RTUN=<n>] [C.DIRECT=<n>]
        [AUGN=<n>] [AUGP=<n>] [ARICHN=<n>] [ARICHP=<n>]
        [N0.BGN=<n>] [V0.BGN=<n>] [CON.BGN=<n>]
        [A.EHS=<n>] [B.EHS=<n>] [C.EHS=<n>]
      [N.IONIZA=<n>] [N.ION.1=<n>] [N.ION.2=<n>] [ECN.II=<n>] [EXN.II=<n>]
      [P.IONIZA=<n>] [P.ION.1=<n>] [P.ION.2=<n>] [ECP.II=<n>] [EXP.II=<n>]
        [LAMHN=<n>] [LAMRN=<n>] [LAMHP=<n>] [LAMRP=<n>]
        [A.BTBT=<n>] [B.BTBT=<n>] [A.FN=<n>] [B.FN=<n>]
        {[KAPPA.QM=<n>] | ( [KAPPA.N=<n>] [KAPPA.P=<n>] )}
        [N.ACCUM=<n>] [P.ACCUM=<n>] [DREF.QM=<n>]
        [QM.NORP=<n>] [QM.EFIEL=<n>]
        [U.STRESS=<n>] [D.STRESS=<n>] [A.STRESS=<n>]
        [B.STRESS=<n>] [C.STRESS=<n>]
        [ME.DT=<n>]

Energy Balance Equation Parameters
        [ELE.CQ=<n>] [ELE.TAUW=<n>] [WTN0=<n>] [WTN1=<n>]
        [WTN2=<n>] [WTN3=<n>] [WTN4=<n>] [WTN5=<n>] [WTNL=<n>] [TNL=<n>]
        [HOL.CQ=<n>] [HOL.TAUW=<n>] [WTP0=<n>] [WTP1=<n>]
        [WTP2=<n>] [WTP3=<n>] [WTP4=<n>] [WTP5=<n>] [WTPL=<n>] [TPL=<n>]

Lattice Temperature AAM Parameters for Semiconductors
        [DENSITY=<n>] [DN.LAT=<n>] [DP.LAT=<n>]
        [A.SP.HEA=<n>] [B.SP.HEA=<n>] [C.SP.HEA=<n>] [D.SP.HEA=<n>]
        [F.SP.HEA=<n>] [G.SP.HEA=<n>]

        [A.TH.CON=<n>] [B.TH.CON=<n>] [C.TH.CON=<n>] [D.TH.CON=<n>]
        [E.TH.CON=<n>]
        [OP.PH.EN=<n>] [LAN300=<n>] [LAP300=<n>]

Heterojunction Device AAM Parameters
        [ {X.MOLE=<n> | X.OTHER=<c>} ] [EM.MODEL=<n>]
        [EG.X0=<n>] [EG.X1=<n>] [EG.X2=<n>] [EG.X3=<n>] [EG.X4=<n>]
        [EG.X5=<n>] [EG.X6=<n>] [EG.X7=<n>] [EG.X8=<n>] [EG.X9=<n>]
        [EG.X10=<n>] [EG.X11=<n>] [EG.X12=<n>] [EG.X13=<n>] [EG.X14=<n>]
```

(**MATERIAL**, continued next page)

(MATERIAL, continued from previous page**)**

```
          [AF.X0=<n>] [AF.X1=<n>] [AF.X2=<n>] [AF.X3=<n>] [AF.X4=<n>]
          [AF.X5=<n>] [AF.XL=<n>] [EGALX=<n>] [EGBEX=<n>] [EGGAX=<n>]
          [EGALL=<n>] [EGBEL=<n>] [EGGAL=<n>]
          [MEG=<n>] [MEG.X1=<n>] [MEX=<n>] [MEX.X1=<n>] [MEL=<n>]
          [MEL.X1=<n>] [MH0=<n>] [MH0.X1=<n>] [ML0=<n>] [ML0.X1=<n>]
        )

Insulator Parameters
     | ( {OXIDE | NITRIDE | SAPPHIRE | OXYNITRI | INSULATO | REGION=<c>}
          [PERMITTI=<n>] [AFFINITY=<n>] [EG300=<n>]
          [ECN.GC=<n>] [BARLN=<n>] [TUNLN=<n>]
          [ECP.GC=<n>] [BARLP=<n>] [TUNLP=<n>]
          [ME.DT=<n>]

Lattice Temperature AAM Parameters for Insulators
          [DENSITY=<n>]
          [A.SP.HEA=<n>] [B.SP.HEA=<n>] [C.SP.HEA=<n>] [D.SP.HEA=<n>]
          [F.SP.HEA=<n>] [G.SP.HEA=<n>]
          [A.TH.CON=<n>] [B.TH.CON=<n>] [C.TH.CON=<n>] [D.TH.CON=<n>]
          [E.TH.CON=<n>]
        )

Electrode Parameters
     | ( ELECTROD=<c> [A.FN=<n>] [B.FN=<n>] [ME.DT=<n>] [BARR.DT=<n>] )
    }
  ]

Circuit Analysis AAM Parameters
  [STRUCTUR=<c>]

Optical Device AAM Parameters
  [PR.TABLE]

 Optical Energy Gap
  [EGO300=<n>] [EGOALPH=<n>] [EGOBETA=<n>]

  Real Refractive Index
  [ {( WAVE.RE=<a> INDEX.RE=<a> ) | RRI.FILE=<c>} [FIRST] [LAST] ]

  Imaginary Refractive Index or Absorption Coefficient
  [ {  ( {  ( WAVE.IM=<a> {INDEX.IM=<a> | ABSORPTI=<a>} )
           | ( {IRI.FILE=<c> | ABS.FILE=<c>} )
          }
          [FIRST] [LAST]
        )
     | ( BTBT.AB B.BB=<a> E.PHONON=<a> E1.BB=<a> EXP.BB=<a> )
    }
  ]
  [ BATA.AB [E1.BT=<n>] {E.URBACH=<n> | G.BT=<n>} ]
  [ FRCA.AB [EL.EMAS=<n>] [HO.EMAS=<n>] ]
```

MESH - Page 3-15

```
Initial Mesh Generation
  {  ( [ {RECTANGU | CYLINDRI} ] [DIAG.FLI] )

Mesh File Input
   | ( IN.FILE=<c> [PROFILE]
       [ {  ASCII.IN
          | ( TSUPREM4 [ ELEC.BOT [Y.TOLER=<n>] ] [POLY.ELE]
              [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
              [FLIP.Y] [SCALE.Y=<n>]
            )
          | ( TIF [ ELEC.BOT [Y.TOLER=<n>] [POLY.ELE] ] )
         }
       ]
New Automatic Boundary Conforming Mesh Generation
      [{ ( ABC
            [GRIDTOP] [VOIDELEC] [RFN.CRNR] [JUNC.ABC]
            [CRITICAL=<n>] [SPACING=<n>]
            [N.SEMICO=<n>] [N.INSULA=<n>] [N.CONDUC=<n>]
            [NORMGROW=<n>] [LATERAL=<n>] [ELIMINAT]
          ) |
Old Automatic Boundary Conforming Mesh Generation
         ( ABC.OLD
            [SPACING=<n>][RATIO=<n>] [ANGLE=<n>]
            [MAX.SPAC=<n>] [CORNER=<n>] [LAYERS=<n>]
            [ATTEMPTS=<n>] [RELAX=<n>] [ELIMINAT]
            [EXTERNAL=<n>] [OPTIMIZE]   [JUNC.ABC]
            [N.SEMICO=<n>] [N.INSULA=<n>] [N.CONDUC=<n>]
            [N.SILICO=<n>] [N.POLYSI=<n>] [N.OXIDE=<n>]
            [N.NITRID=<n>] [N.OXYNIT=<n>] [N.SAPPHI=<n>]
            [N.BPSG=<n>]   [N.INAS=<n>]   [N.GAAS=<n>]
            [N.ALGAAS=<n>] [N.HGCDTE=<n>] [N.S.OXID=<n>]
            [N.SIC=<n>]    [N.INGAAS=<n>] [N.INP=<n>]
            [N.GERMAN=<n>] [N.DIAMON=<n>] [N.ZNSE=<n>]
            [N.ZNTE=<n>]   [N.A-SILI=<n>] [N.SIGE=<n>]
         )  |
Quadtree Mesh Generator
         ( QUADTREE
            [MIN.THIC=<n>] [MINDEL=<n>] [MAXDEL=<n>]
         )
       }
      ]
     )
  } Mesh Adjustments
  [PERIODIC [PBC.TOL=<n>] ] [ORDER] [ADJUST] [VIRTUAL]
  [CENTROID] [ WIDTH=<n> N.SPACES=<n> [X.SPLIT=<n>] ]
  [OBTUSE.A=<n>]

Mesh File Output
  [ OUT.FILE=<c> [ASCII.OU] [NO.TTINF] ] [SMOOTH.K=<n>]
```

METHOD - Page 3-139

```
  [ITLIMIT=<n>] [XNORM] [RHSNORM] [XRNORM [NODE.ERR=<n>] ]
  [PX.TOLER=<n>] [CX.TOLER=<n>] [PR.TOLER=<n>] [CR.TOLER=<n>]
  [NO.CARR=<c>] [LIMIT] [PRINT] [FIX.QF] [ITER.TTY] [ASMB.OLD]

ILUCGS Solver Parameters
  [ILU.ITER=<n>] [ILU.TOL=<n>] [ILU.XTOL=<n>]

Gummel's Method Parameters
  {  ( [ {  DVLIMIT=<n>
          | ( DAMPED [DELTA=<n>] [DAMPLOOP=<n>] [DFACTOR=<n>] )
         }
       ]
       [ ICCG [LU1CRIT=<n>] [LU2CRIT=<n>] [MAXINNER=<n>] ]
       [ SINGLEP
         [ ACCELERA [ACCSTART=<n>] [ACCSTOP=<n>] [ACCSTEP=<n>] ]
       ]
     )

Newton's Method Parameters
    | ( [ AUTONR [NRCRITER=<n>] [ERR.RAT=<n>] ]
       [ CONT.RHS [ITRHS=<n>] ] [CONT.PIV] [ {CONT.ITL | STOP.ITL} ]
       [CONT.STK] [STACK=<n>] [ACONTINU=<n>]
       [TAUTO] [2NDORDER] [TOL.TIME=<n>] [L2NORM] [DT.MIN=<n>]
       [EXTRAPOL]
       [CARR.MIN=<n>] [CARR.FAC=<n>] [N.DVLIM=<n>] [N.DVMAX] [N.DAMP]
     )
  }

Energy Balance Parameters
  [ETX.TOLE=<n>] [ETR.TOLE=<n>] [N.MAXBL=<n>] [N.MAXEB=<n>]

Lattice Temperature AAM Parameters
  [LTX.TOLE=<n>] [LTR.TOLE=<n>] [LTX.FACT=<n>] [LTR.FACT=<n>]
  [MAX.TEMP=<n>]
```

M<*name*> - Page 3-337

```
  <noded> <nodeg> <nodes> <nodeb> <mname>
  [W=<n>] [L=<n>] [AS=<n>] [AD=<n>] [PS=<n>] [PD=<n>]
```

MOBILITY - Page 3-275

```
  [SILICON] [GAAS] [POLYSILI] [SEMICOND] [SIGE] [ALGAAS]
  [GERMANIU] [SIC] [S.OXIDE] [HGCDTE] [INGAAS] [INP] [INAS]
  [DIAMOND] [ZNSE] [ZNTE] [A-SILICO] [REGION=<c>] [PRINT]
  [ALINAS] [GAASP] [INGAP] [INASP]

Constant Mobility Parameters
  [MUN0=<n>] [MUP0=<n>]

Mobility Table Parameters
  [ CONCENTR=<a> [ELECTRON=<a>] [HOLE=<a>] [FIRST] [LAST] ] [PR.TABLE]

Analytic Mobility Model Parameters
  [MUN.MIN=<n>] [MUN.MAX=<n>] [NREFN=<n>]
  [NUN=<n>] [XIN=<n>] [ALPHAN=<n>]
  [MUP.MIN=<n>] [MUP.MAX=<n>] [NREFP=<n>]
  [NUP=<n>] [XIP=<n>] [ALPHAP=<n>]

III-V Compound Semiconductor Analytic Mobility Model Parameters
  [MIN.X1=<n>] [MIN.X2=<n>] [MAN.X1=<n>] [MAN.X2=<n>] [NREFN2=<n>]
  [MIP.X1=<n>] [MIP.X2=<n>] [MAP.X1=<n>] [MAP.X2=<n>] [NREFP2=<n>]

Arora Mobility Model Parameters
  [MUN1.ARO=<n>] [MUN2.ARO=<n>] [CN.ARORA=<n>] [AN.ARORA=<n>]
  [EXN1.ARO=<n>] [EXN2.ARO=<n>] [EXN3.ARO=<n>] [EXN4.ARO=<n>]
  [MUP1.ARO=<n>] [MUP2.ARO=<n>] [CP.ARORA=<n>] [AP.ARORA=<n>]
  [EXP1.ARO=<n>] [EXP2.ARO=<n>] [EXP3.ARO=<n>] [EXP4.ARO=<n>]

Carrier-Carrier Scattering Mobility Model Parameters
  [A.CCS=<n>] [B.CCS=<n>] [A.LIC=<n>] [B.LIC=<n>]
  [C.LIC=<n>] [EX.LIC=<n>]
  [MUN0.LAT=<n>] [EXN.LAT=<n>] [AN.IIS=<n>] [BN.IIS=<n>]
  [MUP0.LAT=<n>] [EXP.LAT=<n>] [AP.IIS=<n>] [BP.IIS=<n>]

Philips Unified Mobility Model Parameters
  [MMNN.UM=<n>] [MMXN.UM=<n>] [NRFN.UM=<n>] [ALPN.UM=<n>]
  [TETN.UM=<n>] [NRFD.UM=<n>] [CRFD.UM=<n>]
  [MMNP.UM=<n>] [MMXP.UM=<n>] [NRFP.UM=<n>] [ALPP.UM=<n>]
  [TETP.UM=<n>] [NRFA.UM=<n>] [CRFA.UM=<n>]

Effective Field Parameters
  [ETAN=<n>]  [ZETAN=<n>] [ETAP=<n>] [ZETAP=<n>]

Surface Degradation Factors
  [GSURFN=<n>] [GSURFP=<n>]
Lombardi Surface Mobility Model Parameters
  [MUN0.LSM=<n>] [MUN1.LSM=<n>] [MUN2.LSM=<n>]
  [CRN.LSM=<n>] [CSN.LSM=<n>]
  [BN.LSM=<n>] [CN.LSM=<n>] [DN.LSM=<n>]
  [EXN1.LSM=<n>] [EXN2.LSM=<n>] [EXN3.LSM=<n>] [EXN4.LSM=<n>]
```

(**MOBILITY**, continued next page)

(**MOBILITY**, continued from previous page)

```
   [EXN8.LSM=<n>
   [MUP0.LSM=<n>] [MUP1.LSM=<n>] [MUP2.LSM=<n>]
   [CRP.LSM=<n>] [CSP.LSM=<n>]
   [BP.LSM=<n>] [CP.LSM=<n>] [DP.LSM=<n>]
   [EXP1.LSM=<n>] [EXP2.LSM=<n>] [EXP3.LSM=<n>] [EXP4.LSM=<n>] [EXP8.LSM=<n>]
   [PC.LSM=<n>]

 Generalized Mobility Curve Model
   [BN.GMC=<n>] [CN.GMC=<n>] [DN.GMC=<n>]
   [D1N.GMC=<n>] [D2N.GMC=<n>]
   [EXN4.GMC=<n>] EXN5.GMC=<n>] EXN6.GMC=<n>] [EXN7.GMC=<n>]
   [EXN8.GMC=<n>]
   [BP.GMC=<n>] [CP.GMC=<n>] [DP.GMC=<n>]
   [D1P.GMC=<n>] [D2P.GMC=<n>]
   [EXP4.GMC=<n>] EXP5.GMC=<n>] EXP6.GMC=<n>] [EXP7.GMC=<n>]
   [EXP8.GMC=<n>]

 Shirahata Mobility Model Parameters
   [E1N.SHI=<n>] [EX1N.SHI=<n>] [E2N.SHI=<n>] [EX2N.SHI=<n>]
   [E1P.SHI=<n>] [EX1P.SHI=<n>] [E2P.SHI=<n>] [RX2P.SHI=<n>]

 Surface Mobility Model Parameters
   [EREFN=<n>] [EXN.SM=<n>] [MUREFN=<n>]
   [EREFP=<n>] [EXP.SM=<n>] [MUREFP=<n>]

 Enhanced Surface Mobility Model Parameters
   [MUN1.SM=<n>] [MUN2.SM=<n>] [MUN3.SM=<n>]
   [EXN1.SM=<n>] [EXN2.SM=<n>] [EXN3.SM=<n>]
   [MUP1.SM=<n>] [MUP2.SM=<n>] [MUP3.SM=<n>]
   [EXP1.SM=<n>] [EXP2.SM=<n>] [EXP3.SM=<n>]

 Universal Mobility Model Parameters
   [MUN.UNI=<n>] [ECN.UNI=<n>] [EXN.UNI=<n>]
   [MUP.UNI=<n>] [ECP.UNI=<n>] [EXP.UNI=<n>]

 Perpendicular Field Mobility Parameters
   [ECN.MU=<n>] [ECP.MU=<n>]

 Hewlett-Packard Mobility Model Parameters
   [MUN0.HP=<n>] [ECN.HP=<n>] [VSN.HP=<n>] [VCN.HP=<n>] [GN.HP=<n>]
   [NRFN.HP=<n>]
   [MUP0.HP=<n>] [ECP.HP=<n>] [VSP.HP=<n>] [VCP.HP=<n>] [GP.HP=<n>]
   [NRFP.HP=<n>]

 Field Dependent Mobility Model Parameters
   [VSATN=<n>] [BETAN=<n>] [E0N=<n>]
   [VSATP=<n>] [BETAP=<n>] [E0P=<n>]
   [FLDMOB=<n>]
```

(**MOBILITY**, continued next page)

(**MOBILITY**, continued from previous page)

```
III-V Compound Semiconductor Field Dependent Mobility Model Parameters
  [VSN.X1=<n>] [VSN.X2=<n>] [EN.X1=<n>] [EN.X2=<n>]

Tranverse Field Dependent Mobility Model Parameters
  [TEMPN.UT=<n>] [PHONN.UT=<n>] [SURFN.UT=<n>] [COULN.UT=<n>]
  [TEMPP.UT=<n>] [PHONP.UT=<n>] [SURFP.UT=<n>] [COULP.UT=<n>]
  [ACC.N.UT=<n>] [ACC.P.UT=<n>] [INV.N.UT=<n>] [INV.P.UT=<n>]

Stress-Induced Mobility Model Parameters
  [MLT.STR=<n>] [MUL0.STR=<n>]

Device Selection (Circuit Analysis AAM)
  [STRUCTUR=<c>]
```

.MODEL - Page 3-346

```
<mod.nam>
{  Diode Parameters
   ( D
     [IS=<n>] [CJO=<n>] [M=<n>] [VJ=<n>] [N=<n>] [FC=<n>]
     [BV=<n>] [EG=<n>] [XTI=<n>] [TT=<n>]
   )

   Bipolar Junction Transistor Parameters
 | ( {NPN | PNP}
     [IS=<n>] [BF=<n>] [BR=<n>] [TF=<n>] [TR=<n>] [CJC=<n>]
     [CJE=<n>] [VJC=<n>] [VJE=<n>] [MJC=<n>] [MJE=<n>]
     [IKF=<n>] [IKR=<n>] [NE=<n>] [NC=<n>] [NF=<n>] [NR=<n>]
     [VAF=<n>] [VAR=<n>] [ISC=<n>] [ISE=<n>] [FC=<n>]  [FE=<n>]
     [VTF=<n>] [ITF=<n>] [XTF=<n>] [RB=<n>] [RBM=<n>]
     [XTI=<n>] [XTB=<n>] [EG=<n>]
   )

   MOS Transistor Parameters (Medici Built-in Models)
 | ( {NMOS | PMOS}

    Common and Level 1 parameters
     [LEVEL=<n>] [LD=<n>] [TOX=<n>] [NSUB=<n>] [NSS=<n>] [UO=<n>]
     [PHI=<n>] [GAMMA=<n>] [TPG=<n>] [VTO=<n>] [KP=<n>] [JS=<n>]
     [XJ=<n>] [LAMBDA=<n>]

    Level 2 adds the following to the common set
     [UCRIT=<n>] [UEXP=<n>] [VMAX=<n>] [NFS=<n>] [NEFF=<n>]
     [DELTA=<n>]
```

(**.MODEL**, continued on next page)

(**.MODEL**, continued from previous page)

```
       Level 3 adds the following to the common set
        [KAPPA=<n>] [DELTA=<n>] [THETA=<n>] [VMAX=<n>] [ETA=<n>]
        [NFS=<n>]

       Capacitance Parameters
        [CGSO=<n>] [CGDO=<n>] [CJ=<n>] [CJSW=<n>][MJ=<n>] [MJSW=<n>]
        [MCAP=<n>] [FC=<n>] [PB=<n>] [XQC=<n>] [K1=<n>]
      )

      MOS Transistor Parameters (Star-Hspice Levels 1, 2, and 3)
    | ( { ( HSPICE TYPE=<n> ) | NMOS | PMOS}
        LEVEL=<n> [COX=<n>] [KP=<n>] [LAMBDA=<n>] [TOX=<n>] [UO=<n>]
        [DEL=<n>] [LD=<n>] [LDAC=<n>] [LMLT=<n>] [WD=<n>] [WDAC=<n>]
        [WMLT=<n>] [XJ=<n>] [XL=<n>] [XW=<n>] [GAMMA=<n>] [NFS=<n>]
        [NSUB=<n>] [PHI=<n>] [VTO=<n>] [ECRIT=<n>] [NEFF=<n>] [VMAX=<n>]
        [LREF=<n>] [WREF=<n>] [DELTA=<n>] [LND=<n>] [LN0=<n>] [ND=<n>]

        [N0=<n>] [WIC=<n>] [WND=<n>] [WN0=<n>] [MOB=<n>] [THETA=<n>]
        [UCRIT=<n>] [UEXP=<n>] [UTRA=<n>] [DERIV=<n>] [KAPPA=<n>]
        [ETA=<n>] [JS=<n>] [CAPOP=<n>] [CJ=<n>] [CJSW=<n>] [MJ=<n>]
        [MJSW=<n>] [FC=<n>] [PB=<n>] [CGBO=<n>] [CGSO=<n>] [CGDO=<n>]
        [XQC=<n>] [K1=<n>]
      )
     MOS Transistor Parameters (Star-Hspice Level 28)
    | ( { ( HSPICE28 TYPE=<n> ) | NMOS | PMOS}
        LEVEL=<n> [LD=<n>] [LDAC=<n>] [LMLT=<n>] [LREF=<n>] [XLREF=<n>]
        [WD=<n>] [WDAC=<n>] [WMLT=<n>] [XL=<n>] [XW=<n>] [WREF=<n>]
        [XWREF=<n>] [BEX=<n>] [FEX=<n>] [TCV=<n>] [B1=<n>] [LB1=<n>]
        [WB1=<n>] [B2=<n>] [LB2=<n>] [WB2=<n>] [CGBO=<n>] [CGDO=<n>]
        [CGSO=<n>] [ETA0=<n>] [LETA=<n>] [WETA=<n>] [ETAMN=<n>]
        [LETAMN=<n>] [WETAMN=<n>] [GAMMN=<n>] [LGAMN=<n>] [WGAMN=<n>]
        [K1=<n>] [LK1=<n>] [WK1=<n>] [K2=<n>] [LK2=<n>] [WK2=<n>]
        [MUZ=<n>] [LMUZ=<n>] [WMUZ=<n>] [N0=<n>] [LN0=<n>] [WN0=<n>]
        [NB0=<n>] [LNB=<n>] [WNB=<n>] [ND0=<n>] [LND=<n>] [WND=<n>]
        [PHI0=<n>] [LPHI=<n>] [WPHI=<n>] [TOXM=<n>] [U00=<n>] [LU0=<n>]
        [WU0=<n>] [U1=<n>] [LU1=<n>] [WU1=<n>] [VDDM=<n>] [VFB0=<n>]
        [LVFB=<n>] [WVFB=<n>] [WFAC=<n>] [LWFAC=<n>] [WWFAC=<n>]
        [WFACU=<n>] [LWFACU=<n>] [WWFACU=<n>] [X2E=<n>] [LX2E=<n>]
        [WX2E=<n>] [X2M=<n>] [LX2M=<n>] [WX2M=<n>] [X2U0=<n>] [LX2U0=<n>]
        [WX2U0=<n>] [X2U1=<n>] [LX2U1=<n>] [WX2U1=<n>] [X33M=<n>]
        [LX33M=<n>] [WX33M=<n>] [X3E=<n>] [LX3E=<n>] [WX3E=<n>] [X3MS=<n>]
        [LX3MS=<n>] [WX3MS=<n>] [X3U1=<n>] [LX3U1=<n>] [WX3U1=<n>]
        [XPART=<n>]
      )

      MOS Transistor Parameters (Star-Hspice Levels 49 and 53)
    | ( { ( BSIM3V3 TYPE=<n> ) | NMOS | PMOS}
        LEVEL=<n> [VERSION=<n>] [HSPVER=<n>] [PARAMCHK=<n>] [APWARN=<n>]
        [BINFLAG=<n>] [MOBMOD=<n>] [CAPMOD<n>] [NOIMOD=<n>] [NLEV=<n>]
        [NQSMOD=<n>] [SFVTFLAG=<n>] [VFBFLAG=<n>] [VGSLIM=<n>] [TOX=<n>]
        [XJ=<n>] [NGATE=<n>] [VTH0=<n>] [NSUB=<n>] [NCH=<n>] [NLX=<n>]
```

(**.MODEL**, continued on next page)

(**.MODEL**, continued from previous page)

```
          [K1=<n>] [K2=<n>] [K3=<n>] [K3B=<n>] [W0=<n>] [DVT0W=<n>]
         [DVT1W=<n>] [DVT2W=<n>] [DVT0=<n>] [DVT1=<n>] [DVT2=<n>]
         [ETA0=<n>] [ETAB=<n>] [DSUB=<n>] [VBM=<n>] [U0=<n>] [UA=<n>]
         [UB=<n>] [UC=<n>] [A0=<n>] [AGS=<n>] [B0=<n>] [B1=<n>] [KETA=<n>]
         [VOFF=<n>] [VSAT=<n>] [A1=<n>] [A2=<n>] [RDSW=<n>] [PRWG=<n>]
         [PRWB=<n>] [WR=<n>] [NFACTOR=<n>] [CIT=<n>] [CDSC=<n>] [CDSCD=<n>]
         [CDSCB=<n>] [PCLM=<n>] [PDIBLC1=<n>] [PDIBLC2=<n>] [PDIBLCB=<n>]
         [DROUT=<n>] [PSCBE1=<n>] [PSCBE2=<n>] [PVAG=<n>] [DELTA=<n>]
         [ALPHA0=<n>] [BETA0=<n>] [RSH=<n>] [XPART=<n>] [CGSO=<n>]
         [CGDO=<n>] [CGBO=<n>] [CGS1=<n>] [CGD1=<n>] [CKAPPA=<n>] [CF=<n>]
         [CLC=<n>] [CLE=<n>] [VFBCV=<n>] [WINT=<n>] [WLN=<n>] [WW=<n>]
         [WWN=<n>] [WWL=<n>] [DWG=<n>] [DWB=<n>] [LINT=<n>] [LL=<n>]
         [LLN=<n>] [LW=<n>] [LWN=<n>] [LWL=<n>] [DLC=<n>] [DWC=<n>]
         [KT1=<n>] [KT1L=<n>] [KT2=<n>] [UTE=<n>] [UA1=<n>] [UB1=<n>]
         [UC1=<n>] [AT=<n>] [PRT=<n>] [XTI=<n>] [LMIN=<n>] [LMAX=<n>]

         [WMIN=<n>] [WMAX=<n>] [BINUNIT=<n>] [GAMMA1=<n>] [GAMMA2=<n>]
         [VBX=<n>] [XT=<n>] [NOIA=<n>] [NOIB=<n>] [NOIC=<n>] [EM=<n>]
         [AF=<n>] [KF=<n>] [EF=<n>] [ACM=<n>] [JS=<n>] [JSW=<n>] [NJ=<n>]
         [N=<n>] [CJ=<n>] [CJSW=<n>] [CJSWG=<n>] [CJGATE=<n>] [PB=<n>]
         [PBSW=<n>] [PHP=<n>] [PBSWG=<n>] [MJ=<n>] [MJSW=<n>] [MJSWG=<n>]
         [ELM=<n>] [TOXM=<n>] [VFB=<n>] [NOFF=<n>] [VOFFCV=<n>] [IJTH=<n>]
         [ALPHA1=<n>] [ACDE=<n>] [MOIN=<n>] [TPB=<n>] [TPBSW=<n>]
         [TPBSWG=<n>] [TCJ=<n>] [TCJSW=<n>] [TCJSWG=<n>] [LLC=<n>]
         [LWC=<n>] [LWLC=<n>] [WLC=<n>] [WWC=<n>] [WWLC=<n>]
      )
   }
```

(**.MODEL**, continued on next page)

MODELS - Page 3-106

```
[ {SRH | CONSRH} [R.TUNNEL] ] [AUGER] [BGN] [FN.CUR]
[ {BOLTZMAN | FERMIDIR} ] [IMPACT.I] [II.TEMP]
[ INCOMPLE [ENERGY.L] [HIGH.DOP] [IMPURITY=<c>] ]
[ BTBT [BT.MODEL=<n>] [BT.LOCAL=<n>]
  [ BT.QUAD [BT.ATOL=<n>] [BT.RTOL=<n>] [BT.TINY=<n>] ]
]
[ {  CONMOB | ANALYTIC | ARORA | CCSMOB | PHUMOB | LSMMOB
   | GMCMOB | SHIRAMOB
  }
]
[ {  SRFMOB | SRFMOB2 | UNIMOB | PRPMOB | LSMMOB
   | GMCMOB | SHIRAMOB | TFLDMOB | HPMOB
  }
]
[ {HPMOB | FLDMOB | TMPMOB } [ {ND.MOB | C.ND.MOB=<n>} ] ]
[E.EFFECT] [EJ.MOBIL] [EHSCAT] [STRMOB]
[ QM.PHILI [QM.AC] [QM.OLD] [QM.METHO=<n>] [QM.NORP=<n>]
  [QM.EFIEL=<n>] [QM.EMIN=<n>] [QM.EXTEN]
]
[ {GATE1 | ( GATE2 [GATE.SUR] )} [GATE.GEN=<n>] [GATE.TEM] ]
[TMPDIFF] [ET.MODEL] [EF.TMP] [EFI.TMP] [COMP.ET] [EBLT.HT]
[TMPTAUWN]  [TMPTAUWP] [EB.SRH.G] [TEMPERAT=<n>] [3KT.LT] [ECII.LAT]
[ {HJSC2 | ( HJTEM [HJTUN] )} ] [STRESS] [Y.ORIENT=<n>] [PRINT]
```

.NODESET - Page 3-353

V(<*node1*>**)=<n> V(<***node2***>)=<n> .....**

OPTION - Page 3-367

```
[G.DEBUG] [N.DEBUG] [ CPU.STAT [CPU.FILE=<c>] ] [I.ERROR]
[MAXNODES] [ SAVE.SOL [SOL.FILE=<c>] ]
```

.OPTIONS - Page 3-355

```
[T.TOL=<n>] [T.MIN=<n>] [P.TOL=<n>] [C.TOL=<n>] [ITLIM=<n>]
[DELVMAX=<n>] [G.FORCE=<n>] [T.NOM=<n>] [V.MIN=<n>] [V.MAX=<n>]
[2ND] [AUTO] [ALT] [ILUCGS] [LAT.TEMP] [COUP.LAT] [ELE.TEMP]
[HOL.TEMP] [TIF] [HSPICE]
```

PHOTOGEN - Page 3-114

```
   [X.START=<n>] [Y.START=<n>] X.END=<n> Y.END=<n>
   [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
   [ELECTRON] [HOLES]

 Spatial Terms
   [R.CHAR=<n>]
   [A1=<n>] [A2=<n>] [A3=<n>] [A4=<n>]
   [C1=<n>] [C2=<n>] [C3=<n>] [C4=<n>]
   [RECO=<n>] [ IN.FILE=<c> [RD.CHAR] ] [PC.UNITS] [CLEAR]
   [G.INTEG] [N.INTEG=<n>]

 Temporal Terms
   {  UNIFORM
    | ( GAUSSIAN TC=<n> [T0=<n>] )
    | ( DELTA [T0=<n>] )
    | ( PULSE TRS=<n> TPD=<n> TFS=<n> TPRD=<n> [T0=<n>] )
   }

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]

Optical Device AAM Parameters

 Incident Ray Quantities
   [ RAYTRACE X.ORG=<n> Y.ORG=<n> [ANGLE=<n>] [SPLIT.RA]
     {  ( WAVELENG=<n> {FLUX=<n> | INTENSIT=<n>} )
      | ( {SP.FILE=<c> | ( BB.RADIA [BB.TEMP=<n>] ) }
          WAVE.STA=<n> WAVE.END=<n> WAVE.NUM=<n>
        )
     }
     [RAY.WIDT=<n>] [RAY.NUM=<n>]
     [WAVE.SCA=<n>] [INT.SCAL=<n>]
     {  ( [POLARIZA=<n>] [PHASE.DI=<n>] )
      | ( [A.ELLIPS=<n>] [R.ELLIPS=<n>] )
     }
   ]

 Ray-Tracing Quantities
   [ {INT.RATI=<n> | INT.LIMI=<n>} ]
   [BOT.RFLT=<n>] [TOP.RFLT=<n>] [SID.RFLT=<n>] [SID.INCI] [TRANSPAR]
   [AMB.REFR=<n>] [WIDTH.CH] [QUAN.EFF=<n>] [PRINT.AB]

 Film Quantities
   [FILM.REG=<c>]
```

PLOT.1D - Page 3-182

```
Distance Plot Quantities
  {  ( {  POTENTIA | QFN | QFP | VALENC.B | CONDUC.B | VACUUM
        | E.FIELD | ARRAY1 | ARRAY2 | ARRAY3
        | ( {TRAPS | TRAP.OCC} [LEVEL=<n>] )
        | DOPING | ELECTRON | HOLES | NIE | NET.CHAR | NET.CARR
        | J.CONDUC | J.ELECTR | J.HOLE | J.DISPLA | J.TOTAL
        | RECOMBIN | N.RECOMB | P.RECOMB | II.GENER | BB.GENER
        | ( PHOTOGEN [WAVE.NUM=<n>] ) | N.MOBILI | P.MOBILI | SIGMA
        | ELE.TEMP | HOL.TEMP | ELE.VEL | HOL.VEL | J.EFIELD
        | G.GAMN | G.GAMP | G.GAMT | G.IN | G.IP | G.IT
        | IMPURITY=<c> | OTHER=<c>

Lattice Temperature AAM Parameters
        | LAT.TEMP

Heterojunction Device AAM Parameters
        | X.MOLE
       }

AC Small-Signal Analysis Quantity Parameters
       [ {AC.REAL | AC.IMAG | AC.MAGN | AC.PHAS} ]

Distance Plot Parameters
       [X.COMPON] [Y.COMPON]
       X.START=<n> Y.START=<n> X.END=<n> Y.END=<n> [HORZ.STA=<n>]
       [ {FIND.MIN | FIND.MAX} [SEMICOND] [INSULATO] [FIND.DIS=<n>] ]
     )

Terminal Characteristics Plot Parameters
   | ( X.AXIS=<c> Y.AXIS=<c> [ORDER] [IN.FILE=<c>]
       [X.MIN=<n>] [X.MAX=<n>] [CONDITIO=<c>]
     )
  }

Plot Controls
  [ SPLINE [NSPLINE=<n>] ]
  [LEFT=<n>] [RIGHT=<n>] [BOTTOM=<n>] [TOP=<n>] [UNCHANGE]
  [ {Y.LOGARI | S.LOGARI | INTEGRAL} ] [ABSOLUTE] [NEGATIVE]
  [CLEAR] [AXES] [LABELS] [MARKS] [TITLE=<c>] [T.SIZE=<n>]
  [X.OFFSET=<n>] [X.LENGTH=<n>] [X.SIZE=<n>] [X.LOGARI]
  [Y.OFFSET=<n>] [Y.LENGTH=<n>] [Y.SIZE=<n>]
  [CURVE] [ {SYMBOL=<n> | POINTS} ] [C.SIZE=<n>]
  [LINE.TYP=<n>] [COLOR=<n>] [DEVICE=<c>] [PAUSE]
  [PLOT.OUT=<c>] [PLOT.BIN=<c>] [PRINT] [OUT.FILE=<c>]
  [TIMESTAM [TIME.SIZ=<n>] ]

Circuit Analysis AAM Parameters
  [STRUCTUR=<c>]
```

PLOT.2D - Page 3-195

```
  [BOUNDARY [REGION] ] [JUNCTION] [DEPLETIO] [LUMPED] [CON.RESI]
  [ GRID [ELEM.NUM] [NODE.NUM] [REG.NUM] [N.SIZE=<n>] [OBTUSE] ]
  [CROSSES] [FILL] [SCALE]
  [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
  [CLEAR] [LABELS] [MARKS] [TOP.MARK] [TITLE=<c>] [T.SIZE=<n>]
  [L.BOUND=<n>] [L.JUNCT=<n>] [L.DEPLE=<n>] [L.GRID=<n>] [L.ELECT=<n>]
  [C.BOUND=<n>] [C.JUNCT=<n>] [C.DEPLE=<n>] [C.GRID=<n>] [C.ELECT=<n>]
  [X.OFFSET=<n>] [X.LENGTH=<n>] [X.SIZE=<n>]
  [Y.OFFSET=<n>] [Y.LENGTH=<n>] [Y.SIZE=<n>]
  [DEVICE=<c>] [PLOT.OUT=<c>] [PLOT.BIN=<c>] [PAUSE]
  [TIMESTAM [TIME.SIZ=<n>] ]

Circuit Analysis AAM Parameters
  [STRUCTUR=<c>]

Optical Device AAM Parameter
  [RAYPLOT [WAVE.NUM=<n>] ]
```

PLOT.3D - Page 3-201

```
Plot Quantities
  {  POTENTIA | QFN | QFP | VALENC.B | CONDUC.B | VACUUM | E.FIELD
   | DOPING | ELECTRON | HOLES | NIE | NET.CHAR | NET.CARR
   | J.CONDUC | J.ELECTR | J.HOLE | J.DISPLA | J.TOTAL
   | RECOMBIN | N.RECOMB | P.RECOMB | II.GENER | BB.GENER | PHOTOGEN
   | ELE.TEMP | HOL.TEMP | ELE.VEL | HOL.VEL | J.EFIELD
   | G.GAMN | G.GAMP | G.GAMT | G.IN | G.IP | G.IT
   | ARRAY1 | ARRAY2 | ARRAY3 | ( {TRAPS | TRAP.OCC} [LEVEL=<n>] )
   | N.MOBILI | P.MOBILI | SIGMA | LAT.TEMP | X.MOLE
   | IMPURITY=<c> | OTHER=<c>
 }

AC Small-Signal Analysis Quantity Parameters
 [ {AC.REAL | AC.IMAG | AC.MAGN | AC.PHAS} ]

Plot Controls
  [X.COMPON] [Y.COMPON] [Z.MIN=<n>] [Z.MAX=<n>] [ABSOLUTE] [LOGARITH]

Device Bounds
  [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]

Viewing Parameters
  [THETA=<n>] [PHI=<n>] [X.LINES=<n>] [Y.LINES=<n>] [EQUIDIST]

Axis and Labels
  [AXES] [LABELS] [MARKS] [TITLE=<c>] [T.SIZE=<n>]
  [X.LENGTH=<n>] [Y.LENGTH=<n>] [Z.LENGTH=<n>]
  [X.LABEL=<c>]  [Y.LABEL=<c>]  [Z.LABEL=<c>]
  [X.SIZE=<n>]   [Y.SIZE=<n>]   [Z.SIZE=<n>]

Display Parameters
  [CLEAR] [FRAME] [CENTER] [FILL.VIE]
  [XV.LENGT=<n>] [XV.OFFSE=<n>] [YV.LENGT=<n>] [YV.OFFSE=<n>]
  [X.OFFSET=<n>] [Y.OFFSET=<n>]
  [DEVICE=<c>] [L.BOX=<n>] [C.BOX=<n>] [PAUSE]
  [ TIMESTAM [TIME.SIZ=<n>] ]

File Output
  [PLOT.OUT=<c>] [PLOT.BIN=<c>]

Circuit Analysis AAM Parameters
  [STRUCTUR=<c>]
```

P<*name*> - Page 3-338

```
<node1>=<term1> <node2>=<term2> .........
FILE=<c> WIDTH=<n>
```

PRINT - Page 3-179

```
   [ {  ( [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>] )
      | ( [IX.MIN=<n>] [IX.MAX=<n>] [IY.MIN=<n>] [IY.MAX=<n>] )
     }
   ]
   [POINTS] [ELEMENTS] [GEOMETRY] [INTERFAC] [IMPURITY] [OTHER]
   [SOLUTION] [ CURRENT [ {X.COMPON | Y.COMPON} ] ] [E.FIELD]
   [NET.CHAR] [RECOMBIN] [II.GENER] [II.EJG] [CONC.DEP]
   [BB.GENER] [BB.EG] [TEMPERAT] [BAND.STR]

 Circuit Analysis AAM Parameters
   [STRUCTUR=<c>]
```

PROFILE - Page 3-82

```
   [REGION=<c>]
   [X.MIN=<n>] [ {WIDTH=<n> | X.MAX=<n>} ]
   [Y.MIN=<n>] [ {DEPTH=<n> | Y.MAX=<n>} ]

 Output Doping File
   [OUT.FILE=<c>]

 Uniform Profile
   {  ( UNIFORM {N-TYPE | P-TYPE | IMPURITY=<c> | OTHER=<c>} N.PEAK=<n>
)

 Analytic Profiles
    | ( {N-TYPE | P-TYPE | IMPURITY=<c> | OTHER=<c>} {N.PEAK=<n> |
DOSE=<n>}
       {Y.CHAR=<n> | Y.JUNCTI=<n>} {X.CHAR=<n> | XY.RATIO=<n>} [X.ERFC]
      )

 Analytic Polygonal Profiles
    | ( POLYGON {N-TYPE | P-TYPE | IMPURITY=<c> | OTHER=<c>} N.PEAK=<n>
        X.POLY=<a> Y.POLY=<a> N.CHAR=<n> [N.ERFC]
      )

 Analytic Rotated Profiles
    | ( ROTATE {N-TYPE | P-TYPE | IMPURITY=<c> | OTHER=<c>} N.PEAK=<n>
        X.CENTER=<n> Y.CENTER=<n> R.INNER=<n> R.OUTER=<n> R.CHAR=<n>
[R.ERFC]
      )
```

(**PROFILE** statement continued on next page)

(**PROFILE** statement continued from previous page)

```
One-Dimensional Profiles from Data Files
   | ( IN.FILE=<c> [N.OFFSET=<n>] [Y.OFFSET=<n>]
       {  ( 1D.PROC [N-TYPE] [P-TYPE] )
        | ( SUPREM2 [N-TYPE] [P-TYPE] )
        | ( 1D.ASCII [Y.COLUMN=<n>]
            {  ( [N.COLUMN=<n>] [P.COLUMN=<n>] )
             | ( D.COLUMN=<n> {IMPURITY=<c> | OTHER=<c>} )
            }
          )
       }
       {X.CHAR=<n> | XY.RATIO=<n>} [X.ERFC]
     )

Two-Dimensional Profiles from Data Files
   | ( IN.FILE=<c> [N.OFFSET=<n>] [X.OFFSET=<n>] [Y.OFFSET=<n>]
       {  ( 2D.PROC [N-TYPE] [P-TYPE] )
        | ( SUPRA [N-TYPE] [P-TYPE] )
        | ( TSUPREM4 [N-TYPE] [P-TYPE] )
        | ( 2D.ASCII [X.COLUMN=<n>] [Y.COLUMN=<n>]
            {  ( [N.COLUMN=<n>] [P.COLUMN=<n>] )
             | ( D.COLUMN=<n> {IMPURITY=<c> | OTHER=<c>} )
            }
          )
        | ( TIF [ {IMPURITY=<c> | ( [N-TYPE] [P-TYPE] )} ]
                [OTHER=<c> [INSULATO] ]
          )
        | ( MEDICI [ {IMPURITY=<c> | ( [N-TYPE] [P-TYPE] )} ] [OTHER=<c>] )
       }
       [X.CHAR=<n>] [X.ERFC] [Y.CHAR=<n>] [Y.ERFC]
     )
  }
```

Q*<name>* - Page 3-339

<nodec> <nodeb> <nodee> <mname> **[AREA=<n>]**

REGION - Page 3-70

```
NAME=<c>
 Semiconductor Materials
{  ( {  SILICON | GAAS | POLYSILI | GERMANIU | SIC | SEMICOND
      | SIGE | ALGAAS | A-SILICO | DIAMOND | HGCDTE | INAS | INGAAS |
       | INP | S.OXIDE | ZNSE | ZNTE | ALINAS | GAASP | INGAP | INASP
     }

 Semiconductor Material Parameters
     [X.MOLE=<n>]
     [ {X.END=<n> | X.SLOPE=<n>} {X.LINEAR | Y.LINEAR} ]
   )

 Insulator Materials
 | OXIDE | NITRIDE | SAPPHIRE | OXYNITRI | INSULATO
}

 Location
{  ( [ {X.MIN=<n> | IX.MIN=<n>} ] [ {X.MAX=<n> | IX.MAX=<n>} ]
     [ {Y.MIN=<n> | IY.MIN=<n>} ] [ {Y.MAX=<n> | IY.MAX=<n>} ]
     [ { (ROTATE R.INNER=<n> R.OUTER=<n> X.CENTER=<n> Y.CENTER=<n> )
       | (POLYGON X.POLY=<a> Y.POLY=<a> )
       }
     ]
   )
 | [ X=<n> Y=<n> ]
}

[VOID]
```

REGRID - Page 3-94

```
Regrid Criteria
  {  POTENTIA | ( E.FIELD [ {X.COMPON | Y.COMPON} ] ) | QFN | QFP
   | DOPING | ELECTRON | HOLES | NET.CHAR | NET.CARR
   | ( MIN.CARR [LOCALDOP] ) | II.GENER | BB.GENER | PHOTOGEN
   | ELE.TEMP | HOL.TEMP | TRUNC | ARRAY1 | ARRAY2 | ARRAY3
   | IMPURITY=<c> | OTHER=<c>

Lattice Temperature AAM Parameters
   | LAT.TEMP
  }

Regrid Controls
  (RATIO=<n> | FACTOR=<n>) [IN.FILE=<c>]
  [CHANGE] [ABSOLUTE] [LOGARITH] [MAX.LEVE=<n>] [SMOOTH.K=<n>]
  [X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
  [REGION=<c>] [IGNORE=<c>] [COS.ANGL=<n>]
  [ OUT.FILE=<c> [NO.TTINF] [ASCII] ]

Quadtree Regrid Controls
  [BOUNDARY]
  [MAXDEL=<n>] [MAXDEL.X=<n>] [MAXDEL.Y=<n>]
  [MINDEL=<n>] [MINDEL.X=<n>] [MINDEL.Y=<n>]
  [ASINH] [UNREFINE=<n>]
```

RENAME - Page 3-81

```
(ELECTROD | REGION | T.ELECTR) OLDNAME=<c> NEWNAME=<c>
```

RETURN - Page 3-405

```
  [<c>]
```

R<*name*> - Page 3-340

```
  <node+> <node-> <value> [T1=<n>]  [T2=<n>]
```

SAVE - Page 3-246

```
  OUT.FILE=<c>

  {  ( SOLUTION [STRUCTUR=<c>] [ASCII] )
   | ( MESH [W.MODELS] [ASCII] )

Technology Interchange Format
   | ( TIF [ALL] [BANDS] [CURRENTS] [GENERATI] [COMPONEN]
       [AC.POTEN] [AC.CN] [AC.CP] [AC.TN] [AC.TP] [AC.TL]
       [AC.JDISP] [AC.JN] [AC.JP] [AC.JCOND] [AC.JTOT]
       [AC.COMP] [AC.SCOMP] )

Structure Parameters
       [MESH] [BOUND]

Scalar Quantities
       [POTENTIA] [QFN] [QFP] [VALENC.B] [CONDUC.B] [VACUUM]
       [DOPING] [ELECTRON] [HOLES] [NET.CHAR] [NET.CARR]
       [RECOMBIN] [II.GENER] [BB.GENER] [PHOTOGEN]
       [ELE.TEMP] [HOL.TEMP] [ELE.VEL] [HOL.VEL] [J.EFIELD]
       [G.GAMN] [G.GAMP] [G.GAMT] [G.IN] [G.IP] [G.IT]

Vector Quantities
       [J.CONDUC] [J.ELECTR] [J.HOLE] [J.DISPLA] [J.TOTAL] [E.FIELD]

Lattice Temperature AAM Parameters
       [LAT.TEMP]

Heterojunction Device AAM Parameters
       [X.MOLE]
    )
  }
```

.SAVE - Page 3-419

```
  [MESH=<c>] [SOLUTION=<c>] [IVFILE=<c>] [ASCII]
```

SOLVE - Page 3-150

```
 Initial Guesses, Biasing, and Fermi Potentials
  [ {INITIAL | PREVIOUS | PROJECT | LOCAL | P.LOCAL} ]
  [ {V(name1)=<n> | I(name1)=<n> | T(name1)=<n> | Q(name1)=<n>} ]....
  [ {V(name200)=<n> | I(name200)=<n> | T(name200)=<n> | Q(name200)=<n>} ]
  [N.REGION=<c>] [N.BIAS=<a>]
  [P.REGION=<c>] [P.BIAS=<a>]

Steady State Analysis Parameters
  [ {  ( ELECTROD=<c> {VSTEP=<n> | ISTEP=<n>} NSTEPS=<n> )

Continuation Method Parameters
     | ( CONTINUE ELECTROD=<c> C.VSTEP=<n> [ C.AUTO [C.TOLER=<n>] ]
         [C.VMIN=<n>] [C.VMAX=<n>] [C.IMIN=<n>] [C.IMAX=<n>]
         [C.DVMAX=<n>] [C.RMAX]
       )
Transient Analysis Parameters
     | ( TSTEP=<n> {TSTOP=<n> | NSTEPS=<n>} [TMULT=<n>]
         [ {RAMPTIME=<n> | ENDRAMP=<n>} ] [DT.MAX=<n>]
       )
    }
  ]
Hot Carrier and Parasitic Analysis Parameters
  [IMPACT.I] [GATE.CUR] [DQDV]

Programmable Device AAM Parameters
  [FN.CUR]

Direct Tunneling Analysis Parameters
  [DT.CUR] [DT.METH=<n>]

AC Small-Signal Analysis Parameters
  [ AC.ANALY FREQUENC=<n> [ FSTEP=<n> NFSTEP=<n> [MULT.FRE] ]
    [VSS=<n>] [TERMINAL=<c>]
    [S.OMEGA=<n>] [MAX.INNE=<n>] [TOLERANC=<n>] [HI.FREQ]
    [ S.PARAM [R.SPARA=<n>] ]
  ]
AC Charge-Partition Analysis Parameters
  [ AC.CHARG [TERMINAL=<c>] ]
Circuit Analysis AAM Parameters
  [ ELEMENT=<c> V.ELEMEN=<n> [VSTEP=<n> NSTEPS=<n>] ] [UIC]

AC Analysis with a Circuit
  [ AC.ANALY FREQUENC=<n> AC.SOURC=<c>
    [ FSTEP=<n> NFSTEP=<n> [MULT.FRE] ]
  ]
```

(**SOLVE**, continued next page)

(**SOLVE**, continued from previous page)

```
Output Choices
  [ OUT.FILE=<c> [SAVE.BIA]
    [ {  ( TIF [ALL] [BANDS] [CURRENTS] [GENERATI] [COMPONEN] )
       | ( [CURRENTS] [ASCII] [STRUCTUR=<c>] )
      }
    ]
  ]
Optical Device AAM Parameters
  [ {  ( {  WAVE=<n>
          | ( [WAVE.STA=<n>] [WAVE.END=<n>] )
          | SPECTR
         }
       )
     | ( [FLUX=<n>]
         { [LAMBDA=<n>]
          | ( LAMBDA.S=<n> LAMBDA.E=<n> LAMBDA.N=<n> )
         }
       )
     | ( INTENSIT=<n> [INT.STEP=<n>] )
    }
  ]
  [L.MODULA  LSS=<n>]
```

SPREAD - Page 3-52

```
{LEFT | RIGHT} WIDTH=<n> UPPER=<n> LOWER=<n> [ENCROACH=<n>]
{Y.LOWER=<n> | FIX.LOWE | ( THICKNES=<n> [VOL.RAT=<n>] )}
[GRADING=<n>] [ MIDDLE=<n> Y.MIDDLE=<n> [GR1=<n>] [GR2=<n>] ]
```

START - Page 3-328

```
 CIRCUIT [INITIAL]
```

STITCH - Page 3-101

```
IN.FILE=<c> [ASCII.IN] [ {TIF | TSUPREM4} [POLY.ELE] ]
{ TOP | BOTTOM | LEFT | RIGHT } [X.OFFSET=<n>] [Y.OFFSET=<n>]
[FLIP.X] [FLIP.Y] [ELEC.MER] [REG.MERG]
```

STOP - Page 3-405

```
[<c>]
```

SYMBOLIC - Page 3-137

```
{NEWTON | GUMMEL}
CARRIERS=<N> [ {ELECTRON | HOLES} ]
[ ELE.TEMP [COUP.ELE] ] [ HOL.TEMP [COUP.HOL] ] [EB.POST]
[ LAT.TEMP [COUP.LAT] ]
[MIN.DEGR] [ {[ILUCGS] | [BICGS]} ] [STRIP] [VIRTUAL]
[BLOCK.MA] [PRINT]
```

TITLE - Page 3-365

```
[<c>]
```

.TRAN - Page 3-420

```
DT=<n> TSTOP=<n> TMAX=<n> UIC
```

TRAPS - Page 3-130

```
Energy Level Creation
 {  ( DISTRIB [N.LEVEL=<n>] [ OUT.FILE=<c> X.PLOT=<n> Y.PLOT=<n> ] )
  | ( [E1=<n>] ... [E50=<n>] )
 }
 [MIDGAP] [CHARGE1] ... [CHARGE50] [ALL.CHAR] [DGEN1=<n>] ...
[DGEN50=<n>]

Trap Parameters
 [ [TAUN=<c>] [TAUP=<c>] [N.TOTAL=<c>] [Q.FIX=<c>] [CONDITIO=<c>]
   [FREEZE]
 ]

Transient Parameter
 [TIME.DEP]
```

TSUPREM4 - Page 3-67

```
IN.FILE=<c>
[X.LEFT=<n>] [X.RIGHT=<n>] [Y.TOP=<n>] [Y.BOT=<n>]
[X.MIN=<n>] [X.MAX=<n>] [Y.MIN=<n>] [Y.MAX=<n>]
[X.OFFSET=<n>] [Y.OFFSET=<n>] [X.INTERF=<n>]
[IMPURITY] [FLIP] [SYMMETRI]
```

VECTOR - Page 3-218

```
  {  J.CONDUC | J.ELECTR | J.HOLE | J.DISPLA | J.TOTAL | E.FIELD
   | AC.POTEN | AC.CN | AC.CP | AC.TN | AC.TP | AC.TL

Optical Device AAM Parameters
   | ( RAYTRACE [INCIDENT] [INTERNAL] [EXITING] )
  }

AC Small-Signal Analysis Vector Quantity Parameters
  [ {  AC.VECT | AC.XCOMP | AC.YCOMP | AC.REAL | AC.IMAG
     | AC.MAGN | AC.PHAS
    }
  ]

Plot Control Parameters
  [LOGARITH [NORM.LOG=<n>] ] [V.SIZE=<n>] [CLIPFACT=<n>]
  [MINIMUM=<n>] [MAXIMUM=<n>] [LINE.TYP=<n>] [COLOR=<n>] [PAUSE]
```

V<*name*> - Page 3-341

```
  <node+> <node->
{  <value>
 | ( PULSE <v0> <va> <td> <tr> <tf> <tp> <per> )
 | ( EXP   <v0> <va> <td1> <tau1> <td2> <tau2> )
 | ( SIN   <v0> <va> <freq> <tds> <theta> )
 | ( SFFM <v0> <va> <fc> <mdi> <fs> )
}
```

W<*name*> -Page 3-344

<*node+*> <*node-*> <*nodec1+*> <*nodec1-*> <*nodec2+*> <*nodec2-*> <*value*>

X.MESH - Page 3-32

```
{ LOCATION=<n> | ( {WIDTH=<n> | X.MAX=<n>} [X.MIN=<n>] ) }
[ {NODE=<n> | N.SPACES=<n>} ]
[ {SPACING=<n> | H2=<n>} ] [H1=<n>] [H3=<n>]
[RATIO=<n>] [MIN.SPAC=<n>] [SUMMARY]
```

Y.MESH - Page 3-35

```
{ LOCATION=<n> | ( {DEPTH=<n> | Y.MAX=<n>} [Y.MIN=<n>] ) }
[ {NODE=<n> | N.SPACES=<n>} ]
[ {SPACING=<n> | H2=<n>} ] [H1=<n>] [H3=<n>]
[RATIO=<n>] [MIN.SPAC=<n>] [SUMMARY]
```

N-Channel MOSFET Examples

Example Specifications

This chapter illustrates some of the analysis that might be performed on an N-channel MOS device. The specifications for most of the examples are as follows:

- The channel length is 1.5 microns.
- The input file *mdex1* develops the simulation structure.
- The input files *mdex1g* and *mdex1d* simulate the gate and drain characteristics, respectively.
- The input file *mdex1i* calculates gate current and substrate current due to impact ionization.

  This file also shows an efficient means for applying moderate to high biases to a structure.
- The effect of fast interface states on the gate characteristics of the device is studied using the input file *mdex1f*.
- An analysis of current leakage caused by band-to-band tunneling is considered using the input file *mdex1t*.
- The input file *mdex1a* performs an avalanche breakdown analysis using ionization integrals.
- The input file *mdex1b* performs an avalanche breakdown simulation when a gate bias just above threshold is applied to the device.

  For this simulation, the impact ionization generated current is included self-consistently in the solution.
- The input file *mdex1dt* performs an analysis of the gate leakage current produced by direct tunneling in a MOSCAP with a 15Å gate oxide.
- The input file *mdex1qt* illustrates how to use the quadtree mesh option to regrid a MOSFET.

Generation of the Simulation Structure

The input file *mdex1* creates the simulation structure for the n-channel MOS device. The output associated with the execution of Medici for the input file *mdex1* is shown in Figures 4-1 through 4-8.

```
 1... TITLE     Avant! MEDICI Example 1 - 1.5 Micron N-Channel MOSFET

 2... COMMENT   Specify a rectangular mesh
 3... MESH      SMOOTH=1
 4... X.MESH    WIDTH=3.0  H1=0.125

 5... Y.MESH    N=1   L=-0.025
 6... Y.MESH    N=3   L=0.
 7... Y.MESH    DEPTH=1.0  H1=0.125
 8... Y.MESH    DEPTH=1.0  H1=0.250

 9... COMMENT   Eliminate some unnecessary substrate nodes
10... ELIMIN    COLUMNS  Y.MIN=1.1

11... COMMENT   Increase source/drain oxide thickness using SPREAD
12... SPREAD    LEFT   WIDTH=.625  UP=1  LO=3  THICK=.1  ENC=2
13... SPREAD    RIGHT  WIDTH=.625  UP=1  LO=3  THICK=.1  ENC=2

14... COMMENT   Use SPREAD again to prevent substrate grid distortion
15... SPREAD    LEFT   WIDTH=100   UP=3  LO=4  Y.LO=0.125

16... COMMENT   Specify oxide and silicon regions
17... REGION    SILICON
18... REGION    OXIDE    IY.MAX=3

19... COMMENT   Electrode definition
20... ELECTR    NAME=Gate    X.MIN=0.625  X.MAX=2.375  TOP
21... ELECTR    NAME=Substrate  BOTTOM
22... ELECTR    NAME=Source  X.MAX=0.5  IY.MAX=3
23... ELECTR    NAME=Drain   X.MIN=2.5  IY.MAX=3

24... COMMENT   Specify impurity profiles and fixed charge
25... PROFILE   P-TYPE  N.PEAK=3E15  UNIFORM     OUT.FILE=MDEX1DS
26... PROFILE   P-TYPE  N.PEAK=2E16  Y.CHAR=.25
27... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=0.0  WIDTH=.5
  ... +         XY.RAT=.75
28... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=2.5  WIDTH=.5
  ... +         XY.RAT=.75
29... INTERFAC  QF=1E10

30... PLOT.2D   GRID  TITLE="Example 1 - Initial Grid"  FILL  SCALE

31... COMMENT   Regrid on doping
32... REGRID    DOPING  LOG  IGNORE=OXIDE  RATIO=2  SMOOTH=1
  ... +         IN.FILE=MDEX1DS
33... PLOT.2D   GRID  TITLE="Example 1 - Doping Regrid"  FILL  SCALE

34... COMMENT   Specify contact parameters
35... CONTACT   NAME=Gate  N.POLY

36... COMMENT   Specify physical models to use
37... MODELS    CONMOB  FLDMOB  SRFMOB2

38... COMMENT   Symbolic factorization, solve, regrid on potential
39... SYMB      CARRIERS=0
40... METHOD    ICCG  DAMPED
41... SOLVE
```

Figure 4-1 First portion of the listing of file *mdex1*

```
42... REGRID    POTEN  IGNORE=OXIDE  RATIO=.2  MAX=1  SMOOTH=1
  ... +         IN.FILE=MDEX1DS
  ... +         OUT.FILE=MDEX1MS
43... PLOT.2D   GRID  TITLE="Example 1 - Potential Regrid"  FILL  SCALE

44... COMMENT   Solve using the refined grid, save solution for later use
45... SYMB      CARRIERS=0
46... SOLVE     OUT.FILE=MDEX1S

47... COMMENT   Impurity profile plots
48... PLOT.1D   DOPING  X.START=.25  X.END=.25  Y.START=0  Y.END=2
  ... +         Y.LOG  POINTS  BOT=1E15  TOP=1E21  COLOR=2
  ... +         TITLE="Example 1 - Source Impurity Profile"
49... PLOT.1D   DOPING  X.START=1.5  X.END=1.5  Y.START=0  Y.END=2
  ... +         Y.LOG  POINTS  BOT=1E15  TOP=1E17  COLOR=2
  ... +         TITLE="Example 1 - Gate Impurity Profile"
50... PLOT.2D   BOUND  TITLE="Example 1 - Impurity Contours"  FILL  SCALE
51... CONTOUR   DOPING  LOG  MIN=16   MAX=20   DEL=.5  COLOR=2
52... CONTOUR   DOPING  LOG  MIN=-16  MAX=-15  DEL=.5  COLOR=1  LINE=2
```

Figure 4-2 Second portion of the listing of file *mdex1*

Mesh

The device structure is created by use of the mesh. Various regions of the device, i.e., semiconductor, insulator, and electrodes, are defined in terms of the mesh. Distortions of the mesh are then used to give the device its designed surface topography.

Defining the Initial Mesh

The first step in creating a device structure is to define an initial mesh. This is shown in lines 3 through 8 of the input file in Figure 4-1. At this point the initial mesh does not need to be fine enough for a simulation. It only needs to be fine enough to define the regions of the device. The mesh is refined at a later stage (see "Grid Refinement" on page 4-6).

Initiating and Smoothing

The mesh generation is initiated by specifying a **MESH** statement. The **MESH** statement is also used to request smoothing. Smoothing minimizes problems caused by obtuse triangles that may be generated as the result of subsequent **SPREAD** statements.

Dimensions and Properties

The **X.MESH** and **Y.MESH** statements specify how the mesh is generated.

Horizontal

The horizontal spacing of mesh lines is specified with the **X.MESH** statements. The **X.MESH** statement at line 4 creates a grid section extending from x=0 microns (the default starting location) to x=3 microns.

Specifying the single parameter **H1**=0.125 creates a uniform mesh in the horizontal direction with a grid spacing of 0.125 microns.

Vertical

The vertical spacing of mesh lines is specified with the **Y.MESH** statements. The first three horizontal mesh lines are intended to define a surface oxide with a thickness of 0.025 microns (the gate oxide thickness for this device).

Use **Y.MESH** statements to explicitly place the first line of nodes at *y*=-0.025 microns and the third line of nodes at *y*=0 microns. (It is convenient to set up a grid that places the insulator-semiconductor interface at *y*=0, although this is not required by the program.)

Grid Sections

The next **Y.MESH** statement adds a grid section to the structure with a depth of 1 micron and a uniform spacing between mesh lines of 0.125 microns. The final **Y.MESH** statement adds another 1 micron grid section that has a uniform spacing of 0.250 microns.

Triangular Grid

A rectangular grid is inefficient because a requirement of fine grid in one region of the device propagates fine grid throughout the device. A triangular grid does not suffer from this limitation, and has the advantage that a fine mesh is only needed near the surface and not deep in the bulk.

The **ELIMIN** statement in line 10 terminates many of the vertical grid lines within the device by removing every other column of nodes in the structure for values of y greater than 1.1 microns. This is possible because Medici uses a triangular grid.

Distorting the Oxide Grid Lines

A nonuniform oxide thickness is achieved by distorting the grid lines that define the oxide. This is done by using the following parameters:

- The first two **SPREAD** statements change the thickness of the first three grid lines from their original 0.025 micron thickness to 0.1 micron over the source and drain regions of the device.
- The **ENCROACH** parameter determines the characteristic length of the transition from the thicker to unchanged grid regions. **WIDTH** refers to the half-way point of the transition, measured from the **LEFT** or **RIGHT** edge of the device. The spread is accomplished by moving the upper lines up and the lower lines down.
- The ratio of the movement of the bottom line to the net oxide thickness change is controlled by **VOL.RAT**. **VOL.RAT** defaults to 0.44, an appropriate value for thermally grown oxide.
- The mesh above and below the specified region is also distorted by the spread operation. To maintain a rectangular grid in the substrate, the third **SPREAD** statement places the fourth grid line at the vertical coordinate 0.125 microns, its original location.
- A very large **WIDTH** is given to place the transition region outside the device. Placing the first nonspread grid line at its original location prevents any distortion of the rest of the grid.

Device Regions

The regions of the device are defined with the **REGION** statements. The first **REGION** statement defines the entire structure to be silicon. The second **REGION** statement then redefines the three uppermost grid lines to be oxide.

Electrode Locations

The **ELECTR** statements specify the location of the electrodes within the device. In this example:

- The gate is placed at the top surface of the oxide.
- The substrate contact is placed along the bottom of the device.
- The source and drain contacts are placed along the oxide-silicon interface at the left and right edges of the device.

Impurity Profiles

The impurity profiles are created analytically from Gaussian functions. Alternatively, they could also have been read in from Avant! SUPREM-3, TSUPREM-4, or 1D and 2D formatted files. In this example:

- Because an n-channel enhancement device is being created:
  - The first **PROFILE** statement specifies that a uniform p-type substrate is to be used.
  - The second **PROFILE** statement introduces a p-type threshold adjustment profile.
- The remaining **PROFILE** statements define the n+ source and drain regions.

  The source and drain are specified to have a junction depth of 0.34 microns with a lateral extent that is 0.75 times their vertical extent.
- The **INTERFAC** statement places a uniform fixed charge along the entire oxide-silicon interface.

Output File Specification

The output file specification on the first **PROFILE** statement saves the profile information so that whenever the grid is refined, the impurity distribution can be regenerated from the original profile specification.

Note:

If the output file specification is not done, the doping at the nodes of the refined mesh are interpolated from the doping at the nodes of the unrefined mesh. An output file should be specified when possible to avoid interpolation errors.

Initial Grid Plot

Figure 4-3 shows the device structure and the initial grid before any refinement is done.

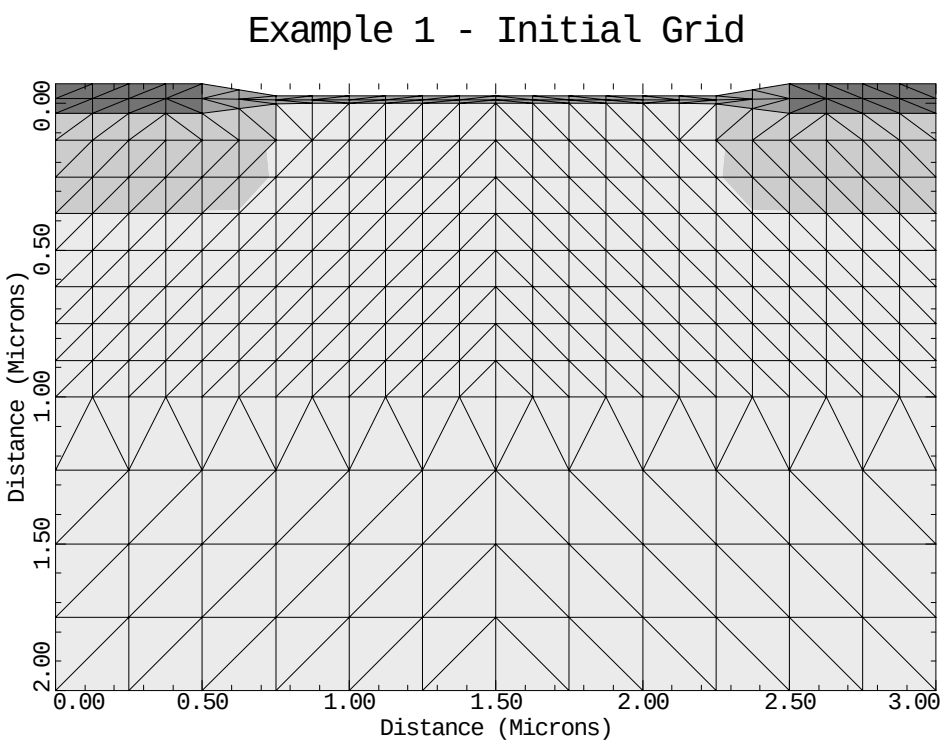


Figure 4-3 Initial grid from **PLOT.2D** at line 30 in file *mdex1*, Figures 4-1 and 4-2

Grid Refinement

At this point the device structure has been defined. It is now necessary to refine the grid so that it is adequate for a simulation.

Doping Regrid

The first phase of grid refinement is requested with the **REGRID** statement on line 32 in Figure 4-1.

Triangulation

The **REGRID** statement causes an existing triangle to be subdivided into four congruent triangles whenever the impurity concentrations at the nodes of the triangle differ by more than two orders of magnitude.

Smoothing

Smoothing is specified to minimize the adverse effects caused by obtuse triangles. The **IGNORE** parameter is set equal to the oxide region so that neither grid refinement nor smoothing are done in the oxide.

Profile File

The saved profile file is used for finding the impurity concentrations on the new grid.

Doping Regrid Plot

The resulting grid is shown in Figure 4-4 where the junction locations are clearly discernible with the increased grid density.

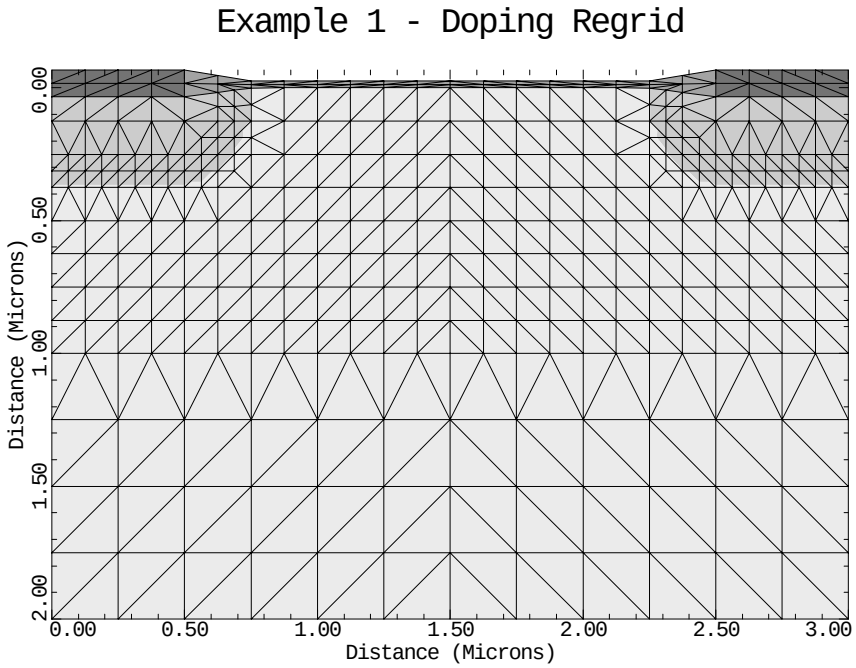


Figure 4-4 Doping regrid from **PLOT.2D** statement at line 33 in file *mdex1,* Figures 4-1 and 4-2

Potential Regrid

The second phase of grid refinement is based on the potential difference between nodes and therefore requires a solution be obtained on the existing grid.

Material and Contact

The gate material is selected to be n+ polysilicon with the **CONTACT** statement at line 35.

Models

Various models are chosen before beginning a solution.

- Concentration and electric field dependent mobility models are chosen with the parameters **CONMOB** and **FLDMOB**, respectively.
- Surface mobility reduction is accounted for by specifying **SRFMOB2**.

Solution

- A Poisson-only solution is selected by setting **CARRIERS** equal to zero on the **SYMB** statement, because only the potential is needed at this point
- In most cases, specifying **ICCG** and **DAMPED** on the **METHOD** statement results in the most efficient zero-carrier simulation.
- The **SOLVE** statement is used to generate the solution. The initial biases are defaulted to zero.

Absolute Change in Potential

The grid refinement based on potential (line 42), is performed in much the same way as the refinement based on impurity concentration. The absence of the **LOG** parameter means that refinement is based on the absolute change in potential which is specified with **RATIO** to be 0.2V.

Triangulation

The **MAX** parameter is set to 1 to prevent the triangles of the original mesh from being subdivided more than once. **MAX** refers to the maximum number of times the grid can be subdivided relative to the original grid. It defaults to 1 more than the current maximum level of the grid.

The most efficient grid refinement occurs when **MAX** is 1 more than the previous maximum for a refinement based on the same quantity. Because this is the first potential refinement, **MAX** should start at 1.

Output File

This is the last refinement, so the mesh is saved in an output file for subsequent simulations.

Potential Regrid Plot

The final mesh is plotted in Figure 4-5.

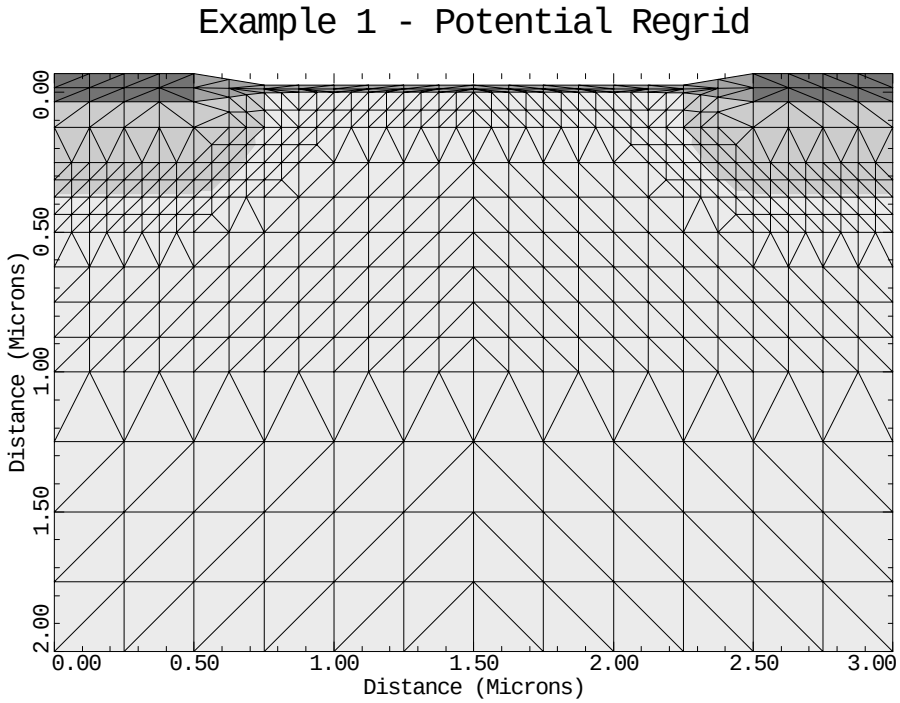


Figure 4-5 Potential regrid from **PLOT.2D** at line 43 in file *mdex1*, Figures 4-1 and 4-2

Saving Zero Bias Solution

To provide a starting point for subsequent simulations, a zero bias solution is obtained and saved for the final mesh.

The **SYMB** statement must be specified again before using the **SOLVE** statement to obtain the next solution. This is because the number of nodes in the mesh has changed since the last solution was obtained.

The current level in the device is expected to be very low with no bias applied, so it is sufficient to obtain and save a zero-carrier solution.

Impurity Distribution Plots

Figures 4-6 through 4-8 show the impurity distribution for this device as a consequence of the plot statements at the end of the input file.

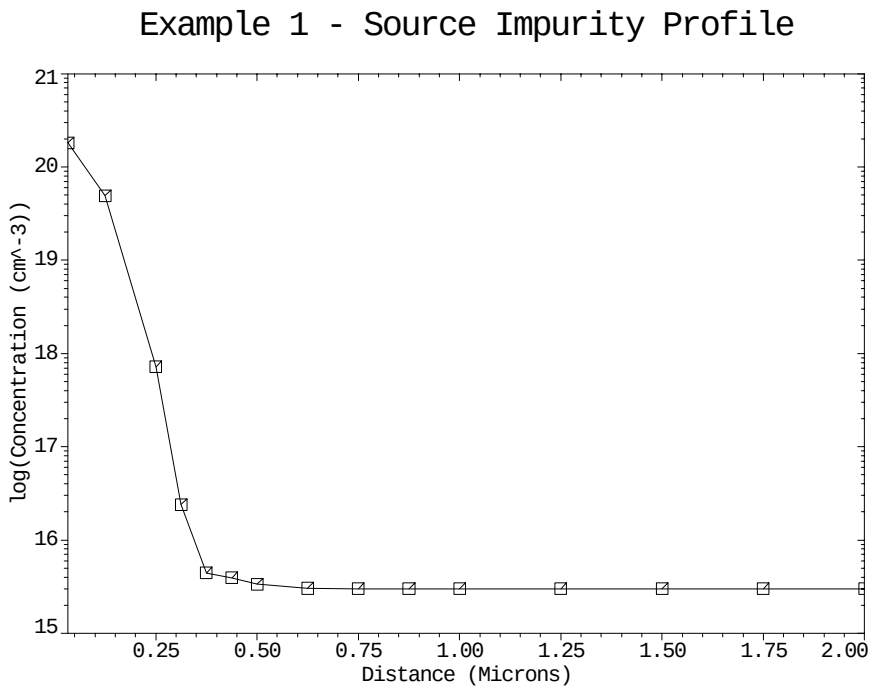


Figure 4-6 Source impurity profile from **PLOT.1D** at line 48 in file *mdex1*, Figures 4-1 and 4-2

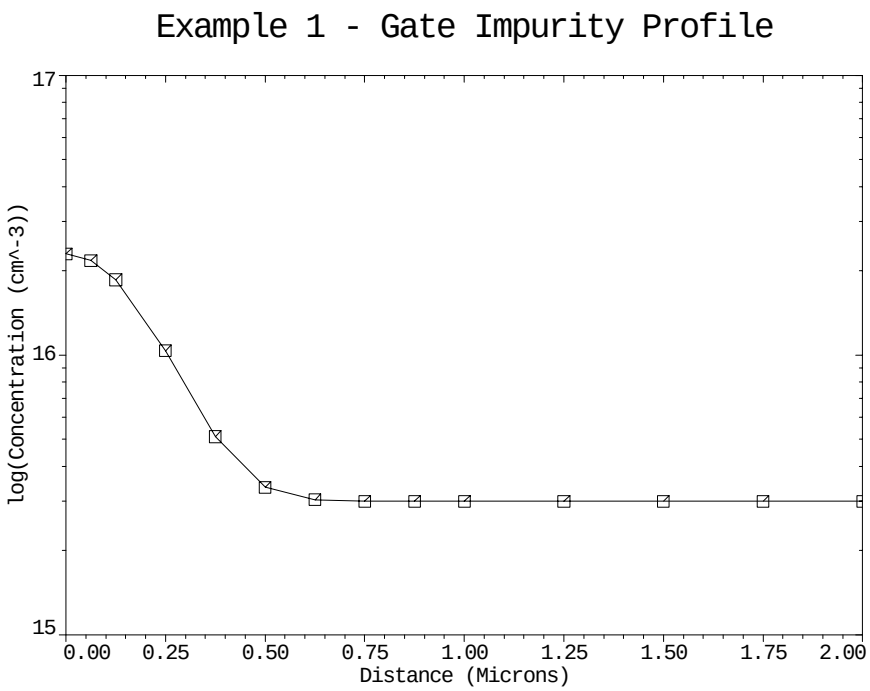


Figure 4-7 Gate impurity profile from **PLOT.1D** at line 49 in file *mdex1,* Figures 4-1 and 4-2

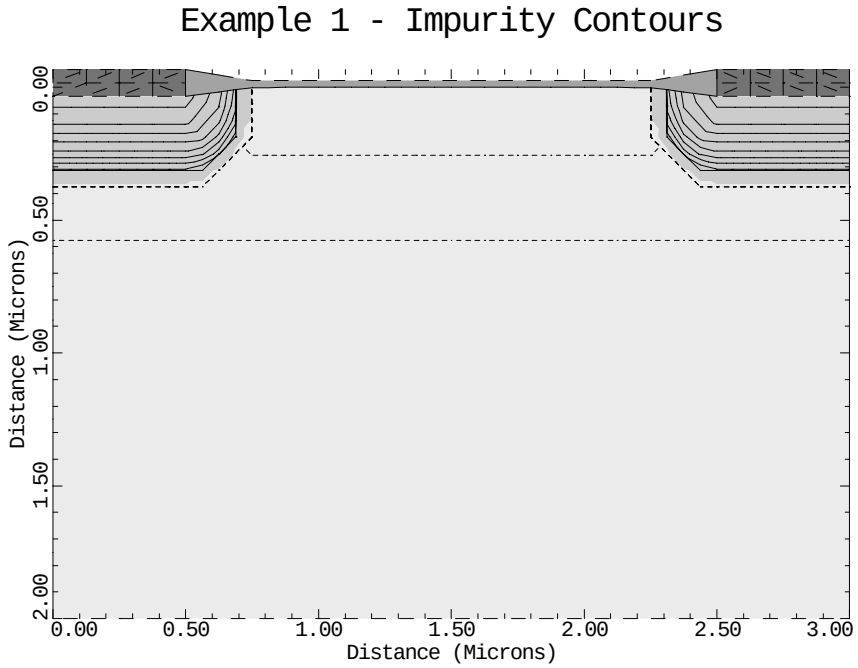


Figure 4-8 Impurity contours from **PLOT.2D** and **CONTOUR** at lines 50 through 52 in file *mdex1*, Figures 4-1 and 4-2

Simulation of Gate Characteristics

The device structure and initial solution that were created and saved by the input file *mdex1* are read by the input file *mdex1g*. Simulations are performed for a drain bias of 0.1V, and gate biases of 0V to 2V. Figures 4-9 and 4-10 contain the output associated with the execution of Medici for the input file *mdex1g*.

```
 1... TITLE     Avant! MEDICI Example 1G - 1.5 Micron N-Channel MOSFET
 2... COMMENT   Calculate Gate Characteristics

 3... COMMENT   Read in simulation mesh
 4... MESH      IN.FILE=MDEX1MS

 5... COMMENT   Read in saved solution
 6... LOAD      IN.FILE=MDEX1S

 7... COMMENT   Use Newton's method for the solution
 8... SYMB      NEWTON  CARRIERS=1  ELECTRONS

 9... COMMENT   Setup log file for IV data
10... LOG       OUT.FILE=MDEX1GI

11... COMMENT   Solve for Vds=0.1 and then ramp gate
12... SOLVE     V(Drain)=.1
13... SOLVE     V(Gate)=.2  ELEC=Gate  VSTEP=.2  NSTEP=9

14... COMMENT   Plot Ids vs. Vgs
15... PLOT.1D   Y.AXIS=I(Drain)  X.AXIS=V(Gate)  POINTS  COLOR=2
  ... +         TITLE="Example 1G - Gate Characteristics"
16... LABEL     LABEL="Vds = 0.1v"  X=1.6  Y=0.7E-6
```

Figure 4-9 Output of the simulation input file *mdex1g*

Calculating Gate Characteristics

The simulation of gate characteristics uses the following inputs.

Initialization

Because the solution file does not contain mesh information, both the mesh and the solution files must be loaded using the **LOAD** statement on line 6.

Solve Options

Newton's method is chosen as the most efficient solution technique on the **SYMB** statement. Because this is an n-channel MOSFET, it is only necessary to solve Poisson's equation and the electron continuity equation. This is accomplished by specifying **CARRIERS**=1 and **ELECTRONS** on the **SYMB** statement.

Log File

Before performing the solutions, a log file is specified to save the I-V data for later plotting.

Gate Characteristics

In obtaining the solutions, 0.1V is applied to the drain and then the gate is stepped to 2V in 0.2V increments. The resulting I-V curve, which is plotted in Figure 4-10, is useful in determining the threshold voltage. The threshold voltage can also be extracted directly using the **EXTRACT** statement with the **MOS.PARA** parameter.

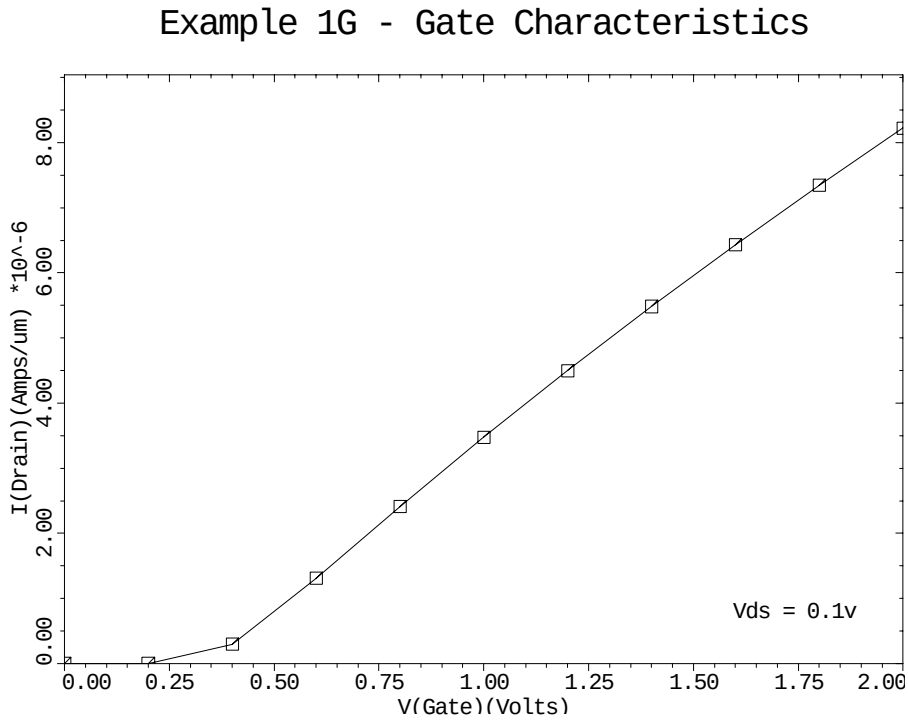


Figure 4-10 Gate characteristics from **PLOT.1D** and **LABEL** at lines 15 and 16 file *mdex1g*, Figure 4-9

Simulation of Drain Characteristics

The device structure and initial solution that were created and saved by the input file *mdex1* are read by the input file *mdex1d*. Simulations are performed for:

- A gate bias of 3V
- Drain biases of 0V to 3V

Figures 4-11 through 4-13 contain the output associated with the execution of Medici for the input file *mdex1d*.

```
 1... TITLE     Avant! MEDICI Example 1D - 1.5 Micron N-Channel MOSFET
 2... COMMENT   Calculate Drain Characteristics

 3... COMMENT   Read in simulation mesh
 4... MESH      IN.FILE=MDEX1MS

 5... COMMENT   Read in initial solution
 6... LOAD      IN.FILE=MDEX1S

 7... COMMENT   Do a Poisson solve only to bias the gate
 8... SYMB      CARRIERS=0
 9... METHOD    ICCG  DAMPED
10... SOLVE     V(Gate)=3.0

11... COMMENT   Use Newton's method and solve for electrons
12... SYMB      NEWTON  CARRIERS=1  ELECTRON

13... COMMENT   Setup log file for IV data
14... LOG       OUT.FILE=MDEX1DI

15... COMMENT   Ramp the drain
16... SOLVE     V(Drain)=0.0  ELEC=Drain  VSTEP=.2  NSTEP=15

17... COMMENT   Plot Ids vs. Vds
18... PLOT.1D   Y.AXIS=I(Drain)  X.AXIS=V(Drain)  POINTS  COLOR=2
  ... +         TITLE="Example 1D - Drain Characteristics"
19... LABEL     LABEL="Vgs = 3.0v"  X=2.4  Y=0.1E-4

20... COMMENT   Potential contour plot using most recent solution
21... PLOT.2D   BOUND  JUNC  DEPL  FILL  SCALE
  ... +         TITLE="Example 1D - Potential Contours"
22... CONTOUR   POTENTIA  MIN=-1  MAX=4  DEL=.25  COLOR=6
23... LABEL     LABEL="Vgs = 3.0v"  X=0.2  Y=1.6
24... LABEL     LABEL="Vds = 3.0v"
```

Figure 4-11 Output of the simulation input file *mdex1d*

Procedures

The simulation of drain characteristics uses the following procedures.

Initial Solution

To bias the gate before generating the drain characteristics, first perform a zero-carrier solution where 3V is applied directly to the gate, without stepping the gate bias up from 0V.

Solve Options

After obtaining the solution, use Newton's method to solve for electrons.

Log File

Before performing the final solution sequence, line 14 uses the **LOG** statement to specify a log file to save the I-V data for later plotting.

Drain Characteristics

In line 16, the drain ramps to 3V in steps of 0.2V. The resulting I-V curve is plotted in Figure 4-12.

Potential Contours Plot

Figure 4-13 shows the potential contours for the final bias point.

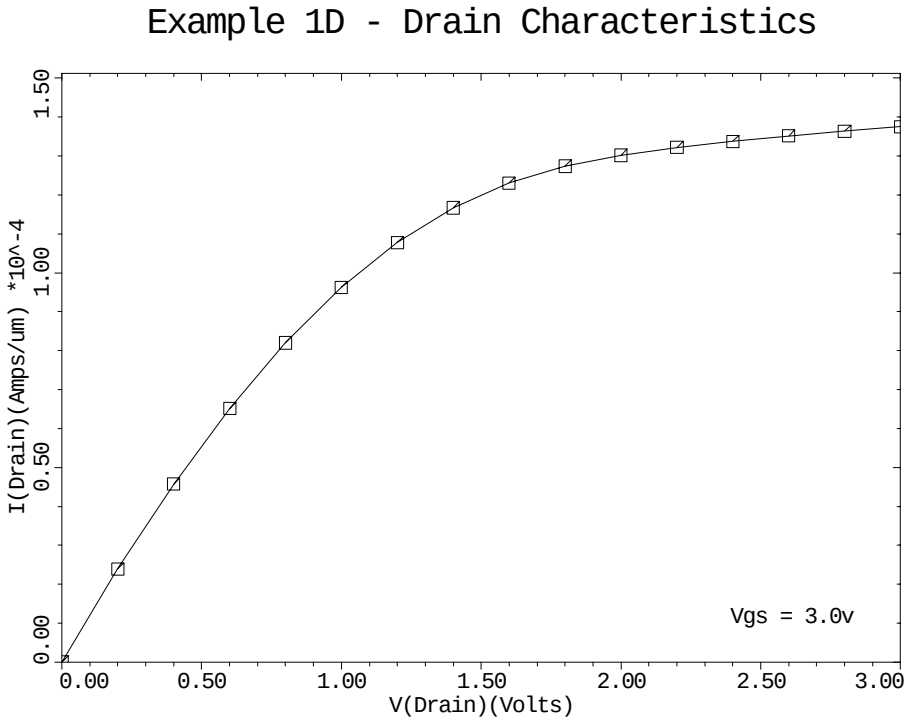


Figure 4-12 Drain characteristics from **PLOT.1D** and **LABEL** at lines 18 and 19 in file *mdex1d,* Figure 4-11

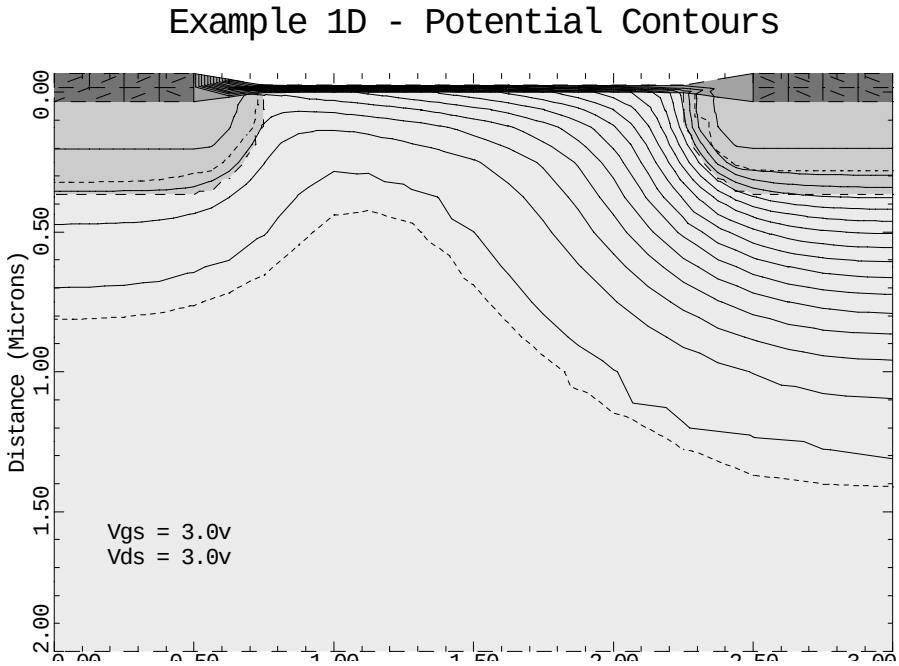


Figure 4-13 Potential contours from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 21 through 24 in file *mdex1d,* Figure 4-11

Substrate and Gate Current Calculation

The device structure and initial solution that were created and saved by the input file *mdex1* are read by the input file *mdex1i*.

- Simulations are performed for a drain bias of 5V and gate biases of 0V to 7.5V.
- An impact ionization and gate current analysis are performed after each solution.

Figures 4-14 through 4-17 contain the output associated with the execution of Medici for the input file *mdex1i*.

```
 1... TITLE     Avant! MEDICI Example 1I - 1.5 Micron N-Channel MOSFET
 2... COMMENT   Substrate and Gate Current Calculation

 3... COMMENT   Read in simulation mesh
 4... MESH      IN.FILE=MDEX1MS

 5... COMMENT   Read in saved solution
 6... LOAD      IN.FILE=MDEX1S

 7... COMMENT   Perform a zero carrier solution to bias the drain
 8... SYMB      CARR=0
 9... METHOD    DVLIMIT=1  ICCG

10... COMMENT   Solve for Vds=5.
11... SOLVE     V(Drain)=5.  LOCAL

12... COMMENT   Switch to 1-Carrier Newton
13... SYMB      NEWTON  CARRIERS=1  ELECTRONS

14... COMMENT   Setup log file for IV data
15... LOG       OUT.FILE=MDEX1II

16... COMMENT   Ramp the gate, performing an impact ionization
17... $         and gate current analysis at each bias.
18... SOLVE     V(Gate)=0.  ELEC=Gate  VSTEP=.5  NSTEP=15
  ... +         IMPACT.I  GATE.CUR

19... COMMENT   Plot substrate current versus gate bias
20... PLOT.1D   Y.AXIS=II  X.AXIS=V(Gate)  Y.LOG  POINTS  COLOR=2
  ... +         TITLE="Example 1I - Substrate Current vs. Vgs"
21... LABEL     LABEL="Vds = 5.0v"  X=6.0  Y=4E-12

22... COMMENT   Plot gate current versus gate bias
23... PLOT.1D   Y.AXIS=HE(Gate)  X.AXIS=V(Gate)  Y.LOG  POINTS  COLOR=2
  ... +         LEFT=0.  BOT=1E-19  TOP=1E-13
  ... +         TITLE="Example 1I - Gate Current vs. Vgs"
24... LABEL     LABEL="Vds = 5.0v"  X=.4  Y=3E-14

25... COMMENT   Plot generation rate contours
26... PLOT.2D   BOUND  JUNC  X.MIN=2.0  Y.MAX=.7  FILL  SCALE
  ... +         TITLE="Example 1I - Generation Rate Contours"
27... CONTOUR   II.GENER  LOG  ABS  MIN=17  MAX=27  DEL=1  COLOR=2
28... LABEL     LABEL="Oxide"       X=2.4  Y=-0.01
29... LABEL     LABEL="Drain"       X=2.6  Y=0.20
30... LABEL     LABEL="Substrate"   X=2.4  Y=0.55
31... LABEL     LABEL="Vgs = 7.5v"  X=2.8  Y=0.55
32... LABEL     LABEL="Vds = 5.0v"
```

Figure 4-14 Output of the simulation input file *mdex1i*

Procedures

The simulation of substrate and gate current calculation uses the following procedures.

Initial Solution

Sometimes initial biases applied to a structure are such that only small amounts of current flow in the device (for example, when junctions are reverse biased and/or the device is turned off). It is often possible to apply these biases directly to the structure (that is, without stepping the bias to its final value). This can be accomplished by first performing a zero-carrier solution at the desired bias and then performing a solution with carriers.

In this example, it is desired to apply 5V to the drain of the device with 0V applied to the gate. This is accomplished by first specifying a zero-carrier solution on the **SYMB** statement.

For large bias changes, it is often advantageous to specify the maximum potential update allowed at a node during an iteration (**DVLIMIT**). In line 9, **DVLIMIT** is increased from its default of 0.1V to 1V.

On the **SOLVE** statement in line 11, in addition to specifying the applied drain bias of 5V, **LOCAL** is also specified. This causes Medici to use the previous solution as the initial guess, but modifies the quasi-Fermi potential in the drain region to be equal to the applied bias.

After the zero-carrier solution is obtained, the example uses Newton's method and solve for electrons. Since this is an MOS device, it is only necessary to solve for the channel carrier.

Log File

Before performing the solutions, a log file is specified to save the I-V data for later plotting.

Impact Ionization and Gate Current

In line 18 the gate bias is ramped from 0V to 7.5V in steps of 0.5V. The parameters **IMPACT.I** and **GATE.CUR** request that an impact ionization analysis and gate current analysis, respectively, be performed after each solution.

Substrate Current vs. Vgs Plot

Figure 4-15 plots the resulting substrate current obtained from the impact ionization analysis. The figure shows that the substrate current rises very rapidly as the device is turned on due to the increase of channel current.

As the gate bias is increased, the electric field in the drain region of the device decreases. The trade-off between increasing current density and decreasing electric field as gate bias is increased causes the substrate current to be peaked. In this case, the peak value occurs at a gate bias of approximately 2.5V.

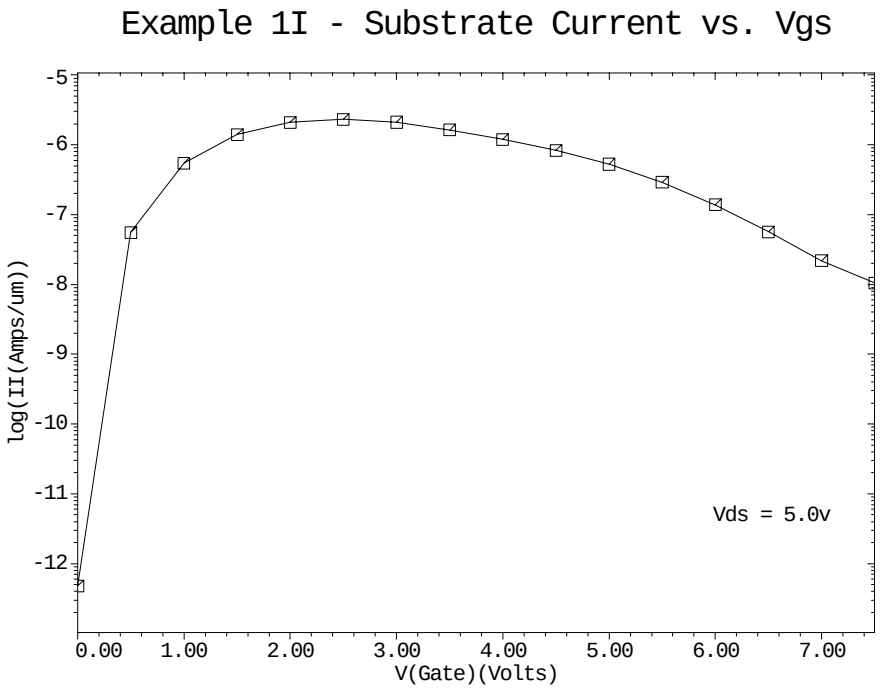


Figure 4-15 Substrate current vs. Vgs from **PLOT.1D** and **LABEL** at lines 20 and 21 in file *mdex1i*, Figure 4-14

Gate Current vs. Vgs Plot

Figure 4-16 plots gate current as a function of gate bias obtained from the gate current analysis. At low gate biases the electric field in the drain region of the device is high, but the oxide electric field near the drain is such that electron injection into the gate is inhibited. This causes the gate current for low values of V_{gs} to be extremely small.

As V_{gs} is increased towards V_{ds}, the oxide electric field becomes more favorable for electron injection and there is a rapid rise in the gate current. As in the substrate current case described above, the decreasing electric field in the drain region as V_{gs} is increased causes the gate current curve to be peaked.

In this example, the peak value occurs at a gate bias of approximately 4.0V. Also note that the peak value of gate current is approximately eight orders of magnitude smaller than the peak value of substrate current.

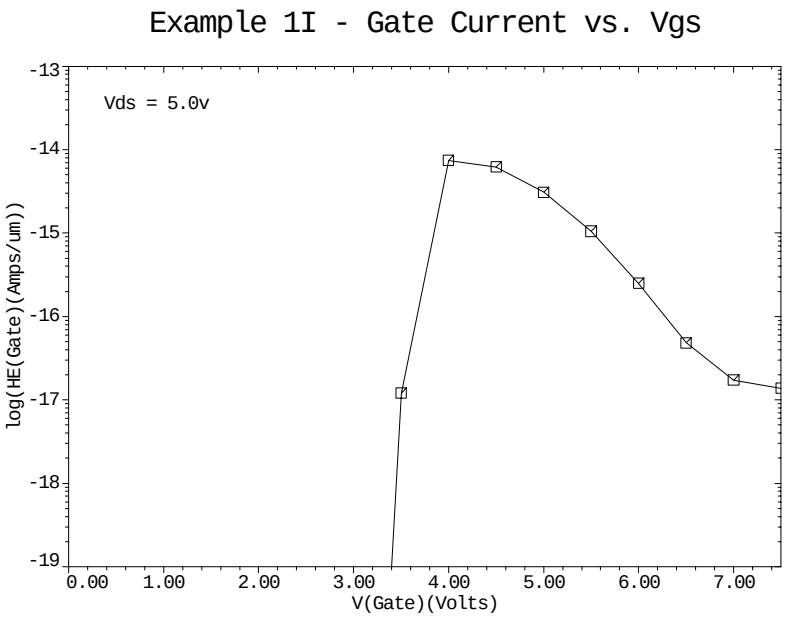


Figure 4-16 Gate current vs. Vgs from **PLOT.1D** and **LABEL** at lines 23 and 24 in file *mdex1i*, Figure 4-14

Generation Rate Contours Plot

Contours of the total impact ionization generation rate for the last bias point (V_{gs}=7.5V) are plotted in Figure 4-17. The interval between contours is one order of magnitude. This figure shows that most of the impact ionization occurs at the silicon-oxide interface in the vicinity of the channel-drain region.

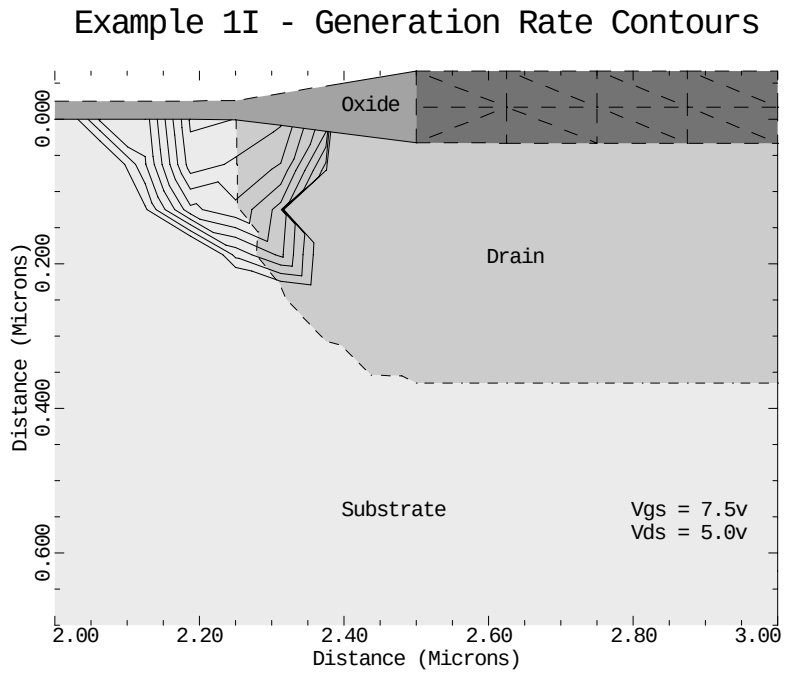


Figure 4-17 Generation rate contours from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 26 through 32 in file *mdex1i,* Figure 4-14

Analysis Including Fast Interface States

The device structure and initial solution that were created and saved by the input file *mdex1* are read by the input file *mdex1f*. Gate characteristic simulations are performed for three separate cases:

- No interface charge
- Fast positive interface states
- Fast negative interface states

Figures 4-18 through 4-23 contain the output associated with the execution of Medici for the input file *mdex1f*.

```
 1... TITLE     Avant! MEDICI Example 1F - 1.5 Micron N-Channel MOSFET
 2... COMMENT   Gate Characteristics Including Fast Interface States

 3... COMMENT   Read in saved mesh
 4... MESH      IN.FILE=MDEX1MS

 5... COMMENT   Read solution to get models
 6... LOAD      IN.FILE=MDEX1S
 7... INTERFAC  CLEAR

 8... COMMENT   0-carrier solution with Vg=-0.6v, Vd=0.1v
 9... SYMB      CARRIERS=0
10... SOLVE     INIT  V(Gate)=-0.6  V(Drain)=0.1  OUT.FILE=TEMPSOL

11... COMMENT   Switch to 1-carrier
12... SYMB      NEWTON  CARRIERS=1  ELECTRON

13... COMMENT   Gate characteristics with zero, positive, and negative
   ... +        fast interface states (Vd=0.1v)
14... LOOP      STEPS=3
15...   ASSIGN    NAME=NDON    N.VALUE=(0.0, 5E11, 0.0)
16...   ASSIGN    NAME=NACC    N.VALUE=(0.0, 0.0 , 5E11)
17...   ASSIGN    NAME=LOGFIL  C1="MDE1FIZ"  C2="MDE1FIP"  C3="MDE1FIN"
18...   ASSIGN    NAME=SOLFIL  C1="MDE1SZ"   C2="MDE1SP"   C3="MDE1SN"

19...   LOAD      IN.FILE=TEMPSOL
20...   INTERFAC  N.DON=@NDON  N.ACC=@NACC
21...   LOG       OUT.FILE=@LOGFIL
22...   SOLVE     V(Gate)=-0.6  OUT.FILE=@SOLFIL"1"
23...   SOLVE     V(Gate)=-0.4  ELEC=Gate  VSTEP=0.2  NSTEP=11
24...   SOLVE     V(Gate)=2.0   OUT.FILE=@SOLFIL"2"
25... L.END
```

Figure 4-18 First part of the simulation input file *mdex1f*

```
26... COMMENT   Plot log(drain current) versus Vgs
27... PLOT.1D   IN.FILE=MDE1FIP  Y.AXIS=I(Drain)  X.AXIS=V(Gate)  Y.LOG
  ... +         BOT=1E-14  TOP=1E-4  LINE=1  COLOR=2  SYMB=1
  ... +         TITLE="Example 1F - Log(Id) vs. Vgs"
28... PLOT.1D   IN.FILE=MDE1FIZ  Y.AXIS=I(Drain)  X.AXIS=V(Gate)  Y.LOG
  ... +         LINE=1  COLOR=3  SYMB=2  UNCHANGE
29... PLOT.1D   IN.FILE=MDE1FIN  Y.AXIS=I(Drain)  X.AXIS=V(Gate)  Y.LOG
  ... +         LINE=1  COLOR=4  SYMB=3  UNCHANGE
30... LABEL     LAB="Positive States" COL=2  SYMB=1  START.LE  LX.FIN=-.5
  ... +         X=-0.3
31... LABEL     LAB="Zero Charge"     COL=3  SYMB=2  START.LE  LX.FIN=-.5
32... LABEL     LAB="Negative States" COL=4  SYMB=3  START.LE  LX.FIN=-.5

33... COMMENT   Plot band diagrams at X=1.5
34... LOOP      STEPS=3
35...   ASSIGN    NAME=SOLFIL  C1="MDE1SZ"   C2="MDE1SP"   C3="MDE1SN"
36...   ASSIGN    NAME=TITLE   C1="Zero Charge"  C2="Positive States"
  ...   +                      C3="Negative States"

37...   LOOP      STEPS=2
38...     ASSIGN    NAME=NUM    N.VALUE=(1,2)
39...     ASSIGN    NAME=XOFF   N.VALUE=(2.0,11.0)
40...     ASSIGN    NAME=CLEAR  L.VALUE=(T,F)
41...     ASSIGN    NAME=BIAS   C1="Vg = -0.6v"  C2="Vg = 2.0v"

42...     LOAD      IN.FILE=@SOLFIL@NUM

43...     IF COND=@CLEAR
44...       PLOT.1D   CONDUC  COL=4  BOT=2  TOP=-2  X.LEN=7  X.OFF=@XOFF
  ...       +         X.ST=1.5  X.EN=1.5  Y.ST=0  Y.EN=0.5  TITLE=@TITLE
45...     ELSE
46...       PLOT.1D   CONDUC  COL=4  BOT=2  TOP=-2  X.LEN=7  X.OFF=@XOFF
  ...       +         X.ST=1.5  X.EN=1.5  Y.ST=0  Y.EN=0.5  TITLE=@TITLE
  ...       +         ^CLEAR
47...     IF.END

48...     PLOT.1D   VALENC  COL=4  UNCHANGE  X.LEN=7  X.OFF=@XOFF
  ...     +         X.ST=1.5  X.EN=1.5  Y.ST=0  Y.EN=0.5
49...     PLOT.1D   POTEN   COL=3  UNCHANGE  X.LEN=7  X.OFF=@XOFF
  ...     +         X.ST=1.5  X.EN=1.5  Y.ST=0  Y.EN=0.5
50...     PLOT.1D   QFN     COL=2  UNCHANGE  X.LEN=7  X.OFF=@XOFF  LINE=2
  ...     +         X.ST=1.5  X.EN=1.5  Y.ST=0  Y.EN=0.5

51...     LABEL     LABEL=@BIAS
52...     LABEL     LABEL=Conduction  X=.33  Y=-0.95  C.SI=.2
53...     LABEL     LABEL=Potential   X=.33  Y=-0.40  C.SI=.2
54...     LABEL     LABEL=Qfn         X=.33  Y=-0.05  C.SI=.2
55...     LABEL     LABEL=Valence     X=.33  Y=0.35   C.SI=.2
56...   L.END

57... L.END
```

Figure 4-19 Second part of simulation input file *mdex1f*

Procedures

The analysis with fast interface states uses the following procedures.

Removing the Interface Charge

The simulation structure created by the input file *mdex1* and shown in Figures 4-1 and 4-2 specifies a fixed charge to be placed at the interface. In preparation for studying the effect of fast interface states on the results of a simulation, this interface charge should be removed before creating any new solutions. This is accomplished by the **INTERFAC** statement at line 7 in the input file *mdex1f* shown in

Figures 4-18 and 4-19, which is specified *after* the device structure and initial solution are read in.

Note:

Parameters associated with interfaces are stored in solution files (such as MDEX1S) and not in mesh files.

Initial Solution

At line 10 in the input file *mdex1f*, a 0-carrier solution is performed at the starting bias which is used for each of the subsequent gate sweeps. The drain voltage is specified to be 0.1V and the gate voltage is specified to be -0.6V, which is low enough to allow subthreshold characteristics to be studied.

This initial solution is stored in a temporary solution file, and is used as the initial guess for the first solution of each of the three gate sweeps to follow.

Interface State Parameters

After specifying that 1-carrier solutions for electrons (using Newton's method) are desired, the input statement loop from lines 14 through line 25 is used to sweep the gate bias for the three cases of interest. Within this loop, the assigned names *NDON* and *NACC* are used to specify values that are used for the **INTERFAC** statement parameters **N.DONOR** and **N.ACCEPT**, respectively.

- The parameter **N.DONOR** represents the density of fast electron-donor states.

  These states are positively charged above the electron quasi-Fermi potential and neutral below.
- The parameter **N.ACCEPT** represents the density of fast electron-acceptor states.

  These states are neutral above the electron quasi-Fermi potential and negatively charged below.

Three Interface State Cases

The program now processes three passes through the loop.

- For the first pass through the loop, the values for both **N.DONOR** and **N.ACCEPT** are set to 0.0.

  This pass represents the "zero charge" case.
- The second pass specifies a state density of $5 \times 10^{11}/\text{cm}^2\text{-eV}$ for **N.DONOR**.

  This corresponds to the "positive states" case.
- The third pass specifies a state density of $5 \times 10^{11}/\text{cm}^2\text{-eV}$ for **N.ACCEPT**.

  This corresponds to the "negative states" case.

Within the loop, the gate bias is swept from -0.6V to 2.0V.

Log File

A separate log file for storing the I-V characteristics is created for each gate sweep. The statements within the loop also specify that the solutions corresponding to the first and last bias point of each gate sweep should be saved in files.

Graphical Output

The results of this simulation are plotted using the **PLOT.1D** and **LABEL** statements at lines 27 through 32. The graphical output generated by these lines is shown in Figure 4-20.

Subthreshold Characteristics

As might be expected, the inclusion of fast positive states shifts the $\log(I_{Drain})$ vs. V_{Gate} curve to the left, and the inclusion of fast negative states shifts this curve to the right.

It is also evident that the subthreshold slope changes when fast interfaces states are included. This can be understood by studying the band diagrams associated with different bias conditions for each of the cases. This is explained in the following paragraphs.

Band Diagrams

The doubly nested input statement loop beginning at line 37 and ending at line 57 is used to plot band diagrams for all three cases considered (zero charge, positive states, and negative states) for two different bias conditions (V_{Gate} =-0.6V and 2.0V). The output generated by the statements within this loop are shown in Figures 4-21 through 4-23.

Positive States Case

Consider first the positive states case (Figure 4-22). The **N.DONOR** states are positively charged above the electron quasi-Fermi potential. This results in the total positive charge that is included at the interface to be proportional to the potential difference between the electron quasi-Fermi potential and the conduction band potential.

As the gate bias is increased, the band-bending occurring at the surface causes this potential difference to become smaller. Consequently, a smaller amount of positive charge is included at the interface. This causes the "positive states" curve shown in Figure 4-20 to be shifted farther to the left of the "zero charge" curve at lower gate biases than at higher gate biases.

Negative States Case

For the negative states case (Figure 4-23), a similar explanation can be given. As mentioned previously, the **N.ACCEPT** states are negatively charged below the electron quasi-Fermi potential. This results in the total negative charge that is included at the interface to be proportional to the potential difference between the electron quasi-Fermi potential and the valence band potential.

As the gate bias is increased, the band bending occurring at the surface causes this potential difference to become larger. Consequently, a larger amount of negative charge is included at the interface. This causes the "negative" states curve shown in Figure 4-20 to be shifted farther to the right of the "zero charge" curve at higher gate biases than at lower gate biases.

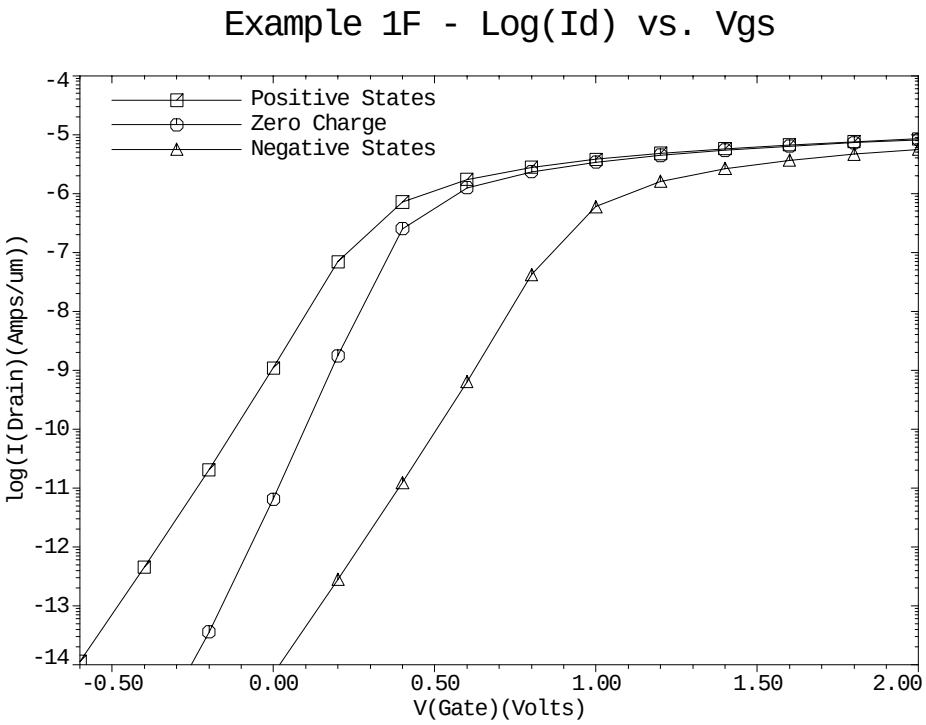


Figure 4-20 Log(Id) vs. Vgs from **PLOT.1D** and **LABEL** at lines 27 through 32 in file *mdex1f*, Figure 4-19

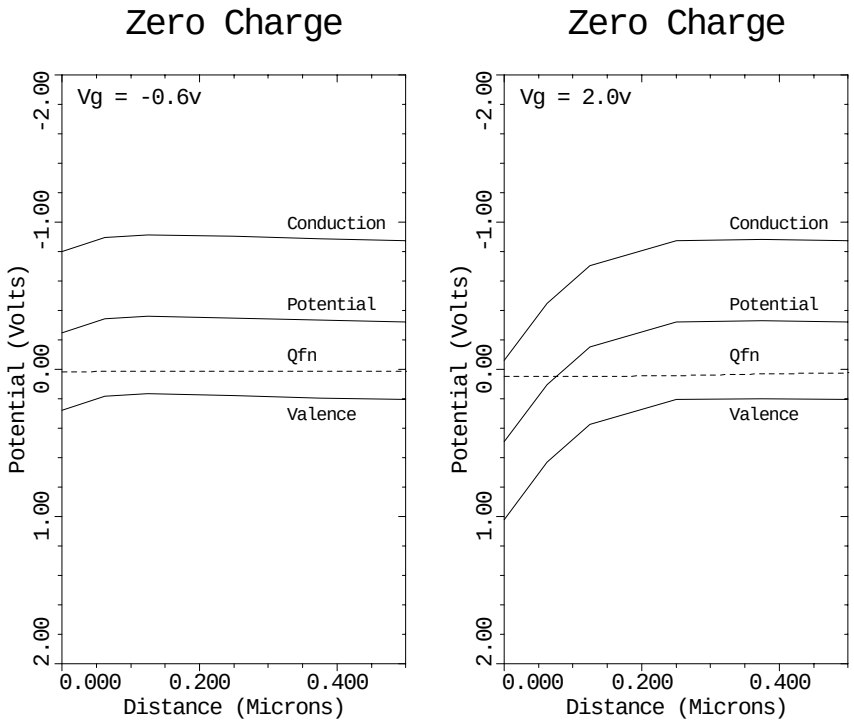


Figure 4-21 First pass through the outermost loop from **PLOT.1D** and **LABEL** at lines 34 through 57 in file *mdex1f,* Figure 4-19

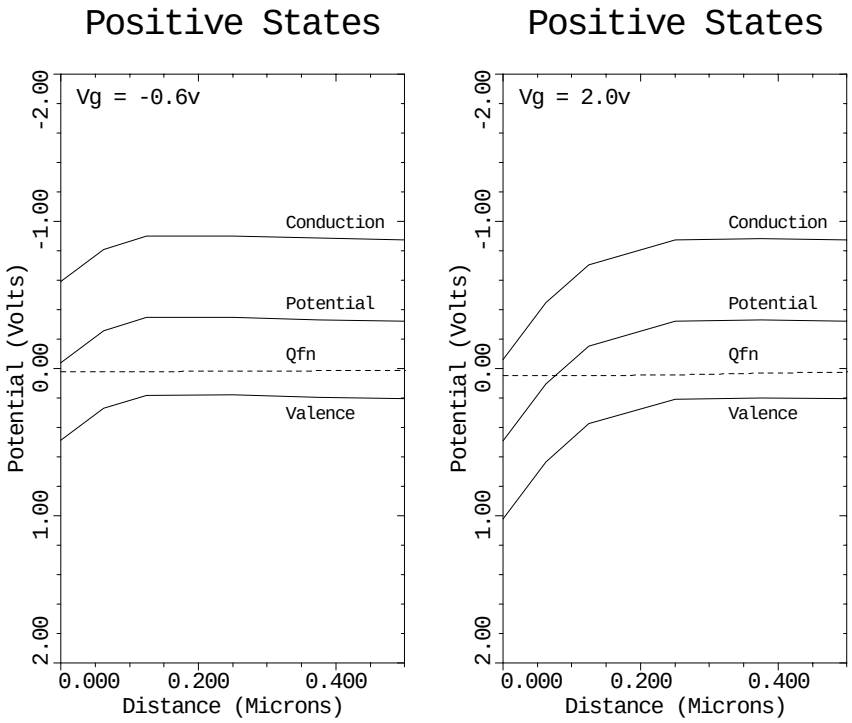


Figure 4-22 Second pass through the outermost loop from **PLOT.1D** and **LABEL** at lines 34 through 57 in file *mdex1f,* Figure 4-19

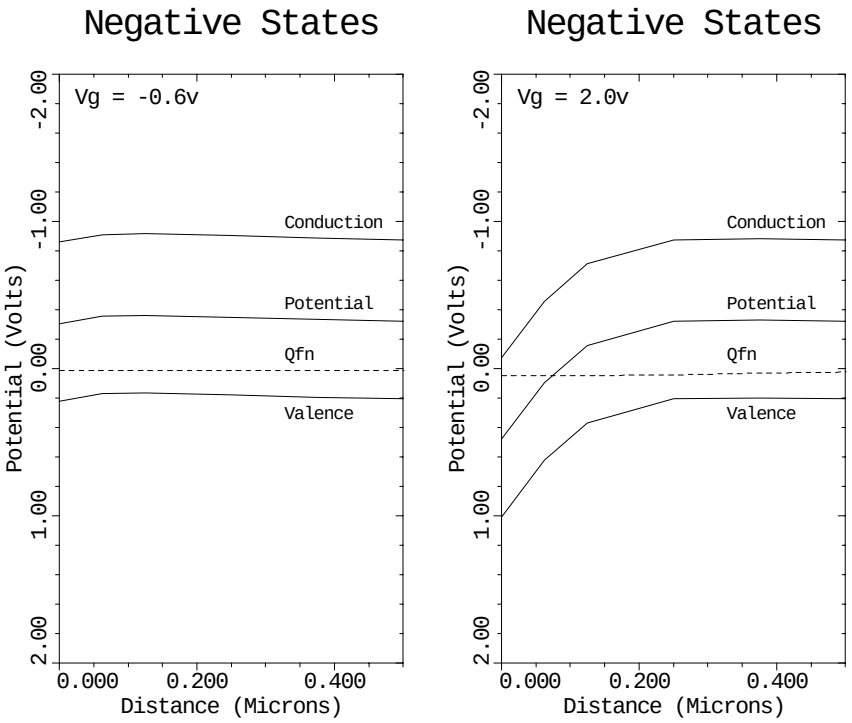


Figure 4-23 Third pass through the outermost loop from **PLOT.1D** and **LABEL** at lines 34 through 57 in file *mdex1f*, Figure 4-19

Analysis Including Band-to-Band Tunneling

The device structure and initial solution that were created and saved by the input file *mdex1* are read by the input file *mdex1t*.

Simulations are then performed to study the leakage current generated as a result of band-to-band tunneling when a 10V drain bias and negative gate biases are applied to the n-channel MOSFET. Figures 4-24 through 4-27 contain the output associated with the execution of Medici for the input file *mdex1t*.

```
 1... TITLE     Avant! MEDICI Example 1T - 1.5 Micron N-Channel MOSFET
 2... COMMENT   Simulation Including Band-to-Band Tunneling

 3... COMMENT   Read in saved mesh
 4... MESH      IN.FILE=MDEX1MS

 5... COMMENT   Read solution to get models
 6... LOAD      IN.FILE=MDEX1S

 7... COMMENT   Regrids on band-to-band tunneling rate with Vd=10v, Vg=-4v
 8... LOOP      STEPS=3
 9...   ASSIGN    NAME=INITIAL  L.VALUE=(T,F,F)
10...   SYMB      CARRIERS=0
11...   METHOD    ICCG  DAMPED

12...   IF COND=@INITIAL
13...     SOLVE     V(Gate)=-4  V(Drain)=10  INITIAL
14...   ELSE
15...     SOLVE     V(Gate)=-4  V(Drain)=10
16...   IF.END

17...   ASSIGN    NAME=BBRATE  N.VALUE=(6,10,14)
18...   REGRID    BB.GENER  IGNORE=OXIDE  LOG  ^CHANGE  RATIO=@BBRATE
  ...   +         SMOOTH=1  IN.FILE=MDEX1DS
19... L.END

20... COMMENT   Plot the simulation grid
21... PLOT.2D   GRID  FILL  SCALE
  ... +         TITLE="Example 1T - BTBT Simulation Grid"

22... COMMENT   Turn on band-to-band tunneling model
23... MODELS    BTBT

24... COMMENT   Initial 0-carrier solution with Vd=10v, Vg=0v
25... SYMB      CARRIERS=0
26... SOLVE     INITIAL  V(Gate)=0  V(Drain)=10

27... COMMENT   Switch to 2-carriers and then ramp the gate
28... SYMB      NEWTON  CARRIERS=2
29... METHOD    ^AUTONR
30... LOG       OUT.FILE=MDEX1TI
31... SOLVE     V(Gate)=0  ELEC=Gate  VSTEP=-0.25  NSTEP=16

32... COMMENT   Plot drain current versus gate bias
33... PLOT.1D   BOT=1E-14  TOP=1E-10  LEFT=-4  RIGHT=0  COLOR=2
  ... +         X.AXIS=V(Gate)  Y.AXIS=I(Drain)  Y.LOGARITH  POINTS
  ... +         TITLE="Example 1T - Drain Current vs. Vgs"
34... LABEL     LABEL="Vds = 10.0V"  Y=3E-14

35... COMMENT   Plot band-to-band generation rate contours
36... PLOT.2D   X.MIN=2.1  X.MAX=2.7  Y.MAX=0.3  BOUND  JUNC  FILL  SCALE
  ... +         TITLE="Example 1T - Generation Rate Contours"
37... CONTOUR   BB.GENER  LOG  ABS  MIN=13  MAX=23  DEL=1
38... LABEL     LABEL="Oxide"       X=2.40  Y=-0.01
39... LABEL     LABEL="Drain"       X=2.50  Y=0.10
40... LABEL     LABEL="Vgs = -4v"   X=2.55  Y=0.25
41... LABEL     LABEL="Vds = 10v"
```

Figure 4-24 Output of the simulation input file *mdex1t*

Procedures

The analysis including band-to-band tunneling uses the following procedures.

Grid Refinement

With a negative bias applied to the gate, it is expected that very little current flows through an n-channel MOSFET. But when a high drain bias is also applied, the band bending at the surface makes it possible for valence band electrons to tunnel through the forbidden gap to the conduction band, leaving behind a hole.

Thus, band-to-band tunneling results in the generation of electron-hole pairs that can take part in conduction and can therefore contribute to the drain current in the device.

To accurately study band-to-band tunneling, the simulation grid must be fine enough to resolve the high fields in the region where band-to-band tunneling occurs.

After reading in the simulation structure created by the input file *mdex1* in Example 1, three additional regrids are performed to refine the mesh for this analysis. Each regrid is based on the band-to-band tunneling generation rate (**BB.GENER**) and is performed after a solution is obtained with V_d=10V and V_g=-4V, a bias conducive to band-to-band tunneling.

The resulting simulation mesh is shown in Figure 4-25.

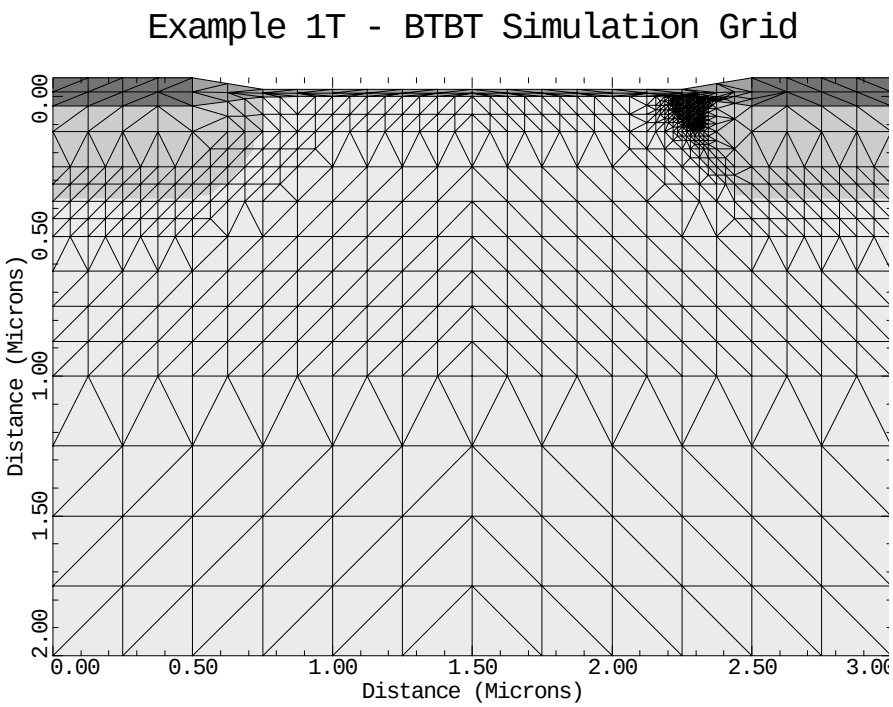


Figure 4-25 BTBT simulation grid from **PLOT.2D** at line 21 in file *mdex1t,* Figure 4-24

Models

To include band-to-band tunneling self-consistently in the solution of the device equations, the parameter **BTBT** is specified on the **MODELS** statement. The other model parameters that were specified previously in the input file *mdex1* shown in Figures 4-1 and 4-2 are automatically included in this simulation as well. This is because all the model parameters specified in *mdex1* were automatically stored in the solution file *MDEX1S,* which was read in at the beginning of this simulation.

Solution

Since band-to-band tunneling is a phenomenon involving both electrons and holes, 2-carrier solutions are required for this analysis.

The most efficient way to apply an initial bias to the device is by performing a 0-carrier solution that is used as the initial guess for the full 2-carrier solution. An initial 0-carrier solution with V_{Drain}=10V is created at line 26 of the input file *mdex1t* shown in Figure 4-24. The full 2-carrier solutions are generated by the **SOLVE** statement at line 31 where the gate bias is swept from 0V to -4V.

Drain Current vs. Vgs Plot

The results of this analysis are shown in Figure 4-26 where the logarithm of drain current is plotted as a function of the applied gate bias. This figure clearly shows the increase in drain current due to band-to-band tunneling generated electrons as the gate bias is decreased to more negative values.

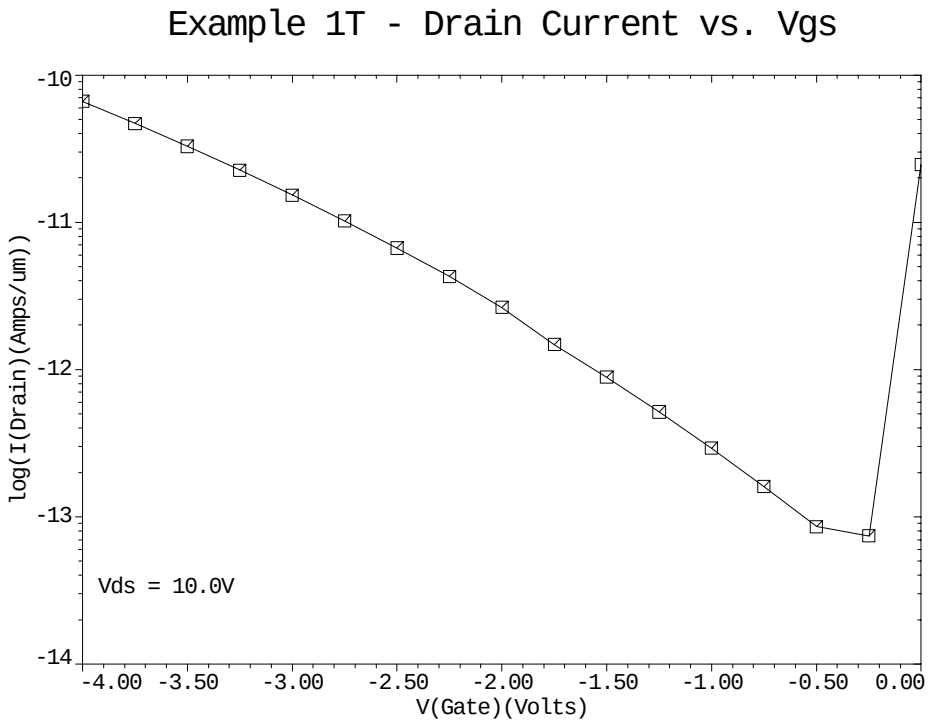


Figure 4-26 Drain current vs. Vgs from **PLOT.1D** and **LABEL** at lines 33 and 34 in file *mdex1t,* Figure 4-24

Band-to-Band Generation Contour Plot

Figure 4-27 plots contours of band-to-band tunneling generation rate occurring within the device. As the figure indicates, most of the band-to-band tunneling occurs at the surface in the high field region where the gate overlaps the drain.

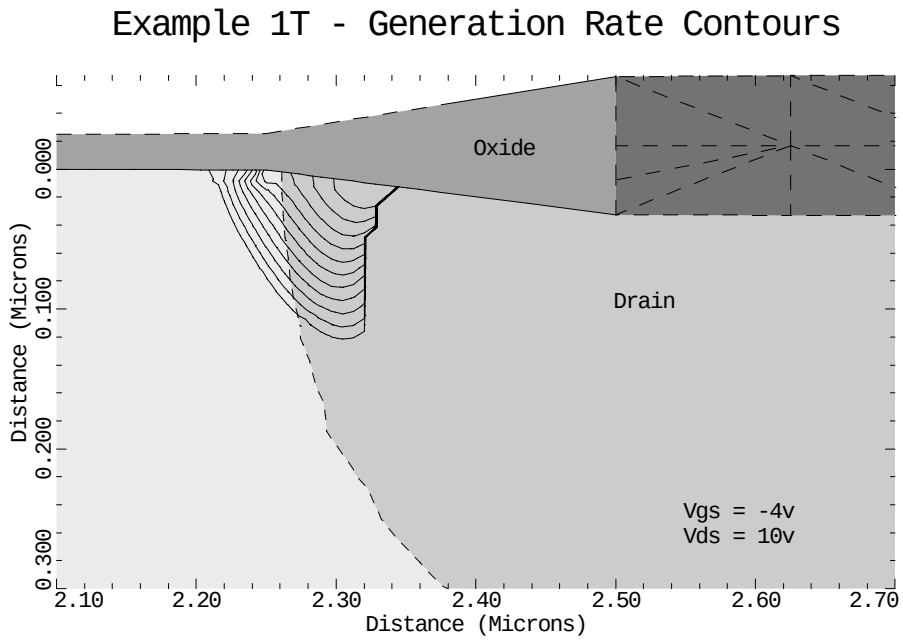


Figure 4-27 Generation rate contours from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 36 through 41 in file *mdex1t,* Figure 4-24

Avalanche Breakdown Analysis

An avalanche breakdown analysis of the same n-channel MOS device considered in the previous examples is performed using the input file *mdex1a*.

For this analysis, the maximum electron and hole ionization integrals are calculated for a gate bias of 0V and drain biases of 6V to 12V.

Figures 4-28 through 4-33 contain the output associated with the execution of Medici for the input file *mdex1a*.

Procedures

The simulation of avalanche breakdown includes the following procedures.

Modifying Simulation Structure

The structure created with the input file *mdex1a* shown in Figures 4-28 and 4-29 is the same as the structure created by the input file *mdex1*, but includes the modifications discussed below. "Generation of the Simulation Structure" on page 4-2 contains a complete discussion of the input file *mdex1*. Only the changes to *mdex1* necessary to create the modified structure are discussed here.

Because of the increased drain bias that is applied to the device, the simulation structure created by the input file *mdex1* shown in Figures 4-1 and 4-2 needs to be modified in two ways.

1. The depth of the structure needs to be increased to accommodate the increased depletion region size associated with the increased drain bias.

   The necessary depth can be estimated by assuming that the depletion region size increases as the square root of the applied bias.

2. The drain region needs to be widened to insure that the electric fields in this critical region are accurately represented. This is done in one of two ways.

   a. The drain width used in the simulation should be one half the actual device drain width. or,

   b. In the case of a very wide drain, the drain width should be approximately equal to the depth of the depletion region.

To accommodate drain biases as high as 20V-30V, the depth of the structure is increased from 2 microns to 4 microns. This is accomplished by specifying one additional **Y.MESH** statement shown in line 11 of Figure 4-28.

```
 1... TITLE     Avant! MEDICI Example 1A - 1.5 Micron N-Channel MOSFET
 2... COMMENT   Avalanche Breakdown Analysis

 3... COMMENT   Initial mesh specification
 4... MESH      SMOOTH=1
 5... X.MESH    WIDTH=3.0  H1=0.125
 6... X.MESH    WIDTH=1.5  H1=0.125  H2=.3

 7... Y.MESH    N=1   L=-0.025
 8... Y.MESH    N=3   L=0.
 9... Y.MESH    DEPTH=1.0  H1=0.125
10... Y.MESH    DEPTH=1.0  H1=0.250
11... Y.MESH    DEPTH=2.0  H1=0.250  RATIO=1.4

12... COMMENT   Eliminate nodes, increase source/drain oxide thickness
13... ELIMIN    COLUMNS  Y.MIN=1.1
14... SPREAD    LEFT   WIDTH=0.625  UP=1  LO=3  THICK=.1  ENC=2
15... SPREAD    RIGHT  WIDTH=2.125  UP=1  LO=3  THICK=.1  ENC=2
16... SPREAD    LEFT   WIDTH=100    UP=3  LO=4  Y.LO=.125

17... COMMENT   Specify oxide and silicon regions
18... REGION    SILICON
19... REGION    OXIDE    IY.MAX=3

20... COMMENT   Electrode definition
21... ELECTR    NAME=Gate    X.MIN=0.625  X.MAX=2.375  TOP
22... ELECTR    NAME=Substrate  BOTTOM
23... ELECTR    NAME=Source  X.MAX=0.5  IY.MAX=3
24... ELECTR    NAME=Drain   X.MIN=2.5  IY.MAX=3

25... COMMENT   Specify impurity profiles and fixed charge
26... PROFILE   P-TYPE  N.PEAK=3E15  UNIFORM     OUT.FILE=MDEX1DS
27... PROFILE   P-TYPE  N.PEAK=2E16  Y.CHAR=.25
28... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=0.0  WIDTH=0.5
  ... +         XY.RAT=.75
29... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=2.5  WIDTH=2.0
  ... +         XY.RAT=.75
30... INTERFAC  QF=1E10

31... COMMENT   Regrid on doping
32... REGRID    DOPING  LOG  IGNORE=OXIDE  RATIO=2  SMOOTH=1
  ... +         IN.FILE=MDEX1DS

33... COMMENT   Specify contact parameters
34... CONTACT   NAME=Gate  N.POLY

35... COMMENT   Specify physical models to use
36... MODELS    CONMOB  FLDMOB  SRFMOB2
37... COMMENT   Initial solution, regrid on potential
38... SYMB      CARRIERS=0
39... METHOD    ICCG  DAMPED
40... SOLVE
41... REGRID    POTEN  IGNORE=OXIDE  RATIO=.2  MAX=1  SMOOTH=1
  ... +         IN.FILE=MDEX1DS
42... PLOT.2D   GRID  TITLE="Example 1A - Avalanche Simulation Mesh"
  ... +         FILL  SCALE

43... COMMENT   Solve using the refined grid, Vds=4
44... SYMB      CARRIERS=0
45... SOLVE     V(Drain)=4  LOCAL

46... COMMENT   Calculate ionization integrals for Vds=6,8,10,12
47... LOOP      STEPS=4
48...   ASSIGN    NAME=BIAS  N.VALUE=6  DELTA=2
49...   SOLVE     V(Drain)=@BIAS
50...   EXTRACT   IONIZATI  X.MIN=2.0
51... L.END
```

Figure 4-28 First part of the simulation input file *mdex1a*

```
52... COMMENT   Potential contours and electric field lines for most
53... $         recent solution
54... PLOT.2D   BOUND  JUNC  DEPL  FILL  SCALE
  ... +         TITLE="Example 1A - Poten. Contours & E-Lines"
55... CONTOUR   POTENTIA  MIN=-1  MAX=13  DEL=2  COLOR=6
56... E.LINE    X.START=2.3  Y.START=0.02  S.DELTA=-.4  N.LINES=8
  ... +         E.MARK=1.75E5  I.HOLE  I.ELEC  LINE.TYP=3  COLOR=1
57... LABEL     LABEL="Vgs =  0.0v"  X=0.3  Y=3.3
58... LABEL     LABEL="Vds = 12.0v"
```

Figure 4-29 Second part of the simulation input file *mdex1a*

Additional Structure Modification

The following additional changes need to be made to the structure:

- To minimize the number of additional nodes added to the structure, the **RATIO** parameter is used to increase the size of each successive grid spacing.
- The width of the simulation structure is increased by 1.5 microns on the drain side by specifying one additional **X.MESH** statement shown in line 6.
- Change the **WIDTH** parameter on the **PROFILE** statement for the drain from 0.5 microns to 2.0 microns.

Figure 4-30 shows the resulting simulation mesh after regrids on both impurity concentration and potential have been performed.

Solutions

Because this analysis is performing with the device turned off (V_{Gate}=0V), it is sufficient to perform a Poisson-only (0-carrier) solution for each drain bias.

- In line 45 of Figure 4-28, the drain is initially biased to 4V.
- An input statement loop, as shown in lines 47 to 51, is used to ramp the drain from 6V to 12V in 2V increments.

Ionization Integrals

The **IONIZATI** parameter on the **EXTRACT** statement is used to request the calculation of both electron and hole ionization integrals after each solution. The program automatically finds the maximum ionization integrals by calculating the integrals along potential gradient paths starting at each node in the device.

Because this example focuses on the breakdown of the drain, the starting nodes are limited to the drain region to reduce computation time by using the **X.MIN** parameter. The gradient paths, however, are allowed to extend beyond this limit.

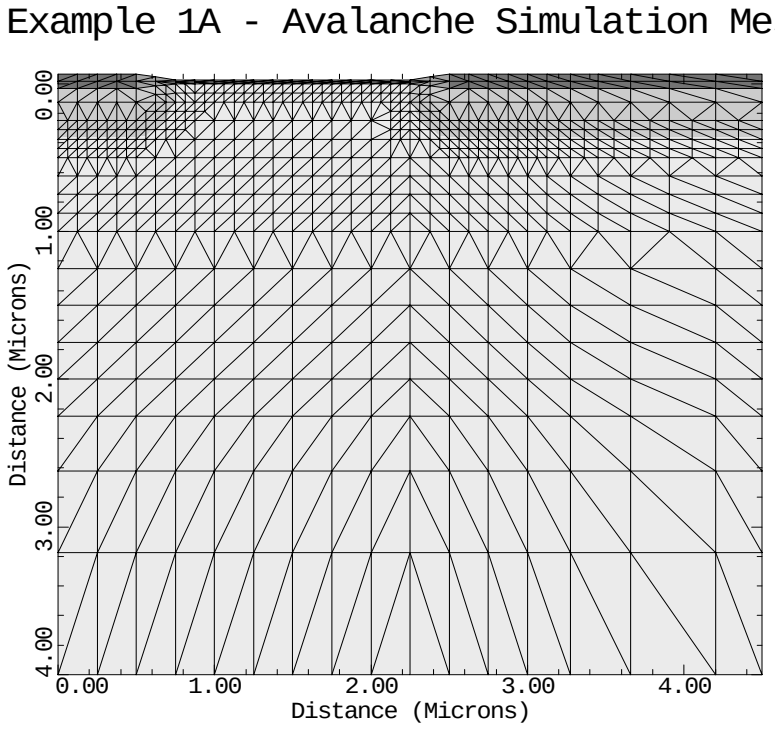


Figure 4-30 Avalanche simulation mesh from **PLOT.2D** at line 42 in file *mdex1a*, Figures 4-28 and 4-29

Figure 4-31 contains a portion of the output associated with the execution of Medici for the input file *mdex1a*. The output indicates that avalanche breakdown occurs at a drain bias of approximately 12V (ionization integrals > 1).

The output also indicates that the maximum ionization integrals are caused by fields located at the surface and near the left edge of the drain. The breakdown in this case is aided by the gate.

Potential Contours and E-Line Plots

Lines 54 and 55 use the **PLOT.2D** and **CONTOUR** statements. A two-dimensional plot of potential contours and electric field lines for a drain bias of 12V is shown in Figure 4-32. Potential contours are shown as solid lines and electric field lines are dashed. This example uses the following parameters and values:

- The **E.MARK** parameter on the **E.LINE** statement is used to place brackets on the field lines at the specified value of 1.75×10^5 V/cm.

  The starting point for the first field line is specified to be at the surface in the drain region.
- **S.DELTA** specifies that the starting point for a subsequent line occurs at a distance of 0.4 microns from the previous starting point on a line perpendicular to the previous field line.
- The sign of **S.DELTA** determines which direction is used to step to the next starting point.

In this case, a negative value indicates that the step is to be taken to the left of the field vector.

Values of the plotted potential contours and the electron and hole ionization integrals calculated for each field line are shown in Figure 4-33.

```
Solution for Bias:

  V(Gate)         = 0.0000000E+00
  V(Substrate)    = 0.0000000E+00
  V(Source)       = 0.0000000E+00
  V(Drain)        = 1.0000000E+01
.
.
.

  Electrode       Electron   Peak Field    X Location   Y Location
    Name         Ionization    (V/cm)      (microns)    (microns)
--------------  ----------  ----------    ----------   ----------
Substrate       0.6847      5.7214E+05       2.272     1.2536E-02
Drain           0.6847      5.7214E+05       2.272     1.2536E-02

  Electrode          Hole     Peak Field    X Location   Y Location
    Name         Ionization    (V/cm)      (microns)    (microns)
--------------  ----------  ----------    ----------   ----------
Substrate       0.5299      5.7214E+05       2.272     1.2536E-02
Drain           0.5299      5.7214E+05       2.272     1.2536E-02
.
.
.
Solution for Bias:

  V(Gate)         = 0.0000000E+00
  V(Substrate)    = 0.0000000E+00
  V(Source)       = 0.0000000E+00
  V(Drain)        = 1.2000000E+01
.
.
.

  Electrode       Electron   Peak Field    X Location   Y Location
    Name         Ionization    (V/cm)      (microns)    (microns)
--------------  ----------  ----------    ----------   ----------
Substrate       1.002       6.9988E+05       2.326     4.1325E-02
Drain           1.002       6.9988E+05       2.326     4.1325E-02

  Electrode          Hole     Peak Field    X Location   Y Location
    Name         Ionization    (V/cm)      (microns)    (microns)
--------------  ----------  ----------    ----------   ----------
Substrate       1.003       6.9988E+05       2.326     4.1325E-02
Drain           1.003       6.9988E+05       2.326     4.1325E-02
```

Figure 4-31 A portion of the ionization integral results from loop lines 47 through 51 in file *mdex1a*, Figures 4-28 and 4-29

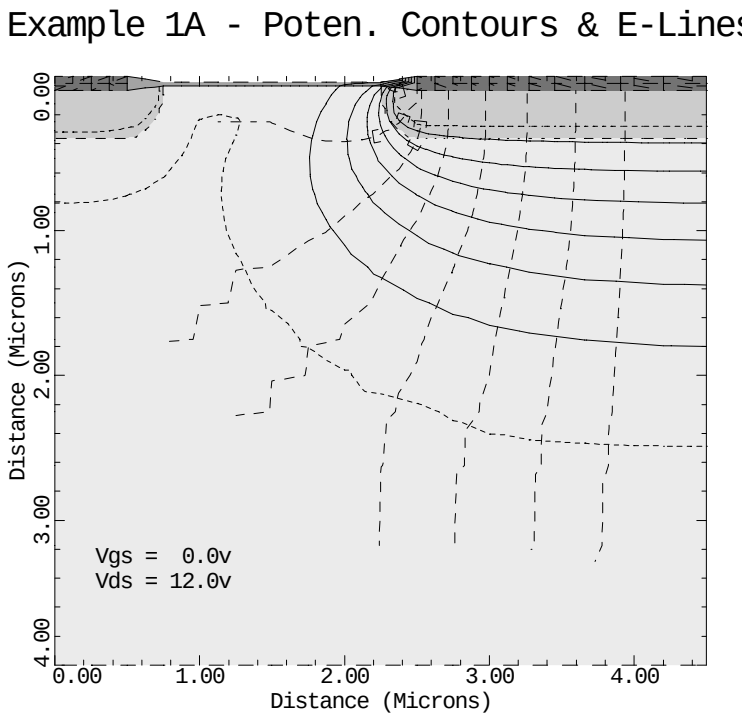


Figure 4-32 Potential contours and E-lines from **PLOT.2D**, **CONTOUR**, **E.LINE**, and **LOOP** at lines 54 through 58 in file *mdex1a*, Figures 4-28 and 4-29

```
Two dimensional contour plot
Input line #      55

  Values plotted:
   1.000000E+00    3.000000E+00    5.000000E+00
   7.000000E+00    9.000000E+00    1.100000E+01

  Values not plotted:
  -1.000000E+00    1.300000E+01



Carrier ionization integral information
Input line #      56

        Electron       Hole     Peak Field   X Location   Y Location
Line   Ionization   Ionization    (V/cm)      (microns)    (microns)
----  -----------  -----------  -----------  -----------  -----------
  1    0.8819       0.8295       6.9988E+05    2.329       3.4022E-02
  2    0.2196       5.2025E-02   3.7078E+05    2.360       0.2698
  3    4.4254E-02   2.4136E-03   2.3826E+05    2.521       0.2937
  4    6.0908E-03   6.9613E-05   1.6573E+05    2.710       0.3562
  5    1.4377E-03   5.8706E-06   1.3720E+05    2.971       0.3609
  6    5.8673E-04   1.3061E-06   1.2472E+05    3.258       0.3687
  7    3.3304E-04   5.1460E-07   1.1711E+05    3.594       0.3559
  8    2.4512E-04   3.1156E-07   1.1364E+05    3.935       0.3499

Maximum electron ionization :   0.8819
Maximum hole ionization:   0.8295
```

Figure 4-33 Plotted contour values and the calculated ionization integrals from **CONTOUR** and **E.LINE** at lines 55 through 56 in file *mdex1a*, Figures 4-28 and 4-29

Breakdown Simulation with Impact Ionization Model

The input file *mdex1b* shown in Figures 4-34 and 4-35 performs a breakdown simulation for the same n-channel MOS device considered in the previous examples.

For this simulation:

- Carriers generated by impact ionization are included self-consistently in the solution.
- Gate bias is set just above threshold at 0.5V and the drain bias is ramped in 0.5V increments beginning at 9.0V.
- When the structure nears breakdown, current boundary conditions are used instead of voltage boundary conditions.

  This allows direct simulation of the structure near and past the breakdown point.

Figures 4-34 through 4-37 contain the output associated with the execution of Medici for the input file *mdex1b*.

Procedures

The simulation of breakdown with impact ionization model uses the following procedures.

Structure

The structure created with the input file *mdex1b* is identical to the structure created by the input file *mdex1a*, which is discussed in "Avalanche Breakdown Analysis" on page 4-28.

Models

The impact ionization model is selected by specifying **`IMPACT.I`** on the **`MODELS`** statement at line 36. With this model selected, any carriers generated due to impact ionization are included in the solution.

Initial Solution

A zero-carrier solution is used to initially bias the device with 9.0V on the drain and 0.5V on the gate. Before performing solutions that include the current-continuity equations, a lumped substrate resistance is specified at line 46 to take into account the resistance of the structure not explicitly included in the simulation domain.

Newton's method with two carriers is required when performing solutions which include impact ionization. This is specified on the **SYMB** statement at line 48.

```
 1... TITLE     Avant! MEDICI Example 1B - 1.5 Micron N-Channel MOSFET
 2... COMMENT   Breakdown Simulation with Gate Drive

 3... COMMENT   Initial mesh specification
 4... MESH      SMOOTH=1
 5... X.MESH    WIDTH=3.0  H1=0.125
 6... X.MESH    WIDTH=1.5  H1=0.125  H2=.3

 7... Y.MESH    N=1   L=-0.025
 8... Y.MESH    N=3   L=0.
 9... Y.MESH    DEPTH=1.0  H1=0.125
10... Y.MESH    DEPTH=1.0  H1=0.250
11... Y.MESH    DEPTH=2.0  H1=0.250  RATIO=1.4

12... COMMENT   Eliminate nodes, increase source/drain oxide thickness
13... ELIMIN    COLUMNS  Y.MIN=1.1
14... SPREAD    LEFT   WIDTH=0.625  UP=1  LO=3  THICK=.1  ENC=2
15... SPREAD    RIGHT  WIDTH=2.125  UP=1  LO=3  THICK=.1  ENC=2
16... SPREAD    LEFT   WIDTH=100    UP=3  LO=4  Y.LO=.125

17... COMMENT   Specify oxide and silicon regions
18... REGION    SILICON
19... REGION    OXIDE    IY.MAX=3

20... COMMENT   Electrode definition
21... ELECTR    NAME=Gate    X.MIN=0.625  X.MAX=2.375  TOP
22... ELECTR    NAME=Substrate  BOTTOM
23... ELECTR    NAME=Source  X.MAX=0.5  IY.MAX=3
24... ELECTR    NAME=Drain   X.MIN=2.5  IY.MAX=3

25... COMMENT   Specify impurity profiles and fixed charge
26... PROFILE   P-TYPE  N.PEAK=3E15  UNIFORM     OUT.FILE=MDEX1DS
27... PROFILE   P-TYPE  N.PEAK=2E16  Y.CHAR=.25
28... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=0.0  WIDTH=0.5
  ... +         XY.RAT=.75
29... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=2.5  WIDTH=2.0
  ... +         XY.RAT=.75
30... INTERFAC  QF=1E10

31... COMMENT   Regrid on doping
32... REGRID    DOPING  LOG  IGNORE=OXIDE  RATIO=2  SMOOTH=1
  ... +         IN.FILE=MDEX1DS

33... COMMENT   Specify contact parameters
34... CONTACT   NAME=Gate  N.POLY

35... COMMENT   Specify physical models to use
36... MODELS    IMPACT.I  CONMOB  FLDMOB  SRFMOB2

37... COMMENT   Initial solution, regrid on potential
38... SYMB      CARRIERS=0
39... METHOD    ICCG  DAMPED
40... SOLVE
41... REGRID    POTEN  IGNORE=OXIDE  RATIO=.2  MAX=1  SMOOTH=1
  ... +         IN.FILE=MDEX1DS  OUT.FILE=MDE1BMS

42... COMMENT   Perform a 0-carrier solution at the initial bias
43... SYMB      CARRIERS=0
44... SOLVE     V(Gate)=0.5  V(Drain)=9.0  LOCAL
45... COMMENT   Specify a lumped substrate resistance
46... CONTACT   NAME=Substrate  RESIST=2E5

47... COMMENT   Obtain solutions using 2-carrier Newton with continuation
48... SYMB      CARRIERS=2  NEWTON
49... LOG       OUT.FILE=MDEX1BI
50... SOLVE     ELEC=Drain  CONTINU  C.VMAX=13  C.IMAX=0.8E-4
  ... +         C.VSTEP=0.55  C.TOLER=0.10

51... COMMENT   Drain current vs. drain voltage
```

Figure 4-34 First part of the simulation input file *mdex1b*

```
52... PLOT.1D   X.AXIS=V(Drain)  Y.AXIS=I(Drain)  POINTS  COLOR=2  ^ORDER
  ... +         LEFT=8.0  RIGHT=11.5  BOT=0.0  TOP=1.0E-4
  ... +         TITLE="Example 1B - Drain Current Snapback"
53... LABEL     LABEL="Vgs = 0.5v"  X=10.7  Y=0.9E-4

54... COMMENT   Flowlines for last solution
55... PLOT.2D   BOUND  JUNC  DEPL  TITLE="Example 1B - 5% Flowlines"
  ... +         FILL  SCALE
56... CONTOUR   FLOWLINES  NCONT=21  COLOR=1
57... LABEL     LABEL="Vgs = 0.5v"  X=3.0  Y=3.25
58... LABEL      LABEL="Id  = 8e-5 A/um"
```

Figure 4-35 Second part of simulation input file *mdex1b*

Continuation Method

When performing a breakdown simulation, it may not be known in advance at what voltage the device breaks down. In such a case it is possible to step the bias of the electrode in question (in this example the drain) from a value that is expected to be below the breakdown voltage to one that is expected to be higher.

As breakdown is approached, the current increases and if a simple voltage boundary condition is used, the program stops converging once the device snaps back.

The continuation method can be used to overcome convergence problems. The continuation method automatically selects bias points and switches smoothly from a voltage boundary condition to a current boundary condition as the slope of the IV characteristic increases. As the device snaps back, the program takes negative voltage steps and continues to follow the curve until one of the termination criteria is reached.

The continuation method is enabled by the following:

- Specify **CONTINU** on the **SOLVE** statement (see line 50).
- Specify the electrode to be stepped (**ELEC**=Drain), the initial voltage step size (**C.VSTEP**=0.55), and the terminating voltage and current (**C.VMAX=**13, **C.IMAX=**0.8e-4).

  If the terminal voltage or current reaches either of these termination criteria, the simulation stops.
- An optional tolerance (**C.TOL**=0.10) may be specified to control the spacing between the bias points.

  A smaller **C.TOL** gives more closely spaced bias points, but takes longer to run.

In this particular example the breakdown voltage was anticipated to be about 11V. **C.VMAX** was set to a value greater than 11V (13V in this case) so the device breaks down. Likewise, **C.IMAX** is simply set to a value greater than the anticipated snap-back current.

Drain Current Snapback Plot

Figure 4-36 shows the resulting drain current versus drain voltage curve obtained from this simulation. The device breaks down at approximately 11V, but as seen in the figure, the curve snaps back on itself, indicating that the breakdown can be sustained at a lower drain voltage.

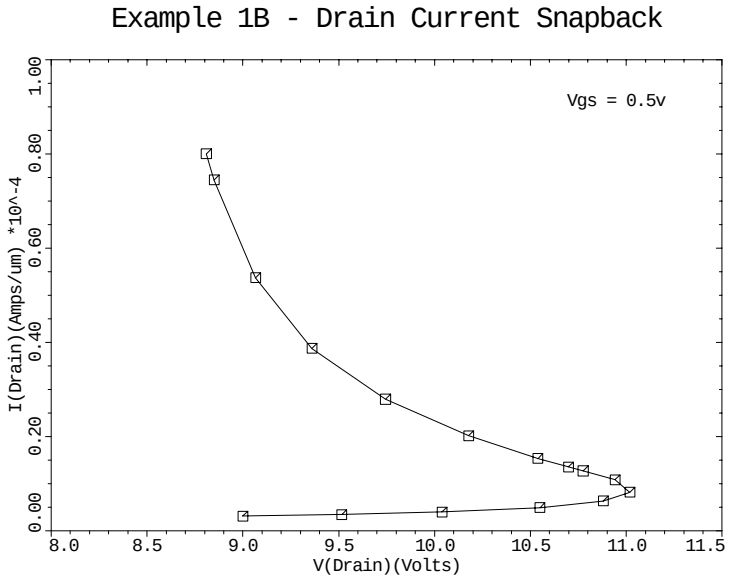


Figure 4-36 Drain current snapback from **PLOT.1D** and **LABEL** at lines 52 and 53 in file *mdex1b,* Figures 4-34 and 4-35

Flowline Plot

A two-dimensional plot of total current flowlines for the last solution (corresponding to a drain bias of approximately 8.4V and a drain current of 8×10^{-5} A/μm) is shown in Figure 4-37. The flowlines are such that 5% of the total current flows between each pair of lines. As can be seen, in addition to the channel current, much of the current flow from source to drain is now in the bulk.

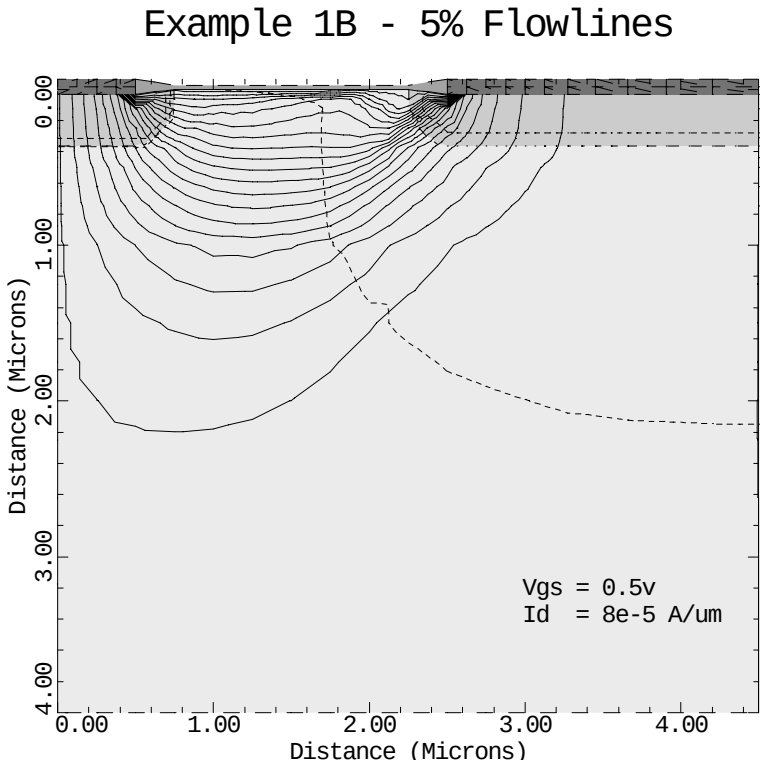


Figure 4-37 Five percent flowlines from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 55 through 58 in file *mdex1b,* Figures 4-34 and 4-35

Direct Tunneling Analysis in a N-MOSCAP

The input file *mdex1dt* shown in Figures 4-38 and 4-39 illustrates how to use the direct tunneling model to calculate the gate leakage current in a N-MOSCAP. The three different methods of evaluating the direct tunneling current are compared in post-processing mode and illustrate the importance of direct tunneling at low voltages for thin gate oxides. A self-consistent calculation is also performed.

Mesh

Line 3 defines the thickness of the gate oxide using the variable TOX to be 15Å. Defining the gate oxide thickness using an **ASSIGN** statement in this way allows the thickness to be easily changed. Using this thickness, the mesh is constructed in lines 5 through 10. A rather fine mesh in the y-direction is constructed, while only two mesh spacings are used in the x-direction.

```
 1... Title Avant! Medici Example 1DT- Direct Tunneling in NMOSCAP

 2... $ Oxide thickness: 15 Angstroms
 3... ASSIGN NAME=TOX N.VAL=15E-4

 4... $ Create mesh
 5... MESH
 6... X.MESH WIDTH=1 N.SPACES=2
 7... Y.MESH NODE=1 LOCATION=-50E-3
 8... Y.MESH WIDTH=50E-3 H1=5E-3 H2=1E-4
 9... Y.MESH WIDTH=@TOX H1=1E-4
10... Y.MESH WIDTH=200E-3 H1=1E-4 RATIO=1.1

11... $ Define regions
12... REGION SILICON NAME=SILICON2
13... REGION OXIDE   NAME=OXIDE1   Y.MAX=@TOX
14... REGION SILICON NAME=SILICON1 Y.MAX=0.0

15... $ Define electrodes. Turn top Si region into electrode.
16... ELECTRODE NAME=BOTTOM BOTTOM
17... ELECTRODE NAME=GATE REGION=SILICON1

18... $ Uniform P-type doping profile
19... PROFILE P-TYPE UNIFORM N.PEAK=2.E17 REGION=SILICON2

20... $ Set parameters for direct tunneling
21... MATERIAL OXIDE   AFFINITY=0.95 ME.DT=.35
22... MATERIAL SILICON AFFINITY=4.10 ME.DT=.19
23... MATERIAL ELECTRODE=GATE        ME.DT=.32
24... CONTACT NAME=GATE ALUMI

25... $ Use Fermi-Dirac statistics
26... MODELS FERMI.DIR

27... $ Only solve Poisson
28... SYMBOLIC CARRIERS=0 NEWTON

29... $ Print material parameters
30... MATERIAL PRINT

31... $ Post-process DT model 1 (Analytical)
32... LOG OUT.FILE=DT1.LOG
33... SOLVE V(GATE)=0.0 DT.CURR DT.METH=1 ELEC=GATE VSTEP=.25
  ... +     NSTEPS=20
34... LOG CLOSE

35... $ Post-process DT model 2 (WKB)
36... SOLVE V(GATE)=0.0 INIT
37... LOG OUT.FILE=DT2.LOG
38... SOLVE V(GATE)=0.0 DT.CURR DT.METH=2 ELEC=GATE VSTEP=.025
  ... +     NSTEPS=10
39... SOLVE DT.CURR DT.METH=2 ELEC=GATE VSTEP=.25 NSTEPS=19
40... LOG CLOSE
```

Figure 4-38 First part of input file *mdex1dt*

```
41... $ Post-process DT model 3 (Airy)
42... SOLVE V(GATE)=0.0 INIT
43... LOG OUT.FILE=DT3.LOG
44... SOLVE V(GATE)=0.0 DT.CURR DT.METH=3 ELEC=GATE VSTEP=.025
  ... +     NSTEPS=10
45... SOLVE DT.CURR DT.METH=3 ELEC=GATE VSTEP=.25 NSTEPS=19
46... LOG CLOSE

47... $ Self-consistent DT model 3 (Airy)
48... MODELS SRH AUGER DT.CUR DT.METH=3
49... MATERIAL SILICON TAUN0=1.E-9 TAUP0=1.E-9
50... SOLVE V(GATE)=0.0 INIT
51... SYMBOLIC CARRIERS=1
52... LOG OUT.FILE=DT4.LOG
53... SOLVE V(GATE)=0.0 ELEC=GATE VSTEP=.025 NSTEPS=10
54... SOLVE             ELEC=GATE VSTEP=.25  NSTEPS=19
55... LOG CLOSE


56... $ Make plots of tunneling current
 57... PLOT.1D LEFT=0 BOTTOM=1E-15 TITLE="Direct Tunneling Current"
  ... +       IN.FILE=DT1.LOG X.AXIS=V(GATE) Y.AXIS=FE(GATE) ABS
  ... +       LOG COLOR=2
58... PLOT.1D IN.FILE=DT2.LOG X.AXIS=V(GATE) Y.AXIS=FE(GATE) ABS
  ... +       LOG COLOR=3 UNCH
59... PLOT.1D IN.FILE=DT3.LOG X.AXIS=V(GATE) Y.AXIS=FE(GATE) ABS
  ... +       LOG COLOR=4 UNCH
60... PLOT.1D IN.FILE=DT4.LOG X.AXIS=V(GATE) Y.AXIS=I(GATE)  ABS
  ... +       LOG COLOR=1 UNCH

61... $ Plot labels
62... LABEL LABEL="DT1"    X=0.1 Y=1E-5  COLOR=2
63... LABEL LABEL="DT2"    X=1.0 Y=2E-8  COLOR=3
64... LABEL LABEL="DT3"    X=4.0 Y=5E-5  COLOR=4
65... LABEL LABEL="SC-DT3" X=4.0 Y=5E-14 COLOR=1
```

Figure 4-39 Second part of input file *mdex1dt*

Regions

In Figure 4-38, lines 12 through 14 define the three regions in the MOSCAP:

- Substrate is named **SILICON2**
- Gate oxide is named **OXIDE1**
- Gate is named **SILICON1**

The electrodes are then defined in lines 16 and 17. A flat electrode is defined at the back of the substrate, while the top silicon region, **SILICON1**, is turned into a region electrode. The device structure is constructed in this way to make it easy to change the gate from an electrode to a polysilicon gate. In line 19, an uniform p-type doping profile is specified for the substrate.

Tunneling Parameters

The parameters related to the direct tunneling model are specified in lines 21-24. The electron affinity and electron effective tunneling mass for the oxide and silicon regions are specified in lines 21 and 22, respectively. The electron effective tunneling mass for the gate electrode is specified in line 23. The workfunction of the gate is set to that of aluminum in line 24.

Models

Line 26 specifies that Fermi-Dirac statistics are to be used. Line 28 specifies that only Poisson's equation is to be solved. Line 30 specifies that the material parameters in all the regions should be printed to the output file.

Direct Tunneling Analysis

Lines 31 through 46 perform a post-processing analysis of the direct tunneling current for a positive voltage ramp on the gate using the three different methods of evaluating the direct tunneling model. The tunneling current calculated by each method during the voltage ramp is stored in a separate log file. Line 33 specifies that the analytical direct tunneling model should be used (**DT.METH=1**), lines 38 and 39 specify that numerical integration of the WKB tunneling coefficient should be used to evaluate the tunneling current (**DT.METH=2**), and lines 44 and 45 specify that numerical integration of the Gundlach tunneling coefficient should be used to evaluate the tunneling current (**DT.METH=3**). Lines 46-55 perform a self-consistent calculation of the direct tunneling using the Gundlach method. This method is activated as a model on line 48 along with the **SRH** and **AUGER** models. By default only the CBET contribution to direct tunneling is calculated. The SRH lifetimes are set in line 49. A one carrier solution using electrons is specified in line 51. The plot produced by lines 56 through 65 in Figure 4-39 is shown in Figure 4-40 and shows a comparison of the three evaluation methods in post-process mode. These results would be valid for a MOSCAP that has access to a ready supply of electrons such as in a MOSFET or under illumination. At low bias voltage, methods 2 and 3 agree fairly well with each other while method 1, the analytical method, shows significant discrepancy. This is not surprising since the analytical model is only valid for moderate to large voltage drops across the gate oxide. At high bias voltage, methods 1 and 2 converge to the conventional Fowler-Nordheim current while method 3 begins to show signs of oscillations in the Fowler-Nordheim regime as a result of a more accurate analysis. The self-consistent result using the Gundlach method is also shown. The current saturates at a low value that is limited by the carrier generation rate in the substrate.

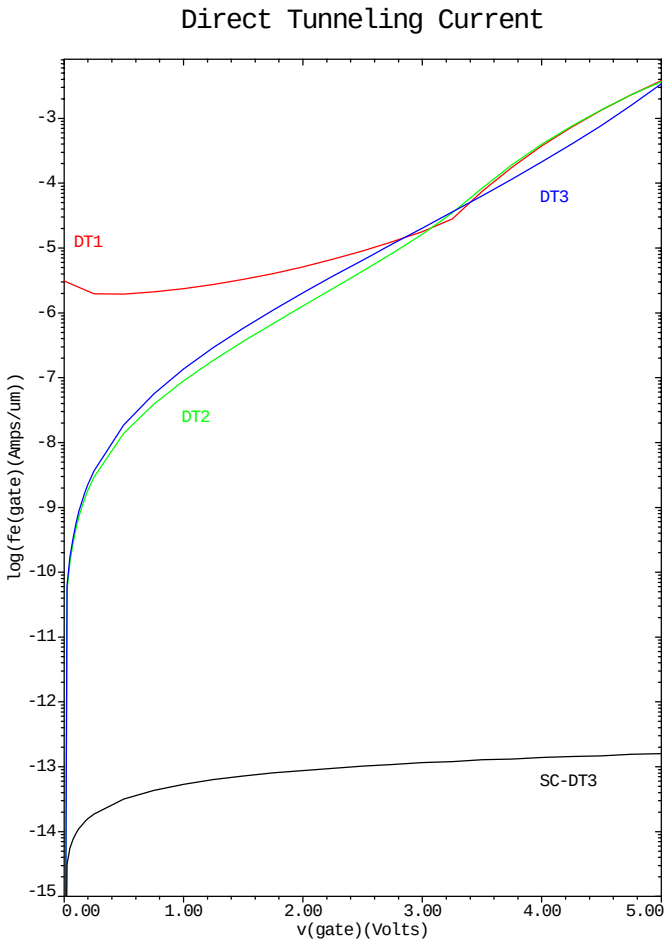


Figure 4-40 Plot produced by lines 56 through 65 in input file *mdex1dt*. DT1, DT2, and DT3 indicate the three different methods used for evaluating the direct tunneling model in post-processing mode. SC-DT3 is the result of a self-consistent calculation of direct tunneling using evaluation method 3.

Quadtree Mesh for a MOSFET

The input file *mdex1qt* shown in Figure 4-41 illustrates how to use the quadtree mesh option to regrid a MOSFET. Some of the main features of the quadtree mesh generator are highlighted.

Initial Quadtree Mesh

Lines 5 and 6 issue statements to construct the initial quadtree mesh. The **MESH** statement at line 5 reads in the original structure from a TIF file and initiates the construction of a quadtree mesh via the specification of the **QUADTREE** parame-

ter. The **MIN.THIC** parameter is used to specify the thickness of the thinnest layer of the device, in this case the thickness of the gate oxide. An immediate doping regrid is performed at line 6. Unless a doping file is used, it is a good idea to always do an immediate regrid on the doping in order to accurately capture the doping profile. The regrid is performed on the **ASINH** of the doping in order to get a nicely graded mesh. The **MINDELTA** parameter is used to specify a minimum allowed mesh spacing during the regrid. During a regrid on a quantity, **MINDELTA** should always be specified in order to limit the amount of regridding. Finally, the **COS.ANGL** parameter is set to 0.0 (90^o) in order to suppress the creation of obtuse elements in the silicon substrate. Figure 4-42 shows the resulting quadtree mesh that is plotted at line 8. The mesh is saved to a TIF file at line 9.

```
 1... TITLE    Avant! Medici Example 1QT - Quadtree Mesh

 2... COMMENT  Read in a TIF file and create a quadtree mesh.
 3... COMMENT  Do an immediate doping regrid to accurately capture
 4... COMMENT  the doping profile while suppressing obtuse triangles.
 5... MESH     IN.FILE=mdex1qt.tif TIF QUADTREE MIN.THIC=65.0E-4
 6... REGRID   DOPING RATIO=1.0 MINDELTA=100E-4 ASINH COS.ANGL=0.0

 7... COMMENT  Plot and save initial quadtree mesh.
 8... PLOT.2D  GRID FILL SCALE TITLE="Initial Quadtree Mesh"
 9... SAVE     OUT.FILE=quadtree0.tif TIF

10... COMMENT  Do a potential solve at zero bias.
11... SYMB     CARRIERS=0
12... SOLVE

13... COMMENT  Regrid in gate oxide. Then immediately regrid on the
  ... +        potential
14... COMMENT  in the substrate while suppressing obtuse triangles.
15... REGRID   REGION=oxide1 Y.MIN=-.025 MAXDEL.X=.015
16... REGRID   POTENTIAL REGION=silicon1 RATIO=0.8 MINDELTA=100E-4
  ... +        COS.ANGL=0.0

17... COMMENT  Plot the regridded mesh.
18... PLOT.2D  GRID FILL SCALE TITLE="Potential Regrid"

19... COMMENT  Do a solve with both carriers.
20... SYMB     CARRIERS=2
21... SOLVE    V(substrate)=0 V(source)=0 V(drain)=0.05 V(gate)=1.0

22... COMMENT  Save the mesh and solution in a TIF file.
23... SAVE     OUT.FILE=quadtree1.tif TIF ALL
```

Figure 4-41 List of input file *mdex1qt*

Potential Regrid

In Figure 4-41, line 11 specifies that a Poisson-only solution is to be performed in line 12. Two sequential regrids are performed in lines 15 and 16. The first regrid on line 15 is used to do an anisotropic regrid along the x direction in the gate oxide. The regrid is restricted to the oxide region by specifying the name of the oxide region, **oxide1**, using the **REGION** parameter. The **Y.MIN** parameter is used to restrict the regrid to the section of the oxide near the bottom of the gate, while the **MAXDEL.X** parameter specifies the desired mesh spacing in the x direc-

tion. A potential regrid is then performed on line 16. This regrid is restricted to the silicon region by again using the **REGION** parameter. As should always be done when regridding on a quantity, the **MINDELTA** parameter specifies the minimum allowed mesh spacing. Obtuse elements are suppressed by specifying a maximum desired element angle of 90^{o} using **COS.ANGL**=0.0. The resulting mesh is plotted in line 18 and is shown in Figure 4-43.

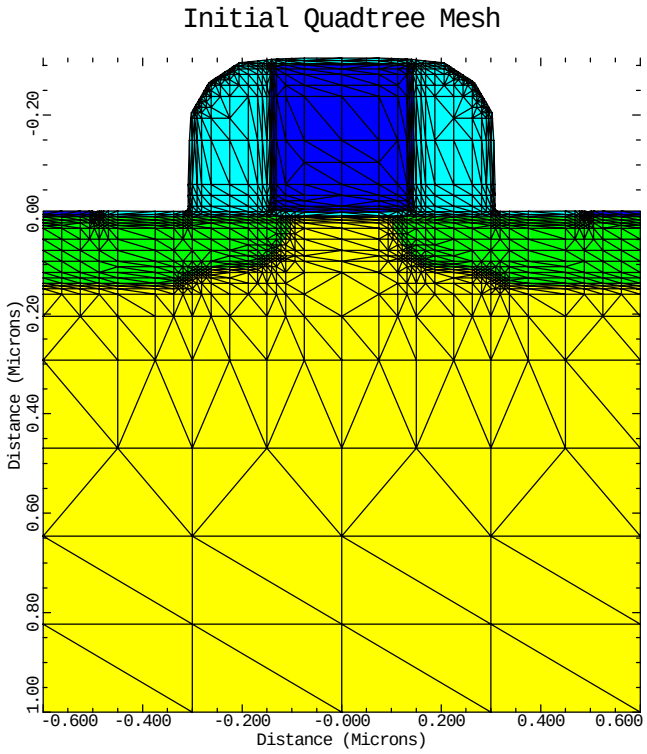


Figure 4-42 Plot produced by line 8 in input file *mdex1qt*. Initial mesh created using the quadtree mesher

Two-Carrier Solve

Line 20 specifies that a two-carrier solution should be performed, and line 21 performs the solution at a drain bias of 50mV and a gate bias of 1V. The mesh and solution are then saved to a TIF file in line 23.

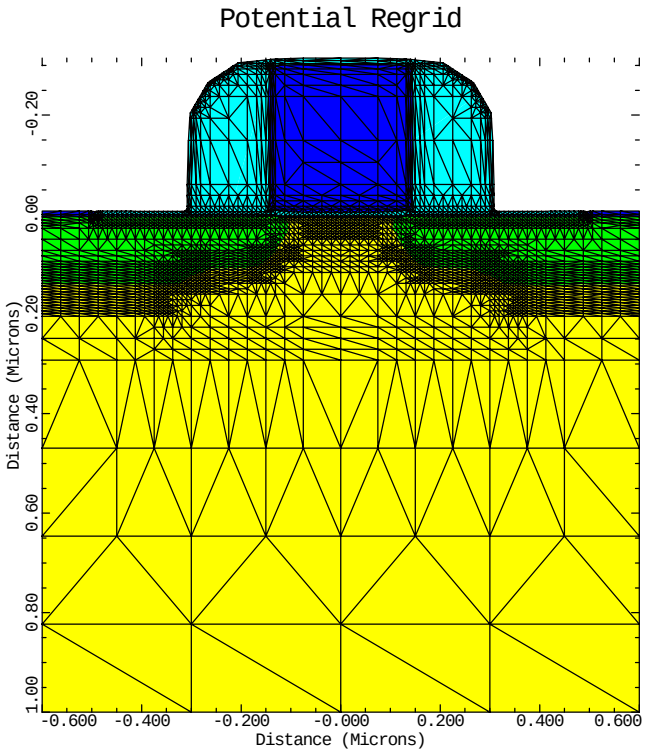


Figure 4-43 Plot produced by line 18 in input file *mdex1qt*. Quadtree mesh after a potential regrid

NPN Bipolar Transistor Examples

Example Specifications

The use of Medici is illustrated by going through some of the analysis that might be performed on an NPN bipolar transistor. The analysis is divided into six parts.

- The input file *mdex2* develops the simulation structure.
- The input file *mdex2f* then simulates the forward current characteristics for the device.

  The results of this simulation are examined with the input file *mdex2fp*.
- The input file *mdex2p* modifies the emitter region of the device and specifies different material properties for the modified region.
- The forward current characteristics are then repeated for the modified device.
- The results of the simulation with the modified emitter are examined with the input file *mdex2pp*.
- The input file *mdex2m* illustrates a one-dimensional analysis of a bipolar transistor.

Generation of the Simulation Structure

The input file *mdex2* creates the simulation structure for an NPN bipolar device. The output associated with the execution of Medici for the input file *mdex2* is shown in Figures 5-1 through 5-5.

Defining the Initial Mesh

As with the MOS example in Chapter 4, the first step in creating a device structure is to generate an initial mesh. Since this initial mesh will be refined, it needs to be

adequate for defining the structure, but does not need to be fine enough to perform a solution on.

The mesh generation is initiated with the **MESH** statement at line 4 of the input file shown in Figure 5-1.

```
 1... TITLE     Avant! MEDICI Example 2 - NPN Transistor Simulation
 2... COMMENT   Grid Generation and Initial Biasing

 3... COMMENT   Specify a rectangular mesh
 4... MESH
 5... X.MESH    WIDTH=6.0  H1=0.250
 6... Y.MESH    DEPTH=0.5  H1=0.125
 7... Y.MESH    DEPTH=1.5  H1=0.125  H2=0.4

 8... COMMENT   Region definition
 9... REGION    NAME=Silicon  SILICON

10... COMMENT   Electrodes
11... ELECTR    NAME=Base  X.MIN=1.25  X.MAX=2.00  TOP
12... ELECTR    NAME=Emitter  X.MIN=2.75  X.MAX=4.25  TOP
13... ELECTR    NAME=Collector  BOTTOM

14... COMMENT   Specify impurity profiles
15... PROFILE   N-TYPE  N.PEAK=5e15  UNIFORM     OUT.FILE=MDEX2DS
16... PROFILE   P-TYPE  N.PEAK=6e17  Y.MIN=.35  Y.CHAR=.16
   ... +                X.MIN=1.25  WIDTH=3.5   XY.RAT=.75
17... PROFILE   P-TYPE  N.PEAK=4e18  Y.MIN=0    Y.CHAR=.16
   ... +                X.MIN=1.25  WIDTH=3.5   XY.RAT=.75
18... PROFILE   N-TYPE  N.PEAK=7e19  Y.MIN=0    Y.CHAR=.17
   ... +                X.MIN=2.75  WIDTH=1.5   XY.RAT=.75
19... PROFILE   N-TYPE  N.PEAK=1e19  Y.MIN=2    Y.CHAR=.27

20... PLOT.2D   GRID  TITLE="Example 2 - Initial Grid"  SCALE  FILL

21... COMMENT   Regrid on doping
22... REGRID    DOPING  LOG  RATIO=3  SMOOTH=1  IN.FILE=MDEX2DS
23... PLOT.2D   GRID  TITLE="Example 2 - 1st Doping Regrid" SCALE FILL

24... REGRID    DOPING  LOG  RATIO=3  SMOOTH=1  IN.FILE=MDEX2DS
25... PLOT.2D   GRID  TITLE="Example 2 - 2nd Doping Regrid" SCALE FILL

26... COMMENT   Extra regrid in emitter-base junction region only.
27... REGRID    DOPING  LOG  RATIO=3  SMOOTH=1  IN.FILE=MDEX2DS
   ... +        X.MIN=2.25  X.MAX=4.75  Y.MAX=0.50  OUT.FILE=MDEX2MS
28... PLOT.2D   GRID  TITLE="Example 2 - 3rd Doping Regrid" SCALE FILL

29... COMMENT   Define models
30... MODELS    CONMOB  CONSRH  AUGER  BGN

31... COMMENT   Solve for Vce=3 volts
32... SYMB      CARRIERS=0
33... METHOD    ICCG  DAMPED
34... SOLVE     V(Collector)=3.0

35... COMMENT   Switch to Newton and two carriers - save solution
36... SYMB      NEWTON  CARRIERS=2
37... SOLVE     OUT.FILE=MDEX2S
```

Figure 5-1 Output of the simulation input file *mdex2*

Mesh Specifications

The **X.MESH** and **Y.MESH** specify how the initial rectangular mesh is generated.

- The **X.MESH** statement that follows creates a grid section extending from x=0 microns (the default starting location) to x=6 microns.
- A uniform spacing of 0.25 microns is specified with the **H1** parameter.
- The first **Y.MESH** statement creates a 0.5 micron grid section at the top of the device that has a uniform spacing of 0.125 microns.

- The next **Y.MESH** statement adds a 1.5 micron grid section beneath this with a grid spacing that increases from 0.125 microns at *y*=0.5 microns to 0.4 microns at the bottom of the structure (*y*=2.0 microns).

Device Regions

The entire structure is defined as silicon with the **REGION** statement. The **ELECTR** statements are used to place the contacts. The base and the emitter are placed on the surface, and the collector is placed along the entire bottom of the structure.

Impurity Profiles

The impurity profiles for the device were specified using analytic functions, although they could also have been read from Avant! SUPREM-3, TSUPREM-4, or 1D and 2D formatted files.

The first **PROFILE** statement specifies a uniform n-type background concentration. The next two **PROFILE** statements specify p-type impurities for forming the base. High concentration n-type profiles are then used to form the emitter and buried collector for the structure.

The specification of an output file on the first **PROFILE** statement saves the profiles to be used whenever the grid is refined. This should always be done to avoid having to interpolate impurity concentrations from the nodes of an unrefined grid to the nodes of a refined grid.

Regrid

In lines 22 and 24, the grid is refined based on impurity concentration. During the regrids, a triangle is subdivided into four congruent triangles whenever the impurity concentrations at the nodes of the triangle differ by more than three orders of magnitude.

In line 27, a third refinement based on impurity concentration is performed. This regrid is confined to the vicinity of the emitter-base junction. Confining the regrid in this manner allows a fine grid to be placed in this important region and at the same time keeps the total node count from becoming excessive.

The final mesh is saved in a file for use in later simulations.

The various stages of the mesh refinement are shown in Figures 5-2 through 5-5.

Models and Initial Solution

The **MODELS** statement at line 30 is used to select various physical models that are included during the solution phase.

At this point it is desired to obtain a solution with V_{ce} =3V and V_{be} =0V which can be used as a starting point for subsequent simulations. Under these bias conditions, current flow is not expected to be significant.

The desired solution can then be obtained most efficiently by first performing a zero-carrier solution with 3V applied to the collector. This can then be used as the initial guess for a full two-carrier solution.

Following this approach, a zero-carrier solution is performed at line 34. The two-carrier solution (using Newton's method as the most efficient solution technique) is performed at line 37. Since no biases were specified here, they are defaulted to

those used during the previous solution. The **OUT.FILE** parameter causes the solution to be saved in a file for later use.

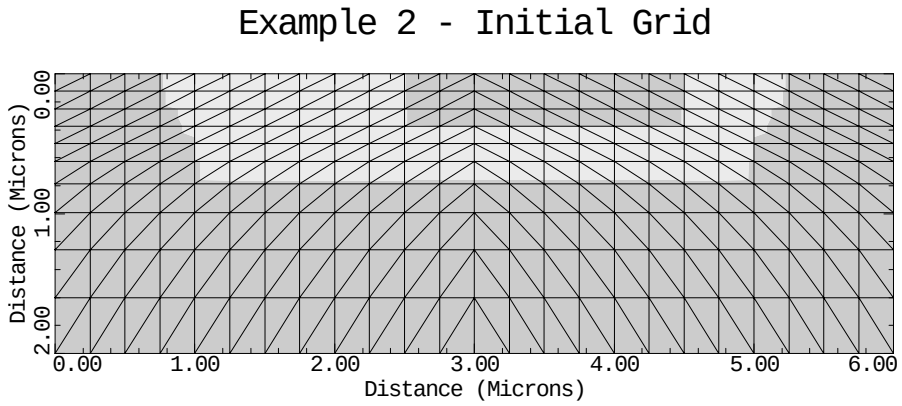


Figure 5-2 Initial grid from **PLOT.2D** at line 20 in file *mdex2*, Figure 5-1

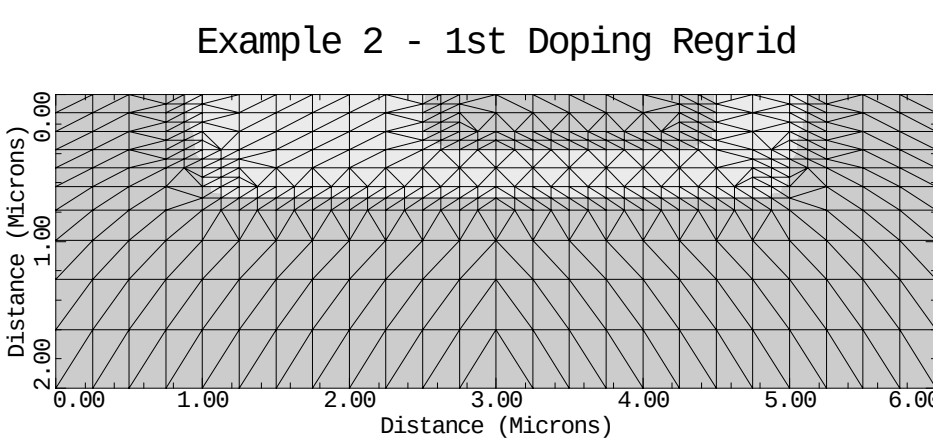


Figure 5-3 First doping regrid from **PLOT.2D** at line 23 in file *mdex2,* Figure 5-1

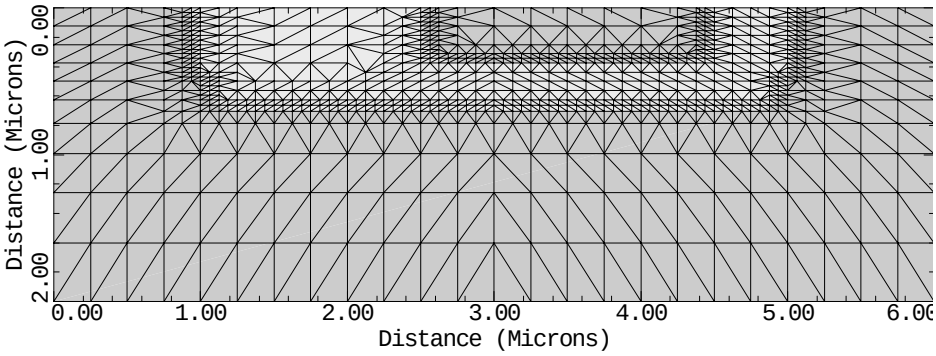


Figure 5-4 Second doping regrid from **PLOT.2D** at line 25 in file *mdex2*, Figure 5-1

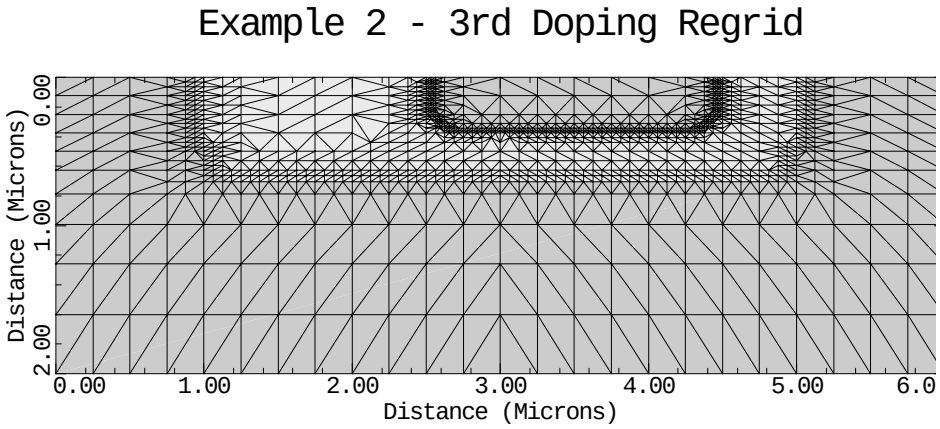


Figure 5-5 Third doping regrid from **PLOT.2D** at line 28 in file *mdex2*, Figure 5-1

Simulation of Forward Characteristics

The device structure and initial solution that were created and saved by the input file *mdex2* are read by the input file *mdex2f*. Simulations are performed for:

- Base-emitter biases of 0.2V to 0.9V.
- For each bias, an AC small-signal analysis is performed at a frequency of 10^6 Hz.

Figure 5-6 contains a portion of the output associated with the execution of Medici for the input file *mdex2f*.

```
 1... TITLE     Avant! MEDICI Example 2 - NPN Transistor Simulation
 2... COMMENT   Forward Bias Points

 3... COMMENT   Read in simulation mesh
 4... MESH      IN.FILE=MDEX2MS

 5... COMMENT   Load previous solution:  Vce=3.0 Vbe=0.0
 6... LOAD      IN.FILE=MDEX2S

 7... COMMENT   Use Newton's method with 2 carriers
 8... SYMB      NEWTON  CARRIERS=2

 9... COMMENT   Setup log file for I-V and AC data
10... LOG       OUT.FILE=MDEX2FI

11... COMMENT   Forward bias the base-emitter junction and
12... $         calculate the admittance matrix at 1.0 MHz

13... SOLVE     V(Base)=0.2  ELEC=Base  VSTEP=0.1  NSTEP=4
  ... +         AC.ANAL FREQ=1E6 TERM=Base
14... SOLVE     V(Base)=0.7  ELEC=Base  VSTEP=0.1  NSTEP=2
  ... +         AC.ANAL FREQ=1E6 TERM=Base OUT.FILE=MDEX2S7
```

Figure 5-6 Output of the simulation input file *mdex2f*

Input Statements

Newton's method is chosen as the most efficient solution technique. Before performing any solutions, the I-V and AC log file is created in line 10 to store the I-V and AC data, for later plotting.

In this example, it is desired to plot the carrier concentrations for V_{be} =0.7V. Since this is not the last bias, it is necessary to save the solution for this bias. To do this and not have to save the solutions for all the biases, two **SOLVE** statements are used.

- The first statement solves for biases through 0.6V and does not specify an output file.
- The second statement solves for the remaining biases and saves the solutions as a result of the output file specification.

Each solution on the second **SOLVE** statement is saved in a different file.

AC Small-Signal

The **SOLVE** statements also requests that an AC small-signal be performed at a frequency of 10^6 Hz after each DC solution is obtained. The parameter **TERM** is used to specify which electrode biases are to be perturbed when performing the AC small-signal analysis.

The default is to perturb all electrode biases, one at a time, so that a full admittance matrix is calculated. In this example, only the base voltage is perturbed by specifying **TERM**=Base.

Post-Processing of Forward Bias Results

For performing a post-processing analysis of the simulation results, input file *mdex2fp* reads the following:

- The mesh file created and saved by the input file *mdex2*.
- The solution and log files that were created and saved by the input file *mdex2f*.

Figures 5-7 through 5-13 contain the output associated with the execution of Medici for the input file *mdex2fp*.

Input Statements

The post-processing of forward bias results uses the following input statements.

Collector and Base Currents

The input file *mdex2fp* is shown in Figure 5-7. The statements in lines 4 through 8 use the I-V log file *MDEX2FI* to plot the collector and base currents as a function of V_{be}. The **LABEL** statement uses the default settings from I-V log file *MDEX2FI*. The resulting plot is shown in Figure 5-8.

```
 1... TITLE     Avant! MEDICI Example 2FP - NPN Transistor Simulation
 2... COMMENT   Post-Processing of MDEX2F Results

 3... COMMENT   Plot Ic and Ib vs. Vbe
 4... PLOT.1D   IN.FILE=MDEX2FI  Y.AXIS=I(Collector)  X.AXIS=V(Base)
  ... +         LINE=1  COLOR=2  TITLE="Example 2FP - Ic & Ib vs. Vbe"
  ... +         Y.LOG  POINTS  BOT=1E-14  TOP=1E-3
 5... PLOT.1D   IN.FILE=MDEX2FI  Y.AXIS=I(Base)  X.AXIS=V(Base)
  ... +         Y.LOG  POINTS  LINE=2  COLOR=3  UNCHANGE
 6... LABEL     LABEL="Ic"  X=.525  Y=1E-8
 7... LABEL     LABEL="Ib"  X=.550  Y=2E-10
 8... LABEL     LABEL="Vce = 3.0v"  X=.75  Y=1E-13

 9... COMMENT   Plot the current gain (Beta) vs. collector current
10... EXTRACT   NAME=Beta  EXPRESS=@I(Collector)/@I(Base)
11... PLOT.1D   IN.FILE=MDEX2FI  X.AXIS=I(Collector)  Y.AXIS=Beta
  ... +         TITLE="Example 2FP - Beta vs. Collector Current"
  ... +         BOTTOM=0.0  TOP=25  LEFT=1E-14  RIGHT=1E-3
  ... +         X.LOG  POINTS  COLOR=2
12... LABEL     LABEL="Vce = 3.0v"  X=5E-14  Y=23
13... COMMENT   Plot the cutoff frequency Ft=Gcb/(2*pi*Cbb)
14... EXTRACT   NAME=Ft  UNITS=Hz
  ... +         EXPRESS="@G(Collector,Base)/(6.28*@C(Base,Base))"
15... PLOT.1D   IN.FILE=MDEX2FI  X.AXIS=I(Collector)  Y.AXIS=Ft
  ... +         TITLE="Example 2FP - Ft vs. Collector Current"
  ... +         BOTTOM=1  TOP=1E10  LEFT=1E-14  RIGHT=1E-3
  ... +         X.LOG  Y.LOG  POINTS  COLOR=2
16... LABEL     LABEL="Vce = 3.0v"  X=5E-14  Y=1E9

17... COMMENT   Read in the simulation mesh and solution for Vbe=0.9v
18... MESH      IN.FILE=MDEX2MS
19... LOAD      IN.FILE=MDEX2S9

20... COMMENT   Vector plot of total current for Vbe=0.9v
21... PLOT.2D   BOUND  JUNC  SCALE  FILL
  ... +         TITLE="Example 2FP - Total Current Vectors"
22... VECTOR    J.TOTAL   COLOR=2
23... LABEL     LABEL="Vbe = 0.9v"  X=0.4  Y=1.55
24... LABEL     LABEL="Vce = 3.0v"

25... COMMENT   Potential contour plot for Vbe=0.9v
26... PLOT.2D   BOUND  JUNC  DEPL  SCALE  FILL
  ... +         TITLE="Example 2FP - Potential Contours"
27... CONTOUR   POTEN  MIN=-1  MAX=4  DEL=.25  COLOR=6
28... LABEL     LABEL="Vbe = 0.9v"  X=0.4  Y=1.55
29... LABEL     LABEL="Vce = 3.0v"

30... COMMENT   Plot doping and carrier concentrations for Vbe=0.7v
31... LOAD      IN.FILE=MDEX2S7

32... PLOT.1D   DOPING  Y.LOG  SYMBOL=1  COLOR=2  LINE=1
  ... +         BOT=1E10  TOP=1E20
  ... +         X.STA=3.5  X.END=3.5  Y.STA=0  Y.END=2
  ... +         TITLE="Example 2FP - Carrier & Impurity Conc."

33... PLOT.1D   ELECTR  Y.LOG  SYMBOL=2  COLOR=3  LINE=2  UNCHANGE
  ... +         X.STA=3.5  X.END=3.5  Y.STA=0  Y.END=2

34... PLOT.1D   HOLES   Y.LOG  SYMBOL=3  COLOR=4  LINE=3  UNCHANGE
  ... +         X.STA=3.5  X.END=3.5  Y.STA=0  Y.END=2

35... LABEL     LABEL="Vbe = 0.7v"  X=1.55  Y=4E12
36... LABEL     LABEL="Vce = 3.0v"
37... LABEL     LABEL="Doping"     SYMBOL=1  COLOR=2
38... LABEL     LABEL="Electrons"  SYMBOL=2  COLOR=3
39... LABEL     LABEL="Holes"      SYMBOL=3  COLOR=4
```

Figure 5-7 Post-processing results for input file *mdex2fp*

Beta

The **EXTRACT** statement is used in line 10 to define the symbol Beta (the collector current gain). This is then used in the **PLOT.1D** statement which follows,

along with the I-V log file *MDEX2FI*, to plot current gain as a function of the collector current. The results are shown in Figure 5-9.

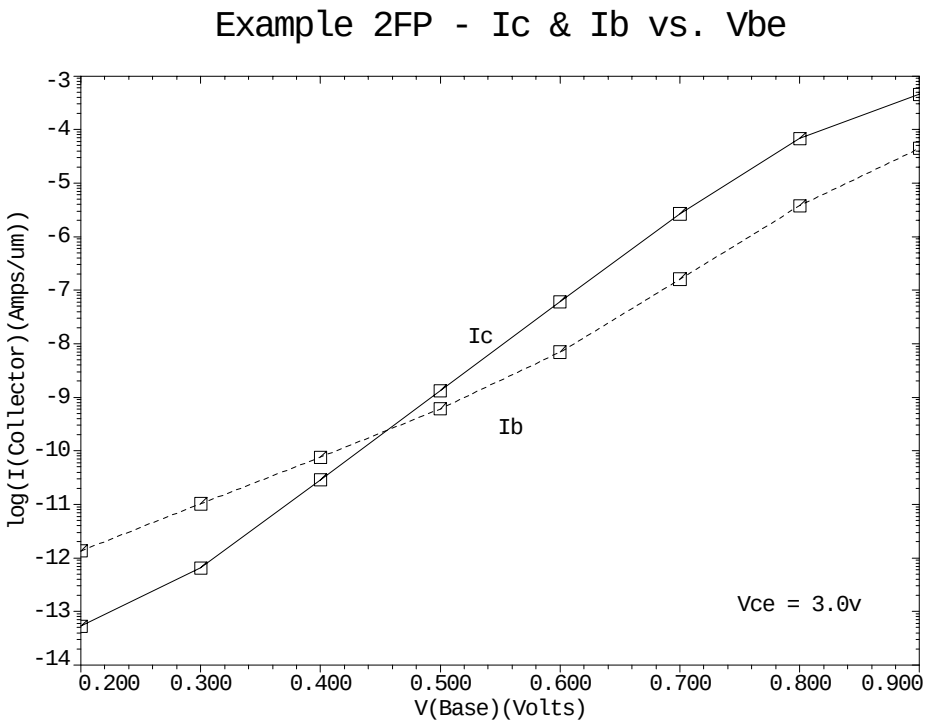


Figure 5-8 Ic and Ib vs. Vbe from **PLOT.1D** and **LABEL** at lines 4 through 8 in file *mdex2fp*, Figure 5-7

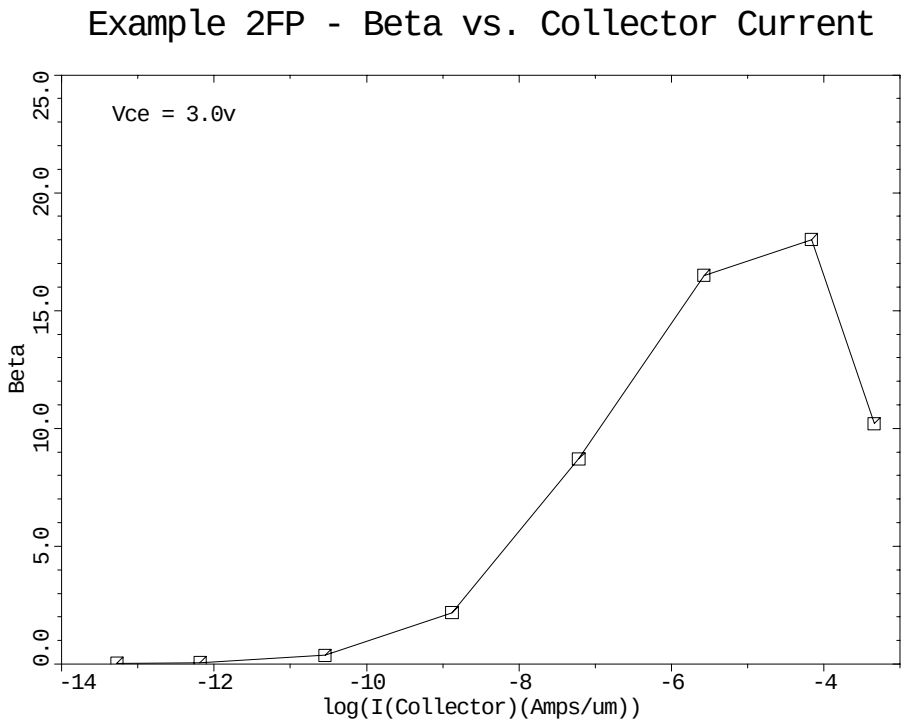


Figure 5-9 Beta vs. collector current **PLOT.1D** and **LABEL** at lines 11 through 12 in file *mdex2fp*, Figure 5-7

Cutoff Frequency

In line 14, the **EXTRACT** statement is used in conjunction with the capacitance and conductance components obtained from the AC small-signal analysis. This to calculate an approximate expression for the cutoff frequency, Ft.

The **PLOT.1D** statement at line 15 uses this definition of Ft, along with the AC small-signal analysis data stored in the file *MDEX2FI*, to plot cutoff frequency as a function of collector current. The results are shown in Figure 5-10.

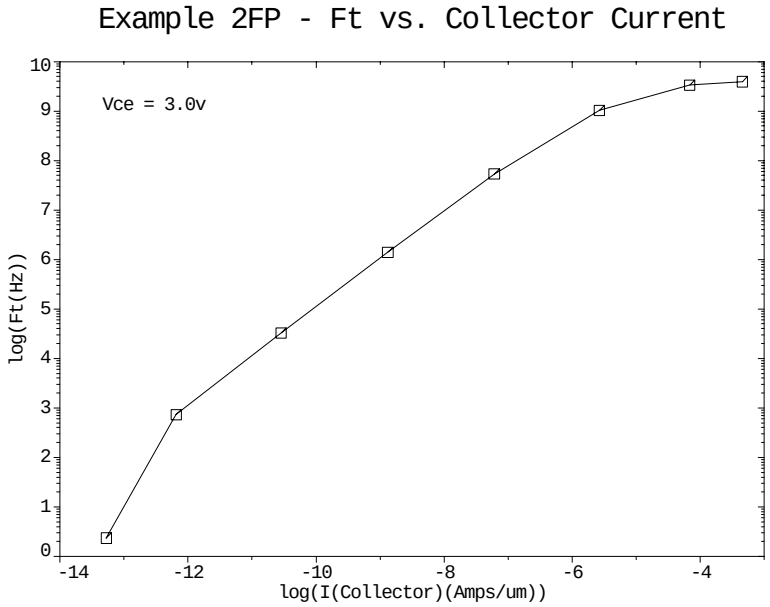


Figure 5-10 Ft vs. collector current from **PLOT.1D** and **LABEL** at lines 15 through 16 in file *mdex2fp*, Figure 5-7

Current Vectors and Potential Contours

The **MESH** statement at line 18 reads the saved mesh and the **LOAD** statement at line 19 reads the saved solution corresponding to V_{be}=0.9V. Current vectors and potential contours within the structure for this bias condition are illustrated in Figures 5-11 and 5-12.

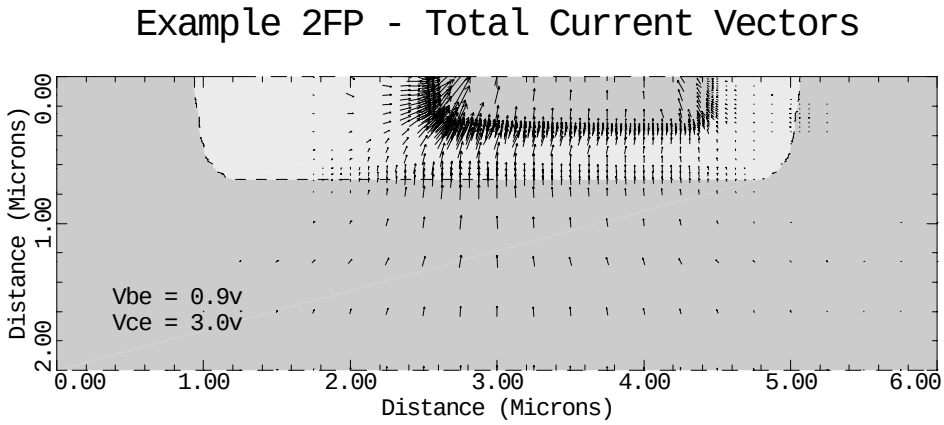


Figure 5-11 Total current vectors from **PLOT.2D**, **VECTOR**, and **LABEL** at lines 21 through 24 in file *mdex2fp*, Figure 5-7

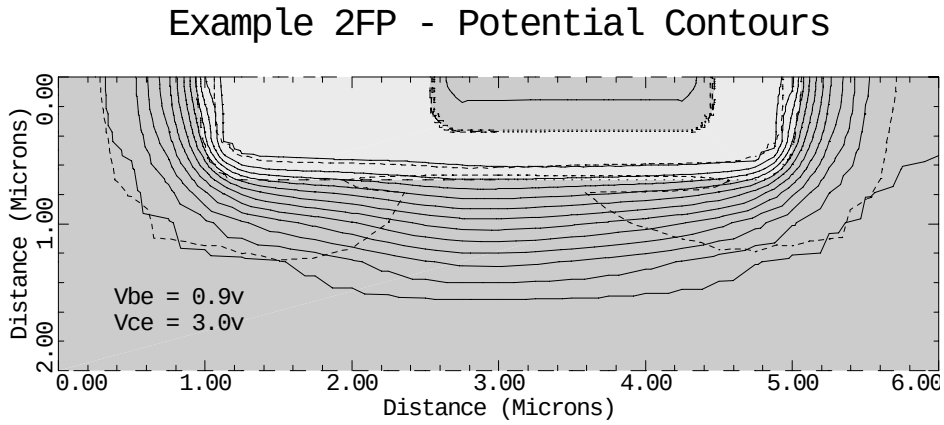


Figure 5-12 Potential contours from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 26 through 29 in file *mdex2fp*, Figure 5-7

Impurity and Carrier Concentrations

The solution for V_{be} =0.7V is read using the **LOAD** statement at line 31, and Figure 5-13 shows the impurity and carrier concentrations along a slice through the emitter for this bias.

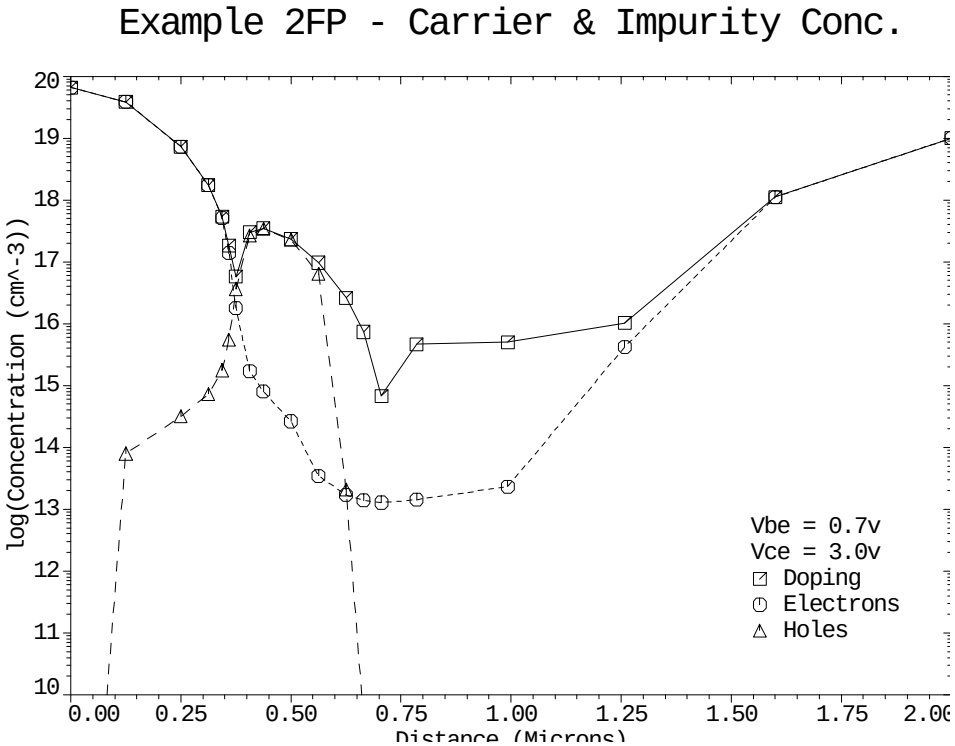


Figure 5-13 Carrier and impurity concentrations from **PLOT.1D** and **LABEL** at lines 32 through 39 in file *mdex2fp*, Figure 5-7

Simulation with Modified Emitter Region

This section details the simulation with modified emitter region. This simulation requires numerous modifications. In this example, the emitter region of the NPN transistor considered in the previous examples is modified and the forward current characteristics are repeated. The modification is such that the emitter contact at y=0 is replaced by an additional 0.25 microns of silicon and the new contact location is placed at y=-0.25 microns.

```
 1... TITLE     Avant! MEDICI Example 2P - NPN Transistor Simulation
 2... COMMENT   Simulation with Modified Emitter Region

 3... COMMENT   Initial mesh specification
 4... MESH
 5... X.MESH    WIDTH=6.0  H1=0.250
 6... Y.MESH    Y.MIN=-0.25  Y.MAX=0.0  N.SPACES=2
 7... Y.MESH    DEPTH=0.5  H1=0.125
 8... Y.MESH    DEPTH=1.5  H1=0.125  H2=0.4

 9... COMMENT   Region definition
10... REGION    NAME=Silicon  SILICON
11... REGION    NAME=Oxide  OXIDE     Y.MAX=0
12... REGION    NAME=Poly  POLYSILI  Y.MAX=0  X.MIN=2.75  X.MAX=4.25

13... COMMENT   Electrodes
14... ELECTR    NAME=Base  X.MIN=1.25  X.MAX=2.00  Y.MAX=0.0
15... ELECTR    NAME=Emitter  X.MIN=2.75  X.MAX=4.25  TOP
16... ELECTR    NAME=Collector  BOTTOM

17... COMMENT   Specify impurity profiles
18... PROFILE   N-TYPE  N.PEAK=5e15  UNIFORM  OUT.FILE=MDEX2DS
19... PROFILE   P-TYPE  N.PEAK=6e17  Y.MIN=0.35  Y.CHAR=0.16
  ... +                 X.MIN=1.25  WIDTH=3.5  XY.RAT=0.75
20... PROFILE   P-TYPE  N.PEAK=4e18  Y.MIN=0.0  Y.CHAR=0.16
  ... +                 X.MIN=1.25  WIDTH=3.5  XY.RAT=0.75
21... PROFILE   N-TYPE  N.PEAK=7e19  Y.MIN=-0.25  DEPTH=0.25  Y.CHAR=0.17
  ... +                 X.MIN=2.75  WIDTH=1.5  XY.RAT=0.75
22... PROFILE   N-TYPE  N.PEAK=1e19  Y.MIN=2.0  Y.CHAR=0.27

23... COMMENT   Regrids on doping
24... REGRID    DOPING  LOG  RATIO=3  SMOOTH=1  IN.FILE=MDEX2DS
25... REGRID    DOPING  LOG  RATIO=3  SMOOTH=1  IN.FILE=MDEX2DS

26... COMMENT   Extra regrid in emitter-base junction region only.
27... REGRID    DOPING  LOG  RATIO=3  SMOOTH=1  IN.FILE=MDEX2DS
  ... +         X.MIN=2.25  X.MAX=4.75  Y.MAX=0.50  OUT.FILE=MDEX2MP

28... PLOT.2D   GRID  SCALE  FILL
  ... +         TITLE="Example 2P - Modified Simulation Mesh"

29... COMMENT   Modify properties of polysilicon-emitter region
30... MOBILITY  POLYSILI  CONC=7E19  HOLE=2.3  FIRST  LAST
31... MATERIAL  POLYSILI  TAUP0=8E-8
32... MODEL     CONMOB  CONSRH  AUGER  BGN

33... COMMENT   Initial solution
34... SYMB      CARRIERS=0
35... METHOD    ICCG  DAMPED
36... SOLVE     V(Collector)=3.0
37... SYMB      NEWTON  CARRIERS=2
38... SOLVE

39... COMMENT   Setup log files, forward bias base-emitter junction, and
  ... +         calculate the admittance matrix at 1.0 MHz
40... LOG       OUT.FILE=MDEX2PI
41... SOLVE     V(Base)=0.2  ELEC=Base  VSTEP=0.1  NSTEP=4
  ... +         AC.ANAL  FREQ=1E6  TERM=Base
42... SOLVE     V(Base)=0.7  ELEC=Base  VSTEP=0.1  NSTEP=2
  ... +         AC.ANAL  FREQ=1E6  TERM=Base  OUT.FILE=MDEX2P7
```

Figure 5-14 Output of the simulation input file *mdex2p*

The mobility and lifetime of the minority carrier in this additional region are modified from their default silicon values to approximately represent this region as a material other than silicon.For example, this region may represent n+ polysilicon in a real device. Figures 5-14 and 5-15 contain the output associated with the execution of Medici for the input file *mdex2p*.

Modification of Mesh

To include an additional 0.25 microns of emitter material without altering the rest of the structure, it is necessary to make some modifications to the input file *mdex2* shown in Figure 5-1 for creating the device structure. This is done with the following statements:

- Two additional lines of nodes are added to the top of the initial simulation mesh by including an additional **Y.MESH** statement that places the first line of nodes at *y*=-0.25 microns.
- Two additional **REGION** statements are necessary.
  - The first additional **REGION** statement defines the top 0.25 microns of the structure to be silicon dioxide.
  - The second additional **REGION** statement redefines the portion of this region that is to be part of the emitter as polysilicon.
- The base electrode is modified by replacing "**TOP**" with "**Y.MAX**=0.0" so that contact is made to the silicon.
- The **PROFILE** statement that defines the emitter doping (line 21) is modified so that the additional emitter material has a uniform n-type concentration of 7×10^{19} cm^{-3}.

The modified simulation mesh after three regrids on impurity concentration is shown in Figure 5-15.

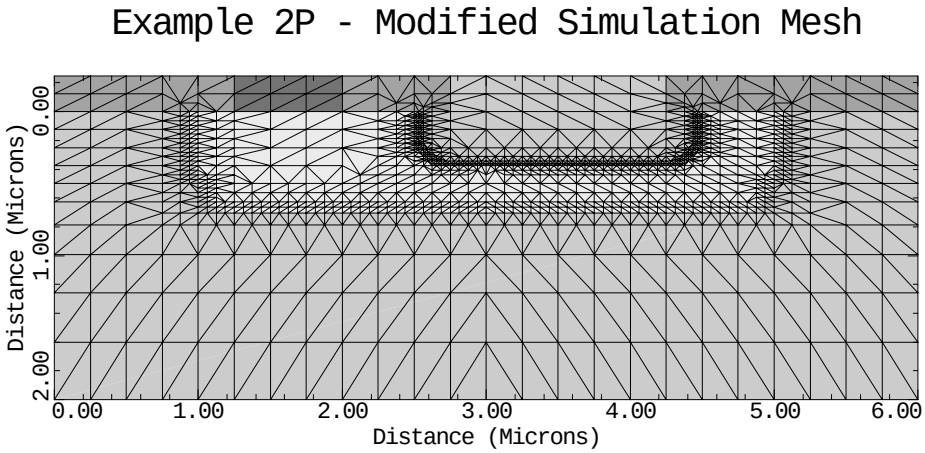


Figure 5-15 Modified simulation mesh from **PLOT.2D** at line 28 in the file *mdex2p*, Figure 5-14

Hole Mobility and Lifetime

The minority carrier (hole) mobility in the Poly region is adjusted at line 30 by specifying an entry for the concentration-dependent hole mobility table. The parameters **FIRST** and **LAST** cause this entry to be the only value in the table for the polysilicon region. This is so that the specified hole mobility will in fact apply

to any impurity concentration value in this region. The hole lifetime is also modified (line 31).

Final Adjustments and Saves

After making the above adjustments to the simulation structure, the forward current characteristics and AC small-signal analysis are repeated. The I-V and AC log file is saved, as well as the modified mesh and solutions for biases of V_{be}=0.7V, 0.8V, and 0.9V.

Post-Processing of Device with Modified Emitter

The mesh, solution, and log files that were created and saved by the input file *mdex2p* are read by the input file *mdex2pp* for performing a post-processing analysis of the simulations results. Figures 5-16 through 5-23 contain the output associated with the execution of Medici for the input file *mdex2pp*.

Metal Contact vs. Metal-Poly-Silicon

The input file *mdex2pp* shown in Figures 5-16 and 5-17 is similar to the input file *mdex2fp* shown in Figure 5-7. They differ in that the saved mesh, solution, and log files are read in from the simulations of the structure with the modified emitter.

```
  1... TITLE     Avant! MEDICI Example 2PP - NPN Transistor Simulation
  2... COMMENT   Post-Processing of MDEX2P Results

  3... COMMENT   Plot Ic and Ib vs. Vbe
  4... PLOT.1D   IN.FILE=MDEX2PI  Y.AXIS=I(Collector)  X.AXIS=V(Base)
    ... +        LINE=1  COLOR=2  TITLE="Example 2PP - Ic & Ib vs. Vbe"
    ... +        BOT=1E-14  TOP=1E-3  Y.LOG  POINTS
  5... PLOT.1D   IN.FILE=MDEX2PI  Y.AXIS=I(Base)  X.AXIS=V(Base)
    ... +        Y.LOG  POINTS  LINE=2  COLOR=3  UNCHANGE
  6... LABEL     LABEL="Ic"  X=.525  Y=1E-8
  7... LABEL     LABEL="Ib"  X=.550  Y=2E-10
  8... LABEL     LABEL="Vce = 3.0v"  X=.75  Y=1E-13

  9... COMMENT   Plot the current gain (Beta) vs. collector current
 10... EXTRACT   NAME=Beta  EXPRESS=@I(Collector)/@I(Base)
 11... PLOT.1D   IN.FILE=MDEX2PI  X.AXIS=I(Collector)  Y.AXIS=Beta
    ... +        TITLE="Example 2PP - Beta vs. Collector Current"
    ... +        BOTTOM=0.0  TOP=25  LEFT=1E-14  RIGHT=1E-3
    ... +        X.LOG  POINTS  COLOR=2
 12... LABEL     LABEL="Vce = 3.0v"  X=5E-14  Y=23
```

Figure 5-16 First part of the simulation input file *mdex2pp*

The results shown in Figures 5-18 through 5-23, however are not significantly changed from those shown in Figures 5-8 through 5-13 where the emitter region was not modified. This indicates that replacing a metal contact with a metal-poly-silicon contact has a small effect on the device behavior for the structure under consideration.

You may anticipate this result by considering the diffusion length of the minority carrier holes in the quasi-neutral emitter region $y>0$. The diffusion length for holes is given by the square root of the product of the diffusion coefficient ($D_p = (KT/q)$ (hole mobility)) and the hole lifetime.

```
13... COMMENT   Plot the cutoff frequency Ft=Gcb/(2*pi*Cbb)
14... EXTRACT   NAME=Ft  UNITS=Hz
  ... +         EXPRESS="@G(Collector,Base)/(6.28*@C(Base,Base))"
15... PLOT.1D   IN.FILE=MDEX2FI  X.AXIS=I(Collector)  Y.AXIS=Ft
  ... +         TITLE="Example 2FP - Ft vs. Collector Current"
  ... +         BOTTOM=1  TOP=1E10  LEFT=1E-14  RIGHT=1E-3
  ... +         X.LOG  Y.LOG  POINTS  COLOR=2
16... LABEL     LABEL="Vce = 3.0v"  X=5E-14  Y=1E9

17... COMMENT   Read in the simulation mesh and solution for Vbe=0.9v
18... MESH      IN.FILE=MDEX2MS
19... LOAD      IN.FILE=MDEX2S9

20... COMMENT   Vector plot of total current for Vbe=0.9v
21... PLOT.2D   BOUND  JUNC  SCALE  FILL
  ... +         TITLE="Example 2FP - Total Current Vectors"
22... VECTOR    J.TOTAL   COLOR=2
23... LABEL     LABEL="Vbe = 0.9v"  X=0.4  Y=1.55
24... LABEL     LABEL="Vce = 3.0v"

25... COMMENT   Potential contour plot for Vbe=0.9v
26... PLOT.2D   BOUND  JUNC  DEPL  SCALE  FILL
  ... +         TITLE="Example 2FP - Potential Contours"
27... CONTOUR   POTEN  MIN=-1  MAX=4  DEL=.25  COLOR=6
28... LABEL     LABEL="Vbe = 0.9v"  X=0.4  Y=1.55
29... LABEL     LABEL="Vce = 3.0v"

30... COMMENT   Plot doping and carrier concentrations for Vbe=0.7v
31... LOAD      IN.FILE=MDEX2S7

32... PLOT.1D   DOPING  Y.LOG  SYMBOL=1  COLOR=2  LINE=1
  ... +         BOT=1E10  TOP=1E20
  ... +         X.STA=3.5  X.END=3.5  Y.STA=0  Y.END=2
  ... +         TITLE="Example 2FP - Carrier & Impurity Conc."

33... PLOT.1D   ELECTR  Y.LOG  SYMBOL=2  COLOR=3  LINE=2  UNCHANGE
  ... +         X.STA=3.5  X.END=3.5  Y.STA=0  Y.END=2

34... PLOT.1D   HOLES   Y.LOG  SYMBOL=3  COLOR=4  LINE=3  UNCHANGE
  ... +         X.STA=3.5  X.END=3.5  Y.STA=0  Y.END=2

35... LABEL     LABEL="Vbe = 0.7v"  X=1.55  Y=4E12
36... LABEL     LABEL="Vce = 3.0v"
37... LABEL     LABEL="Doping"     SYMBOL=1  COLOR=2
38... LABEL     LABEL="Electrons"  SYMBOL=2  COLOR=3
39... LABEL      LABEL="Holes"        SYMBOL=3   COLOR=4
```

Figure 5-17 Second part of the simulation input file *mdex2pp*

Using a concentration-dependent hole mobility value of 85 cm^2/V-s and a concentration dependent lifetime value of 2×10^{-10} seconds (corresponding to an average impurity concentration of 3×10^{19} cm^{-3}), the hole diffusion length is found to be approximately 0.2 microns.

Since the distance from the emitter-base depletion edge to the location y=0 is approximately 0.34 microns, most of the excess holes recombine before reaching the modified emitter material ($y>0$). Therefore, the base current, and consequently the gain, for this device is not significantly affected by the presence of the modified emitter material.

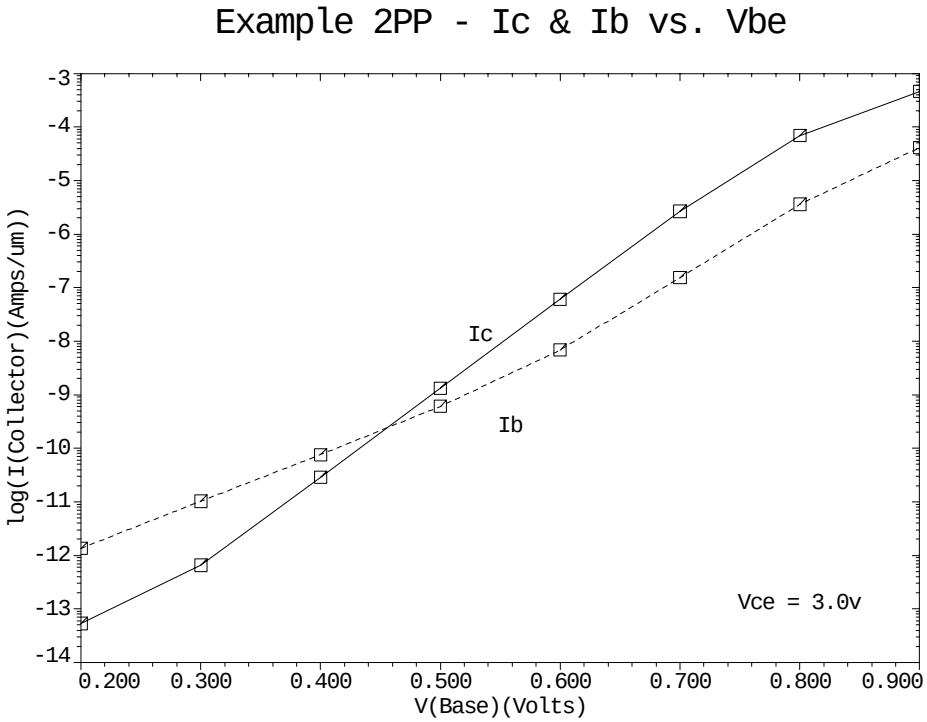


Figure 5-18 Ic and Ib vs. Vbe from **PLOT.1D** and **LABEL** at lines 4 through 8 in file *mdex2pp,* Figures 5-16 and 5-17

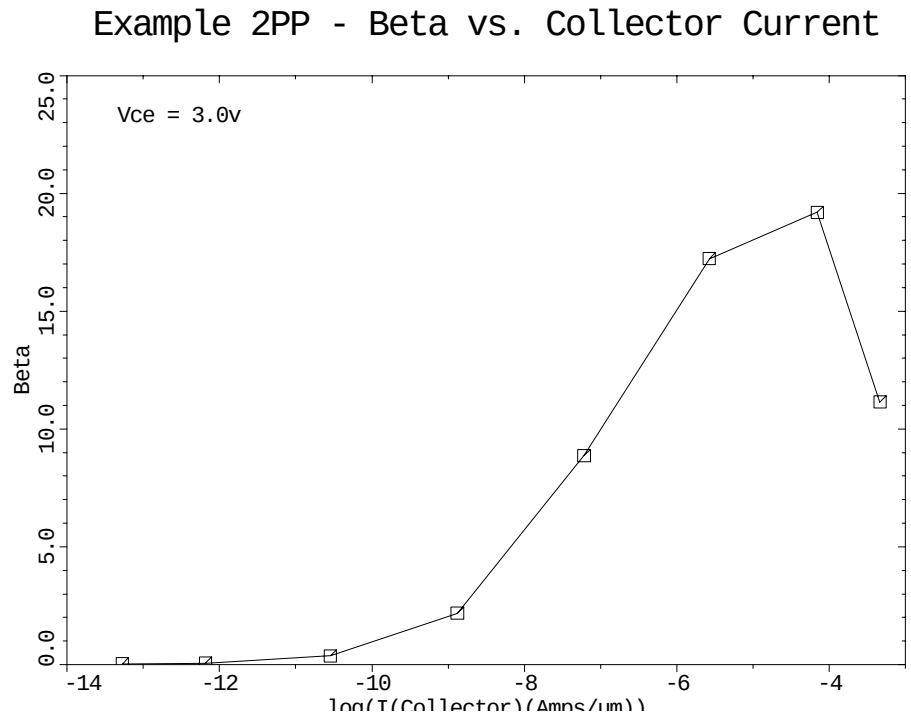


Figure 5-19 Beta vs. collector current from **PLOT.1D** and **LABEL** at lines 11 through 12 in file *mdex2pp*, Figures 5-16 and 5-17

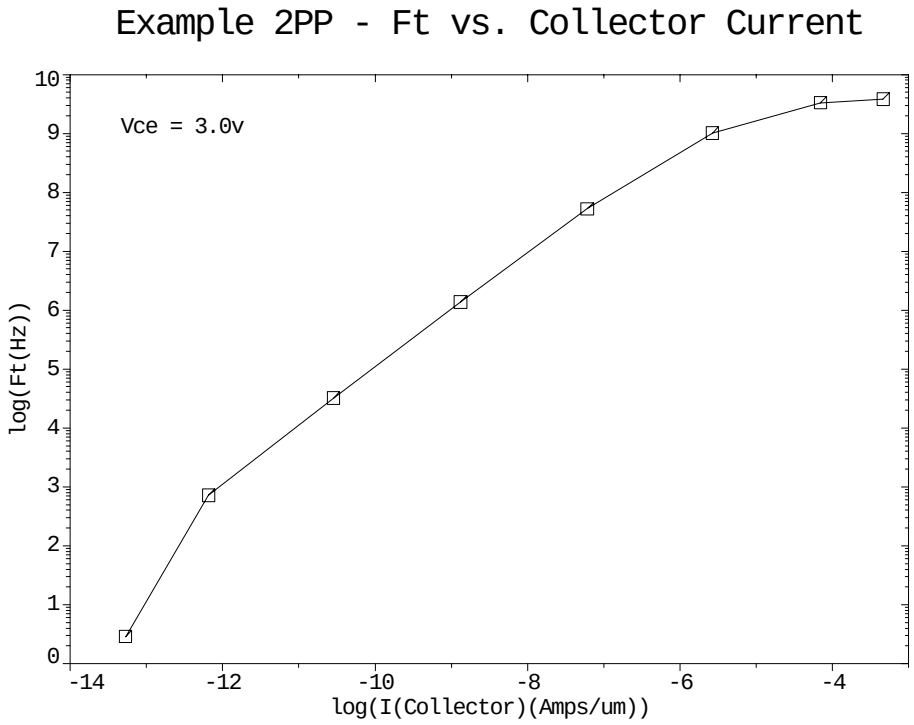


Figure 5-20 Ft vs. collector current from **PLOT.1D** and **LABEL** at lines 15 through 16 in file *mdex2pp*, Figures 5-16 and 5-17

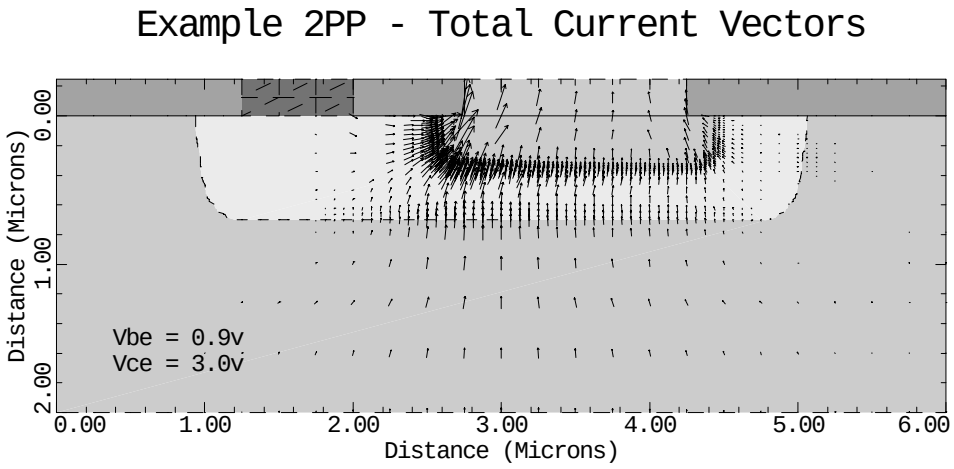


Figure 5-21 Total current vectors from **PLOT.2D**, **VECTOR**, and **LABEL** at lines 21 through 24 in file *mdex2pp*, Figures 5-16 and 5-17

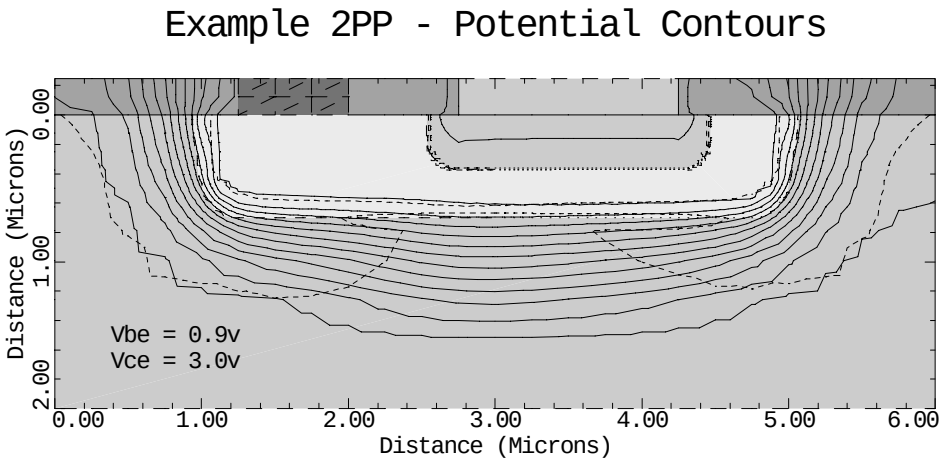


Figure 5-22 Potential contours from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 26 through 29 in file *mdex2pp*, Figures 5-16 and 5-17

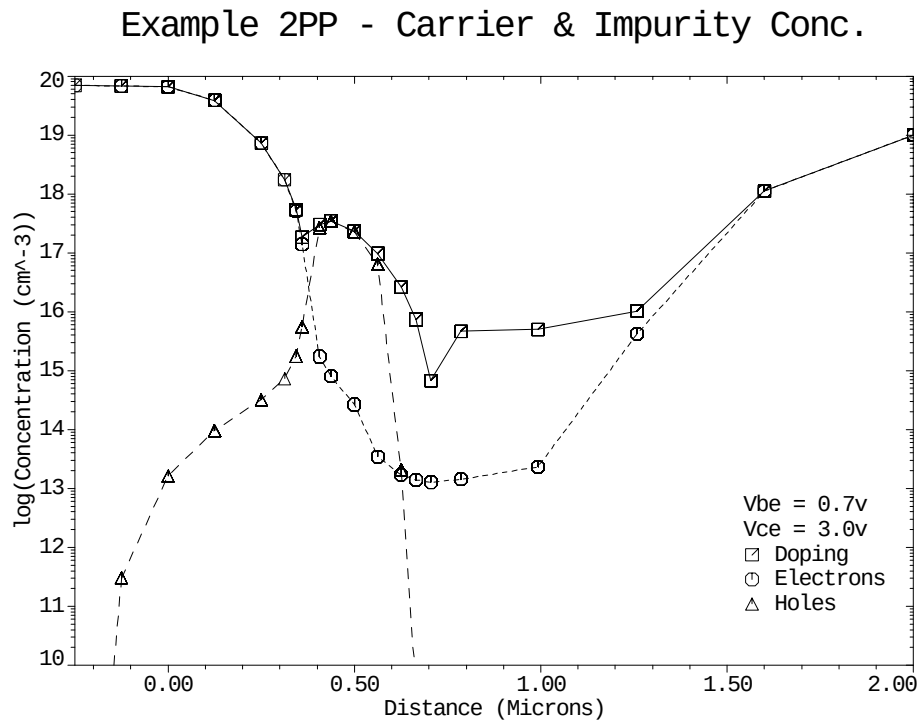


Figure 5-23 Carrier and impurity concentrations from **PLOT.1D** and **LABEL** at lines 32 through 39 in file *mdex2pp*, Figures 5-16 and 5-17

Simulation of a One-Dimensional Bipolar Transistor

In this example, a one-dimensional simulation of a bipolar transistor is performed. One-dimensional analysis allows extremely rapid device simulation, but multi-dimensional effects like emitter current crowding or variations in the parasitic base resistance cannot be simulated.

Even with these limitations, quite accurate results can be obtained and a wide variety of physical effects can be accounted for. Some examples include:

- The Early effect and its effect on output conductance
- Base push-out and other high current effects
- Low current beta roll-off due to recombination in space charge regions
- Charge storage in the base and collector and various time-dependent effects

Creating a One-Dimensional Device Structure

A one-dimensional device structure is created in Medici using a single column of triangular elements. This produces a structure with two columns of nodes. The resulting structure is not truly one-dimensional since there are two columns of nodes. A true one-dimensional structure would have only a single column of nodes.

Note:

The results of the analysis are the same as a true one-dimensional analysis as long as there is no variation in the device structure in the direction perpendicular to the column of nodes.

Base Contact

The simulation of a bipolar transistor requires that a contact be made to the base of the transistor. In a one-dimensional simulation this contact is placed across the device within the base of the transistor.

A normal electrical contact cannot be used since it would force the electron and hole concentrations to their equilibrium values with the result that no current could cross the base of the transistor from the emitter to the collector.

A **MAJORITY** carrier contact is used for the base contact. The **MAJORITY** contact only sets the quasi-Fermi potential of the majority carrier to the contact potential. (A normal electrode sets both the majority and minority carrier quasi-Fermi potentials to the contact potential.)

The result is that when the **MAJORITY** contact is used only majority carriers can leave the base via the contact. In addition, the concentration of both majority and minority carriers can deviate from the equilibrium levels within the **MAJORITY** contact.

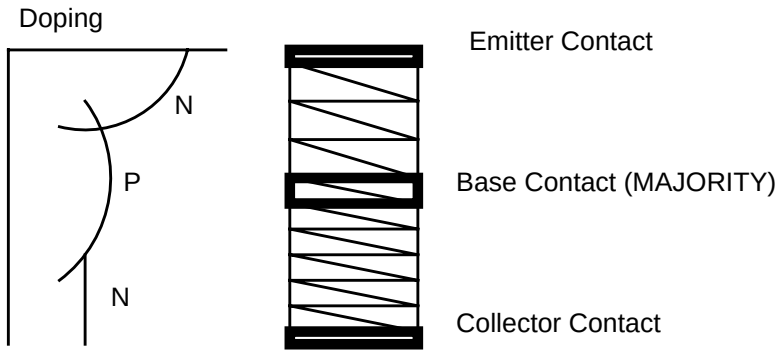


Figure 5-24 A one-dimensional bipolar transistor

Grid

The simulation input file is shown in Figures 5-25 through 5-26. The grid is created by lines 5 through 7.

- A single column of elements is created in the *y* direction by specifying (at line 5) that **N.SPACES**=1.
- The **WIDTH** of device is set to 2 microns.

  This value was chosen to be the same as the emitter width in the previous example.

In both cases the emitter area is $2.0 \times 1.0 = 2.0$ square microns. The grid spacing for the first 0.8 microns of the device is 0.01 microns. Beyond 0.8 microns the grid is allowed to expand to a spacing of 0.04 microns. The total device is 2.0 microns high and the final grid has only 272 grid points.

Electrodes

The electrodes are created by lines 12 through 14.

- The emitter is on top and covers the entire top edge of the device.
- The collector is on the bottom and covers the entire bottom edge of the device.
- The base covers a single row of nodes (i.e., 2 nodes) located at y=0.45 microns.

  The base is specified as a **MAJORITY** carrier contact. The **MAJORITY** contact is also be written as part of the mesh file and does not need to be re-specified when the mesh file is read.

Doping Profiles

The doping profiles are specified at lines 16 through 20. These profiles are identical to the two-dimensional case with the exception that the *x* coordinate information (**X.MIN**, **WIDTH**) has not been specified.

No **REGRID** operations have been performed. While regrids can be used to refine the grid in the *y* direction, they also refine the grid in the *x* direction resulting in a rapid increase in the number of nodes. It is more efficient to simply specify a fine initial grid.

```
  1... TITLE     Avant! MEDICI Example 2M - 1-D NPN Transistor Simulation
  2... COMMENT   Grid Generation and Initial Biasing

  3... COMMENT   Specify a rectangular mesh
  4... MESH
  5... X.MESH    WIDTH=2.0  N.SPACES=1
  6... Y.MESH    DEPTH=0.8  H1=0.01  H2=0.01
  7... Y.MESH    DEPTH=1.2  H1=0.01  H2=0.04

  8... COMMENT   Region definition
  9... REGION    NUM=1  SILICON

 10... COMMENT   Electrodes
 11... $         Use a majority carrier electrode for the base.
 12... ELECTR    NAME=Base  Y.MIN=0.45  Y.MAX=0.45  MAJORITY
 13... ELECTR    NAME=Emitter  TOP
 14... ELECTR    NAME=Collector  BOTTOM

 15... COMMENT   Specify impurity profiles
 16... PROFILE   N-TYPE  N.PEAK=5e15  UNIFORM     OUT.FILE=MDEX2DS
 17... PROFILE   P-TYPE  N.PEAK=6e17  Y.MIN=.35  Y.CHAR=.16
 18... PROFILE   P-TYPE  N.PEAK=4e18  Y.MIN=0    Y.CHAR=.16
 19... PROFILE   N-TYPE  N.PEAK=7e19  Y.MIN=0    Y.CHAR=.17
 20... PROFILE   N-TYPE  N.PEAK=1e19  Y.MIN=2    Y.CHAR=.27

 21... PLOT.2D   TITLE="Example 2M - 1-D Structure"  BOUND  FILL  SCALE
 22... LABEL     LABEL="n-emitter"     X=0.87  Y=0.20
 23... LABEL     LABEL="p-base"        X=0.91  Y=0.57
 24... LABEL     LABEL="base contact"  X=0.87  Y=0.43  C.SI=0.2
 25... LABEL     LABEL="n-collector"   X=0.85  Y=1.50

 26... COMMENT   Specify some models
 27... MODELS    CONMOB  CONSRH  AUGER  BGN

 28... COMMENT   Use Newton's method with 2 carriers
 29... SYMB      NEWTON  CARRIERS=2

 30... COMMENT   Setup log file for I-V data
 31... LOG       OUT.FILE=MDEX2MI

 32... COMMENT   Find the base width (defined as p>1e15)
 33... EXTRACT   NAME=w1  COND="@p>1e15"  EXPRESS="min(@w1;@y)" INIT=1000
 34... EXTRACT   NAME=w2  COND="@p>1e15"  EXPRESS="max(@w2;@y)" INIT=-1000
 35... EXTRACT   NAME=wb  EXPRESS="@w2-@w1"  UNITS=Microns
 36... COMMENT   Forward bias the base-emitter junction

 37... SOLVE     V(Collector)=3.0  V(Base)=0.2  ELEC=Base
   ... +         VSTEP=0.05  NSTEP=9  AC.ANAL  TERM=Base  FREQ=1E6
 38... SOLVE     V(Base)=0.70  OUT.FILE=MDE2MS7
   ... +         AC.ANAL  TERM=Base  FREQ=1E6
 39... SOLVE     V(Base)=0.75  ELEC=Base  VSTEP=0.05  NSTEP=3
   ... +         AC.ANAL  TERM=Base  FREQ=1E6
```

Figure 5-25 First part of the simulation input file *mdex2m*

Solutions

The remainder of the input file is very much like the files *mdex2f* and *mdex2fp* presented in the previous examples.

- The base voltage is ramped from 0.2V to 0.9V. Since this one-dimensional example runs significantly faster than its two-dimensional counterpart, more bias points have been requested.

- AC small-signal analysis is used to calculate the transconductance "gm" and the total base capacitance. These are used to estimate the cutoff frequency "Ft".

```
40... COMMENT   Plot Ic and Ib vs. Vbe
41... PLOT.1D   IN.FILE=MDEX2MI  Y.AXIS=I(Collector)  X.AXIS=V(Base)
  ... +         Y.LOG  POINTS  LINE=1  COLOR=2
  ... +         TITLE="Example 2M - Ic & Ib vs. Vbe"
42... PLOT.1D   IN.FILE=MDEX2MI  Y.AXIS=I(Base)  X.AXIS=V(Base)
  ... +         Y.LOG  POINTS  LINE=2  COLOR=3  UNCHANGE
43... LABEL     LABEL="Vce = 3.0v"
44... LABEL     LABEL="Ic"  X=.525  Y=1.5E-8
45... LABEL     LABEL="Ib"  X=.550  Y=2.0E-10

46... COMMENT   Plot the current gain (Beta) vs. collector current
47... EXTRACT   Name=Beta  EXPRESS=@I(Collector)/@I(Base)
48... PLOT.1D   IN.FILE=MDEX2MI  X.AXIS=I(Collector)  Y.AXIS=Beta
  ... +         X.LOG  POINTS  COLOR=2
  ... +         TITLE="Example 2M - Beta vs. Collector Current"
49... LABEL     LABEL="Vce = 3.0v"

50... COMMENT   Plot cutoff frequency (Ft) vs collector current
51... COMMENT   Ft = Gcb/(2*pi*Cbb)
52... EXTRACT   NAME=Ft  UNITS=Hz
  ... +         EXPRESS="@G(Collector,Base)/(6.28*@C(Base,Base))"
53... PLOT.1D   X.AX=I(Collector)   Y.AX=Ft
  ... +         TITLE="Example 2M - Ft vs. Collector Current"
  ... +         X.LOG  Y.LOG  POINTS  COLOR=2  IN.FILE=MDEX2MI
  ... +         BOTTOM=1  TOP=1E10  LEFT=1E-14  RIGHT=1E-3
54... LABEL     LABEL="Vce = 3.0v"


55... COMMENT   Plot doping and carrier concentrations for Vbe=0.7v
56... LOAD      IN.FILE=MDE2MS7

57... PLOT.1D   DOPING  Y.LOG  SYMBOL=1  COLOR=2  LINE=1
  ... +         BOT=1E10  TOP=1E20
  ... +         X.STA=0  X.END=0  Y.STA=0  Y.END=2  C.SIZE=0.15
  ... +         TITLE="Example 2M - Carrier & Impurity Conc."

58... PLOT.1D   ELECTR  Y.LOG  SYMBOL=2  COLOR=3  LINE=2  UNCHANGE
  ... +         X.STA=0  X.END=0  Y.STA=0  Y.END=2  C.SIZE=0.15

59... PLOT.1D   HOLES   Y.LOG  SYMBOL=3  COLOR=4  LINE=3  UNCHANGE
  ... +         X.STA=0  X.END=0  Y.STA=0  Y.END=2  C.SIZE=0.15

60... LABEL     LABEL="Vbe = 0.7v"  X=1.55  Y=4E12
61... LABEL     LABEL="Vce = 3.0v"
62... LABEL     LABEL="Doping"     SYMBOL=1  COLOR=2
63... LABEL     LABEL="Electrons"  SYMBOL=2  COLOR=3
64... LABEL     LABEL="Holes"      SYMBOL=3  COLOR=4

65... COMMENT   Plot base width vs collector current
66... PLOT.1D   X.AXIS=I(Collector)  Y.AXIS=wb  X.LOG
  ... +         IN.FILE=MDEX2MI  COLOR=2  POINTS
  ... +         TITLE="Example 2M - Electrical Base Width"
67... LABEL     LABEL="Vce = 3.0v"
```

Figure 5-26 Second part of the simulation input file *mdex2m*

Graphical Output

It is interesting to compare the results, shown in Figures 5-27 through 5-31 with the results of the two-dimensional analysis shown in Figures 5-8 through 5-13. The results with one- and two-dimensional analyses are very similar in this particular example.

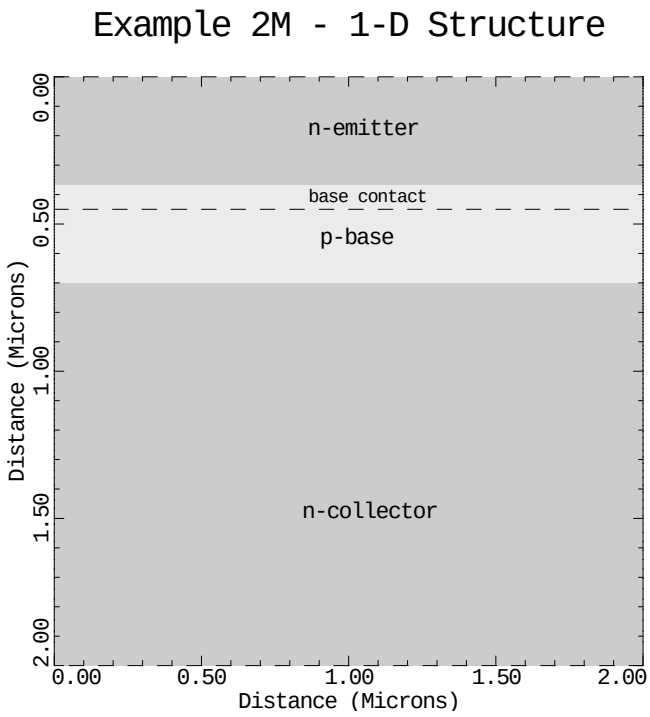


Figure 5-27 Device structure from **PLOT.2D** and **LABEL** at lines 21 through 25 in file *mdex2m*, Figure 5-25

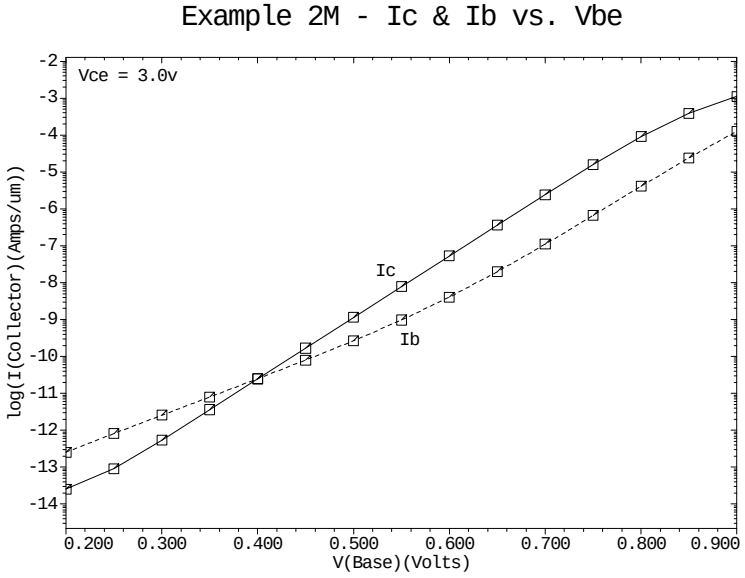


Figure 5-28 Base and collector current as a function of the base-emitter voltage from **PLOT.1D** and **LABEL** at lines 41 through 45 of the input file *mdex2m*

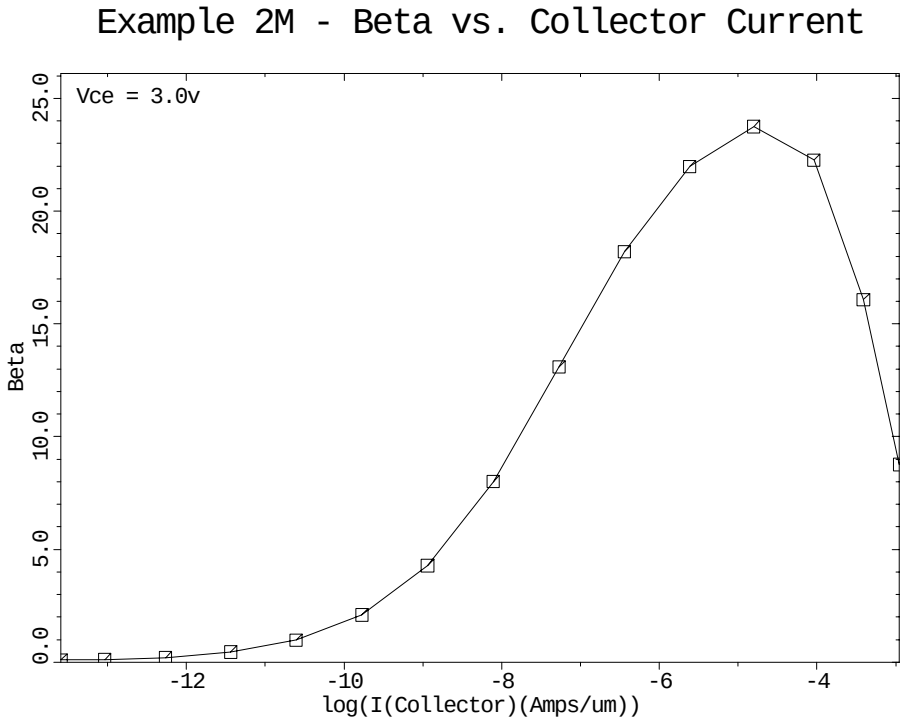


Figure 5-29 Current gain versus collector current from **EXTRACT**, **PLOT.1D**, and **LABEL** at lines 47 through 49 in file *mdex2m*

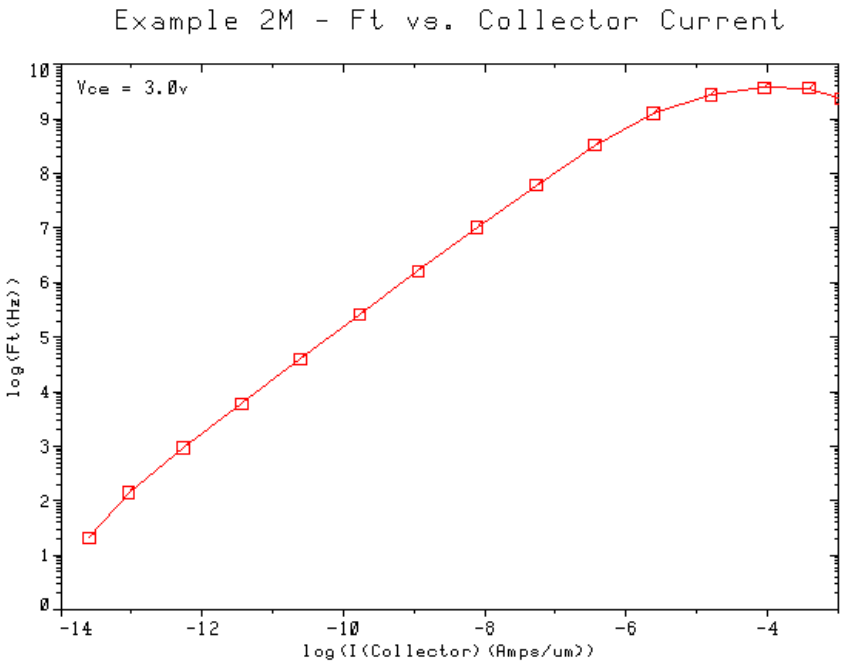


Figure 5-30 Cutoff frequency versus collector current from **EXTRACT**, **PLOT.1D**, and **LABEL** at lines 52 through 54 in *mdex2m*

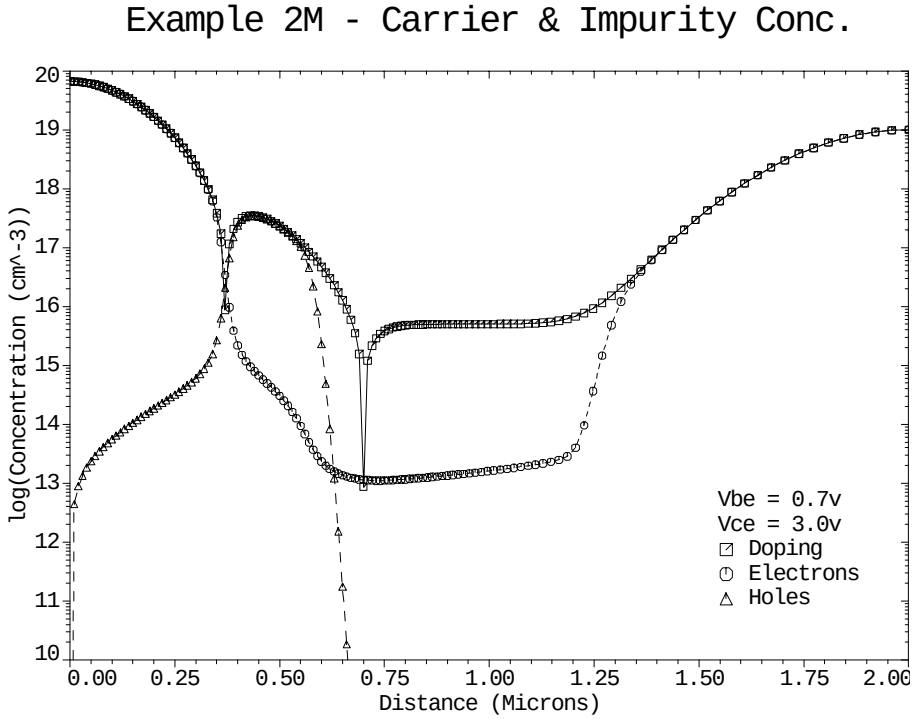


Figure 5-31 Electron, hole and doping concentrations from **PLOT.1D** and **LABEL** at lines 57 through 64 in file *mdex2m*

Diode and Lumped Element Examples

Example Specifications

This chapter details three capabilities of the Medici program.

- A transient simulation of a PN diode.
- The effects of and differences between lumped resistive elements and distributed contact resistance.
- The use of lumped resistance, capacitance, and inductance as the load for a MOSFET.

The simulation uses the following files and structure:

- The input file *mdex3* develops the simulation structure and simulates the transient turn-on characteristics for the diode.
- The input file *mdex3h* plots the hole concentration through the device at various times during the turn-on.
- The input *mdex3e* plots electron concentrations through the device at various times during the turn-on.

Transient Simulation of a PN Diode

The input file *mdex3* creates the simulation structure for the PN-diode device and then simulates the transient turn-on characteristics. The output associated with the execution of Medici for the input file *mdex3* is shown in Figures 6-1 through 6-4.

Mesh Generation

The first step in creating the device structure is to generate an initial mesh, as is done in lines 3 through 5 of the input file shown in Figure 6-1.

```
 1... TITLE     Avant! MEDICI Example 3 - PN Diode Transient Simulation

 2... COMMENT   Create an initial simulation mesh
 3... MESH      ^DIAG.FLI
 4... X.MESH    X.MAX=3.0  H1=0.50
 5... Y.MESH    Y.MAX=3.0  H1=0.25

 6... COMMENT   Region and electrode statements
 7... REGION    NAME=Silicon  SILICON
 8... ELECTR    NAME=Anode  TOP  X.MAX=1.0
 9... ELECTR    NAME=Cathode  BOTTOM

10... COMMENT   Specify impurity profiles
11... PROFILE   N-TYPE  N.PEAK=1E15  UNIF      OUT.FILE=MDEX3DS
12... PROFILE   P-TYPE  N.PEAK=1E19  X.MIN=0  WIDTH=1.0  X.CHAR=.2
  ... +                              Y.MIN=0  Y.JUNC=.5

13... COMMENT   Refine the mesh with doping regrids
14... REGRID    DOPING  LOG  RAT=3  SMOOTH=1  IN.FILE=MDEX3DS
15... REGRID    DOPING  LOG  RAT=3  SMOOTH=1  IN.FILE=MDEX3DS
16... REGRID    DOPING  LOG  RAT=3  SMOOTH=1  IN.FILE=MDEX3DS
  ... +         OUT.FILE=MDEX3MS
17... PLOT.2D   GRID  TITLE="Example 3 - Simulation Mesh"  SCALE FILL

18... COMMENT   Attach a lumped resistance to the Anode contact
19... CONTACT   NAME=Anode  RESIST=1E5

20... COMMENT   Specify physical models to use
21... MODELS    SRH  AUGER  CONMOB  FLDMOB

22... COMMENT   Symbolic factorization
23... SYMB      NEWTON  CARRIERS=2

24... COMMENT   Create a log file for the transient I-V data
25... LOG       OUT.FILE=MDEX3I

26... COMMENT   Perform a 0-volt steady state solution, then simulate
27... $         the transient turn-on characteristics for the diode.
28... SOLVE     OUT.FILE=MDE3S00
29... SOLVE     V(Anode)=2  TSTEP=1E-12  TSTOP=10E-9  OUT.FILE=MDE3S01

30... COMMENT   Plot the diode current and contact voltage vs. time
31... PLOT.1D   X.AXIS=TIME  Y.AXIS=I(Anode)
  ... +         POINTS  TOP=2E-5  RIGHT=.5E-9
  ... +         TITLE="Example 3 - Current vs. Time"  COLOR=2
32... PLOT.1D   X.AXIS=TIME  Y.AXIS=V(Anode)
  ... +         POINTS  TOP=1.1   RIGHT=.5E-9
  ... +          TITLE="Example 3 - Contact Voltage vs. Time"  COLOR=2
```

Figure 6-1 Output of the simulation input file *mdex3*

The **MESH** statement initiates the mesh generation. The **X.MESH** and **Y.MESH** statements are used to define the placement of lines of nodes within the structure. The structure created by the **X.MESH** and **Y.MESH** statements extends from 0 to 3 microns in both the x and y directions. The grid spacing in each direction is uniform.

With initial mesh defined, it is now the time to define the device. The entire device is defined as silicon with the **REGION** statement in line 7.

The **ELECTR** statements locate the contacts within the device structure.

- The anode is placed along the left edge of the top surface and makes contact with what will eventually be the p-type material of the diode.
- The cathode is placed along the bottom surface and makes contact with what will eventually be the n-type material of the diode.

Two **PROFILE** statements define the impurity distribution for the structure.

- The first **PROFILE** statement places a uniform concentration of n-type material over the entire device.
- The second **PROFILE** statement then adds a Gaussian distribution of p-type material to the top-left portion of the structure to form the diode.

The profile information is saved in a file to be used during grid refinement by specifying the **OUT.FILE** parameter on the first **PROFILE** statement.

Doping Regrid

It is now necessary to refine the grid so it is adequate for simulation. In lines 14 through 16, the simulation grid is refined based on **DOPING**. If the impurity concentration varies by more than three orders of magnitude over a triangle of the existing mesh, the triangle is divided into four congruent triangles by adding new nodes.

Each **REGRID** statement creates one new level of triangles. The impurity concentration at the new nodes of each refined mesh is calculated using the profile information stored in the file specified by **IN.FILE**. Performing the regrids one level at a time and using the doping file specified by **IN.FILE** avoids problems that may arise due to interpolation errors.

The final refined mesh is shown in Figure 6-2.

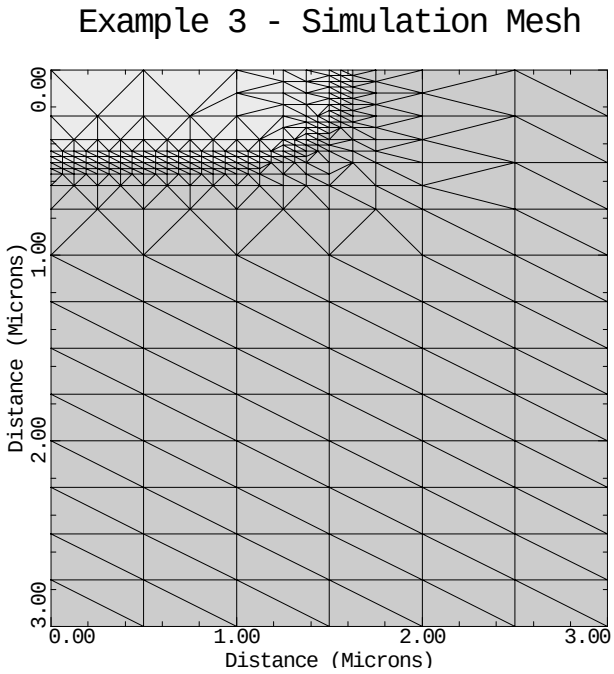


Figure 6-2 Simulation mesh from **PLOT.2D** at line 17 in file *mdex3,* Figure 6-1

Transient Analysis

The solution part of the input file is described in the following sections.

Models and Boundary Conditions

Before generating solutions, any special boundary conditions or physical models to be used should be specified. For this example:

- A lumped resistance of 10^5 Ω-μm is attached to the anode.
- Specify that both Shockley-Read-Hall and Auger recombination are to be used.
- Specify concentration and field-dependent mobilities.

Solution Methods

Since the behavior of both electrons and holes will be important in this example, a two-carrier simulation is requested on the **SYMB** statement at line 23. Newton's method is also selected as the most efficient solution technique.

Before actually performing the first solution, a log file is specified that will store the terminal I-V data at each bias and/or time point of the simulation.

Initial Solution

Since no bias values are specified on the first **SOLVE** statement (line 28), all biases default to zero. The results of this solution point are stored for future processing or plotting in the file specified by **OUT.FILE**.

Transient Solutions

The second **SOLVE** statement (line 29) begins the transient analysis of the turn-on characteristics for the diode. The bias applied to the anode is instantaneously switched to 2 volts at time t=0.

To perform a transient analysis with Medici, it is only necessary to specify the first time step to use and the stopping time. Medici will automatically choose the intermediate time steps to insure an accurate solution. In this example, a 1 picosecond initial time step is chosen with the **TSTEP** parameter, and a stopping time of 10 nanoseconds is chosen with the **TSTOP** parameter (line 29).

Specifying the **OUT.FILE** parameter (line 29) causes the solution information to be stored in files for future processing or plotting. The solution for each time step will be stored in a separate file, with the identifier specified by **OUT.FILE** being incremented for each solution.

CAUTION

It should be kept in mind that since Medici chooses the time steps automatically, it is unknown at this point how many solutions will actually be performed. Therefore, specifying OUT.FILE in a transient analysis such as this can potentially create a very large number of output files.

Graphical Results

Line 31 requests that a plot of diode current versus time be drawn. This is shown in Figure 6-3. Note the variation in time step size selected automatically by Medici during the analysis.

It should also be noted that although the analysis above was carried out to 10 nanoseconds, the plot shown in Figure 6-3 was limited to the first 0.5 nanoseconds.

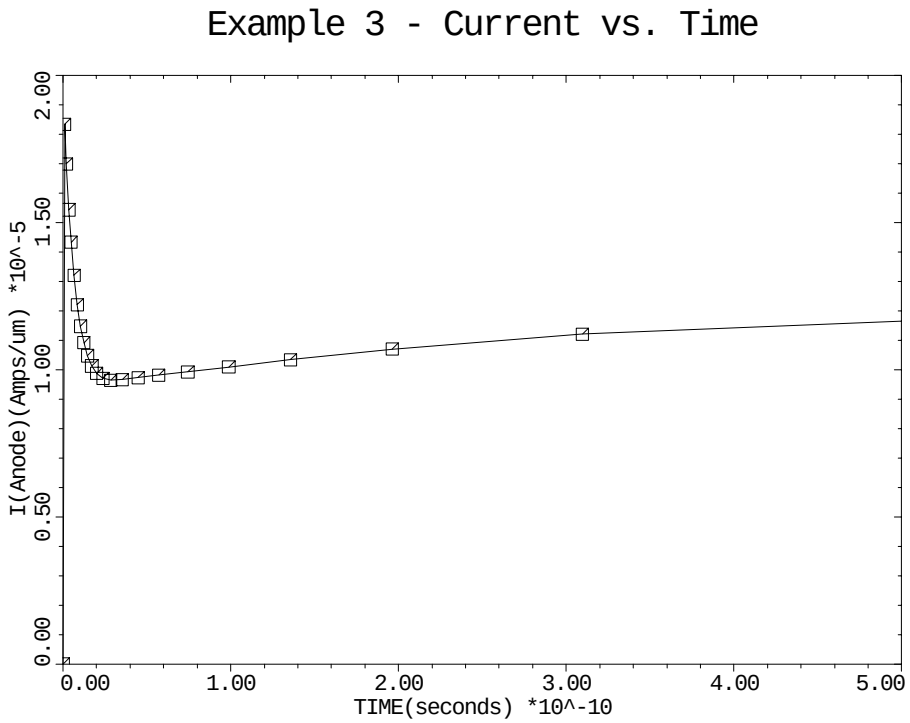


Figure 6-3 Current vs. time from **PLOT.1D** at line 31 in file *mdex3*, Figure 6-1

Line 32 requests that a plot of the anode voltage versus time be drawn. This is shown in Figure 6-4. Although the applied bias remained at 2 volts during the entire analysis, the actual contact voltage is always less than 2 volts and varies with time. This is due to the lumped resistance attached between the applied bias and the anode.

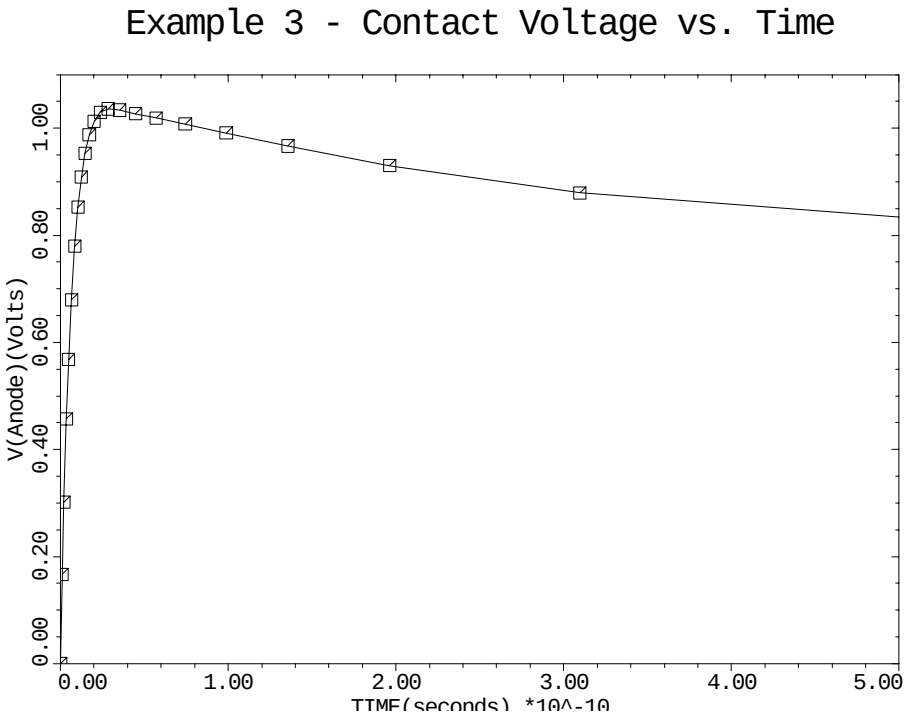


Figure 6-4 Contact voltage from **PLOT.1D** at line 32 in file *mdex3*, Figure 6-1

One-Dimensional Plots of Hole Concentrations

The device structure that was created and saved by the input file *mdex3* is read by the input file *mdex3h*. Additionally, solution files created by the input file *mdex3* at various simulation times are read by the input file *mdex3h* and are used to plot the hole concentration along a one-dimensional vertical slice through the structure at those times.

Figures 6-5 and 6-6 contain the output associated with the execution of Medici input file *mdex3h*.

In Figure 6-5, the **UNCH** parameter is used on the second and subsequent **PLOT.1D** statements to allow all the curves to be plotted in the same figure with the scaling established by the first **PLOT.1D** statement.

In Figure 6-6, the input file *mdex3h* adds labels to the figure to identify each curve with the corresponding simulation time. The simulation times were obtained from the printed output associated with the execution of *mdex3*.

```
 1... TITLE     Avant! MEDICI Example 3H - PN Diode Transient Simulation
 2... COMMENT   Plot hole concentration at various times

 3... COMMENT   Read in simulation structure
 4... MESH      IN.FILE=MDEX3MS

 5... TITLE     Example 3H - Hole Concentration
 6... LOAD      IN.FILE=MDE3S00
 7... PLOT.1D   HOLES  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.
 8... LOAD      IN.FILE=MDE3S05
 9... PLOT.1D   HOLES  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
10... LOAD      IN.FILE=MDE3S07
11... PLOT.1D   HOLES  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
12... LOAD      IN.FILE=MDE3S10
13... PLOT.1D   HOLES  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
14... LOAD      IN.FILE=MDE3S15
15... PLOT.1D   HOLES  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
16... LOAD      IN.FILE=MDE3S19
17... PLOT.1D   HOLES  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
18... LOAD      IN.FILE=MDE3S28
19... PLOT.1D   HOLES  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH

20... LABEL     LABEL="0"            X=1.10  Y=5.0E6
21... LABEL     LABEL="7"            X=1.3   Y=2.5E7
22... LABEL     LABEL="10"           X=1.5   Y=5.0E7
23... LABEL     LABEL="17"           X=1.80  Y=8.5E8
24... LABEL     LABEL="44"           X=2.10  Y=1.0E12
25... LABEL     LABEL="135"          X=2.13  Y=2.0E14
26... LABEL     LABEL="10000 psec"   X=2.10  Y=1.5E16
```

Figure 6-5 Output of the simulation input file *mdex3h*

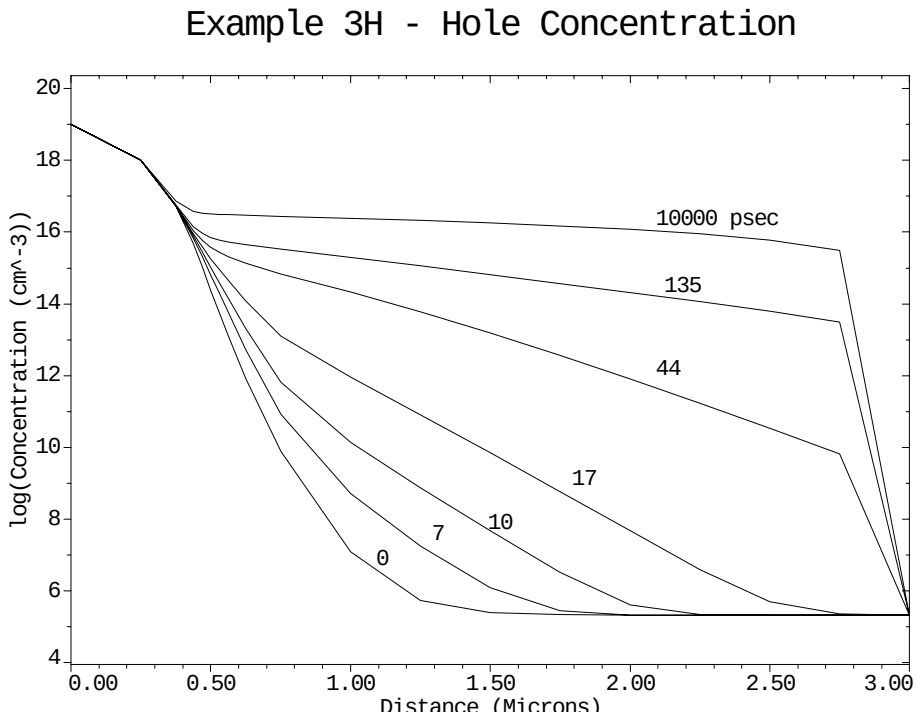


Figure 6-6 Hole concentration from **PLOT.1D** at lines 7, 9, 11, 13, 15, 17, and 19, and **LABEL** at lines 20 through 26 in file *mdex3h,* Figure 6-5

One-Dimensional Plots of Electron Concentration

The device structure that was created and saved by the input file *mdex3* is read by the input file *mdex3e*. Additionally, solution files created by the input file *mdex3* at various simulation times are read by the input file *mdex3e* and are used to plot the electron concentration along a one-dimensional vertical slice through the structure at those times.

Figures 6-7 and 6-8 contain the output associated with the execution of Medici for the input file *mdex3e*.

In Figure 6-7, the **UNCH** parameter is used on the second and subsequent **PLOT.1D** statements to allow all the curves to be plotted in the same figure with the scaling established by the first **PLOT.1D** statement.

In Figure 6-8, the input file *mdex3e* also adds labels to the figure to identify each curve with the corresponding simulation time. The simulation times were obtained from the printed output associated with the execution of *mdex3*.

```
 1... TITLE     Avant! MEDICI Example 3E - PN Diode Transient Simulation
 2... COMMENT   Plot electron concentration at various times

 3... COMMENT   Read in simulation structure
 4... MESH      IN.FILE=MDEX3MS

 5... TITLE     Example 3E - Electron Concentration
 6... LOAD      IN.FILE=MDE3S00
 7... PLOT.1D   ELECT  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.
  ... +         TOP=1E18
 8... LOAD      IN.FILE=MDE3S05
 9... PLOT.1D   ELECT  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
10... LOAD      IN.FILE=MDE3S07
11... PLOT.1D   ELECT  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
12... LOAD      IN.FILE=MDE3S10
13... PLOT.1D   ELECT  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
14... LOAD      IN.FILE=MDE3S15
15... PLOT.1D   ELECT  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
16... LOAD      IN.FILE=MDE3S19
17... PLOT.1D   ELECT  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH
18... LOAD      IN.FILE=MDE3S28
19... PLOT.1D   ELECT  Y.LOG  X.ST=0.  X.EN=0.  Y.ST=0.  Y.EN=3.  UNCH

20... LABEL     LABEL="0"           X=0.77  Y=1.2E11
21... LABEL     LABEL="7"           X=0.65  Y=8E11
22... LABEL     LABEL="10"          X=0.60  Y=1E13
23... LABEL     LABEL="17"          X=0.57  Y=2E14
24... LABEL     LABEL="44"          X=0.57  Y=1.7E15
25... LABEL     LABEL="135"         X=0.65  Y=6E15
26... LABEL     LABEL="10000 psec"  X=0.75  Y=4E16
```

Figure 6-7 Output of the simulation input file *mdex3e*

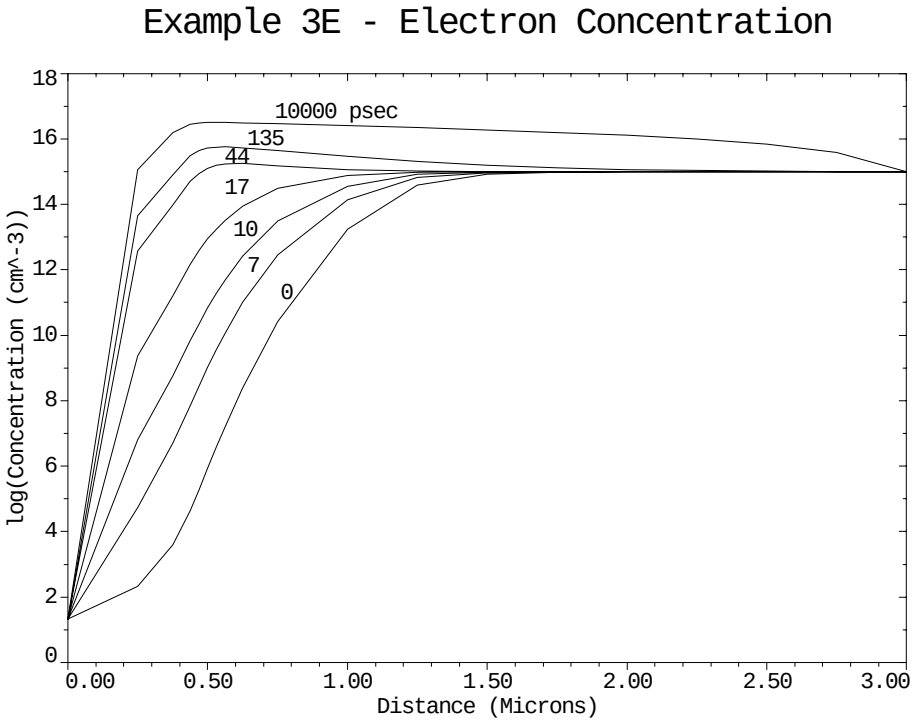


Figure 6-8 Electron concentration from **PLOT.1D** at lines 7, 9, 11, 13, 15, 17, and 19, and **LABEL** at lines 20 through 26 in file *mdex3e*, Figure 6-7

Stress-Induced Leakage Example

The following example describes the simulation of a silicon PN diode created near a LOCOS isolation structure. Such isolation structures produce localized regions of large stress which can severely impact device performance. The analysis is performed in one input file, *mdexstress.inp*, and consists of three steps:

1. The basic device structure and stress tensor are loaded from a *TIF* file created by TSUPREM-4. A simple doping profile for a PN diode is created using the **PROFILE** statement.
2. A static simulation of the reverse-bias leakage current *without* stress effects is performed.
3. A static simulation of the reverse-bias leakage current *with* stress effects is performed.

Reading and Specification of the Device Structure

As shown in Figure 6-9, the name of the TIF file containing the device structure and the simulated stress tensor is given in line 3 as STRESS1.TIF. The structure is loaded into Medici using the **MESH** statement on line 5. The thin top region named NITRIDE1 is turned into an electrode in line 7. A doping profile is then

created in lines 9 and 10. In line 12, the stress tensor is read from the *TIF* file using the **OTHER** parameter on the **PROFILE** statement.

```
 1... TITLE     MEDICI Example - Stress-Induced Leakage Near LOCOS
 2... $         Name of TIF file.This file is LOCOS simulation from TS4.
 3... ASSIGN    NAME=MESHFILE C.VALUE="STRESS1.TIF"

 4... $         Load TIF file
 5... MESH      IN.FILE=@MESHFILE TIF
 6... $         Convert nitride1 to a region electrode
 7... ELECTRODE NAME=CATHODE REGION=NITRIDE1
 8... $         Doping profile for diode
 9... PROFILE   P-TYPE UNIFORM N.PEAK=1E17
10... PROFILE   N-TYPE Y.JUNC=.1 N.PEAK=2E19 X.MAX=.8
11... $         Read stress tensor from TIF file.
12... PROFILE   IN.FILE=@MESHFILE TIF OTHER=(SXX,SYY,SXY)
13... $         Surface Recombination
14... INTERFACE REGION=(SILICON1,OXIDE1) S.N=1E4 S.P=1E4
15... $         Plot of device structure
16... PLOT.2D   GRID FILL SCALE

17... $         Solve for reverse bias IV without stress
18... MODELS    CONMOB FLDMOB CONSRH
19... SYMB      CARRIERS=2 NEWTON
20... LOG       OUT.FILE=STRESS1.LOG
21... SOLVE     V(CATHODE)=0.0
22... SOLVE     V(CATHODE)=0.2
23... SOLVE     V(CATHODE)=0.4
24... SOLVE     V(CATHODE)=0.6
25... SOLVE     V(CATHODE)=1.0 ELEC=CATHODE VSTEP=1 NSTEPS=4

26... PLOT.2D   FILL SCALE TITLE="GENERATION RATE WITHOUT STRESS"
27... CONTOUR   RECOMBINATION NCONT=10 MIN.VALU=16 MAX.VALU=21 ABS LOG
  ... +         FILL C.START=13

28... $         Now solve for reverse bias IV with stress
29... MODELS    CONMOB FLDMOB CONSRH STRESS STRMOB Y.ORIENT=100
30... LOG       OUT.FILE=STRESS2.LOG
31... SOLVE     V(CATHODE)=0.0 INITIAL
32... SOLVE     V(CATHODE)=0.2
33... SOLVE     V(CATHODE)=0.4
34... SOLVE     V(CATHODE)=0.6
35... SOLVE     V(CATHODE)=1.0 ELEC=CATHODE VSTEP=1 NSTEPS=4

36... PLOT.2D   FILL SCALE TITLE="GENERATION RATE WITH STRESS"
37... CONTOUR   RECOMBINATION NCONT=10 MIN.VALU=16 MAX.VALU=21 ABS LOG
  ... +         FILL  C.START=13

38... $         PLOT IV CURVES
39... PLOT.1D   IN.FILE=STRESS1.LOG X.AXIS=V(CATHODE) Y.AXIS=I(CATHODE)
  ... +         POINTS ABS LOG TOP=1E-12
  ... +         TITLE="STRESS-INDUCED LEAKAGE CURRENT"
40... PLOT.1D   IN.FILE=STRESS2.LOG X.AXIS=V(CATHODE) Y.AXIS=I(CATHODE)
  ... +          POINTS ABS LOG UNCH COLOR=2
41... LABEL     LABEL="WITH STRESS" X=3.8 Y=2E-13 COLOR=2
42... LABEL     LABEL="WITHOUT STRESS" X=3.8 Y=1E-14
```

Figure 6-9 Output listing of the simulation input file *mdexstress*

CAUTION

When reading the stress tensor from a *TIF* file, the names of the components of the stress tensor must be given as SXX, SYY, and SXY.

The recombination velocities at the silicon-oxide interface are then set on line 14 to model interface recombination centers. A plot of the device structure is made on line 16 using a **PLOT.2D** statement and is shown in Figure 6-10.

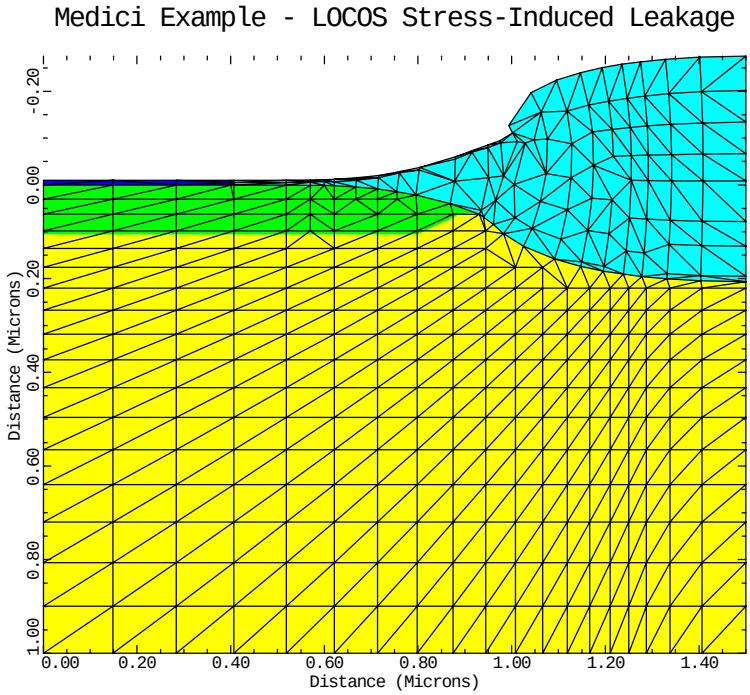


Figure 6-10 LOCOS device structure and mesh generated by the **PLOT.2D** statement on line 16 in the input file *mdexstress.inp*

Reverse Leakage Simulation Without Stress

The simulation of the leakage current without stress is begun on line 18 where some common models are selected. A two carrier solution is specified on line 19 and lines 21-25 perform a solution at different biases up to 5 V. A 2D contour plot of the net generation rate at 5 V is initiated on lines 26 and 27 and is shown in Figure 6-11a. The primary source of leakage current in silicon PN diodes is from generation in the depletion region. The main effect of stress is to alter the bandgap which should produce a significant effect on the generation rate, and hence the leakage current.

Reverse Leakage Simulation With Stress

The impact of stress on the leakage current is simulated by first specifying the various stress-related parameters on the **MODELS** statement on line 29. The three stress-related parameters that are specified are:

- **STRESS**, which turns on the stress-induced bandgap model
- **STRMOB**, which turns on the stress-induced mobility model
- **Y.ORIENT=100**, which is used to specify a <100> oriented substrate for the stress calculations

The reverse leakage is simulated for biases up to 5 V using a series of bias steps similar to the solution without stress. Lines 31-34 specify solves at discrete biases while line 35 specifies a voltage ramp from 1 to 5 V. A 2D contour plot of the resulting stress-induced generation rate is shown in Figure 6-11b The large com-

pressive stress in the silicon near the LOCOS bird's beak causes a narrowing of the bandgap and a subsequent increase in the generation rate.

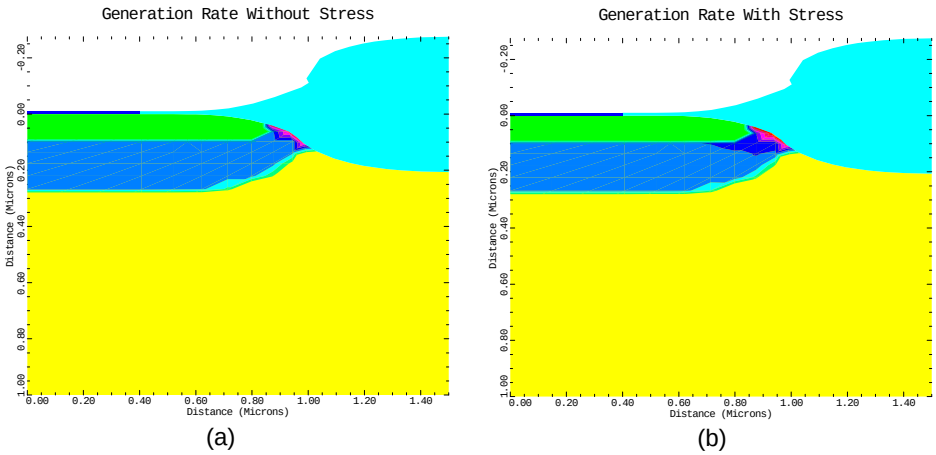


Figure 6-11 Simulated generation rate near a LOCOS isolation structure *with* and *without* stress

An IV plot of the leakage current, *with* and *without* stress, is performed on lines 39-42 with a **PLOT.1D** statement. Figure 6-12 shows that when stress is included in the simulation, the leakage current increases by a factor of two.

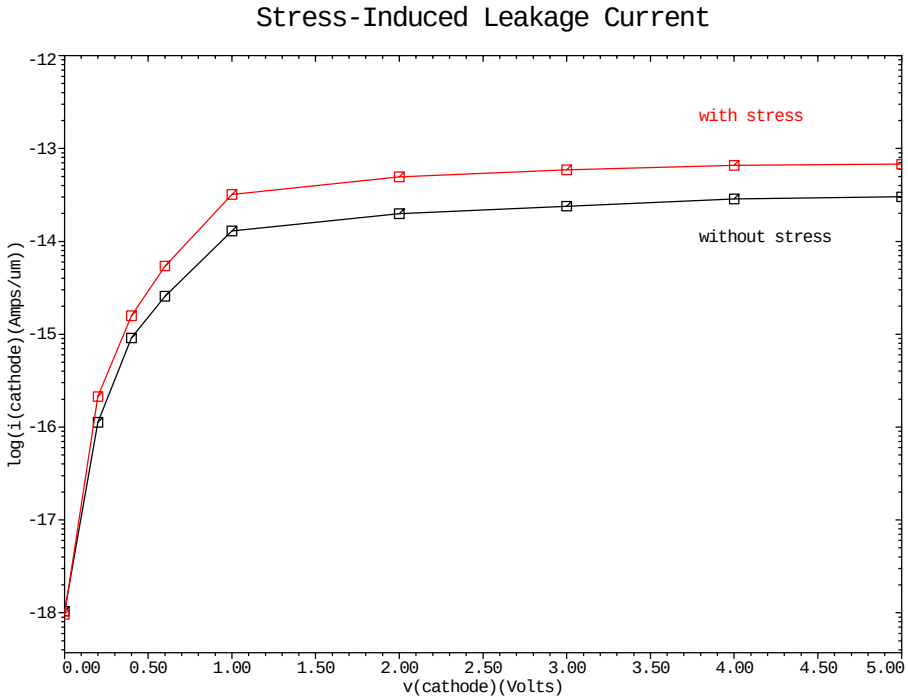


Figure 6-12 A comparison of leakage currents produced with a **PLOT.1D** statement in the input file *mdexstress.inp*

Lumped Resistance and Distributed Contact Resistance Example

This section details an example that shows the difference between using a lumped resistive element and a distributed contact resistance in a simulation. The input file *mdex4r* creates the simulation structure and then performs two separate solutions:

- A lumped resistive element attached to one of the contacts
- A nonzero contact resistance specified instead.

The results are then examined graphically.

The output associated with the execution of Medici for the input file *mdex4r* is shown in Figures 6-13 through 6-15.

Mesh Generation

The structure used in this example is created in lines 3 through 9 of the input file shown in Figure 6-13. These statements create a structure measuring 4 microns by 0.2 microns covered by a uniform mesh in both directions. The entire structure consists of uniformly doped n-type silicon with a concentration of $2 \times 10^{19} cm^{-3}$. Two electrodes are defined for the structure:

- One along the entire left edge of the device

- One along the right half of the top surface of the device

```
 1... TITLE     Avant! MEDICI Example 4R - Lumped and Contact Resistance

 2... COMMENT   Create the simulation structure
 3... MESH
 4... X.MESH    X.MAX=4.0  N.SPACES=20
 5... Y.MESH    Y.MAX=0.2  N.SPACES=10

 6... REGION    NAME=Silicon  SILICON

 7... ELECTR    NAME=Anode  LEFT
 8... ELECTR    NAME=Cathode  TOP  X.MIN=2.0

 9... PROFILE   N-TYPE  N.PEAK=2E19  UNIFORM

10... COMMENT   Attach a lumped resistance to the Cathode, obtain a
11... $         solution, and  plot results

12... CONTACT   NAME=Cathode  RESIST=50  PRINT

13... SYMB      CARR=1  NEWTON  ELEC
14... SOLVE     INIT  V(Anode)=0.1

15... PLOT.2D   BOUND  LUMPED  TITLE="Example 4R - Lumped Resistance"
16... VECTOR    J.ELEC  COLOR=2

17... COMMENT   Change to contact resistance at Cathode, obtain a
18... $         solution, and plot results
19... CONTACT   NAME=Cathode  RESIST=0.
20... CONTACT   NAME=Cathode  CON.RES=1E-6  PRINT

21... SYMB      CARR=1  NEWTON  ELEC
22... SOLVE     V(Anode)=0.1

23... PLOT.2D   BOUND  CON.RES  TITLE="Example 4R - Contact Resistance"
24... VECTOR    J.ELEC  COLOR=2
```

Figure 6-13 Output of the simulation input file *mdex4r*

Lumped Resistance Solution

Special boundary conditions to be used at contacts are specified with the **CONTACT** statement. At line 12, a $50\ \Omega\text{-}\mu m$ resistive element is attached to the cathode.

A single-carrier solution for electrons is then obtained with a 0.1V bias applied to the anode.

A vector plot of the resulting current is requested with the **PLOT.2D** and **VECTOR** statements at lines 15 and 16 of the input file. The result is shown in Figure 6-14. Note that this figure is not drawn to scale. Although a $50\ \Omega\text{-}\mu m$ resistance is attached to the contact, the contact itself has no resistance and the current crowds into the nearest corner of the electrode.

Contact Resistance Solution

Lines 19 and 20 of the input file change the boundary condition at the cathode.

- The lumped resistance is removed by specifying a value for **RESIST** of 0.
- A distributed contact resistance is specified with the **CON.RES** parameter.

The value $10^{-6}\Omega.cm^2$ is chosen so that total resistance of the contact is $50\ \Omega\text{-}\mu m$ (the same value used for the resistive element above).

A single-carrier solution for electrons is obtained with 0.1V applied to the anode.

A vector plot of the resulting current is shown in Figure 6-15. The contact now has a resistivity associated with it; the current crowding experienced above has diminished; and the current is now distributed along the entire electrode.

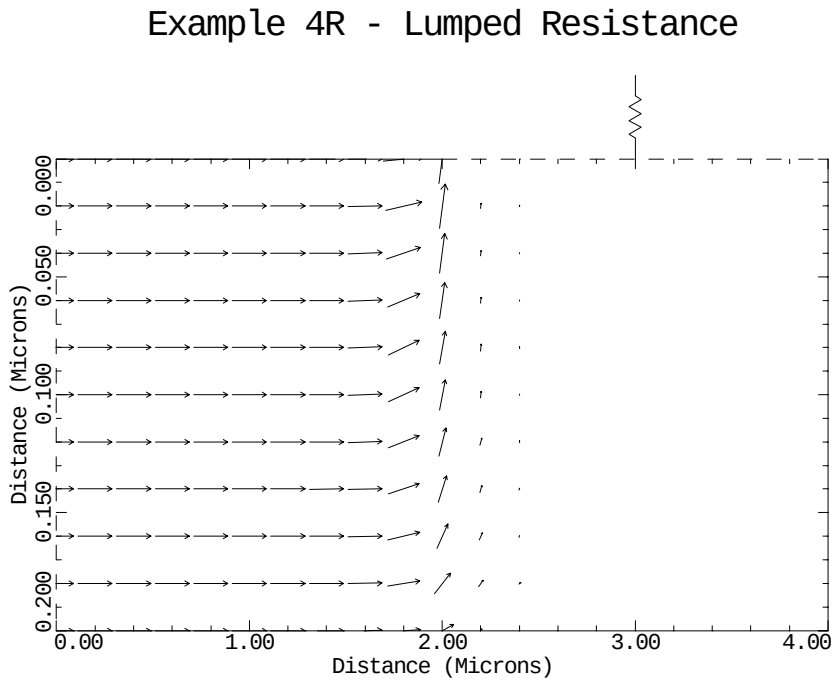


Figure 6-14 Lumped resistance from **PLOT.2D** and **VECTOR** at lines 15 and 16 in file *mdex4r*, Figure 6-13

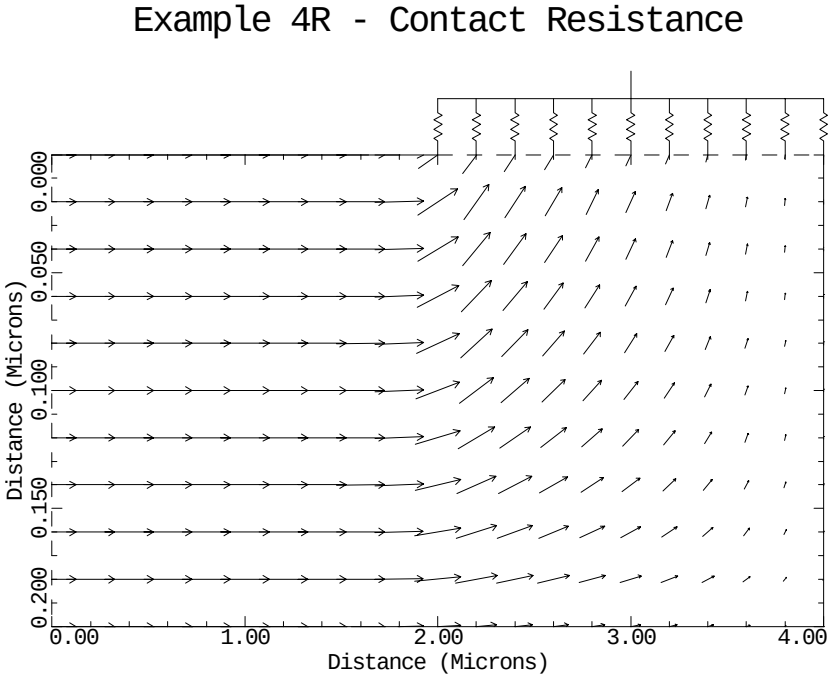


Figure 6-15 Contact resistance from **PLOT.2D** and **VECTOR** at lines 23 and 24 in file *mdex4r*, Figure 6-13

MOSFET with Inductive Load

This section simulates the turn-off characteristics of a MOSFET with a complex load impedance. Using Medici, it is possible to create a load consisting of a resistor, a capacitor, and an inductor in parallel.

This particular example emphasizes the inductive portion of the load. Expect to see the voltage at the drain of the MOSFET spike up above the power-supply voltage during turn-off.

```
 1... TITLE     Avant! MEDICI Example 4L - MOSFET With Inductive Load
 2... COMMENT   Turn-off Characteristics for MOSFET with Inductive Load

 3... COMMENT   Read in the structure created by example 1
 4... MESH      IN.FILE=MDEX1MS

 5... COMMENT   Specify the models to be used
 6... MODELS    CONMOB  PRPMOB  FLDMOB  CONSRH  AUGER

 7... COMMENT   Obtain initial solution with the MOSFET fully conducting
 8... SYMBOLIC  GUMMEL  CARRIERS=0
 9... METHOD    ICCG  DAMPED

10... SOLVE     V(Gate)=5  V(Substrate)=0  V(Source)=0  V(Drain)=5

11... SYMBOLIC  NEWTON  CARRIERS=2
12... METHOD    N.DVLIM=0.4
13... SOLVE

14... COMMENT   Attach the inductive, resistive, and capacitive
   ... +        load elements to the drain and get steady-state solution
15... CONTACT   NAME=Drain  RESISTAN=1E5  CAPACITA=5E-14  INDUCTAN=2E-5
16... SOLVE

17... COMMENT   Open up a log file to store the terminal currents
18... LOG       OUT.FILE=MDEX4LI

19... COMMENT   Increase Newton voltage update limit
20... METHOD    N.DVLIM=0.8  ITLIMIT=10

21... COMMENT   Perform the time dependent solution.
   ... +        Ramp the gate down to zero volts to turn off the MOSFET
22... SOLVE     V(Gate)=0  TSTEP=2E-10  RAMPTIME=1E-8  TSTOP=2E-8

23... COMMENT   Plot the drain voltage to show the inductive spike
24... PLOT.1D   X.AXIS=TIME  Y.AXIS=V(Drain)  POINTS  MIN=4.5  MAX=6.5
  ... +         TITLE="Example 4L - Drain Voltage During Turn Off"
```

Figure 6-16 Output of the simulation input file *mdex4l*

Mesh

Line 4 of the input file *mdex4l* (shown in Figure 6-16) reads in the mesh which was saved from example 1 (*mdex1).* This mesh has 480 grid points and 890 elements.

Models

Line 6 specifies the models to be used. These are the typical models used during simulation of a MOSFET.

Initial Device Solution

Lines 8 through 10 obtain a zero-carrier solution to be used as an initial guess for the two-carrier simulation that will follow. Bias the N-channel MOSFET in the fully conducting state, with 5 volts on the gate and 5 volts on the drain.

Lines 11 through 13 obtain a 2-carrier initial solution that will be used to start the transient simulation. Specify **N.DVLIM** equal to 0.4 volts. 0.4 volts limits the potential update during any single Newton iteration to 0.4 volts and improves the stability of the solution process. This allows the fully conducting state to be obtained in a single bias step.

Adding Lumped Elements

Line 15 attaches the resistor, capacitor, and inductor to the drain of the MOSFET. Note that while these three elements are connected in parallel, they are still in series with the 5 volt source that was originally attached to the drain. The units for the resistor and inductor are $\Omega.\mu m$ or $H.\mu m$.

To calculate the parameters **RESISTAN** and **INDUCTAN** multiply the real resistance and inductance (in ohms and Henries respectively) by the MOSFET channel width in microns.

The units for capacitance are $F/\mu m$, so to obtain the parameter **CAPACITA** divide the real capacitance (in farads) by the MOSFET channel width.

Line 13 performs the last steady-state solution. This solution sets the current in the inductor and the voltage on the capacitor to their initial values.

Line 18 opens a file to save the terminal currents (for plotting).

Transient Simulation

Line 20 increases **N.DVLIM** and reduce the maximum number of Newton iterations. Reducing Newton iterations causes the program to stop, cut back the time step, and cease to work on a solution that will never converge. This can sometimes speed up a simulation. Most time points in transient analysis converge in less than 10 iterations.

Line 22 perform the actual transient simulation. The gate bias is dropped from 5 volts to zero volts over a period of 10 nanoseconds. The entire simulation, however, lasts for 20 nanoseconds. Line 24 produces a plot of the MOSFET drain voltage versus time.

As seen in Figure 6-17, the drain voltage spikes up about one volt above the power supply voltage and then dips below 5 volts before finally settling to 5 volts.

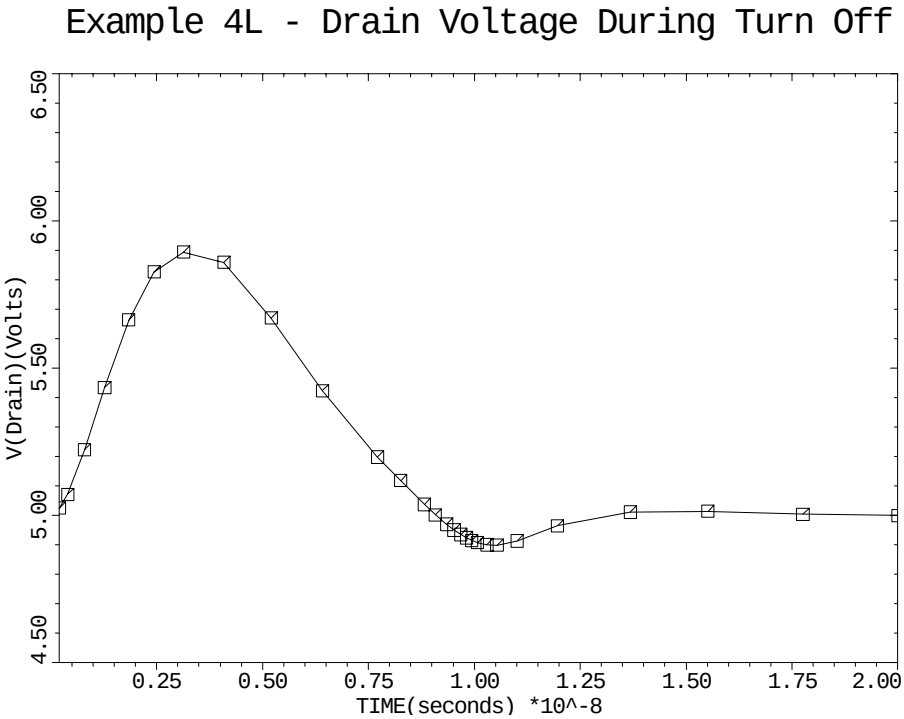


Figure 6-17 MOSFET drain voltage from **PLOT.1D** at line 24 in file *mdex4l,* Figure 6-16

Photogeneration Examples

Example Specifications

The Medici photogeneration model can be used to model the perturbations in a semiconductor device caused by many forms of radiation. Two examples are presented in this chapter:

- The first example is a back-lit solar cell that illustrates the influence of visible light on device behavior.

  The input file *mdex5* develops the simulation structure and simulates the operation of the cell when exposed to light with a wavelength 0.6 microns.
- The second example involves the simulation of single-event upset (SEU) in a static random access memory (SRAM) cell.

  The input file *mdex6* develops the simulation structure and simulates upset by a 100 MeV argon ion passing through the drain of the p-channel MOSFET in an SRAM cell.

Back-Lit Solar Cell

This section details some of the analyses that might be performed on a solar cell. The purpose of the simulation is to calculate the power-versus-load resistance curve for the cell so that the maximum power point and cell efficiency can be determined.

The structure simulated is a back-lit solar cell. The term *back-lit* is used because light enters from the back side of the wafer, away from the side with the contacts and diffusions. The entire wafer is 150 microns thick and this example simulates a narrow 10 micron wide piece. The structure is constructed using a uniformly doped p-type wafer with an impurity concentration of $10^{14} cm^{-3}$. A 6 micron wide n-type collector diffusion is used to collect the minority carriers (electrons). The

collector junction depth is 2 microns. Contacts are situated only on the top surface of the wafer.

Wavelength of Light

An important consideration in the design of solar cells is the wavelength of light used to illuminate the structure. The light incident on the silicon surface is absorbed exponentially with distance as it penetrates the crystal lattice. The distance over which the light is absorbed is characterized by an absorption coefficient (measured in units of inverse length) and is usually very wavelength dependent.

The inverse of the absorption coefficient, which is called the absorption distance, is the distance at which the incident photon flux drops to (1/e) of its initial value. In Figure 7-1, the light absorption distance in silicon is plotted as a function of wavelength.

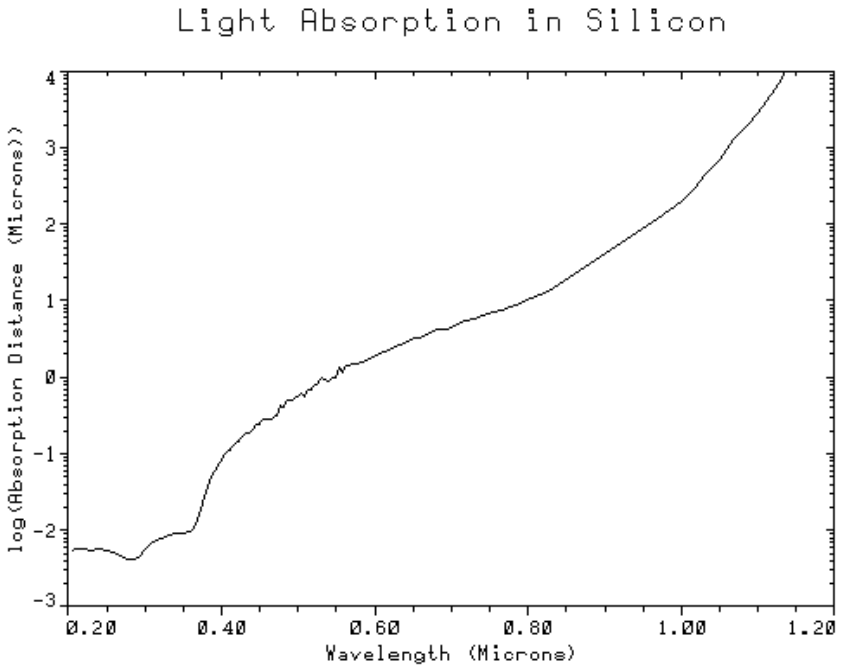


Figure 7-1 Absorption distance of light in silicon as a function of wavelength

Generation of the Simulation Structure and Solutions

The input file *mdex5* creates the simulation structure for the solar cell and simulates the steady-state characteristics. The output associated with the execution of Medici for the input file *mdex5* is shown in Figures 7-2 through 7-6.

Initial Mesh

The first step in creating the structure is the generation of a mesh. The **MESH** statement at line 3 of the input file shown in Figures 7-2 and 7-4 is used to initialize the mesh generation.

```
 1... TITLE     Avant! MEDICI Example 5 - Back-lit Solar Cell.

 2... COMMENT   Mesh Generation
 3... MESH
 4... X.MESH    WIDTH=10.0   H1=0.50
 5... Y.MESH    DEPTH=2.0    H1=0.40
 6... Y.MESH    Y.MAX=75.0   H1=0.40  H2=15.0
 7... Y.MESH    Y.MAX=150.0  H1=15.0  H2=0.30

 8... REGION    NAME=Silicon SILICON

 9... COMMENT   Electrodes:
10... ELECTR    NAME=Substrate  TOP  X.MIN=9.0
11... ELECTR    NAME=Collector  TOP  X.MAX=7.0

12... COMMENT   Substrate layer and collector diffusion
13... PROFILE   P-TYPE  N.PEAK=1E14  UNIF  OUT.FILE=MDEX5DS
14... PROFILE   N-TYPE  N.PEAK=1E17  WIDTH=6.0  JUNC=2.0  XY.RATIO=0.75

15... COMMENT   Grid refinement based on doping.
16... REGRID    DOPING  LOG  RATIO=1  SMOOTH=1  IN.FILE=MDEX5DS

17... COMMENT   Display the grid at the top and bottom on the same plot.
18... PLOT.2D   TITLE="Example 5 - Simulation MESH"
  ... +         ^MARKS  ^LABELS  X.LEN=15
19... PLOT.2D   GRID  SCALE  FILL  Y.MAX=20  TITLE=""
  ... +         X.LEN=6  X.OFF=2  ^CLEAR
20... PLOT.2D   GRID  SCALE  FILL  Y.MIN=130 TITLE=""
  ... +         X.LEN=6  X.OFF=11 ^CLEAR

21... COMMENT   Specify physical models to use.
22... MATERIAL  REGION=Silicon  TAUN0=5E-5  TAUP0=5E-5
23... MODELS    CONMOB  AUGER  CONSRH
```

Figure 7-2 First part of the simulation input file *mdex5*

Grid and Device Dimensions

A single **X.MESH** statement is used that specifies the device is 10 microns wide with a uniform grid spacing of 0.50 microns.

Three **Y.MESH** statements further define the grid:

- The first **Y.MESH** statement creates a grid section with a depth of 2 microns and a vertical grid spacing of 0.4 microns.
- The second **Y.MESH** statement gradually increases the grid spacing so that at a depth of 75 microns the grid spacing is 15 microns.
- The third **Y.MESH** statement gradually reduces the grid spacing to give a final mesh spacing of 0.3 microns at the back surface.

Fine Grid for Photogeneration

A fine grid spacing is needed at the back surface to prevent discretization error during the photogeneration process. Medici calculates the generation at nodes, and if an insufficient number of nodes are used at the back surface, the exponential decay of the generation curve is poorly approximated. This results in the generation of too many electron-hole pairs.

Referring to Figure 7-1, light with a wavelength of 0.6 microns has a 1/e decay length of 2 microns. Therefore, a fine grid spacing is only needed for approximately the bottom 4 microns of device depth.

Device Specification

It is now necessary to specify the following:

- The entire device is fabricated from silicon, which is specified with the **REGION** statement at line 8.
- The **ELECTR** statements (line 8) are used to specify the location of contacts to the structure.
- The substrate electrode (defined in line 10) only touches the top right edge of the device.
- The collector (line 11) contacts the n-type diffusion region at the top left edge of the device.

Doping

Lines 13 and 14 specify the structure doping. Line 13 specifies the entire device to have a uniform p-type concentration of $10^{14} cm^{-3}$. The file *MDEX5DS* is used to store a description of the doping profiles. This information is used later to find the impurity concentration at new nodes added to the structure as the result of a grid refinement using the **REGRID** statement. Line 14 creates the n-type collector diffusion with a peak doping of $10^{17} cm^{-3}$ and a junction depth of 2 microns.

Regrid

At line 16, the simulation grid is refined based on **DOPING**. New nodes are added if the impurity concentration differs by more than one order of magnitude (as specified with **LOG** and **RATIO**=1) over an existing triangle. Smoothing is also requested to help reduce the number of obtuse triangles that may be created and their adverse effects.

The doping information stored in the file *MDEX5DS* is used to calculate the doping at the new nodes. If *MDEX5DS* were not used, the doping at the new nodes would be calculated by interpolation from the unrefined mesh, which is likely to introduce some inaccuracies in the doping profile.

Figure 7-3 shows portions of the resulting simulation mesh.

Models

Line 22 modifies the carrier lifetimes to values that are more appropriate for solar cell simulation. Line 23 specifies the models to be used in the simulation.

Since recombination is important in solar cell operation, both **AUGER** and concentration-dependent Shockley-Read-Hall (**CONSRH**) recombination are used. The mobility also affects the diffusion of carriers, and a concentration-dependent mobility model (**CONMOB**) is used. Since the applied bias will be low, and the substrate is lightly doped, relatively small electric fields will be present and field-dependent mobility models are not needed.

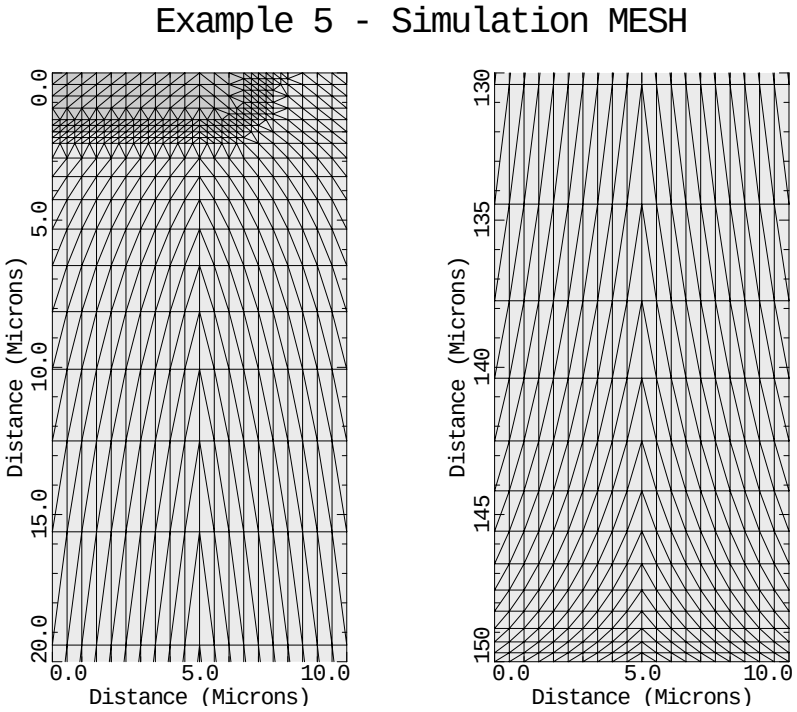


Figure 7-3 Simulation mesh from **PLOT.2D** at lines 18 through 20 in file *mdex5*, Figure 7-2

Photogeneration Rate Model

It is now time to include statements that describe the photogeneration process. The photogeneration rate G_{photo} (electron-hole-pairs/cm^3-s) as a function of distance y (microns) from the surface upon which the radiation is incident can be expressed as:

$$G_{photo} = FLUX \frac{\exp\left[-\frac{y}{Y.CHAR}\right]}{10^{-4} \cdot Y.CHAR} \qquad \text{Equation 7-1}$$

where:

- *FLUX* is the photon flux (photons/cm^2-sec)
- *Y.CHAR* is the absorption distance (microns)

The 10^{-4} factor in the denominator is a conversion factor from microns to cm. The **PHOTOGEN** statement, however, requires the photogeneration rate to be expressed in the following form (where the radial and time-dependent factors default to unity):

$$G_{photo} = \mathbf{A3} \cdot \exp(\mathbf{A4} \cdot y) \qquad \text{Equation 7-2}$$

Comparing this with the above expression, the following equivalences can be made:

$$\mathbf{A3} = \frac{10^4 \cdot FLUX}{Y.CHAR} \qquad \text{Equation 7-3}$$

$$\mathbf{A4} = \frac{-1}{Y.CHAR} \qquad \text{Equation 7-4}$$

```
24... COMMENT   Photogeneration rates (#/cm^3/sec) can be expressed
  ... +         in terms of a photon flux (#/cm^2/sec) and a
  ... +         characteristic absorption distance (microns).  Assuming
  ... +         that each photon generates one electron-hole pair, these
  ... +         can be related to parameters available on the
  ... +         PHOTOGEN statement (A3 and A4) as follows:
  ... +
  ... +           G(photo) = (1e4*FLUX/Y.CHAR)*exp[-(y/Y.CHAR)] =
  ... +                    = A3*exp[A4*y]
  ... +
  ... +            ==> A3=1e4*FLUX/Y.CHAR and A4=-1/Y.CHAR

25... COMMENT   Assign names and values for the characteristic
  ... +         absorption distance and the photon flux.
26... ASSIGN    NAME=Y.CHAR  N.VALUE=2.0
27... ASSIGN    NAME=FLUX    N.VALUE=4E17

28... COMMENT   Specify the photogeneration rate and that the photo
  ... +         source illuminates the entire back surface.
29... PHOTOGEN  A3=1E4*@FLUX/@Y.CHAR  A4=-1/@Y.CHAR
  ... +         X.START=0.  Y.START=150.  X.END=0.  Y.END=0.

30... COMMENT   Set up a log file for terminal currents.
31... LOG       OUT.FILE=MDEX5I

32... COMMENT   Perform steady state solutions to find
  ... +         the maximum power point
33... SYMBOL    NEWTON  CARRIERS=2
34... SOLVE     V(Substrate)=0.00  ELEC=Substrate  VSTEP=.05  NSTEP=6
35... SOLVE     V(Substrate)=0.32  ELEC=Substrate  VSTEP=.02  NSTEP=2
36... SOLVE     V(Substrate)=0.37  ELEC=Substrate  VSTEP=.01  NSTEP=2

37... COMMENT   Plot the cell current-vs-voltage curve.
38... PLOT.1D   X.AXIS=V(Substrate)  Y.AXIS=I(Substrate)
  ... +         TITLE="Example 5 - Cell Current vs. Voltage"
  ... +         POINTS  COLOR=2  BOTTOM=0  TOP=-7E-9  LEFT=0  RIGHT=0.4

39... COMMENT   Calculate the power generated and load impedance.
40... EXTRACT   NAME=Power  EXP="-1e7*@I(Substrate)*@V(Substrate)"
  ... +         UNITS=Watts/cm^2
41... EXTRACT   NAME=Load   EXP="-1E-7*@V(Substrate)/@I(Substrate)"
  ... +         UNITS=Ohms-cm^2

42... COMMENT   Plot the power-vs-load resistance curve.
43... PLOT.1D   X.AXIS=Load  Y.AXIS=Power  POINTS  COLOR=2
  ... +         TITLE="Example 5 - Cell Power vs. Load Resistance"
  ... +         BOTTOM=0  TOP=2E-2  LEFT=0  RIGHT=20
```

Figure 7-4 Second part of the simulation input file *mdex5*

Specifying Photogeneration Model Parameters

The **ASSIGN** statements at lines 26 and 27 of the input file *mdex5* create assigned names representing the following:

- The photon flux (*FLUX*)

  The photon flux is assumed to be 4×10^{17} photons/cm^2-sec.

- The light absorption distance (*Y.CHAR*)

  The absorption distance is chosen to be 2.0 microns corresponding to light of wavelength 0.6 microns.

The photogeneration rate is then specified with the **PHOTOGEN** statement at line 29 using these assigned names and the equivalences for **A3** and **A4** established above.

Generation Path

The **PHOTOGEN** statement also allows a line to be specified along which the carriers are generated. This line is specified in terms of its starting and ending points. Because the light is to be incident on the back surface, the following points are used:

- Starting points corresponding to the bottom of the device (**X.START**=0, **Y.START**=150)
- Ending points corresponding to the top of the device (**X.END**=0, **Y.END**=0)

Solutions File

Before solutions are obtained, the **LOG** statement is used to open a file that saves the I-V information for later plotting.

Newton's method is chosen as the solution technique on the **SYMBOL** statement. Two-carrier solutions are necessary to allow the modeling of both minority and majority carrier effects.

Lines 34 through 36 perform the steady-state solutions to calculate the collected current. The initial bias step size is 0.05 Volts. This step is reduced as the voltage approaches the open circuit voltage, V_{oc}, to allow better resolution in the plotted output. If V_{oc} is exceeded the junction current passes through zero and change sign.

In Figure 7-5, the cell's I-V characteristics are plotted. It can be seen that the short circuit photocurrent (the current with V=0) is approximately 6.14 $\times 10^{-9}$ amps. The open circuit voltage, on the other hand, can be found where the current reaches zero. From the figure, V_{oc} is approximately 0.4V.

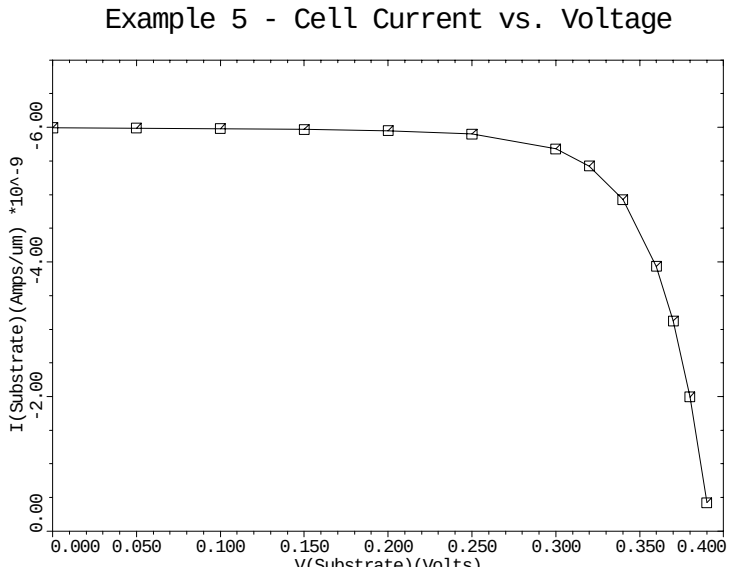


Figure 7-5 Cell current vs. voltage from **PLOT.1D** at line 38 in file *mdex5*, Figure 7-4

Total Generated Charge

The amount of charge generated by the cell can be calculated by multiplying the cell area by the photon flux and the electron charge.

$$Q = Area \cdot FLUX \cdot q \qquad \text{Equation 7-5}$$

$$= (z \times 10^{-4}\text{cm}) \left(\frac{4 \times 10^{17}}{\text{cm}^2.\text{sec}}\right) (1.602 \times 10^{-19}\text{Coul}) \left(10^{-4}\frac{\text{cm}}{\mu\text{m}}\right)$$

or

$$\frac{Q}{z} = 6.408 \times 10^{-9}\frac{\text{Coul}}{\mu\text{m}.\text{sec}} \qquad \text{Equation 7-6}$$

Medici calculates the total generated charge by summing the contributions at the nodes. This quantity can be found in the standard output under the heading *Photogeneration*.

In this simulation, Medici calculated 6.499×10^{-9} Coul/μm.sec, a difference of 0.7%. This difference is due to discretization error during the charge generation process. Even though the error is negligible, it is a good idea to check this value to be certain the grid in the generation area is fine enough.

Cell Power and Load Resistance

Lines 40 to 43 define new plot quantities with the **EXTRACT** statement.

- The first is Power, the cell power. This is simply the product of the cell voltage and the cell current.
- Line 41 calculates the effective load resistance. Because the cell voltage and current are the same as the load voltage and current, the load resistance is the cell voltage divided by the cell current.
- Line 42 scales the power and resistance by the cell area. Assuming the cell has a 1 micron depth into the simulation plane (the z dimension), the cell area is 10^{-7} cm^2 (since the device is 10 microns wide).

The cell power is proportional to the cell area, so it is necessary to divide the power by the cell area to obtain the power density. The resistance, on the other hand is inversely proportional to the cell area so it is necessary to multiply the resistance by the cell area to obtain the scaled resistance. To improve the readability of the plots, the resistance and power were negated to yield positive quantities.

Figure 7-6 shows a plot of power versus resistance for the cell. The peak power (18 mWatts/cm^2) is generated at a load resistance of about 6 .Ωcm^2.

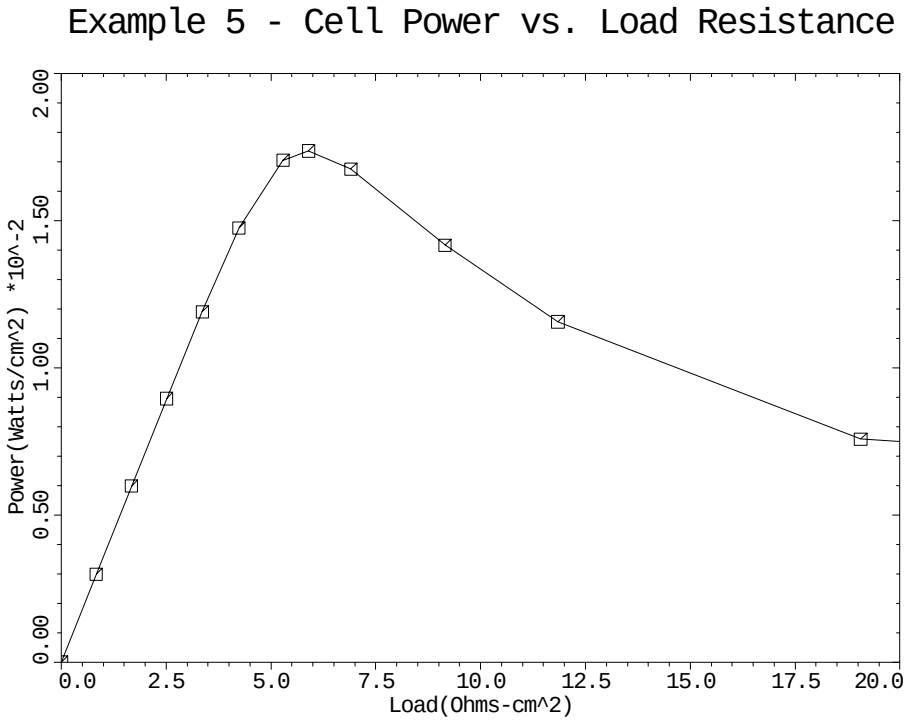


Figure 7-6 Cell power vs. load resistance from **PLOT.1D** at line 43 in file *mdex5*, Figures 7-2 and 7-4

Single-Event Upset of a SRAM Cell

The use of Medici is illustrated by going through some of the analyses that might be performed to study single-event upset of a SRAM cell.

The input file *mdex6* develops the simulation structure and simulates upset by a 100 MeV argon ion passing through the drain of the p-channel MOSFET in an SRAM cell.

The input file *mdex6h* then plots the internal hole and potential distributions at various times after the ion strike.

Overview

Single-event upset occurs when an energetic particle (ion) strikes a memory circuit such as a DRAM or SRAM cell and causes the stored information to be lost. For this simulation, assume the following:

- The drain of one of the two p-channel MOSFETs in a six transistor SRAM cell is struck by a single 100 MeV argon ion.
- The MOSFET is in an OFF state (which it must be to be upset sensitive), that the drain is at a potential of -3.0V, and that the n-type substrate is grounded (0 Volts).

These voltages reverse bias the drain-substrate junction. During the upset process, the conductive charge track generated by the passage of the ion temporarily short circuits the drain-substrate junction and pulls the drain up to the higher potential of the substrate. If the drain reaches the substrate potential (0 Volts in this case), the SRAM cell latches into the inverted state and upset occurs. (Refer to the section on circuit analysis for more information on this topic.)

- The p-channel MOSFET being considered is fabricated in an n-type epitaxial layer on top of an n-type substrate. The relevant process parameters are indicated in the following table:

```
Process Parameter            Value
-------------------------  ----------
epitaxial layer thickness    2.0 um
epitaxial layer doping       1e17/cm^3
substrate doping             5e18/cm^3
p-channel junction depth     0.3 um
p-channel peak S/D doping    1E20/cm^3
power supply voltage        -3.0v
```

Geometric Symmetry

The first important effect to consider is the geometry of the problem. The charge column generated as the ion passes through the device shows cylindrical symmetry. Assuming that the ion enters the semiconductor at normal incidence and places the charge column along the z axis of a cylindrical coordinate system, then full three-dimensional accuracy can be obtained by only simulating the radial (r) and depth (z) dependence.

Limitation with Medici Simulation

Unfortunately these arguments do not apply to the rectangular MOSFET drain diffusion or the MOSFET itself. These can not be modeled in cylindrical coordinates. Instead, this example models the square drain diffusion as a circular disk, and does not model the MOSFET at all. Converting the square drain to a circular disk is a good assumption. If the MOSFET is to be modeled, as well as the charge column, a full three-dimensional simulation (such as *Avant!* TCAD's Davinci) is needed.

Generation of the Simulation Structure and Transient Solutions

The input file *mdex6* creates the structure and performs a transient simulation to analyze the device behavior after a 100 MeV argon ion passes through the device. The output associated with the execution of Medici for the input file *mdex6* is shown in Figures 7-7 through 7-11.

The structure created for this analysis consists of a p+ drain diffusion in an n-type epitaxial layer with an n+ substrate. Cylindrical coordinates are used. It is assumed that the ion path is along the cylindrical coordinate *z*-axis (*r*=0).

The device structure is created at lines 4 through 17 of the input file *mdex6* shown in Figures 7-7 and 7-9.

```
 1... TITLE     Avant! MEDICI Example 6 - Single Event Upset Simulation
 2... COMMENT   P+/EPI/N+ Structure Using Cylindrical Coordinates

 3... COMMENT   Define a non-uniform mesh using cylindrical coordinates
  ... +         (the x-direction corresponds to the radial direction).
  ... +         Put the finest grid along the r=0 column and at the
  ... +         junction.
 4... MESH      CYLINDRI
 5... X.MESH    WIDTH=3.0  H1=0.02  H2=0.30
 6... Y.MESH    DEPTH=0.3  H1=0.10
 7... Y.MESH    DEPTH=3.7  H1=0.10  H2=0.50

 8... REGION    NAME=Silicon  SILICON

 9... COMMENT   Electrodes:
10... ELECTR    NAME=Drain  TOP  X.MAX=1.0
11... ELECTR    NAME=Substrate  BOTTOM

12... COMMENT   Profiles for the Epi layer, N+ substrate,
  ... +         and P+ diffusion
13... PROFILE   N-TYPE  N.PEAK=1.0E17  UNIF  OUT.FILE=MDEX6DS
14... PROFILE   N-TYPE  N.PEAK=5.0E18  Y.MIN=2.0  DEPTH=2.0  Y.CHAR=0.1
15... PROFILE   P-TYPE  N.PEAK=1.0E20  JUNC=0.3
  ... +         WIDTH=1.0  XY.RATIO=0.75

16... COMMENT   Grid refinement based on doping.
17... REGRID    DOPING  LOG  RATIO=1  SMOOTH=1  Y.MAX=1.0
  ... +         IN.FILE=MDEX6DS  OUT.FILE=MDEX6MS

18... PLOT.2D   GRID  TITLE="Example 6 -  Mesh"  SCALE  FILL

19... COMMENT   Specify physical models to use
20... MODELS    CCSMOB  FLDMOB  CONSRH  AUGER  BGN

21... COMMENT   Calculate a steady state solution with a reverse bias of
  ... +         of 3.0 volts.  Perform a zero carrier solution to use
  ... +         as an initial guess for the full two carrier solution
  ... +         below.
22... SYMBOL    CARR=0
23... METHOD    DAMPED
24... SOLVE     V(Drain)=-3.0
```

Figure 7-7 First part of the simulation input file *mdex6*

Mesh Generation

The first step in creating the device structure is to generate an initial mesh. The mesh generation is initiated with the **MESH** statement at line 4.

Cylindrical Coordinates

The parameter **CYLINDRI** indicates that cylindrical coordinates are to be used. In this case, the **X.MESH** statements are used to define the grid placement in the cylindrical radial direction and the **Y.MESH** statements are used to define the grid placement in the cylindrical z-direction.

Structure Definitions

The **X.MESH** and **Y.MESH** statements specify spacing for different areas of the grid.

The **X.MESH** statement at line 5 creates a grid section that is 3 microns wide. A grid spacing of 0.02 microns (**H1**) is used near the left side of the device where the charge column is placed.

This fine grid spacing assures that the charge column is adequately resolved. The grid is gradually expanded to a spacing of 0.3 microns (**H2**) at the right edge of the device.

The **Y.MESH** statement at line 6 specifies a 0.1 micron grid spacing at the top of the structure (near the drain-substrate junction). The second **Y.MESH** statement gradually expands the grid spacing to 0.5 microns at the bottom of the structure.

The **REGION** statement defines the entire device to be silicon. The **ELECTR** statements place the p+ drain contact at top left side of the structure and the substrate contact along the bottom edge of the device.

Lines 13 through 15 specify the structure doping. Line 13 generates the n-type epitaxial layer, which extends through the entire device. Line 14 creates the n+ substrate region of the device. In this case, the substrate starts at a depth of 2.0 microns. Line 15 creates the drain diffusion. The drain junction is placed at a depth of 0.3um.

Line 17 performs a grid refinement based on doping to add more nodes in the vicinity of the junction.

Figure 7-8 shows the resulting mesh.

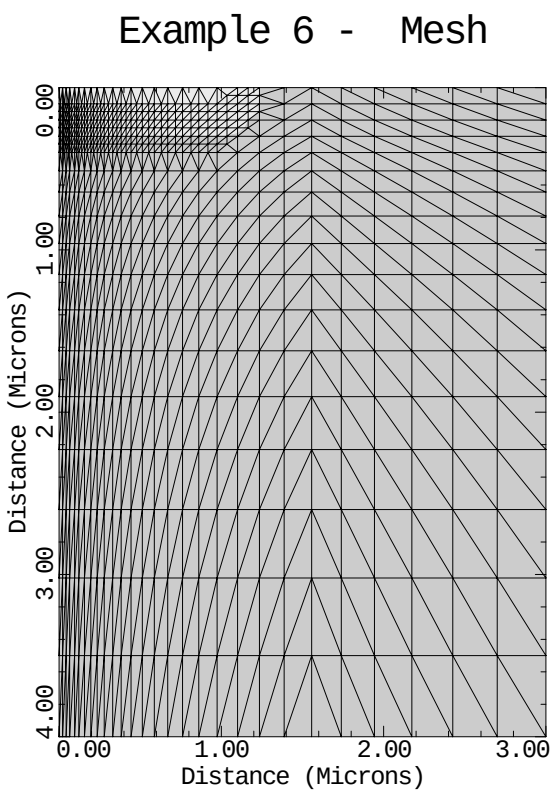


Figure 7-8 Final mesh from **PLOT.2D** at line 18 in file *mdex6*, Figure 7-7

Models

Line 20 specifies the physical models used. Due to the large carrier concentrations present in the charge column, the carrier-carrier scattering mobility model

(**CCSMOB**) is used. **CCSMOB** also includes the effects of doping and temperature on mobility.

Since the drain-substrate is reverse biased by 3V, high electric fields exist in the deletion region of the junction. Because of this the field dependent mobility model (**FLDMOB**) should also be used.

Since recombination of the carriers is important, both the concentration dependent Shockley-Read-Hall and Auger recombination models are activated. Finally, since the pn-junction is a bipolar device, the band-gap-narrowing (**BGN**) model is also used.

Initial Solution

Lines 22 through 24 generate an initial solution to be used to start the transient simulation. Assume that the MOSFET drain (the p-type region) is at the power supply potential (-3V), and that the n-type substrate is grounded.

Since the junction is reverse-biased and little current is flowing (before the ion hits) a zero-carrier solution is sufficient. **DAMPED** is specified on the **METHOD** statement since it improves the convergence of the Poisson solution.

Boundary Conditions

Line 26 specifies that lumped element boundary conditions are to be used. Applying a fixed potential directly to the P+ diffusion is an inappropriate choice of boundary conditions since the drain potential of the MOSFET changes from -3 to 0 volts during the upset process.

Using a resistor and capacitor, however, the effects of an attached circuit can be included. Included in this example is an effective resistance attached to the drain of 1000 Ohms and an effective capacitance of 10^{-13} Farads.

Solution

In line 28 a **LOG** statement is used to save the terminal currents and voltages for plotting.

Line 30 switches to a full Newton 2-carrier solution. The Newton method must be used for time dependent solutions. The Newton method is also required if lumped elements are attached to the device terminals. Line 31 calculates the initial steady-state solution to be used to start the transient simulation.

Generation Track

The **PHOTOGEN** statement at line 33 specifies the charge track generated by the ion. The ion enters at the upper left corner (**X.START**=0.0, **Y.START**=0.0) and exits at the lower left corner (**X.END**=0.0, **Y.END**=4.0).

Note:

It is important to remember that when using cylindrical coordinates, the charge track must be placed along the cylindrical z-axis. Placement of the charge track at other locations results in rings or cones of charge being generated.

The charge column is assumed to have a characteristic 1/e radius of 0.2 microns as specified by **DCHR**=0.2. This radius is measured perpendicular to the specified line segment.

```
25... COMMENT   Attach a 1000 Ohm resistor and a 100 fF capacitor to the
  ... +         Drain. These simulate the loading of an external circuit.
26... CONTACT   NAME=Drain  RESIS=1.0E3  CAPAC=1.0E-13  PRINT

27... COMMENT   Specify a log file for storing the terminal data
28... LOG       OUT.FILE=MDEX6I

29... COMMENT   Switch to Newton and two carriers and solve for the
  ... +         time=0 reverse bias solution.
30... SYMBOL    NEWTON  CARRIERS=2
31... SOLVE

32... COMMENT   Create the ion track with a PHOTOGEN statement.  The
  ... +         linear energy transfer data vs. depth for a 100 MeV argon
  ... +         atom incident on silicon is read from the file mde6let.

33... PHOTOGEN  X.START=0.0  X.END=0.0  Y.START=0.0  Y.END=4.0  DCHR=0.2
  ... +         T0=3.0E-12  TC=1.5E-12 LETFILE=mde6let  PC.UNITS GAUSS

34... COMMENT   Simulate the 1st 100 picoseconds of the
  ... +         transient response
35... SOLVE     TSTEP=0.5E-12  TSTOP=100E-12  OUT.FILE=MDE6S01

36... COMMENT   Plot the terminal characteristics.
37... PLOT.1D   X.AXIS=TIME  Y.AXIS=I(Drain)
  ... +         POINTS  BOTTOM=-8E-3  COLOR=2
  ... +         TITLE="Example 6 - Drain Current"
38... PLOT.1D   X.AXIS=TIME  Y.AXIS=V(Drain)  POINTS  TOP=-2.2  COLOR=2
  ... +         TITLE="Example 6 - Drain Voltage"
```

Figure 7-9 Second part of the simulation input file *mdex6*

Generation Waveform

Since **GAUSS** is specified, the charge is generated over a period of about 6 picoseconds using a Gaussian waveform. The Gaussian has a 1/e characteristic time of 1.5 picoseconds, as specified by **TC**=1.5e-12, and the peak of the Gaussian occurs at 3.0 picoseconds as specified by **T0**=3.0e-12.

The Gaussian is normalized by its integral over time. This causes the total charge generated over the duration of the pulse to remain constant, even if **TC** or **T0** is altered.

Generation Rate vs. Depth of Penetration

The generation rate versus depth of penetration of the ion is read from the file *mde6let*. This file contains the following data[1]:

```
/ Linear Energy Transfer (LET) data vs. depth for a 100 MeV
/ argon atom incident on a silicon wafer.
/
/ Depth (um)  LET (pCoul/um)
  0.0         0.17
  5.16        0.18
  9.86        0.185
  14.2        0.190
  19.1        0.177
  22.1        0.15
  23.0        0.00
```

1.This data corresponds to a 100 MeV argon ion and was obtained from the tables found in "The Stopping and Ranges of Ions in Matter" by J. F. Ziegler, J. P. Biersack, and U. Littmark, Pergamon Press, 1985.

The slashes (/) indicate comment lines, which are ignored. The depth is measured from the starting point of the line segment (**X.START**, **Y.START**) and is measured parallel to the line. Only the first two entries in this file are significant since the structure being simulated is only 4.0 microns deep. The final parameter on the **PHOTOGEN** statement, **PC.UNITS**, specifies that the data in file *mde6let* is in units of picoCoulombs per micron instead of electron-hole pairs/cm^3.

Transient Solution

In line 35 the transient solution is performed. An initial time step of 0.5 picoseconds is used. Medici calculates all subsequent time steps based on a local truncation error criteria. The total simulation is 100 picoseconds long. This is sufficient to resolve the drift component of the charge collection process. The diffusion charge collection, however, continues for a longer period.

Each time step generates its own solution file. The solution from the first time point is stored in *MDE6S01*. File names to store subsequent solutions are obtained by incrementing this name. The second time-step is stored in *MDE6S02*, the third in *MDE6S03*, and so on.

The terminal current obtained as a result of this simulation is plotted in Figure 7-10. The large spike (approximately 6.8 mAmp/micron) is due to drift collection. The drift collection process is quickly extinguished (at about 40 psec) as all the charge in the depletion region is collected.

After this time, the diffusion collection continues, as is evident from the slowly decaying tail. This tail continues for several nanoseconds.

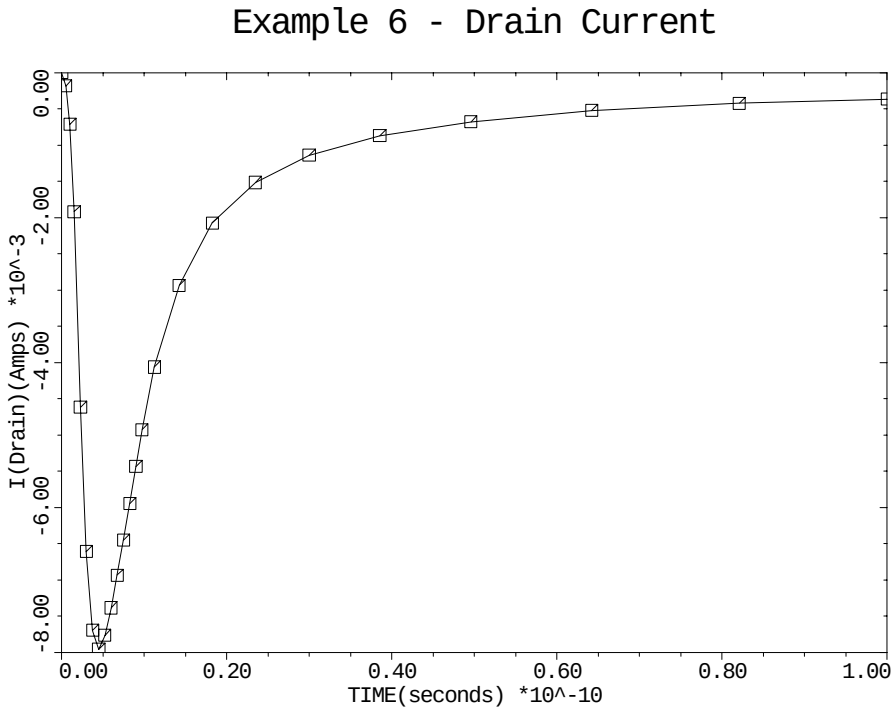


Figure 7-10 Drain current from **PLOT.1D** at line 37 in file *mdex6*, Figures 7-7 and 7-9

The terminal voltage is plotted in Figure 7-11. This plot can be used to determine if the SRAM cell upsets. The figure shows that the voltage reaches a maximum of about -2.4 volts and slowly starts to return to -3.0. Since the voltage did not reach the high state (0.0V), the cell would not upset with this ion. If a heavier ion had been used, upset would be more likely to occur.

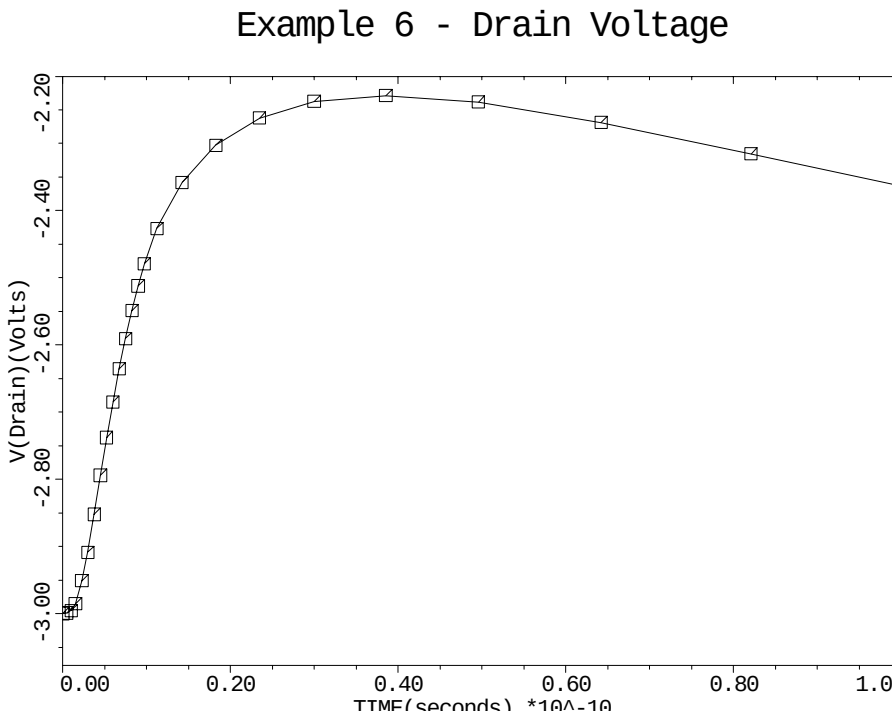


Figure 7-11 Voltage current from **PLOT.1D** at line 38 in file *mdex6,* Figure 7-9

Examination of Internal Device Behavior

The mesh and solution files created and saved by the input file *mdex6* are read by the input file *mdex6h* and are used to examine the internal device behavior at various times after the ion strike. Figures 7-12 through 7-16 contain the output associated with the execution of Medici for the input file *mdex6h*.

Plot Generation

The input file *mdex6h* shown in Figure 7-12 uses an input statement loop to read in four solutions corresponding to four simulation times (1.5 psec, 5.2 psec, 14 psec, 100.0 psec). For each solution, a filled contour plot of hole concentration is generated. Potential contours are plotted as solid lines on top of this.

```
  1... TITLE     Avant! MEDICI Example 6H - Single Event Upset Simulation
  2... COMMENT   Plot hole concentration contours and potential contours
   ... +         at various simulation times

  3... COMMENT   Read in simulation structure
  4... MESH      IN.FILE=MDEX6MS

  5... LOOP      STEPS=4
  6...   ASSIGN    NAME=FILE  C1=MDE6S03  C2=MDE6S08  C3=MDE6S14
   ...   +         C4=MDE6S24
  7...   ASSIGN    NAME=TIME  C1=1.5      C2=5.2      C3=14
   ...   +         C4=100.0

  8...   COMMENT   Load a solution file
  9...   LOAD      IN.FILE=@FILE

 10...   COMMENT   Plot internal hole and potential distributions
 11...   PLOT.2D   BOUND  TITLE="Holes & Potential, "@TIME" psec"  SCALE
 12...   CONTOUR   HOLE  LOG  FILL  MIN=4.0  DEL=2.0
 13...   CONTOUR   POTENTIA  MIN=-4.0  DEL=0.5
 14... L.END
```

Figure 7-12 Output of the simulation input file *mdex6h*

Evolution of the Charge Column

Referring to Figure 7-13, the filled contours show the path of the charge column down the left edge of the device. Since **LOG** and **MIN**=4 were specified on the

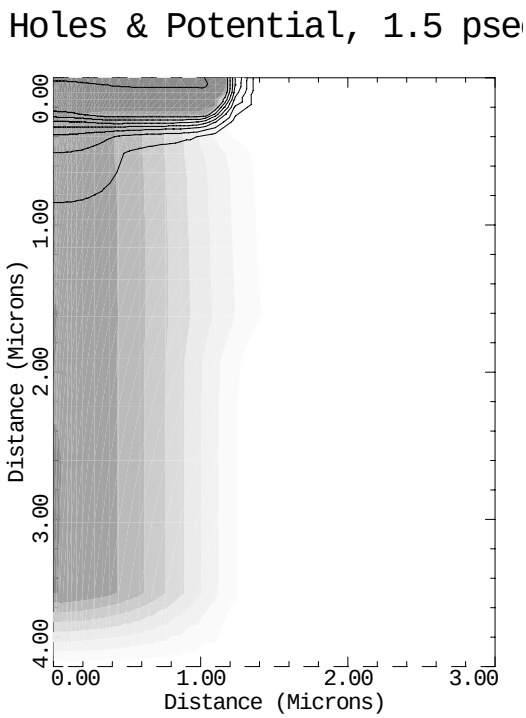


Figure 7-13 Holes and potential at 1.5 psec, generated in the first pass of the loop from **PLOT.2D** and **CONTOUR** at lines 11 through 13 in file *mdex6h*, Figure 7-12

CONTOUR statement at line 12, the white (non-filled) area indicates a region where the hole concentration is less than 10^4 cm^{-3}. Each change of shade indicates a factor of 100 increase in hole concentration.

It can be seen, by examining the sequence of plots, that the charge column widens by the outward diffusion of carriers. The charge column also pinches off at the junction due to collection of charge from the depletion region, and at 100 psec a large "body" of uncollected charge can be seen sitting under the junction.

It can also be seen that the column does not spread as rapidly within the substrate layer. This is due to a reduction in carrier mobility due to the heavily doped material.

Potential Funneling

A *funneling* behavior is evident by examining the potential contours in the series of plots. Before the ion strikes, the equipotentials are parallel to the junction.

At 1.5 psec (Figure 7-13), the equipotentials begin to extend into the substrate due to the voltage drop along the charge column. The funneling reaches its peak at 5.2 psec (Figure 7-14). At 13.5 psec (Figure 7-15), the funnel starts to collapse and the drift current starts to decrease as charge is swept from the depletion region. At 100 psec (Figure 7-16), the depletion region is effectively restored and only diffusion charge collection is occurring.

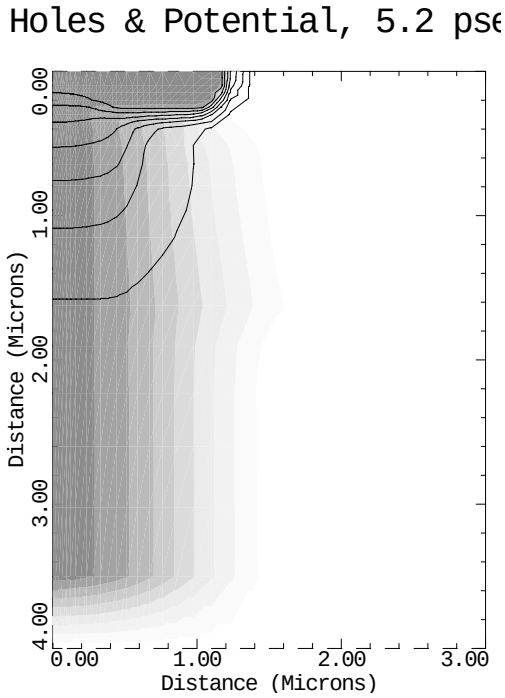


Figure 7-14 Holes and potential at 5.2 psec, generated in the second pass of the loop from **PLOT.2D** and **CONTOUR** at lines 11 through 13 in file *mdex6h*, Figure 7-12

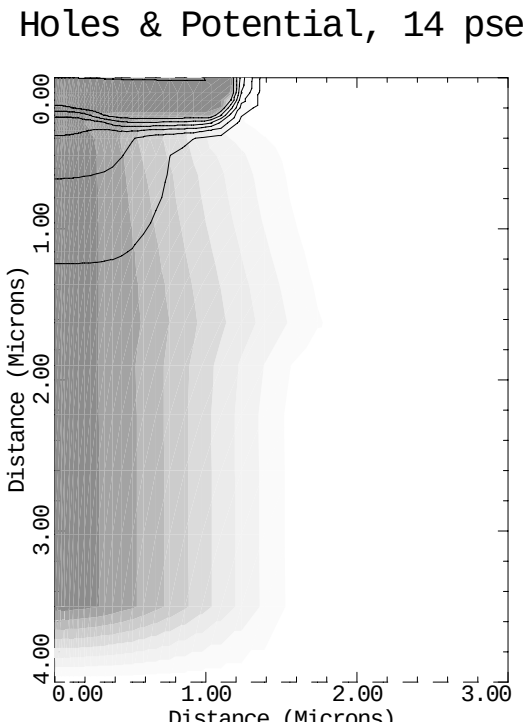


Figure 7-15 Holes and potential at 14 psec, generated in the third pass of the loop from **PLOT.2D** and **CONTOUR** at lines 11 through 13 in file *mdex6h,* Figure 7-12

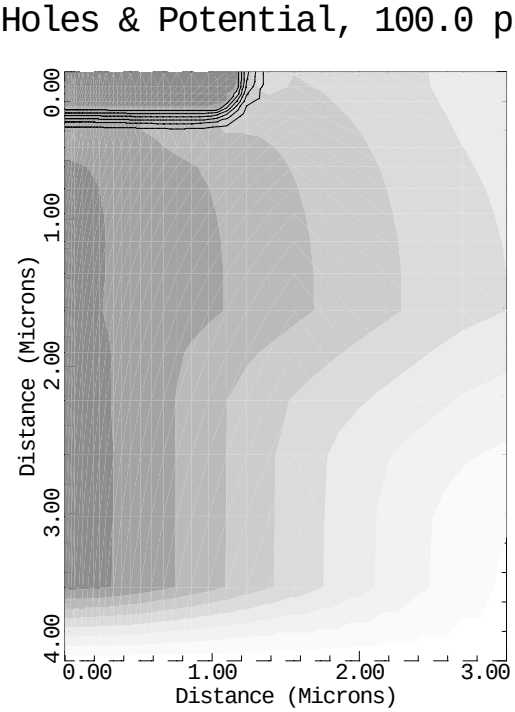


Figure 7-16 Holes and potential at 100.00 psec, generated in the fourth pass of the loop from **PLOT.2D** and **CONTOUR** at lines 11 through 13 in file *mdex6h* shown in Figure 7-12

Template Examples

Example Specifications

This chapter contains three examples that illustrate the use of parameterized template files for the automatic creation of standard MOS and bipolar structures. The examples also illustrate the use of parameterized template files for the automatic simulation of standard bias sequences. The examples create the following:

- A 1.5 micron n-channel LDD MOSFET and then simulates its drain characteristics.
- A 3.0 micron p-channel MOSFET and then simulates its gate characteristics.
- An NPN bipolar device and then simulates its forward-active device characteristics.

A complete discussion of the available template files is given in Appendix A.

N-Channel LDD MOSFET Example

This example illustrates the use of parameterized template files for the automatic creation of a 1.5 micron n-channel LDD MOSFET and the automatic simulation of its drain characteristics. The output for this example is shown in Figures 8-1 through 8-9.

Templates Used

The input file *mdex7n* uses **`CALL`** statements to enter the parameterized template files *mosdef0*, *mosstr1*, and *mosdrn0* into the input.

Suppressing Content Listing

The **`^PRINT`** parameter is specified on the **`CALL`** statements so that the contents of the parameterized files are used but not printed to the standard output.

MOS Definition Template

Since it is desired to create an MOS device, the input file shown in Figure 8-1 begins by using a **CALL** statement to enter the file *mosdef0* into the input. This file contains default values for all parameters used by the MOS templates.

The default parameters in *mosdef0* are for a 1 micron gate n-channel LDD MOSFET with the structure doping defined by analytic profiles. See Appendix A for a complete description of the available parameters and their default values.

```
  1... TITLE     Avant! MEDICI Example 7N - MOS Template File Example
  2... COMMENT   Drain Characteristics for an LDD Structure

  3... COMMENT   Load the default parameters defining the
   ... +         structure and biases
  4... CALL      FILE=mosdef0 ^PRINT

 59... COMMENT   Modify the parameters to define
   ... +   a 1.5 micron n-channel LDD
 60... ASSIGN    NAME=TRANTYPE  C.VALUE=NMOS
 61... ASSIGN    NAME=FILE      C.VALUE=N1
 62... ASSIGN    NAME=LGATE     N.VALUE=1.5

 63... COMMENT   Create the structure using template "mosstr1"
 64... CALL      FILE=mosstr1 ^PRINT

236... COMMENT   Specify parameters for obtaining Vgs=1
   ... +   and Vgs=3 drain curves
237... ASSIGN    NAME=VB0     N.VALUE=0.0
238... ASSIGN    NAME=VG0     N.VALUE=1
239... ASSIGN    NAME=VGSTEP  N.VALUE=2
240... ASSIGN    NAME=NGSTEP  N.VALUE=2
241... ASSIGN    NAME=VD0     N.VALUE=0.0
242... ASSIGN    NAME=VDSTEP  N.VALUE=0.25
243... ASSIGN    NAME=NDSTEP  N.VALUE=13

244... COMMENT   Generate the solutions and plot results
245... CALL      FILE=mosdrn0 ^PRINT
```

Figure 8-1 Output of the simulation input file *mdex7n*

Default Changes

In this example, the only desired change from the default device is to increase the gate length to 1.5 microns. This is accomplished with the **ASSIGN** statement at line 62.

A value is also assigned to the name *FILE*, which is used as a prefix for the names of any output files that are generated by the templates.

MOS Structure Template

The program is now ready to call a template that actually creates the device structure. From Appendix A: Template Files, there are two choices for templates that create MOS devices: *mosstr0* and *mosstr1*. In this example, *mosstr1* is chosen.

Regrid

This template uses the **REGRID** statement to successively refine a coarse initial mesh until it is fine enough for creating solutions. This type of template tends to work best for short-channel devices.

Although this example is using analytic impurity profiles, this template is also a good choice if the impurity profiles come from data files. This is because the

REGRID approach to mesh refinement does not require a knowledge of structure parameters such as junction depths in advance in order to correctly allocate the grid.

Regrid Plots

Figures 8-2 through 8-8 contain the output associated with the call to the template file *mosstr1* at line 64.

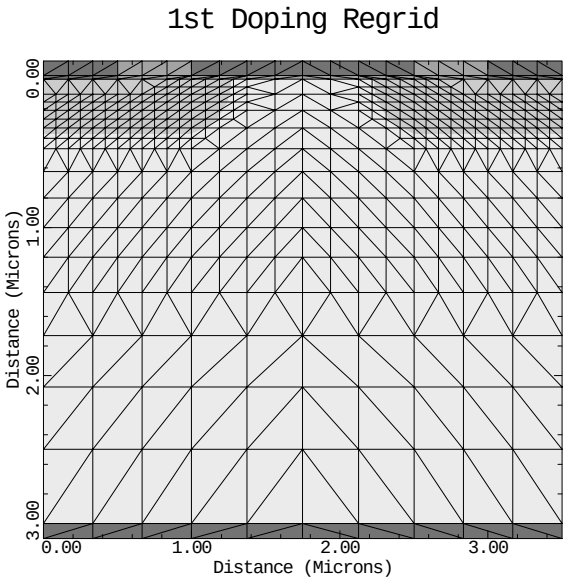


Figure 8-2 Mesh after first doping regrid from **CALL** to template file *mosstr1* at line 64 in file *mdex7n*, Figure 8-1

Doping Plots

Figures 8-2 and 8-3 show the results of regrids based on doping which are used to refine the mesh where large variations of impurity concentration occur (for example, at junctions).

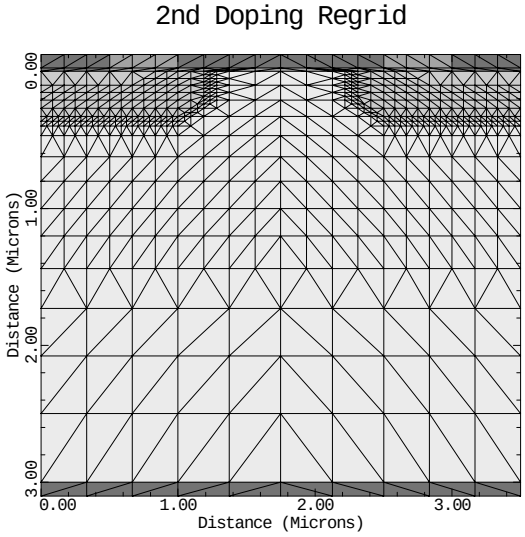


Figure 8-3 Mesh after second doping regrid from **CALL** to template file *mosstr1* at line 64 in the file *mdex7n*, Figure 8-1

Potential Plots

Figures 8-4 through 8-6 show the results of regrids based on potential which are used to refine the mesh in the channel region.

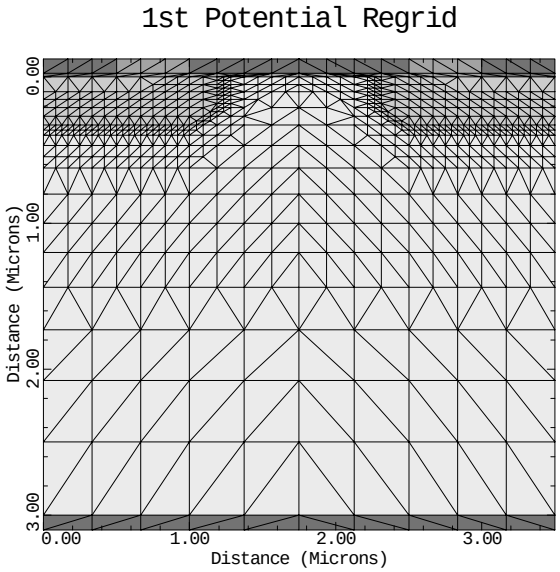


Figure 8-4 Mesh after first potential regrid from **CALL** to template file *mosstr1* at line 64 in file *mdex7n*, Figure 8-1

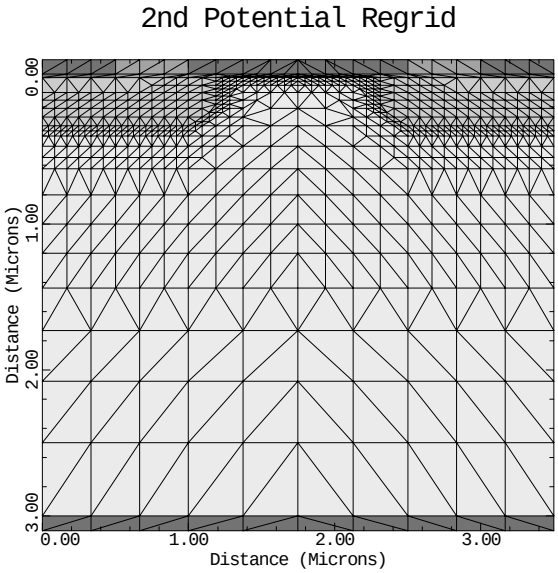


Figure 8-5 Mesh after second potential regrid from **CALL** to template file *mosstr1* at line 64 in file *mdex7n,* Figure 8-1

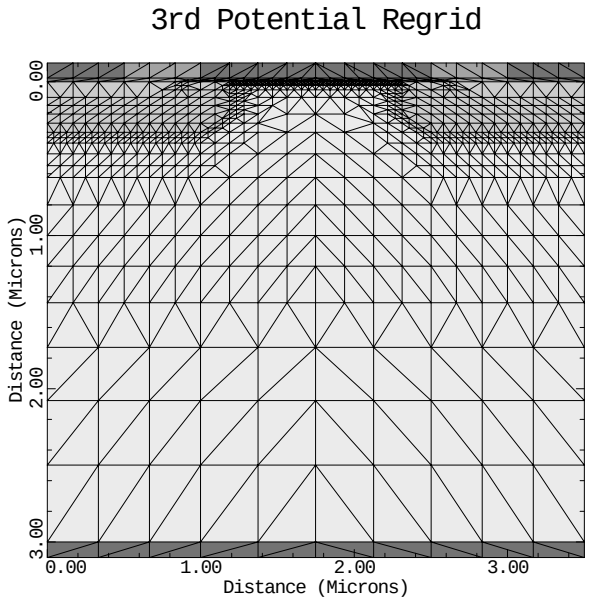


Figure 8-6 Mesh after third potential regrid from **CALL** to template file *mosstr1* at line 64 in file *mdex7n*, Figure 8-1

Impurity Concentration

Two-dimensional and one-dimensional plots of impurity concentration for the final structure are shown in Figures 8-7 and 8-8.

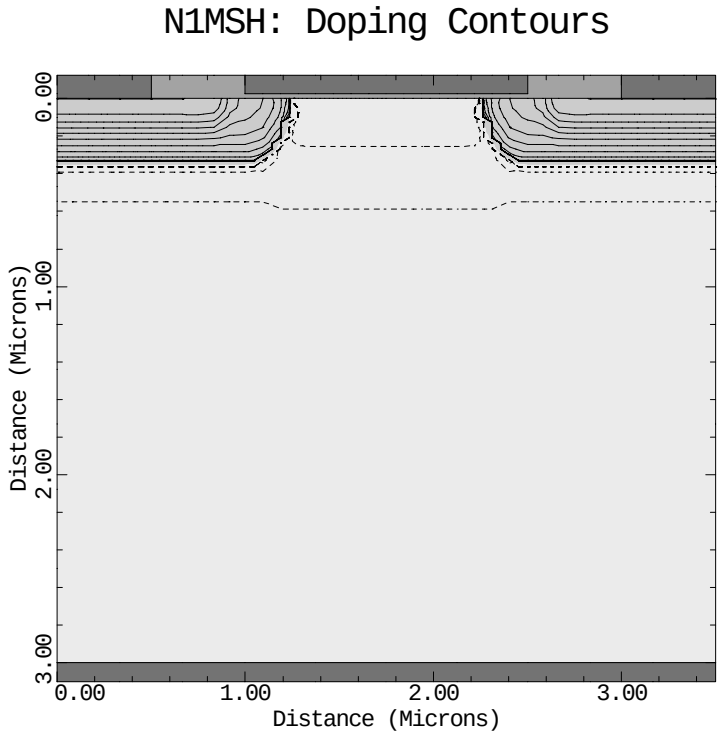


Figure 8-7 Doping contours from **CALL** to template file *mosstr1* at line 64 in file *mdex7n*, Figure 8-1

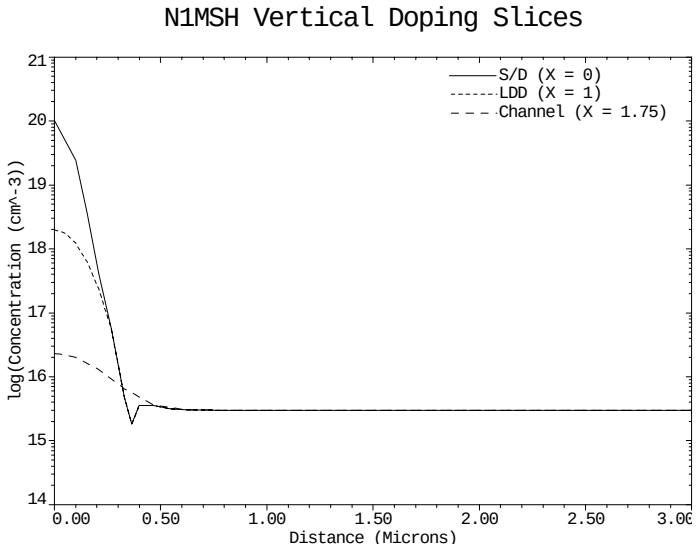


Figure 8-8 ID doping profiles from **CALL** to template file *mosstr1* at line 64 in file *mdex7n*, Figure 8-1

Template

Having created the structure, a set of drain characteristics can now be obtained. Before doing so, values should be assigned to various parameters for describing the range of biases to use. The assignments in lines 237 through 243 are such that $V_{Substrate}$=0V, V_{Gate}=1V and 3V, and V_{Drain}=0V to 3V in 0.25V increments. The call to the template file *mosdrn0* at line 245 performs the desired simulations. The output associated with this **CALL** is shown in Figure 8-9.

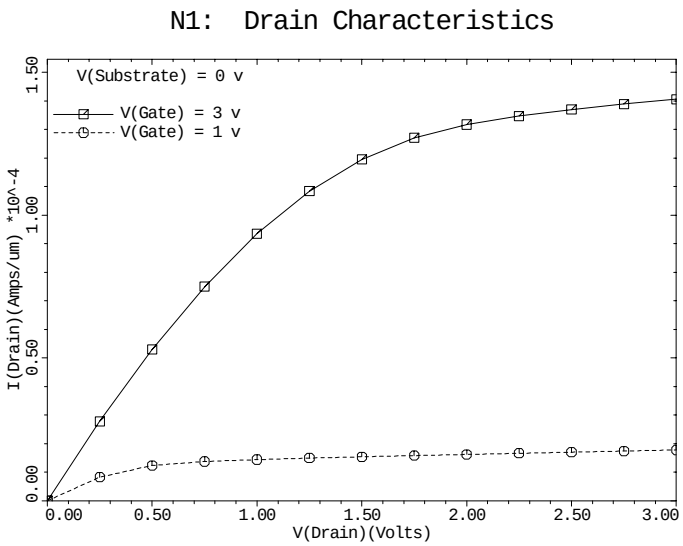


Figure 8-9 Drain characteristics from **CALL** to template file *mosstr1* at line 245 in file *mdex7n*, Figure 8-1

P-Channel MOSFET Example

This example illustrates the use of parameterized template files for the automatic creation of a 3.0 micron p-channel MOSFET and the automatic simulation of its gate characteristics. The output associated with the execution of Medici for the input file *mdex7p* is shown in Figures 8-10 through 8-15.

Template Used

The input file *mdex7p* uses **CALL** statements to enter the parameterized template files *mosdef0*, *mosstr0*, and *mosgat0* into the input.

```
  1... TITLE     Avant! MEDICI Example 7P - MOS Template File Example
  2... COMMENT   Gate Characteristics for a 3 Micron Gate
   ... +         P-channel Device

  3... COMMENT   Load the default parameters defining
   ... +         the structure and biases
  4... CALL      FILE=mosdef0  ^PRINT

 59... COMMENT   Modify the parameters to define a 3 micron
   ... +         p-channel device
 60... ASSIGN    NAME=TRANTYPE  C.VALUE=PMOS
 61... ASSIGN    NAME=FILE      C.VALUE=P3
 62... ASSIGN    NAME=LGATE     N.VALUE=3.0
 63... ASSIGN    NAME=LSPACER   N.VALUE=0
 64... ASSIGN    NAME=LDDPEAK   N.VALUE=0
 65... ASSIGN    NAME=SDJUNC    N.VALUE=.30
 66... ASSIGN    NAME=VTPEAK    N.VALUE=1E16
 67... ASSIGN    NAME=VTCHAR    N.VALUE=0.1

 68... COMMENT   Create the structure using template "mosstr0"
 69... CALL      FILE=mosstr0  ^PRINT

493... COMMENT   Specify parameters for obtaining Vbs=0 and Vbs=2.5 gate
494... ASSIGN    NAME=VD0     N.VALUE=-0.1
495... ASSIGN    NAME=VB0     N.VALUE=0.0
496... ASSIGN    NAME=VBSTEP  N.VALUE=2.5
497... ASSIGN    NAME=NBSTEP  N.VALUE=2
498... ASSIGN    NAME=VG0     N.VALUE=0
499... ASSIGN    NAME=VGSTEP  N.VALUE=-0.25
500... ASSIGN    NAME=NGSTEP  N.VALUE=9

501... COMMENT   Generate the solutions and plot results
502... CALL      FILE=mosgat0  ^PRINT
```

Figure 8-10 Output of the simulation input file *mdex7pci*

Suppress Content Listing

The **^PRINT** parameter is specified on the **CALL** statements so that the contents of the parameterized files are used but not printed to the standard output.

MOS Definition Template

Because it is desired to create an MOS device, the input file shown in Figure 8-10 begins by using a **CALL** statement to enter the file *mosdef0* into the input. This file contains default values for all parameters used by the MOS templates.

The default parameters in *mosdef0* are for a 1 micron gate n-channel LDD MOSFET with the structure doping defined by analytic profiles. See Appendix A: Template Files for a complete description of the available parameters and their default values.

Default Changes

In this example, the following changes must be made from the default device:

- The value of "PMOS" must be assigned to the name *TRANTYPE* to specify that a PMOS structure is to be created.
- The gate length is set to 3 microns using the name *LGATE*. Because the default device is an LDD, a few changes must be made to obtain a standard MOSFET.
- To avoid obtaining an LDD profile, the LDD peak impurity concentration, *LDDPEAK*, is set to zero.
- Set *LSPACER* to zero so there is no offset between the gate electrode and the source/drain profiles.
- Make adjustments to the source/drain junction depth (*SDJUNC*) and the threshold adjustment profile (*VTPEAK* and *VTCHAR*).
- Also assign a value to the name *FILE*, which is used as a prefix for the names of any output files that are generated by the templates.

MOS Structure Template

The program is now ready to call a template that actually creates the device structure. Appendix A shows that there are two choices for templates that create MOS devices: *mosstr0* and *mosstr1*. In this example, *mosstr0* is chosen.

Mesh

The template *mosstr0* creates a mesh where the spacing between node lines is fine near junction locations and in the channel region, and coarse away from these regions. Eliminates are used to remove closely spaced lines in regions where a fine grid spacing is not needed. This template is very versatile, efficiently creating grids for both short- and long-channel devices.

Simulation Mesh Plot

Figures 8-11 through 8-13 contain the output associated with the call to the template file *mosstr0* at line 69. Figure 8-11 shows the simulation mesh that is created.

Doping Plots

Two-dimensional and one-dimensional plots of impurity concentration for the final structure are shown in Figures 8-12 and 8-13.

Template

Having created the structure, a set of gate characteristics can now be obtained. Before doing so, however, values should be assigned to various parameters for describing the range of biases to use. The assignments at lines 494 through 500 are the following:

- V_{Drain} =-0.1V
- $V_{Substrate}$ =0V and 2.5V
- V_{Gate} =0V to -2V in -0.25V increments

The call to the template file *mosgat0* at line 502 performs the desired simulations. The output associated with this **CALL** is shown in Figures 8-14 and 8-15.

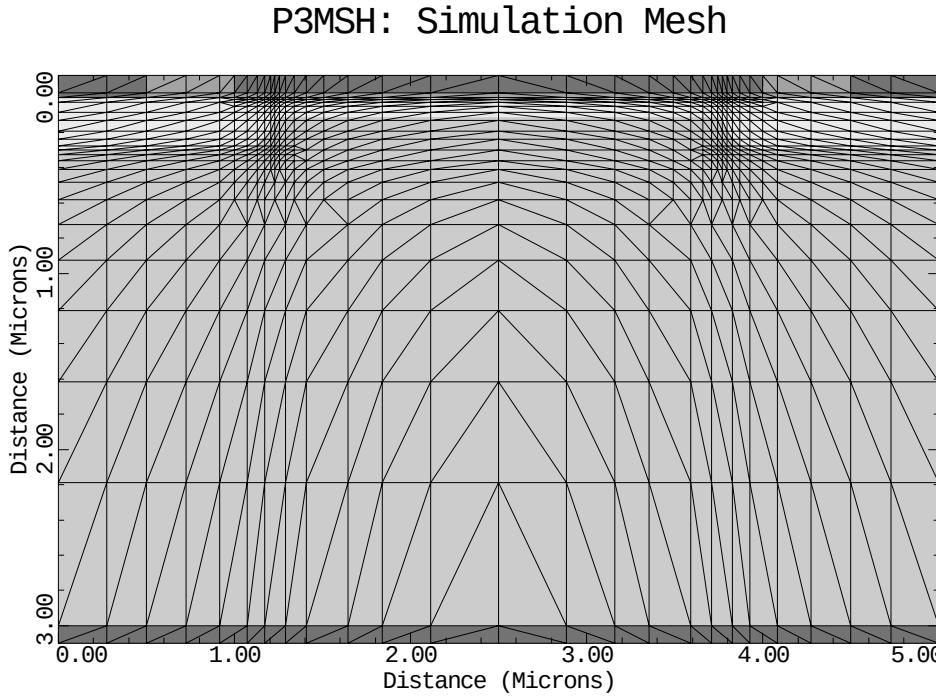


Figure 8-11 Simulation Mesh from **CALL** to template file *mosstr0* at line 69 in file *mdex7p*, Figure 8-10

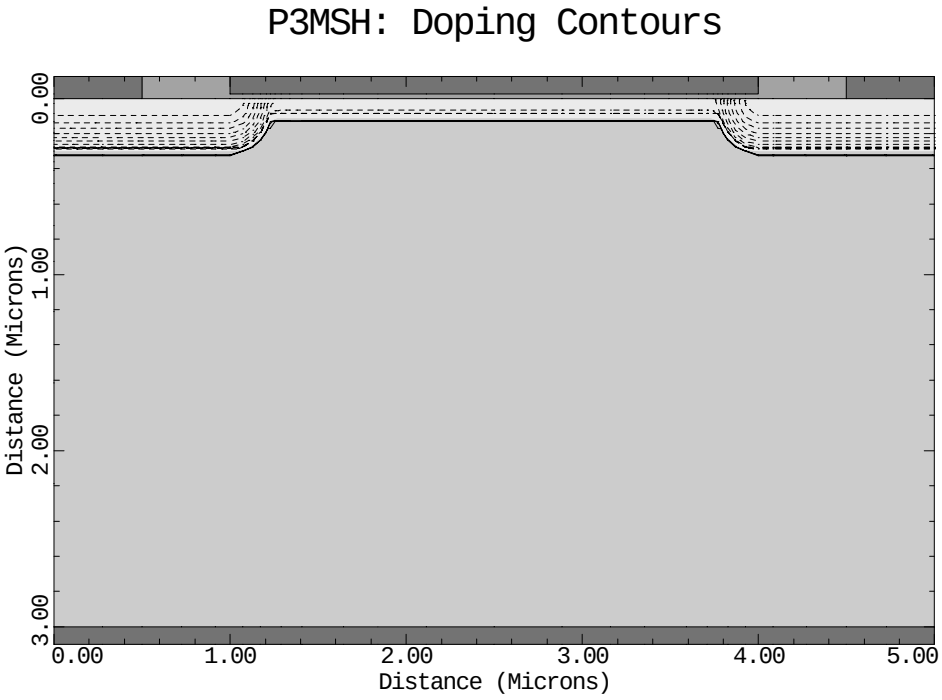


Figure 8-12 Doping contours from **CALL** to template file *mosstr0* at line 69 in file *mdex7p*, Figure 8-10

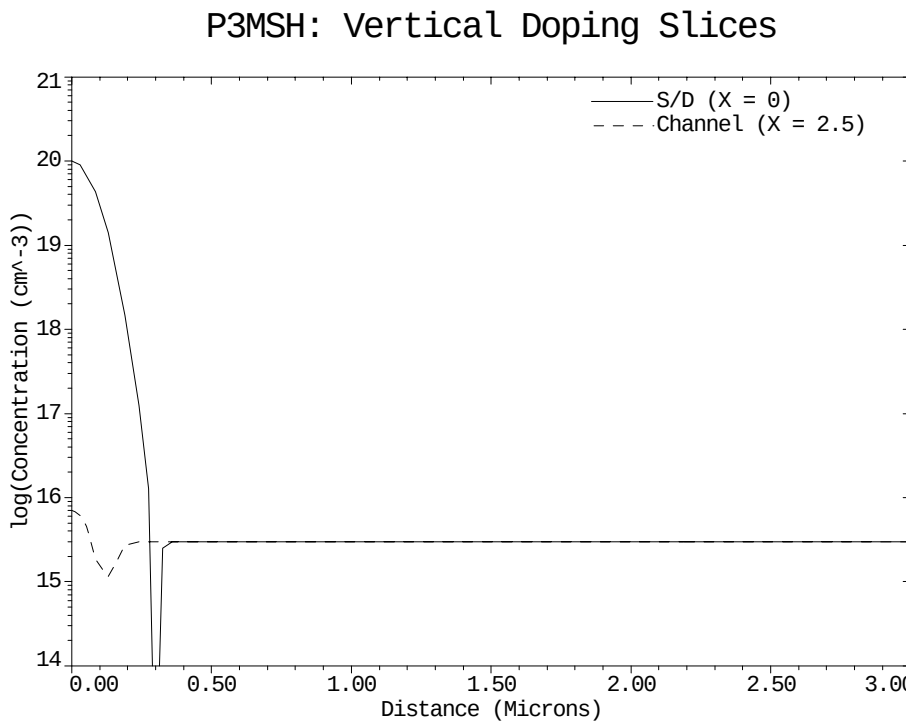


Figure 8-13 1D doping profiles from **CALL** to template file *mosstr0* at line 69 in file *mdex7p*, Figure 8-10

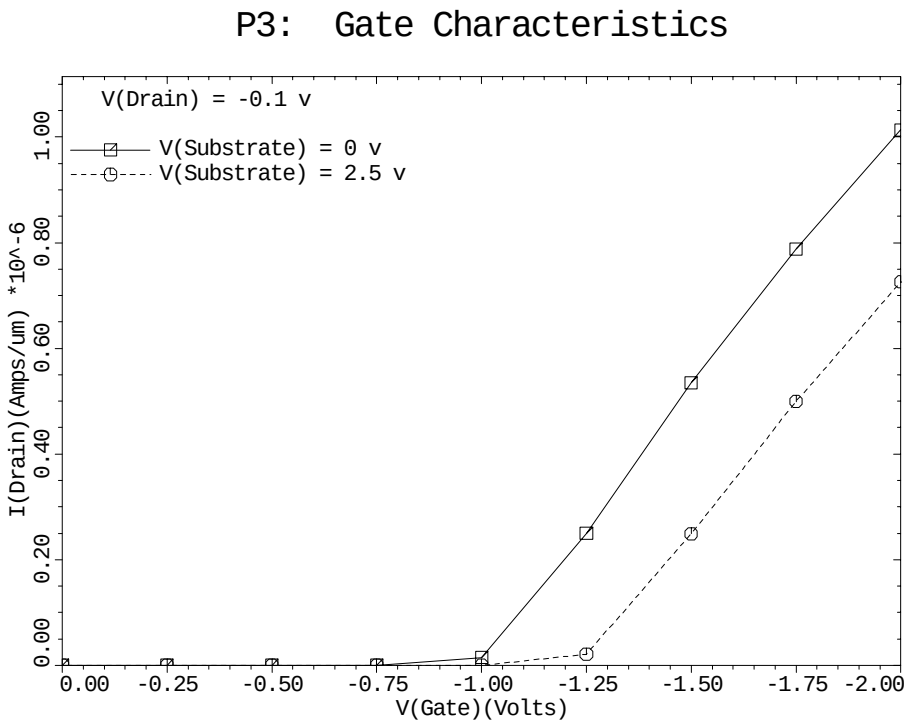


Figure 8-14 Turn on characteristics from **CALL** to template file *mosstr0* at line 502 in file *mdex7p*, Figure 8-10

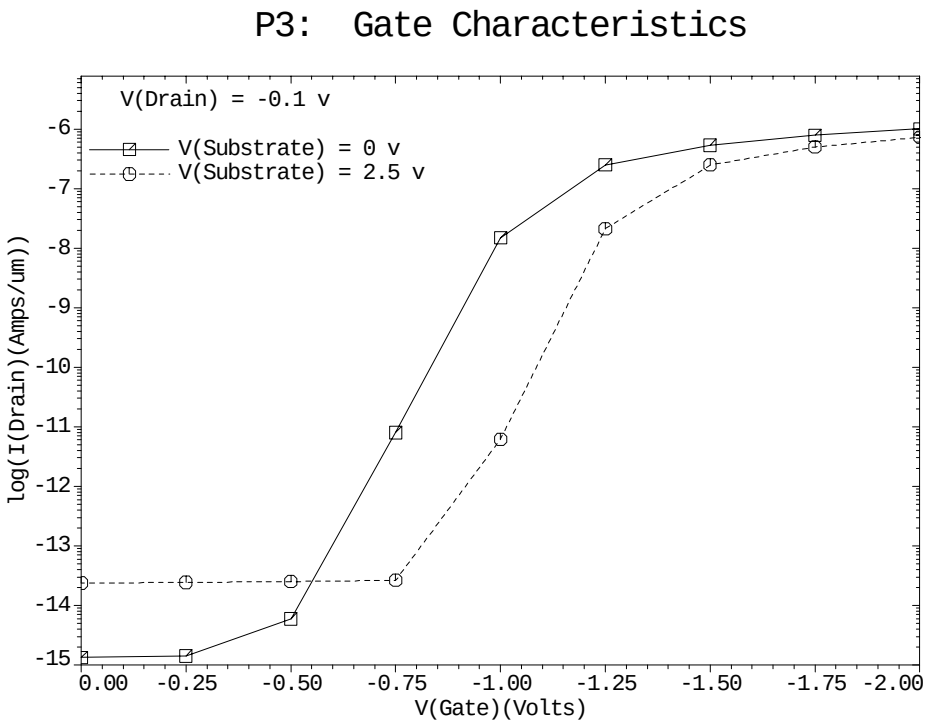


Figure 8-15 Gate characteristics from **CALL** to file *mosstr0* at line 502 in file *mdex7p*, Figure 8-10

NPN Bipolar Junction Transistor Example

This example illustrates the use of parameterized template files for the automatic creation of an NPN bipolar junction transistor and the automatic simulation of its forward-active device characteristics.

Templates Used

The input file *mdex7p* uses **CALL** statements to enter the parameterized template files *bipdef0*, *bipstr0*, and *bipgum0* into the input to accomplish this.

Suppress Content Listing

The **^PRINT** parameter is specified on the **CALL** statements so that the contents of the parameterized files are used but not printed to the standard output. The output associated with the execution of Medici for the input file *mdex7p* is shown in Figures 8-16 through 8-23.

Bipolar Definition Template

Because it is desired to create a bipolar device, the input file shown in Figure 8-16 begins by using a **CALL** statement to enter the file *bipdef0* into the input. This file contains default values for all parameters used by the bipolar templates.

The default parameters in *bipdef0* are for an NPN bipolar junction transistor with the structure doping defined by analytic profiles. See Appendix A: Template Files for a complete description of the available parameters and their default values.

```
  1... TITLE     Avant! MEDICI Example 7B - BJT Template File Example
  2... COMMENT   Forward Bias Points and AC Small-Signal Analysis

  3... COMMENT   Load the default parameters defining the
                 structure and biases
  4... CALL      FILE=bipdef0  ^PRINT

 59... COMMENT   Use the default NPN transistor
 60... ASSIGN    NAME=TRANTYPE  C.VALUE=NPN
 61... ASSIGN    NAME=FILE      C.VALUE=NPN

 62... COMMENT   Create the structure using template "bipstr0"
 63... CALL      FILE=bipstr0  ^PRINT

339... COMMENT   Display doping using a 3D projection plot
340... PLOT.3D   DOPING  LOG  TITLE="NPN Doping Surface"  ^FRAME
341... 3D.SURF   COLOR=4

342... COMMENT   Specify collector voltage and range of base biases.
343... ASSIGN    NAME=VC0     N.VALUE=3.0
344... ASSIGN    NAME=VB0     N.VALUE=0.0
345... ASSIGN    NAME=VBSTEP  N.VALUE=0.1
346... ASSIGN    NAME=NBSTEP  N.VALUE=10

347... COMMENT   Generate the solutions and plot results.
348...  CALL     FILE=bipgum0  ^PRINT
```

Figure 8-16 Output of the simulation input file *mdex7b*

In this example, the default device structure specified in *bipdef0* is used. The **ASSIGN** statement at line 61 is used to give a value to the name *FILE*, which is used as a prefix for the names of any output files that are generated by the templates.

Bipolar Structure Template

The program is now ready to call the template that actually creates the device structure. This is accomplished at line 63 where the template *bipstr0* is called.

Mesh

This template creates a mesh where the spacing between node lines is fine near junction locations and coarse away from the junctions. Eliminates are used to remove closely spaced lines in regions where a fine grid spacing is not needed.

Figures 8-17 through 8-19 contain the output associated with the call to the template file *bipstr0*.

Simulation Mesh

Figure 8-17 shows the simulation mesh that is created by the output.

Doping Plots

Figure 8-18 shows a two-dimensional plot of the doping contours in the device. Figure 8-19 shows one-dimensional plots of impurity concentration along slices through the final structure. A 3D projection plot of doping is requested with the **PLOT.3D** and statement at lines 340 and 341 of the input file and is plotted in Figure 8-20.

Gummel Plot Template

Having created the structure, a set of forward-active device characteristics can now be obtained. Before doing so, however, values should be assigned to various parameters for describing the range of biases to use. The assignments in lines 343 through 346 are such that $V_{Collector}$=3V and V_{Base}=0V through 0.9V in 0.1V increments. The call to the template file *bipgum0* at line 348 performs the desired simulations.

The output associated with this **CALL** is shown in Figures 8-21 through 8-23. These figures represent plots of $I_{Collector}$ and I_{Base} versus V_{Base}, current gain versus $I_{Collector}$, and cutoff frequency versus $I_{Collector}$.

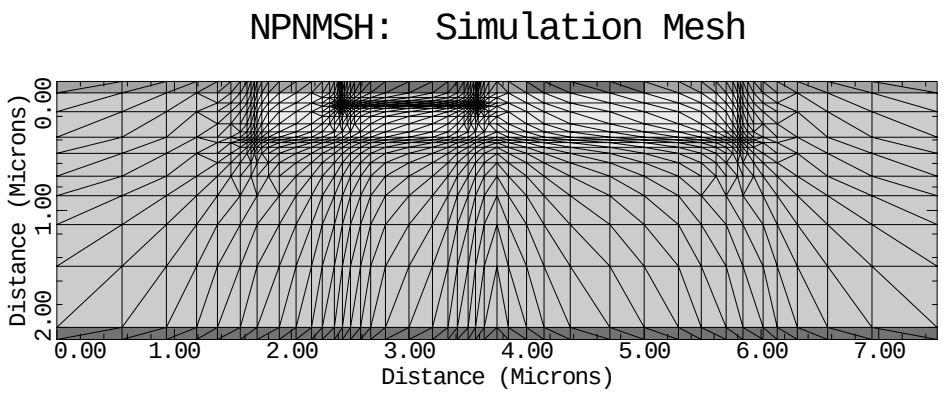


Figure 8-17 Simulation mesh from **CALL** to template file *bipstr0* at line 63 in file *mdex7b*, Figure 8-16

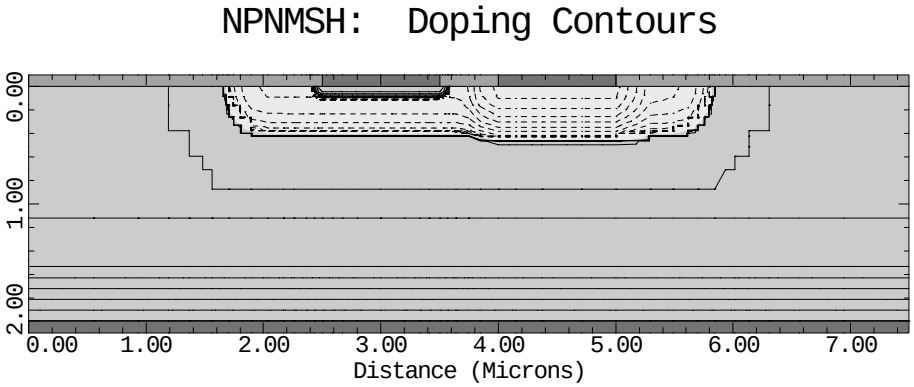


Figure 8-18 Doping contours from **CALL** to template file *bipstr0* at line 63 in file *mdex7b*, Figure 8-16

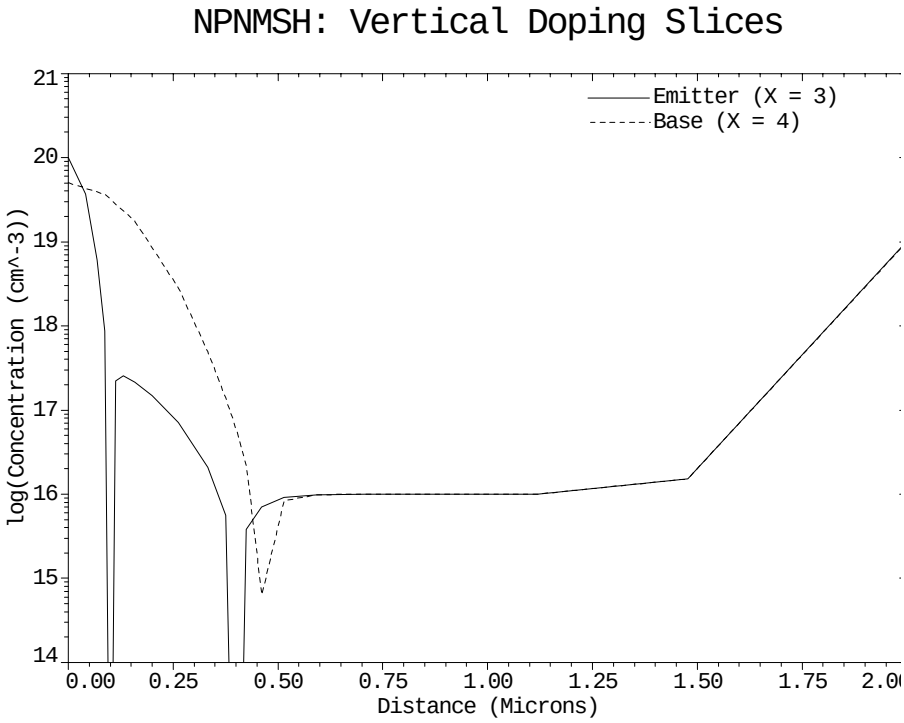


Figure 8-19 1D doping profiles from **CALL** to template file *bipstr0* at line 63 in file *mdex7b*, Figure 8-16

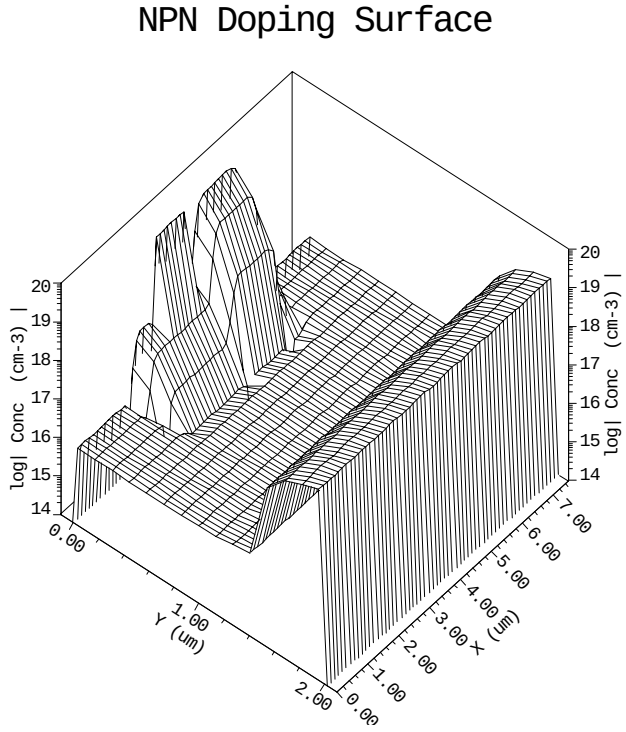


Figure 8-20 3D doping plot from **PLOT.3D** and **3D.SURF** at lines 340 and 341 in file *mdex7b*, Figure 8-16

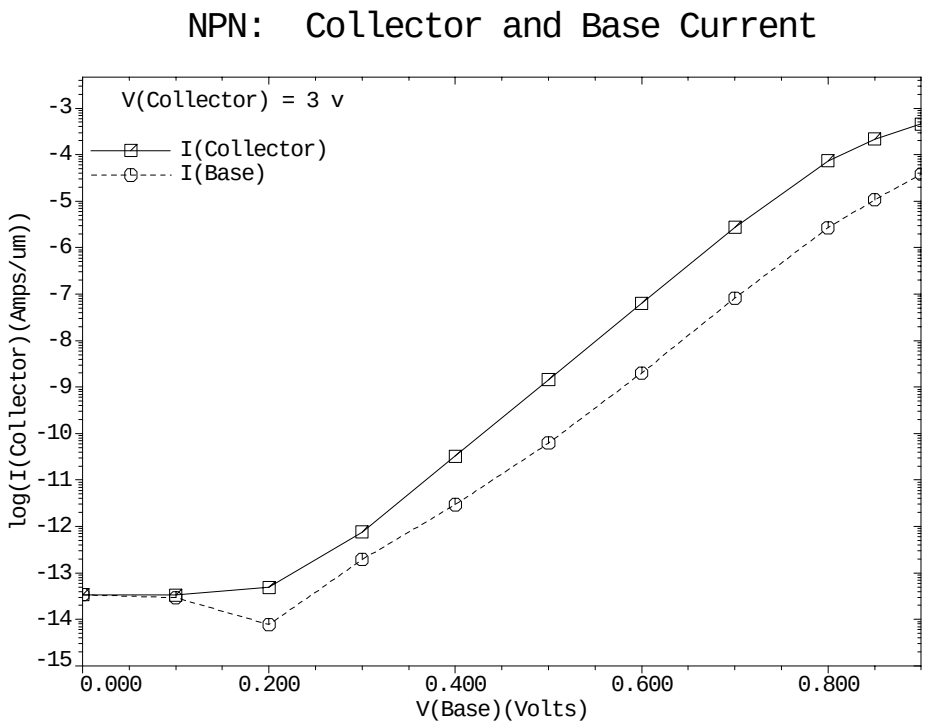


Figure 8-21 Gummel plot from **CALL** to template file *bipgum0* at line 348 in file *mdex7b*, Figure 8-16

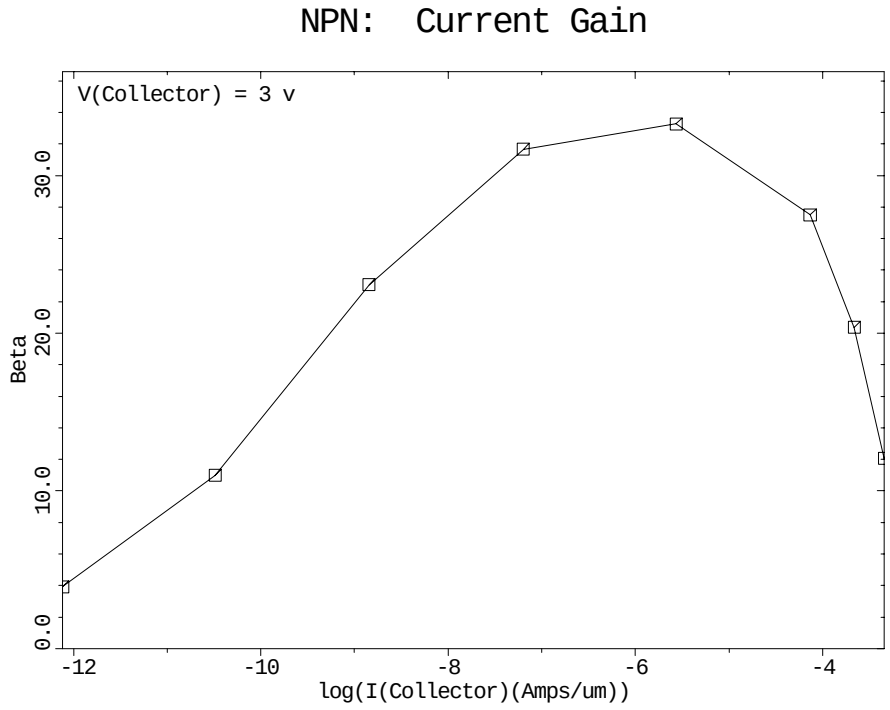


Figure 8-22 Current gain vs. collector current from **CALL** to template file *bipgum0* at line 348 file *mdex7b*, Figure 8-16

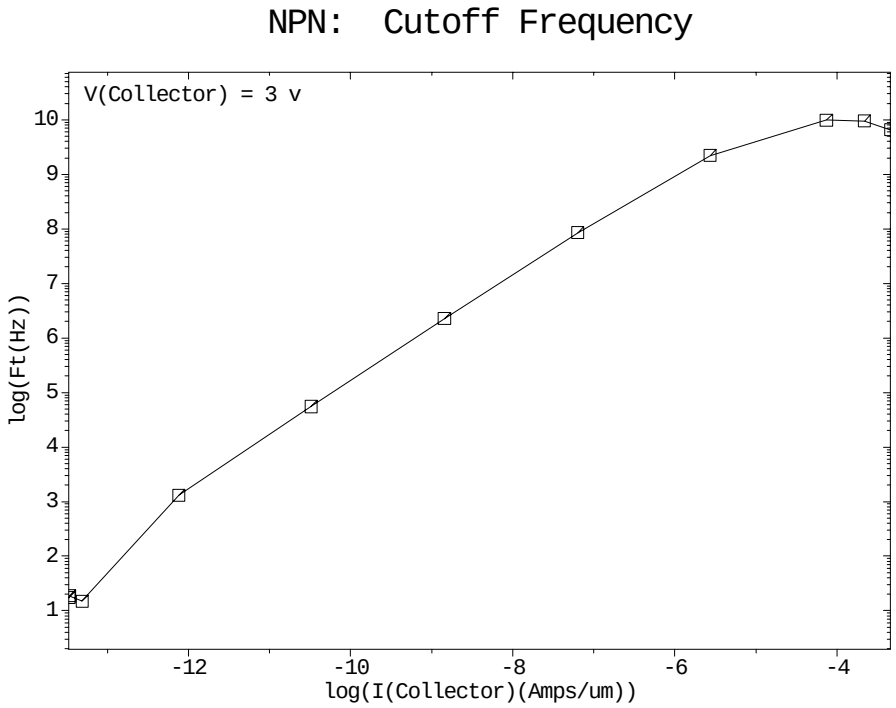


Figure 8-23 Cutoff frequency vs. collector current from **CALL** to template file *bipgum0* at line 348 in file *mdex7b*, Figure 8-16

Energy Balance Examples

Example Specifications

This chapter demonstrates application of the self-consistent solution of the carrier energy balance equations in addition to the standard set of drift-diffusion equations. This is done using two examples:

- A MOSFET simulation comparing substrate currents calculated with the conventional local electric field-based impact ionization model and a carrier temperature-based model.
- A bipolar junction transistor (BJT) simulation showing carrier velocity overshoot and the impact of the energy balance model on the transistor's Gummel plot and static current gain.

Substrate Current Simulation in an LDD MOSFET

This example demonstrates the use of the energy balance model for the calculation of substrate currents in a submicron LDD MOSFET.

Mesh Specifications

The example device is an N-FET with an effective channel length of 0.5 µm. Oxide thickness is 10 nm, spacer width is 150 nm, S/D junctions are 0.2 µm deep and the LDD junctions are 0.1 µm deep. Device structure and grid are generated using the MOS template file *mosstr1*. For the associated files and graphical plots see the following:

- Structure specification file, see Figure 9-1.
- Simulation file, see Figure 9-2.
- Input deck for plots, see Figure 9-3

- Mesh and doping plots, see Figures 9-4 and 9-5.

Solutions

After the creation of the mesh and device structure three sets of solutions are performed in a loop.

```
  1... TITLE     MEDICI Example 8A - LDD-MOSFET Simulation
  2... COMMENT   Calculation of substrate current using the carrier
   ... +         temperature-based impact ionization model

  3... COMMENT   Create the structure using the MOSFET templates
  4... CALL      FILE=mosdef0  ^PRINT

 59... COMMENT   Modify the mosdef0 parameters to define a
   ... +         0.5 micron n-channel LDD
 60... ASSIGN    NAME=TRANTYPE  C.VALUE=NMOS
 61... ASSIGN    NAME=FILE      C.VALUE=E8A
 62... ASSIGN    NAME=LGATE     N.VALUE=0.5
 63... ASSIGN    NAME=LSOURCE   N.VALUE=0.5
 64... ASSIGN    NAME=LSCONT    N.VALUE=0.25
 65... ASSIGN    NAME=LDRAIN    N.VALUE=0.5
 66... ASSIGN    NAME=LDCONT    N.VALUE=0.25
 67... ASSIGN    NAME=LSPACER   N.VALUE=0.15
 68... ASSIGN    NAME=TOX       N.VALUE=0.01
 69... ASSIGN    NAME=NSUB      N.VALUE=5E15
 70... ASSIGN    NAME=VTTYPE    C.VALUE=P
 71... ASSIGN    NAME=VTPEAK    N.VALUE=5E15
 72... ASSIGN    NAME=VTCHAR    N.VALUE=0.10
 73... ASSIGN    NAME=SDPEAK    N.VALUE=1E20
 74... ASSIGN    NAME=SDJUNC    N.VALUE=0.20
 75... ASSIGN    NAME=LDDPEAK   N.VALUE=2E18
 76... ASSIGN    NAME=LDDJUNC   N.VALUE=0.1
 77... ASSIGN    NAME=CHANSP    N.VALUE=.005
 78... ASSIGN    NAME=JUNCSP    N.VALUE=.03
 79... ASSIGN    NAME=RATIO     N.VALUE=1.4
 80... ASSIGN    NAME=VDBMAX    N.VALUE=3

 81... COMMENT   Create the structure using the template "mosstr0"
 82... CALL      FILE=mosstr0  ^PRINT

506... COMMENT   Specify models
507... CONTACT   NAME=Gate  P.POLY
508... MODELS    CONMOB  PRPMOB  FLDMOB  CONSRH  AUGER
```

Figure 9-1 First part of the simulation input file *mdex8a* (structure specification)

Conventional Drift-Diffusion Solution

The energy balance equation is *not* solved and the impact ionization rate is calculated based on the local electric field. Since **IMPACT.I** is specified on the **SOLVE** statement and *not* on the **MODELS** statement, impact ionization is calculated as a post-processing analysis. The ionization rates are not fed back into the carrier continuity equations.

Approximate Energy Balance Solution

In this mode the energy balance equation is solved to determine the mean carrier temperature. The carrier temperature, however, is never fed back to the drift diffusion equations. This mode of operation is almost as fast as a standard drift-diffusion solution since the energy balance equation is only solved once per bias point.

The coupling of the energy balance equation to the drift diffusion model is disabled by turning off the models **TMPDIF** and **TMPMOB**. (The standard field-

dependent mobility model **FLDMOB** is used). Post-processing impact ionization is used to calculate the substrate currents. Since **II.TEMP** is specified, the calculations are based on carrier temperature instead of electric field.

Full Energy Balance Model

This model is self consistent. In this model the electron temperature is fed-back into the continuity equations through the temperature dependent mobility model **TMPMOB** and the temperature diffusion model **TMPDIF** (both parameters are now set *true*).

This analysis mode is the slowest since multiple passes are required to solve the drift-diffusion and energy balance model. Impact ionization (based on carrier temperature) is again computed as a post processing step.

```
509... COMMENT   Apply drain bias (2.0v)
510... SYMBOL    CARR=0
511... METHOD    ICCG  DAMPED
512... SOLVE     V(Drain)=2.0  OUT.FILE=TEMPSOL

513... COMMENT   Perform three sets of solutions:
   ... +         1. DDE solution:       mobility(E), impact ioniz(E)
   ... +         2. Approx EB solution: mobility(E), impact ioniz(T),
   ... +                                no temp. enhanced diffusion
   ... +         3. Full EB solution:   mobility(T), impact ioniz(T)

514... LOOP   STEPS=3
515...   ASSIGN    NAME=ELETEMP  L.VALUE=(F,T,T)
516...   ASSIGN    NAME=FULLEB   L.VALUE=(F,F,T)
517...   ASSIGN    NAME=LOGFIL  C1="MDE8AID"  C2="MDE8AIE"  C3="MDE8AIF"
518...   ASSIGN    NAME=SOLFIL  C1="MDE8ASD"  C2="MDE8ASE"  C3="MDE8ASF"

519...   LOAD      IN.FILE=TEMPSOL

520...   IF COND=@ELETEMP

521...     IF COND=@FULLEB
522...       MODELS     TMPDIF  TMPMOB  II.TEMP
523...     ELSE
524...       MODELS    ^TMPDIF ^TMPMOB  II.TEMP
525...     IF.END

526...     SYMBOL    CARR=1  NEWTON  ELECTRON  ELE.TEMP
527...     METHOD    ETX.TOL=0.10
528...   ELSE
529...     SYMBOL    CARR=1  NEWTON  ELECTRON
530...   IF.END

531...   COMMENT   Sweep gate bias from 0v to 6v.
532...   LOG       OUT.FILE=@LOGFIL
533...   SOLVE     V(Gate)=0.00  IMPACT.I
534...   SOLVE     V(Gate)=0.25  ELEC=Gate  VSTEP=0.25  NSTEP=4  IMPACT.I
535...   SOLVE     V(Gate)=1.50  ELEC=Gate  VSTEP=0.50  NSTEP=8  IMPACT.I
536...   SOLVE     V(Gate)=6.00  IMPACT.I  OUT.FILE=@SOLFIL
537... L.END
```

Figure 9-2 Second part of simulation input file *mdex8a* (the simulation)

Plots

Upon completion of the loop, the calculated substrate currents, internal distribution of the impact ionization rate according to the drift-diffusion model and the

energy balance model using a carrier temperature-based impact ionization model, and the electron temperature distribution at V_{Drain}=2V, V_{Gate}=6V are plotted. The graphical output is shown in Figures 9-6 through 9-9.

Significant differences between the conventional and the energy balance simulation are observed: the substrate current curves have the characteristic "double hump" shape. The first maximum of the curve is due to impact ionization at the drain junction, whereas the second maximum is largely produced by impact ionization near the source junction.

The energy balance model predicts lower substrate currents for all gate biases, with most significant differences at high gate biases, where source-side impact ionization is dominant. The reason for this behavior is primarily the inability of the conventional local electric field impact ionization model to take nonlocal dark space effects into account. This is because the carriers traveling in a strong accelerating electric field require a certain time until they gain sufficient energy for impact ionization. This is accounted for in the carrier temperature-based impact ionization model, thus leading to lower impact ionization in the source.

```
538... COMMENT   Plot substrate current for the three cases
539... PLOT.1D   IN.FILE=MDE8AID  X.AX=V(Gate)  Y.AX=I(Drain)
   ... +         LOG  LINE=2  SYMB=1 BOT=1E-18  TOP=1E-2
   ... +   TITLE="Example 8A - Substrate Current"
540... PLOT.1D   IN.FILE=MDE8AID  X.AX=V(Gate)  Y.AX=II
   ... +         LOG  COL=2  SYMB=2  UNCH
541... PLOT.1D   IN.FILE=MDE8AIE  X.AX=V(Gate)  Y.AX=II
   ... +         LOG  COL=3  SYMB=3  UNCH
542... PLOT.1D   IN.FILE=MDE8AIF  X.AX=V(Gate)  Y.AX=II
   ... +         LOG  COL=4  SYMB=4  UNCH
543... LABEL     LAB="V(Drain) = 2v"
544... LABEL     LAB="I(Drain)"  X=3.5  Y=1E-5  LINE=2
   ... +         START.L  LX.FIN=3.2  COL=1  SYMB=1
545... LABEL     LAB="I(Sub.) (DD)"
   ... +         START.L  LX.FIN=3.2  COL=2  SYMB=2
546... LABEL     LAB="I(Sub.) (EB-full)"
   ... +         START.L  LX.FIN=3.2  COL=4  SYMB=4
547... LABEL     LAB="I(Sub.) (EB-approx)"
   ... +         START.L  LX.FIN=3.2  COL=3  SYMB=3

548... COMMENT   Impact ionization generation rate
   ... +       and electron temperature
549... LOAD      IN.FILE=MDE8ASD
550... PLOT.2D   BOUND  FILL  SCALE  Y.MAX=0.4  L.ELEC=-1
   ... +         TITLE="Example 8A - DD:  Impact Ionization"
551... CONTOUR   II.GENER  LOG  FILL  MIN=13  DEL=1  LINE=1
552... LABEL     LABEL="V(Drain) = 2v"  X=0.1  Y=0.3
553... LABEL     LABEL="V(Gate) = 6v"

554... LOAD      IN.FILE=MDE8ASF
555... PLOT.2D   BOUND  FILL  SCALE  Y.MAX=0.4  L.ELEC=-1
   ... +         TITLE="Example 8A - EB:  Impact Ionization"
556... CONTOUR   II.GENER  LOG  FILL  MIN=13  DEL=1  LINE=1
557... LABEL     LABEL="V(Drain) = 2v"  X=0.1  Y=0.3
558... LABEL     LABEL="V(Gate) = 6v"

559... PLOT.2D   BOUND  FILL  SCALE  Y.MAX=0.4  L.ELEC=-1
   ... +         TITLE="Example 8A - EB:  Electron Temperature"
560... CONTOUR   ELE.TEMP  FILL  MIN=500  DEL=250  LINE=1
561... LABEL     LABEL="V(Drain) = 2v"  X=0.1  Y=0.3
562... LABEL     LABEL="V(Gate) = 6v"
```

Figure 9-3 Third part of the simulation input file *mdex8a* (plots)

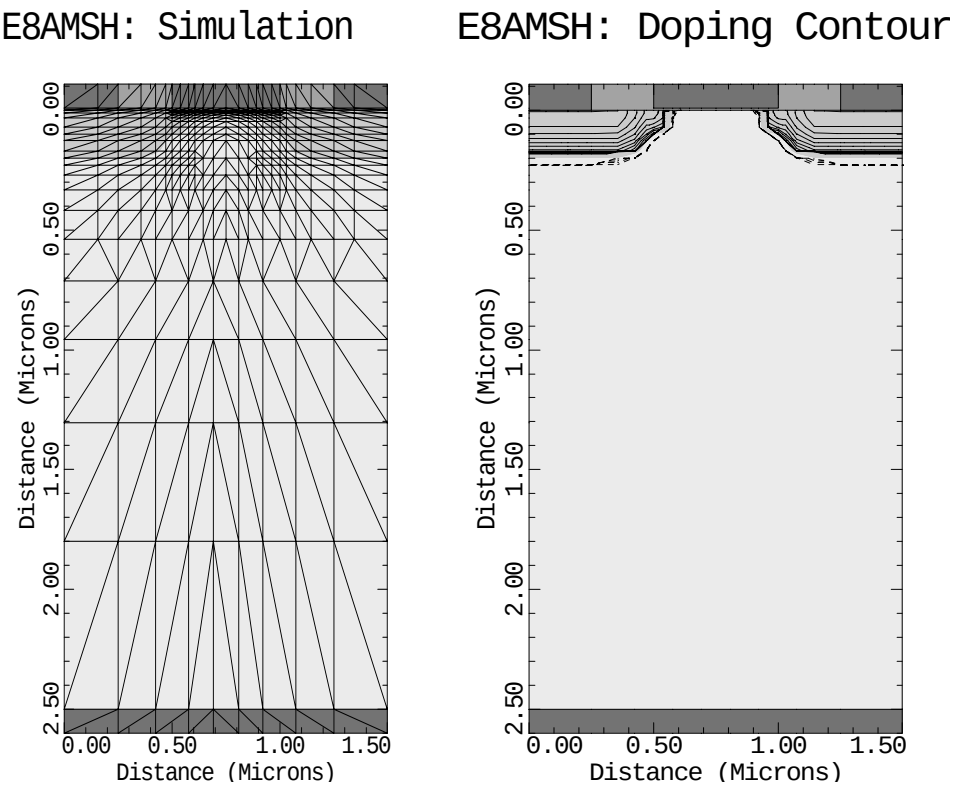


Figure 9-4 Device structure and grid on the left, doping concentration contours on the right

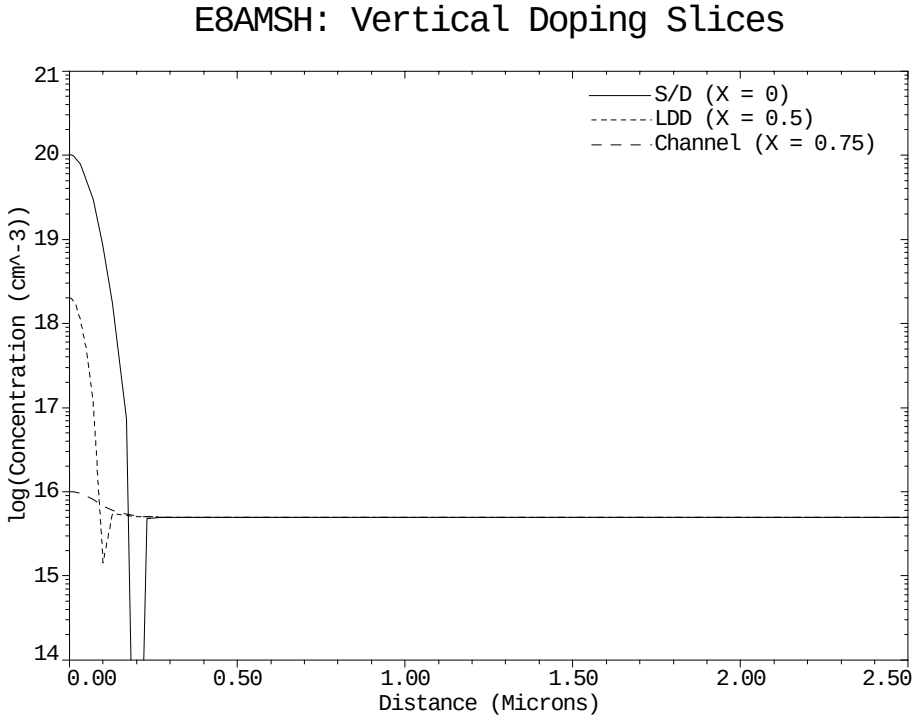


Figure 9-5 Vertical doping slices in source/drain S/D, LDD, and channel

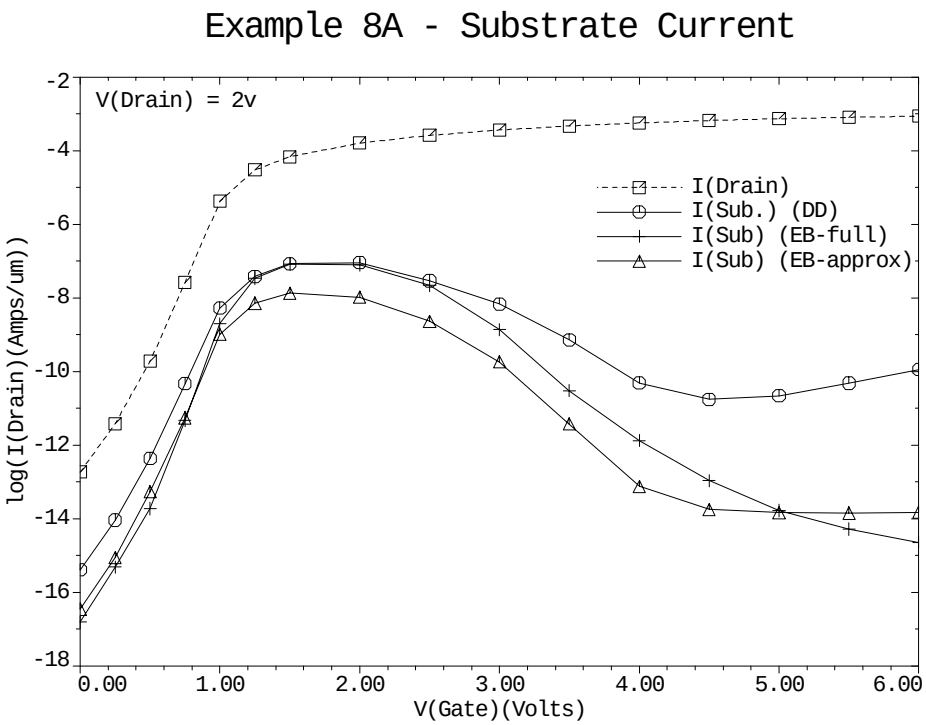


Figure 9-6 Substrate currents from lines 539 through 542 in file *mdex8a*, Figure 9-3

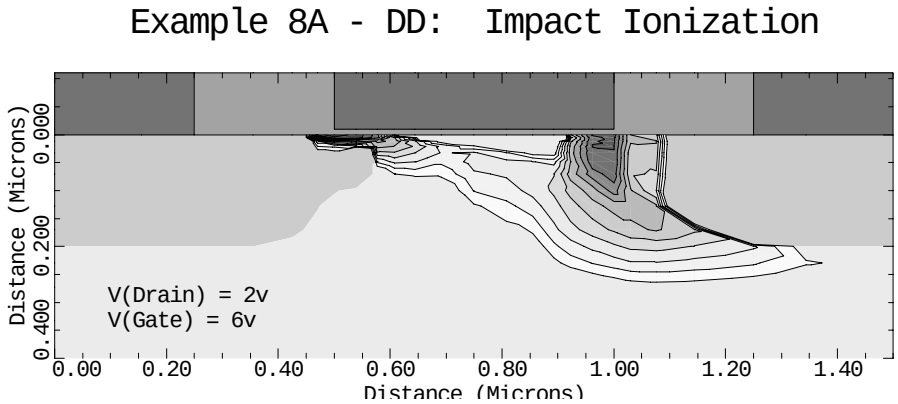


Figure 9-7 Impact ionization rate calculated using the local electric field impact ionization model (lines 549 through 553 in file *mdex8a*, Figure 9-3)

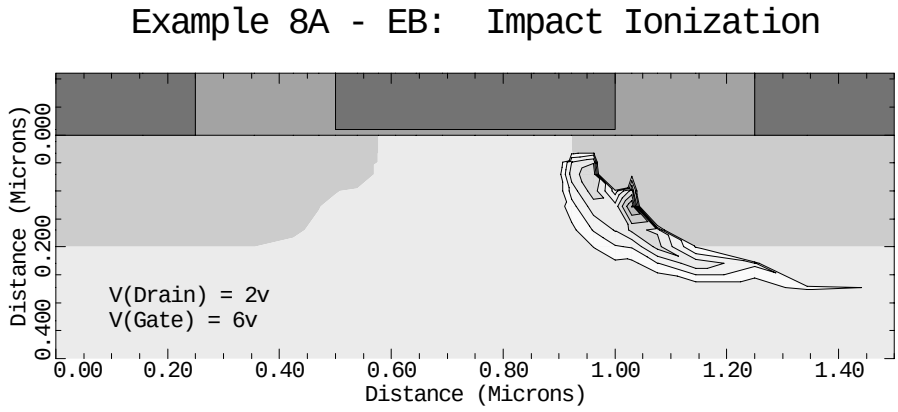


Figure 9-8 Impact ionization rate calculated using the carrier temperature based impact ionization model (lines 554 through 558 in file *mdex8a*, Figure 9-3)

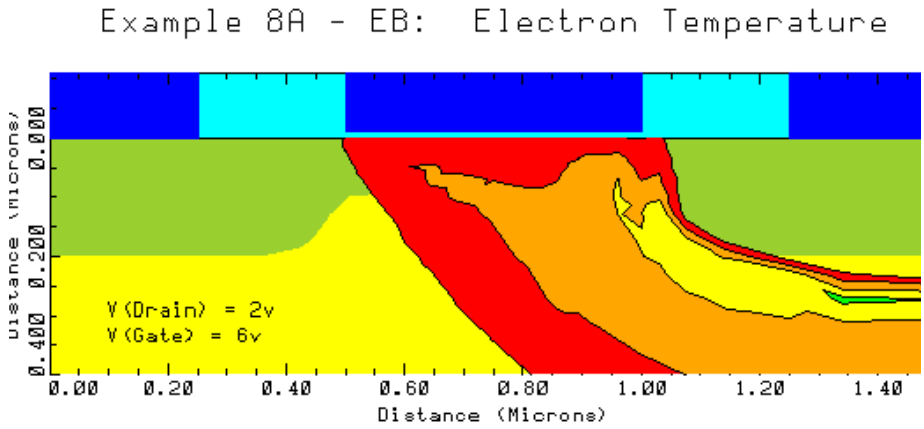


Figure 9-9 Electron temperature from lines 559 through 562 in file *mdex8a*, Figure 9-3

Bipolar Junction Transistor Example

The example demonstrates the use of the energy balance model for the simulation of a bipolar junction transistor.

Mesh Specification

The example uses an NPN device with a base width of 0.15 µm. Device structure and grid are generated using the BJT template *bipstr0*. The mesh consists of 1466 triangles and 787 nodes. It is optimized to resolve the base-emitter junction,

which is important to obtain accurate current gain results. For the associated files and graphical plots, see the following:

- Structure specification and simulation, Figure 9-10
- Simulation mesh, Figure 9-11
- Doping, Figures 9-12 and 9-13
- Input file for plots, Figures 9-14 and 9-15

Solution

After the creation of the mesh and device structure a collector-emitter bias of 2V and a base-emitter bias of 0.3V are applied in one step using a zero-carrier Gummel cycle algorithm (specified in line 351). The solution thus obtained is used as an initial guess for stepping the base-emitter voltage from 0.3V to 0.85V in steps of 0.05V. The simulation is then performed in a loop (lines 353-369) with two methods.

Standard Drift-Diffusion Solution

Both carrier-continuity equations as well as Poisson's equation are solved in a coupled Newton procedure. A field-dependent mobility model is used to describe the velocity saturation effect.

Electron Energy Balance Method

This method solves the electron energy balance equation, specified in line 361. Specifying the **MODEL** parameter **TMPMOB** (line 360) selects a carrier temperature-based mobility model. The field-dependent mobility model is switched off automatically since **TMPMOB** and **FLDMOB** describe the same physical effect (velocity saturation) and thus cannot be used simultaneously.

Plots

Following the simulation, two-dimensional contour plots of the potential and electron temperature are created using the last computed solution (V_{Base}=0.85V, $V_{Collector}$=2V). Base push-out is clearly visible in both plots. See Figures 9-17 and 9-18.

For the same solution (V_{Base}=0.85V, $V_{Collection}$=2V) one-dimensional cross-sectional plots through the middle of the base x=2.5 μm are also presented (see Figures 9-19 through 9-21). The displayed functions are:

- Potential
- Electron temperature
- Electron mean velocity

The potential plot shows a significant voltage drop in the collector due to high current injection. The electron temperature peak is substantially lower at this base-

emitter bias than it was at lower biases. A comparison of the electron temperature to the so-called static temperature

$$T_n^{static} = \frac{2}{3}\left(\frac{v_{sat} \cdot \tau_w}{kT_0/q}\right)T_0 \cdot E + T_0 \qquad \text{Equation 9-1}$$

shows the carrier cooling in the base-emitter junction, carrier heating in the collector as well as the separation of the peak electric field and peak carrier temperature in the direction of electron flow. This separation leads to a velocity overshoot at the base-collector junction. The velocity overshoot is moderate at this bias.

Finally, a Gummel plot and the static current gain $\beta = I_c/I_b$ versus base bias are plotted for the conventional drift-diffusion model and the energy balance solution. The energy balance model predicts a slightly higher base current and a slightly lower collector current at base biases V_{Base}>0.5V. The overall effect for the exam-

ple device is reduced static current gain in comparison to the conventional simulation.

```
  1... TITLE     MEDICI Example 8B - BJT with Energy Balance
  2... COMMENT   Demonstrates carrier heating and velocity overshoot

  3... COMMENT   Use the bipolar templates to define the structure
  4... CALL      FILE=bipdef0  ^PRINT

 59... COMMENT   Modify the default NPN transistor
 60... ASSIGN    NAME=TRANTYPE  C.VALUE=NPN
 61... ASSIGN    NAME=FILE      C.VALUE=E8B
 62... ASSIGN    NAME=BCJUNC    N.VALUE=0.30
 63... ASSIGN    NAME=XBJUNC    N.VALUE=0.35
 64... ASSIGN    NAME=BPEAK     N.VALUE=5E18
 65... ASSIGN    NAME=EBJUNC    N.VALUE=0.15
 66... ASSIGN    NAME=EBSP      N.VALUE=.02
 67... ASSIGN    NAME=BCSP      N.VALUE=.03
 68... ASSIGN    NAME=RATIO     N.VALUE=1.5
 69... ASSIGN    NAME=VCBMAX    N.VALUE=2

 70... COMMENT   Create the structure using the template ”bipstr0”
 71... CALL      FILE=bipstr0  ^PRINT

347... COMMENT   Specify models and obtain an initial solution
348... MODELS    CONMOB  FLDMOB  SRH  AUGER  BGN
349... SYMBOL    CARR=0
350... METHOD    ICCG  DAMPED
351... SOLVE     V(Collector)=2.0  V(Base)=0.3  OUT.FILE=TEMPSOL

352... COMMENT   The first step is for drift-diffusion, the second
   ... +         step is for energy balance.

353... LOOP      STEPS=2
354...   ASSIGN    NAME=ELETEMP  L.VALUE=(F,T)
355...   ASSIGN    NAME=LOGFIL   C1=MDE8BID  C2=MDE8BIE
356...   ASSIGN    NAME=SOLFIL   C1=MDE8BSD  C2=MDE8BSE

357...   LOAD      IN.FILE=TEMPSOL

358...   IF COND=@ELETEMP
359...     COMMENT   Use temperature dependent mobility instead of FLDMOB
360...     MODELS    TMPMOB
361...     SYMB      CARRIERS=2  NEWTON  ELE.TEMP
362...     METHOD    ETX.TOL=0.1
363...   ELSE
364...     SYMB      CARRIERS=2  NEWTON
365...   IF.END

366...   LOG       OUT.FILE=@LOGFIL
367...   SOLVE     V(Base)=0.30  ELEC=Base  VSTEP=0.05  NSTEP=10
368...   SOLVE     V(Base)=0.85  OUT.FILE=@SOLFIL
369...L.END
```

Figure 9-10 Output of the simulation input file *mdex8b* (structure specification and simulation)

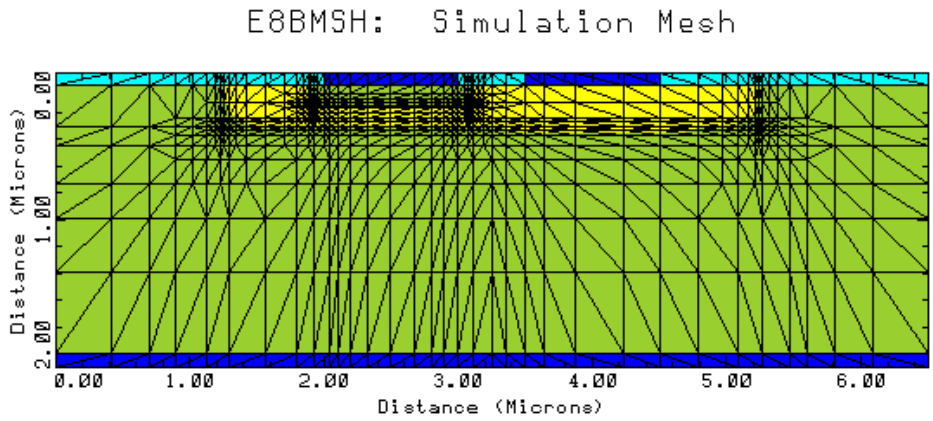


Figure 9-11 BJT device structure and grid

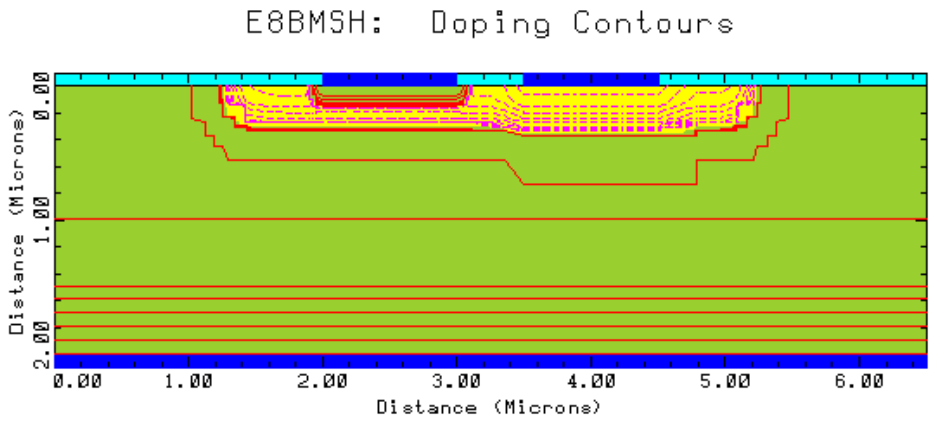


Figure 9-12 BJT doping concentration contour lines

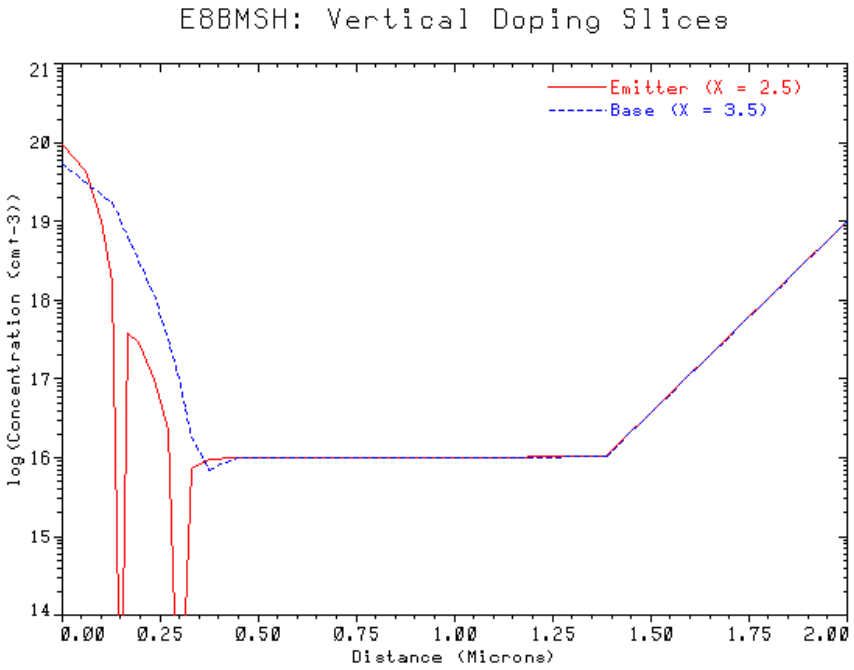


Figure 9-13 Vertical cross-section plots of the doping concentration in the BJT through the emitter and the base

```
 1... COMMENT   MEDICI Example 8BP - BJT with Energy Balance
 2... COMMENT   Post-processing of Example 8B simulation results

 3... COMMENT   Load the structure and the energy balance solution
  ... +         for V(Collector)=2.0 and V(Base)=0.85
 4... MESH      IN.FILE=E8BMSH
 5... LOAD      IN.FILE=MDE8BSE
 6... COMMENT   2d plots using the energy balance solution
  ... +         for V(Base)=0.85v
 7... ASSIGN    NAME=BIAS  C.VALUE="V(Coll.)=2.0v, V(Base)=0.85v"

 8... PLOT.2D   SCALE  TITLE="Example 8BP - Potential"
 9... CONTOUR   POTEN  FILL  LINE=1
10... PLOT.2D   BOUND  SCALE  JUNC  L.JUNC=1  L.ELEC=-1  ^CLEAR
11... LABEL     LABEL=@BIAS  Y=1.9

12... PLOT.2D   SCALE  FILL  TITLE="Example 8BP - Electron Temperature"
13... CONTOUR   ELE.TEMP  FILL  MIN=400  DEL=150
14... PLOT.2D   BOUND  SCALE  JUNC  L.JUNC=1  L.ELEC=-1  ^CLEAR
15... LABEL     LABEL=@BIAS  Y=1.9

16... COMMENT   1D plots using the energy balance solution
  ... +         for V(Base)=0.85v
17... ASSIGN    NAME=XP    N.VALUE=2.5

18... PLOT.1D   POTEN  COLOR=2  TITLE="Example 8BP - Potential, X="@XP
  ... +         X.START=@XP  X.END=@XP  Y.START=0  Y.END=2
19... LABEL     LABEL=@BIAS
```

Figure 9-14 First part of the simulation input file *mdex8bp*, used to generate plots

```
20... COMMENT   Plot the electron temperature and the "static
  ... +         temperature." The static temperature can be approximated
  ... +         using the electric field (E) as follows:
  ... +
  ... +             2*vsat*tauw*T0          2*1.035e7*2e-13*300
  ... +         T = -------------- E + T0 = ------------------- E + 300
  ... +               3*(k*T0/q)                 3*0.02585
  ... +
  ... +         or T = E/62.4 + 300

21... ASSIGN    NAME=TMIN  N.VALUE=0
22... ASSIGN    NAME=TMAX  N.VALUE=800
23... ASSIGN    NAME=EMIN  N.VALUE=((@TMIN-300)*62.4)
24... ASSIGN    NAME=EMAX  N.VALUE=((@TMAX-300)*62.4)

25... PLOT.1D   ELE.TEMP  TITLE="Electron and Static Temperature, X="@XP
  ... +         X.START=@XP  X.END=@XP  Y.START=0  Y.END=2
  ... +         BOT=@TMIN  TOP=@TMAX  COLOR=2  SYMB=2
26... PLOT.1D   E.FIELD  Y.COMP  NEGATIVE  ^AXES  ^CLEAR
  ... +         X.START=@XP  X.END=@XP  Y.START=0  Y.END=2
  ... +         BOT=@EMIN  TOP=@EMAX  COLOR=3  SYMB=3  LINE=2
27... LABEL     LABEL=@BIAS
28... LABEL     LABEL="Electron Temperature"  START.L  LX.FIN=1  X=1.15
  ... +         SYMB=2  COLOR=2
29... LABEL     LABEL="Static Temperature"  START.L  LX.FIN=1
  ... +         SYMB=3  COLOR=3  LINE=2

30... PLOT.1D   ELE.VEL  TITLE="Example 8BP - Electron Velocity, X="@XP
  ... +         X.START=@XP  X.END=@XP  Y.START=0  Y.END=2
  ... +         SYMB=2  COLOR=2
31... LABEL     LABEL=@BIAS  X=1.0
```

Figure 9-15 Second part of simulation input file *mdex8bp*

```
32... COMMENT   I-V plots for drift-diffusion and energy balance
33... EXTRACT   NAME=Beta  EXPRESS=@I(Collector)/@I(Base)

34... PLOT.1D   IN.FILE=MDE8BIE  X.AXIS=V(Base)  Y.AXIS=I(Collector)  LOG
  ... +         SYMB=2  COLOR=2  BOT=1E-14  TOP=1E-3
  ... +         TITLE="Example 8BP - Gummel Plot"
35... PLOT.1D   IN.FILE=MDE8BID  X.AXIS=V(Base)  Y.AXIS=I(Collector)  LOG
  ... +         SYMB=3  COLOR=3  LINE=2  UNCHANGE
36... PLOT.1D   IN.FILE=MDE8BIE  X.AXIS=V(Base)  Y.AXIS=I(Base)  LOG
  ... +         SYMB=4  COLOR=4  UNCHANGE
37... PLOT.1D   IN.FILE=MDE8BID  X.AXIS=V(Base)  Y.AXIS=I(Base)  LOG
  ... +         SYMB=5  COLOR=5  LINE=2  UNCHANGE
38... LABEL     LABEL="V(Coll.)=2.0v"  X=0.36
39... LABEL     LABEL="I(Coll.) Energy Balance"  START.L  LX.FIN=0.32
  ... +         SYMB=2  COLOR=2  X=0.35  Y=5E-5
40... LABEL     LABEL="I(Coll.) Drift-Diffusion" START.L  LX.FIN=0.32
  ... +         SYMB=3  COLOR=3  LINE=2
41... LABEL     LABEL="I(Base) Energy Balance"  START.L  LX.FIN=0.32
  ... +         SYMB=4  COLOR=4
42... LABEL     LABEL="I(Base) Drift-Diffusion" START.L  LX.FIN=0.32
  ... +         SYMB=5  COLOR=5  LINE=2

43... COMMENT   Beta plots for drif-diffusion and energy balance
44... PLOT.1D   IN.FILE=MDE8BIE  X.AXIS=V(Base)  Y.AXIS=Beta
  ... +         BOT=0  TOP=110 COLOR=2  SYMB=2
  ... +         TITLE="Example 8BP - I(Coll.)/I(Base)"
45... PLOT.1D   IN.FILE=MDE8BID  X.AXIS=V(Base)  Y.AXIS=Beta  UNCHANGE
  ... +         COLOR=3  SYMB=3  LINE=2
46... LABEL     LABEL="V(Coll.)=2.0v"  X=0.7
47... LABEL     LABEL="I(Coll.)/I(Base) Energy Balance"
  ... +         START.L  LX.FIN=0.32  X=0.35 SYMB=2  COLOR=2
48... LABEL     LABEL="I(Coll.)/I(Base) Drift-Diffusion"
  ... +         START.L  LX.FIN=0.32 SYMB=3  COLOR=3  LINE=2
```

Figure 9-16 Third part of the simulation input file *mdex8bp*

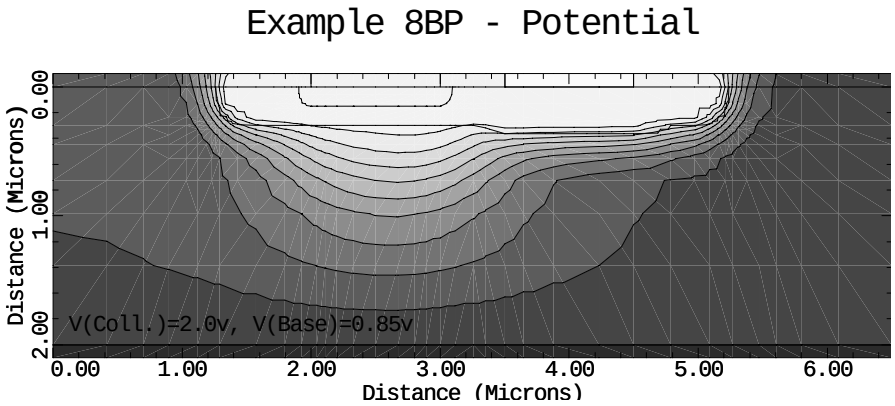


Figure 9-17 Contour lines of the electric potential generated by lines 9 through 11 in file *mdex8bp*, Figure 9-14

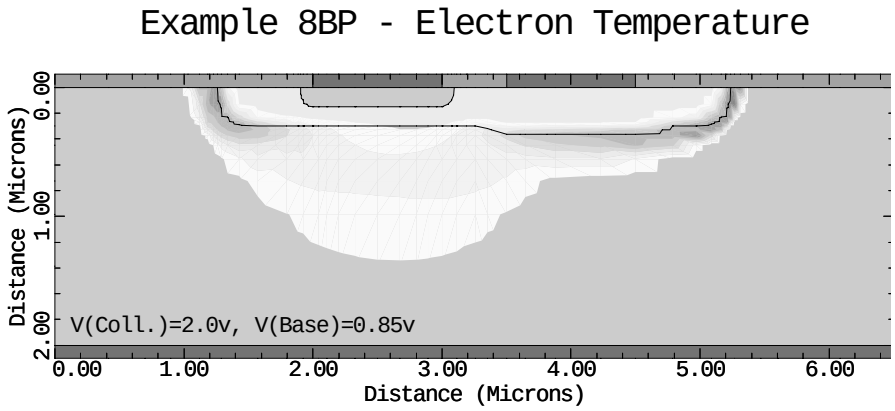


Figure 9-18 Contour lines of the electron temperature generated lines 12 through 15 in file *mdex8bp*, Figure 9-14

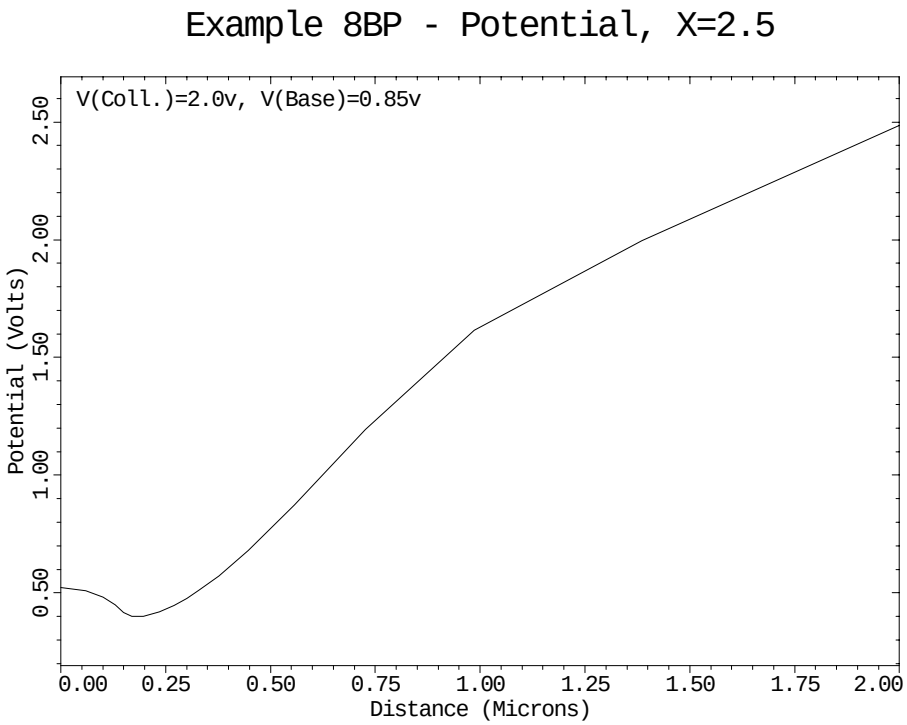


Figure 9-19 Cross-sectional plot of the electric potential generated by lines 18 and 19 in file *mdex8bp*, Figure 9-15

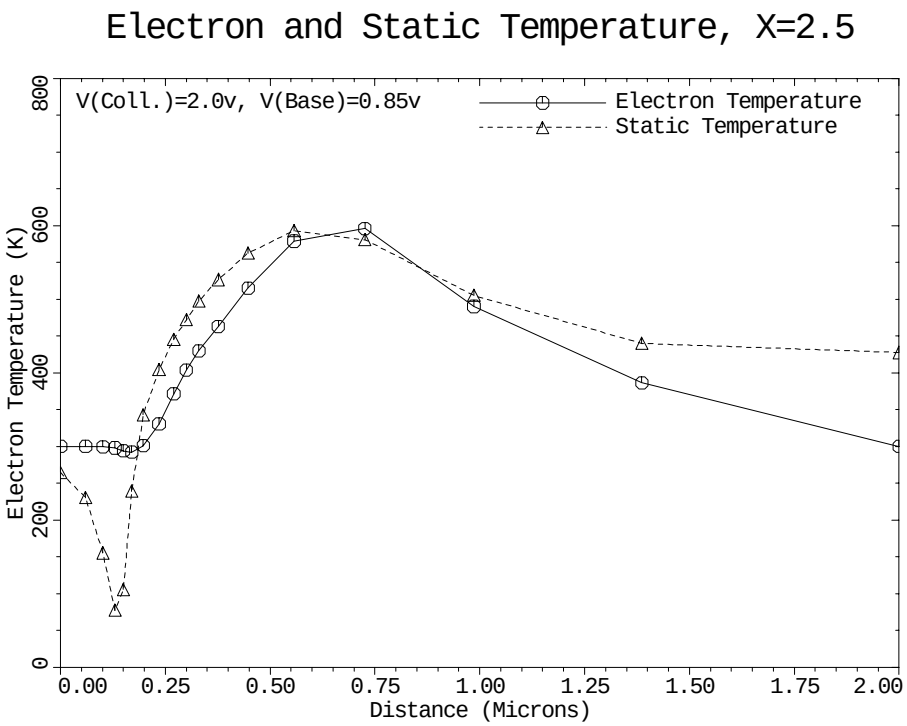


Figure 9-20 Cross-sectional plot of the electric potential generated by lines 25 through 29 in file *mdex8bp*, Figure 9-15

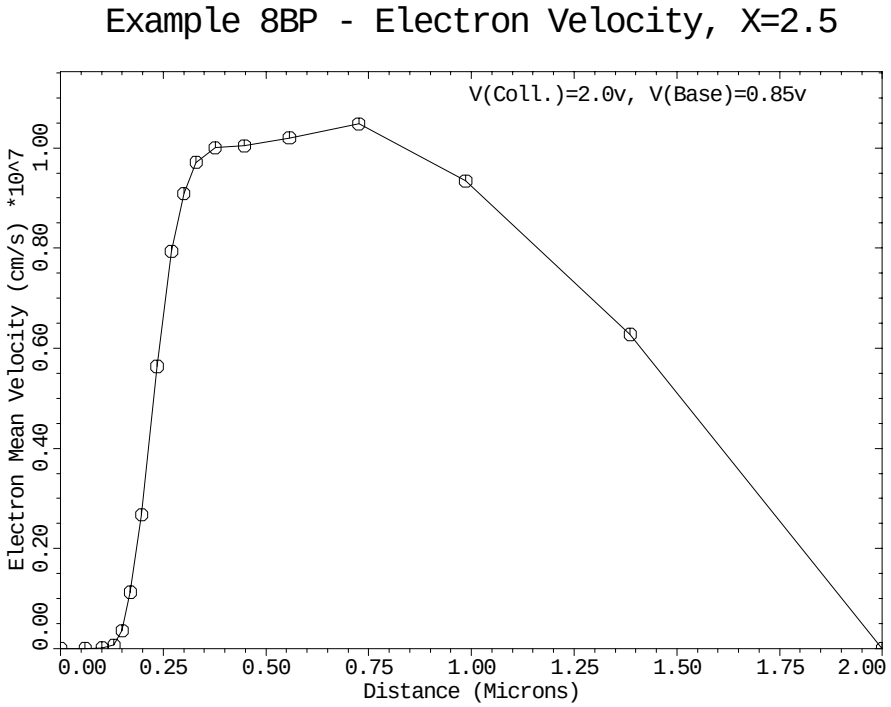


Figure 9-21 Cross-sectional plot of the electron velocity generated by lines 30 and 31 in file *mdex8bp*, Figure 9-15

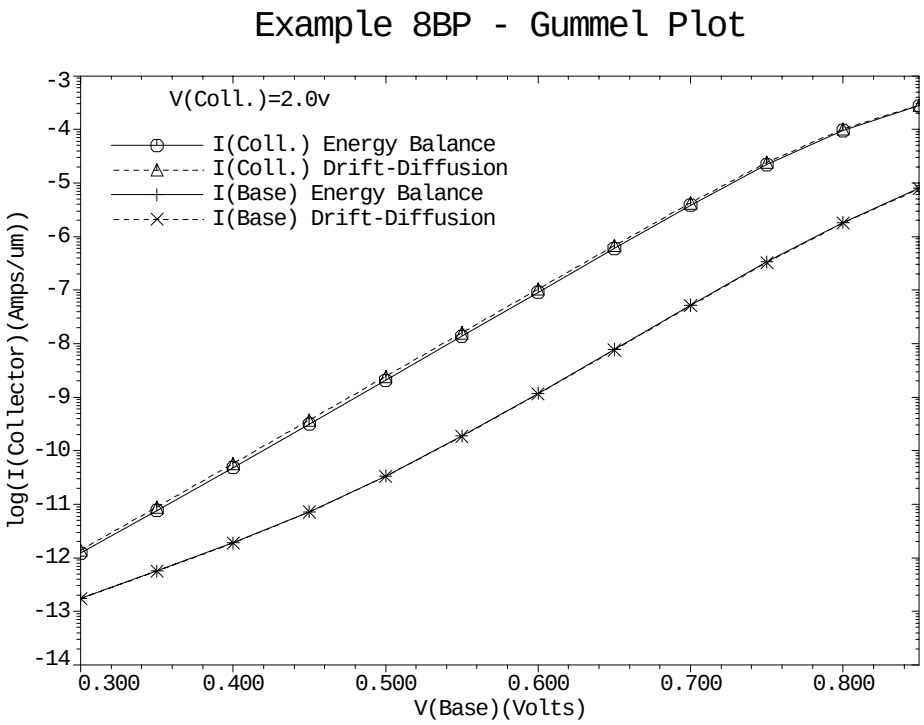


Figure 9-22 Gummel plots generated by lines 34 through 42 in file *mdex8bp*, Figure 9-16

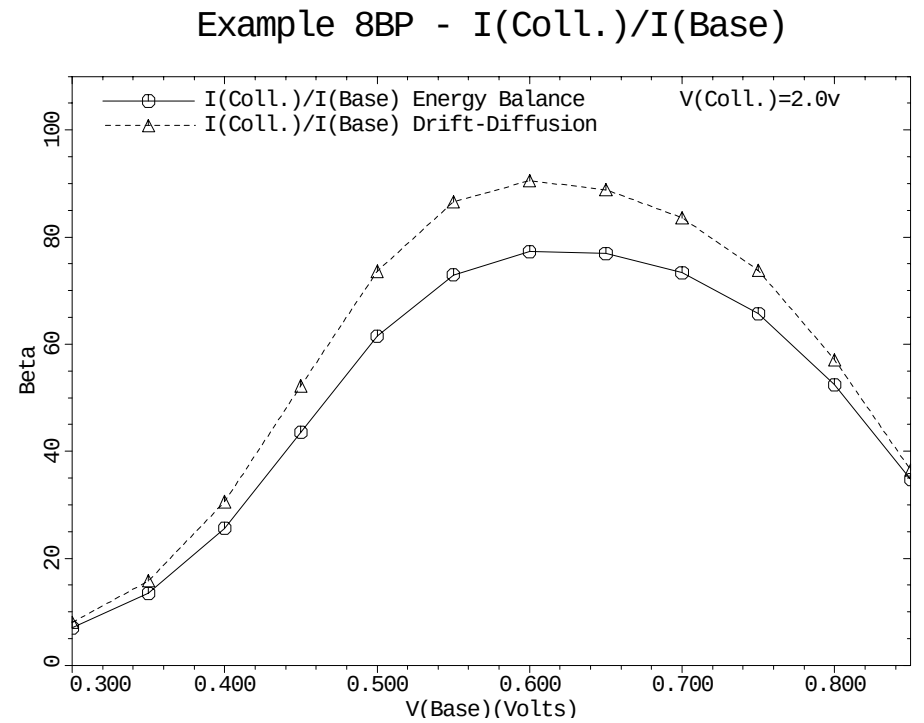


Figure 9-23 Static current gain generated by lines 44 through 48 in file *mdex8bp*, Figure 9-16

Interface Examples

Introduction

This chapter contains examples that illustrate how Medici interfaces to both process simulation and parameter extraction programs.

Interfaces to Process Simulators

The output from a process simulation program can consist of the following:

- Profile information, which describes the impurity distribution within a device structure.
- Topography information, which describes the shape of the various material regions which make up a device structure.

Both kinds of information can be passed to Medici, either together or separately, allowing the electrical characteristics of very complex structures to be analyzed easily.

This section presents examples that illustrate the interfaces between Medici and the following:

- TMA SUPREM-3—*Avant!* TCAD's one-dimensional process simulation program
- TSUPREM-4—*Avant!* TCAD's two-dimensional process simulation program
- Taurus-Lithography—*Avant!* TCAD's two-dimensional process simulation program for deposition, etching, and photolithography

Interfaces to Parameter Extractors

After simulating the electrical characteristics of a device structure with Medici, it is often desired to extract circuit model parameters from the various simulated I-V characteristics. Medici interfaces directly to a number of parameter extraction programs, which can be used for this purpose.

This section presents examples that illustrate the interfaces between Medici and the following:

Aurora—*Avant!* TCAD's parameter extraction program

IC-CAP—A parameter extraction program available from Hewlett-Packard

Interface to TMA SUPREM-3

The interface between Medici and TMA SUPREM-3 is illustrated by first creating the impurity profiles for a n-channel MOSFET structure with TMA SUPREM-3 and then passing these profiles to Medici to create the full device structure. Medici is used to simulate the gate characteristics for the resulting structure.

TMA SUPREM-3 Simulation

The TMA SUPREM-3 simulation of channel and source/drain profiles for a n-channel MOSFET is accomplished by the input file *s3ex9a*. The output associated

with the execution of TMA SUPREM-3 for the input *s3ex9a* is shown in Figures 10-1 through 10-6.

```
 1... TITLE       Example 9A - SUPREM-3/MEDICI Interface
 2... COMMENT     SUPREM-3 File

 3... COMMENT     Define masks.            lower      upper
  ... +                                    left       right
  ... +                                  X     Z     X   Z
 4... MASK        NAME=NITRIDE  RECT.1=(-4.0  -3.0   4.0 3.0)  CLEAR
 5... MASK        NAME=THINOX   RECT.1=(-3.5  -2.5   3.5 2.5)
 6... MASK        NAME=POLY     RECT.1=(-1.0 -10.0   1.0 4.0)  CLEAR
 7... MASK        NAME=NSD      RECT.1=(-3.0  -2.0   3.0 2.0)
 8... MASK        NAME=CONTACT  RECT.1=(-2.5  -1.5  -1.5 1.5)
  ... +                         RECT.2=( 2.5  -1.5   1.5 1.5)

 9... COMMENT     Initialize the silicon substrate.
10... INITIALIZE  <100>  SILICON  BORON=1E15
  ... +           THICKNESS=5.0  DX=.002  XDX=.02  SPACES=200

11... COMMENT     Plot and label the layout.
12... PLOT        MASK  NAME=NITRIDE  LINE=2  BOTTOM=-5
  ... +           TITLE="Example 9A - NMOS Layout"
13... PLOT        MASK  NAME=THINOX   ADD  COLOR=7  FILL
14... PLOT        MASK  NAME=NSD      ADD  COLOR=3  FILL
15... PLOT        MASK  NAME=POLY     ADD  COLOR=2  FILL
16... PLOT        MASK  NAME=CONTACT  ADD  COLOR=1  FILL

17... COMMENT     Replot without fill to show the hidden lines.
18... PLOT        MASK  NAME=NITRIDE  ADD           LINE=2
19... PLOT        MASK  NAME=THINOX   ADD  COLOR=7  LINE=3
20... PLOT        MASK  NAME=NSD      ADD  COLOR=3  LINE=4
21... LABEL       LABEL="Poly"        X=-.5    Y=3.2
22... LABEL       LABEL="Nitride"     X=-3.75  Y=3.2
23... LABEL       LABEL="Thin Oxide"  X=-3.2   Y=-2.25  ANGLE=90

24... COMMENT     Simulate the gate and source/drain regions.
25... LOOP        STEPS=2

26... COMMENT     Initialize the silicon substrate.
27... ASSIGN      NAME=XLOC  N.VALUE=(0    2)
28... ASSIGN      NAME=ZLOC  N.VALUE=(0    0)
29... INITIALIZE  <100>  SILICON  BORON=1E15  X.LOCAT=@XLOC Z.LOCAT=@ZLOC
  ... +           THICKNESS=5.0  DX=.002  XDX=.02  SPACES=200
```

Figure 10-1 First part of the simulation input file *s3ex9a*

Mask Layout

The input file *s3ex9a* shown in Figures 10-1 through 10-3 uses the masking capability that is available in TMA SUPREM-3 in order to simulate both the channel

profile and the source/drain profile from a single input file. The mask layout that is used in this simulation is set up at lines 4 through 8 and is shown in Figure 10-4.

```
30... COMMENT     Grow the pad oxide.
31... DIFFUSION   TEMPERATURE=1050  THICKNESS=.05  DRYO2

32... COMMENT     Deposit a nitride masking layer.
33... DEPOSIT     NITRIDE  THICKNESS=.2  DX=.02  SPACES=10

34... COMMENT     Pattern the nitride.
35... MASK        APPLY  NAME=NITRIDE
36... ETCH        NITRIDE
37... MASK        REMOVE

38... COMMENT     Grow the field oxide under high pressure.
39... DIFFUSION   TEMPERATURE=1000  TIME=30  STEAM  PRESSURE=5  PRINT

40... COMMENT     Etch the oxide and nitride layers.
41... ETCH        OXIDE   thick=.04
42... ETCH        NITRIDE

43... MASK        APPLY  NAME=THINOX
44... ETCH        OXIDE  THICK=.05
45... MASK        REMOVE

46... COMMENT     Implant boron to shift the threshold voltage.
47... IMPLANT     BORON  DOSE=4E12  ENERGY=50

48... COMMENT     Oxidize the gate in a dilute ambient with HCl.
49... ASSIGN      NAME=N2  N.VALUE=23.6
50... DIFFUSION   TEMPERATURE=1000 TIME=40 HCL%=5 F.O2=100-@N2 F.N2=@N2
  ... +           PRINT

51... COMMENT     Deposit and dope the polycrystalline silicon gate.
52... DEPOSIT     POLYSILICON  THICKNESS=0.5  TEMPERATURE=620
53... DIFFUSION   TEMPERATURE=950  TIME=60  SS.PHOSPHORUS  PRINT

54... COMMENT     Etch the poly and oxide over the source/drain.
55... MASK        APPLY  NAME=POLY
56... ETCH        POLYSILICON
57... ETCH        OXIDE
58... MASK        REMOVE

59... COMMENT     Implant the arsenic source/drain regions.
60... MASK        APPLY  NAME=NSD
61... IMPLANT     ARSENIC  DOSE=5E15  ENERGY=50  GAUSSIAN
62... MASK        REMOVE

63... COMMENT     Drive in the source/drain regions.
64... DIFFUSION   TEMPERATURE=1000  TIME=15  DRYO2
65... DIFFUSION   TEMPERATURE=1000  TIME=15  STEAM
66... DIFFUSION   TEMPERATURE=1000  TIME=20

67... ETCH        OXIDE
68... ETCH        POLYSILICON
69... ETCH        OXIDE
```

Figure 10-2 Second part of the simulation input file *s3ex9a*

Processing Steps

The processing steps used to create the desired profiles are located within the input statement loop between lines 25 and 83. The assigned names *XLOC* and *ZLOC* are used to specify the point within the mask layout where the processing is performed.

- For the first pass through the loop, processing is performed in the center of the channel by specifying *XLOC*=0 and *ZLOC*=0.
- For the second pass through the loop, processing is performed in the source/drain region by specifying *XLOC*=2 and *ZLOC*=0.

Files for Medici

After the required processing is complete, the resulting profiles are saved in files for Medici on the **SAVE** statement at line 82. The parameter **DEVICE** specifies that the profiles are used in a device simulation program. The channel profile is saved in a file with the identifier *S3EX9A0*. The source/drain profile is saved in a file with the identifier *S3EX9A2*.

```
70... COMMENT     Plot the resulting chemical impurity distributions.
71... ASSIGN      NAME=TSTRING  C1="NMOS Channel"  C2="NMOS Source/Drain"
72... PLOT        ACTIVE  NET  RIGHT=1.5
  ... +           TITLE="Example 9A - "@TSTRING" x="@XLOC", z="@ZLOC
73... PLOT        ACTIVE  BORON      ADD  COLOR=2  LINE=2
74... PLOT        ACTIVE  PHOSPHOR   ADD  COLOR=3  LINE=3
75... PLOT        ACTIVE  ARSENIC    ADD  COLOR=4  LINE=4
76... PLOT        ACTIVE  NET        ADD  COLOR=1  LINE=1
77... LABEL       LABEL="Active Net"  START.LE  LX.F=.9  X=1.1    Y=1e20
78... LABEL       LABEL="Boron"       START.LE  LX.F=.9  COLOR=2  LINE=2
79... LABEL       LABEL="Phosphorus"  START.LE  LX.F=.9  COLOR=3  LINE=3
80... LABEL       LABEL="Arsenic"     START.LE  LX.F=.9  COLOR=4  LINE=4

81... COMMENT     REG_0 - CHANNEL, REG_2 - SOURCE/DRAIN
82... SAVE        FILE="S3EX9A"@XLOC  DEVICE

83... L.END
```

Figure 10-3 Third part of the simulation input file *s3ex9a*

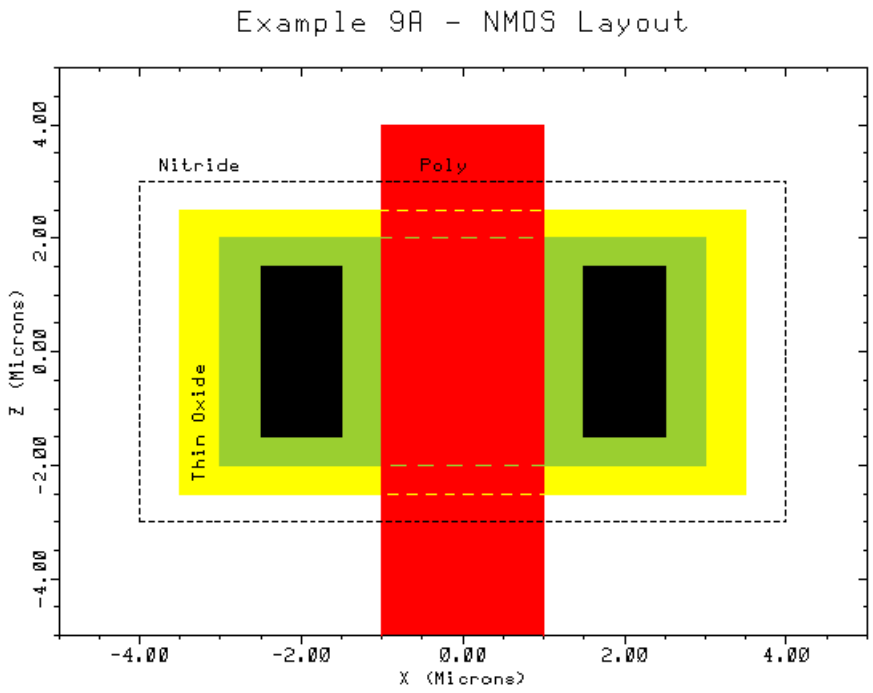


Figure 10-4 Mask layout used in file *s3ex9a*, Figures 10-1 through 10-3

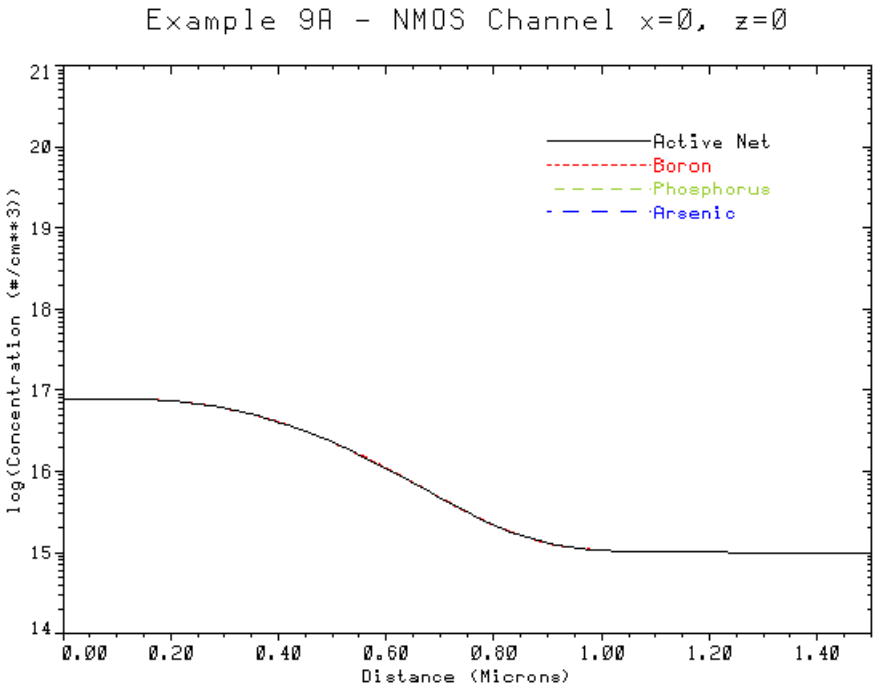


Figure 10-5 Channel profile from file *s3ex9a*, Figures 10-1 through 10-3

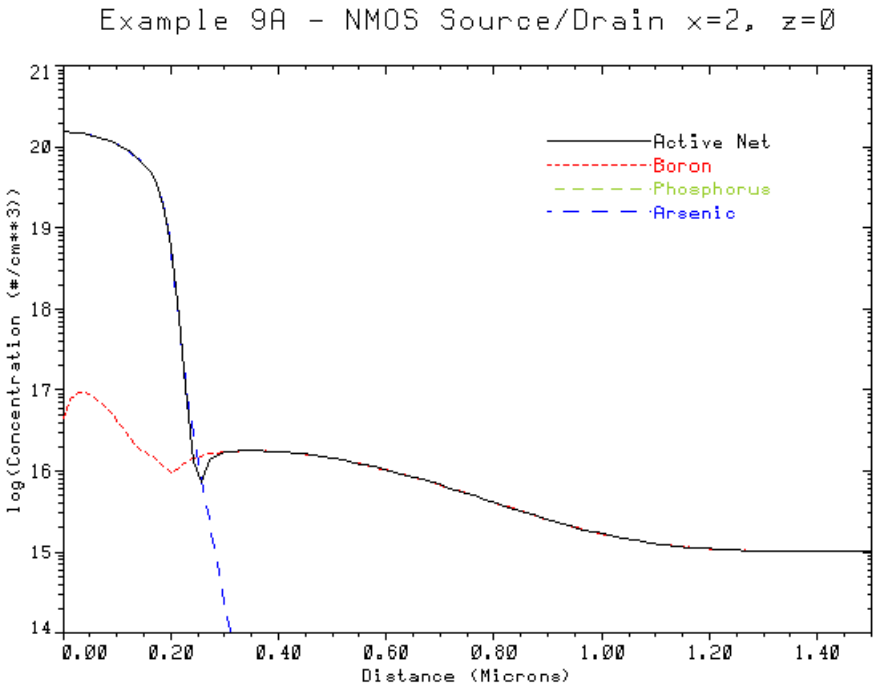


Figure 10-6 Source/drain profile from file *s3ex9a*, Figures 10-1 through 10-3

Medici Simulation of Gate Characteristics

So far, TMA SUPREM-3 has been used to create one-dimensional profiles for the channel and source/drain regions of an n-channel MOSFET. Now these profiles are used to describe the two-dimensional impurity distribution in a Medici device for the simulation of the device's gate characteristics.

The input file *mdex9a* is used to create the device structure for a 1.4 micron channel length MOSFET that uses the profiles generated by TMA SUPREM-3.

```
  1... TITLE     Example 9A - SUPREM-3/MEDICI Interface
  2... COMMENT   MEDICI Input File

  3... COMMENT   Specify a rectangular mesh.  Parameterize file with
   ... +         channel length:  LENGTH.
  4... ASSIGN    NAME=LENGTH  N.VALUE=1.4

  5... MESH      SMOOTH=1

  6... X.MESH    WIDTH=0.8  H2=.05  RATIO=1.25
  7... X.MESH    WIDTH=@LENGTH/2  H1=.05  RATIO=1.25
  8... X.MESH    WIDTH=@LENGTH/2  H2=.05  RATIO=1.25
  9... X.MESH    WIDTH=0.8  H1=.05  RATIO=1.25

 10... Y.MESH    DEPTH=0.0125  N.SPACES=2  Y.MIN=-0.0125
 11... Y.MESH    DEPTH=0.3  H1=0.01  H2=0.05
 12... Y.MESH    DEPTH=2.2  H1=0.05  RATIO=1.25
```

Figure 10-7 First part of the simulation input file *mdex9a*

Figures 10-7 through 10-13 contain the output associated with the execution of Medici for the input file *mdex9a*.

Device Specification

In this example the grid, electrodes, and regions are explicitly specified, as they were for the structure created in Figure 4-3 in Chapter 4, Generation of the Simulation Structure on page 4-2. The difference is that in this example, the impurity profiles generated by the TMA SUPREM-3 program are used.

Using TMA SUPREM-3 Profiles

To read a profile from TMA SUPREM-3 it is only necessary to specify the **1D.PROC** parameter on the **PROFILE** statement, and to use the **IN.FILE** parameter to specify the file containing the profile. The doping profiles are defined in this example as follows:

- The one-dimensional profiles created by TMA SUPREM-3 are oriented along the vertical axis in Medici.

  By default, the origin for the impurity profile is aligned with the vertical origin in Medici.
- The horizontal extent of the profile is determined by specifying the parameters **X.PEAK** (synonym of **X.MIN**) and **WIDTH**.
- The impurity profile is assumed to be constant in the horizontal direction in the region between the locations **X.PEAK** (synonym of **X.MIN**) and **X.PEAK**+**WIDTH**.
- Outside of this region, the profile falls off as a Gaussian if **X.CHAR** is specified, as the difference of two complimentary error functions if **X.ERFC** is

specified, or in a manner determined by rotating the vertical profile if **XY.RATIO** is specified.

```
13... COMMENT   Eliminate some unnecessary nodes
14... ELIMIN    COLUMNS  Y.MIN=1.1

15... COMMENT   Specify oxide and silicon regions
16... REGION    NAME=Silicon  SILICON
17... REGION    NAME=Oxide  OXIDE    Y.MAX=0

18... COMMENT   Electrodes:
19... ELECTR    NAME=Gate  X.MIN=0.65  X.MAX=0.95+@LENGTH  TOP
20... ELECTR    NAME=Substrate  BOTTOM
21... ELECTR    NAME=Source  X.MAX=0.5          Y.MAX=0
22... ELECTR    NAME=Drain  X.MIN=1.1+@LENGTH  Y.MAX=0

23... COMMENT   Specify impurity profiles and fixed charge
24... PROFILE   P-TYPE  1D.PROC  IN.FILE=S3EX9A0
25... PROFILE   N-TYPE  1D.PROC  IN.FILE=S3EX9A2  DEPTH=0.4
  ... +         X.MIN=0.0  WIDTH=0.6  XY.RATIO=0.75
26... PROFILE   N-TYPE  1D.PROC  IN.FILE=S3EX9A2  DEPTH=0.4
  ... +         X.MIN=1.0+@LENGTH  WIDTH=0.6  XY.RATIO=0.75
27... INTERFAC  QF=1E10

28... COMMENT   Plots of structure and impurity profiles
29... PLOT.2D   GRID  FILL  SCALE  TITLE="Example 9A  - Grid Structure"
30... PLOT.1D   DOPING  X.START=0  X.END=0  Y.START=0  Y.END=2.5
  ... +         Y.LOG  POINTS  BOT=1E14  TOP=1E21  COLOR=2
  ... +         TITLE="Example 9A - Source Impurity Profile"
31... PLOT.1D   DOPING  X.START=0.8+@LENGTH/2  X.END=0.8+@LENGTH/2
  ... +         Y.START=0  Y.END=2.5  Y.LOG  POINTS  BOT=1E14  TOP=1E18
  ... +         TITLE="Example 9A - Gate Impurity Profile"  COLOR=2
32... PLOT.2D   BOUND  FILL  SCALE
  ... +         TITLE="Example 9A - Impurity Contours"
33... CONTOUR   DOPING  LOG  MIN=16   MAX=20   DEL=.5  COLOR=2
34... CONTOUR   DOPING  LOG  MIN=-16  MAX=-15  DEL=.5  COLOR=1  LINE=2

35... COMMENT   Specify contact parameters and physical models
36... CONTACT   NAME=Gate  N.POLY
37... MODELS    CONMOB  PRPMOB  FLDMOB  CONSRH  AUGER  BGN

38... COMMENT   Specify a 1-carrier solution for electrons
39... SYMB      CARRIERS=1  ELECTRONS  NEWTON

40... COMMENT   Open log file for IV data, ramp the gate voltage
41... LOG       OUT.FILE=MDEX9AI
42... SOLVE     V(Drain)=0.1  V(Gate)=0.0  ELEC=Gate
  ... +         VSTEP=0.2  NSTEP=10

43... COMMENT   Plot Ids vs. Vgs
44... PLOT.1D   Y.AXIS=I(Drain)  X.AXIS=V(Gate)  POINTS  COLOR=2
  ... +         TITLE="Example 9A - Gate Characteristics"
45... LABEL      LABEL="Vds = 0.1v"
```

Figure 10-8 Second part of the simulation input file *mdex9a*

- The first **PROFILE** statement at line 24 in the input file *mdex9a* specifies that the channel profile stored in the file *S3EX9A0* is to be read in.
- Because neither **X.PEAK** and **WIDTH** is specified, the profile is assumed to span the entire device width.
- The **INTERFACE** statement at line 27 sets the fixed charge at the silicon dioxide interface equal to 1E10 cm^{-2}.
- The source/drain profile stored in the file *S3EX9A2* is used to define the source and drain regions of the MOSFET on the **PROFILE** statements at lines 25 and 26, respectively.

Device Plots

Figures 10-9 through 10-12 show the simulation mesh and impurity profiles for the final structure.

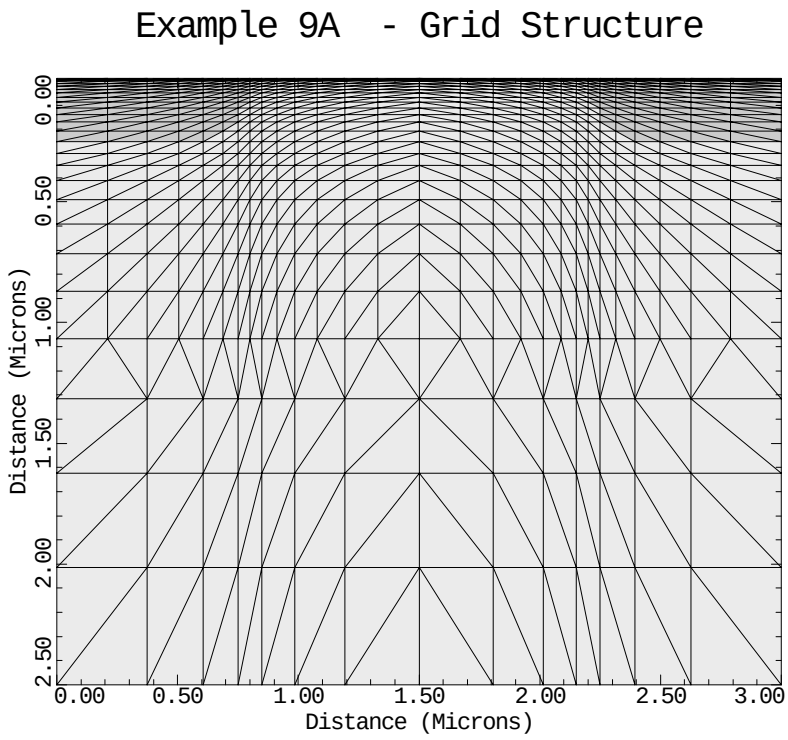


Figure 10-9 Grid Structure from **PLOT.2D** at line 29 in file *mdex9a*, Figures 10-7 and 10-8

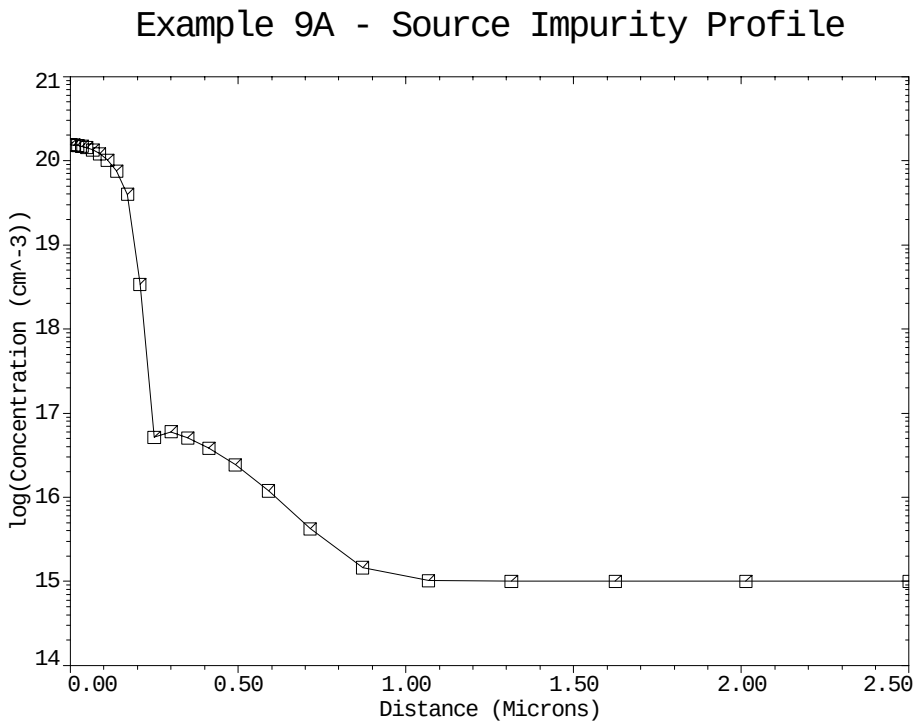


Figure 10-10 Source impurity profile from **PLOT.1D** at line 30 in file *mdex9a,* in Figures 10-7 and 10-8

Gate Characteristics Simulation

The remainder of the input file is used to simulate the gate characteristics of the device. The I-V results are stored in the log file *MDEX9AI* and a plot of the results is shown in Figure 10-13.

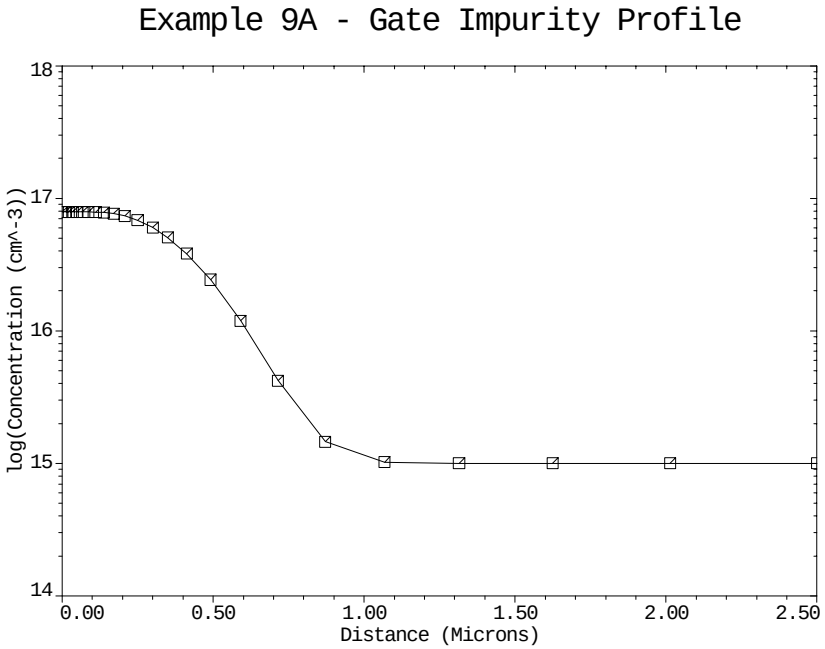


Figure 10-11 Gate impurity profile from **PLOT.1D** at line 31 in input file *mdex9a*, Figures 10-7 and 10-8

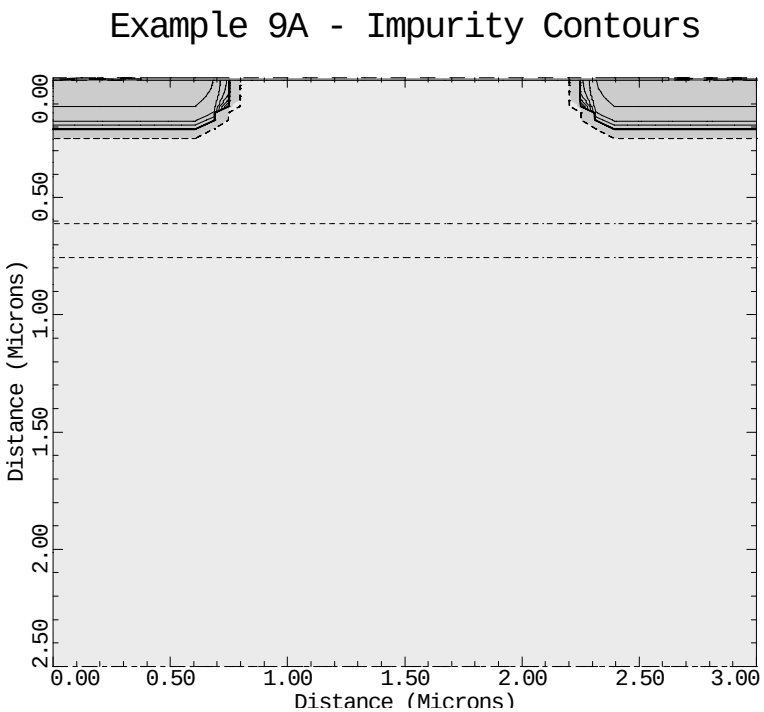


Figure 10-12 Impurity profile from **PLOT.2D** and **CONTOUR** at lines 32 through 34 in file *mdex9a*, Figures 10-7 and 10-8

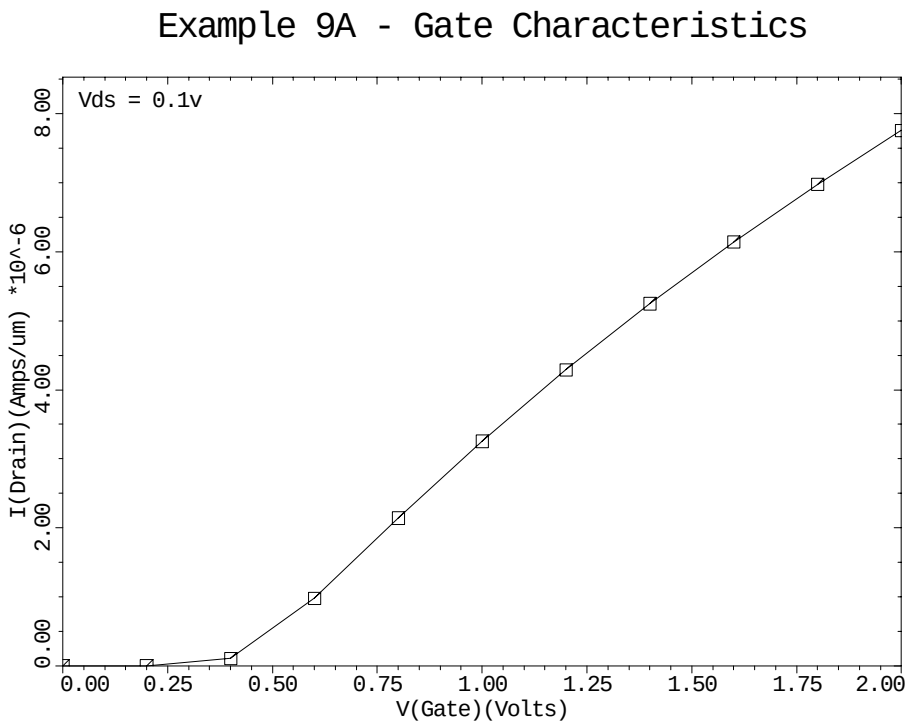


Figure 10-13 Gate characteristics from **PLOT.1D** and **LABEL** at lines 44 through 45 in file *mdex9a*, Figures 10-7 and 10-8

Interface to TSUPREM-4

The interface between Medici and TSUPREM-4 is illustrated by first performing the processing associated with creating an n-channel MOSFET in TSUPREM-4. The resulting two-dimensional structure (including the mesh, boundaries, and impurity profiles) are then passed to Medici. In Medici, a simulation of the drain characteristics for the device are performed.

TSUPREM-4 Simulation

The input file *s4ex9b* is used to simulate the processing of a two-dimensional cross-section of an n-channel MOSFET using TSUPREM-4. The output associated with the execution of TSUPREM-4 for the input file *s4ex9b* is shown in Figures 10-14 and 10-17.

Initial Mesh

The **LINE** statements at the beginning of the input file specify the initial mesh for the left-hand side of the MOSFET. Since the final MOSFET structure in this example is symmetric about a vertical axis passing through the center of the channel, it is only necessary to simulate half of the structure. This results in a significant savings in CPU time.

Medici Considerations

In this example, the entire structure created as a result of the TSUPREM-4 simulation is passed to Medici. This not only includes the impurity profiles, but also

the structure boundary and grid. Because the grid created by TSUPREM-4 is also used in Medici, it is important that some care is taken to make sure that the grid is adequate for device simulation.

Note:

The most important consideration in this regard is to make sure that the vertical grid spacing in the channel is adequate for device simulation. In most cases, a vertical spacing of approximately 100 Å is sufficient. Figure 10-16 shows the initial TSUPREM-4 ***mesh.***

Processing Steps

After creating the initial mesh structure, the remainder of the file goes through the processing steps necessary to create the MOSFET. Metallization is the final step for this simulation, before saving the structure for Medici. This step is intended to make the source/drain contact that is used by Medici.

Electrodes

When the final structure is passed to Medici, all aluminum regions are automatically converted to electrodes. In addition, any aluminum node, within the aluminum region, that does not contact another nonelectrode material region is eliminated from the structure. This helps to reduce the total node count while still providing the electrode boundaries for Medici.

Note:

The source and drain electrode contacts could also have been made in Medici ***itself. After passing the*** TSUPREM-4 ***structure into*** Medici***, specify*** **ELECTROD** ***statements that place the electrodes at the desired locations.***

Completing the Device Structure

The TSUPREM-4 simulation is done on half of the device while the Medici simulation needs the entire device. Because of this the final step in this example is to create the complete device structure by using the **REFLECT** parameter on the **STRUCTUR** statement. The **MEDICI** parameter on the second **STRUCTUR** state-

ment causes TSUPREM-4 to save the final structure in a form that Medici can read.

```
COMMENT   Example 9B - TSUPREM-4/MEDICI Interface
COMMENT   TSUPREM-4 Input File

OPTION    DEVICE=PS

COMMENT   Specify the mesh
LINE  X   LOCATION=0    SPACING=0.20
LINE  X   LOCATION=0.9  SPACING=0.06
LINE  X   LOCATION=1.8  SPACING=0.2

LINE  Y   LOCATION=0    SPACING=0.01
LINE  Y   LOCATION=0.1  SPACING=0.01
LINE  Y   LOCATION=0.5  SPACING=0.10
LINE  Y   LOCATION=1.5  SPACING=0.2
LINE  Y   LOCATION=3.0  SPACING=1.0

ELIMIN    ROWS  X.MIN=0.0  X.MAX=0.7  Y.MIN=0.0   Y.MAX=0.15
ELIMIN    ROWS  X.MIN=0.0  X.MAX=0.7  Y.MIN=0.06  Y.MAX=0.20
ELIMIN    COL   X.MIN=0.8  Y.MIN=1.0

COMMENT   Initialize  and plot mesh structure
INITIALIZ <100>  BORON=1E15
SELECT    TITLE="TSUPREM-4:  Initial Mesh"
PLOT.2D   GRID

COMMENT   Initial oxide
DEPOSIT   OXIDE  THICKNESS=0.03

COMMENT   Models selection.  For this simple example, the OED
COMMENT   model is not turned on (to reduce CPU time).
METHOD    VERTICAL

COMMENT   P-well implant
IMPLANT   BORON  DOSE=3E13  ENERGY=45

COMMENT   P-well drive
DIFFUSE   TEMP=1100  TIME=500  DRYO2  PRESS=0.02
ETCH      OXIDE ALL

COMMENT   Pad oxidation
DIFFUSE   TEMP=900  TIME=20  DRYO2

COMMENT   Pad nitride

DEPOSIT   NITRIDE  THICKNESS=0.1

COMMENT   Field oxidation
DIFFUSE   TEMP=1000  TIME=360  WETO2
ETCH      NITRIDE  ALL

COMMENT   Vt adjust implant
IMPLANT   BORON  ENERGY=40  DOSE=1E12
ETCH      OXIDE  ALL
```

Figure 10-14 First part of the TSUPREM-4 input file *s4ex9b*

```
COMMENT   Gate oxidation
DIFFUSE   TEMP=900  TIME=35  DRYO2
DEPOSIT   POLYSILICON  THICKNESS=0.3  DIVISIONS=4

COMMENT   Poly and oxide etch
ETCH      POLY   LEFT  P1.X=0.8  P1.Y=-0.5  P2.X=0.8  P2.Y=0.5
ETCH      OXIDE  LEFT  P1.X=0.8  P1.Y=-0.5  P2.X=0.8  P2.Y=0.5
DEPOSIT   OXIDE  THICKNESS=0.02

COMMENT   LDD implant
IMPLANT   PHOS  ENERGY=50  DOSE=5E13

COMMENT   LTO
DEPOSIT   OXIDE  THICK=0.2  DIVISIONS=10

COMMENT   Spacer etch
ETCH      OXIDE  DRY  THICK=0.22

COMMENT   S/D implant
IMPLANT   ARSENIC  ENERGY=100  DOSE=2E15

COMMENT   Oxide etch
ETCH      OXIDE  LEFT  P1.X=0.5

COMMENT   S/D reoxidation
DIFFUSE   TEMP=950  TIME=30  DRYO2  PRESS=0.02

COMMENT   BPSG
DEPOSIT   OXIDE  THICK=0.3
ETCH      OXIDE  LEFT  P1.X=0.3  P1.Y=-2  P2.Y=2

SELECT    Z=LOG10(DOPING)  TITLE="TSUPREM-4:  S/D Doping Profile"
PLOT.1D   X.VALUE=0  LINE.TYP=5  BOUNDARY  Y.MIN=14  Y.MAX=21

COMMENT   Metallization
DEPOSIT   ALUMINUM  THICK=0.5  SPACES=3
ETCH      ALUMINUM  RIGHT  P1.X=0.6  P2.X=0.55  P1.Y=-2  P2.Y=2

COMMENT   Save simulation results
STRUCTUR  REFLECT  RIGHT
STRUCTUR  MEDICI   OUT.FILE=S4EX9BS
```

Figure 10-15 Second part of the TSUPREM-4 input file *s4ex9b*

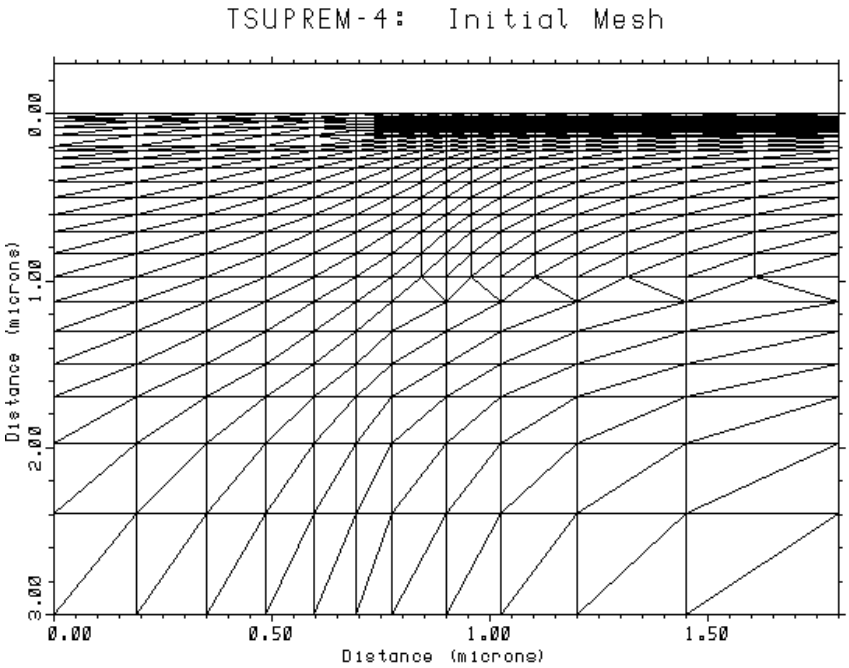


Figure 10-16 TSUPREM-4 initial mesh in file *s4ex9b*, Figures 10-14 and 10-15

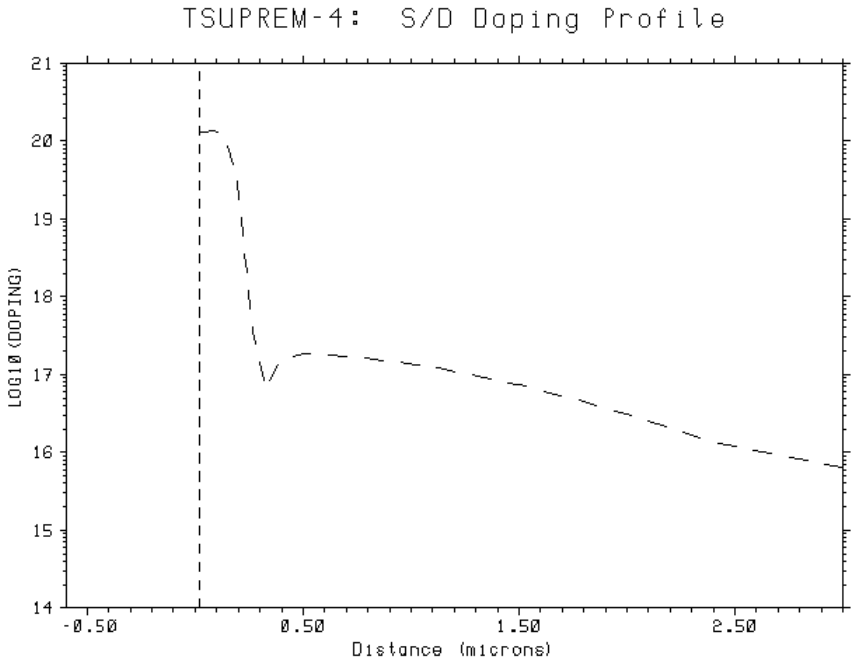


Figure 10-17 TSUPREM-4 source/drain profile in file *s4ex9b*, Figures 10-14 and 10-15

Medici Simulation

There are a number of ways the results of a TSUPREM-4 simulation is used in Medici.

- Create the grid entirely in Medici using the usual meshing statements and only read the profile information from the TSUPREM-4 results using the **PROFILE** statement.

  Advantages—There is full control over the grid used in Medici, allowing an optimal grid for device simulation to be created.

  Disadvantage—It is difficult to reproduce complex topographies.

- Create an initial rectangular mesh in Medici and then use the **BOUNDARY** statement to read the topography from TSUPREM-4. Medici then modifies the initial mesh to assure that the topography is well represented by the mesh. The profile information from TSUPREM-4 can then be read in using the **PROFILE** statement.

  This method has the advantage that you have some control over the grid used in Medici, but also assures that the structure topography is captured.

- Use the complete structure created by TSUPREM-4, including the grid, boundaries, and profile information, by reading the TSUPREM-4 results on the **MESH** statement in Medici.

  This is the simplest method available and is recommended in most cases as long as the TSUPREM-4 mesh is adequate for device simulation.

This example uses the last method described above.

Gate Characteristics Simulation

The input file *mdex9b* is used to read the results of the TSUPREM-4 simulation for an n-channel MOSFET and then performs a simulation of the drain characteristics for the device. Figures 10-18 through 10-24 contain the output associated with the execution of Medici for the input file *mdex9b.*

Structure Definition with TSUPREM-4

The **MESH** statement reads the structure created by TSUPREM-4.

The following parameters are used in the simulation:

- The parameter **TSUPREM4** indicates that the structure file specified with the **IN.FILE** parameter was created by TSUPREM-4.
- The **RENAME** statement is used to provide meaningful Medici names read in from TSUPREM-4.
- The **Y.MAX** parameter indicates that only the portion of the structure with *y*-coordinates less than 3 microns is read into Medici.
- The **ELEC.BOT** parameter indicates that an electrode should be placed along the bottom of the structure.

  This is the substrate contact for the device.

- The parameter **POLY.ELEC** indicates that all polysilicon regions in the TSUPREM-4 structure should be converted to electrodes in Medici.

This causes the polysilicon gate material in TSUPREM-4 to be treated as an ideal conductor in Medici. If **^POLY.ELEC** is specified, polysilicon is treated as a semiconductor material.

- The **OUT.FILE** parameter on the **SAVE** statement at line 10 is used to specify the file where the resulting Medici structure is stored.

Examining the Device Structure

Once the structure is read in, examine the portion of the output listing generated by the **MESH** statement. This is shown in Figure 10-19. The output indicates the different material types for each region, as well as the minimum and maximum coordinates for each region and electrode.

This makes it possible to identify the region number corresponding to each material and also the electrode number corresponding to the following:

- Source
- Drain
- Gate
- Substrate

Note:

Region 3 has a material type of* ELECTROD*. This represents the* TSUPREM-4 *polysilicon region that was converted to an electrode as a result of the* POLY.ELEC *parameter.

Device Plots

Figures 10-20 through 10-23 show the simulation mesh and impurity profiles for the final Medici structure.

Drain Characteristics Simulation

The remainder of the input file is used to simulate the drain characteristics of the device. The I-V results are stored in the log file *MDEX9BI* and a plot of the results is shown in Figure 10-24.

```
  1... TITLE     Example 9B - TSUPREM-4/MEDICI Interface
  2... COMMENT   MEDICI Input File
  3... COMMENT   Simulation of NMOS device output characteristics

  4... COMMENT   Read in simulation mesh
  5... MESH      IN.FILE=S4EX9BS  TSUPREM4  ELEC.BOT  POLY.ELEC  Y.MAX=3

  6... COMMENT   Rename some electrodes from TSUPREM-4 to standard names.
  7... RENAME    ELECTR  OLDNAME=1  NEWNAME=Source
  8... RENAME    ELECTR  OLDNAME=1  NEWNAME=Source

  9... COMMENT   Save the mesh with the new electrode names
 10... SAVE      MESH  OUT.FILE=MDEX9BM

 11... CONTACT   NUMBER=Gate  N.POLY
 12... MODELS    CONMOB  PRPMOB  FLDMOB  CONSRH  AUGER  BGN

 13... PLOT.2D   GRID  SCALE  FILL  TITLE="Structure from TSUPREM-4"
 14... PLOT.1D   DOPING  LOG  X.START=0  X.END=0  Y.START=0  Y.END=2
   ... +         POINTS  BOT=1E14  TOP=1E21  TITLE="S/D Profile"
 15... PLOT.1D   DOPING  LOG  X.START=1.8  X.END=1.8  Y.START=0  Y.END=2
   ... +         POINTS  BOT=1E14  TOP=1E19  TITLE="Channel Profile"
 16... PLOT.2D   BOUND  SCALE  FILL  L.ELEC=-1  TITLE="Impurity Contours"
 17... CONTOUR   DOPING  LOG  MIN=14   MAX=20   DEL=1  COLOR=2
 18... CONTOUR   DOPING  LOG  MIN=-20  MAX=-14  DEL=1  COLOR=1  LINE=2

 19... COMMENT   Simulate a drain curve with Vg=2v
 20... SYMB      CARR=0
 21... METHOD    ICCG  DAMPED
 22... SOLVE     V(Gate)=2
 23... SYMB      CARR=1  NEWTON  ELECTRON
 24... LOG       OUT.FILE=MDEX9BI
 25... SOLVE     V(Drain)=0.0  ELEC=Drain  VSTEP=0.1  NSTEP=2
 26... SOLVE     V(Drain)=0.5  ELEC=Drain  VSTEP=0.5  NSTEP=5

 27... COMMENT   Plot results
 28... PLOT.1D   X.AXIS=V(Drain)  Y.AXIS=I(Drain)  TOP=2.2E-5
   ... +         TITLE="Ids vs. Vds"  COLOR=2  POINTS
 29... LABEL     LABEL="Vgs = 2v"  COLOR=2
```

Figure 10-18 Output of the simulation input file *mdex9b*

```
Read TSUPREM-4 file from S4EX9BS

Mesh statistics (rectangular) :
    Total grid points = 1549
    Total no. of triangles = 2966
    Obtuse triangles =  149  ( 5.0%)

  Region    Material      X-min        X-max        Y-min        Y-max
   Name       Type      (microns)    (microns)    (microns)    (microns)
----------  --------    ---------    ---------    ---------    ---------
1           Silicon        0.0000       3.6000       0.0215       3.0000
2           Oxide          0.3000       3.3000      -0.5942       0.0223
3           Electrod       0.8000       2.8000      -0.2922       0.0078

Electrode    Number       X-min        X-max        Y-min        Y-max
   Name     of Nodes    (microns)    (microns)    (microns)    (microns)
----------  --------    ---------    ---------    ---------    ---------
1              14          0.0000       0.5818      -0.5447       0.0223
2              14          3.0182       3.6000      -0.5447       0.0223
Poly          149          0.8000       2.8000      -0.2922       0.0078
Sub            23          0.0000       3.6000       3.0000       3.0000

Renamed Electrode:  1 ==> Source

Renamed Electrode:  2 ==> Source

Grid written to MDEX9BM
```

Figure 10-19 Output from **MESH** at line 5 in file *mdex9b* shown in Figure 10-18

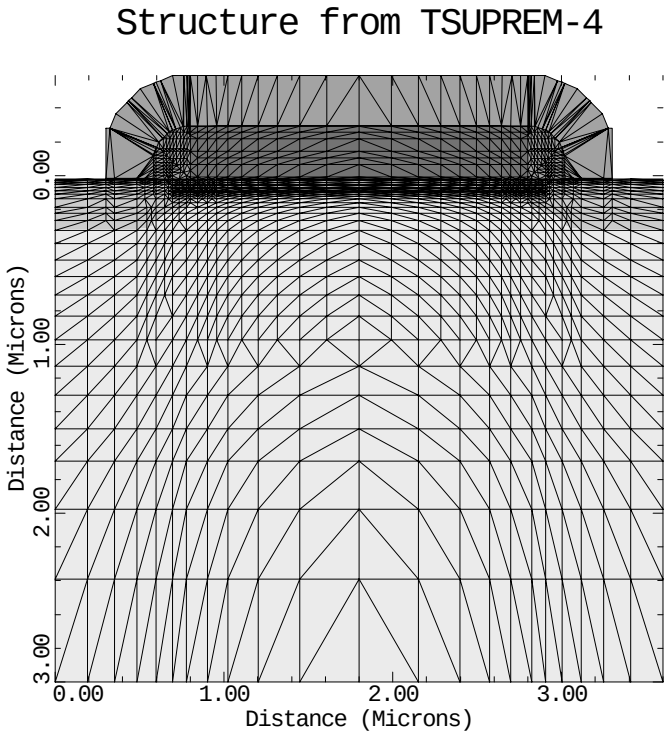


Figure 10-20 TSUPREM-4 structure from **PLOT.2D** at line 13 in file *mdex9b*, Figure 10-18

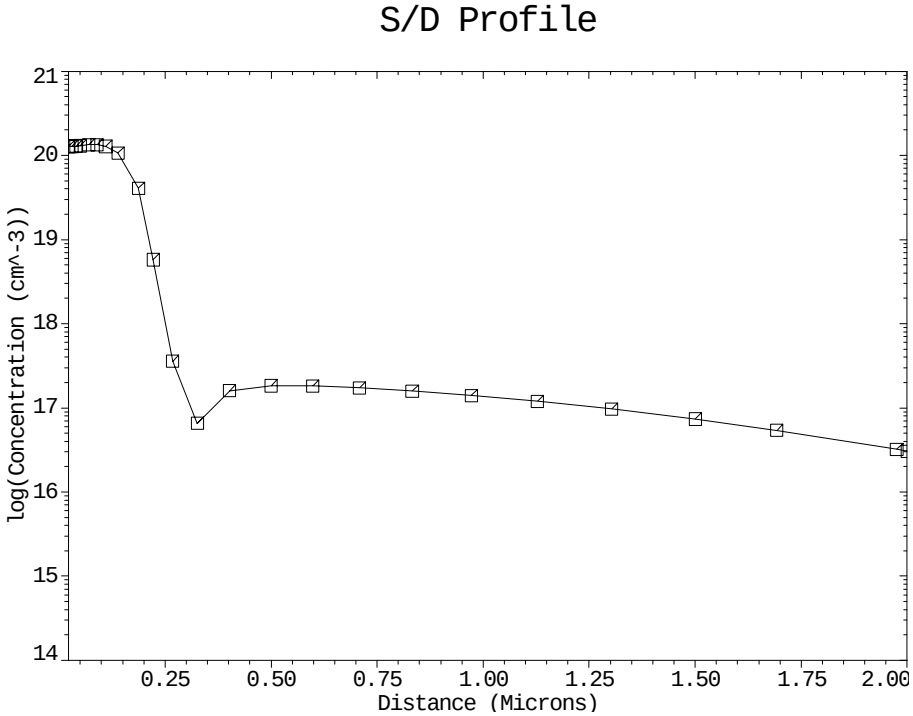


Figure 10-21 Source/Drain profile from **PLOT.1D** statement at line14 in file *mdex9b*, Figure 10-18

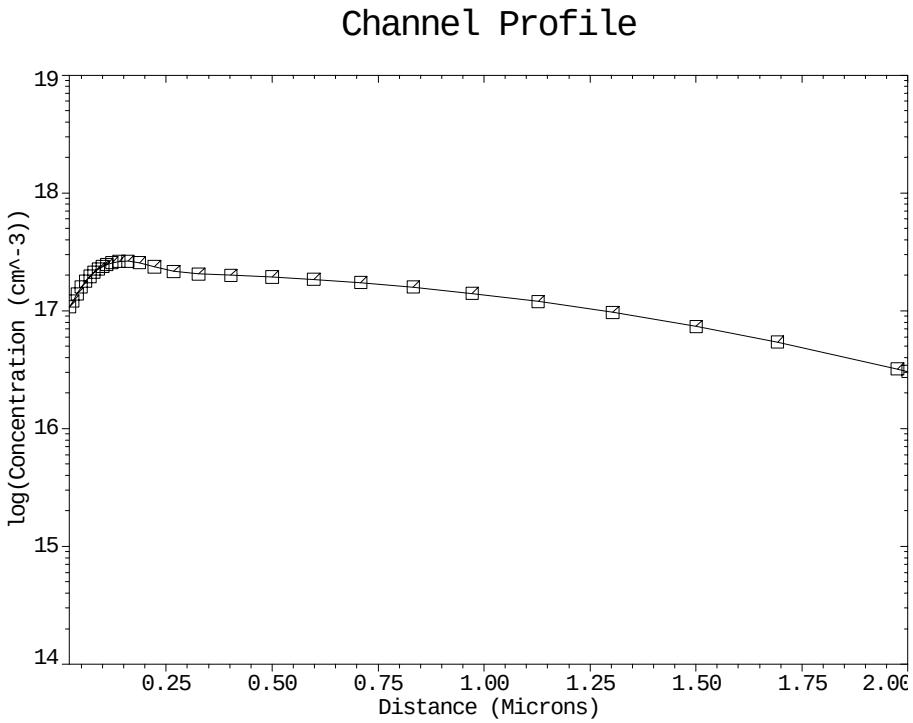


Figure 10-22 Channel profile from **PLOT.1D** at line15 in file *mdex9b*, Figure 10-18

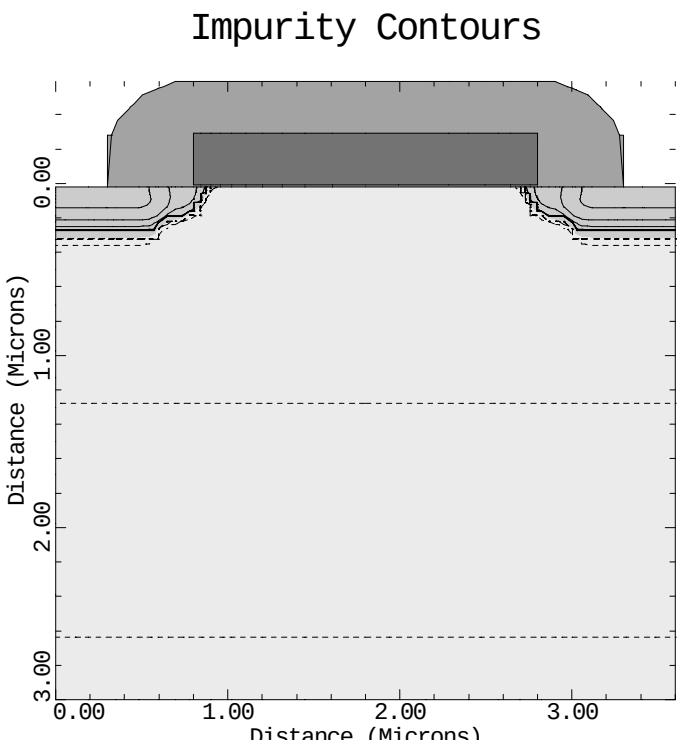


Figure 10-23 Impurity contours from **PLOT.1D** and **CONTOUR** at lines 16 through 18 in file *mdex9b*, Figure 10-18

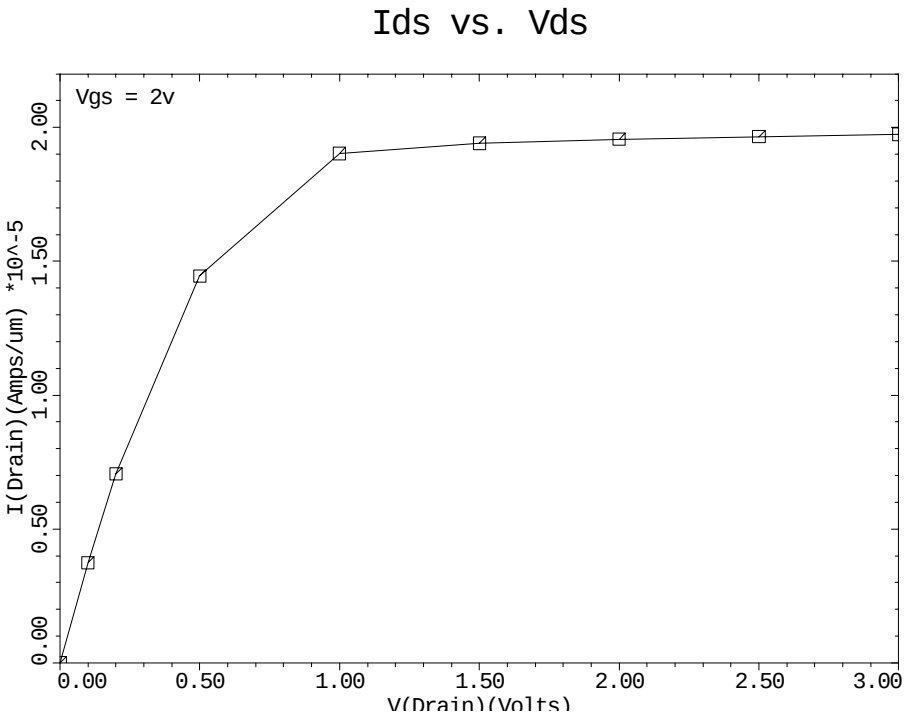


Figure 10-24 Ics vs. Vds from **PLOT.1D** and **LABEL** at lines 28 and 29 in file *mdex9b*, Figure 10-18

Interface to Taurus-Lithography

The interface between Medici and Taurus-Lithography is illustrated by first creating the topography for an interconnect structure using Taurus-Lithography and then passing the topography information to Medici. In Medici, the potential and electric field distributions of the structure are analyzed.

Taurus-Lithography Simulation

The Taurus-Lithography simulation of a two-dimensional cross-section of an interconnect structure is accomplished by the input file *deex9c*. The output associated with the execution of Taurus-Lithography for the input file *deex9c* is shown in Figures 10-25 and 10-26.

The structure created by the input file *deex9c* and shown in Figure 10-25 consists of a silicon substrate on top of which three poly lines are patterned. The oxide is assumed to be deposited with the thermal decomposition of silane. This deposition is simulated using the hemispherical deposition machine. A metal crossing line is then created by depositing aluminum. The resulting structure is shown in Figure 10-26.

Files for Medici

A **SAVE** statement is used to save the topography information for Medici. The **SUPRA** parameter specifies that the output file is saved in a format that can be read by TMA SUPRA, which is also the same format that can be read by Medici.

```
 1... TITLE     Example 9C - Taurus-Lithography/MEDICI Interface
 2... COMMENT   Taurus-Lithography Input File

 3... INITIAL   WIDTH=4.5
 4... DEPOSIT   MAT=SILICON THICK=1
 5... DEPOSIT   MAT=POLY    THICK=0.5
 6... ETCH      MAT=POLY    COORD=(0.5,0 1.5,0)
 7... ETCH      MAT=POLY    COORD=(2.75,1.5 3,1 4,1 4.25,1.5)
 8... SAVE      STRUCTURE  FILE=TEMP

 9... TITLE     Taurus-Lithography - Silane Process
10... LOAD      STRUCTURE FILE=TEMP

11... DEPOSIT   MAT=OXIDE    MACH=HEMI  RATE=0.5  TIME=1
12... DEPOSIT   MAT=ALUMINU  MACH=HEMI  RATE=0.5  TIME=1

13... PLOT      SCALE
14... SHADE     ALL
15... LABEL     LABEL=SILICON   X=1.8  Y=.75
16... LABEL     LABEL=POLY      X=1.8  Y=1.2
17... LABEL     LABEL=OXIDE     X=1.8  Y=1.7
18... LABEL     LABEL=ALUMINUM  X=1.8  Y=2.2

19... SAVE      SUPRA  STRUCTURE  FILE=DEEX9CS
```

Figure 10-25 Taurus-Lithography output of file *deex9c*

.

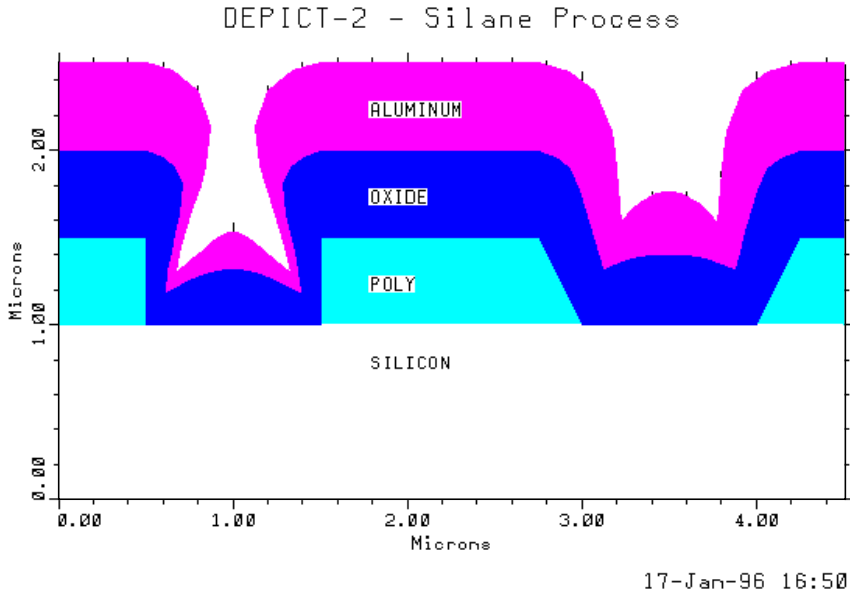


Figure 10-26 Taurus-Lithography plot created by file *deex9c*, Figure 10-25

Medici Simulation of Electric Field Distribution

The input file *mdex9c* is used to read the interconnect structure created by Taurus-Lithography into Medici, and a solution is obtained with -10V applied to the aluminum line. The resulting potential and electric field distributions in the struc-

ture are plotted. The output associated with the execution of Medici for the input file *mdex9c* is shown in Figures 10-27 through 10-31.

An initial rectangular mesh is created using the **X.MESH** and **Y.MESH** statements.

Topology from Taurus-Lithography

Then the topography information created by Taurus-Lithography is read by Medici using the **BOUNDARY** statement. The **BOUNDARY** statement itself should specify the file created and saved by Taurus-Lithography with the **IN.FILE** parameter. The parameter **2D.PROC** is used to specify that the file was created using a *Avant!* TCAD two-dimensional process simulation program.

Note:

When boundary information is read into Medici ***using the*** **2D.PROC** ***parameter, it is necessary to explicitly define each region and electrode in the structure. This can be accomplished quite easily, however, by using the*** **X** ***and*** **Y** ***parameters on the*** **REGION** ***and*** **ELECTROD** ***statements to simply point to a location within the area that encloses the region or electrode to be defined.***

Regrid

After the structure specification is complete, two regrids based on potential are performed when -10V is applied to the aluminum line. This causes more nodes to be added to the mesh in the high field regions in the oxide.

Plots

The initial and refined simulation meshes are shown in Figures 10-28 and 10-29. A final solution is obtained and the potential and electric field distributions are plotted. The results are shown in Figures 10-30 and 10-31.

```
 1... COMMENT   Example 9C - Taurus-Lithography/MEDICI Interface
 2... COMMENT   MEDICI Input File

 3... COMMENT   Grid structure
 4... MESH
 5... X.MESH    WIDTH=4.5  N.SPACES=18
 6... Y.MESH    DEPTH=1.7  N.SPACES=12  Y.MIN=-1.7
 7... Y.MESH    DEPTH=1.0  N.SPACES=7
 8... Y.MESH    DEPTH=0.2  N.SPACES=1

 9... COMMENT   Taurus-Lithography boundaries
10... BOUNDARY  2D.PROC  IN.FILE=DEEX9CS

11... COMMENT   Regions:
12... REGION    NAME=Silicon  X=1.0  Y= 0.2  SILICON
13... REGION    NAME=Oxide1  X=1.0  Y=-0.3  OXIDE
14... REGION    NAME=Oxide2  X=1.0  Y=-1.6  OXIDE
15... REGION    NAME=Poly1  X=0.2  Y=-0.2  POLY
16... REGION    NAME=Poly2  X=2.0  Y=-0.2  POLY
17... REGION    NAME=Poly3  X=4.3  Y=-0.2  POLY
18... REGION    NAME=Aluminum  X=0.2  Y=-1.2  POLY
19... REGION    NAME=Substrate  X=1.0  Y= 1.1  POLY
20... COMMENT   Electrodes:
21... ELECTROD  NAME=Substrate  X=1.0  Y=1.1   VOID
22... ELECTROD  NAME=Poly1  X=0.2  Y=-0.2
23... ELECTROD  NAME=Poly2  X=2.0  Y=-0.2
24... ELECTROD  NAME=Poly3  X=4.3  Y=-0.2
25... ELECTROD  NAME=Aluminum  X=0.2  Y=-1.2

26... COMMENT   Doping profiles
27... PROFILE   N-TYPE  N.PEAK=1.E15  UNIFORM

28... COMMENT  Plots
29... PLOT.2D  GRID  SCALE  FILL
  ... +        TITLE="Example 9C - Silane Process from T-Lithography"


30... COMMENT  Set V(Aluminum)=-10v, regrid on potential
31... SYMB     CARR=0
32... METHOD   DAMPED
33... SOLVE    V(Aluminum)=-10

34... REGRID   POTEN  RATIO=2  SMOOTH=1
35... SYMB     CARR=0
36... SOLVE

37... REGRID   POTEN  RATIO=2  SMOOTH=1
38... SYMB     CARR=0
39... SOLVE

40... PLOT.2D  GRID  SCALE  FILL
  ... +        TITLE="Example 9C - Mesh After Potential Regrids"

41... PLOT.2D  BOUND  SCALE  FILL  TITLE="Example 9C - Potential"
  ... +        L.ELEC=-1
42... CONTOUR  POTEN  COLOR=2
43... LABEL    LABEL="V(Aluminum)=-10v"  X=1.7  Y=-1.2  COLOR=0
44... LABEL    LABEL="V(Poly)=0v"        X=1.9  Y=-0.2  COLOR=0
45... LABEL    LABEL="poly"              X=0.1  Y=-0.2  COLOR=0
46... LABEL    LABEL="poly"              X=4.2  Y=-0.2  COLOR=0
47... LABEL    LABEL="oxide"             X=2.0  Y=-0.7
48... LABEL    LABEL="n-substrate"       X=1.8  Y=0.5

49... PLOT.2D  SCALE  FILL  TITLE="Example 9C - Electric Field"
50... CONTOUR  E.FIELD  MIN=0  FILL  LINE=1
51... PLOT.2D  BOUND  SCALE  L.ELEC=-1  ^CLEAR
52... LABEL    LABEL="V(Aluminum)=-10v"  X=1.7  Y=-1.2  COLOR=0
53... LABEL    LABEL="V(Poly)=0v"        X=1.9  Y=-0.2  COLOR=0
54... LABEL    LABEL="poly"              X=0.1  Y=-0.2  COLOR=0
55... LABEL    LABEL="poly"              X=4.2  Y=-0.2  COLOR=0
```

Figure 10-27 Simulation input file *mdex9c*

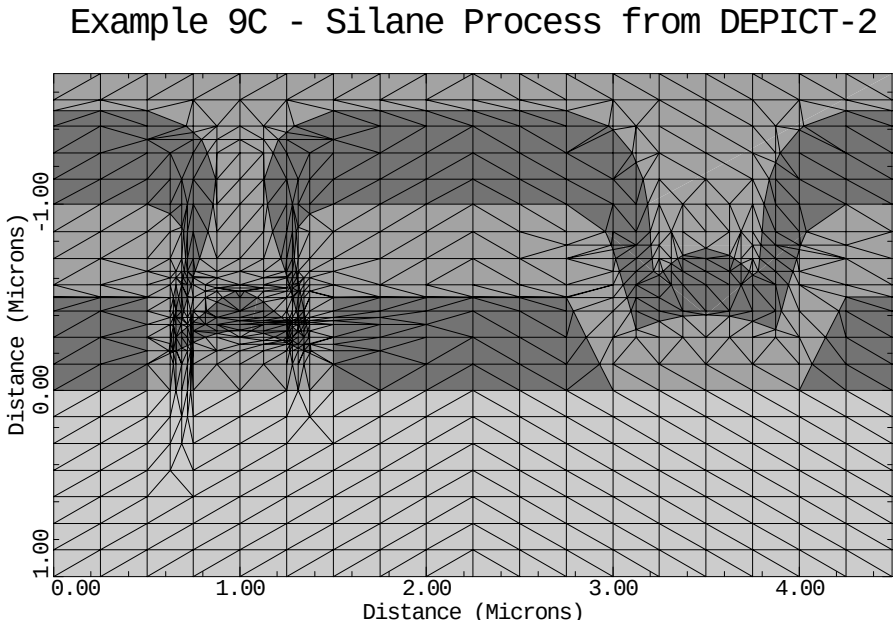


Figure 10-28 Mesh from Taurus-Lithography-2 from **PLOT.2D** at line 29 in file *mdex9c,* Figure 10-27

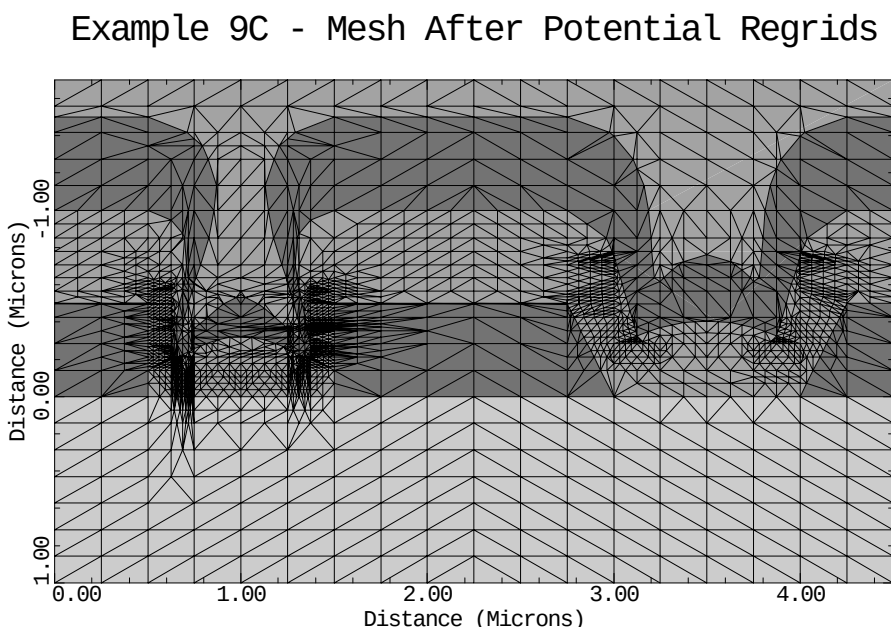


Figure 10-29 Mesh after potential regrids from **PLOT.2D** at line 40 in file *mdex9c*, Figure 10-27

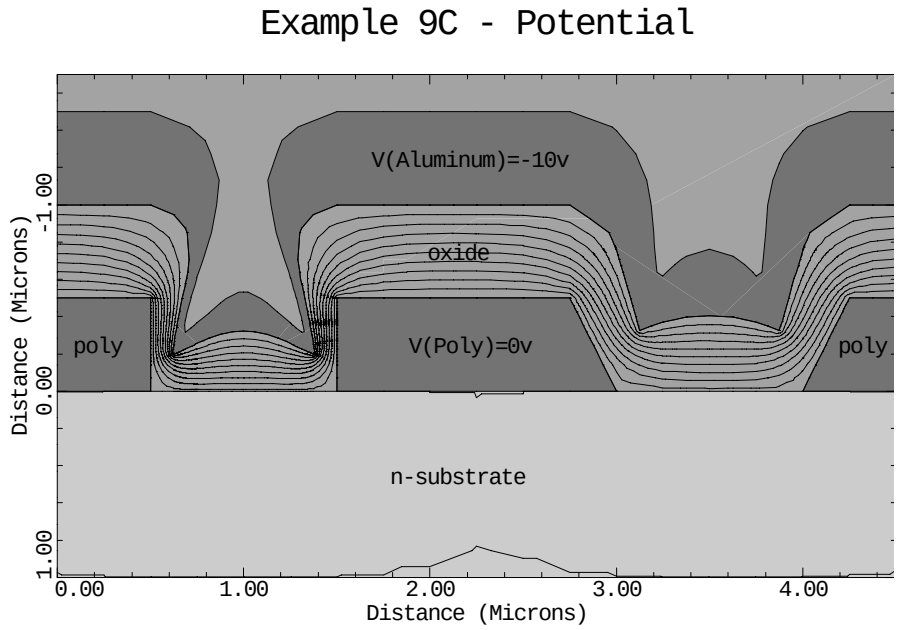


Figure 10-30 Potential contour from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 41 through 48 in file *mdex9c*, Figure 10-27

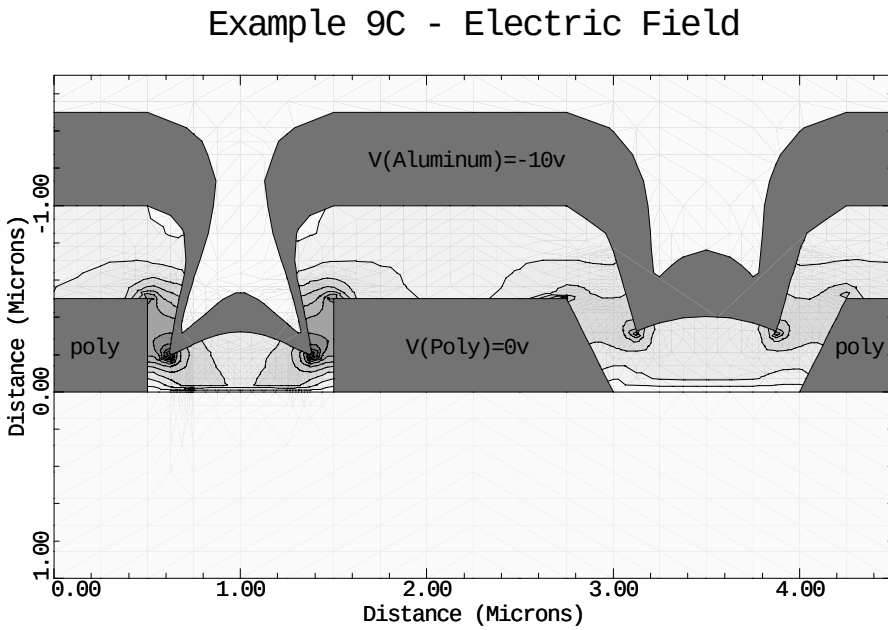


Figure 10-31 Electric field distribution from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 49 through 55 in file *mdex9c*, Figure 10-27

Interface to Aurora

The interface between Medici and Aurora is illustrated by first converting the Medici I-V log file created using the input file *mdex9a* into a data file that is directly readable by Aurora. The Aurora program is then used to perform a simple parameter extraction of a few of the MOS/SPICE model parameters using the data file as input.

Conversion of Medici Log File to Aurora Format

In the example input file *mdex9a*, a Medici I-V log file, *MDEX9AI*, was created that contains the results of a gate characteristic simulation for an n-channel MOSFET. Although it is possible for Aurora to read this file, it would require having a detailed knowledge of the contents of *MDEX9AI*.

Using the **LOG** statement, however, it is possible to convert the Medici I-V log file into a data file that is directly readable by Aurora. The input file *mdex9d*, shown in Figure 10-32, is used for this purpose.

Note:

Various parameters that must be used to specify conversion processes are shown below.

Input and Output Files

The **AURORA** parameter on the **LOG** statement is used to specify that the Medici I-V log file specified with the **IN.FILE** parameter is to be converted into Aurora format and stored in the file specified with the **OUT.FILE** parameter.

Electrode

The parameters **GATE**, **DRAIN** and **SUBST** identify the gate, drain, and substrate electrode names, respectively. The specification of only the **GATE**, **DRAIN** and **SUBST** parameters also indicates that only the I-V information relating to the gate, drain and substrate should be written to the output file. For the simple extraction that is to be performed, this is all that is necessary.

Device Parameters

The **WIDTH** and **LENGTH** parameters identify the channel width and length, respectively, for the device. The values specified here are in units of microns. The resulting Aurora data file stores these parameters in units of meters. The value specified for **WIDTH** multiplies all current values before the date is written to the output file.

```
1... COMMENT   Example 9D - MEDICI/AURORA Interface
2... COMMENT   MEDICI Input File

3... COMMENT   Convert gate characteristic file obtained from Example 9A
 ... +         to a form which can be directly read by AURORA

4... LOG       IN.FILE=MDEX9AI  OUT.FILE=AUEX9DI  AURORA
 ... +         GATE=Gate  DRAIN=Drain  SUBST=Substrate
 ... +          LENGTH=1.4  WIDTH=40
```

Figure 10-32 Output of the simulation input file *mdex9d*

Aurora Extraction

In the previous section, Medici was used to convert an I-V log file containing the gate characteristics for V_{bs}=0V to a data file that can be read directly by Aurora.

Parameter Extraction

In this example, the Aurora input file *auex9d* is used to read the data file, *AUEX9DI*, and a simple extraction of the MOS/SPICE model parameter **LD** is performed.

Figures and Plot

Figure 10-33 shows the parameter initialization file used for the extraction. Figure 10-34 shows the Aurora output listing which contains the input statements specified in *auex9d*, as well as the results of the extraction. The Medici simulated data and the resulting fit to this data are shown in Figure 10-35.

```
$ AURORA Parameter initialization for MOS/SPICE model

  TYPE      1.0
  LEVEL     3
  VTO       0.90      -2.0       2.0
  KP
  GAMMA
  PHI
  LAMBDA
  TOX       2.5E-8
  NSUB      1.0E16     5.0E14    1.0E17
  NSS
  NFS
  TPG
  XJ        0.0E-6     0.0E-6    5.0E-6
  LD        0.0E-6    -1.0E-6    1.0E-6
  UO      400.0      300.0     900.0
  UCRIT
  UEXP
  UTRA
  VMAX      1.0E5      1.0E4     1.0E6
  NEFF
  DELTA     1.0        0.0       5.0
  THETA     0.0        0.0       0.0
  ETA       0.0        0.0       3.0
  KAPPA     0.0        0.0       10.0
  DW        0.0E-6    -1.0E-6    1.0E-6
  R
  USM       0.0
  RSH
  RD        0.0        0.0       1.00E2
  RS        0.0        0.0       1.00E2
  TNOM     27.0
  USUB      0.0         0.0        5.0
```

Figure 10-33 Aurora parameter initialization file *lev3par* for the MOS/SPICE model

```
  1... TITLE     NMOS Parameter Extraction Using AURORA with MEDICI Data
  2... COMMENT   AURORA Input File

  3... COMMENT   Select MOS model and read in data
  4... MODEL     NAME=MOS/SPICE  INIT=lev3par

  5... DATA      FILE=AUEX9DI

  6... COMMENT   Fit short-channel gate characteristics for Vb=0 to
   ... +         get LD
  7... SELECT    W=40E-6  L=1.4e-6
  8... SELECT    Vd=0.1
  9... SELECT    Vb=0.0
 10... INCLUDE   Id  MIN=0.4E-6

   9 target values were included from the data files
There are currently    9 target values included for optimization.

 11... FIX       ALL
 12... EXTRACT   VTO  LD
 13... OPTIMIZE
------------------------------------------------------------------------

  *** Optimization successful:
       Smooth minimum found.
         23 function evaluations in  5 iterations.
       Condition number of solution:   1.49E+02

  parameter   init value  final value   % change    % sens    signif
   vto        9.0000E-01   3.2924E-01    -63.42      0.14      1.14
   ld         0.0000E+00   1.7120E-07  > 999.00      1.42      1.14

  RMS error =  4.42 %

------------------------------------------------------------------------

 14... COMMENT   Plot gate characteristic
 15... PLOT      Id  VARIABLE=Vg  TITLE="GATE:  Fit for LD"

  11 target values were included from the data files
There are currently   11 target values selected for processing.

 16... LABEL     LABEL="W/L=40/1.4"
 17... LABEL     LABEL="Vbs= 0"
```

Figure 10-34 Aurora output of input file *auex9d* and the extraction results

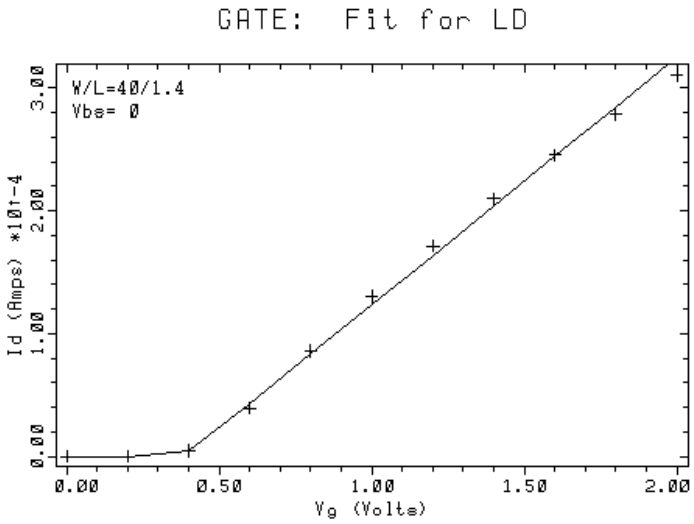


Figure 10-35 Aurora plot of the model fit to the simulated data

Interface to IC-CAP

The interface between Medici and Hewlett-Packard's IC-CAP program is illustrated by converting the Medici I-V log file created using the input file *mdex9a* into a form that is directly readable by IC-CAP.

Conversion of Medici Log File into IC-CAP Format

In the example input file *mdex9a*, a Medici I-V log file, *MDEX9AI*, was created that contains the results of a gate-characteristic simulation for an n-channel MOSFET.

In this example, the **LOG** statement is used to convert the Medici I-V log file into a form that is directly readable by Hewlett-Packard's IC-CAP program. The input file *mdex9e*, shown in Figure 10-36, is used for this purpose.

Note:

Various parameters that must be used to specify conversion processes are shown below.

Input and Output Files

The **ICCAP** parameter on the **LOG** statement is used to specify that the Medici I-V log file specified with the **IN.FILE** parameter is to be converted into IC-CAP format and stored in the file specified with the **OUT.FILE** parameter.

Electrodes

The parameters **GATE**, **SUBSTRAT**, **SOURCE**, and **DRAIN** must all be specified to identify the gate, substrate, source, and drain electrode names, respectively.

Input Variables

Additionally, all the input variables must be identified with the parameters **INP1**, **INP2**, **INP3**, and **INP4**. The most rapidly varying input variable, *V(Gate)*, which represents the gate voltage, should be specified first. In this example, all other

input variables are constant, and therefore, their order is unimportant. Finally, the output variable, *I(Drain)*, which represents the drain current, is identified with the parameter **OUT1**.

Output

Figure 10-36 shows the output listing which contains the input statements specified in *mdex9e*, as well as the results of the conversion.

```
  1... COMMENT   Example 9E - MEDICI/IC-CAP Interface
  2... COMMENT   MEDICI Input File

  3... COMMENT   Convert gate characteristic file obtained from Example 9A
   ... +         to a form which can be directly read by HP's IC-CAP

  4... LOG       IN.FILE=MDEX9AI  OUT.FILE=gat1.set  ICCAP
   ... +         GATE=Gate  SUBSTRAT=Substrate  SOURCE=Source  DRAIN=Drain
   ... +         OUT1=I(Drain)  INP1=V(Gate)  INP2=V(Substrate)
   ... +         INP3=V(Drain)  INP4=V(Source)

Creating IC-CAP data file from MEDICI log file:
Input line #       4

  Number of points in MEDICI log file:             11
  Number of points written to IC-CAP data file:    11

  Input      Start         Stop          Step
  -----  ------------  ------------  ------------
   vg     0.00000E+00   2.00000E+00   2.00000E-01
   vb     0.00000E+00   0.00000E+00   0.00000E+00
   vd     1.00000E-01   1.00000E-01   0.00000E+00
   vs     0.00000E+00   0.00000E+00   0.00000E+00

  Output:  id
```

Figure 10-36 Output from file *mdex9e* and the results of the conversion to an IC-CAP data file

Programmable Device Examples

Example Specifications

The Medici Programmable Device Advanced Application Module (PD-AAM) is capable of simulating the programming characteristics of nonvolatile memory devices such as EPROMs, EEPROMs, and flash EEPROMs. This chapter presents examples of the simulation of flash EEPROM programming. Three models are most important for these applications:

- Charge boundary conditions to describe floating electrodes
- Hot-carrier injection model to describe charge injection into floating electrodes
- Fowler-Nordheim tunneling model to describe charge removal from electrodes (erasing) in EEPROMs

The set of input files discussed in "Writing and Erasing of a Flash EEPROM Cell" on page 11-1 demonstrates the use of these models.

Writing and Erasing of a Flash EEPROM Cell

Writing and erasing are the two basic steps in the operation of a flash EEPROM cell.

Writing

Writing is accomplished by hot-carrier injection from the channel onto the floating gate. The device is biased to create a large number of high-energy (hot) electrons in the channel and a gate-oxide field attracting the carriers from the channel towards the floating gate. This requires a high drain bias and a high positive bias at the control gate.

Erasing

Erasing of the cell is accomplished through the thin tunneling oxide by creating a high enough field pushing the electrons from the floating gate into the semiconductor.

To erase the cell, a positive bias is applied to the erasing electrode (source in the example) and a high negative bias is applied to the control gate. Both writing and erasing are transient processes.

Procedures

The Programmable Device Advanced Application Module (PD-AAM) examples have the following procedures.

Device Structure

The first input file in this set is *mdex10a* shown in Figure 11-1. The example flash EEPROM structure and grid (Figure 11-2) are created from this file.

Gate Characteristics

After the creation of the structure and grid for the example device, its gate characteristics are calculated (shown in Figure 11-3). This is compared to the gate characteristics after writing to demonstrate the shift of the threshold voltage due to charge stored on the floating gate.

Grid

The grid (Figure 11-2) consists of 792 nodes and 1480 triangles. The thickness of the oxide between the channel and floating gate is 10 nm. The gate overlap is larger on the side of the erasing electrode (source) to improve the erasing speed and reduce the sensitivity of the device to processing conditions.

Writing of the EEPROM

The writing is performed by the simulation input file *mdex10b* shown in Figure 11-4. The programming biases chosen for the example are 12V at the control gate and 5.5V at the drain.

Initial Solution

An additional initial condition for the transient writing process is a zero net charge on the floating gate. To obtain this initial solution, a zero-carrier Gummel solution is performed first with voltage boundary conditions at the floating gate. The floating gate potential is set to 1/3 times the control gate potential (line 10 of the listing in Figure 11-4).

Charge Boundary and Solution

After the initial solution is obtained, charge boundary conditions are specified for the floating electrode (line 12) and a one-carrier Newton solution is performed with zero charge specified on the floating electrode (line 16).

Simulation

Line 19 performs the actual writing simulation. Fowler-Nordheim **FN.CUR** and hot carrier Gate Current **GATE.CUR** models as well as **TSTEP** and **TSTOP** are specified to initiate the transient writing of the EEPROM.

Plots

Results are plotted in Figures 11-5 through 11-8. Figure 11-5 shows the hot electron current versus time (note the logarithmic time scale), Figures 11-6 and 11-7 show the floating gate charge and its potential versus time. The hot electron current is highest in the beginning of the simulation but begins to drop after about 1 microsecond as the negative charge injected onto the gate becomes significant and lowers the floating gate potential. This in turn leads to an increased threshold voltage (as shown in Figure 11-8) and decreasing hot carrier injection.

Erasing of the EEPROM

The erase process is performed by the simulation input file *mdex10c* shown in Figure 11-9. The erase biasing is -12V at the control gate and 8V at the source (the drain is held at 0V). Under these bias conditions, there is sufficiently strong electric field in the oxide between the floating gate and the source for the negative charge to tunnel through the oxide.

Solution and Erase

The simulation starts with the steady-state solution with erasing biases applied as specified above. This solution is used as an initial guess for the simulation using charge boundary conditions. The result of the writing simulation is used as charge for the floating gate (line 15).

The actual erase is a transient process, initiated by the **SOLVE** statement on line 18. As in the writing simulation Fowler-Nordheim (**FN.CUR**) and hot carrier gate current (**GATE.CUR**) models as well as **TSTEP** and **TSTOP** are specified.

Plots

Results are displayed in Figures 11-10 through 11-12. Figure 11-10 shows the Fowler-Nordheim tunneling current versus time, Figure 11-11 shows the floating gate charge versus time and Figure 11-12 presents the gate characteristics of the flash EEPROM after erase.

The tunneling current is highest in the beginning of the erase but decreases as the floating gate becomes less negative, thus reducing the field in the tunneling oxide. As a result, the threshold voltage decreases.

```
 1... TITLE     Avant! MEDICI Example 10A
 2... COMMENT   EEPROM Simulation Structure

 3... MESH      OUT.FILE=MDE10MS
 4... X.MESH    WIDTH=.4   H1=.1    H2=.1
 5... X.MESH    WIDTH=.4   H1=.1    H2=.025
 6... X.MESH    WIDTH=.25  H1=.025  H2=.05
 7... X.MESH    WIDTH=.25  H1=.05   H2=.025
 8... X.MESH    WIDTH=.5   H1=.025  H2=.1

 9... Y.MESH    DEPTH=0.050  N.SPACE=1  Y.MIN=-0.125
10... Y.MESH    DEPTH=0.040  N.SPACE=2
11... Y.MESH    DEPTH=0.025  N.SPACE=2
12... Y.MESH    DEPTH=0.010  N.SPACE=1
13... Y.MESH    DEPTH=2.5    H1=0.01  RATIO=1.30

14... COMMENT   Eliminate unnecessary nodes
15... ELIMINAT  COLUMNS  X.MIN=.6  X.MAX=1.5  Y.MIN=1.0

16... COMMENT   Define regions
17... REGION    NUM=1  SILICON
18... REGION    NUM=2  OXIDE  Y.MAX=0.0

19... COMMENT   Electrodes:  Drain=1, Gate=2, Source=3, Subst=4,
  ... +                      Floating Gate=5
20... ELECTROD  NUM=1  X.MIN=1.5  Y.MIN=-0.125  Y.MAX=0
21... ELECTROD  NUM=2  X.MIN=0.5  X.MAX=1.4  IY.MAX=2
22... ELECTROD  NUM=3  X.MAX=0.3  Y.MIN=-0.125  Y.MAX=0
23... ELECTROD  NUM=4  BOTTOM
24... ELECTROD  NUM=5  X.MIN=0.5  X.MAX=1.4  IY.MIN=4  IY.MAX=6

25... COMMENT   Define profiles
26... PROFILE   P-TYPE  N.PEAK=6E16  UNIFORM  OUT.FILE=MDE10DS
27... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=0.632  X.MIN=0.0
  ... +         X.MAX=0.74  X.CHAR=3.2951E-2
28... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=0.332  X.MIN=1.4
  ... +         X.MAX=1.8  X.CHAR=3.2951E-2

29... COMMENT   Create the floating gate
30... CONTACT   NUM=5  CHARGE  N.POLY

31... COMMENT   Specify models
32... MODELS    CONMOB  PRPMOB  FLDMOB  CONSRH  AUGER  BGN

33... COMMENT   Plot the structure
34... PLOT.2D   GRID  SCALE  FILL  X.OFF=5
  ... +         TITLE="Example 10A - EEPROM Simulation Structure"

35... COMMENT   Find the initial threshold voltage
36... SYMBOL    NEWTON  CARRIERS=1  ELECTRONS
37... LOG       OUT.FILE=MD10AIV
38... SOLVE     V1=0.1  V2=0  V3=0  V4=0  Q5=0  ELEC=2
  ... +         VSTEP=.25  NSTEP=10
39... EXTRACT   MOS.PARA  DRAIN=1  GATE=2  IN.FILE=MD10AIV

40... COMMENT   Plot the gate curve.
41... PLOT.1D   X.AXIS=V2  Y.AXIS=I1  POINTS  IN.FILE=MD10AIV
  ... +         TITLE="Example 10A - Initial Threshold Voltage"
```

Figure 11-1 Input file *mdex10a* creating the EEPROM device structure and an initial solution

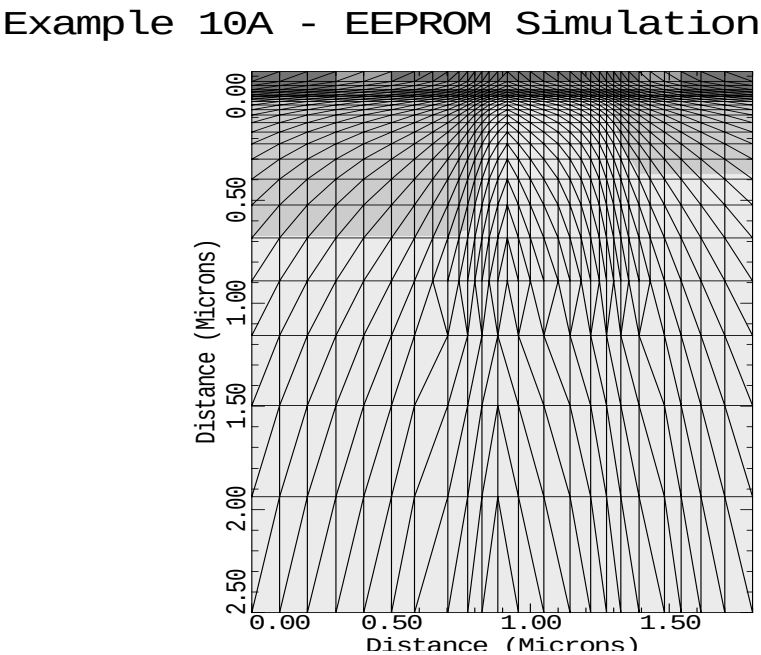


Figure 11-2 Device structure and grid for the flash EEPROM from **PLOT.2D** at line 34 in input file *mdex10a,* Figure 11-1

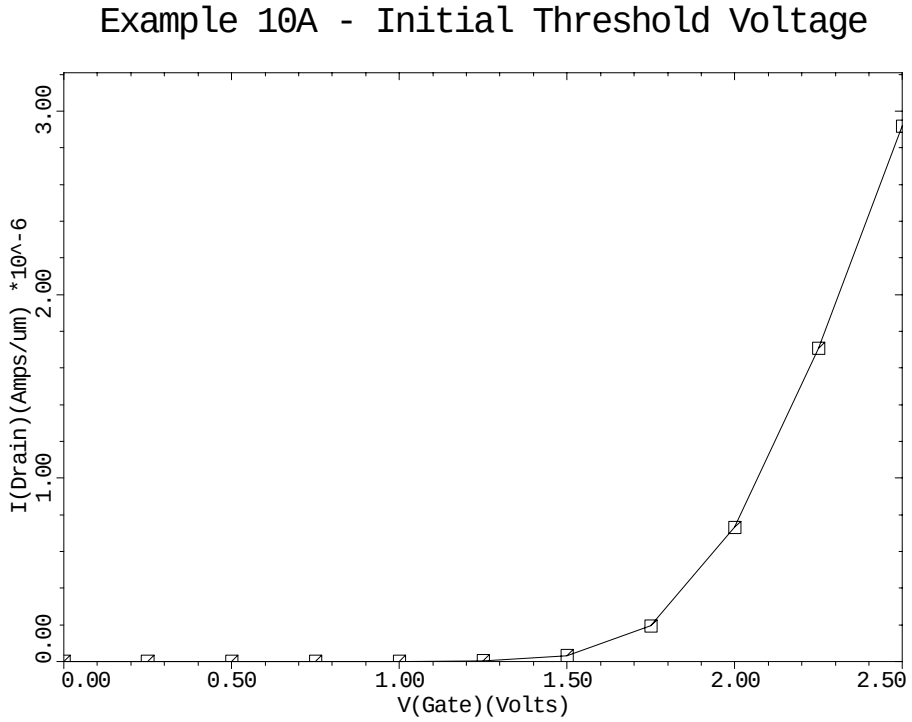


Figure 11-3 Gate characteristics before writing from **PLOT.1D** at line 41 of the input file *mdex10a*, Figure 11-1

```
 1... TITLE     Avant! MEDICI Example 10B - EEPROM Simulation

 2... COMMENT   Load the saved structure and solution, apply bias for
  ... +         a write operation, and then perform the transient.
 3... MESH      IN.FILE=MDE10MS

 4... COMMENT   Programming Bias:  Control Gate = 12 V, Drain = 5.5 V

 5... COMMENT   Specify models
 6... MODELS    CONMOB  CONSRH  AUGER  BGN  FLDMOB  PRPMOB

 7... COMMENT   Use a zero-carrier gummel for the intial guess
  ... +         Estimate the floating gate will bias up at Vg/3
 8... SYMB      CARRIERS=0
 9... METHOD    DAMPED
10... SOLVE     V(Drain)=5.5  V(Gate)=12.0  V(Flt_Gate)=4.0

11... COMMENT   Specify the floating gate as a charge boundary condition
12... CONTACT   NAME=Flt_Gate  CHARGE  N.POLY

13... COMMENT   Initial solution with zero charge on the floating gate
14... SYMB      CARRIERS=1  NEWTON  ELECTRONS
15... METHOD    N.DVLIM=.26
16... SOLVE     Q(Flt_Gate)=0.0

17... COMMENT   Charge the floating region via transient analysis
18... LOG       OUT.FILE=MD10BIT
19... SOLVE     TSTEP=1E-9  TSTOP=1E-2  FN.CUR  GATE.CUR

20... COMMENT   Programing currents, charge, and floating gate voltage
21... PLOT.1D   X.AXIS=TIME  Y.AXIS=HE(Flt_Gate)  IN.FILE=MD10BIT  X.LOG
  ... +         POINTS  TITLE="Example 10B - Hot Electron Current"
22... PLOT.1D   X.AXIS=TIME  Y.AXIS=Q(Flt_Gate)  IN.FILE=MD10BIT  X.LOG
  ... +         POINTS  TITLE="Example 10B - Floating Gate Charge"
23... PLOT.1D   X.AXIS=TIME  Y.AXIS=V(Flt_Gate)  IN.FILE=MD10BIT  X.LOG
  ... +         POINTS  TITLE="Example 10B - Floating Gate Voltage"

24... COMMENT   Find the new threshold voltage
25... LOG       LOG.FILE=MD10BIV
26... SOLVE     INITIAL  V(Drain)=0.1  V(Gate)=4.0
27... SOLVE     ELEC=Gate  VSTEP=0.5  NSTEP=8
28... EXTRACT   MOS.PARA  DRAIN=Drain  GATE=Gate  IN.FILE=MD10BIV

29... COMMENT   Plot the gate curve
30... PLOT.1D   X.AXIS=V(Gate)  Y.AXIS=I(Drain)  POINTS  IN.FILE=MD10BIV
  ... +         TITLE="Example 10B - Threshold After Programming"
```

Figure 11-4 Input file *mdex10b* for the writing characteristics of the flash EEPROM

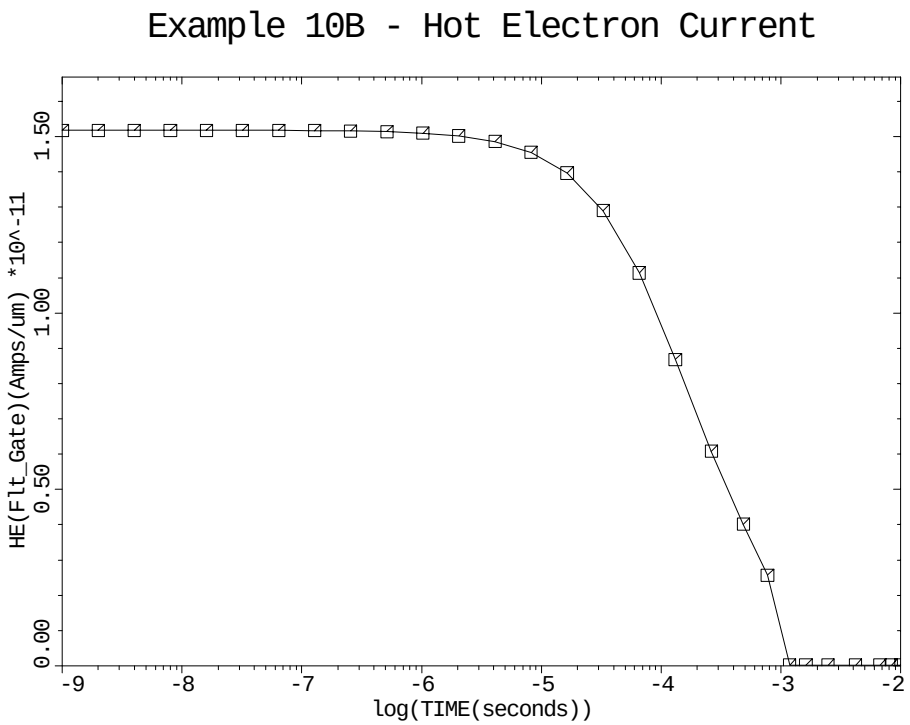


Figure 11-5 Hot electron current during writing from **PLOT.1D** at line 21 in input file *mdex10b*, Figure 11-4

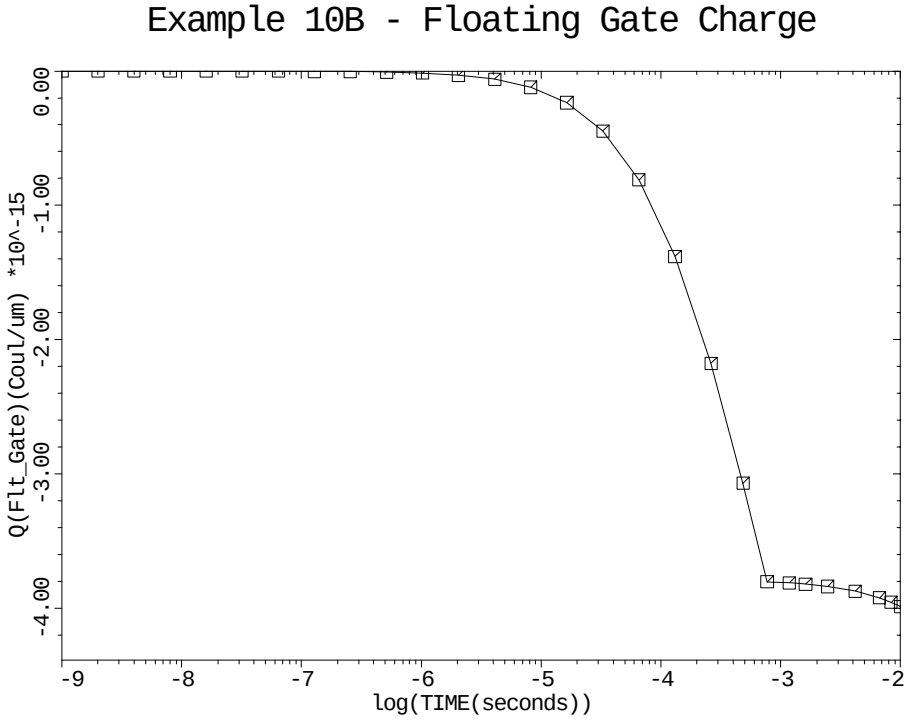


Figure 11-6 Floating gate charge during writing from **PLOT.1D** at line 22 in input file *mdex10b*, Figure 11-4

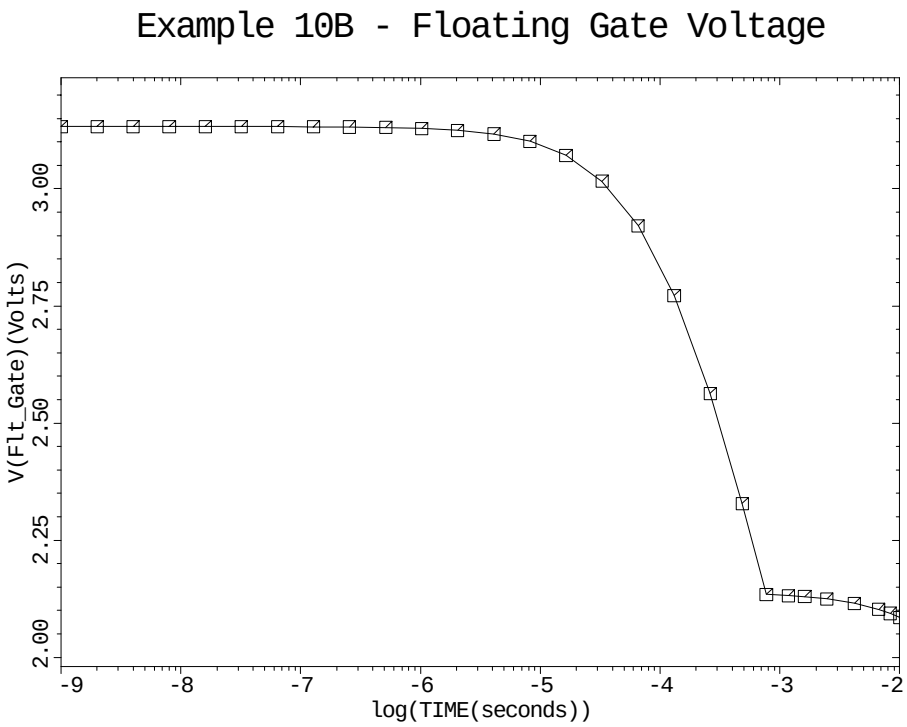


Figure 11-7 Floating gate potential during writing from **PLOT.1D** at line 23 in input file *mdex10b*, Figure 11-4

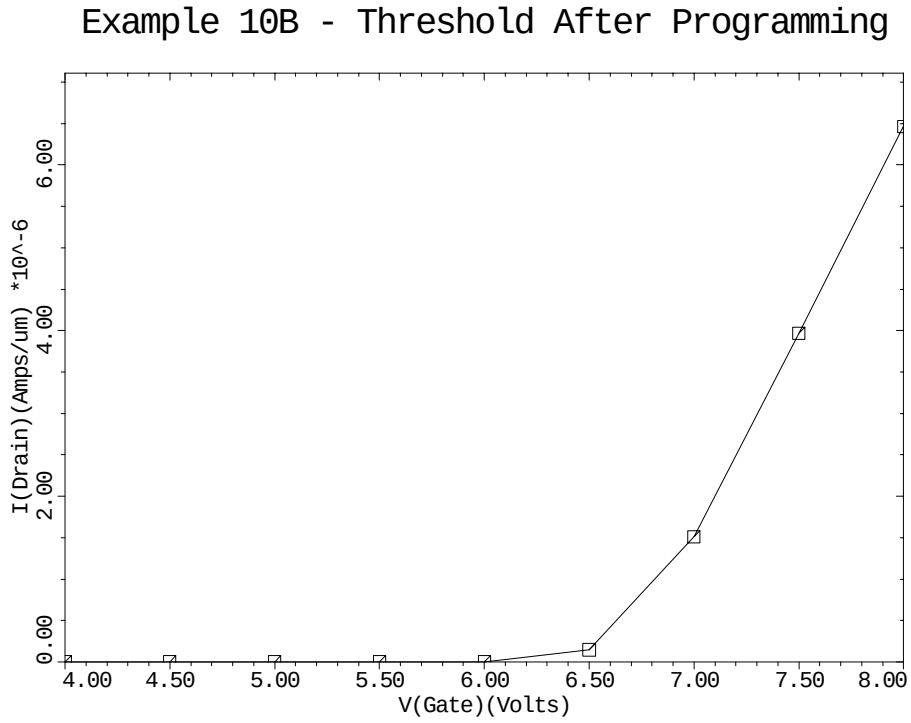


Figure 11-8 Gate characteristics of the flash EEPROM after writing from **PLOT.1D** at line 30 in output file *mdex10b* shown in Figure 11-4

```
  1... TITLE     Avant! MEDICI Example 10C - EEPROM Simulation

  2... COMMENT   Load the saved structure and solution, apply bias for
   ... +         an erase operation, and then perform the transient
  3... MESH      IN.FILE=MDE10MS

  4... COMMENT   Programming Bias:  Control Gate = -12 V, Source = 8.0 V

  5... COMMENT   Specify models
  6... MODELS    CONMOB  FLDMOB  CONSRH  AUGER  BGN  PRPMOB

  7... COMMENT   Use a zero-carrier gummel for the intial guess
   ... +         Estimate the floating gate will bias up at Vg/3
  8... SYMB      CARRIERS=0
  9... METHOD    DAMPED
 10... SOLVE     V(Source)=8.0  V(Gate)=-12.0  V(Flt_Gate)=-4.0

 11... COMMENT   Use a charge boundary condition on the floating gate
 12... CONTACT   NUM=Flt_Gate  CHARGE  N.POLY

 13... COMMENT   Obtain the initial solution with charge on the
   ... +         floating gate. The value for Q(Flt_Gate) is obtained
   ... +         from Example 10B at t=1e-2
 14... SYMB      CARRIERS=1  NEWTON  ELECTRONS
 15... SOLVE     Q(Flt_Gate)=-3.988E-15  OUT.FILE=MDE10S1

 16... COMMENT   Erase the floating electrode using transient analysis
 17... LOG       OUT.FILE=MD10CIT
 18... SOLVE     TSTEP=1E-6  TSTOP=1E-3  FN.CUR  GATE.CUR

 19... COMMENT   Programing currents and floating gate charge
 20... PLOT.1D   X.AXIS=TIME  Y.AXIS=FE(Flt_Gate)  IN.FILE=MD10CIT  X.LOG
   ... +         POINTS  TITLE="Example 10C - Fowler-Nordheim Current"
 21... PLOT.1D   X.AXIS=TIME  Y.AXIS=Q(Flt_Gate)  IN.FILE=MD10CIT  X.LOG
   ... +         POINTS  TITLE="Example 10C - Floating Gate Charge"

 22... COMMENT   Find the new threshold voltage
 23... LOG       OUT.FILE=MD10CIV
 24... SOLVE     INIT  V(Drain)=0.1  V(Gate)=0.0  V(Source)=0.0
 25... SOLVE     ELEC=Gate  VSTEP=0.5  NSTEP=10
 26... EXTRACT   MOS.PARA  DRAIN=Drain  GATE=Gate  IN.FILE=MD10CIV

 27... COMMENT   Plot the gate curve
 28... PLOT.1D   X.AXIS=V(Gate)  Y.AXIS=I(Drain)  IN.FILE=MD10CIV
   ... +         POINTS  TITLE="Example 10C - Threshold After Erase"
```

Figure 11-9 Output of file *mdex10c* that erases the flash EEPROM

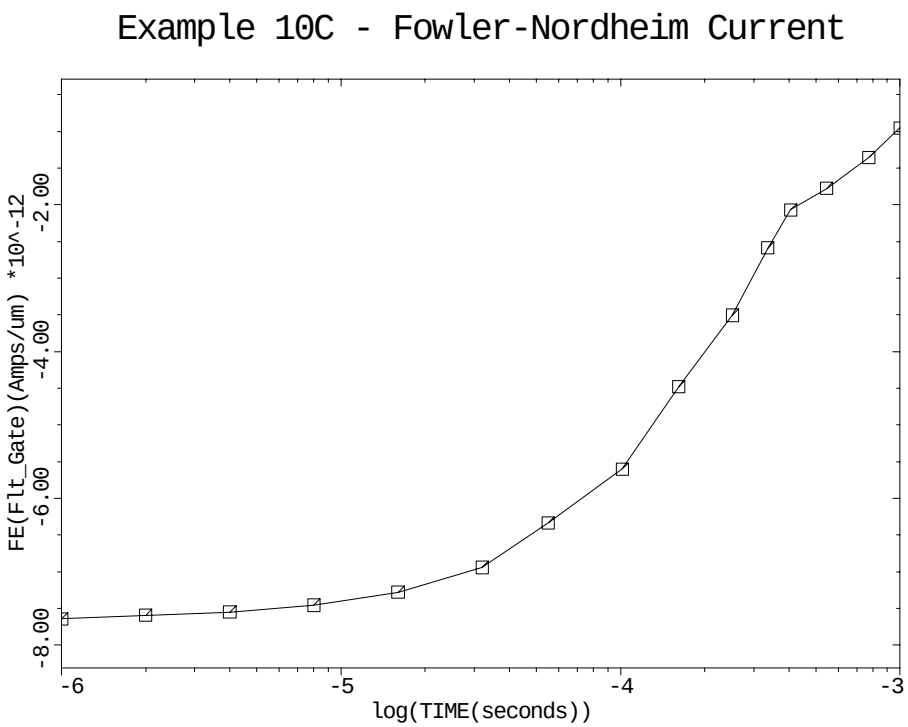


Figure 11-10 Fowler-Nordheim tunneling current during the erase from **PLOT.1D** at line 20 in input file *mdex10c* shown in Figure 11-9

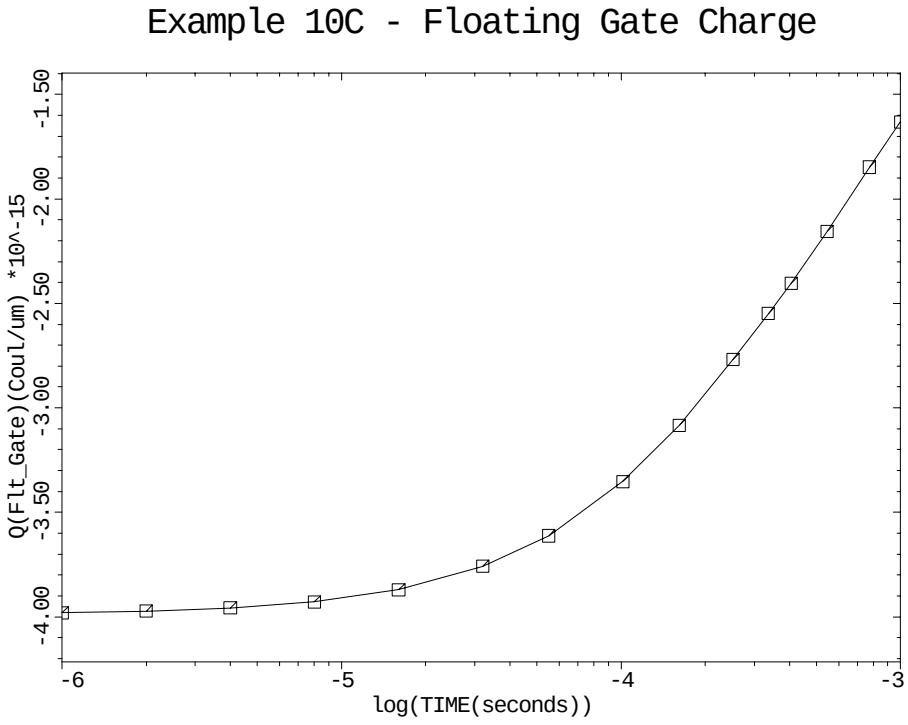


Figure 11-11 Floating gate charge during the erase from **PLOT.1D** at line 21 in input file *mdex10c*, Figure 11-9

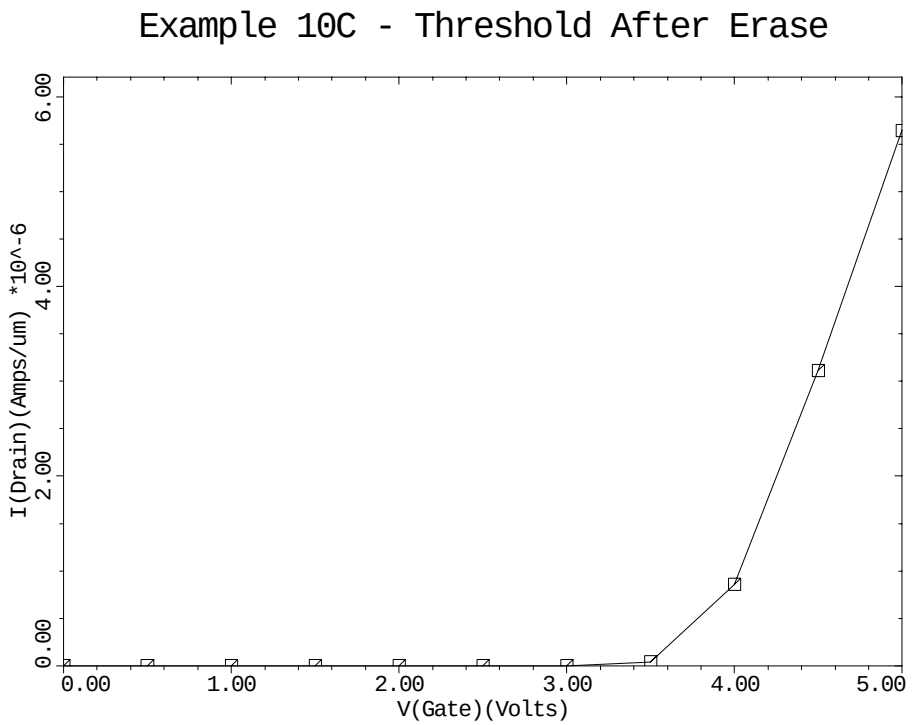


Figure 11-12 Gate characteristics after the erase from **PLOT.1D** at line 28 in input file *mdex10c*, Figure 11-9

Circuit Analysis Examples

Example Specifications

This chapter details the Medici Circuit Analysis Advanced Application Module (CA-AAM). The CA-AAM is used to model a wide variety of circuits. The following analyses are presented:

- A single bipolar transistor connected to a resistive load.

  This example uses a template to create the bipolar junction transistor (BJT) and illustrates the basic use of the circuit analysis module.
- A transient single event upset simulation of a diode in an SRAM circuit.
- Calculate the characteristic for a CMOS inverter constructed from Medici transistors including:
  - The transient transfer curve for the inverter
  - The frequency response of the circuit
  - The DC transfer curve

Bipolar Transistor With Resistive Load

The use of the Circuit Analysis AAM with Medici is illustrated by calculating a single bias point for a BJT connected to a resistive load. Although this example is simple, and can be performed without the circuit analysis module, (using the **`RESISTAN`** parameter on the **`CONTACT`** statement) it serves as a good introductory problem. A schematic diagram of the circuit to be analyzed is presented in Figure 12-1.

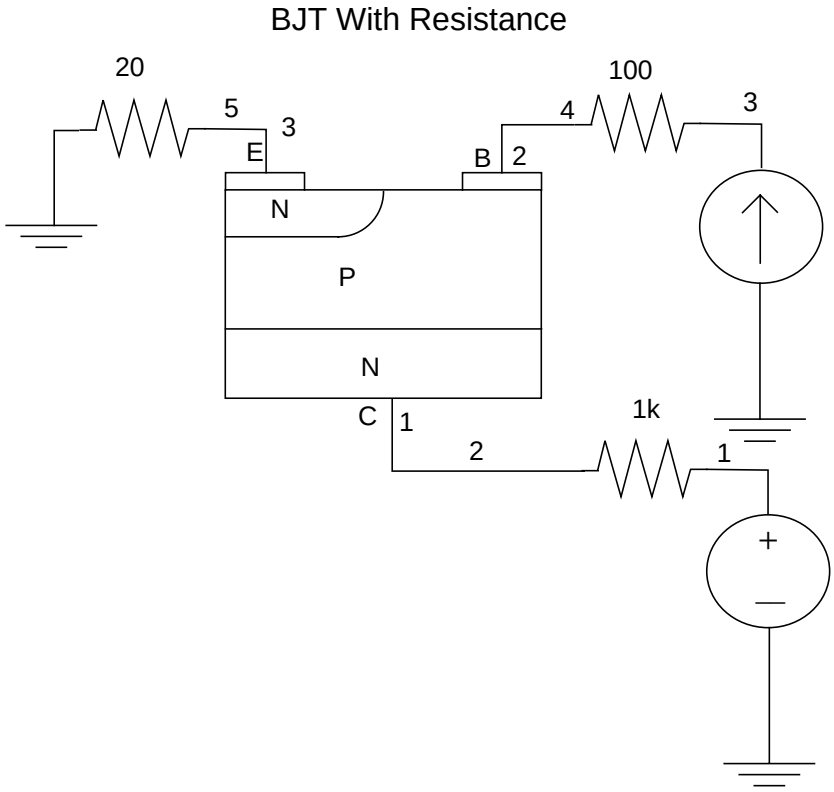


Figure 12-1 Bipolar transistor with resistive load

Generation of the Simulation Structure and Solutions

The input file *mdex11* creates the simulation structure for the BJT and circuit and simulates the steady-state characteristics. The output associated with the execution of Medici for the input file *mdex11* is shown in Figures 12-2 through 12-6.

Mesh File

The first step in creating the structure is the generation of a mesh file for the circuit simulator. The predefined template *bipstr0* is used for this purpose. First the default values for the parameters are read in from the file *bipdef0* at line 3.

Transistor Type

The transistor type is assigned as *NPN* at line 59 and the peak values for the base and emitter doping are reset to 10^{18} and 1.5×10^{20} respectively. The default values for the other parameters such as the emitter width, collector doping etc. are assigned in file *bipdef0*.

Models

At line 341 the models to be used in the simulation are specified. At line 343 the mesh and models are written to the file *MDE11MS*. The **`W.MODELS`** parameter causes all the **`MODELS`**, **`MOBILITY`**, **`MATERIAL`**, and **`INTERFACE`** information

to be included in the file. This file is used by the circuit simulator to create the device structure.

Circuit Mode

Line 345 places the program into circuit mode using the **START** statement. From now until the **FINISH** statement is encountered at line 368, the simulator is in circuit mode and the SPICE syntax is used.

Voltage Source

Line 347 creates a 5 volt voltage source connected between circuit nodes 1 and 0 (ground). Line 362 creates a 1000 ohm resistor connected between circuit nodes 1 and 2 (the collector). Lines 351, 353, and 355 create current source *IB* and resistors at the base and emitter.

Transistor

Line 358 creates the Medici transistor. Note that the letter *P* in the name *PBjt* indicates a Medici device, the *Bjt* is an arbitrary choice (it could have been *P2*, *Pa*, *Pdog* etc.) Each of the following character/number pairs assigns a circuit node to a Medici terminal. Circuit node 2 is connected to the collector, circuit node 4 to the base, etc. **FILE** specifies the mesh file used to create the device. In this case the file that was created at line 343 is used.

Initial Guess

Line 360 specifies the initial guess for the circuit node voltages using the **.NODESET** statement. The initial guess need not be exact but should be within about 30 percent for best results. The SPICE syntax is used here again, i.e., node 1 is set to 5 volts, node 2 to 4 volts, etc.

Note:

These values are also used to obtain an initial guess for the internal circuit variables (potential, electrons, holes) so it is very important to specify the initial guess.

Potentials

Line 363 specifies the limit on the size of the potential update both internal to the device and at the circuit nodes using the **.OPTIONS** statement. This parameter aids in convergence by preventing the potentials from changing too rapidly and becoming unstable (oscillating wildly). Since the initial guess is reasonably sure, **DELVMAX** is reduced from its default value to of 0.5V to 0.2V.

Solution

Line 366 calculates the solution. Note that the **.OP** statement is the same as a **.DC** statement without any parameters. The bias point was found to be:

```
VC(1)=5.0   VC(2)=4.755  VC(3)=.858   VC(4)=.857
VC(5)=5.1e-3
IC(VDD)=-2.45e-4
```

Plots

Line 368 places the program back into Medici mode for plotting the internal characteristics. Lines 370-372 generate plots of the electron distribution and current

flow lines which are shown in Figure 12-6. The output associated with the execution of Medici for the input file *mdex11* is shown in Figures 12-2 through 12-6.

```
  1... TITLE     Example 11 - BJT with Series Resistance

  2... COMMENT   Get default values
  3... CALL      FILE=bipdef0 ^PRINT

 58... COMMENT   Select an NPN device
 59... ASSIGN    NAME=TRANTYPE  C.VALUE=NPN

 60... COMMENT   Assign some new values for the doping profiles
 61... ASSIGN    NAME=EPEAK   N.VALUE=1.5e20
 62... ASSIGN    NAME=XBPEAK  N.VALUE=1e18


 63... COMMENT   Create the meshfile using the template "bipstr0"
 64... CALL      FILE=bipstr0 ^PRINT

340... COMMENT   Specify some physical models
341... MODELS    CONMOB  CONSRH  AUGER  BGN

342... COMMENT   Save the meshfile
343... SAVE      OUT.FILE=MDE11MS  MESH  W.MODELS

344... COMMENT   Enter CIRCUIT mode
345... START     CIRCUIT
346...       $ Power source
347...       VDD 1  0   5
348...       $ Collector resistance
349...       RC  1  2   1k
350...       $ Base bias
351...       IB  0  3   .01m
352...       $ Base resistance
353...       RB  3  4   100
354...       $ Emitter resistance
355...       RE  5  0   20
356...       $
357...       $ MEDICI transistor
358...       PBjt  2=Collector   4=Base    5=Emitter     FILE=MDE11MS
359...       $ Initial guess
360...       .NODESET  V(1)=5   V(2)=4  V(3)=.85  V(4)=.85  V(5)=.02
361...       $
362...       $ Due to high confidence in the initial guess, reduce DELVMAX
363...       .OPTIONS  DELVMAX=0.2
364...       $
365...       $ Find the operating point
366...       .OP
367...       $ Switch back to MEDICI mode for plotting
368... FINISH     CIRCUIT

369... COMMENT   Plot contours electron concentration and current flow lines
370... PLOT.2D   STRUCTUR=PBjt
   ... +         TITLE="Example 11 - Electrons and Current Flow"
371... CONTOUR   ELECTRONS  LOG  FILL
372... CONTOUR   FLOWLINES
```

Figure 12-2 Output of the simulation input file *mdex11*

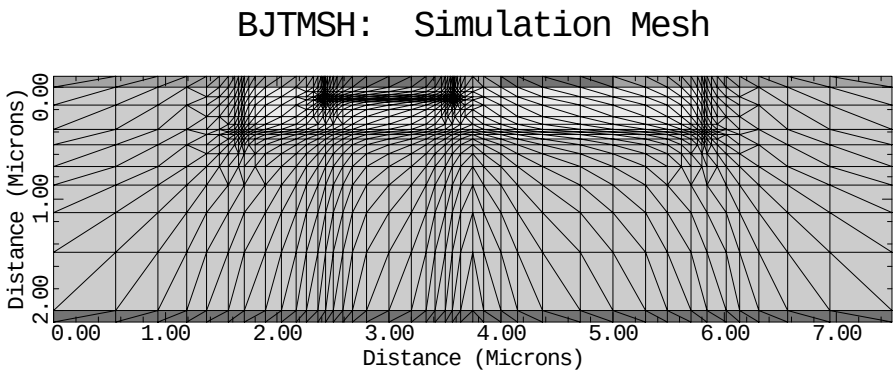


Figure 12-3 Simulation mesh from **CALL** at line 64 in input file *mdex11,* Figure 12-2

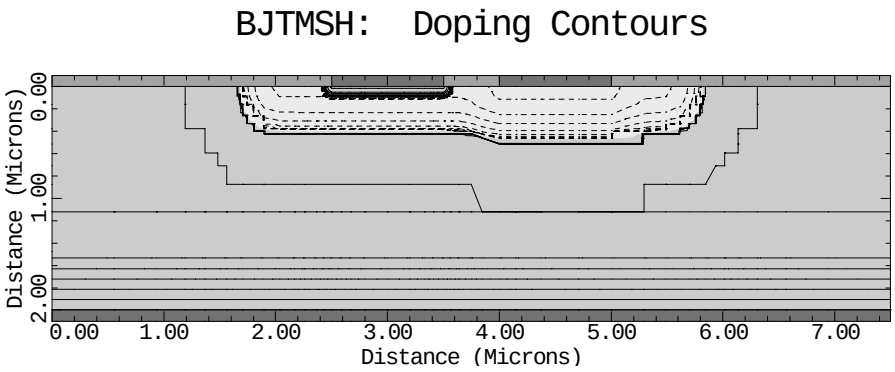


Figure 12-4 Doping contours from **CALL** at line 64 in input file *mdex11*, Figure 12-2

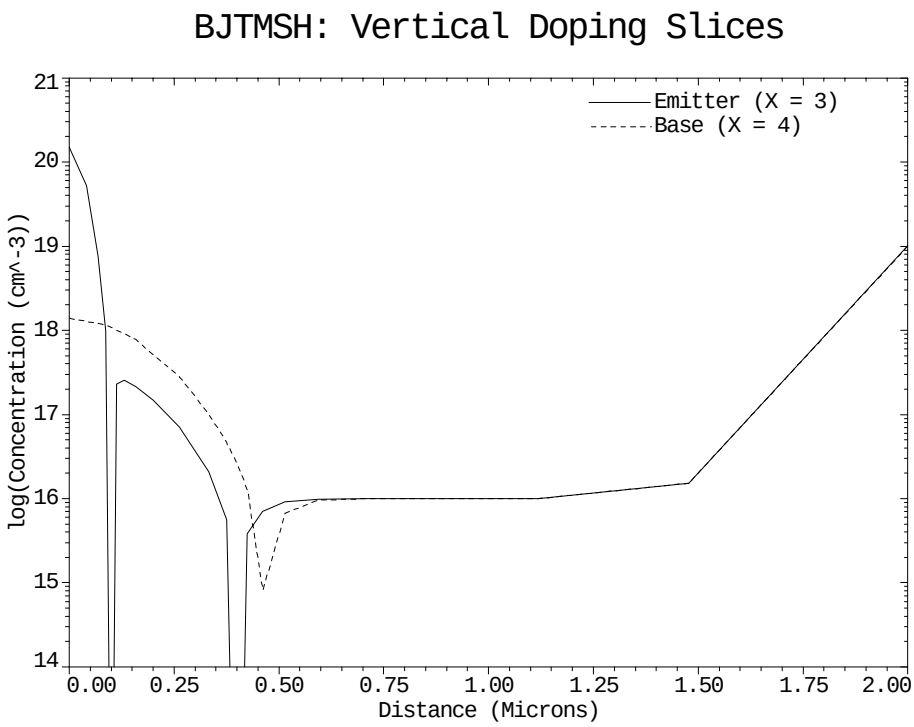


Figure 12-5 Doping slices from **CALL** at line 64 in input file *mdex11,* Figure 12-2

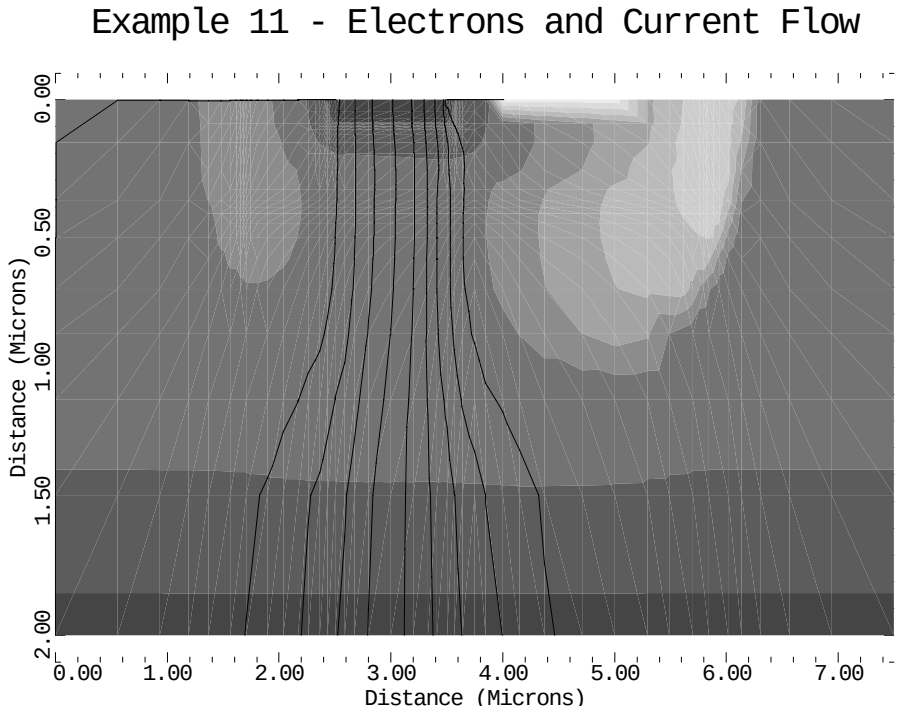


Figure 12-6 Electrons and current flow from **PLOT.2D** and **CONTOUR** at lines 370 through 372 in input file *mdex11*, Figure 12-2

Single Event Upset of SRAM Cell with Circuit

This example is an extension of the single event upset in "Single-Event Upset of a SRAM Cell" on page 7-9. It will be helpful to read this section on Page 7-9 if you have not done so.

This example simulates the same cylindrical diode structure with a small circuit attached to model the transistors of the SRAM cell. In this way the circuit supplies the correct boundary conditions for the diode and the diode supplies the current which upsets the cell.

SEU Schematic

A schematic of the complete circuit with diode is shown in Figure 12-7. Upset of the cell is indicated if the voltages at nodes 2 and 4 (the drains of the MOSFETs) interchange values. The resistors in the circuit (*R5* and *R6*) slow the cells response and reduce the vulnerability to upset by giving the cell more time to recover after the particle hits.

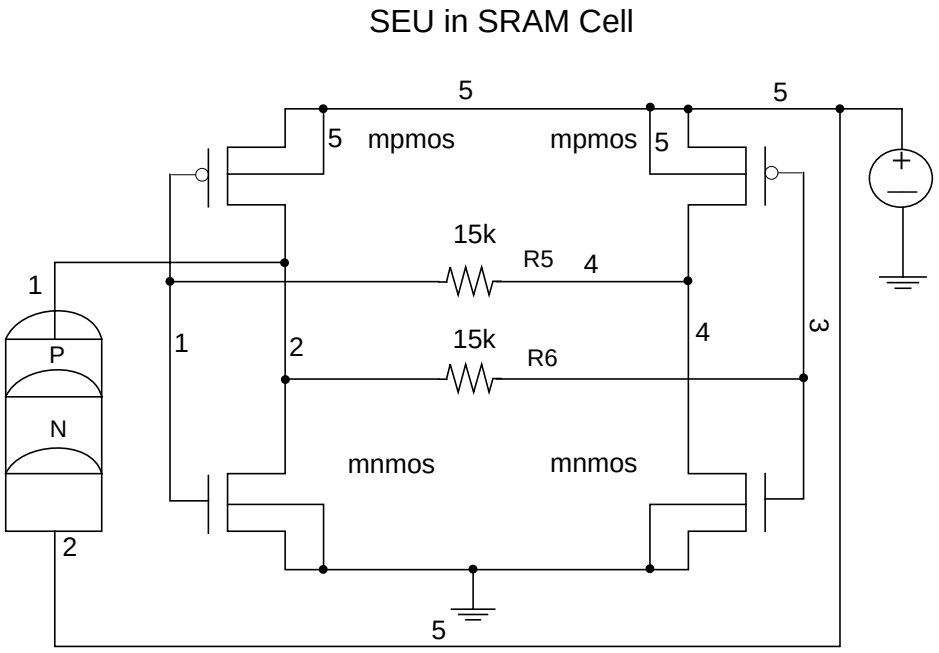


Figure 12-7 SRAM circuit with cylindrical diode

Generation of the Simulation Structure and Solutions

This simulation is divided into two input files, *mdex12a* and *mdex12b.* File *mdex12a* amends *mdex6* to create the cylindrical diode, and then passes the information to the simulation file. The file *mdex12b* performs the actual simulation.

Creating the Cylindrical Diode

This file creates the cylindrical diode and a listing is given in Figure 12-8. The file is similar to the grid generation portion of *mdex6* found in "Single-Event Upset of a SRAM Cell" on page 7-9, with following difference. Instead of performing the solution the mesh is written to the file *MDE12MS* at line 21 for use with simulation file *mdex12b.*

The **W.MODELS** parameter causes the model information to be written to the file as well as the grid. The plot of the device grid generated by line 18 of this file appears in Figure 12-9. The P+ region represents the drain diffusion of a P-channel and the N-type body represents the substrate of the die.

Simulation

The simulation is performed by file *mdex12b*, which is shown in Figure 12-10.

Circuit Mode

Line 3 places the simulator in circuit mode.

Lumped Circuit Elements

Lines 7 through 14 create the lumped circuit elements. The lumped element MOSFETs which form the inverters are at lines 10 through 14.

The ordering for the nodes of a MOSFET is drain, gate, source, and substrate. Note that the NMOS and PMOS transistor model names are *MNMOS* and *MPMOS* respectively. These models refer to the **.MODEL** statements at lines 18 and 19 where the parameters for the transistors are specified. These parameters are identical to those used by SPICE 2G.6 and would be extracted from measured data or Medici simulations using Aurora or some other parameter extractor. It is also possible to model *M7-M10* using Medici transistors for greater accuracy.

Cylindrical Diode

Line 16 creates the Medici cylindrical diode, circuit node 2, which corresponds to the drains in the first inverter attached to the drain terminal, the P+ contact of the diode. Node 5, the power supply, is attached to the substrate terminal, the N+ contact of the diode.

Initial Guess

Line 21 sets the initial guess for the simulation. Chose the state for the circuit so that nodes 2 and 4, are at zero volts and nodes 5,4 and 1 at V_{dd}=3V. This places the P+ contact of the diode at 0 volts, reverse biasing the diode as in "Single-Event Upset of a SRAM Cell" on page 7-9.

Exiting Circuit Mode

Circuit mode is exited at line 23 since this simulation requires a **PHOTOGEN** statement which can only be used in Medici mode.

Solution

Lines 25-29 generate the initial solution using the voltages specified on the **.NODESET** statement as the initial guess. The circuit is included in this calculation. The circuit biases up at:

```
VC(1)=3.0   VC(2)=1.6e-9   VC(3)=1.6e-9  VC(4)=3.0 VC(5)=3.0
IC(VDD)=-9.05e-12 (AMPS)
```

Charge Track

Line 31 specifies the charge track that pierces the junction. The track used is identical to that of Chapter 7, Single-Event Upset of a SRAM Cell on page 7-9. Note that the device *Pcell* is specified with the **STRUCTUR** parameter corresponding to the diode.

Logfile

Line 33 opens up a **LOGFILE** where the terminal currents and circuit voltages are stored.

Time Step and Solve

The **METHOD** statement at line 35 relaxes the time step tolerance to 2 percent. Line 36 performs the simulation. An initial time step of 0.5 picosecond is selected and the simulation runs for 0.5 nanosecond.

Plots

Lines 38-45 generate the waveform plots shown in Figure 12-12. Lines 46-47 generate current and charge plots shown in Figures 12-13 and 12-14.

Observe how the circuit nodes are specified in lines 46 and 47. **EXTRACT** statements also use a similar syntax.

- To plot the voltage at a circuit node, specify **Y.AXIS**=*VC*(<*node*>).
- To plot the current in a voltage source, specify **Y.AXIS**=*IC*(<*source*>) where <*source*> is the name of the voltage source.

- To plot the current or the voltage at a Medici terminal, use **Y.AXIS**=*I*(<*dname*>.<*ter*>) where <*dname*> is the name of the Medici device (*Pcell* in this case) and <*ter*> is the name of the terminal.

```
 1... TITLE     Example 12A - Create Cylindrical P+N Diode
 2... COMMENT   P+/EPI/N+ structure using cylindrical coordinates
  ... +         (based on example MDEX6)

 3... COMMENT   Define a non-uniform mesh using cylindrical coordinates
  ... +         (the x-direction corresponds to the radial direction).
  ... +         Put the finest grid along the r=0 column and at the
  ... +         junction.
 4... MESH      CYLINDRI
 5... X.MESH    WIDTH=3.0  H1=0.02  H2=0.30
 6... Y.MESH    DEPTH=0.3  H1=0.10
 7... Y.MESH    DEPTH=3.7  H1=0.10  H2=0.50

 8... REGION    NAME=Silicon  SILICON

 9... COMMENT   Electrodes:
10... ELECTR    NAME=Drain  TOP  X.MAX=1.0
11... ELECTR    NAME=Substrate  BOTTOM

12... COMMENT   Profiles for the Epi layer, N+ substrate,
  ... +         and P+ diffusion
13... PROFILE   N-TYPE  N.PEAK=1.0E17  UNIF  OUT.FILE=MDE12DS
14... PROFILE   N-TYPE  N.PEAK=5.0E18  Y.MIN=2.0  DEPTH=2.0  Y.CHAR=0.1
15... PROFILE   P-TYPE  N.PEAK=1.0E20  JUNC=0.3
  ... +         WIDTH=1.0  XY.RATIO=0.75

16... COMMENT   Grid refinement based on doping.
17... REGRID    DOPING  LOG  RATIO=1  SMOOTH=1  Y.MAX=1.0
  ... +         IN.FILE=MDE12DS

18... PLOT.2D   GRID  TITLE="Example 12A - Cylindrical Diode"
  ... +         SCALE  FILL

19... COMMENT   Specify physical models to use
20... MODELS    CCSMOB  FLDMOB  CONSRH  AUGER  BGN

21... SAVE      MESH  OUT.FILE=MDE12MS  W.MODELS
```

Figure 12-8 Output of simulation input file *mdex12a*

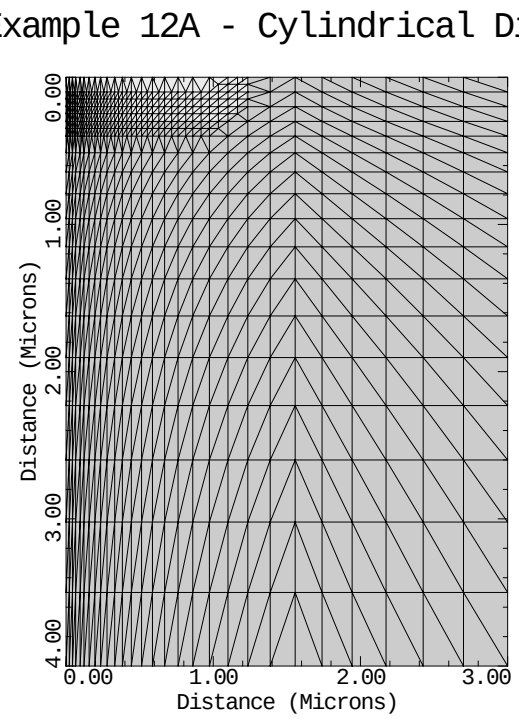


Figure 12-9 Cylindrical diode from **PLOT.2D** at line 18 in file *mdex12a,* Figure 12-8

```
 1... TITLE     Example 12B - Single Event Upset of SRAM Cell

 2... COMMENT   Enter CIRCUIT mode
 3... START     CIRCUIT

 4...       $ Power source
 5...       VDD 5  0  3

 6...       $ Decoupling resistors
 7...       R5 1  4  15K
 8...       R6 2  3  15K

 9...       $ First inverter
10...       M9 2  1  0  0  MNMOS PS=20  PD=20  AS=5  AD=5  W=1.1  L=.7
11...       M7 2  1  5  5  MPMOS PS=40  PD=40  AS=8  AD=8  W=1.5  L=.7

12...       $ Second inverter
13...       M8 4  3  5  5  MPMOS PS=40  PD=40  AS=8  AD=8  W=1.5  L=.7
14...       M10 4  3  0  0 MNMOS PS=20  PD=20  AS=5  AD=5  W=1.1  L=.7

15...       $ MEDICI cylindical diode
16...       Pcell 2=Drain  5=Substrate FILE=MDE12MS

17...       $ MOSFET models
18...       .MODEL MPMOS PMOS VTO=-.6 TOX=150 NSUB=3E16 VMAX=5E6 LEVEL=2
  ...         +  JS=1E-14 UCRIT=1E4 UEXP=.5 UO=300
19...       .MODEL MNMOS NMOS VTO=.7 TOX=150 NSUB=1E16 VMAX=1E7 LEVEL=2
  ...         +  JS=1E-14 UCRIT=1E4 UEXP=.5 UO=600

20...       $ Set up voltages for the initial guess
21...       .NODESET V(5)=3 V(4)=3 V(1)=3

22...       $ Return to MEDICI mode for simulation and plotting
23... FINISH    CIRCUIT

24... COMMENT   Obtain an steady state solution
  ... +         for the initial conditions
25... SYMBOLIC  GUMMEL CARRIERS=0
26... METHOD    DAMPED
27... SOLVE
28... SYMBOLIC  NEWTON CARRIERS=2
29... SOLVE

30... COMMENT   Set up the ion track, refer to MDEX6 for more details
31... PHOTOGEN  X.START=0.0  X.END=0.0  Y.START=0.0  Y.END=4.0
  ... +         R.CHAR=0.2 T0=3.0E-12  TC=1.5E-12  IN.FILE=mde6let
  ... +         PC.UNIT  STRUC="Pcell"  GAUSS
```

Figure 12-10 First part of simulation input file *mdex12b*

```
32... COMMENT   Open a log file for the circuit voltages
  ... +         and terminal currents
33... LOG       OUT.FILE=MDE12BI

34... COMMENT   Simulate the first 500 picoseconds
  ... +         of the transient response.
35... METHOD    TOL.TIME=2e-2
36... SOLVE     TSTEP=0.5E-12  TSTOP=5e-10


37... COMMENT   Plot the circuit voltages
38... PLOT.1D   X.AX=TIME Y.AX=VC(1) TOP=4 BOT=0 SYMB=1
  ... +         TITLE="Example 12B - Waveforms:  SEU with Argon Ion"
39... PLOT.1D   X.AX=TIME  Y.AX=VC(2)  COLOR=2  UNCH  SYMB=2
40... PLOT.1D   X.AX=TIME  Y.AX=VC(3)  COLOR=3  UNCH  SYMB=3
41... PLOT.1D   X.AX=TIME  Y.AX=VC(4)  COLOR=4  UNCH  SYMB=4
42... LABEL     LABEL="VC(1)"  COLOR=1  SYMB=1  X=100  Y=1
43... LABEL     LABEL="VC(2)"  COLOR=2  SYMB=2
44... LABEL     LABEL="VC(3)"  COLOR=3  SYMB=3
45... LABEL     LABEL="VC(4)"  COLOR=4  SYMB=4
46... PLOT.1D   X.AX=TIME  Y.AX=I(Pcell.Substrate)  POINTS  X.MAX=5E-10
  ... +         TITLE="Example 12B - Current:  SEU with Argon Ion"
47... PLOT.1D   X.AX=TIME  Y.AX=I(Pcell.Substrate)  POINTS  X.MAX=5E-10
  ... +         INTEG  TITLE="Example 12B - Charge:  SEU with Argon Ion"
```

Figure 12-11 Second part of the simulation input file *mdex12b*

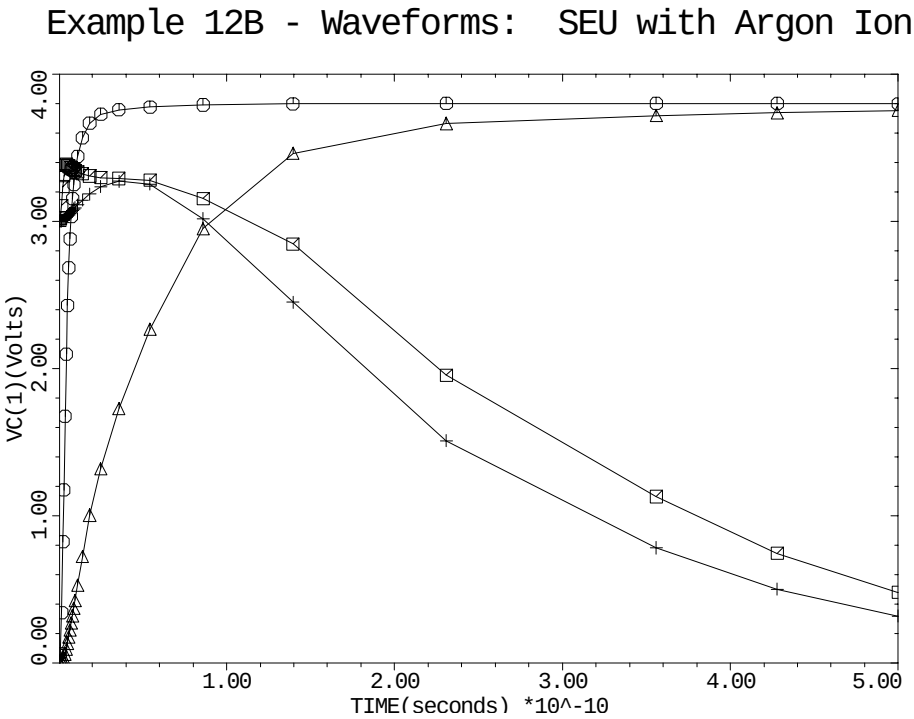


Figure 12-12 Waveform from **PLOT.1D** at lines 38 through 45 in input file *mdex12b*, Figure 12-11

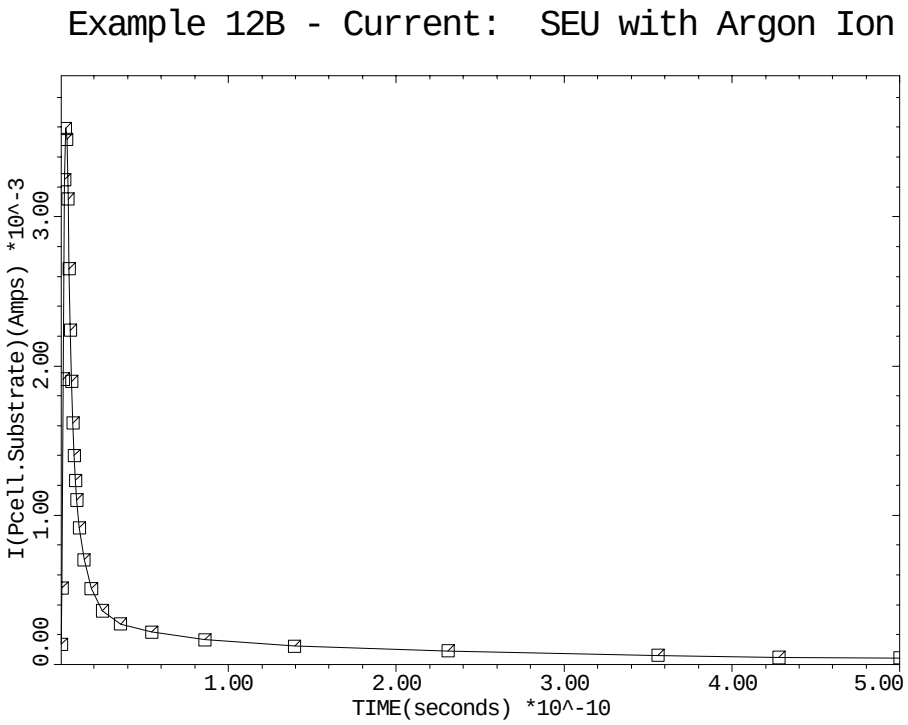


Figure 12-13 Current from **PLOT.1D** at line 46 in input file *mdex12b*, Figure 12-11

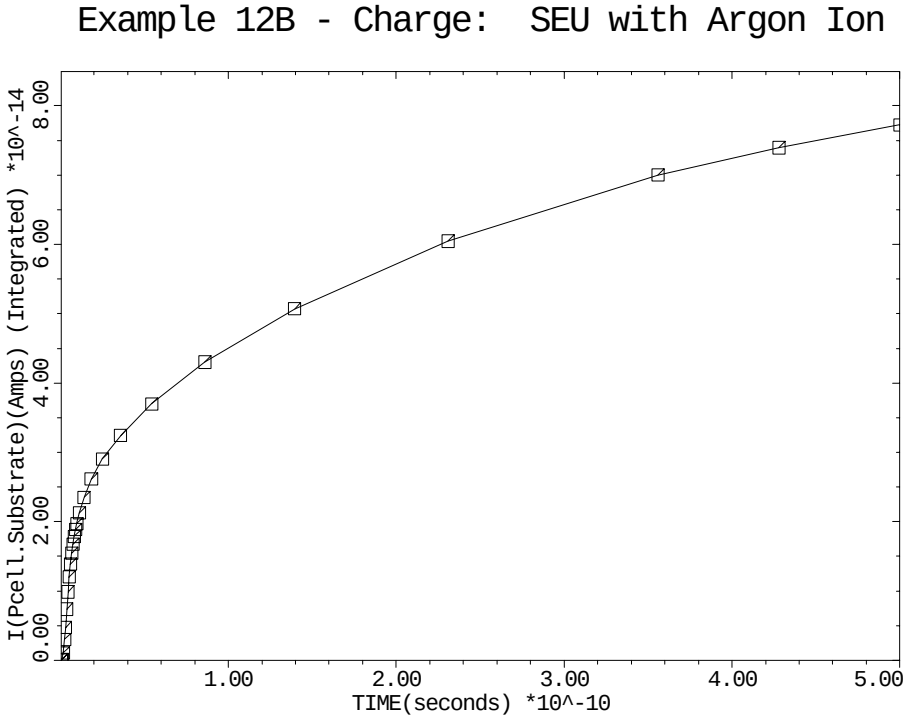


Figure 12-14 Charge from **PLOT.1D** at line 47 in input file *mdex12b*, Figure 12-8

Transfer Curves for CMOS Pair with Compact Load

This example calculates the DC and transient transfer curves for a CMOS pair under realistic loading conditions. The simulation involves two Medici transistors and lumped elements. The example also demonstrates how easy it is to transfer the mesh, circuit and solution between simulations using files.

The simulation has five separate files and steps.

1. File *mdex13a* creates the N-channel transistors.
2. File *mdex13b* creates the P-channel transistors.
3. File *mdex13c* then calculates the DC transfer curve.
4. File *mdex13d* performs the transient analysis.
5. File *mdex13e* calculates the small signal response of the circuit over a range of frequencies.

Circuit Specifications

The circuit simulated is given in Figure 12-15. The first stage is composed of Medici transistors and the second (Load) stage of SPICE transistors. Resistors *R9* and *R10*, and capacitor *C8* represent parasitic interconnect resistance and capacitance. A five-volt power supply is used and all transistors have channel lengths of 1.25 microns. The channel widths of the N-channel transistors is 1.5 microns and 3.0 microns is used for the P-channel devices.

Generation of the Simulation Structure

This section details the generation of the simulation structure.

N-Channel Transistors

The Medici transistors are based upon those developed in "Generation of the Simulation Structure" on page 4-2. Input file *mdex13a* generates the N-channel device, and is shown in Figures 12-16 and 12-17.

The *mdex13a* is similar to *mdex1*, with the following exceptions:

- The intermediate plotting steps have been removed.
- A **SAVE** statement has been added (at line 47).
- Some of the material parameters have been slightly altered.

  The material parameters were altered as a test that they work and are transferred between simulations.

Note that as well as the doping and mesh, all the interface, material, models and contact information are stored in the **SAVE** file.

A plot of the finished structure is given in Figure 12-18.

P-Channel Transistors

The Medici transistors are based upon those developed in "Generation of the Simulation Structure" on page 4-2. Figures 12-19 and 12-20 show the input files which generates the P-channel transistor. This device is similar to *mdex1* with the following exceptions:

- The sign of all doping has been inverted.
- Some of the material parameters have been slightly altered.

A plot of the finished structure is shown in Figure 12-21. Using the mesh files *MDE13AM* and *MDE13BM*, generated by *mdex13a* and *mdex13b*, circuit simulation of the CMOS pair is now possible.

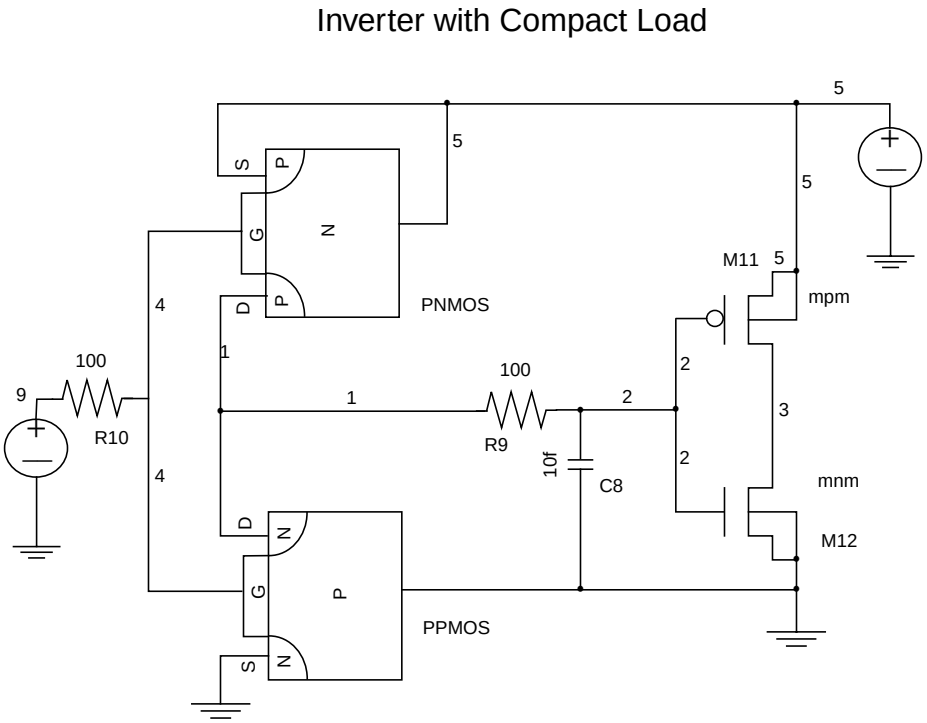


Figure 12-15 CMOS pair with compact load

```
 1... TITLE     Example 13A - 1.5 Micron N-Channel MOSFET

 2... COMMENT   This device is similar to Example 1 from the MEDICI
  ... +         manual.  It has 480 grid points and 890 elements.

 3... COMMENT   Specify a rectangular mesh
 4... MESH      SMOOTH=1
 5... X.MESH    WIDTH=3.0  H1=0.125

 6... Y.MESH    N=1   L=-0.025
 7... Y.MESH    N=3   L=0.
 8... Y.MESH    DEPTH=1.0  H1=0.125
 9... Y.MESH    DEPTH=1.0  H1=0.250

10... COMMENT   Eliminate some unnecessary substrate nodes
11... ELIMIN    COLUMNS  Y.MIN=1.1

12... COMMENT   Increase source/drain oxide thickness using SPREAD
13... SPREAD    LEFT   WIDTH=.625  UP=1  LO=3  THICK=.1  ENC=2
14... SPREAD    RIGHT  WIDTH=.625  UP=1  LO=3  THICK=.1  ENC=2

15... COMMENT   Use SPREAD again to prevent substrate grid distortion
16... SPREAD    LEFT   WIDTH=100   UP=3  LO=4  Y.LO=0.125

17... COMMENT   Specify oxide and silicon regions
18... REGION    NAME=Silicon  SILICON
19... REGION    NAME=Oxide    OXIDE    IY.MAX=3

20... COMMENT   Electrodes:
21... ELECTR    NAME=Drain  X.MIN=2.5  IY.MAX=3
22... ELECTR    NAME=Gate  X.MIN=0.625  X.MAX=2.375  TOP
23... ELECTR    NAME=Source  X.MAX=0.5  IY.MAX=3
24... ELECTR    NAME=Substrate BOTTOM

25... COMMENT   Specify impurity profiles and fixed charge
26... PROFILE   P-TYPE  N.PEAK=3E15  UNIFORM     OUT.FILE=MDE13DS
27... PROFILE   P-TYPE  N.PEAK=2E16  Y.CHAR=.25
28... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=0.0  WIDTH=.5
  ... +         XY.RAT=.75
29... PROFILE   N-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=2.5  WIDTH=.5
  ... +         XY.RAT=.75
30... INTERFAC  QF=1E10

31... COMMENT   Slightly alter some of the material parameters
32... MATERIAL  REGION=Silicon  TAUN0=2e-7  TAUP0=3e-7
33... MATERIAL  REGION=Oxide    PERMITTI=3.87  PRINT
```

Figure 12-16 First part of the simulation input file *mdex13a*

```
34... COMMENT   Regrid on doping
35... REGRID    DOPING  LOG  IGNORE=Oxide  RATIO=2
  ... +         SMOOTH=1  IN.FILE=MDE13DS

36... COMMENT   Specify contact parameters
37... CONTACT   NAME=Gate  N.POLY

38... COMMENT   Specify physical models to use
39... MODELS    CONMOB  SRFMOB2  FLDMOB  AUGER  CONSRH

40... COMMENT   Symbolic factorization, solve, regrid on potential
41... SYMB      CARRIERS=0
42... METHOD    ICCG  DAMPED
43... SOLVE

44... REGRID    POTEN  IGNORE=Oxide  RATIO=.2
  ... +         MAX=1  SMOOTH=1  IN.FILE=MDE13DS
45... PLOT.2D   GRID  TITLE="Example 13A - N-MOSFET Final Grid"
  ... +         FILL  SCALE

46... COMMENT   Save the mesh for circuit analysis.
47...  SAVE     MESH  OUT.FILE=MDE13AM  W.MODELS
```

Figure 12-17 Second part of the simulation input file *mdex13a*

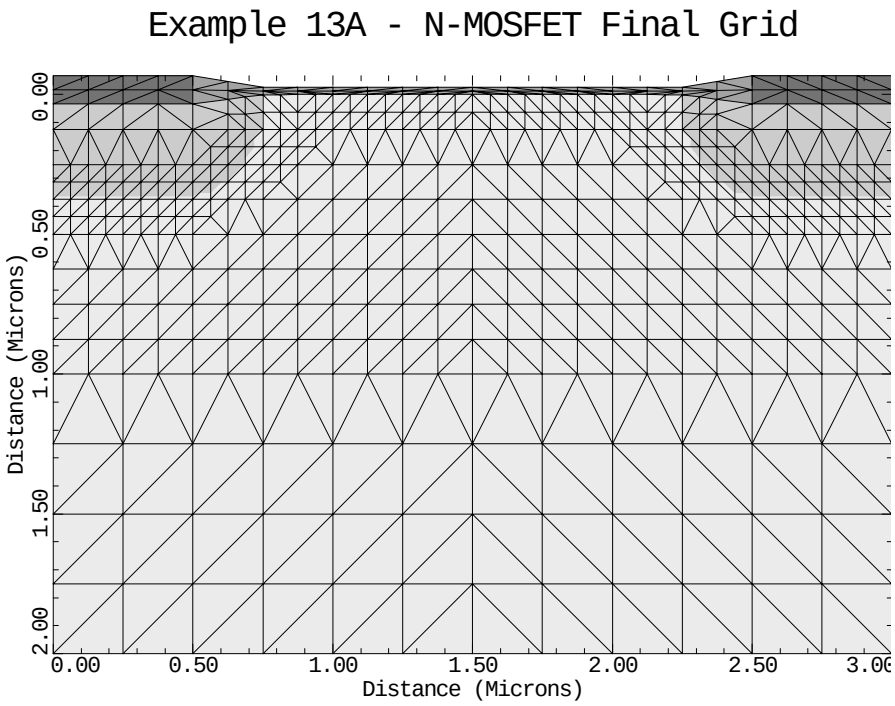


Figure 12-18 N-channel device from **PLOT.2D** at line 45 of file *mdex13a*, Figure 12-17

```
 1... TITLE     Example 13B - 1.5 Micron P-Channel MOSFET

 2... COMMENT   This device is similar to Example 1 from the MEDICI
  ... +         manual.  It has 480 grid points and 890 elements.

 3... COMMENT   Specify a rectangular mesh
 4... MESH      SMOOTH=1
 5... X.MESH    WIDTH=3.0  H1=0.125

 6... Y.MESH    N=1   L=-0.025
 7... Y.MESH    N=3   L=0.
 8... Y.MESH    DEPTH=1.0  H1=0.125
 9... Y.MESH    DEPTH=1.0  H1=0.250

10... COMMENT   Eliminate some unnecessary substrate nodes
11... ELIMIN    COLUMNS  Y.MIN=1.1

12... COMMENT   Increase source/drain oxide thickness using SPREAD
13... SPREAD    LEFT   WIDTH=.625  UP=1  LO=3  THICK=.1  ENC=2
14... SPREAD    RIGHT  WIDTH=.625  UP=1  LO=3  THICK=.1  ENC=2

15... COMMENT   Use SPREAD again to prevent substrate grid distortion
16... SPREAD    LEFT   WIDTH=100   UP=3  LO=4  Y.LO=0.125

17... COMMENT   Specify oxide and silicon regions
18... REGION    NAME=Silicon  SILICON
19... REGION    NAME=Oxide    OXIDE    IY.MAX=3

20... COMMENT   Electrodes:
21... ELECTR    NAME=Drain  X.MIN=2.5  IY.MAX=3
22... ELECTR    NAME=Gate  X.MIN=0.625  X.MAX=2.375  TOP
23... ELECTR    NAME=Source  X.MAX=0.5  IY.MAX=3
24... ELECTR    NAME=Substrate BOTTOM

25... COMMENT   Specify impurity profiles and fixed charge
26... PROFILE   N-TYPE  N.PEAK=3E15  UNIFORM     OUT.FILE=MDE13DS
27... PROFILE   N-TYPE  N.PEAK=2E16  Y.CHAR=.25
28... PROFILE   P-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=0.0  WIDTH=.5
  ... +         XY.RAT=.75
29... PROFILE   P-TYPE  N.PEAK=2E20  Y.JUNC=.34  X.MIN=2.5  WIDTH=.5
  ... +         XY.RAT=.75
30... INTERFAC  QF=1E10

31... COMMENT   Slightly alter some of the material parameters
32... MATERIAL  REGION=Silicon  TAUN0=2.5e-7  TAUP0=3.5e-7
33... MATERIAL  REGION=Oxide    PERMITTI=3.85 PRINT
```

Figure 12-19 First part of the simulation input file *mdex13b*

```
34... COMMENT   Regrid on doping
35... REGRID    DOPING  LOG  IGNORE=Oxide  RATIO=2
  ... +         SMOOTH=1  IN.FILE=MDE13DS

36... COMMENT   Specify contact parameters
37... CONTACT   NAME=Gate  P.POLY

38... COMMENT   Specify physical models to use
39... MODELS    CONMOB  SRFMOB2  FLDMOB  AUGER  CONSRH

40... COMMENT   Symbolic factorization, solve, regrid on potential
41... SYMB      CARRIERS=0
42... METHOD    ICCG  DAMPED
43... SOLVE

44... REGRID    POTEN  IGNORE=Oxide  RATIO=.2  MAX=1
  ... +         SMOOTH=1  IN.FILE=MDE13DS
45... PLOT.2D   GRID  TITLE="Example 13B - P-MOSFET Final Grid"
  ... +         FILL  SCALE

46... COMMENT   Save the mesh for circuit analysis.
47... SAVE      MESH  OUT.FILE=MDE13BM   W.MODELS
```

Figure 12-20 Second part of the simulation input file *mdex13b*

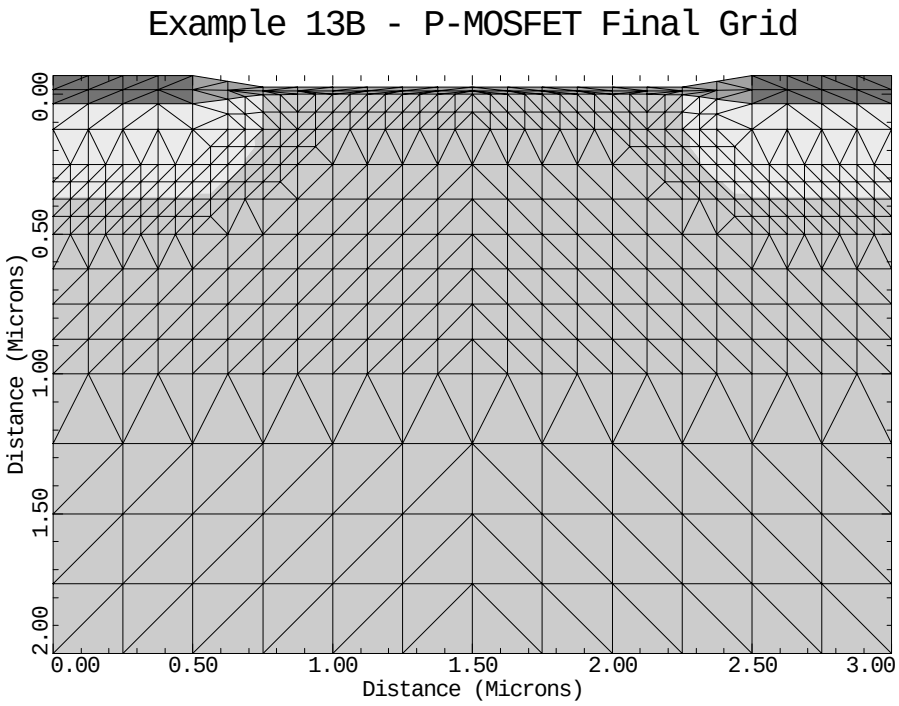


Figure 12-21 P-channel device from **PLOT.2D** at line 45 of the file *mdex13b,* Figure 12-20

Generation of Circuit and DC Simulations

The input file for the circuit simulation is given in Figures 12-22 and 12-23. As in the proceeding, examples begin by entering circuit mode and creating the circuit. The power supply *VDD* at line 5 is a fixed five volt supply.

Procedures

The generation of the circuit and DC simulations have the following procedures.

Time Dependent Input Source

The input source, *VIN* at line 7 is a time dependent source. In the present DC simulation, the time parameters are of no consequence since the value of *VIN* is swept by the **.SOLVE** statements at lines 33-35.

To use the same input file for the transient simulation later on, specify the time dependence of the input source now. Further explanation of this source is given in the next section on transient analysis.

P- and N-Channel Devices

The two Medici devices *PNMOS* and *PPMOS* appear at lines 13 and 11 respectively.

N-Channel

PNMOS is the N-channel device and has a channel width of 1.5 microns. Since the *z* dimension, which corresponds to the width of the MOSFET, is unmodeled; the width is accounted for simply by multiplying the terminal currents by 1.5.

P-Channel

PPMOS is the P-channel transistor and has a width of 3 microns. The terminal currents printed and plotted by Medici are the total currents, which includes these width multipliers (currents are in Amps, not Amps/micron). Lines 11 and 13 the devices meshes created in *mdex13a* and *mdex13b* are referenced with the **FILE** parameters.

Other Circuit Elements

Lines 15 to 24 create the remainder of the circuit and are similar to the proceeding examples.

Initial Guess

At line 26, the initial guess for the circuit voltages are given using a **.NODESET** statement.

Note:

The specification of the initial guess is very important and should never be omitted.

Solution

Line 28 exits circuit mode. Lines 29 and 30 perform a zero carrier (Possion only) solution as an initial guess. The voltages specified on the **.NODESET** statement are used as the bias. At line 32, the voltage updates are limited to 0.3 V. This option prevents the solver from generating physically unrealistic voltages which may hinder convergence.

DC Sweep

Switch to a 2 carrier simulation. Line 33 performs the first DC sweep of the voltage at source *VIN* from 0.0 Volts to 1.5 Volts in 0.5 volt steps (4 steps in all). The proper choice of **PREVIOUS** or **PROJECTI** is automatically made for the initial guess.

Line 34 continues the DC sweep at voltage source *VIN* form 1.8 Volts up to 2.6 Volts using steps of 0.20 Volts. The circuit is expected to change state rapidly from 1.8 to 2.6 Volts so smaller steps are used here to allow better resolution of the curve. Likewise the 1.5 volt solution generated by line 33 serves as the initial guess for the 1.8 volt solution at line 34 and the zero carrier step is not needed. Line 35 finishes the DC sweep. Switch back to 0.5 volt steps for the final part of the curve between 3.0 and 5.0 Volts.

Saving the Solution and Mesh

Lines 36 and 37 specify files to store the solutions and mesh. The solution file *MD13S00* is incremented as the program sweeps through the DC curve just as in normal Medici. The solution file contains potential, electron and hole information for all Medici transistors and all the circuit voltages and currents.

The mesh file *MDE13MS* contains all **MESH**, **DOPING**, **MATERIAL**, **MOBILITY**, and **MODEL** information for all the Medici transistors in the circuit. In addition, the mesh file contains the circuit itself, all component values and all model parameters.

Plots

Line 39 plots the output voltage at the drains of the Medici MOSFETs (Node 1) versus the input voltage (Node 9) at the gates (See Figure 12-24). Note how the node voltages to plot are specified as **AXIS**=*VC*(<*node*>). Line 40 similarly plots the power supply current *IC(VDD)* versus the input voltage (Node 9), shown in Figure 12-25.

Lines 42 and 43 generate a potential plot for the N-channel device (*PNMOS*) when VIN=5.0V, shown in Figure 12-26. Note that the device to plot is specified by the **STRUCTUR** parameter on the **PLOT.2D** statement (line 42). Lines 45 and

46 generate a similar plot for the P-channel transistor *PPMOS* when VIN=5.0V, shown in Figure 12-27.

```
 1... TITLE     Example 13C - DC Transfer curve for CMOS Inverter

 2... COMMENT   Enter CIRCUIT mode
 3... START     CIRCUIT
 4...       $ Power supply
 5...       VDD 5  0  5
 6...       $ Input source
 7...       VIN 9  0  PULSE 5 0 0 10p 10p 2n 5n
 8...       $ Input resistance
 9...       R10 9  4  100

10...       $ P channel MEDICI transistor
11...       PPMOS  1=Drain  4=Gate  5=Source  5=Substrate
  ...       +      FILE=MDE13BM  WIDTH=3.0

12...       $ N channel MEDICI transistor
13...       PNMOS  1=Drain  4=Gate  0=Source  0=Substrate
  ...       +      FILE=MDE13AM  WIDTH=1.5

14...       $ Poly line capacitance
15...       C8 0  2  10f
16...       $ Poly line resistance
17...       R9 1  2  100
18...       $ P channel compact load transistor
19...       M11 5  2  3  5  MPM  PS=8  PD=8  AS=6  AD=6  W=3  L=1.25
20...       $ N channel compact load transistor
21...       M12 3  2  0  0  MNM  PS=5  PD=5  AS=2  AD=2  W=1.5  L=1.25

22...       $ Models to use for the compact MOS transistors
23...       .MODEL MNM NMOS LEVEL=2 TOX=150 NSUB=3E15 LD=.15 UO=600
  ...         +  VMAX=1E7 XJ=.3 JS=1E-15 VTO=.7
24...       .MODEL MPM PMOS LEVEL=2 VTO=-.7 TOX=150 NSUB=3E15 LD=.15
  ...         +  XJ=.3 UO=300 JS=1E-15 VMAX=5E6

25...       $ Initial guess at circuit node voltages
26...       .NODESET V(1)=5.0 V(2)=5.0 V(3)=0.0 V(4)=0 V(5)=5 V(9)=0

27...       $ Return to MEDICI mode for solution and plotting
28... FINISH    CIRCUIT
```

Figure 12-22 First part of the simulation input file *mdex13c*

```
29... SYMBOL    NEWTON CARR=0
30... SOLVE     INIT

31... SYMBOL    NEWTON CARR=2
32... METHOD    N.DVLIM=0.3
33... SOLVE     ELEMENT=VIN V.ELEM=0 VSTEP=0.5 NSTEP=3
34... SOLVE     ELEMENT=VIN V.ELEM=1.80 VSTEP=0.2 NSTEP=4
  ... +         OUT.FILE=MD13S00
35... SOLVE     ELEMENT=VIN V.ELEM=3.0  VSTEP=0.5 NSTEP=4
36... SAVE      SOLUTION OUT.FILE=MD13S10
37... SAVE      MESH OUT.FILE=MDE13MS W.MODELS

38... COMMENT   Plot the circuit voltages and currents
39... PLOT.1D   X.AX=VC(9)  Y.AX=VC(1)  POINTS
  ... +         TITLE="Example 13C - Vout vs. Vin"
40... PLOT.1D   X.AX=VC(9)  Y.AX=IC(VDD)  POINTS
  ... +         TITLE="Example 13C - Power Supply Current vs. Vin"

41... COMMENT   Plot the internal behavior of transistor PNMOS
42... PLOT.2D   STRUCT=PNMOS
  ... +         TITLE="Example 13C - Potential in PNMOS with Vin=5v"
43... CONTOUR   POTEN  MIN=-1  MAX=6  FILL

44... COMMENT   Plot the internal behavior of PPMOS
45... PLOT.2D   STRUCT=PPMOS
  ... +         TITLE="Example 13C - Potential in PPMOS with Vin=5v"
46... CONTOUR   POTEN  MIN=-1  MAX=6  FILL
```

Figure 12-23 Second part of the simulation input file *mdex13c*

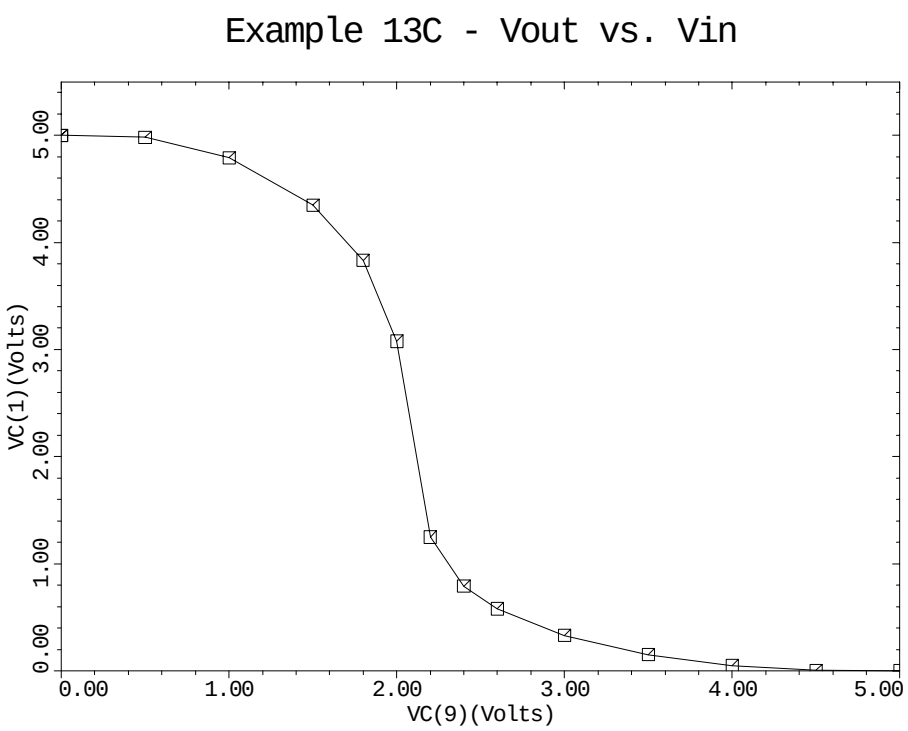


Figure 12-24 Output voltage from **PLOT.1D** at line 39 of the file *mdex13c*, Figure 12-23

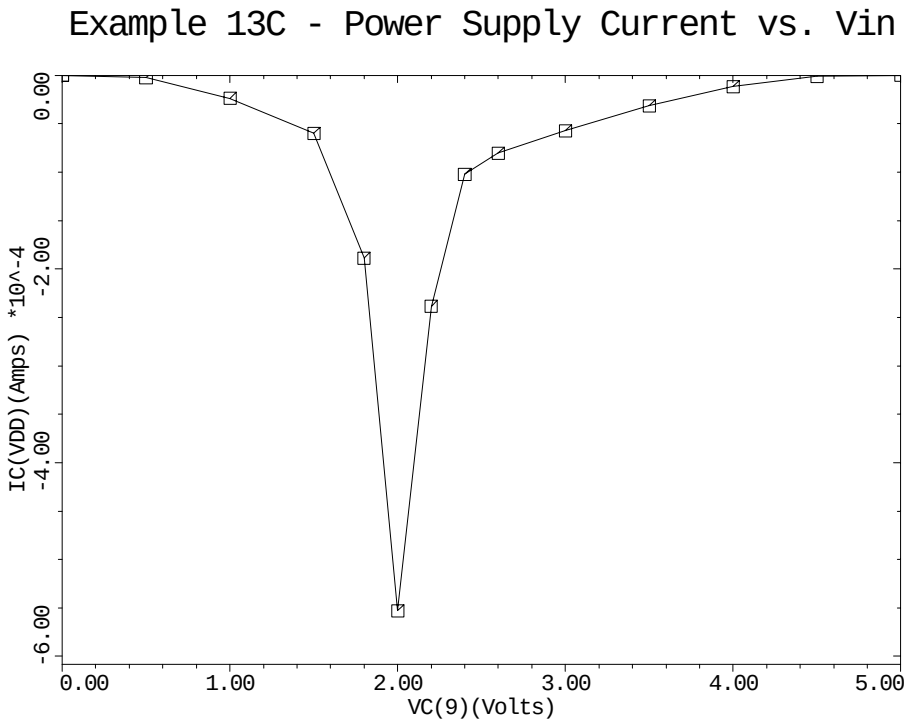


Figure 12-25 Power supply current from **PLOT.1D** at line 40 of the file *mdex13c*, Figure 12-23

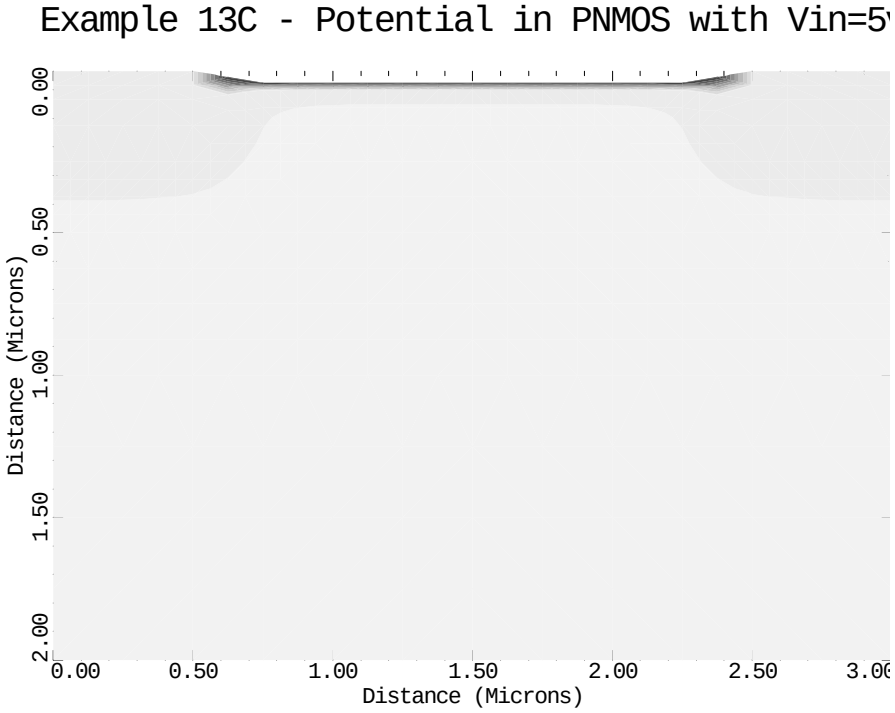


Figure 12-26 Potential distribution in the N-channel device from lines 42 and 43 in file *mdex13c*, Figure 12-23

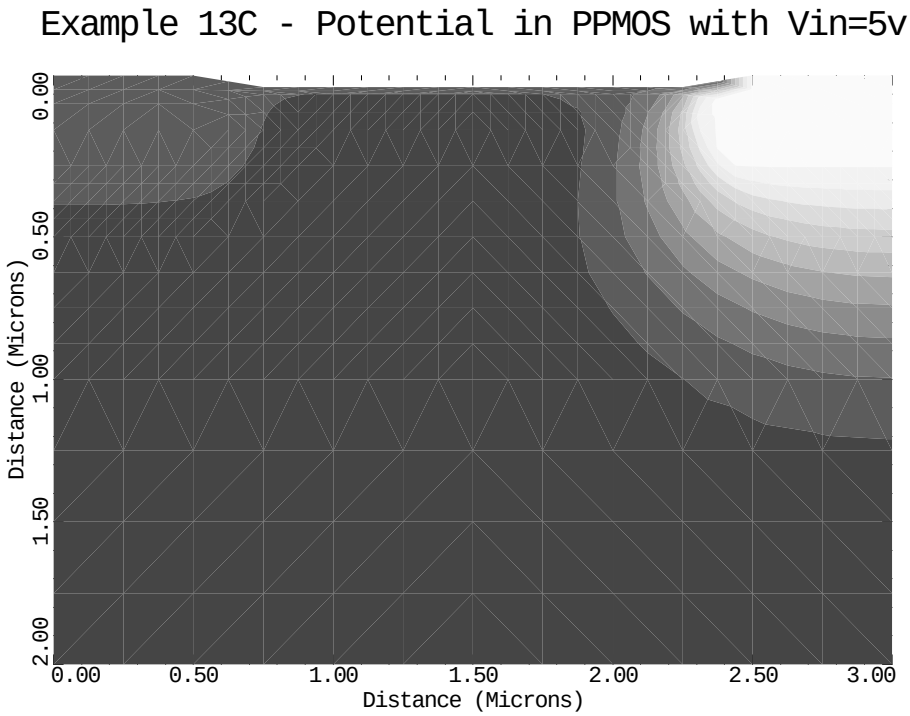


Figure 12-27 Potential distribution in the P-channel device (PPMOS) from lines 45 and 46 of the file *mdex13c*, Figure 12-23

Transient Simulation of CMOS Pair with Compact Load

This example extends the analysis of the previous section to the time domain and examines the dynamic performance of the CMOS inverter. The compact MOSFETs and layout parasitics in the steady state (DC) example had no effect on the Medici inverter. This is because of the infinite input impedance of the inverters in steady state. In this example, however the compact MOSFETs are important due to their gate capacitance, and the layout parasitics add additional delay. The circuit used is the same as in the preceding section (refer to Figure 12-15).

Procedures

The transient simulation of CMOS pair with compact load uses the following procedures.

Structure

The Medici input file to be used is presented in Figure 12-29. Circuit mode is entered with a **`START CIRCUIT`** statement at line 3. The circuit has already been created so the next step is to load in the mesh file created in example *mdex13c*. This is done at line 5 using the **`.LOAD`** statement.

Solution

Load in the solution from the final bias point of the DC sweep (corresponding to *VIN*=5V). This solution is used as the initial condition which starts the simulation process.

Input Source

Returning to *VIN* in Figure 12-22, the time dependence is as follows:

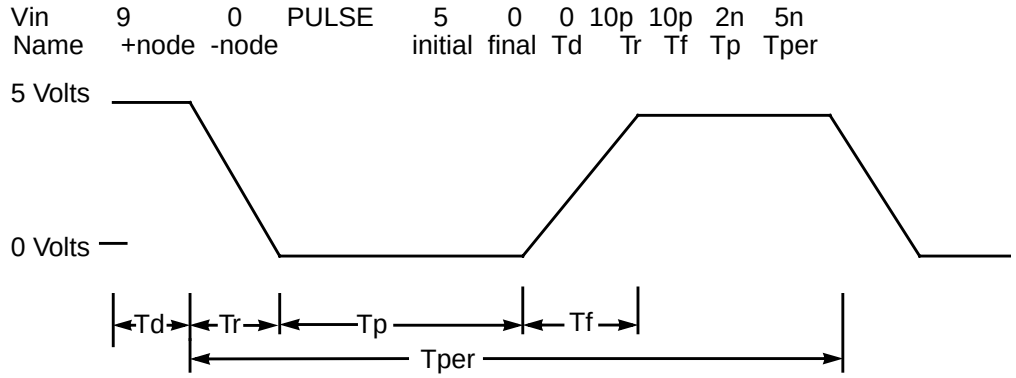


Figure 12-28 Pulse waveform used in transient simulation

The source starts out at 5 volts, immediately starts dropping to zero volts (since T_d=0) and at 2.010ns starts rising to 5 volts again. In the interest of brevity, only the first 2nS are simulated. Note also that the initial value of 5 Volts is consistent with the 5 Volt solution which was loaded at line 5. Circuit mode is exited at line 6.

Time Step Tolerance

In line 8, the time step tolerance is relaxed to 2 percent to speed the simulation.

Note:

It is not necessary to specify the initial conditions or an initial guess with an **`.IC`** ***or*** **`.NODESET`** ***statement since a valid solution was read in at line 5.***

Transient Simulation

Line 9 performs the transient simulation. **`DT`** is the initial time step to use. Subsequent time steps are computed from the local truncation error. **`TSTOP`** is the ending time for the simulation.

Inverter Plot

Line 11 generates a plot of the output voltage of the Medici inverter (Node 1) versus time which is shown in Figure 12-30. The plot shows the typical switching behavior for an inverter.

Load Inverter Plot

Line 12 generates a plot of the output of the compact inverter (Node 3) versus time which is shown in Figure 12-31. This waveform is inverted with respect to that of Figure 12-30 as is expected. No attempt was made to "fit" the compact MOSFET models other than specifying the oxide thickness, channel length and

width, substrate doping and lateral out diffusion. As a result the switching speed of the compact inverter is much faster than the Medici inverter.

Drain Current Plot

Line 13 generates a plot of the N-channel drain current versus time. Note how the drain of *PNMOS* is specified as *I*(*PNMOS.Drain*). This plot appears in Figure 12-32.

Power Supply Plot

At Line 14 a plot of the total power supply current versus time is produced.

```
 1... TITLE     Example 13D - Transient Transfer Curve for CMOS Inverter

 2... COMMENT   Enter circuit mode
 3... START     CIRCUIT

 4...       $ Load in the mesh and solution from the DC solution.
 5...       .LOAD  MESH=MDE13MS  SOLUTION=MD13S10

 6... FINISH    CIRCUIT
 7... SYMBOL    NEWTON  CARRIER=2
 8... METHOD    TOL.TIME=.02
 9... SOLVE     DT=5e-12  TSTOP=2e-9

10... COMMENT   Plot the circuit voltages and currents
11... PLOT.1D   X.AX=TIME  Y.AX=VC(1)  POINTS
  ... +         TITLE="Example 13D - Output Voltage vs. Time"
12... PLOT.1D   X.AX=TIME  Y.AX=VC(3)  POINTS
  ... +         TITLE="Example 13D - Load Output Voltage vs. Time"
13... PLOT.1D   X.AX=TIME  Y.AX=I(PNMOS.Drain)  POINTS
  ... +         TITLE="Example 13D - N-Channel Drain Current vs. Time"
14... PLOT.1D   X.AX=TIME  Y.AX=IC(VDD)  POINTS
  ... +         TITLE="Example 13D - Power Supply Current vs. Time"
```

Figure 12-29 Output of simulation input file *mdex13d*

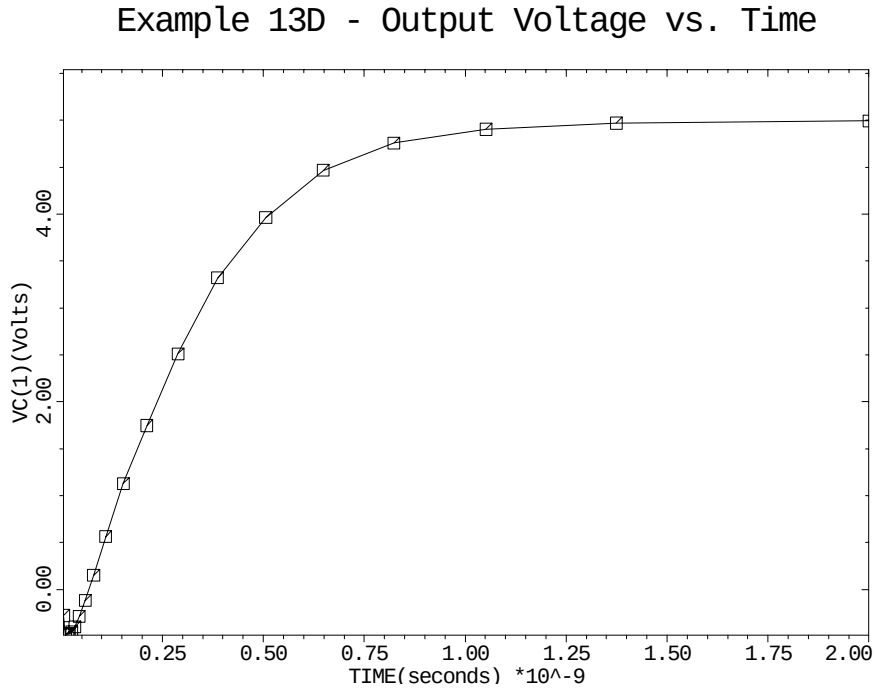


Figure 12-30 Output voltage of the inverter from **PLOT.1D** at line 11 in file *mdex13d*, Figure 12-29

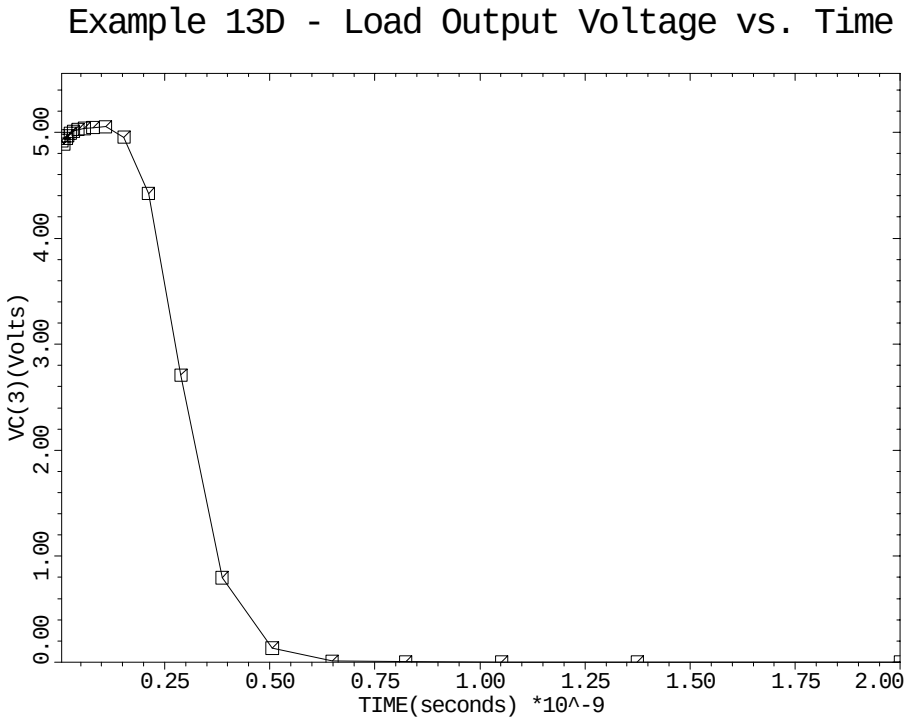


Figure 12-31 Output voltage of the compact inverter from **PLOT.1D** at line 12 in file *mdex13d*, Figure 12-29

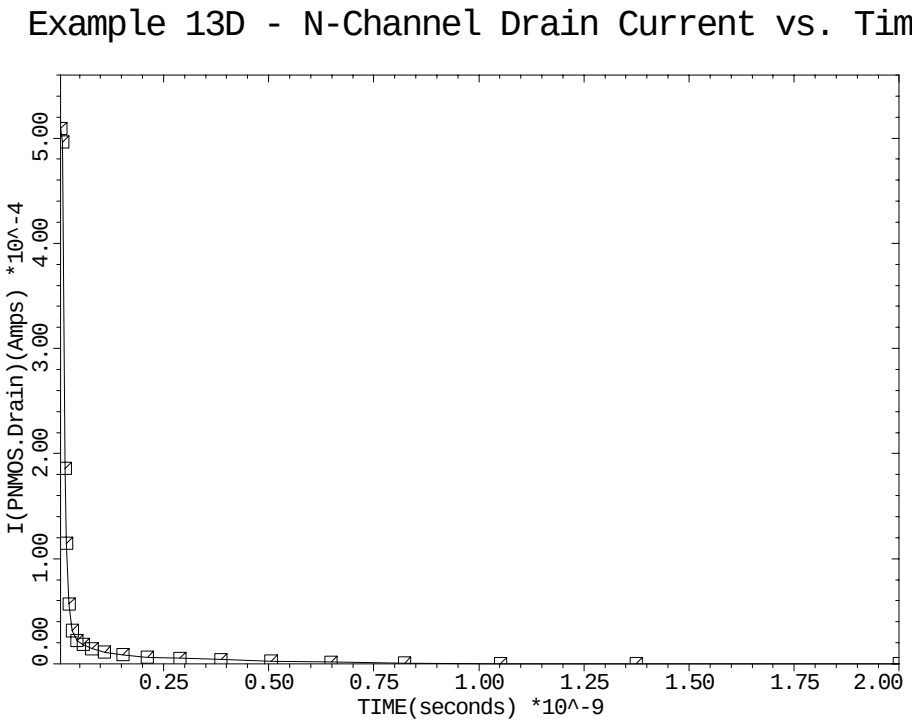


Figure 12-32 N-channel mosfet PNMOS from **PLOT.1D** at line 13 in file *mdex13d*, Figure 12-29

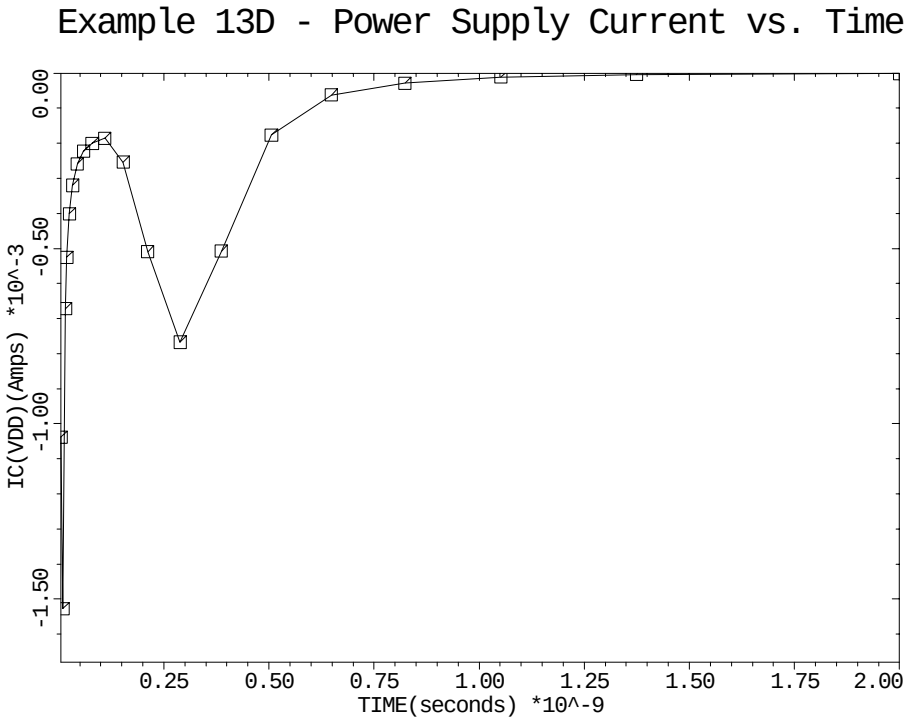


Figure 12-33 Power supply current from **PLOT.1D** at line 14 of the file *mdex13d*, Figure 12-29

AC Analysis Simulation of CMOS Pair with Compact Load

In "Transient Simulation of CMOS Pair with Compact Load" on page 12-26 the dynamic performance of a circuit was analyzed by performing a time domain simulation. This example calculates the small signal response of the circuit for a given range of frequencies, thereby obtaining its frequency domain response.

Procedures

The AC analysis simulation of CMOS pair with compact load uses the following procedures.

Device Structure

The same circuit used in previous sections (see Figure 12-15) is used here. The input file used for the simulation is given in Figure 12-34. Circuit mode is entered at line 3. The circuit and mesh information contained in file *MDE13MS* and the initial solution from *MD13S01* are loaded at line 6 and circuit mode is exited at line 7.

Initial Solution

The initial solution corresponds to a bias point *VIN*=2V. Since p-type and n-type devices are involved, a two-carrier analysis is required on the **SYMBOLIC** statement at line 10. Cases where DC analysis can be accurately performed by simulating only one-carrier transport may still require that both carriers be included in the AC analysis.

AC Analysis Mode

The AC analysis mode is activated on the **SOLVE** statement at line 11 by specifying **AC.ANAL**. The **MULT** statement specifies that **FSTEP** is multiplied by the frequency used for the i^{th} step to calculate the frequency used for the $(i+1)^{th}$ simulation step. The total number of frequency steps is specified by **NFSTEP**. Note that the AC input source is VIN.

AC Analysis Plot

Line 12 generates a plot of the frequency dependence of circuit node #2, shown in Figure 12-35. It has been specified that the real part of the voltage at circuit node 2 is plotted. Refer to "PLOT.1D" on page 3-182 for more details. This is the output of the Medici *PNMOS-PPMOS* pair after low-passing by the *R9-C8* poly line

equivalent circuit. The cut-off appears around 250 Mhz and is due to the Medici devices since the *R9-C8* low-pass filter cuts-off around 150 Ghz.

```
 1... TITLE     Example 13E - AC Analysis CMOS Inverter

 2... COMMENT   Enter circuit mode
 3... START     CIRCUIT

 4...        $ Load in the mesh and solution from the DC solution.
 5...        $ The DC solution was calculated with in input of 2.0V.
 6...        .LOAD  MESH=MDE13MS  SOLUTION=MD13S01

 7... FINISH    CIRCUIT

 8... COMMENT   Resolve at the same bias.
 9... COMMENT   Perform AC analysis 10 points, starting at 1e6 Hz
10... SYMBOL    NEWTON CARR=2
11... SOLVE     AC.ANAL FREQ=1e6 FSTEP=3.3 MULT NFSTEP=10
  ... +         ELEMENT=VIN V.ELEMENT=2.0  AC.SOURCE=VIN
12... PLOT.1D   X.AXIS=freq  Y.AXIS=VCR(2)  POINTS  X.LOG
```

Figure 12-34 Output of the simulation input file *mdex13e*

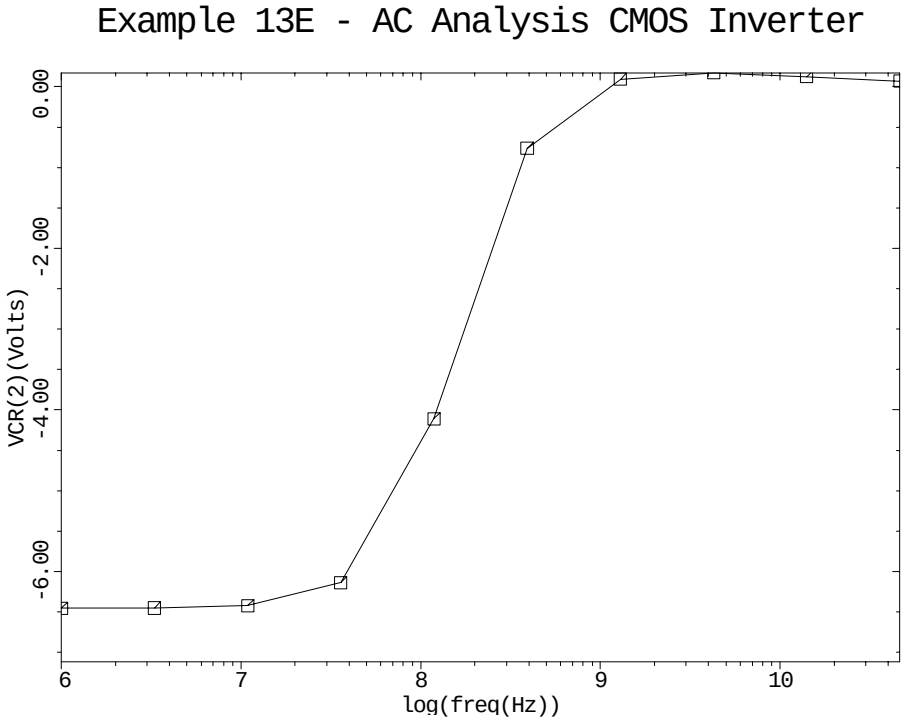


Figure 12-35 AC Analysis CMOS inverter from **PLOT.1D** at line 12 of the file *mdex13e*, Figure 12-34

Lattice Temperature Examples

Example Specifications

The Medici Lattice Temperature Advanced Application Module (LT-AAM), is capable of modeling the effects of temperature, heat generation and transfer in many materials. This chapter presents the following simulations as examples:

- Simulates a silicon on insulator SOI device and illustrates how heating can dramatically alter the drain characteristics.
- Combines the use of the LT-AAM and the Circuit Analysis AAM to simulate thermal-runaway in a bipolar device.
- Analyzes an insulated gate bipolar junction transistor (IGBT) and shows how the breakdown voltage and basic operation of the device are completely different when lattice temperature is included.

SOI Drain Characteristics Example

Most common SOI MOS devices employ a thin layer of insulating material (usually silicon dioxide) just beneath the channel of the MOSFET, to electrically insulate the MOSFET from the bulk of the semiconductor. It is unfortunate that this thin insulating layer also thermally insulates the MOSFET from the bulk (due to the low thermal conductivity of the oxide).

The result is that heat generated in the SOI MOSFET causes a much greater temperature rise than in a bulk device under similar conditions. The rise in the MOSFET temperature results in reduced carrier mobility, and a corresponding reduction in drain current transconductance and speed.

This example examines the effects of heating on the MOSFET performance by generating drain characteristics with and without the effects of lattice temperature. It also looks inside the device using plots of lattice temperature.

Procedures

The SOI drain characteristics example uses the following procedures.

Mesh and Device Structure

The input file *mdex14s* is shown in Figures 13-1 and 13-2. Line 1 uses the **TITLE** statement to reference the file. Line 2 uses the **COMMENT** statement to give the simulation a name. The mesh is generated by lines 4-31. Notice that two oxide regions are specified. The first oxide region (at line 18) is the gate oxide and is 200 Angstroms thick (from *y*=-0.02 to 0.0 microns). The silicon layer, in which the MOSFET is formed is specified at line 19, and is 0.1 micron thick (from *y*=0.0 to 0.1 microns). The buried oxide layer is created at line 20, and is 0.3 microns thick (from *y*=0.1 to 0.4 microns). The remainder of the device (down to *y*=1.0 microns) is silicon.

Electrode

The thermal electrode is created at line 27. This electrode runs along the bottom of the device and is created in the same way as the rest of the electrodes (with the exception of the parameter **THERMAL**).

Simulation Mesh Plot

The final grid and device structure are plotted at line 33, and appears in Figure 13-3.

Models

The contact and model parameters are specified at lines 35 and 36. Note that the **ANALYTIC** low-field mobility is specified instead of **CONMOB** since **CONMOB** is *not* temperature dependent. In general all the mobility models are temperature dependent except **CONMOB** and **HPMOB**.

Gate Bias and Solution

The gate bias is set to 10 volts and a zero carrier solution is performed at lines 38-40. The solution is saved to the file *TEMPSOL* to be used as an initial guess for other solutions.

Generating Drain Curves

The structure is now ready to generate two sets of curves. The first without lattice temperature effects, and the second with. Use a **LOOP** statement to generate the two solutions. Lines 42 through 52 use the **IF**, **ELSE**, **IF.END**, and **L.END** statements to control the loop. The assigned variable *HEAT* (line 43) controls whether lattice temperature is used.

First Loop

The first time through the loop, *HEAT* takes on the value F (false) by using the **ASSIGN** statement. Likewise the assigned character variables *LOGFIL* and *SOLFIL* take on the value associated with **C1**; *MD14SI1* and *MD14SS1* respectively.

Two Carrier Newton Solution

Since *HEAT* is false the first time through, **COND** is false (see line 47) and the second **SYMBOLIC** statement at line 51 is executed (a simple two carrier Newton solution). The drain is then ramped from 0.0 to 6.0 volts at lines 55-58.

Second Loop

The second time through the loop, *HEAT* is set to T for true (again at line 43), and the character variables *LOGFIL* and *SOLFIL* take on the values associated with **C2**.

Fully-Coupled Temperature Solution

Since *HEAT* is now true, the first **SYMBOLIC** statement is executed and this time a fully-coupled lattice temperature solution is performed. The term fully-coupled refers to the way in which the equations are solved; all four equations are solved as one block. With the decoupled method (**^COUP.LAT**) the simulator alternates between the semiconductor and lattice temperature equations until convergence is obtained. Both methods, however, yield the same solution. The main difference is that the fully-coupled method tends to be the most stable, but needs more memory.

Drain Curve Plot

The **PLOT.1D** statements at line 61 and 62 plot the two drain curves. The plot appears in Figure 13-4. It is easy to see how the higher temperatures at the higher drain bias reduce the mobility and cause the drain curve to bend down slightly.

Temperature Distribution Plot

The **PLOT.3D**, and **3D.SURFACE** statements at lines 67 and 68 plot the temperature distribution within the device ($V_{Drain} = 6V$ and $V_{Gate} = 10V$), shown in Fig-

ure 13-5. Notice how the large temperature gradient occurs across the oxide layer which is a poor thermal conductor.

```
 1... TITLE     Avant! MEDICI - Example 14S
 2... COMMENT   Solution of Lattice Heat Equation for SOI Structure

 3... COMMENT   Specify a rectangular mesh
 4... MESH      SMOOTH=1  OUT.FILE=MDE14SM
 5... X.MESH    WIDTH=0.5  H1=0.1    H2=0.020
 6... X.MESH    WIDTH=0.5  H1=0.020  H2=0.020
 7... X.MESH    WIDTH=0.5  H1=0.020  H2=0.1

 8... Y.MESH    N=1   L=-0.02
 9... Y.MESH    N=2   L=0.0
10... Y.MESH    DEPTH=0.1  H1=0.020  H2=0.020
11... Y.MESH    DEPTH=0.3                      SPACING=0.05
12... Y.MESH    DEPTH=1                        SPACING=0.4

13... COMMENT   Some eliminates to reduce node count
14... ELIMINAT  COLUMNS  X.MIN=0.4  X.MAX=1.1  Y.MIN=0.1
15... ELIMINAT  COLUMNS  X.MIN=0.5  X.MAX=1.0  Y.MIN=0.1
16... ELIMINAT  COLUMNS                        Y.MIN=0.5

17... COMMENT   Region definition
18... REGION    NAME=Gate_Ox  Y.MAX=0.0             OXIDE
19... REGION    NAME=Device  Y.MIN=0.0  Y.MAX=0.1  SILICON
20... REGION    NAME=Buried_Ox  Y.MIN=0.1  Y.MAX=0.4  OXIDE
21... REGION    NAME=Bulk  Y.MIN=0.4             SILICON

22... COMMENT   Electrodes
23... ELECTR    NAME=Gate  X.MIN=0.5  X.MAX=1.0
  ... +         Y.MIN=-0.02  Y.MAX=-0.01
24... ELECTR    NAME=Source  X.MAX=0.3  Y.MIN=-0.02  Y.MAX=0.0
25... ELECTR    NAME=Drain  X.MIN=1.2  Y.MIN=-0.02  Y.MAX=0.0
26... ELECTR    NAME=Substrate  BOTTOM
27... ELECTR    NAME=Heat_Sink  BOTTOM  THERMAL

28... COMMENT   Specify impurity profiles
29... PROFILE   P-TYPE  N.PEAK=1E16  UNIFORM
30... PROFILE   N-TYPE  N.PEAK=2E17  X.MIN=0.0  WIDTH=0.5  X.CHAR=0.05
  ... +         Y.MIN=0.0  DEPTH=0.1  Y.CHAR=0.01
31... PROFILE   N-TYPE  N.PEAK=2E17  X.MIN=1.0  WIDTH=0.5  X.CHAR=0.05
  ... +         Y.MIN=0.0  DEPTH=0.1  Y.CHAR=0.01

32... COMMENT   Plot the simulation mesh
33... PLOT.2D   GRID  FILL  SCALE
  ... +         TITLE="Example 14S - SOI Simulation Mesh"
```

Figure 13-1 First part of the simulation output file *mdex14s*

```
34... COMMENT   Contacts and models
35... CONTACT   NAME=Gate  N.POLY
36... MODEL     ANALYTIC  PRPMOB  FLDMOB

37... COMMENT   0-carrier solve with 10v on the gate
38... SYMB      CARRIERS=0
39... METHOD    DAMPED  ICCG
40... SOLVE     V(Gate)=10  OUT.FILE=TEMPSOL

41... COMMENT   The first pass is without the lattice heat equation.
  ... +         The second pass includes the lattice heat equation.
42... LOOP      STEPS=2
43...   ASSIGN    NAME=HEAT    L.VALUE=(F,T)
44...   ASSIGN    NAME=LOGFIL  C1=MD14SI1  C2=MD14SI2
45...   ASSIGN    NAME=SOLFIL  C1=MD14SS1  C2=MD14SS2

46...   LOAD      IN.FILE=TEMPSOL

47...   IF COND=@HEAT
48...     SYMB      CARR=2  NEWTON  LAT.TEMP  COUP.LAT
49...     METHOD    ^AUTONR
50...   ELSE
51...     SYMB      CARRIERS=2  NEWTON
52...   IF.END

53...   COMMENT   Ramp the drain to 6v
54...   LOG       OUT.FILE=@LOGFIL
55...   SOLVE     V(Drain)=0.00
56...   SOLVE     V(Drain)=0.01
57...   SOLVE     V(Drain)=0.60  VSTEP=0.6  ELEC=Drain  NSTEP=8
58...   SOLVE     V(Drain)=6.00  OUT.FILE=@SOLFIL
59... L.END

60... COMMENT   Plot I-V curves
61... PLOT.1D   IN.FILE=MD14SI1  X.AXIS=V(Drain)  Y.AXIS=I(Drain)
  ... +         LEFT=0  RIGHT=6  BOT=0.0  TOP=5E-4
  ... +         LINE=2 SYMBOL=2  COLOR=2
  ... +         TITLE="Example 14S - SOI Device: I(Drain) vs. V(Drain)"
62... PLOT.1D   IN.FILE=MD14SI2  X.AXIS=V(Drain)  Y.AXIS=I(Drain)
  ... +         UNCHANGE  SYMBOL=3  COLOR=3
63... LABEL     LABEL="Without Lattice Heat Equation"  COLOR=2  SYMB=2
  ... +         LINE=2  START.LE  LX.FIN=0.2  X=0.6
64... LABEL     LABEL="With Lattice Heat Equation"  COLOR=3  SYMB=3
  ... +         START.LE  LX.FIN=0.2
65... LABEL     LABEL="V(Gate) = 10v"  X=4.6
66... COMMENT   Lattice temperature projection plot for
  ... +   V(Gate)=10v, V(Drain)=6v
67... PLOT.3D   LAT.TEMP  X.LINES=40  Y.LINES=40  ^FRAME
  ... +         TITLE="Example 14S - SOI Lattice Temperature"
68... SURFACE   C.AUTO
```

Figure 13-2 Second part of the simulation output file *mdex14s*

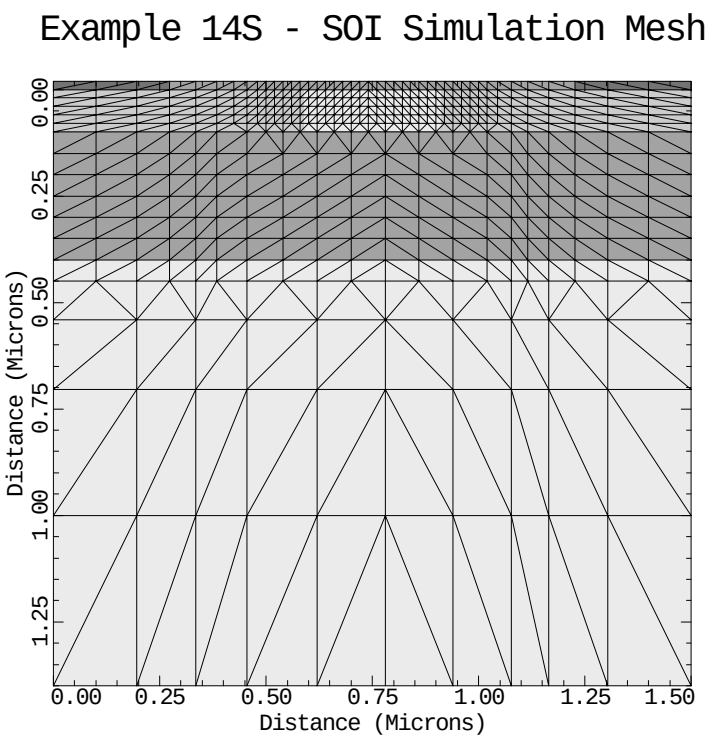


Figure 13-3 SOI simulation mesh from line 33 of file *mdex14s*, Figure 13-1

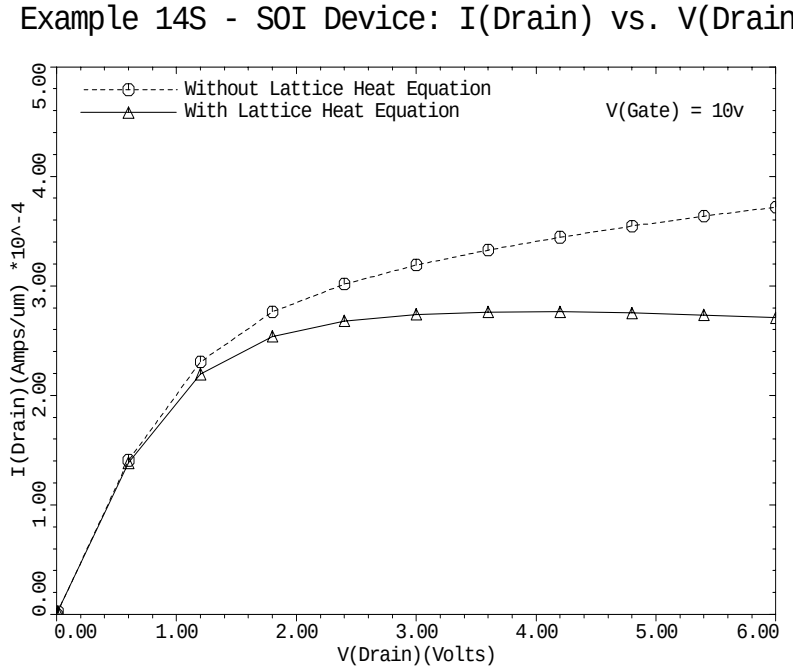


Figure 13-4 Drain curves from lines 61 through 65 of *mdex14s*, Figure 13-2

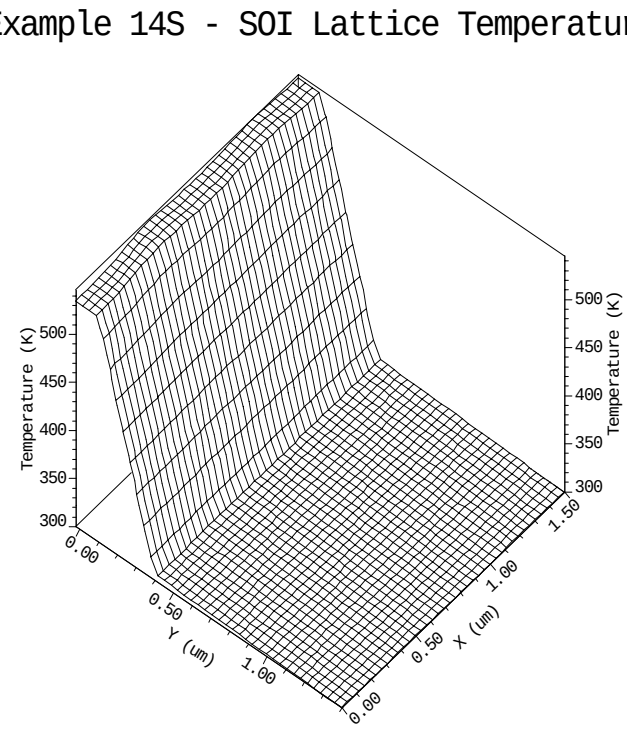


Figure 13-5 SOI lattice temperature from lines 67 and 68 of file *mdex14s*, Figure 13-2

Bipolar Transistor Thermal Run-Away Analysis

In this example both the Lattice Temperature Advanced Application Module (LT-AAM) as well as the Circuit Analysis Advanced Application Module (CA-AAM) are used to analyze thermal run-away in a bipolar structure.

Device Failure

During thermal run-away, the following feedback behavior causes the device to fail:

- The collector current causes heating of the device which raises its temperature.
- The higher temperature causes the intrinsic carrier concentration to increase.
- The increased intrinsic carrier concentration increases the emitter injection efficiency.
- Due to the increased emitter injection efficiency, less bias voltage at the base is required to maintain the level of collector current.
- Since the base bias remains constant, the collector current increases.

- The increased collector current causes greater power dissipation and more heating.

Collector Current

Rather than simply simulate a device entering run-away, however, it is more interesting to carefully examine heating within the device as the collector current is slowly increased while holding the collector-emitter voltage constant. To perform this analysis, the base bias is automatically adjusted to maintain the collector voltage at 3.0 volts. The results of the analysis is a plot of the maximum temperature within the device and the base-emitter voltage as a function of the collector current.

Collector Voltage

In order to slowly increase the collector current it is best to drive the collector of the transistor with a current source. To maintain constant collector voltage an electrical feed-back amplifier is used which automatically adjusts the base-emitter voltage. This amplifier detects the difference between the actual collector voltage and the desired value (3.0 volts) and reduces the base bias if the detected collector voltage falls below the 3.0 volts. The circuit used is shown in Figure 13-6.

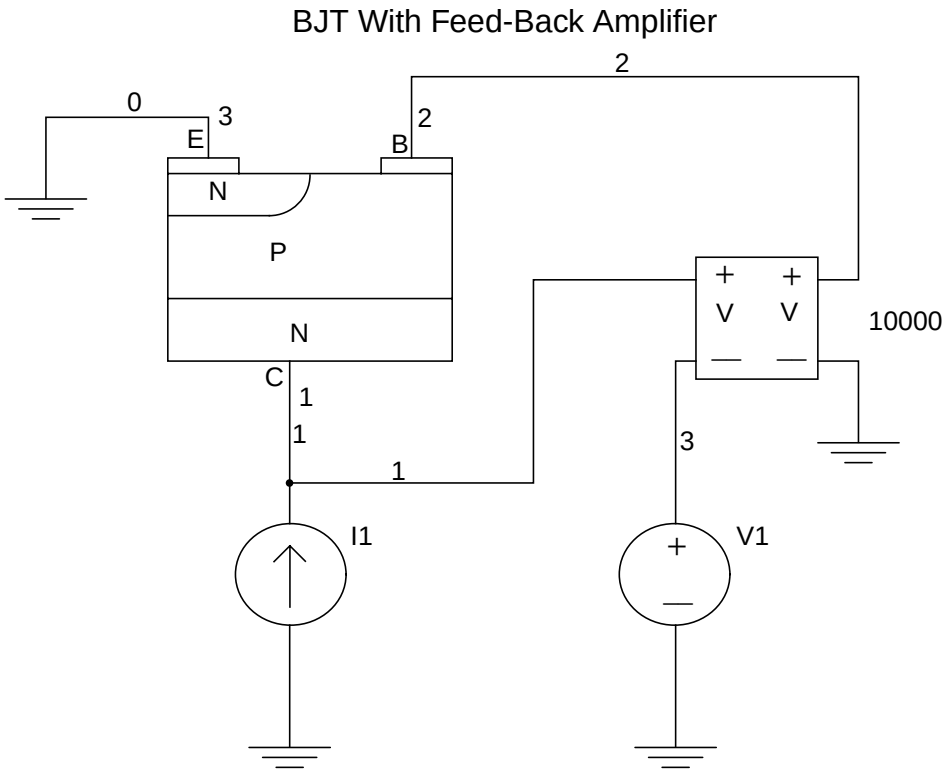


Figure 13-6 Circuit for example *mdex14b* showing transistor with feedback amplifier attached

The current source *I1*, supplies the collector current for the BJT. Voltage source *V1* is set to 3.0 volts and provides the reference for the collector voltage. The feedback amplifier is represented as a small box with four leads. This amplifier has a gain of 10,000 and its operation is governed by the following equation.

$$V_2 = 10{,}000(V_1 - V_3) = V_{be} = 10{,}000(V_{ce} - 3.0) \qquad \text{Equation 13-1}$$

From this equation it is clear that $V_{Collector}$ must be very close to 3.0 volts, since V_{Base} is normally a small voltage (around 0.8 volts).

Procedures

The Bipolar transistor thermal run-away uses the following procedures.

Device Structure

The input file for this example is presented in Figures 13-7 and 13-8.

- The BJT mesh is created using the bipolar template files *bipdef0* and *bipstr0*.
- The **CALL** statement at line 3, sets the default values by reading in the file *bipdef0*.
- Lines 59-62 set some assigned variables used during the mesh creation process. *TRANTYPE* selects whether an NPN or a PNP transistor is created (select NPN). *FILE* specifies a prefix use to create a file name to store the mesh in. Since *M14B* is specified for *FILE*, the mesh created by template *bipstr0* is written to the file *m14bmsh.*
- Lines 61 and 62 specify the vertical grid spacing at the emitter-base junction and at the base-collector junction respectively using the assigned variables *EBSP* and *BCSP.* The default values of 0.0125 and 0.0250 (which are listed in the file *bipdef0)* were increased to make the example run faster.
- At Line 64 the **CALL** statement calls the file *bipstr0* which uses the above information to generate the mesh.

The plots created by this structure are shown Figures 13-9 through 13-11.

Mesh

The **MESH** statement at line 341 reads in the mesh generated previously and reinitializes the simulator (a new **MESH** statement always reinitializes the simulator). This step is necessary since a new thermal electrode is to be added and this cannot be done unless the simulator is reinitialized.

Electrode

The **ELECTRODE** statement at line 342 creates this thermal electrode which runs along the entire bottom edge of the device.

Thermal Resistance

The **CONTACT** statement at line 344 is used to specify a thermal resistance of 8.6×10^4 K-μm/W. This value represents the thermal resistance of the package and what ever heat sink the device is attached to. The above value (8.6×10^4) is obtained by multiplying the total thermal resistance by the device width in microns.

Models

The **MODEL** statement at line 346 specifies the models to be used in the simulation. The **SAVE** statement at line 348 save the new mesh with thermal electrode, thermal resistance and models to the file *m14bmsh*. This destroys the old mesh which was in *m14bmsh,* which is not used again.

Solution

Lines 350-355 create a solution which is used as an initial guess to start the circuit simulation. The **SYMBOLIC** and **SOLVE** statements at lines 350 and 351 perform a zero-carrier solution with 3.0 volts on the collector and 0.7 volts on the base.

The **SYMBOLIC** statement at line 353 specifies the following:

- A two carrier simulation (**NEWTON CARRIERS=2**)
- With the lattice temperature equation (**LAT.TEMP**)
- Fully coupled to the semiconductor equations (**COUP.LAT**).

  The last parameter (**COUP.LAT**) causes all four equations to be solved together as a single set.

  The other alterative (**^COUP.LAT**) causes a block iterative approach to be used where the three semiconductor equations are solve first, then the lattice temperature equation.

The fully coupled method tends to be the most stable and sometimes the fastest approach, but needs more memory.

The **SOLVE** statement at line 355 calculates the solution with lattice temperature and two carriers and stores the results in the file *mde14s7*.

Circuit Simulation

Line 358 starts the circuit simulation, places Medici in **CIRCUIT** mode, and once again reinitializes the simulator. (Note, this example could have been split into three separate files, starting at lines 1,341, and 358.)

The **.OPTIONS** statement at line 360 specifies that the lattice temperature equation (**LAT**) be solved using the fully coupled method (**COU**).

Numerical Bipolar Transistor Model

The numerical bipolar transistor model is created by the **PBjt** statement at line 362.

- Circuit node 1 is attached to the collector.
- Circuit node 2 is attached to the base, and the emitter is grounded (circuit node 0).

Current Source

The device structure is read from the file *m14bmsh* created earlier. Line 364 creates the current source *I1* which drives the collector. Note that current flows into the positive terminal of the current source (which is grounded) and out of the negative terminal (which is attached to node 1, and the collector of the transistor.) Line 366 creates the 3.0 volt reference source, which is connected between ground and node 3.

Amplifier

Line 368 creates the amplifier. A voltage controlled voltage source (type **E**) element is used. The voltage at the output (nodes 2 and 0) is 10,000 times the difference of the voltages at the two inputs (nodes 1 and 3).

Initial Guess

The **.LOAD** statement at line 370 reads in file *mde14s7* which contains the solution generated earlier, at line 355. This solution is used as an initial guess for the device variables. It is doubtful that the circuit with device would have converged by itself and reading in a standard Medici solution like this is one way of helping assure convergence.

The **.NODESET** statement at line 372 specifies the initial guess for the circuit nodes. Even though a solution was read in at line 370, it is required that an initial guess be given for the circuit nodes since the solution read in did not contain any information on the circuit nodes. The specified voltages match those of the solution read in previously.

Curve Calculation

Finally the **.DC** statements at lines 374 and 375 calculate the desired curve of $I_{Collector}$ vs. V_{Base} using 12 bias steps. The starting point ($I_{Collector} = 6.0 \times 10^{-6}$ Amps) was chosen to be near the collector current calculated without the circuit attached.

The bias steps on the first **.DC** statement are smaller than those on the second **.DC** statement since once two solutions are available, projection may be used to obtain the initial guess. Once projection is available convergence improves and larger bias steps may be used.

Exiting Circuit Mode

The **FINISH** statement at line 377 takes Medici out of circuit mode for plotting.

Thermal Runaway Plot

The **PLOT.1D** statement at line 379 generates a plot of $I_{Collector}$ vs. V_{Base} Note how terminals of devices are referenced in circuit mode. For example, in the string "PBjt.Base" the device name is *PBjt* and the terminal is the base, thus this refers to the base of the transistor.

The final plot is shown in Figure 13-12. As expected, the temperature increases as the collector current increases. The base-emitter voltage reaches a maximum of about 0.78 volts and then starts to decrease.

```
  1... TITLE     Avant! MEDICI Example 14B - BJT with Lattice Temperature

  2... COMMENT   Load default parameters defining the structure and biases
  3... CALL      FILE=bipdef0  ^PRINT

 58... COMMENT   Use the default NPN transistor
 59... ASSIGN    NAME=TRANTYPE  C.VALUE=NPN
 60... ASSIGN    NAME=FILE      C.VALUE=M14B
 61... ASSIGN    NAME=EBSP      N.VALUE=.05
 62... ASSIGN    NAME=BCSP      N.VALUE=.05

 63... COMMENT   Create the structure using template "bipstr0"
 64... CALL      FILE=bipstr0  ^PRINT

340... COMMENT   Read in mesh and specify thermal electrode
341... MESH      IN.FILE=M14BMSH
342... ELECTROD  NAME=Heat_Sink  THERMAL  Y.MIN=2

343... COMMENT   Specify thermal contact
344... CONTACT   NAME=Heat_Sink  R.THERM=8.6E4

345... COMMENT   Models specification
346... MODELS    ANALYTIC  FLDMOB  PRPMOB  CONSRH  AUGER  BGN

347... COMMENT   Save mesh with models for circuit analysis
348... SAVE      MESHFILE  OUT.FILE=M14BMSH  W.MODELS

349... COMMENT   Obtain a zero carrier initial guess
350... SYMB      GUMMEL  CARRIERS=0
351... SOLVE     V(Collector)=3  V(Base)=0.7

352... COMMENT   Switch on lattice temperature and full coupling
353... SYMB      NEWTON  CARRIERS=2  LAT.TEMP  COUP.LAT

354... COMMENT   Solve again to get initial guess for
   ... +         analysis with circuit
355... SOLVE     OUT.FILE=MDE14S7
```

Figure 13-7 First part of the simulation input file *mdex14b*

```
356... COMMENT   Start circuit analysis.  Create feed-back circuit to
357... COMMENT   adjust Vbe to maintain constant Vce while increasing Ice
358... START CIRCUIT

359...      $ Specify lattice temperature and full coupling
360...      .OPTIONS LAT COU

361...      $ Numerical device model
362...      PBjt 1=Collector 2=Base 0=Emitter FILE=M14BMSH

363...      $ Collector drive source
364...      I1 0 1 6E-6

365...      $ Collector Voltage reference
366...      V1 3 0 3

367...      $ Feed-back amplifier
368...      E1 2 0 1 3 1E4

369...      $ Read in solution saved previously
370...      .LOAD  SOL=MDE14S7  STRU=PBjt

371...      $ Initial guess at circuit node voltages
372...      .NODESET  V(3)=3  V(2)=.7  V(1)=3

373...      $ Perform a ramp on collector current in two stages
374...      .DC  I1  ISTART=6E-06  ISTOP=1E-5  IINCR=4E-6
375...      .DC  I1  ISTART=5E-5   ISTOP=5E-4  IINCR=5E-5

376... $ Return to MEDICI mode for plotting
377... FINISH CIRCUIT
378... COMMENT   Plot collector current versus base bias

379... PLOT.1D   X.AXIS=V(PBjt.Base)  Y.AXIS=I(PBjt.Collector)
   ... +         ^ORDER  POINTS
   ... +         LEFT=0.6  RIGHT=1.0  TOP=6E-4  BOTTOM=0
   ... +         TITLE="Example 14B - Thermal runaway"

380... LABEL     LABEL="Vce = 3v"  X=0.1   Y=5.5E-4
381... LABEL     LABEL="T = 313K"  X=0.83  y=0.5E-4  C.SIZE=0.2
382... LABEL     LABEL="T = 327K"  X=0.83  y=1.0E-4  C.SIZE=0.2
383... LABEL     LABEL="T = 340K"  X=0.83  y=1.5E-4  C.SIZE=0.2
384... LABEL     LABEL="T = 353K"  X=0.83  y=2.0E-4  C.SIZE=0.2
385... LABEL     LABEL="T = 366K"  X=0.83  y=2.5E-4  C.SIZE=0.2
386... LABEL     LABEL="T = 379K"  X=0.83  y=3.0E-4  C.SIZE=0.2
387... LABEL     LABEL="T = 392K"  X=0.83  y=3.5E-4  C.SIZE=0.2
388... LABEL     LABEL="T = 405K"  X=0.83  y=4.0E-4  C.SIZE=0.2
389... LABEL     LABEL="T = 418K"  X=0.83  y=4.5E-4  C.SIZE=0.2
390... LABEL     LABEL="T = 432K"  X=0.83  y=5.0E-4  C.SIZE=0.2
```

Figure 13-8 Second part of the simulation input file *mdex14b*

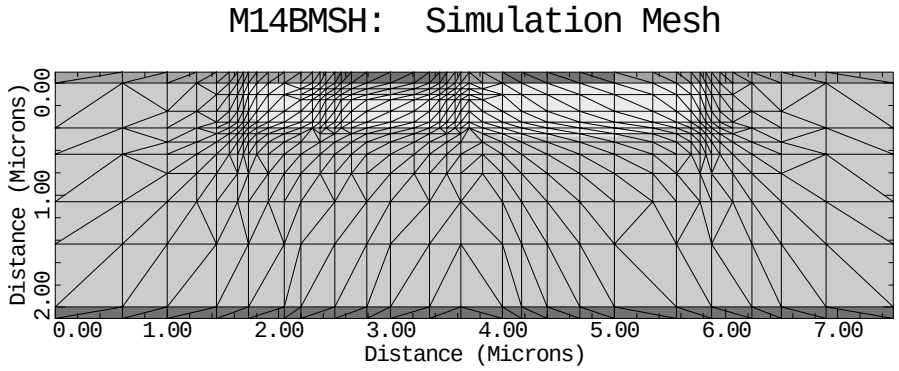


Figure 13-9 Simulation mesh from **CALL** at line 64 in file *mdex14b*, Figure 13-7

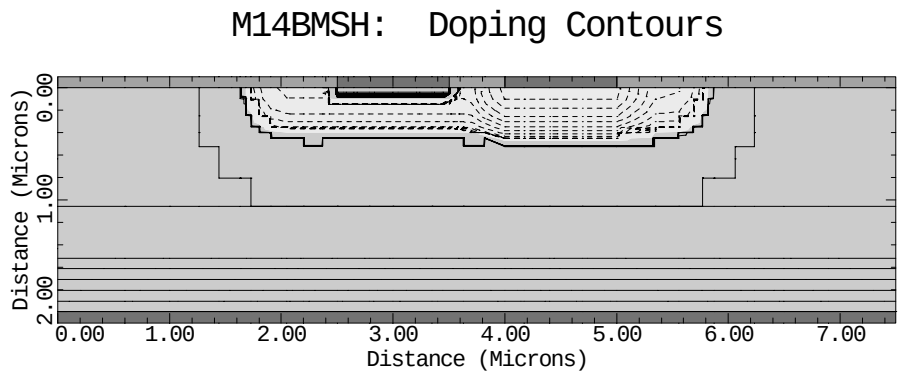


Figure 13-10 Doping contours from **CALL** at line 64 in file *mdex14b*, Figure 13-7

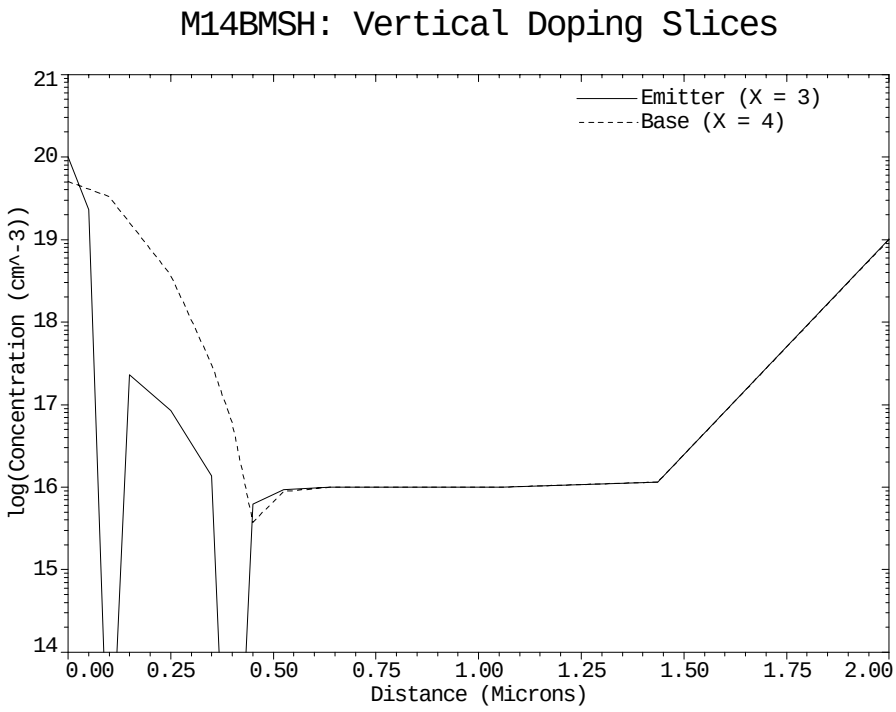


Figure 13-11 Vertical doping slices from **CALL** at line 64 in file *mdex14b*, Figure 13-7

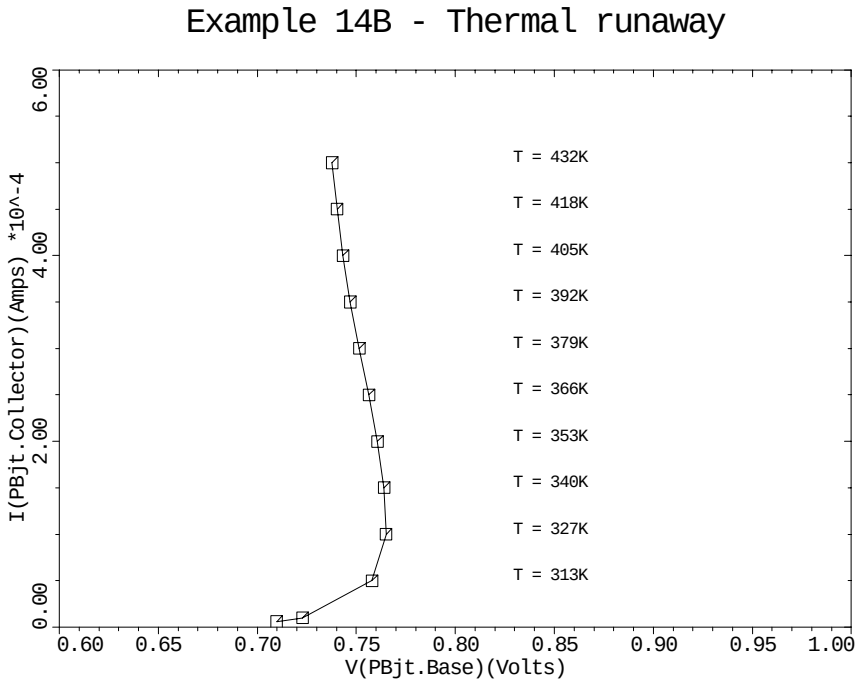


Figure 13-12 Thermal run-away from **PLOT.1D** and **LABEL** at lines 379 through 390 in file *mdex14b*, Figure 13-8

Insulated Gate Bipolar Transistor Example

This set of input files demonstrates the use of the lattice heating module (AAM) for the simulation of self-heating induced breakdown of an Insulated Gate Bipolar Transistor (IGBT). The example shows a comparison of the simulation results to those of an isothermal simulation. Thermal boundary conditions include a finite thermal resistance between the device and a heat sink.

The IGBT is an important practical power device with a low forward voltage drop and a high-impedance MOS gate control over the current flow. The device contains a thyristor-type four layer structure, which can latch-up under certain conditions. If this happens, the device cannot be turned off using the control gate and can be thermally destroyed.

To design an IGBT so that latch-up does not occur in the operating range, the device designer must consider self-heating effects in the simulation. This is demonstrated in the example under discussion, exhibiting differences in latch-up behavior between the isothermal simulation and the simulation including realistic thermal boundary conditions.

IGBT Device Structure

The structure, mesh, and doping distribution are created by the first of the input files *mdex15a* shown in Figures 13-13 and 13-14. The device is represented by a simulation domain of 30 μm by 70 μm. The gate oxide thickness is 30 nm, the drift-region is 50 μm deep with n-type doping concentration of 10^{14} cm^{-3}.

Latch-Up Stability

To reduce the resistivity of the p-wel, a second higher doping has been defined. This has the purpose of improving the latch-up stability of the four layer n-p-n-p structure.

Device Structure Plots

The device structure, grid and doping concentration contours are shown in Figure 13-15. Cross-sectional plot of the doping concentration are shown in Figure 13-16.

Boundary Conditions

A heat sink has been attached to the bottom of the device via a thermal resistance of 10^5 K/W-μm. This corresponds to including an additional 120 μm deep silicon region at the bottom of the device. The thermal resistance specification allows the use of realistic thermal boundary conditions for the nonisothermal simulation of power devices without the necessity to include very large material regions.

Non-Isothermal Solution

The nonisothermal solution is performed by the file *mdex15b* (Figure 13-17). This input file covers the entire I-V curve. After reaching 6V forward bias, the input

file switches from voltage to current boundary conditions to simulate the snap-back. The applied current value is stepped up through both first and second snap-back until very high lattice temperature indicates the thermal destruction of the device. All solution files are saved and can be analyzed with the post-processing input file discussed below.

Isothermal Solution

The isothermal simulation, i.e., simulation under the assumption T_L =300° K throughout the device, is performed by the input file *mdex15c* shown in Figure 13-18. This simulation predicts a latch-up safe device. With the gate bias of +25V, no snap-back, which would indicate that the latch-up effect is taking place, is visible in the I-V curve for forward biases up to 30V.

Plots

This section details the plots created by the preceding procedures.

I-V Curves and Internal Functions

I-V curves and plots of the internal functions clarifying the effects are created by the last input file in this set, *mdex15p*, shown in Figures 13-19 and 13-20.

Forward Bias Current Voltage

The forward bias current-voltage characteristics of the IGBT with 25V applied to the gate are plotted in Figure 13-21, calculated using the isothermal and the non-isothermal simulations. The dashed line represents the results of the isothermal simulation $T = 300^o\ K$. The solid line shows the results of the nonisothermal simulation. The I-V curve for the nonisothermal simulation shows the two snap-backs mentioned above.

Internal Device Behavior

The plots of the internal device behavior show the carrier concentrations, lattice temperature distribution and current flowlines using the solution files *MD15B10* (Figure 13-22) and *MD15B30* (Figure 13-23). These correspond to the bias point at the onset of snap-back and the last point of the simulation, respectively. These plots are of a nonisothermal solution.

It is worthwhile to look at the internal device behavior using the other saved solution files for the entire I-V curve above the first latch-up. This can be accomplished by changing the loop specification in *mdex15p* to:

```
.
.
.
LOOP    STEPS=30
   ASSIGN     NAME=SOLFIL  C.VALUE=MD15B00  DELTA=1
.
.
.
```

```
  1... TITLE     Example 15A - IGBT Grid and Initial Solution

  2... COMMENT   Create the initial mesh
  3... ASSIGN    NAME=EPI  N.VALUE=50.0
  4... ASSIGN    NAME=DPT  N.VALUE=10.0

  5... MESH      OUT.FILE=MDE15AM
  6... X.MESH    X.MAX=5.0   H2=1.25  RATIO=1.4
  7... X.MESH    X.MAX=15.0  H1=1.25  H2=0.5
  8... X.MESH    X.MAX=17.0  H1=0.60  H2=0.75
  9... X.MESH    X.MAX=30.0  H1=0.75  H2=1.5

 10... Y.MESH    N=1  LOC=-0.10
 11... Y.MESH    N=2  LOC=-0.03
 12... Y.MESH    N=3  LOC=0.0
 13... Y.MESH    Y.MAX=1.0        H1=0.02  H2=0.5
 14... Y.MESH    Y.MAX=8.0        H1=0.5   H2=1.0
 15... Y.MESH    Y.MAX=@EPI+@DPT  H1=1.0   H2=1.0  H3=5.0
 16... Y.MESH    DEPTH=@DPT       H1=1.0           RATIO=1.4

 17... ELIMINAT  COLUMNS  Y.MIN=10
 18... ELIMINAT  COLUMNS  Y.MIN=20

 19... ELIMINAT  ROWS  X.MAX=8   IY.MIN=4  Y.MAX=1.0
 20... ELIMINAT  ROWS  X.MAX=8   IY.MIN=4  Y.MAX=1.0
 21... ELIMINAT  ROWS  X.MIN=21  IY.MIN=4  Y.MAX=1.0
 22... ELIMINAT  ROWS  X.MIN=21  IY.MIN=4  Y.MAX=1.0

 23... COMMENT   Specify oxide and silicon regions
 24... REGION    NAME=Silicon  Y.MIN=0  SILICON
 25... REGION    NAME=Oxide  Y.MAX=0  OXIDE

 26... COMMENT   Electrodes
 27... ELECTR    NAME=Gate  X.MIN=5   X.MAX=15  Y.MAX=-0.03
 28... ELECTR    NAME=Cathode  X.MIN=17  Y.MAX=0
 29... ELECTR    NAME=Anode  BOTTOM

 30... COMMENT   Define a thermal electrode
 31... ELECTR    NAME=Heat_Sink  BOTTOM  THERMAL

 32... COMMENT   Specify impurity profiles
 33... PROFILE   N-TYPE  N.PEAK=1E14  UNIFORM
 34... PROFILE   N-TYPE  N.PEAK=1E17  X.MIN=0   X.MAX=2   X.CHAR=2
  ... +          Y.CHAR=6
 35... PROFILE   P-TYPE  N.PEAK=5E17  X.MIN=12  X.MAX=30  X.CHAR=1
  ... +          Y.CHAR=2
 36... PROFILE   P-TYPE  N.PEAK=1E19  X.MIN=25  X.MAX=30  Y.JUNC=8
 37... PROFILE   N-TYPE  N.PEAK=1E20  X.MIN=15  X.MAX=20  Y.JUNC=1
 38... PROFILE   P-TYPE  N.PEAK=1E20  X.MIN=20  X.MAX=30  Y.JUNC=1

 39... PROFILE   N-TYPE  N.PEAK=1E16  Y.MIN=@EPI  DEPTH=2*@DPT  Y.CHAR=1
 40... PROFILE   P-TYPE  N.PEAK=1E18  Y.MIN=@EPI+2*@DPT
   ... +   Y.JUNC=@EPI+@DPT
```

Figure 13-13 First part of the simulation input file *mdex15a*

```
41... COMMENT   Plot grid, structure, and doping
42... PLOT.2D   GRID   SCALE  FILL  TITLE="IGBT Grid"  X.OFF=3  X.LEN=5
43... PLOT.2D   BOUND  SCALE  FILL  L.ELEC=-1  ^CLEAR
  ... +         TITLE="IGBT Doping"  X.OFF=11  X.LEN=5
44... CONTOUR   DOPING  LOG  MIN=15   MAX=20   DEL=1  COLOR=2
45... CONTOUR   DOPING  LOG  MIN=-20  MAX=-15  DEL=1  COLOR=1
46... LABEL     LABEL="n"  X=15  Y=30
47... LABEL     LABEL="p"  X=15  Y=67
48... LABEL     LABEL="p"  X=27  Y=3
49... LABEL     LABEL="n"  X=16  Y=1

50... PLOT.1D   DOPING  LOG  MIN=1E12  MAX=1E21  COLOR=2  SYMB=2
  ... +         X.ST=0  X.EN=0  Y.ST=0  Y.EN=70  C.SI=.2
  ... +         TITLE="Example 15A - IGBT Doping Slices"
51... PLOT.1D   DOPING  LOG  MIN=1E12  MAX=1E21  COLOR=3  SYMB=3  UNCH
  ... +         X.ST=17 X.EN=17 Y.ST=0  Y.EN=70  C.SI=.2
52... PLOT.1D   DOPING  LOG  MIN=1E12  MAX=1E21  COLOR=4  SYMB=4  UNCH
  ... +         X.ST=30  X.EN=30  Y.ST=0  Y.EN=70  C.SI=.2
53... LABEL     LABEL="Doping, X=0"   COL=2  SYMB=2  START.LE  LX.FI=40
  ... +         X=45
54... LABEL     LABEL="Doping, X=17"  COL=3  SYMB=3  START.LE  LX.FI=40
55... LABEL     LABEL="Doping, X=30"  COL=4  SYMB=4  START.LE  LX.FI=40

56... COMMENT   Define a thermal resistance
57... CONTACT   NAME=Heat_Sink  R.THERM=1.0E5

58... COMMENT   Specify the gate workfunction
59... CONTACT   NAME=Gate N.POLY

60... COMMENT   Modify default carrier lifetimes
61... MATERIAL  SILICON  TAUN0=1E-6  TAUP0=1E-6  PRINT

62... COMMENT   Specify physical models to use
63... MODELS    ANALYTIC  PRPMOB  FLDMOB  CONSRH  AUGER  BGN

64... COMMENT   Create and save an initial solution with Vg=25v
65... SYMB      CARRIERS=0
66... METHOD    DAMPED
67... SOLVE     V(Gate)=25  OUT.FILE=MDE15AS
```

Figure 13-14 Second part of the simulation input file *mdex15a*

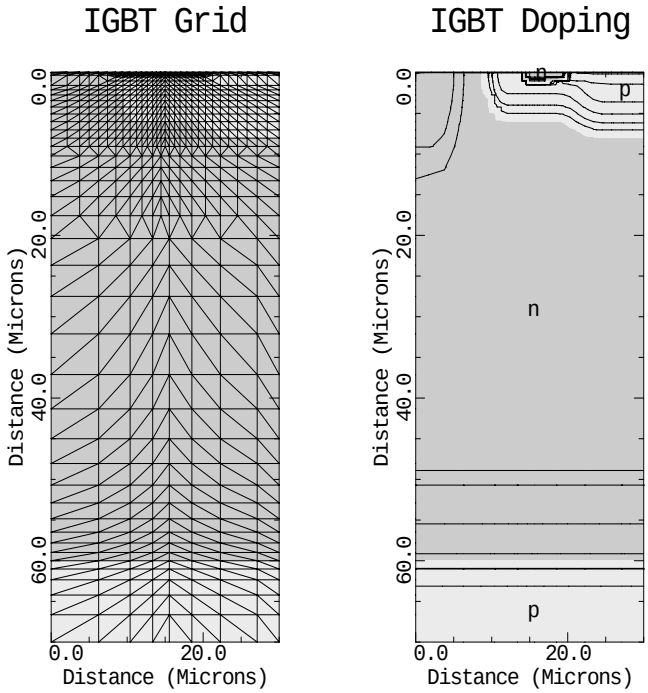


Figure 13-15 Grid and doping concentration contours from **PLOT.2D**, **CONTOUR**, and **LABEL** at lines 42 through 49 in file *mdex15a*, Figure 13-13

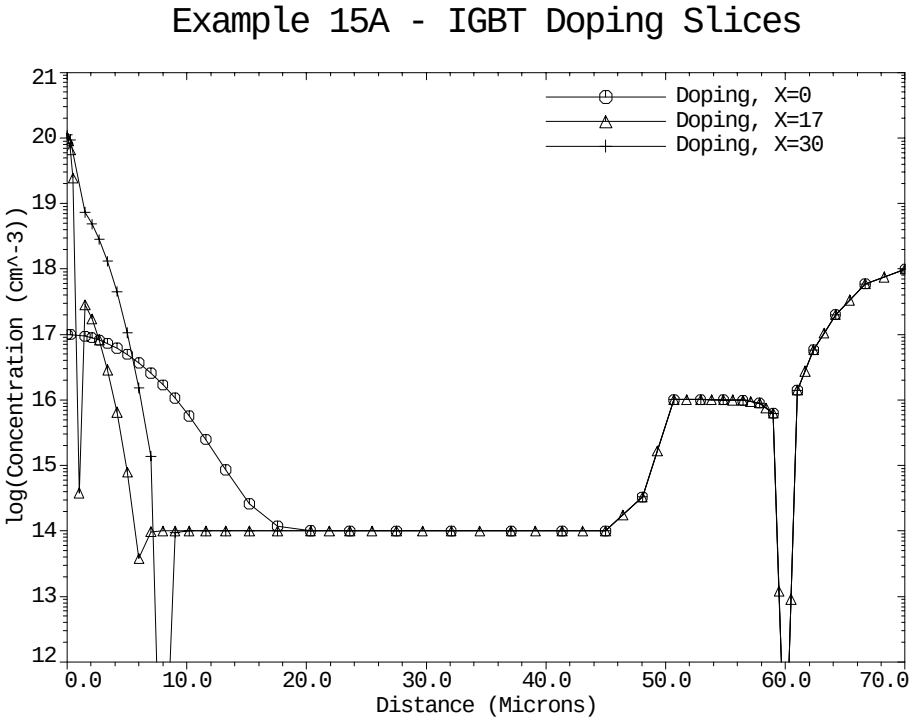


Figure 13-16 Doping concentration along three vertical lines from lines 50 through 55 in file *mdex15a*, Figure 13-13

```
 1... TITLE     Example 15B - Simulation of IGBT current vs. Vc

 2... COMMENT   Read in simulation mesh and initial solution
 3... MESH      IN.FILE=MDE15AM
 4... LOAD      IN.FILE=MDE15AS

 5... COMMENT   Begin ramping Anode, first without lattice
  ... +         lattice temperature (low current)
 6... SYMBOL    CARR=2  NEWTON
 7... LOG       OUT.FILE=MDE15BI
 8... SOLVE     V(Anode)=0.0  ELEC=Anode  VSTEP=0.1  NSTEP=2
 9... SOLVE     V(Anode)=0.5

10... COMMENT   Use block iterative lattice temperature solution
  ... +         as the device begins to heat up
11... SYMBOL    CARR=2  NEWTON  LAT.TEMP
12... SOLVE     V(Anode)=0.6  ELEC=Anode  VSTEP=0.1  NSTEP=2

13... COMMENT   For higher currents and temperatures, use a fully
  ... +         coupled solution
14... SYMBOL    CARR=2  NEWTON  LAT.TEMP  COUP.LAT

15... LOOP      STEPS=10
16...   ASSIGN    NAME=VC  N.VALUE=0.85  RATIO=1.25
17...   ASSIGN    NAME=SOLFIL  C.VALUE="MD15B01"  DELTA=1
18...   SOLVE     V(Anode)=@VC  SAVE.BIA  OUT.FILE=@SOLFIL
19... L.END

20... COMMENT   Switch to current boundary conditions at the collector
21... CONTACT   NAME=Anode  CURRENT
22... SYMB      CARR=2  NEWTON  LAT.TEMP  COUP.LAT

23... LOOP      STEPS=20
24...   ASSIGN    NAME=IC  N.VALUE=2.25E-4  RATIO=1.15
25...   ASSIGN    NAME=SOLFIL  C.VALUE="MD15B11"  DELTA=1
26...   SOLVE     I(Anode)=@IC  SAVE.BIA  OUT.FILE=@SOLFIL
27... L.END
```

Figure 13-17 Output of the simulation input file *mdex15b*

```
 1... TITLE     Example 15C - IGBT Simulation
  ... +         Without Lattice Temperature

 2... COMMENT   Read in simulation mesh initial solution
 3... MESH      IN.FILE=MDE15AM
 4... LOAD      IN.FILE=MDE15AS

 5... SYMBOL    CARR=2  NEWTON
 6... LOG       OUT.FILE=MDE15CI
 7... SOLVE     V(Anode)=0.0  ELEC=Anode  VSTEP=0.1  NSTEP=2
 8... SOLVE     V(Anode)=0.5  ELEC=Anode  VSTEP=0.1  NSTEP=3

 9... LOOP      STEPS=15
10...   ASSIGN    NAME=VC  N.VALUE=0.85  RATIO=1.25
11...   ASSIGN    NAME=SOLFIL  C.VALUE="MD15D01"  DELTA=1
12...   SOLVE     V(Anode)=@VC  SAVE.BIA  OUT.FILE=@SOLFIL
13... L.END
```

Figure 13-18 Output of the simulation input file *mdex15c*

```
 1... TITLE     Example 15P - Plot Results of the IGBT Simulations

 2... COMMENT   Read in the IGBT mesh
 3... MESH      IN.FILE=MDE15AM

 4... COMMENT   Plot I(Anode) vs. V(Anode)
 5... PLOT.1D   IN.FILE=MDE15BI  X.AXIS=V(Anode) Y.AXIS=I(Anode) LOG
  ... +         COLOR=2  SYMBOL=2  LEFT=0  RIGHT=10  BOT=1E-10  TOP=1E-2
  ... +         TITLE="Example 15P - IGBT I-V Characteristics"
  ... +         C.SIZE=0.2  ^ORDER
 6... PLOT.1D   IN.FILE=MDE15CI  X.AXIS=V(Anode)  Y.AXIS=I(Anode)  LOG
  ... +         LINE=2  COLOR=4  SYMBOL=3  UNCHANGE  C.SIZE=0.2
 7... LABEL     LABEL="V(Gate)=25v"  X=5  Y=3E-8
 8... LABEL     LABEL="Including Lattice Heat Equation"  COL=2  SYMB=2
  ... +         START.L  LX.FIN=3.5  X=4  Y=1E-8
 9... LABEL     LABEL="Lattice Temperature Fixed at 300K"  COL=4  SYMB=3
  ... +         START.L  LX.FIN=3.5  LINE=2

10... COMMENT   Plot some internal device characteristics for two
  ... +         sets of bias conditions.
11... ASSIGN    NAME=XOFF  N.VALUE=4.5
12... ASSIGN    NAME=XLEN  N.VALUE=3.0
13... ASSIGN    NAME=YLEN  N.VALUE=11.0
14... ASSIGN    NAME=CSIZ  N.VALUE=0.2

15... LOOP      STEPS=2
16...   ASSIGN    NAME=IV      C1="V=6.3v, I=1.9E-4 A/um"
  ...   +                      C2="V=2.2v, I=3.2E-3 A/um"
17...   ASSIGN    NAME=SOLFIL  C1=MD15B10  C2=MD15B30
18...   ASSIGN    NAME=CARR    C1=1e17     C2=3e18
19...   ASSIGN    NAME=TMIN    N.VALUE=( 423, 999)
20...   ASSIGN    NAME=DT      N.VALUE=(   4,  23)
21...   ASSIGN    NAME=TBOT    N.VALUE=( 423, 999)
22...   ASSIGN    NAME=TTOP    N.VALUE=( 452,1229)

23...   LOAD      IN.FILE=@SOLFIL

24...   PLOT.2D   TITLE="Example 15P - IGBT:  "@IV  ^LABELS  ^MARKS
  ...   +         X.LEN=@XLEN+3*@XOFF  Y.LEN=@YLEN+1

25...   PLOT.2D   TITLE="Holes"  T.SI=@CSIZ  ^LABELS  ^MARKS  ^CLEAR
  ...   +         X.LEN=@XLEN  Y.LEN=@YLEN
26...   CONTOUR   HOLE  LOG  FILL  MIN=15  DEL=1  LINE=1  C.INCR=1
27...   PLOT.2D   BOUND  JUNC  TITLE=""  L.ELEC=-1  ^CLEAR
  ...   +         X.LEN=@XLEN  Y.LEN=@YLEN  X.SIZE=@CSIZ  Y.SIZE=@CSIZ
28...   LABEL     LABEL=@CARR  X=10  Y=35.5  C.SI=@CSIZ

29...   PLOT.2D   TITLE="Electrons"  T.SI=@CSIZ  ^LABELS  ^MARKS  ^CLEAR
  ...   +         X.LEN=@XLEN  Y.LEN=@YLEN  X.OFF=2+@XOFF
30...   CONTOUR   ELEC  LOG  FILL  MIN=15  DEL=1  LINE=1  C.INCR=1
```

Figure 13-19 First part of the simulation input file *mdex15p*

```
31...    PLOT.2D    BOUND  JUNC  TITLE=""  L.ELEC=-1  ^CLEAR
  ...    +          X.LEN=@XLEN  Y.LEN=@YLEN  X.SIZE=@CSIZ  Y.SIZE=@CSIZ
  ...    +          X.OFF=2+@XOFF
32...    LABEL      LABEL=@CARR  X=10  Y=35.5  C.SI=@CSIZ

33...    PLOT.2D    TITLE="Temperature, dT="@DT"K"  T.SI=@CSIZ  ^LABELS
  ...    +          X.LEN=@XLEN  Y.LEN=@YLEN  X.OFF=2+2*@XOFF  ^MARKS  ^CLEAR
34...    CONTOUR    LAT.TEMP  MIN=@TMIN  DEL=@DT  LINE=1  FILL
35...    PLOT.2D    BOUND  JUNC  TITLE=""  L.ELEC=-1  ^CLEAR
  ...    +          X.LEN=@XLEN  Y.LEN=@YLEN  X.SIZE=@CSIZ  Y.SIZE=@CSIZ
  ...    +          X.OFF=2+2*@XOFF
36...    LABEL      LABEL=@TBOT"K"  X=10  Y=69  C.SI=@CSIZ
37...    LABEL      LABEL=@TTOP"K"  X=10  Y=4   C.SI=@CSIZ  COLOR=0

38...    PLOT.2D    TITLE="Flowlines"  T.SI=@CSIZ  ^LABELS  ^MARKS  ^CLEAR
  ...    +          X.LEN=@XLEN  Y.LEN=@YLEN  X.OFF=2+3*@XOFF  FILL
39...    CONTOUR    FLOWLINES  COLOR=2
40...    PLOT.2D    BOUND  JUNC  TITLE=""  L.ELEC=-1  ^CLEAR
  ...    +          X.LEN=@XLEN  Y.LEN=@YLEN  X.SIZE=@CSIZ  Y.SIZE=@CSIZ
  ...    +          X.OFF=2+3*@XOFF
41... L.END
```

Figure 13-20 Second part of the simulation input file *mdex15p*

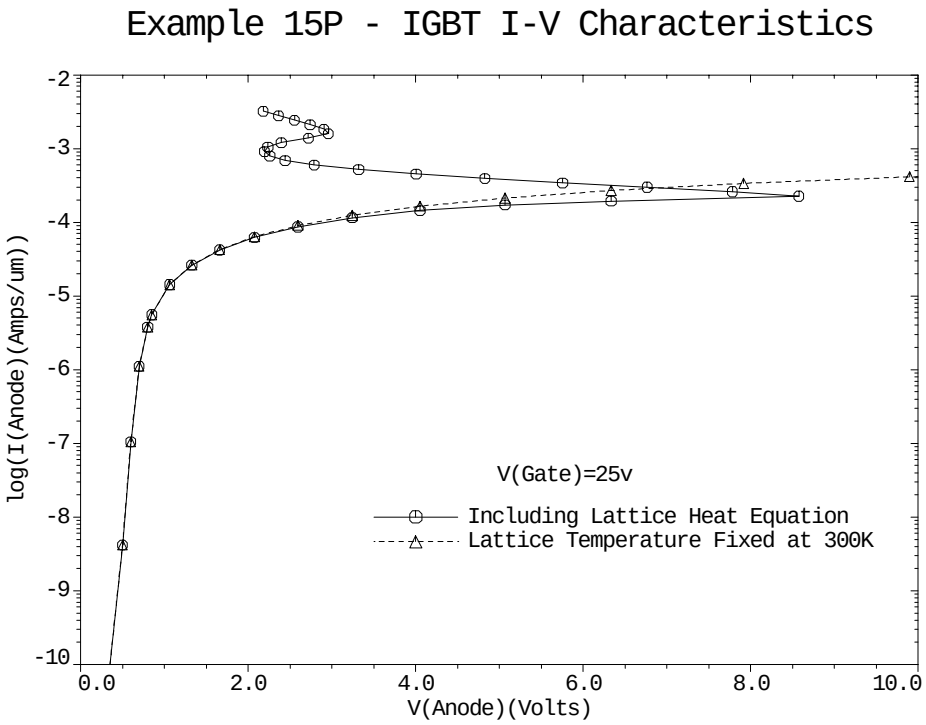


Figure 13-21 Forward bias current-voltage from lines 5 through 9 in file *mdex15p*, Figure 13-19

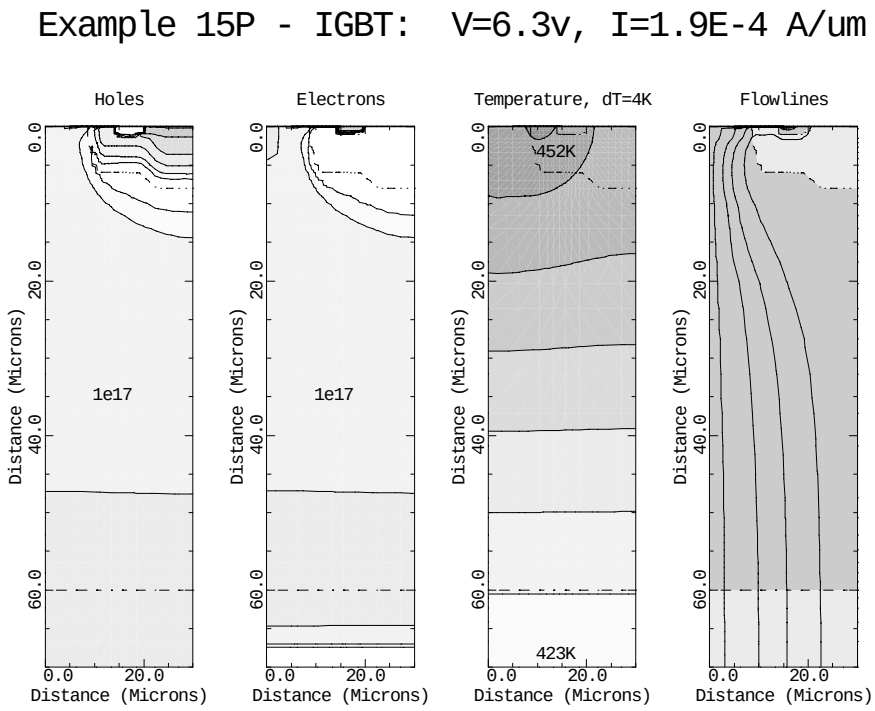


Figure 13-22 IGBT at the onset of latch-up from **PLOT.2D**, **CONTOUR**, and **LABEL** at line 15 in file *mdex15p*, Figure 13-19

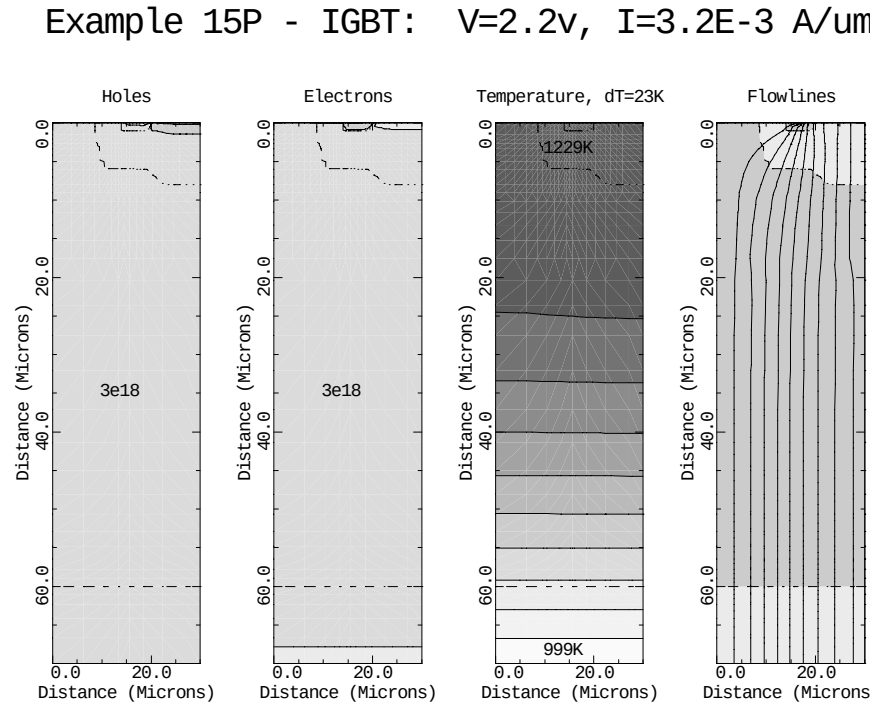


Figure 13-23 IGBT after the second latch-up from **PLOT.2D**, **CONTOUR**, and **LABEL** at line 15 in file *mdex15p*, Figure 13-19

Heterojunction Examples

Example Specification

The Medici Heterojunction Device AAM (HD-AAM) can be used to model a wide variety of semiconductor devices which employ heterojunctions. This chapter provides examples of analyses of two possible devices.

- A simple one-dimensional SiGe Heterojunction Bipolar Transistor (HBT).

  The base region of this structure consists of $Si_{0.8}Ge_{0.2}$ with mole fraction transitions occurring in the emitter-base and base-collector regions. The forward bias characteristics of this HBT are simulated.
- The basic operation of a High Electron Mobility Transistor (HEMT) that employs three different materials (GaAs, AlGaAs, and InGaAs).

  Gate characteristics for the device are calculated, and band diagrams and current flowlines are plotted.

SiGe Heterojunction Bipolar Transistor Simulation

A Heterojunction Bipolar Transistor (HBT) is formed by introducing a heterojunction at the emitter-base junction of a bipolar device. Such devices are extremely attractive due to their potential for high speed operation. HBTs typically have an emitter with a bandgap that is wider than the bandgap in the base. The potential barrier formed at the emitter-base junction under these conditions reduces the minority carrier injection from the base into the emitter to an insignificant level. This results in improved emitter efficiency and higher current gain, and leaves the base doping free as a parameter that can be adjusted for optimizing the performance of these devices.

In this example, the forward characteristics of a one-dimensional *npn* $Si_{1-x}Ge_x$ HBT are simulated. This structure uses silicon for the emitter and collector regions, and the alloy $Si_{1-x}Ge_x$ for the base region. $Si_{1-x}Ge_x$ has a narrower band-

gap than silicon with most of the offset occurring at the valence band. Thus, hole injection into the emitter is drastically reduced for this device, resulting in a very high current gain. In this example, a Ge mole fraction of x=0.2 is used.

Device Structure and Plots

The structure for this simulation is generated using the input file *mdex16* shown in Figures 14-1 and 14-2. The three **REGION** statements that have the **SIGE** parameter specified are used to define the $Si_{0.8}Ge_{0.2}$ section of the device. The first and third such statements are used to specify graded transitions in the emitter-base and base-collector regions. The mesh and structure for the device are shown in Figure 14-3. The specified doping and mole fraction are shown in Figure 14-4.

This input file also performs an initial zero bias solution and then plots the equilibrium band diagram. This is shown in Figure 14-5.

```
 1... TITLE     Avant! MEDICI Example 16 - 1D SiGe HBT Simulation

 2... COMMENT   Specify a "1D" Mesh Structure
 3... MESH      OUT.FILE=MDEX16M
 4... X.MESH    WIDTH=0.50  N.SPACES=1
 5... Y.MESH    DEPTH=0.1  H2=0.005  RATIO=1.2
 6... Y.MESH    DEPTH=0.1  H1=0.005
 7... Y.MESH    DEPTH=0.6  H1=0.005  H2=0.050

 8... COMMENT   Use a SiGe base (X.MOLE=0.2) with a graded mole fraction
  ... +         for the emitter-base and base-collector transitions.
 9... REGION    SILICON
10... REGION    SIGE  Y.MIN=0.100  Y.MAX=0.125  X.MOLE=0.0  X.END=0.2
11... REGION    SIGE  Y.MIN=0.125  Y.MAX=0.200  X.MOLE=0.2
12... REGION    SIGE  Y.MIN=0.200  Y.MAX=0.230  X.MOLE=0.2  X.END=0.0

13... COMMENT   Electrodes:  Use a majority carrier contact for the base.
14... ELECTR    NAME=Emitter  TOP
15... ELECTR    NAME=Base  Y.MIN=0.125  Y.MAX=0.125  MAJORITY
16... ELECTR    NAME=Collector  BOTTOM

17... PROFILE   N-TYPE  N.PEAK=2E16  UNIFORM
18... PROFILE   N-TYPE  N.PEAK=5E19  Y.MIN=0.80  Y.CHAR=0.125
19... PROFILE   P-TYPE  N.PEAK=2E18  Y.MIN=0.12  Y.JUNC=0.200
20... PROFILE   N-TYPE  N.PEAK=7E19  Y.JUNC=0.10

21... PLOT.2D   GRID  FILL  SCALE  TITLE="SiGe HBT Mesh"
22... FILL      SET.COLOR  C.SIGE=5  C.SILI=3  ^NP.COLOR
23... PLOT.2D   BOUND  FILL  SCALE  JUNC  L.JUNC=1  ^CLEAR  X.OFF=11.5
  ... +         TITLE="SiGe HBT Structure"
24... LABEL     LABEL="n-emitter"    X=0.17  Y=0.05
25... LABEL     LABEL="SiGe p-base"  X=0.17  Y=0.18
26... LABEL     LABEL="n-collector"  X=0.17  Y=0.50

27... PLOT.1D   DOPING  LOG  X.ST=0  X.EN=0  Y.ST=0  Y.EN=0.8  COLOR=2
  ... +         TITLE="Doping Through Emitter"
28... LABEL     LABEL="n"  X=0.05  Y=1E16
29... LABEL     LABEL="p"  X=0.15  Y=1E16
30... LABEL     LABEL="n"  X=0.50  Y=1E16

31... PLOT.1D   X.MOLE  X.ST=0  X.EN=0  Y.ST=0  Y.EN=0.8  COLOR=4  LINE=2
  ... +         ^CLEAR  ^AXES  ^MARKS  ^LABELS
32... LABEL     LABEL="mole fraction (max=0.2)"  X=0.3  Y=0.18  START.LE
  ... +         LX.FIN=0.22  ARROW  C.SI=0.2
```

Figure 14-1 First part of the simulation input file *mdex16*

```
33... COMMENT   Perform a 0-bias solution and plot the equilibrium
  ... +         band diagram.
34... SYMBOL    CARR=2  NEWTON
35... SOLVE
36... PLOT.1D   VAL NEG X.ST=0 X.EN=0 Y.ST=0 Y.EN=0.8 MIN=-1.2 MAX=1.2
  ... +         TITLE="HBT Band Diagram, Vce=0.0, Vbe=0.0"
37... PLOT.1D   CON NEG X.ST=0 X.EN=0 Y.ST=0 Y.EN=0.8 UNCH
38... PLOT.1D   POT NEG X.ST=0 X.EN=0 Y.ST=0 Y.EN=0.8 UNCH
39... PLOT.1D   QFP NEG X.ST=0 X.EN=0 Y.ST=0 Y.EN=0.8 UNCH COL=2 LINE=2
40... LABEL     LABEL="conduction"  X=0.40  Y=0.25
41... LABEL     LABEL="potential"   X=0.40  Y=-0.3
42... LABEL     LABEL="valence"     X=0.40  Y=-0.85
43... LABEL     LABEL="Fermi level"  X=0.26   Y=0.05
```

Figure 14-2 Second part of the simulation input file *mdex16*

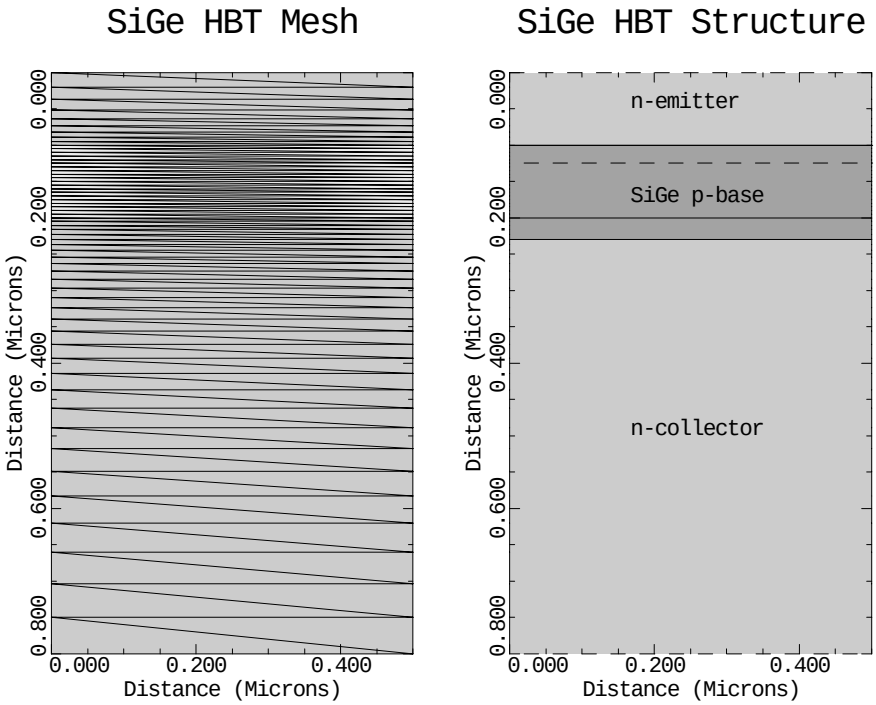


Figure 14-3 Mesh and structure generated by lines 21 through 26 in file *mdex16*, Figures 14-1

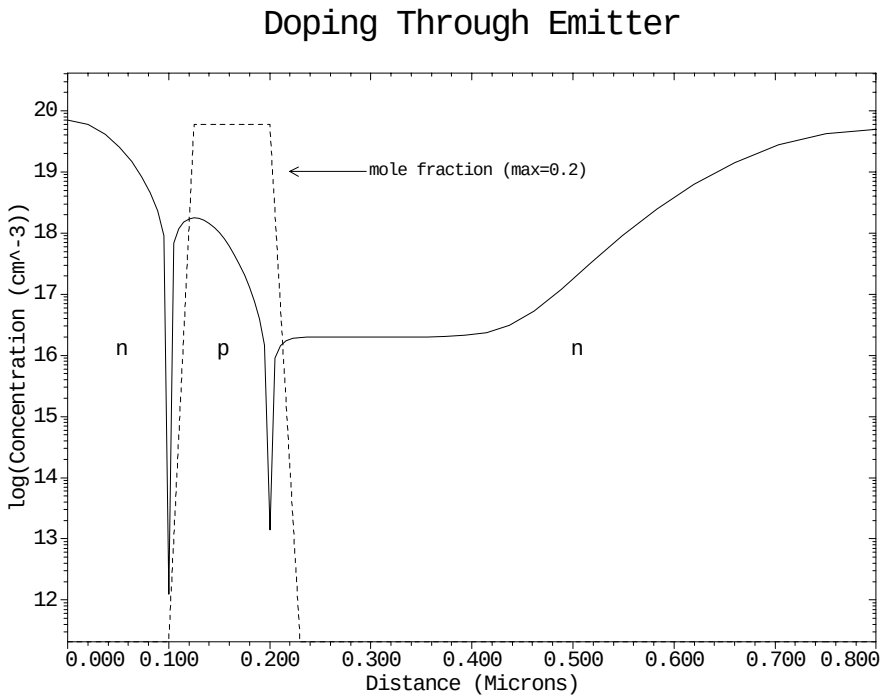


Figure 14-4 Doping and mole fraction generated by lines 27 through 32 in file *mdex16*, Figures 14-1

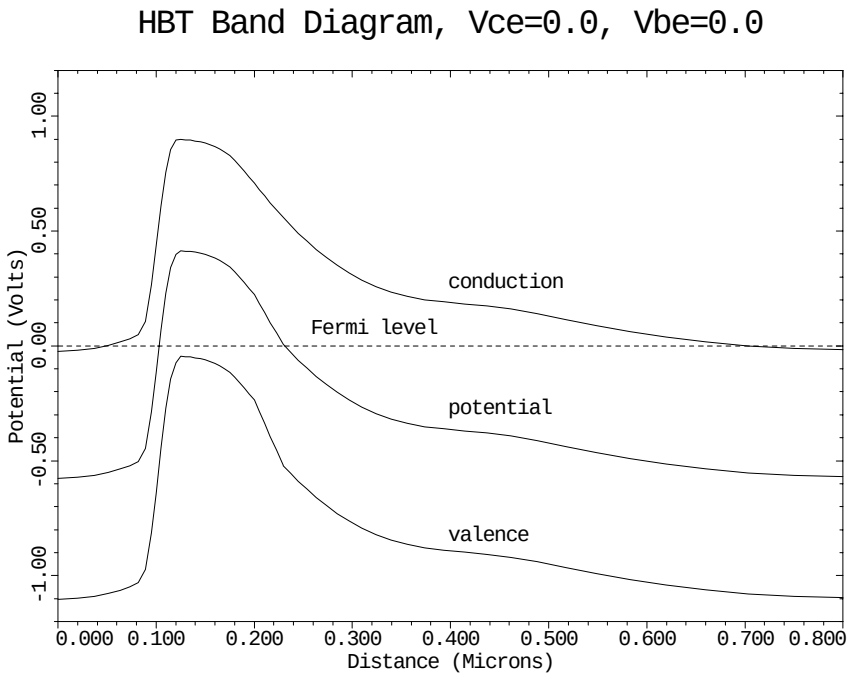


Figure 14-5 Equilibrium band generated by lines 36 through 43 in file *mdex16*, Figures 14-1

Forward Bias Simulation

The input file *mdex16f*, shown in Figure 14-6, is used to read in the HBT structure that was created with the input file *mdex16*, and then simulates its forward bias characteristics. By default, Medici uses an energy bandgap model for *strained* $Si_{1-x}Ge_x$ as a function of mole fraction *x*. A model for *unstrained* $Si_{1-x}Ge_x$ can also be used by specifying a **MATERIAL** statement with the parameters "**SIGE EG.MODEL**=3" prior to the **SOLVE** statement.

Plots

The output generated by this example includes a Gummel plot (I_c and I_b versus V_{be}) and a plot of current gain versus I_c. These plots are shown in Figures 14-7 and 14-8, respectively. The current gain shown in Figure 14-8 is one to two orders of magnitude higher than it would be for a device of the same dimensions and doping levels that uses a silicon base instead of a $Si_{1-x}Ge_x$ base.

```
 1... TITLE     Avant! MEDICI Example 16F - 1D SiGe HBT Simulation

 2... COMMENT   Read in mesh, specify contact and model parameters
 3... MESH      IN.FILE=MDEX16M
 4... CONTACT   NAME=Emitter  SURF.REC  VSURFN=1E5  VSURFP=1E5
 5... MODELS    CONMOB  FLDMOB  CONSRH  AUGER  BGN

 6... COMMENT   Use Vc=2.0, ramp the base voltage
 7... SYMBOLIC  NEWTON  CARRIERS=2
 8... METHOD    CONT.STK
 9... LOG       OUT.FILE=MDE16BI
10... SOLVE     V(Collector)=2.0  V(Base)=0.0
  ... +         ELEC=Base  VSTEP=0.05  NSTEPS=16

11... EXTRACT   NAME=Beta  EXPRESS=@I(Collector)/@I(Base)

12... COMMENT   Plot Ic and Ib vs. Vbe
13... PLOT.1D   IN.FILE=MDE16BI  X.AXIS=V(Base)  Y.AXIS=I(Collector)  LOG
  ... +         LEFT=0  RIGHT=0.9  BOTTOM=1E-15  TOP=1E-3  POINTS
  ... +         TITLE="Example 16F - HBT Gummel Plot"  COLOR=2
14... PLOT.1D   IN.FILE=MDE16BI  X.AXIS=V(Base)  Y.AXIS=I(Base)  LOG
  ... +         UNCH  COLOR=3  LINE=2  POINTS
15... LABEL     LABEL="Ic"  X=0.5  Y=7E-8
16... LABEL     LABEL="Ib"  X=0.5  Y=2E-11

17... COMMENT   Plot the current gain vs. collector current
18... PLOT.1D   IN.FILE=MDE16BI  X.AXIS=I(Collector)  Y.AXIS=Beta  X.LOG
  ... +         Y.LOG  LEFT=1E-10  RIGHT=1E-3  BOT=10  TOP=1E3  COLOR=2
  ... +         TITLE="Example 16F - HBT Gain vs. Collector Current"
   ... +         POINTS
```

Figure 14-6 Output of the simulation input file *mdex16f*

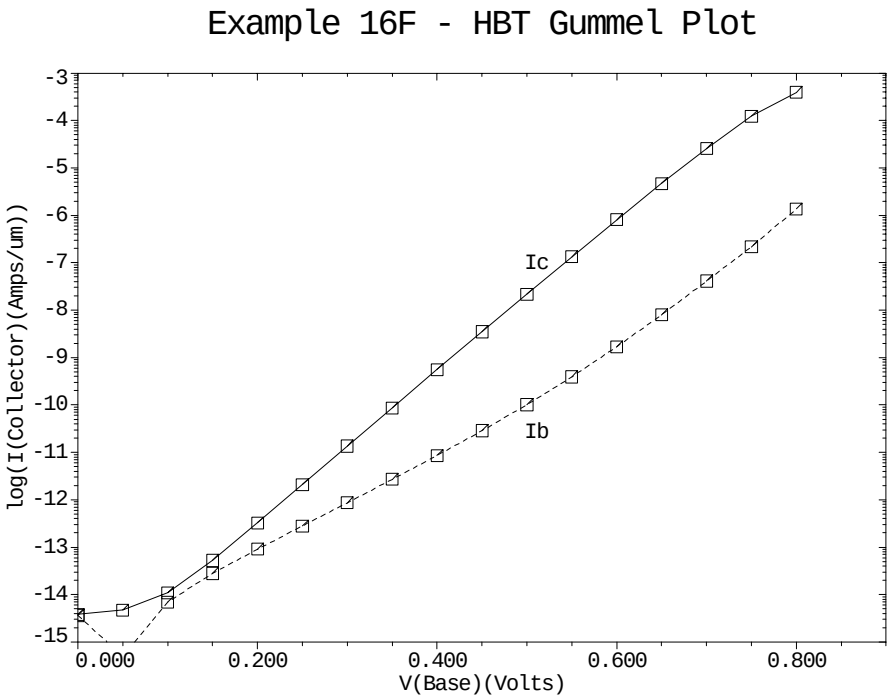


Figure 14-7 Gummel plot generated by lines 13 through 16 in file *mdex16f*, Figure 14-6

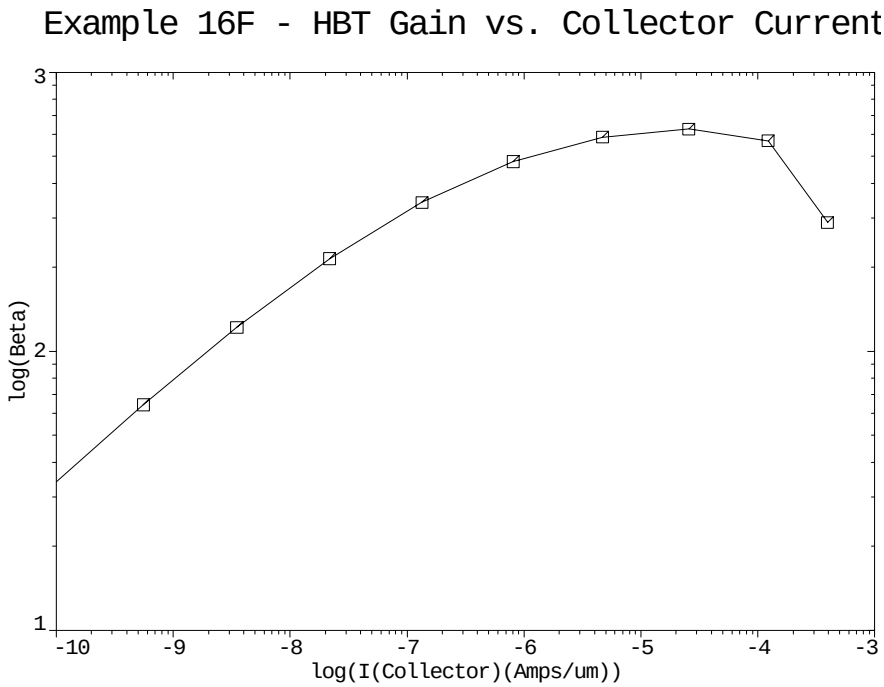


Figure 14-8 Current gain generated by line 18 in file *mdex16f,* Figure 14-6

High Electron Mobility Transistor Simulation

The High Electron Mobility Transistor (HEMT) is a small geometry heterojunction device that exploits the high electron mobility in an undoped region to achieve high speed operation. Heterojunctions are used to create a narrow undoped electron well which forms the channel for current flow. Electrons from surrounding doped regions of the device become trapped in the well resulting in a high concentration of electrons in the channel. This channel is below the surface of the device and separated from the impurity atoms (doping) which supply the electrons for the conduction process. The lack of scattering sites in the channel results in high electron mobility. In addition, the channel itself is normally constructed from a material which possesses a high mobility such as InGaAs.

Structure Generation

The HEMT simulated is shown in Figure 14-11 and the input file *mdex17* that generated the simulation in Figures 14-9 and 14-10 (this device is loosely based upon a device described in the article: "DC and Microwave Characteristics of Sub-0.1-µm Gate-Length Planar-Doped Pseudomorphic HEMT's," P.-C. Chao et al., *IEEE Trans. Electron Devices*, vol. 36, no. 3, pp. 461-473, Mar. 1989).

Device Structure

The device structure is largely planar with constant doping in most of the regions. The device regions and grid were generated using *Avant!* TCAD's Michelangelo. A refine box was used in the channel of the device to create the fine grid needed to resolve the channel. The structure and mesh generated by Michelangelo is stored in the ASCII file *mdex17.msh*.

Note:

The structure and mesh file can be examined and modified (if desired) using Michelangelo***. If changes are made, make sure to save them with a new name so that the original examples are not overwritten.***

Doping

Lines 18-24 specify the doping for the device.

- The bulk (region 1), AlGaAs spacer (region 10), and the InGaAs channel (region 2) are left undoped (although an insignificant doping concentration of 1e2/cm^3 is specified).
- The source and drain contact regions (5 and 6) are heavily doped (n-type, 1e20).
- The AlGaAs region under the gate serves as the source of channel electrons and is doped n-type, 1e18.
- At line 24 a 2D- (delta-) doping is used as an additional source of electrons.
- Line 25 sets the colors that are used to fill various material regions during subsequent plotting.

```
 1... TITLE     Example 17 GaAs - AlGaAs - InGaAs HEMT Device
 2... COMMENT   Read in the mesh file (created by the Device Editor)
 3... MESH      IN.FILE=mdex17.tif  TIF
 4... $
 5... $         Region #1 GaAs            Body
 6... $         Region #2 InGaAs          InGaAs channel
 7... $         Region #3 AlGaAs          n AlGaAs (under gate)
 8... $         Region #4 = Electrode #1 Electrode gate
 9... $         Region #5 GaAs            source N+
10... $         Region #6 GaAs            drain  N+
11... $         Region #7 = Electrode #2 Electrode source
12... $         Region #8 = Electrode #3 Electrode drain
13... $         Region #9 = Electrode #4 subst
14... $         Region #10 AlGaAs          AlGaAs Spacer
15... $
16... $
17... COMMENT   Specify Doping
18... PROFILE   REGION=1  N.TYPE  CONC=1E2  UNIF
19... PROFILE   REGION=2  P.TYPE  CONC=1E2  UNIF
20... PROFILE   REGION=3  N.TYPE  CONC=1E18 UNIF
21... PROFILE   REGION=5  N.TYPE  CONC=2E20 UNIF
22... PROFILE   REGION=6  N.TYPE  CONC=2E20 UNIF
23... PROFILE   REGION=10 N.TYPE  CONC=1E2  UNIF
24... INTERFACE REGION=(3,10) QF=2e12
25... FILL      ^NP.COL SET.COL C.GAAS=2 C.ALGAAS=3
26... $
27... MATERIAL  REGION=(1,5,6) GAAS
28... MATERIAL  REGION=2       INGAAS X.MOLE=0.85
29... MATERIAL  REGION=(3,10)  ALGAAS X.MOLE=0.2
30... $
31... CONTACT   NAME=Gate SCHOTTKY WORK=5.17
32...  $
```

Figure 14-9 The first part of simulation input file *mdex17*

Material and Mobility Parameters

Lines 27-29 assign the appropriate material types and mole fractions to the various regions. Mole-fraction and material dependent models are used during the simulation for quantities such as the band-gap, electron affinity, low- and high-field mobility,... etc (see the Heterojunction Device AAM chapter in the first volume of the manual for more details).

The plot displayed in Figure 14-11 is generated by lines 34-41 of the input file (see Figure 14-10).

Simulation

This section describes the HEMT simulation and generated plots.

Line 43 enables models for concentration dependent recombination SRH recombination, Auger recombination, and the analytic mobility model.

For gate characteristics in this particular device, it is easiest to start the simulation with the device ON and reduce the gate voltage until the device cuts off.

```
33... COMMENT   Generate plot of device structure
34... PLOT.2D   BOUNDARY FILL
35... FILL      REGION=2  COLOR=5  ^NP.COL
36... LABEL     LABEL=GaAs x=.5 y=.1
37... LABEL     LABEL=GaAs x=.1 y=.01
38... LABEL     LABEL=GaAs x=.9 y=.01
39... LABEL     LABEL=AlGaAs x=.5 y=.056
40... LABEL     LABEL=InGaAs x=.5 y=.065
41... LABEL     LABEL=AlGaAs x=.5 y=.037
42... $
43... MODELS    CONSRH  AUGER  ANALYTIC
44... $
45... COMMENT   Initial solution
46... SYMB      NEWT  CARR=0
47... SOLVE     V(Drain)=0.05  V(Gate)=0.6
48... SYMB      NEWT  CARR=2
49... SOLVE
50... $
51... SOLVE     ELEC=Gate VSTEP=-0.1 NSTEP=4
52... PLOT.1D   X.ST=0.5  X.END=0.5  Y.ST=-1  Y.EN=1  DOPING  LOG
  ... +         TITLE="MDEX17 Channel Doping & Electrons Device ON"
53... PLOT.1D   X.ST=0.5  X.END=0.5  Y.ST=-1  Y.EN=1  ELECT  LOG  UNCH  COL=2
54... LABEL     LABEL=Electrons COL=2 X=1.08 Y=1e13
55... LABEL     LABEL=Doping X=1.08 Y=1e5
56... $
57... PLOT.1D   X.ST=0.5  X.EN=0.5  Y.ST=-1  Y.EN=1  COND  TOP=1  BOT=-2
  ... +         NEG TITLE="MDEX17 Band structure Device ON"
58... PLOT.1D   X.ST=0.5  X.EN=0.5  Y.ST=-1  Y.EN=1  VAL   UNCH  NEG
59... PLOT.1D   X.ST=0.5  X.EN=0.5  Y.ST=-1  Y.EN=1  QFN   UNCH  NEG  COL=2
60... LABEL     LABEL=Cond X=1.08  Y=0.5
61... LABEL     LABEL=Qfn X=1.08  Y=0.05
62... LABEL     LABEL=Val X=1.08  Y=-1.3
63... $
64... PLOT.2D   FILL  BOUND
  ... +         TITLE="MDEX17 Current Flow, Device ON"
65... FILL      REGION=2  COLOR=5  ^NP.COL
66... CONTOUR   FLOW
67... $
68... COMMENT   Calculate the gate characteristics.
69... SOLVE     ELEC=Gate  VSTEP=-0.1  NSTEP=6
70... $
71... PLOT.1D   X.AX=V(Gate)  Y.AX=I(Drain)  POINTS
  ... +         TITLE="MDEX17 Gate Characterics of HEMT Device"
```

Figure 14-10 Second part of the simulation input file *mdex17*

Solution

At lines 46 and 47, a zero carrier (Poisson only) solution is performed as an initial guess with 0.05V on the drain and 0.6V on the gate. Then, a two carriers is performed (lines 54 and 55). Then, the gate voltage is reduced in -0.1V steps, stopping at Vg=0.2V, and some plots are generated (line 57).

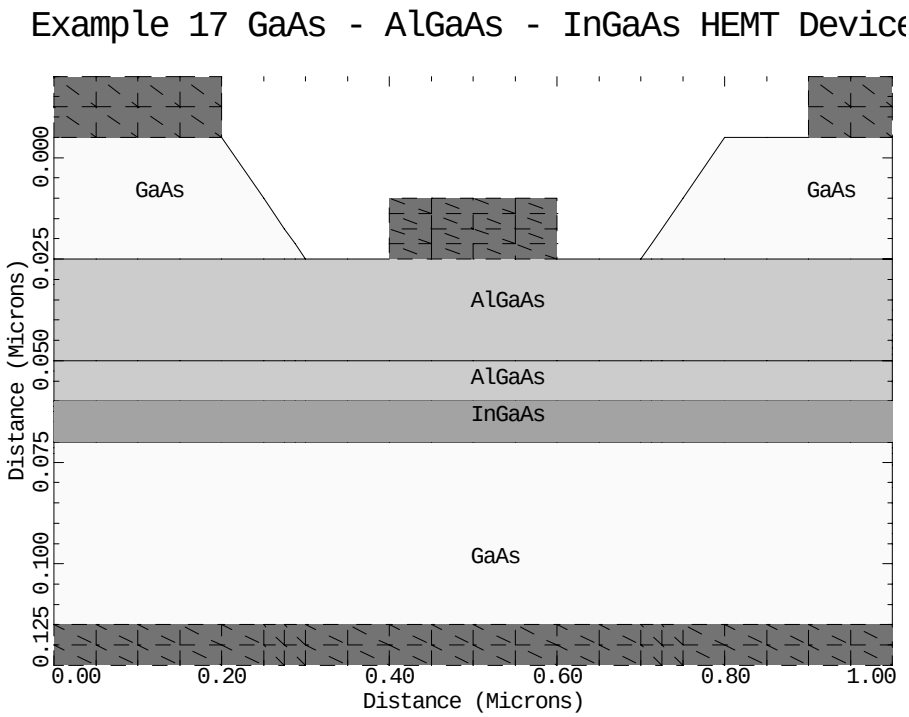


Figure 14-11 HMET device generated by lines 34 through 41 in file *mdex17*

Plots

The first plot (lines 52-55) is shown in Figure 14-12, and displays electron concentration and doping as a function of depth at the center of the channel.

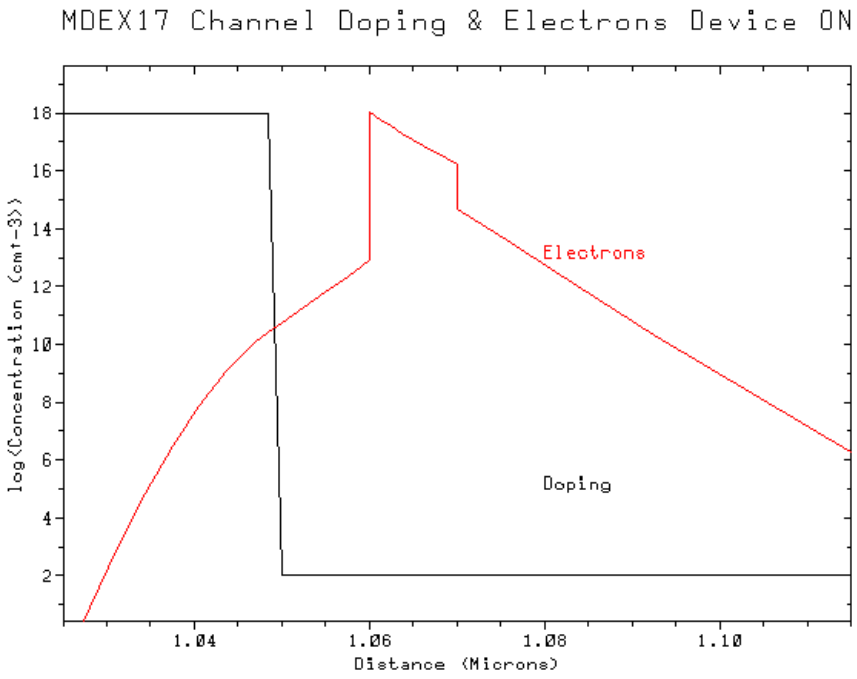


Figure 14-12 Doping and electron concentration generated by lines 52 through 55 in file *mdex17*, Figure 14-10

The peak in doping due to the Gaussian profile is plainly evident. It is also clear how the electrons have moved away from the doped region and flowed into the undoped InGaAs well. Expect these electrons to have very high mobility due to the absence of the ionized doping atoms which cause scattering and reduce the mobility.

Lines 57-62 generate a band diagram (Figure 14-13) by plotting the following:

- Conduction band
- Valence band
- Electron quasi-Fermi level

Parameter NEGATIVE

Note the specification of the **NEGATIVE** parameter on the **PLOT.1D** statements. This is necessary since band diagrams are calculated as the electron charge multiplied by the potential and the electron charge is negative. The channel well in the conduction band is clearly visible.

The quasi-Fermi level is flat in this plot since there is no current flowing in the vertical direction. Also, the conduction band has dipped below the quasi-Fermi level within the quantum well indicating an extremely large concentration of electrons there.

If the gate bias is increased, the first dip in the conduction band becomes closer and closer to the quasi-Fermi level. This results in a large concentration of electrons at the top edge of the AlGaAs spacer (where the delta doping is located). Current would then flow along the top edge of the spacer rather than in the channel. This is undesirable since the mobility of electrons flowing in the heavily doped spacer is much lower than in the undoped channel.

2D Plots

A 2D plot is now generated (lines 64-66) showing current flow in the device (see Figure 14-14). Observe that the current is flowing within the channel as expected, but that a small amount of current (about 10%) is flowing along the top of the spacer. The **FILL** statement is used to set color for a specified region.

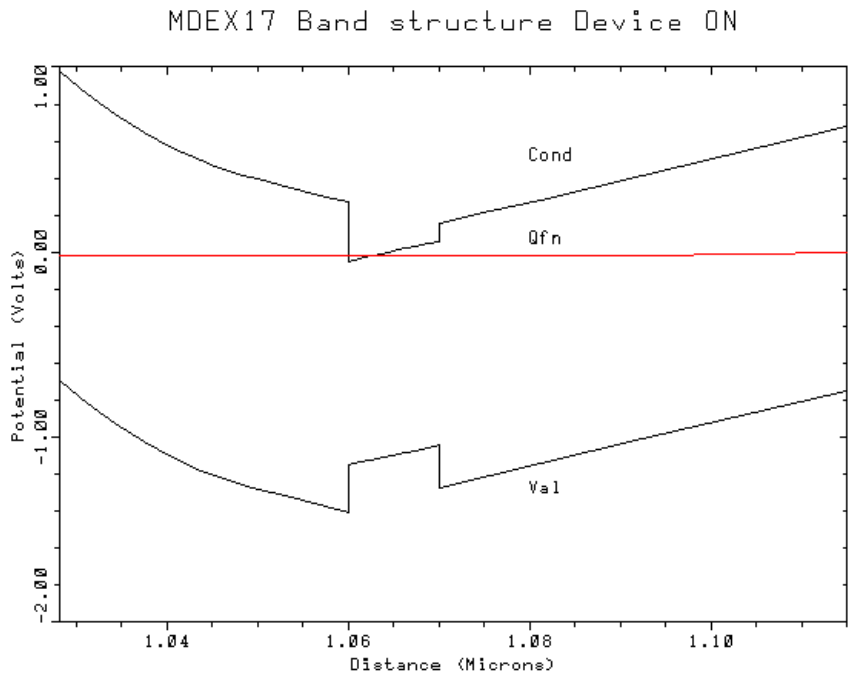


Figure 14-13 Band structure generated by lines 57 through 62 in file *mdex17*, Figure 14-10

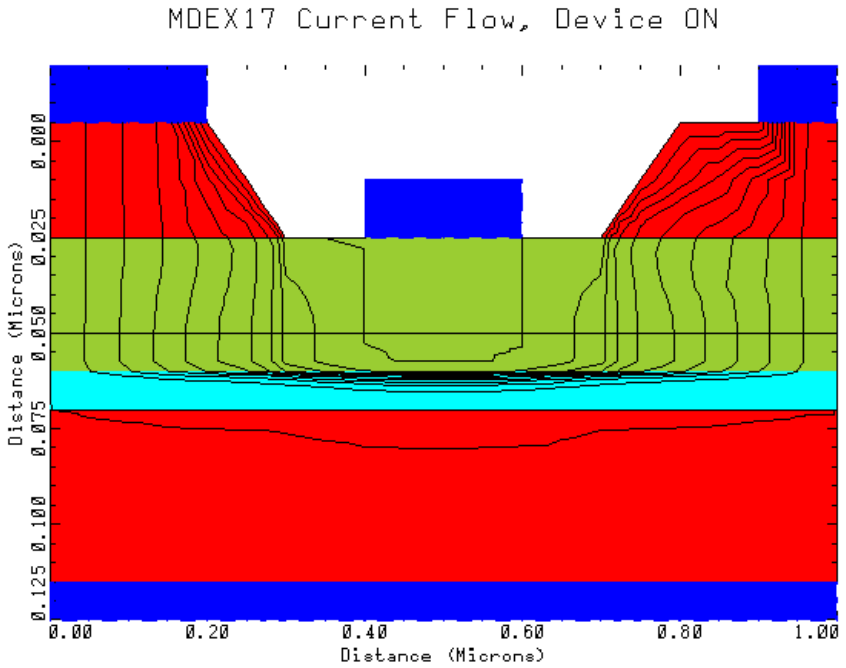


Figure 14-14 Current flow generated by lines 64 through 66 in file *mdex17*, Figure 14-10

Line 69 continues to reduce the gate bias in -0.1V steps until the gate voltage is -0.8V. Line 71 plots the gate characteristics (I_d versus V_{gs}) for the device (Figure 14-15). The device cuts off at about -0.6V. The decreasing slope of the gate characteristic at higher gate biases is caused by electrons from the channel being pulled out of the well and flowing along the highly doped upper edge of the spacer, where the mobility is lower

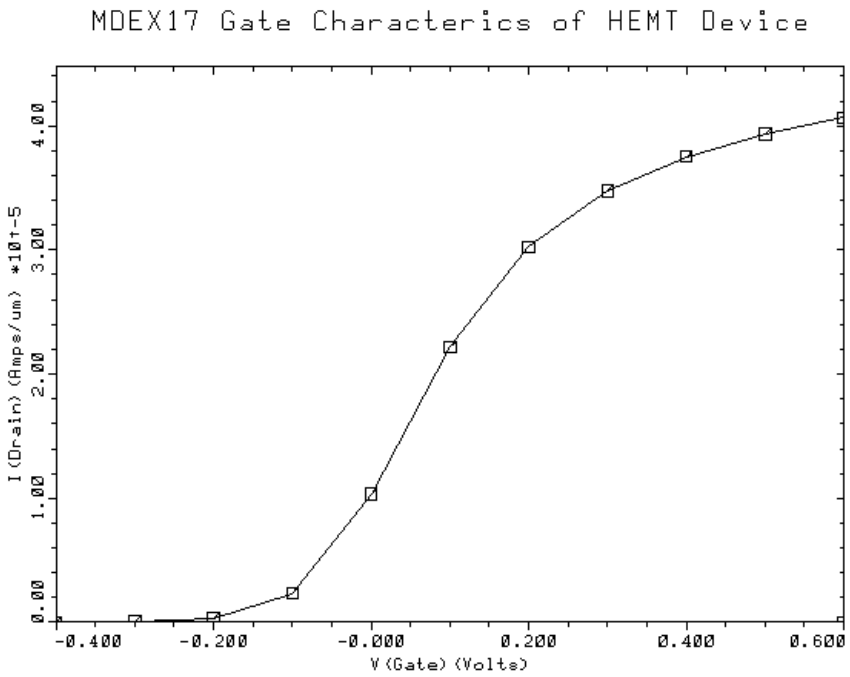


Figure 14-15 Gate characteristics generated by line 71 in file *mdex17*, Figure 14-10

Trapped Charge Examples

Example Specifications

The Trapped Charge Advanced Application Module (TC-AAM) is an extension to Medici which allows simulation of important carrier trapping and de-trapping mechanisms within semiconductor materials. These effects are important in a wide variety of cases such as the simulation of deep level traps, deep donor/acceptor states, or the creation of *lifetime profile*s. This chapter presents examples which illustrate the use of the TC-AAM. Three devices are considered:

- A thin film transistor (TFT)
- A bipolar junction transistor
- A power MOSFET

Thin Film Transistor Example

TFTs are fabricated by depositing a thin film of semiconductor upon an insulating substrate. The semiconductor film is then patterned using etching techniques into the source, drain and body of the transistor. Since the film is deposited rather than grown as a single crystal, the thin film is of an amorphous or polycrystalline nature with a large number of defects. These defects give rise to a continuous distribution of trap states within the band-gap. Since electrons and holes may become trapped and held within these states, the trap states exert a strong effect on the electrical behavior of the device.

Trap States

The trap states in a TFT may be characterized by a density of states distribution (DOS). The DOS describes the number of trap states at a given energy level. In the

general case, the DOS may be a function of position (X,Y) and may also change as the device ages i.e. be a function of time.

DOS Example

For the present example, consider the relatively simple DOS present in Figure 15-1. The center of the energy gap is indicated by Energy E=0. It is also assumed that all traps with E>0 are electron traps, and all traps with E<0 are hole traps. The peak density of traps occurs at the band edges, and a value of 1e19 traps/cm^3/eV is assumed for electrons and 1e18 for holes. The trap density then decreases exponentially toward the center of the band to values of 5e16 and 1e17 for electrons and holes respectively. The exponential slope, however, is different for the electrons and holes. The slope for electrons is such that the density of traps reaches 1e17/cm^3/eV at an energy of 0.4eV. The slope for holes is such that the density of traps reaches 5e15/cm^3/eV at an energy of 0.25eV.

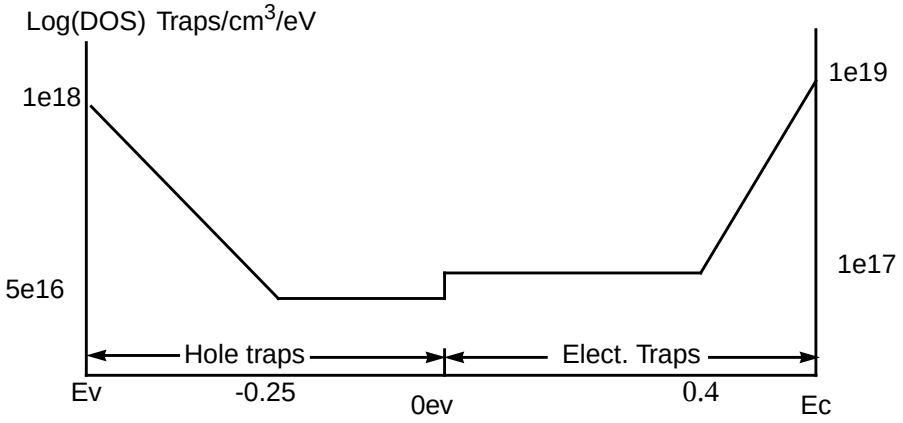


Figure 15-1 Trap density of states function to be modeled

Simulation

This section details the simulation for thin film transistors. The simulation is performed with and without traps.

Device Structure

The input file *mdex18a* for the N-channel TFT example is shown in Figures 15-2 and 15-3. The first 18 lines of the file create the device structure using a simple rectangular mesh with 504 grid points. This particular device has a gate oxide thickness of 300 Angstrom and a silicon film thickness of 1000 Angstrom. The channel length is 2.0 microns. A plot of the device structure is shown in Figure 15-4.

Without Traps

At lines 20-27 a gate sweep is conducted without any traps specified within the device. For this sweep, Vds=0.1V and Vgs is swept from 0 to 3.0V in 0.2 volt

steps. The results of this gate sweep are stored in the file *TFT_NT.IVL* for plotting later on.

```
 1... MESH          OUT.FILE=TFT.MSH
 2... X.MESH        WIDTH=5    N.SPACES=20
 3... Y.MESH        WIDTH=.1   N.SPACES=1
 4... Y.MESH        WIDTH=.03  N.SPACES=2
 5... Y.MESH        WIDTH=.1   N.SPACES=15
 6... Y.MESH        WIDTH=.1   N.SPACES=5

 7... REGION        NUM=1 SILICON
 8... REGION        NUM=2 Y.MIN=0.23 OXIDE
 9... REGION        NUM=3 Y.MAX=0.13 OXIDE

10... COMMENT       1=DRAIN   2=GATE   3=SOURCE    4=SUBST
11... ELECT         NUM=1 IY.MAX=4 X.MIN=4.25
12... ELECT         NUM=2 IY.MAX=2 X.MAX=4 X.MIN=1
13... ELECT         NUM=3 IY.MAX=4 X.MAX=.75
14... ELECT         NUM=4 BOTTOM

15... COMMENT       Specify doping
16... PROFILE       UNIFORM  CONC=1E16  P.TYPE
17... PROFILE       CONC=1E22  X.MAX=1  Y.CHAR=.02  Y.MAX=.21  N.TYPE
18... PROFILE       CONC=1E22  X.MIN=4  Y.CHAR=.02  Y.MAX=.21  N.TYPE
```

Figure 15-2 First part of the simulation input file *mdex18a*

```
19... PLOT.2D       GRID FILL
  ... +             TITLE="MDEX18a  TFT Transistor Structure"

20... MODELS        SRH     ARORA   SRFMOB
21... SYMB          GUMM CARR=0
22... SOLVE         V1=.1 OUT.FILE=TFT.INI

23... COMMENT       Simulation without traps
24... SYMB          NEWT CARR=2
25... LOG           OUT.FILE=TFT_NT.IVL
26... SOLVE         ELEC=2  VSTEP=.2  NSTEP=15
27... LOG           CLOSE


28... COMMENT       Simulation with traps
29... LOAD          IN.FILE=TFT.INI
30... ASSIGN        NAME=EV  N.VAL=-1.06/2
31... ASSIGN        NAME=EC  N.VAL=1.06/2

32... COMMENT       Calculate characteristic length for hole states
33... ASSIGN        NAME=PCHR N.VAL=(-0.25-@EV)*LOG(1E18/5E16) PRINT
34... COMMENT       Gererate hole traps
35... TRAP          DISTR  N.TOT="-(5E16+1E18*EXP(-(@FENER-@EV)/@PCHR))"
  ... +             COND="(@FENER<0)"
  ... +             TAUN="1E-5" TAUP="1E-6"

36... COMMENT       Calculate characteristic length for electron states
37... ASSIGN        NAME=NCHR  N.VAL=(@EC-0.4)*LOG(1E19/1E17) PRINT
38... COMMENT       Generate electron traps
39... TRAP          N.TOT="(1E17+1E19*EXP((@FENER-@EC)/@NCHR))"
  ... +             COND="(@FENER>0)"
  ... +             TAUN="1E-5"  TAUP="1E-6"

40... COMMENT       Simulate with traps
41... SYMB          GUMM CARR=0
42... SOLVE
43... SYMB          NEWT  CARR=2
44... METHOD        N.DAMP
45... SOLVE
46... LOG           OUT.FILE=TFT_TC.IVL
47... SOLVE         ELEC=2  VSTEP=.2  NSTEP=15
48... LOG           CLOSE

49... COMMENT       PLot Results
50... PLOT.1D       Y.AX=I1  X.AX=V2   IN.FILE=TFT_NT.IVL SYMB=1        LOG
  ... +             Title="MDEX18a   Id - vs- Vgs with and without Traps"
  ... +             LEFT=0  BOTTOM=1.E-13
51... PLOT.1D       Y.AX=I1  X.AX=V2   IN.FILE=TFT_TC.IVL SYMB=2 UNCH   LOG
52... LABEL         LABEL="NO TRAPS"    SYMB=1
53... LABEL         LABEL="WITH TRAPS"  SYMB=2
```

Figure 15-3 Second part of the simulation input file *mdex18a*

With Traps

The simulation is now started with traps. At line 29, the initial solution calculated earlier is loaded. To define the trap densities the exponential form is used $NO*\exp(E-EV)/PCHR$. The example predefines some assigned quantities. EC and EV are the energies of the conduction and valence band respectively.

To define the exponential characteristic, the characteristic energy for the exponential is needed, which for holes is given the symbol *PCHR*. Calculate *PCHR* by noting that the hole trap density is 1e18 at E=EV and the hole trap density is 5e16 at E=0.25, therefore:

$$1e18 = NO*\exp((EV - EV)/PCHR) \qquad \text{Equation 15-1}$$

$$5e16 = NO*\exp((-0.25 - EV)/PCHR) \qquad \text{Equation 15-2}$$

Solving for *PCHR* and *NO* gives:

Equation 15-3

$$NO = 1e18 \qquad PCHR = (-0.25 - EV)*\log(1e18/5e16)$$

Creating Traps

The above values are used in line 35 to create the traps. The **DIST** parameter specifies that 20 discrete equally spaced discrete levels are created within the band-gap.

Densities

The trap densities at the 10 points in the lower 1/2 of the band-gap are then assigned by evaluating the equation: **-(5E16+1E18*EXP(-(@FENER-@EV)/@PCHR)**. Note that this expression is always negative since hole traps are designated by giving **N.TOT** a negative value. The value 5e16 was added to make **NTOT** assume a constant value near the center of the band. Note that the energy E has been replaced by the variable **@FENER**, which has never been assigned a value.

Band Gap Energy

FENER represents the band gap energy and its value is assigned automatically by Medici during the trap creation process. **FENER** is one of about 50 pre-assigned variables which can be used in numeric character expressions, others represent position (**@FX**, **@FY**) potential (**@FV**), etc. (the complete list can be found in Chapter 3, EXTRACT on page 3-166). The **MIDGAP** parameter is used to set the energy reference for specifying trap energy levels. When **MIDGAP** is set in conjunction with the **DIST** parameter, as in this example, the distributed trap energies are centered around the middle of the band-gap rather than the intrinsic Fermi level.

The **COND** parameter is used to control when **N.TOT**, **TAUN**, and **TAUP** are evaluated. If the expression for **COND** evaluates to true, then values are assigned. In line 35, **COND** = "@FENER<0". This implies that for this statement (#35), **N.TOT**, **TAUN** and **TAUP** are *only* evaluated for the lower 1/2 of the band, where E<0. Finally **TAUN** and **TAUP** are given values, (constants in this case). Complicated expressions could also have been given for **TAUN** and **TAUP**.

Electron Traps

A similar set of statements (36-39) generates the electron traps in the upper 1/2 of the band. Note that **N.TOT** is always positive in line 39, which indicates electron traps. The **DIST** parameter is not needed in line 39 since the electron traps were

already created by line 35. The **MIDGAP** parameter, however, is needed to specify the midgap as the energy reference for the electron trap levels.

Simulation

The actual simulation with traps is performed at lines 41-48. The IV data is stored in the file *TFT_TC.IVL*. An IV plot of Id vs Vgs is generated by lines 50-53. This is shown in Figure 15-5. It can be seen that when traps are included, the device turns on much more gradually. This is to be expected. All the traps at the surface of the device must be filled with electrons before the conducting channel of the device can form. The electrons in traps contribute to the charge in Poisson's equation. But they are not free to participate in conduction, so the channel must be heavily inverted before appreciable current flows.

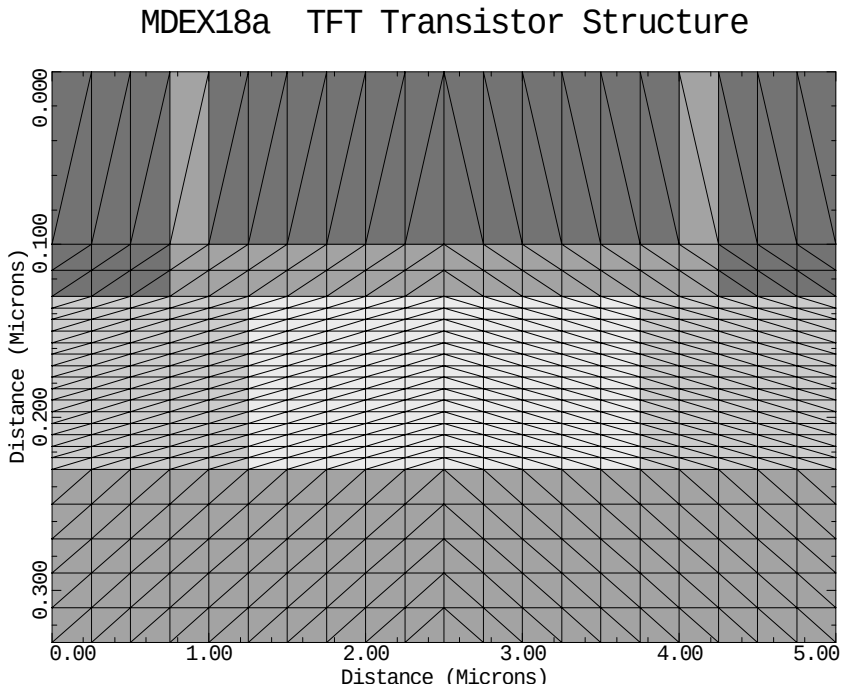


Figure 15-4 Transistor structure at line 19 in file *mdex18a*, Figure 15-3

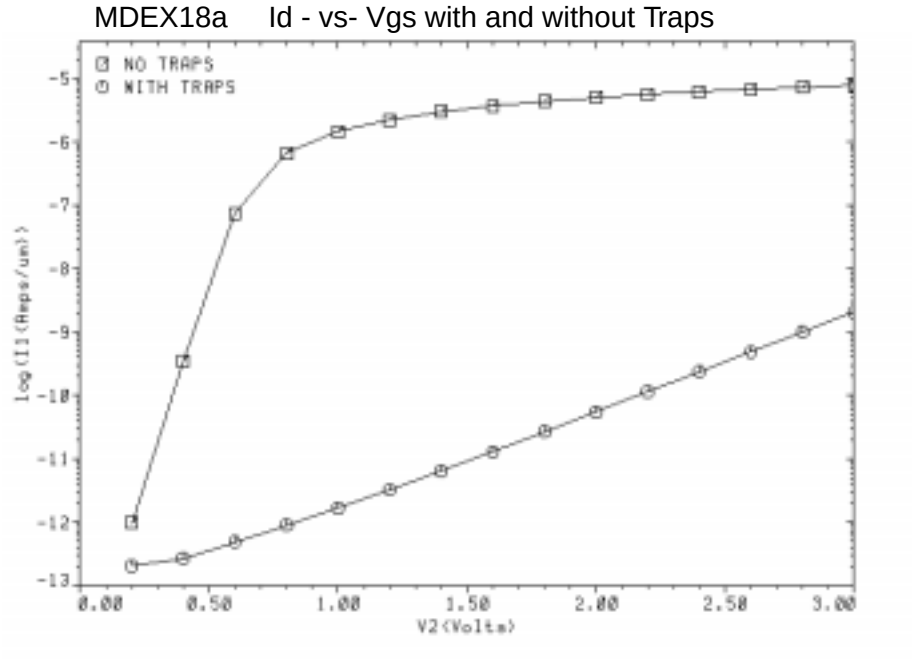


Figure 15-5 Gate characteristics of the TFT device with and without traps at lines 46 through 49 of file *mdex18a*, Figure 15-3

Bipolar Junction Transistor Turn-Off with Traps Simulation

This example investigates how the presence of electron traps effects the switching behavior of a bipolar junction transistor.

Structure Generation

This section details the generation of the device structure to simulate BJT turn-off with traps.

Device Structure

The input file for the example is shown in Figure 15-6. This example uses a quasi one-dimensional transistor structure. The grid is created at lines 2-5. Note that only one vertical column of nodes is used because a one dimensional model is desired.

Electrodes and Doping

The electrodes and doping are generated by lines 6-11. The emitter is at the top edge and the collector at the bottom edge. The base contact extends all the way across the middle of the device at a depth of Y=0.3 microns. Note that the **MAJORITY** type contact is used for the base. This contact is transparent to minority carriers and allows electrons to pass freely from the emitter to the collector.

Initial Solution

Lines 14-18 generate an initial solution with the device turned ON (Vce=8.0 Vbe=0.76). A zero carrier solution (at line 18) is used to start the 2 carrier solution (at line 16). This initial solution is stored in the file *S1* and is used to set the initial conditions for the transient.

Simulation

This section details simulation.

Without Traps

The first transient is performed without traps by line 21. The base voltage is dropped from its initial value (0.76) to zero volts over a period of 1e-11 seconds (the first time step) and the simulation runs for 1e-2 seconds. The results are stored in the file *BJT_NT.*

Time Dependent Traps

The second simulation is performed with time dependent traps, i.e., the *time dependent trap* equations are solved simultaneously with the device equations. The traps are specified by line 25, and single energy level with a trap density of 1e14 traps/cm^3 is used. The trap energy level is explicitly specified as 0.1 volt above the intrinsic Fermi level using the **E1** parameter.

The traps are then specified as time dependent using **TIMEDEP**. The same device is then resimulated, using the **SOLVE** statement of line 28 to reestablish the steady state solution (this time with traps). Finally, the time dependent simulation is performed by line 33.

Fast Traps

A third simulation with *fast traps* is then run for comparison. When fast traps are used, the time dependent trap equation is not solved. Instead the trap occupation function is calculated to be consistent with the carrier concentrations. The trap occupation functions, therefore, adjust instantaneously to the carrier concentrations rather than at their own rate.

Trap Analysis Plot

The results of the analysis are shown in Figure 15-7.

- It can be seen that without any traps, the device turns off the most rapidly.
- When fast traps are included, turn off is delayed because electrons must be removed from the traps in the base and depletion region of the collector before the device can be turned off.
- When time dependent traps are included, the early part of the turn off characteristic looks like the case without any traps, but then a nearly constant *tail* appears.

This tail is created by electrons being slowly excited from the trap centers and participating in conduction. Since the trap level is deep in the band, the rate of thermal excitation is very slow and the tail has a very small magnitude but long

duration. If the trap center were placed closer to the conduction band, the magnitude of the tail would be increased, and the duration reduced.

```
 1... COMMENT     Generate a mesh for a one-dimensional BJT
 2... MESH
 3... X.MESH      WIDTH=1  N.SPACES=1
 4... Y.MESH      DEPTH=2  N.SPACES=49  RATIO=1.1

 5... REGION      SILICON

 6... ELECT       NAME=Collector  BOTTOM
 7... ELECT       NAME=Base       Y.MIN=0.3  Y.MAX=0.32  MAJORITY
 8... ELECT       NAME=Emitter    TOP

 9... PROFILE     N.TYPE  CONC=1E16  UNIF
10... PROFILE     P.TYPE  CONC=1E18  JUNC=0.5
11... PROFILE     N.TYPE  CONC=1E19  JUNC=0.2

12... MODELS      CONSRH   BGN   AUGER

13... COMMENT     Initial Solution
14... SYMB        NEWTON  CARR=0
15... METHOD      TOL.TIM=.1
16... SOLVE       V(Collector)=8.0  V(Base)=0.76
17... SYMB        NEWTON  CARR=2
18... SOLVE       OUT.FILE=S1

19... COMMENT     Solve with no traps
20... LOG         OUT.FILE=BJT_NT
21... SOLVE       V(Base)=0  TSTEP=1E-11  TSTOP=1E-2
22... PLOT.1D     X.AXIS=TIME  Y.AXIS=I(Collector)  Y.LOG  X.LOG
  ... +           TITLE="MDEX18B  BJT Turn Off"  COLOR=1  SYMB=1

23... COMMENT     Solve with time dependent traps
24... LOAD        IN.FILE=S1
25... TRAPS       N.TOTAL="1E14"  E1=0.1  TIME.DEP
26... SOLVE
27... LOG         OUT.FILE=BJT_TT
28... SOLVE       V(Base)=0  TSTEP=1E-11  TSTOP=1E-2
29... PLOT.1D     X.AXIS=TIME  Y.AXIS=I(Collector)  Y.LOG  X.LOG
  ... +           COLOR=2  SYMB=2  UNCHANGE

30... COMMENT     Solve with fast traps
31... LOAD        IN.FILE=S1
32... TRAPS       N.TOTAL="1E14"  ^TIME.DEP  E1=0.1
33... SOLVE
34... LOG         OUT.FILE=BJT_FT
35... SOLVE       V(Base)=0  TSTEP=1E-11  TSTOP=1E-2
36... PLOT.1D     X.AXIS=TIME  Y.AXIS=I(Collector)  Y.LOG  X.LOG
  ... +           COLOR=3  SYMB=3  UNCHANGE

37... LABEL       SYMB=1  LABEL="NO TRAPS"  Y=1E-7  X=1E-6
38... LABEL       SYMB=2  LABEL="TIME DEP. ELECTRON TRAPS"
39... LABEL       SYMB=3  LABEL="FAST ELECTRON TRAPS"
40... LABEL       LABEL="ETRAP=0.1,  NT=1E14"
```

Figure 15-6 Input listing of simulation file *mdex18b*

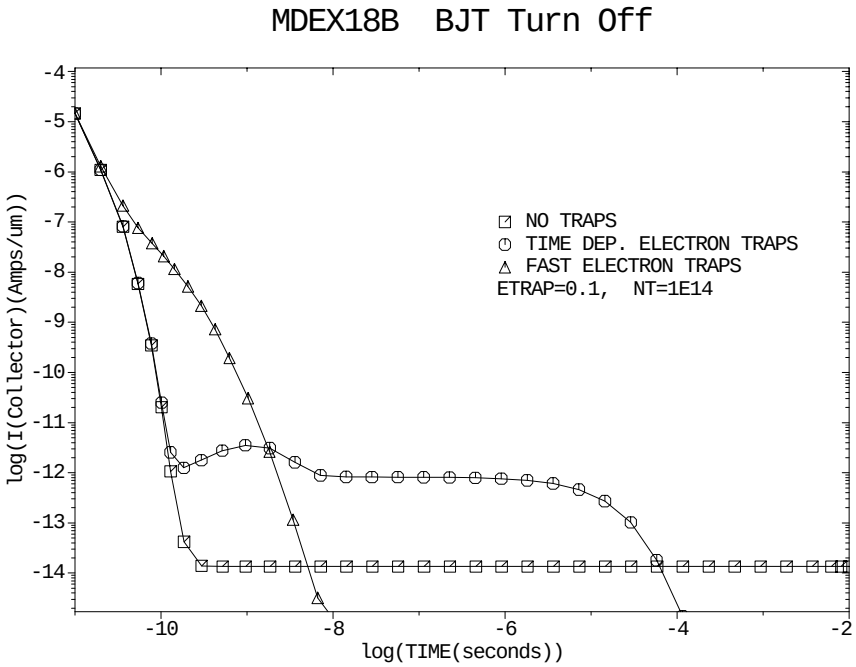


Figure 15-7 Transient turn-off response of BJT with different trap models at lines 22, 29, and 36 through 40 in file *mdex18b*, Figure 15-6

Breakdown Walk-Out in Power MOS Device

This example simulates breakdown walk-out in a power MOS transistor. Breakdown walk-out is a behavior whereby the breakdown voltage of a MOSFET increases with time due to the trapping of carriers within the oxide and at the interface of the device.

This example uses the TC-AAM to simulate this trapping of carriers. A new material called **S.OXIDE** is also used in the simulation. **S.OXIDE** is silicon dioxide treated as a wide band-gap semiconductor. This type of material is used so that the electron and hole continuity equations can then be solved within the oxide as well as the silicon layers. This enables modeling of the transport and trapping of hot electrons injected within the oxide. Since the oxide is modeled as a wide band-gap semiconductor, the Si/SiO2 interface becomes a heterojunction and the HD-AAM must also be authorized to run this example.

Structure Generation and Simulation

This section details the device structure generation and simulation of breakdown walk-out in power MOS device.

```
 1... TITLE     Avant! MEDICI MDEX18C:  Breakdown Walk-out in Power MOS
 2... COMMENT   Create the structure
 3... MESH      OUT.FILE=MD18CMS
 4... X.MESH    WIDTH=5   N.SPACES=5
 5... X.MESH    WIDTH=10  N.SPACES=25
 6... X.MESH    WIDTH=5   N.SPACES=5

 7... Y.MESH    WIDTH=0.5   N.SPACES=3  Y.MIN=-1.02
 8... Y.MESH    WIDTH=0.5   N.SPACES=3
 9... Y.MESH    WIDTH=0.02  N.SPACES=3
10... Y.MESH    WIDTH=10    H1=.05  H2=1.5

11... ELIMIN    COLUMN  Y.MIN=2
12... ELIMIN    COLUMN  Y.MIN=3

13... REGION    SILICON
14... REGION    S.OXIDE  Y.MAX=0  X.MIN=5  X.MAX=15

15... ELECTR    NAME=Drain  Y.MAX=0  X.MIN=16
16... ELECTR    NAME=Drain  TOP  X.MIN=9
17... ELECTR    NAME=Gate   Y.MIN=-.5  Y.MAX=-0.02  X.MIN=5.1  X.MAX=10
18... ELECTR    NAME=Source     Y.MAX=0  X.MAX=4
19... ELECTR    NAME=Substrate  BOTTOM

20... PROFILE   P-TYPE  N.PEAK=5E14  UNIF
21... PROFILE   N-TYPE  N.PEAK=5E19  X.MAX=5  JUNC=1
22... PROFILE   N-TYPE  N.PEAK=5E19  X.MIN=15  JUNC=1
23... PROFILE   N-TYPE  N.PEAK=2.0E16  X.MIN=10  Y.CHAR=0.25  Y.MAX=0.25

24... COMMENT   Plot the structure
25... PLOT.2D   FILL  Title="MDEX18C:  Power MOS Device Structure"
26... FILL      REGION=S.OXIDE  COLOR=5  ^NP.COLOR
27... LABEL     LABEL="S.OXIDE"  X=12  Y=-0.5  C.SIZ=.35
28... LABEL     LABEL="SOURCE"   X=1   Y=0.5   C.SIZ=.35
29... LABEL     LABEL="DRAIN"    X=16  Y=0.5   C.SIZ=.35
30... LABEL     LABEL="DRIFT SPACE"  X=10  Y=0.5  C.SIZ=.35
```

Figure 15-8 First part of the simulation input file *mdex18c*

Device Structure

The Medici input file which generates the structure and performs the simulations is shown in Figures 15-8 and 15-9. The device structure is specified by lines 1-23. Note the **S.OXIDE** material specified for region 2 at line 14.

Electrode

The drain electrode is created using two statements (lines 15 and 16). This is done because it is desired for the drain to extend over the top of the drift space as well as contact the silicon. This electrode is not visible in the device structure plot shown in Figure 15-10.

Grid

The final grid is rather course and has about 900 grid points. Since charge transport within the oxide is being modeled, it is important to have several lines of grid points within the oxide to resolve the current flow. It would also be desirable to have some more fine grid near the edge of the gate on the drain side since this the electric field changes rapidly in this area.

Drain Electrode

The device is a rather large MOS transistor with the exception of the "Drift Space" and the drain electrode which extends over the drift space. The drift space operates like a lightly doped drain (LDD) and extends the depletion region at the

drain side, thereby reducing the peak electric field and increasing the breakdown voltage.

The extension of the drain electrode over the drift space is not normally done in the design of the device, but it is used here because it exaggerates the breakdown walk-out mechanism. Walk-out is exaggerated because the positive drain electrode attracts electrons up into the oxide thereby increasing the hot electron injection efficiency. Without this electrode only traps at the interface would receive hot electrons.

Models

The models used in the simulation are specified at line 31. Note that the **GATE2** gate current model is specified.

Note:

GATE2 is used because in the present release, only the GATE2 model is compatible with S.OXIDE. This restriction will be removed in future versions of the program. The other models are "standard" for a MOS simulation.

Solution

At lines 32 through 35, the initial solution for the transient is generated. First a zero carrier Gummel solution is generated. Since 50V is applied to this device, increase the potential update limits to 10V (parameters **DVLIMIT** and **N.DVLIMIT** on the **METHOD** statement). Since this is a rather large bias step, Newton damping **N.DAMP** is used, and the iteration limit has been increased to 40 **ITLIMIT**. Finally, at line 35, the solution is calculated.

Gate Current

It is important to remember that the gate currents are included explicitly in the model. The gate current calculated at the end of the previous time or bias point is used to supply the hot electron injection for the calculated time point. For this reason **GATE** is specified on the **SOLVE** statement of line 40, since the gate currents calculated at line 40 are used for the first time point calculated at line 40.

This method is perfectly accurate as long as small enough time steps are being taken so that the gate current changes relatively slowly. This last point is always true of simulation on a grid or at discrete time points.

Oxide Traps

At line 39, the oxide traps are specified. Note that the **COND** parameter only evaluates to TRUE inside region number 2 (the **S.OXIDE**) and when NOT at the interface (**FINTER=0**).

Note:

Traps are not specified at the interface due to a restriction in the program.

The effective surface density of traps is $2e12cm^{-2}$ since the density of traps is $2e16cm^{-3}$ and the traps are uniformly distributed throughout the 10^{-4} cm thick

oxide layer. The traps are specified as time dependent and with an energy level of 0eV, (at the intrinsic Fermi level).

Time Dependent Simulation

At line 40 the time dependent simulation is performed. Note that the **GATE** parameter is specified to cause the program to calculate the number of hot electrons which are injected into the **S.OXIDE** layer. Post-processing impact ionization analysis is performed since **IMPACT** is also specified on the **SOLVE** statement. Due to the long time constant of the trap charging process, multiplicative time steps are taken using the **TMULT** parameter.

```
31... MODELS     FLDMOB   PRPMOB   BGN   CONSRH   AUGER   GATE2   ARORA

32... COMMENT    Gerarate an initial solution
33... SYMB       GUMMEL   CARR=0
34... METHOD     DVLIMIT=10   ITLIMIT=40   N.DAMP    N.DVLIM=10   ^TAUTO
35... SOLVE      V(Drain)=50   V(Gate)=2.5
36... SYMB       NEWTON   CARR=2
37... SOLVE      GATE

38... COMMENT    Specify traps in the oxide
39... TRAPS      N.TOTAL=2E16   COND="(@REGION=S.OXIDE)&(@INTERFAC=0)"
  ... +          E1=0   TIME.DEP   TAUN=1E-9   TAUP=1E3

40... SOLVE      OUT.FILE=MD18C00   TSTEP=0.1   TMULT=3.16   GATE   IMPACT
  ... +          NSTEPS=18
```

Figure 15-9 Second part of the simulation input file *mdex18c*

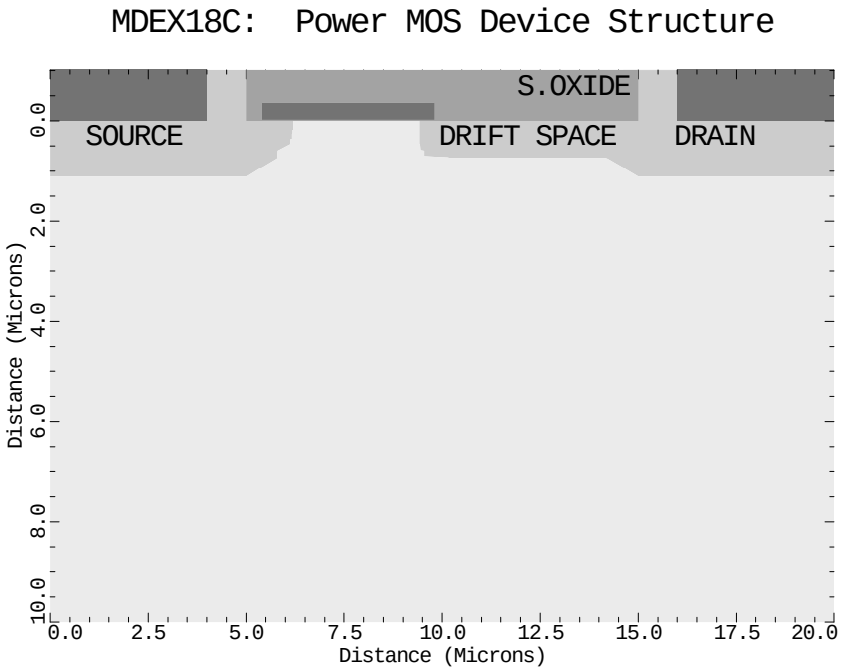


Figure 15-10 Power MOS device structure at lines 26 through 30 of the file *mdex18c*, Figure 15-8

Plots

A series of plots can now be generated using the solution files stored by the preceding example. The input file to generate the plots is shown in Figures 15-11 and 15-12.

Time Dependent: Potential, Electric Field, and Gate Current Plots

Lines 7-15 generate the plot shown in Figure 15-13.

- The top plot shows the potential distribution within the device at time t=1 Sec.

  Note how the equipotentials are spread more or less uniformly across the drift space. This spreading reduces the peak electric field and results in a higher breakdown voltage. During the simulation, electrons which become trapped within the oxide cause the spreading to become even more uniform and as a result the breakdown voltage increases.
- The middle plot in the figure shows the component of electric field parallel to the current (J.E/|J|).

  It can be seen that the peak electric field is right below the edge of the gate. This is the point of peak impact ionization and hot electron injection.
- The bottom plot in Figure 15-13 shows hot electron injection within the device.

  Note that the hot electron injection is highly localized.

Trap Occupation Plots

Lines 16-28 of Figure 15-11 plot trap occupation at three different time points which is shown in Figure 15-14. Observe how traps are filled first closest to the point of gate current injection. You can also see how the gate current fans out as it moves up toward the overlapping drain contact since the pattern of filled traps is wider at the top than at the bottom.

Observe that not all of the electrons are trapped as they move through the oxide. If all the electrons were trapped, then early in the simulation (at time t=1 Sec), only traps near the interface would be filled and not the traps near the top of the oxide.

Impact Ionization Plots

Lines 29-41 of Figure 15-12 generates Figure 15-15 which shows impact ionization during the simulation. It can be seen that the peak ionization rate is decreasing and the size of the peak is also getting smaller.

Figure 15-16 which plots total substrate current also bears out this point. The fact that the ionization rate is getting smaller at this fixed drain bias would indicate that the breakdown voltage is increasing. Unfortunately, to actually run the simu-

lations to generate the breakdown curves for each of these time points would be a very time consuming process.

```
 1... TITLE     MDEX18D:  Plot Results of Break-down Walk-out
 2... MESH      IN.FILE=MD18CMS

 3... ASSIGN    NAME=STRING C.VAL="Y.MAX=1.5 Y.LENG=3 ^LABELS L.ELEC=-1"

 4... COMMENT   Perform some plots at time t=1.
 5... LOAD      IN.FILE=MD18C00
 6... MODELS    GATE2

 7... PLOT.2D   @STRING  Y.OFF=.5 TITLE=""
 8... CONTOUR   LOG   G.IN   ABS  FILL
 9... PLOT.2D   ^CLEAR  BOUND  @STRING  Y.OFF=.5
  ... +          TITLE="Gate Currrent Genertation T=1 Sec"

10... PLOT.2D   @STRING  ^CLEAR  Y.OFF=5.5 TITLE=""
11... CONTOUR   J.EFIE  FILL
12... PLOT.2D   ^CLEAR  BOUND   @STRING  Y.OFF=5.5
  ... +         TITLE="J.E/|J|  T=1 Sec"

13... PLOT.2D   @STRING  ^CLEAR  Y.OFF=10.5 TITLE=""
14... CONTOUR   POTENTIAL  FILL
15... PLOT.2D   ^CLEAR  BOUND   @STRING  Y.OFF=10.5
  ... +         TITLE="MDEX18D:  Potential T=1 Sec"

16... COMMENT   Plot trap occupation at various times.
17... LOAD      IN.FILE=MD18C00
18... PLOT.2D   @STRING  Y.OFF=0.5  TITLE=""
19... CONTOUR   LOG  TRAP.OC  MIN=10  MAX=16.3  FILL
20... PLOT.2D   ^CLEAR  BOUND  @STRING  Y.OFF=.5
  ... +         TITLE="Trap Occupation T=1 Sec"

21... LOAD      IN.FILE=MD18C10
22... PLOT.2D   @STRING  Y.OFF=5.5  ^CLEAR TITLE=""
23... CONTOUR   LOG  TRAP.OC  MIN=10  MAX=16.3  FILL
24... PLOT.2D   ^CLEAR  BOUND  @STRING  Y.OFF=5.5
  ... +         TITLE="Trap Occupation T=4 Hrs"

25... LOAD      IN.FILE=MD18C17
26... PLOT.2D   @STRING  Y.OFF=10.5  ^CLEAR  TITLE=""
27... CONTOUR   LOG  TRAP.OC  MIN=10  MAX=16.3  FILL
28... PLOT.2D   ^CLEAR  BOUND  @STRING  Y.OFF=10.5
  ... +         TITLE="MDEX18D:  Trap Occupation T=1.5 Years"
```

Figure 15-11 First part of the simulation input file *mdex18d*

```
29... Comment   Plot impact generation at various times
30... LOAD      IN.FILE=MD18C00
31... PLOT.2D   @STRING  Y.OFF=0.5  TITLE=""
32... CONTOUR   II.GEN  LOG MIN=20  MAX=25  FILL
33... PLOT.2D   ^CLEAR  BOUND  @STRING  Y.OFF=.5
  ... +         TITLE="Generation (ISUB=1.26E-6) T=1 Sec"

34... LOAD      INF=MD18C10
35... PLOT.2D   @STRING Y.OFF=5.5 ^CLEAR TITLE=""
36... CONTOUR   II.GEN  LOG  MIN=20  MAX=25  FILL
37... PLOT.2D   ^CLEAR BOUND  @STRING Y.OFF=5.5
  ... +         TITLE="Generation (ISUB=4.9e-7) T=4 Hrs"

38... LOAD      INF=MD18C17
39... PLOT.2D   @STRING Y.OFF=10.5 ^CLEAR TITLE=""
40... CONTOUR   II.GEN  LOG  MIN=20  MAX=25  FILL
41... PLOT.2D   ^CLEAR BOUND  @STRING Y.OFF=10.5
  ... +         TITLE="MDEX18D: Generation (ISUB=1.9E-7) T=1.5 Yrs"

42... PLOT.1D   X.AX=TIME  Y.AX=II  IN.FILE=mdex18c.ivl  POINTS  X.LOG
  ... +         TITLE="MEDX18D: Substrate Current During Walk-Out'
```

Figure 15-12 Second part of the simulation input file *mdex18d*

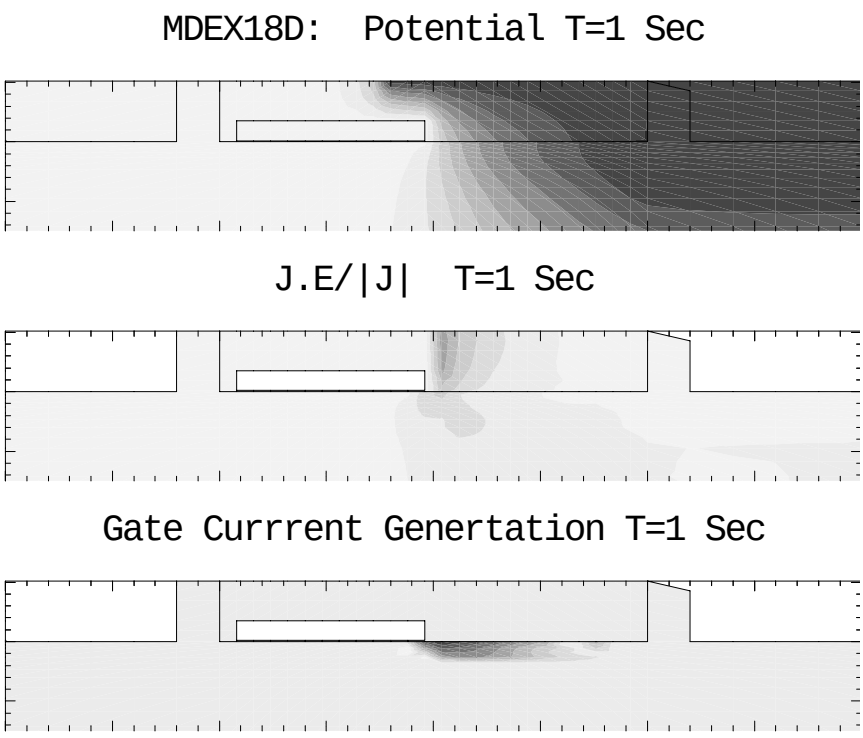


Figure 15-13 Time dependent potential, electric field, and gate current at lines 7 through 15 in file *mdex18d*, Figure 15-11

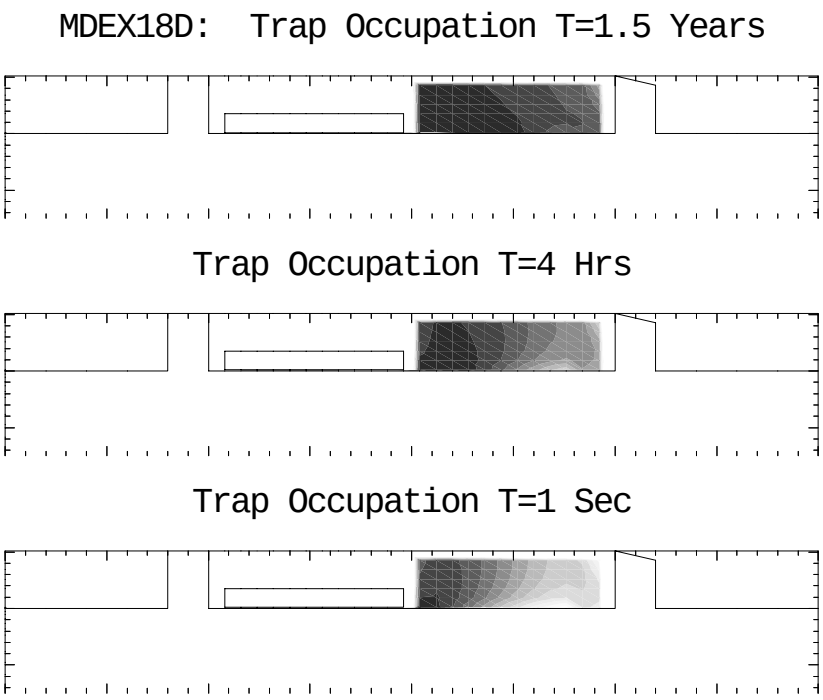


Figure 15-14 Trap occupation at lines 16 through 28 in file *mdex18d*, Figure 15-11

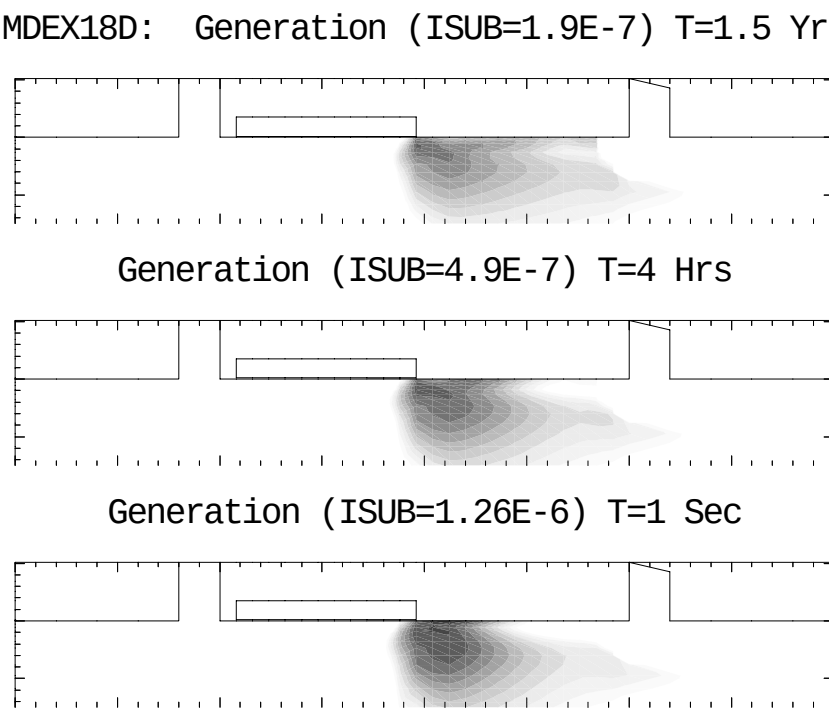


Figure 15-15 Impact Ionization at lines 29 through 41 in file *mdex18d,* Figure 15-12

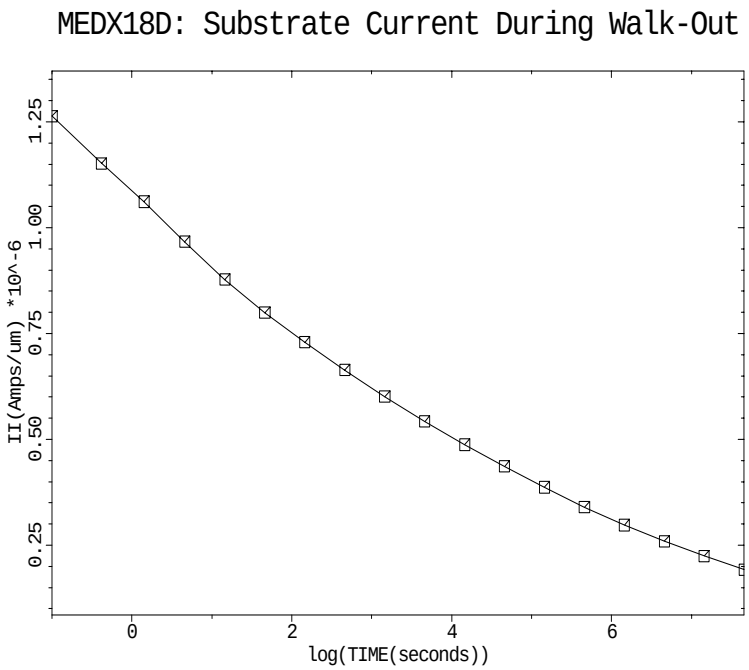


Figure 15-16 Substrate current during walk-out at line 42 in file *mdex18d*, Figure 15-12

Optical Device Examples

Example Specifications

The Medici Optical Device Advanced Application Module (OD-AAM) is used to model propagation of light inside and outside a device. The OD-AAM also allows calculation of the spectral response of the optical devices in steady-state or transient modes. This chapter presents two examples:

- Simulation of an A-Silicon solar cell and how the DC characteristics change according to the wavelength of the incident light
- Simulation of a GaAs photoconductive detector

A-Silicon Solar Cell Example

Amorphous silicon p-i-n solar cells are one of the most promising technologies for both consumer and power photovoltaic applications. This example examines the spectral response and the power conversion efficiency of such a cell. There are many important parameters for solar cell design such as:

- Doping profile
- Structure of illumination window
- Gap-state distribution
- Back surface electrode

This example uses the oxide thickness of the illumination window as a variable to investigate its effect on the cell operation.

Structure Generation

This section describes the generation of the device structure (input file, mesh, etc.).

Input file *mdex19a* is shown in Figures 16-1 and 16-2. The first step in creating the structure is the mesh generation in the lines 9 to 14. The oxide thickness is controlled by the loop parameter L1.

The doping profile is specified by lines 21 to 23. The cell has a typical p+ - i - n+ structure.

Line 25 sets the top ITO electrode as an infinitely thin transparent film. Its refractive index will be set on the **MATERIAL** statement. The bottom a-silicon/Aluminum contact reflectance is set to 0.9 in line 26.

Refractive Index

Line 30 instructs the program to set the refractive index of the ITO film to 2.1 over the [0.2,1.0] micron wavelength band rather than using the default values of the silicon dioxide. The absorption coefficients of a-silicon in this simulation is calculated from the imaginary terms of the refractive index which is read from the default table. The absorption coefficient of a-silicon in the visible region decreases dramatically as the wavelength approaches the ultra-violet region.

Description of Light Source

The description of the light source and the ray-tracing parameters are given on the **PHOTOGEN** statement in line 33. **RAYTRACE** should be specified whenever the OD-AAM is used.

Solar Light-Spectra

The solar light spectrum is approximated by the black-body radiation spectrum (enabled via the **BB.RAD** flag) at 5800 K (**BB.TEMP**=5800) which is nearly identical to Air Mass Zero (AM0) spectrum as shown in Figure 16-3. The total radiation energy is approximately 100m W over the [0.2 μm, 1.0 μm] wavelength-range.

The number of sampled wavelengths, WL is set to 25 in line 2. Each wavelength has its monochrome component of the light intensity calculated internally by Medici. The beam originates at x=0.05 μm and y=-1.5 μm and the incident angle is 90 degrees. Each ray of the beam is traced until its power is attenuated below, **INT.RATIO** *(The initial light power) of the ray, at which point the ray is terminated.

The width of ray is 0.1 μm so that the whole device is illuminated. The ray is not initially split since the simulation device has a simple one-dimensional structure.

One of the major capabilities of OD-AAM is the treatment of the stack of multiple layers as a single film to consider interference effects. You can evaluate the energy of the electromagnetic wave which enters the device and its TM and TE polariza-

tion. In this example, the transmittance and reflectance of the Air/ITO/a-Si structure is modeled.

Solutions for the Spectral Response and I/V Characteristics

This section describes the various solutions and save methods for the examples.

The zero-bias (or short-circuit) solutions for the sampled wavelengths are performed by lines 37 to 39 and the result is saved in the log files, *MDEX19.SPL1*, *MDEX19.SPL2*, and *MDEX19.SPL3*.

The **WAVE** parameter specifies the number of sampled wavelengths. Each solution only accounts for the carriers generated by the current wavelength component. In this example, the **WAVE** loops from 1 to 25 which is the number of sampled wavelengths in the **PHOTOGEN** statement. The log files contain the following:

- Wavelengths
- Transmittance
- Reflectance
- Terminal currents
- Number of generated carriers
- Incident flux

Lines 41 to 44 perform steady-state solutions to calculate the total collected current, the open-circuit voltage and the cell power. The bias step is reduced as the voltage approaches the open-circuit voltage Voc.

Simulation Results and Plots

This section presents the simulation results and the generated graphical plots.

Plots from the simulation clarifying the optical effects are generated by the second input file, *mdex19b*, which is shown in Figures 16-6 and 16-7. Line 3 creates a plot of the simulation mesh of the cell with ITO thickness of 0.06 μm (Figure 16-8).

Figures 16-4 and 16-5 show the simulation results for an ITO thickness of 0.06 μm. The transmittance plot is shown in Figure 16-9 which is the result of the lines 5 through 11 in the input file *mdex19b.*

Short Circuit Current and Spectral Power Density

Figure 16-10 created in lines 13 to 19 shows the short-circuit current through electrode num=1. The cell with a 0.06 μm thick ITO window reaches a peak of generated current for a wavelength of approximately 0.6 μm, very near the λ=0.51 μm maximum spectral power density of the source (see Figure 16-3). Therefore, this cell can absorb more photons than the other two whose transmittance peaks are far from the solar power density peak.

External Collection Efficiency Plot

Lines 21 to 27 plot the external collection efficiency of the three cells. The result is shown in Figure 16-11.

The external collection efficiency is defined as the total photocurrent density $J(\lambda)$ at that wavelength divided by the number of photons incident on the surface of the cell times the electronic charge q(=1.6e-19 C).

This is different from **QNTM.EFF**, the quantum efficiency which is the electron-hole-pair generation rate per photon inside the device and may be defined by the user. The default value used in this example is 1.0.

In this figure, the cell with 0.09 μm ITO exhibits a spike near 0.28 μm reflecting the peak of the transmittance in the Figure 16-9. Both the short-circuit current and the external collection efficiency of the cell with 0.03 μm ITO have much lower values over the wavelength range than the other cells. This is due to the poor transmittance beyond λ=0.4 μm. The 0.06 μm ITO cell shows the greatest efficiency over the 0.4 to 0.7 μm range where most of the solar energy is concentrated.

I-V Plot

In Figure 16-12, the I-V characteristics of the three cells are plotted. The photocurrents are approximately:

- -1.99e-11 Amps for the 0.03 μm "ITO thickness"
- -2.43e-11 Amps for the 0.06 μm "ITO thickness"
- -2.13e-11 Amps for the 0.09 μm "ITO thickness"

Since the device is 0.1 μm wide and 1 μm deep in the z direction, the absolute current densities are 19.9 mA/cm^2, 24.3 mA/cm^2 and 21.3 mA/cm^2, respectively. The open-circuit voltage Voc is found to be approximately 0.94 V for all three cells.

Power and Load Resistance Plot

Lines 37 and 38 define the power and the load resistance quantities for a new plot. The power P is the product of the cell voltage and the cell current and the load resistance L is the cell voltage divided by the cell current.

In Figure 16-13, the ITO peak powers are approximately:

- 15.2 mW/cm^2 for the 0.03 μm "ITO thickness"
- 19.0 mW/cm^2 for the 0.06 μm "ITO thickness"
- 16.7 mW/cm^2 for the 0.09 μm "ITO thickness"

The maximum power is generated by the 0.06 μm ITO cell. The Fill Factor of the 0.06 μm ITO cell is:

$$FF = \frac{P_{max}}{I_{SC}V_{OC}} = \frac{19.0\ mW/cm^2}{24.3\ mA/cm^2 \cdot 0.94V} = 0.83 \qquad \text{Equation 16-1}$$

The power conversion efficiency for 0.06 μm cell is calculated in Equation 16-2. The total incident power is 100.4 mWatts/cm^2, and therefore,

$$Efficiency = \frac{19.0\ mW/cm^2}{100.4\ mW/cm^2} \cong 0.19 \qquad \text{Equation 16-2}$$

```
 1... TITLE  Example 19 a-Silicon Solar Cell Example

 2... ASSIGN     NAME=WL N.VALUE=25
 3... ASSIGN     NAME=EG N.VAL=1.6

 4... COMMENT    loop with 3 different oxide thicknesses
 5... LOOP       STEPS=3
 6...   ASSIGN   NAME=L1  N.VAL=(1,2,3)
 7...   ASSIGN   NAME=TOX N.VAL=(0.03,0.06,0.09)

 8...   COMMENT  mesh generation
 9...   MESH     OUT.FILE="SSMSH"@L1""
10...   X.MESH   WIDTH=0.1 H1=0.1
11...   Y.MESH   Y.MIN=-@TOX Y.MAX=0.0 H1=@TOX/2
12...   Y.MESH   Y.MIN=0.0 Y.MAX=0.02 H1=0.002
13...   Y.MESH   Y.MIN=0.02 Y.MAX=0.45 H1=0.002 H2=0.005 H3=0.04
14...   Y.MESH   Y.MIN=0.45 Y.MAX=0.50 H1=0.005

15...   REGION   NUM=1 Y.MAX=0     INSULATOR
16...   REGION   NUM=2 Y.MIN=0     A-SILICON

17...   COMMENT  Electrodes:  #1=p+ Collector, #2=Substrate
18...   ELECTR   NUM=1 Y.MIN=0 Y.MAX=0
19...   ELECTR   NUM=2 BOTTOM

20...   COMMENT  Specify Doping ( p+nn+ structure )
21...   PROFILE  N-TYPE Y.MIN=0.00 UNIFORM N.PEAK=1e11
22...   PROFILE  P-TYPE Y.MIN=0.00 Y.JUN=0.01 N.PEAK=1E18
23...   PROFILE  N-TYPE Y.MIN=0.50 Y.JUN=0.47  N.PEAK=5E18

24...   COMMENT  Specify Electrode Characteristics
25...   CONTACT  NUM=1  TRANSE
26...   CONTACT  NUM=2  REFLECT=0.9

27...   COMMENT  Specify Optical Parameters
28...   COMMENT  The cell is assumed to be coated with indium tin oxide
29...   MATERIAL A-SILICON  EG300=@EG
30...   MATERIAL REGION=1 PR.TAB WAVE.RE=(0.2,1.0) INDEX.RE=(2.10,2.10)
  ...   +        FIRST LAST
```

Figure 16-1 First part of the simulation input file *mdex19a*

```
31...   MODELS   CONSRH AUGER

32...   COMMENT  Specify light source and photogeneration with raytracing
  ...   +        The light spectrum is black body radiation (T = 5800K)
  ...   +        which is approximated to the AM0 spectrum.
  ...   +        Total incident light power is approximately 100mW/cm^2
  ...   +        for [0.2 um, 1.0 um] wavelength band.

33...   PHOTOGEN RAYTRACE BB.RADI BB.TEMP=5800 WAVE.ST=0.2 WAVE.EN=1.0
  ...   +        WAVE.NUM=@WL
  ...   +        X.ORG=0.05  Y.ORG=-1.5  ANGLE=90  INT.RATIO=1E-2
  ...   +        N.INTEG=10  RAY.N=1  RAY.W=0.1  FILM.REG=(1)

34...   SYMBOL   NEWTON CARRIERS=2
35...   COMMENT  Solve for each wavelength of the spectral response

36...   LOG      OUT.FILE="MDEX19.SPL"@L1""

37...   LOOP  STEPS=@WL
38...            SOLVE WAVE=1:1
39...   L.END

40...   COMMENT  DC characteristics

41...   LOG      OUT.FILE="MDEX19.IVL"@L1""
42...   SOLVE    V1=0.0 VSTEP=0.1  NSTEP=7 ELEC=1
43...   SOLVE    V1=0.8 VSTEP=0.03 NSTEP=3 ELEC=1
44...   SOLVE    V1=0.9 VSTEP=0.01 NSTEP=6 ELEC=1

45... L.END

46... END
```

Figure 16-2 Second part of the simulation input file *mdex19a*

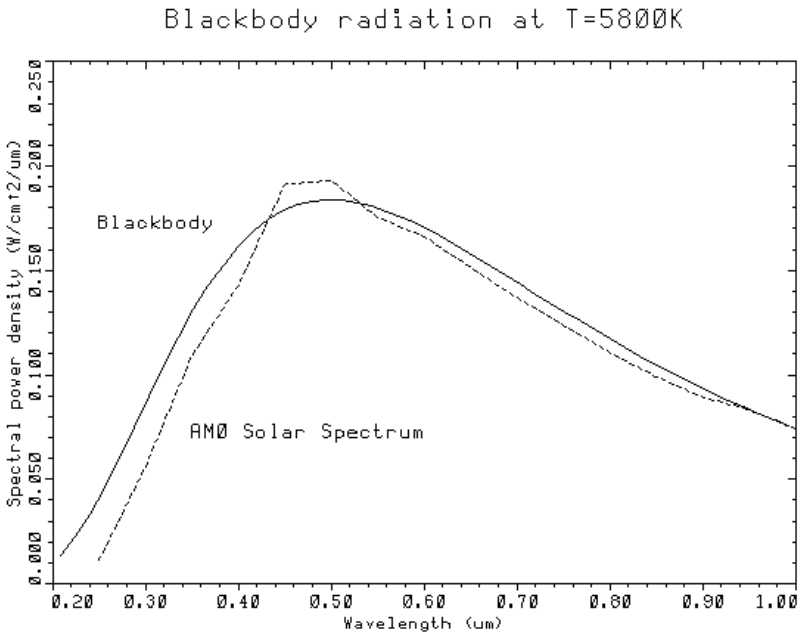


Figure 16-3 Black-body radiation at 5800K and AM0 solar spectrum

Transmittance and Reflectance at Illumination Surface

Number	Wavelength	Transmittance	Reflectance
1	2.160E-01 um	5.955E-01	4.045E-01
2	2.480E-01 um	3.704E-01	6.296E-01
3	2.800E-01 um	3.459E-01	6.541E-01
4	3.120E-01 um	3.900E-01	6.100E-01
5	3.440E-01 um	4.934E-01	5.066E-01
6	3.760E-01 um	6.180E-01	3.820E-01
7	4.080E-01 um	7.551E-01	2.449E-01
8	4.400E-01 um	8.620E-01	1.380E-01
9	4.720E-01 um	9.412E-01	5.882E-02
10	5.040E-01 um	9.833E-01	1.666E-02
11	5.360E-01 um	9.926E-01	7.411E-03
12	5.680E-01 um	9.852E-01	1.477E-02
13	6.000E-01 um	9.662E-01	3.384E-02
14	6.320E-01 um	9.426E-01	5.742E-02
15	6.640E-01 um	9.231E-01	7.692E-02
16	6.960E-01 um	9.048E-01	9.522E-02
17	7.280E-01 um	8.882E-01	1.118E-01
18	7.600E-01 um	8.736E-01	1.264E-01
19	7.920E-01 um	8.609E-01	1.391E-01
20	8.240E-01 um	8.498E-01	1.502E-01
21	8.560E-01 um	8.395E-01	1.605E-01
22	8.880E-01 um	8.307E-01	1.693E-01
23	9.200E-01 um	8.232E-01	1.768E-01
24	9.520E-01 um	8.168E-01	1.832E-01
25	9.840E-01 um	8.093E-01	1.907E-01

Figure 16-4 Transmittance and reflectance from **PHOTOGEN** and **FILE.REG** at line 33 in file *mdex19a.inp*, Figures 16-1 and 16-2

```
 Generation rate and Incident flux per each wavelength

-----------------------------------------------------------------------
 Number    Wavelength      Generated carriers       Incident flux
-----------------------------------------------------------------------
   1      2.160E-01 um      5.919E-14 C/um/s       9.939E-14 C/um/s
   2      2.480E-01 um      9.325E-14 C/um/s       2.517E-13 C/um/s
   3      2.800E-01 um      1.681E-13 C/um/s       4.860E-13 C/um/s
   4      3.120E-01 um      3.051E-13 C/um/s       7.823E-13 C/um/s
   5      3.440E-01 um      5.474E-13 C/um/s       1.109E-12 C/um/s
   6      3.760E-01 um      8.880E-13 C/um/s       1.437E-12 C/um/s
   7      4.080E-01 um      1.314E-12 C/um/s       1.740E-12 C/um/s
   8      4.400E-01 um      1.728E-12 C/um/s       2.005E-12 C/um/s
   9      4.720E-01 um      2.092E-12 C/um/s       2.222E-12 C/um/s
  10      5.040E-01 um      2.352E-12 C/um/s       2.392E-12 C/um/s
  11      5.360E-01 um      2.496E-12 C/um/s       2.515E-12 C/um/s
  12      5.680E-01 um      2.554E-12 C/um/s       2.596E-12 C/um/s
  13      6.000E-01 um      2.545E-12 C/um/s       2.640E-12 C/um/s
  14      6.320E-01 um      2.435E-12 C/um/s       2.654E-12 C/um/s
  15      6.640E-01 um      2.331E-12 C/um/s       2.643E-12 C/um/s
  16      6.960E-01 um      2.183E-12 C/um/s       2.612E-12 C/um/s
  17      7.280E-01 um      1.992E-12 C/um/s       2.565E-12 C/um/s
  18      7.600E-01 um      1.697E-12 C/um/s       2.506E-12 C/um/s
  19      7.920E-01 um      1.253E-12 C/um/s       2.438E-12 C/um/s
  20      8.240E-01 um      5.992E-13 C/um/s       2.364E-12 C/um/s
  21      8.560E-01 um      5.117E-13 C/um/s       2.286E-12 C/um/s
  22      8.880E-01 um      4.294E-13 C/um/s       2.205E-12 C/um/s
  23      9.200E-01 um      3.486E-13 C/um/s       2.123E-12 C/um/s
  24      9.520E-01 um      2.675E-13 C/um/s       2.041E-12 C/um/s
  25      9.840E-01 um      2.370E-13 C/um/s       1.961E-12 C/um/s
-----------------------------------------------------------------------
    Total power of incident ray =  1.004E+02  mWatts/cm^2
-----------------------------------------------------------------------
```

Figure 16-5 Generation rate inside the cell from **PHOTOGEN** and **FILE.REG** at line 33 in file *mdex19a.inp*, Figures 16-1 and 16-2

```
 1... TITLE     MDEX19B:  Plot results of a-silicon solar cell

 2... MESH      IN.FILE=SSMSH2
 3... PLOT.2D   TITLE="Tox=0.06 Microns"  GRID  FILL  BOUND  REGION
  ... +         SCALE  X.OFF=7

 4... COMMENT   Transmittance of the ITO
 5... LOOPS     STEPS=3
 6...   ASSIGN    NAME=TOX    N.VALUE=(0.03,0.06,0.09)
 7...   ASSIGN    NAME=L1     N.VALUE=(1,2,3)
 8...   ASSIGN    NAME=CLEAR  C1=CLEAR  C2=^CLEAR  C3=^CLEAR
 9...   PLOT.1D   IN.FILE="MDEX19.SPL"@L1  X.AXIS=WA  Y.AXIS=TR
  ...   +         COLOR=@L1  SYMB=@L1  LEFT=0.2  RIGHT=1.0
  ...   +         BOT=0.2  TOP=1  @CLEAR
  ...   +         TITLE="MDEX19B Transmittance -vs- Wavelength"
10...   LABEL     LABEL="Tox="@TOX" Microns"  COLOR=@L1  SYMB=@L1
  ...   +         X=0.6  Y=0.3+@L1/25  C.SIZE=0.3
11... L.END

12... COMMENT   Short-Circuit Current
13... LOOPS     STEPS=3
14...   ASSIGN    NAME=TOX    N.VALUE=(0.03,0.06,0.09)
15...   ASSIGN    NAME=L1     N.VALUE=(1,2,3)
16...   ASSIGN    NAME=CLEAR  C1=CLEAR  C2=^CLEAR  C3=^CLEAR
17...   PLOT.1D   IN.FILE="MDEX19.SPL"@L1  X.AXIS=WA  Y.AXIS=I(1)  ABS
  ...   +         COLOR=@L1  SYMB=@L1  @CLEAR  BOT=0  TOP=0.3E-11
  ...   +         TITLE="MDEX19B Short Circuit Current -vs- Wavelength"
18...   LABEL     LABEL="Tox="@TOX" Microns"  COLOR=@L1  SYMB=@L1
  ...   +         X=0.25  Y=2.0E-12+@L1*1.5E-13
19... L.END

20... COMMENT   External Collection Efficiency
21... LOOPS     STEPS=3
22...   ASSIGN    NAME=TOX    N.VALUE=(0.03,0.06,0.09)
```

Figure 16-6 First part of the simulation input file *mdex19b*

```
28... COMMENT   IV Characteristics
29... LOOPS     STEPS=3
30...   ASSIGN    NAME=TOX    N.VALUE=(0.03,0.06,0.09)
31...   ASSIGN    NAME=L1     N.VALUE=(1,2,3)
32...   ASSIGN    NAME=CLEAR  C1=CLEAR  C2=^CLEAR  C3=^CLEAR
33...   PLOT.1D   IN.FILE="MDEX19.IVL"@L1  X.AXIS=V(1)  Y.AXIS=I(1)
  ...   +         COLOR=@L1  SYMB=@L1  @CLEAR
  ...   +         RIGHT=1.0  BOT=-0.3E-10  TOP=0
  ...   +         TITLE="MDEX19B Cell IV Characteristic"
34...   LABEL     LABEL="Tox="@TOX" Microns"  COLOR=@L1  SYMB=@L1
  ...   +         X=0.2  Y=-0.3E-11-@L1*1.5E-12  C.SIZE=0.3
35... L.END

36... COMMENT Power/Load Characteristics
37... EXTRACT   NAME=Power  EXP="-1e9*@I(1)*@V(1)"   UNITS=Watts/cm^2
38... EXTRACT   NAME=Load   EXP="-1e-9*@V(1)/@I(1)"  UNITS=Ohms-cm^2

39... LOOPS     STEPS=3
40...   ASSIGN    NAME=TOX    N.VALUE=(0.03,0.06,0.09)
41...   ASSIGN    NAME=L1     N.VALUE=(1,2,3)
42...   ASSIGN    NAME=CLEAR  C1=CLEAR  C2=^CLEAR  C3=^CLEAR
43...   PLOT.1D   IN.FILE="MDEX19.IVL"@L1  X.AXIS=LOAD  Y.AXIS=POWER
  ...   +         COLOR=@L1  SYMB=@L1  @CLEAR
```

Figure 16-7 Second part of the simulation input file *mdex19b*

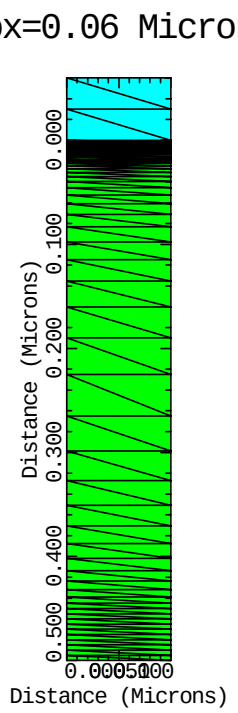


Figure 16-8 Mesh from **PLOT.2D** at line 3 in file *mdex19b.inp*, Figures 16-6 and 16-7

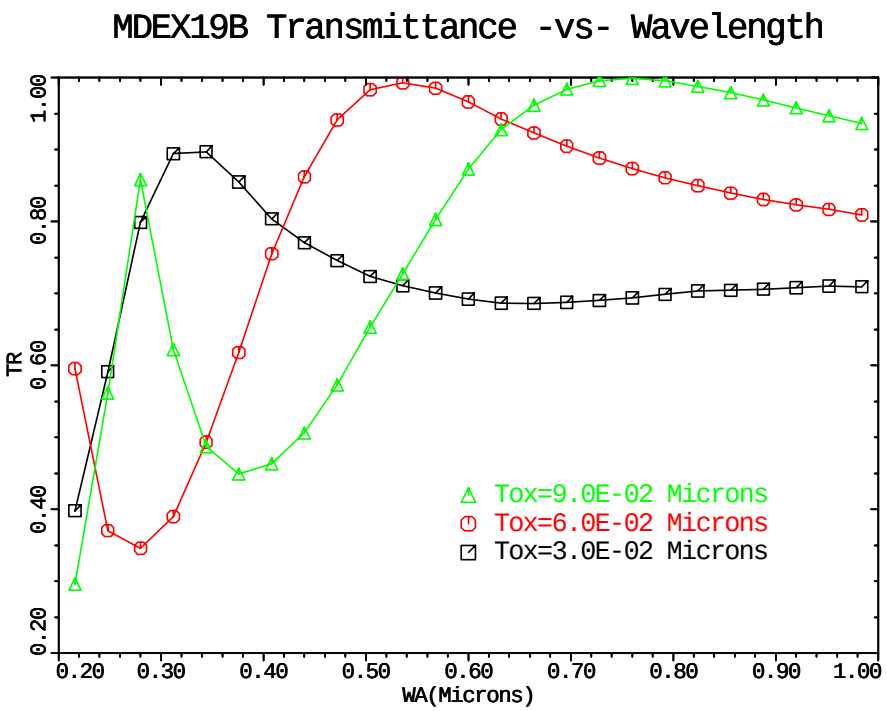


Figure 16-9 Transmittance from lines 5 - 11 in file *mdex19b.inp*, Figure 16-6

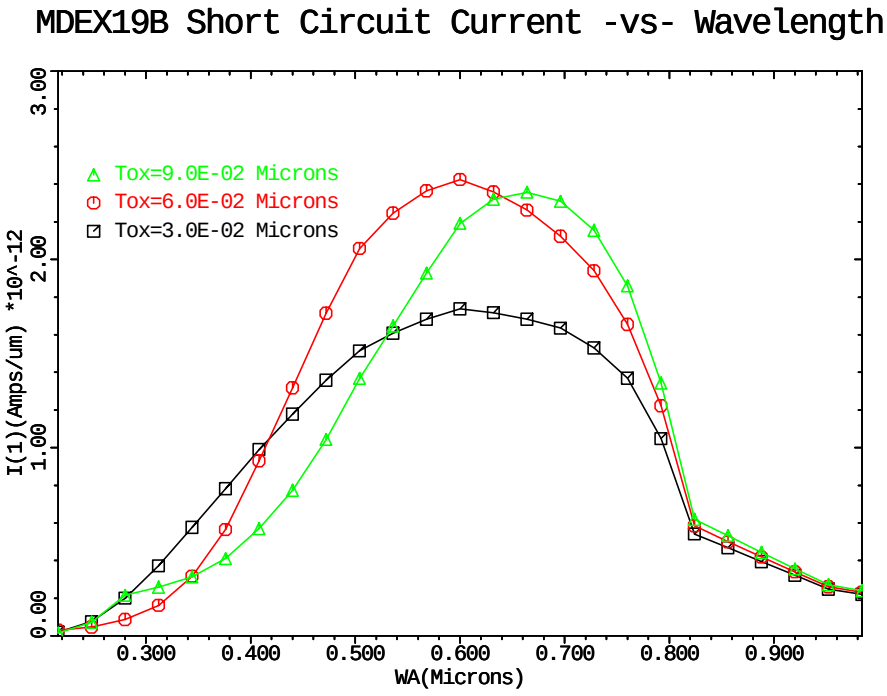


Figure 16-10 Short circuit current plot generated in lines 13 through 19 in file *mdex19b.inp*, Figure 16-6

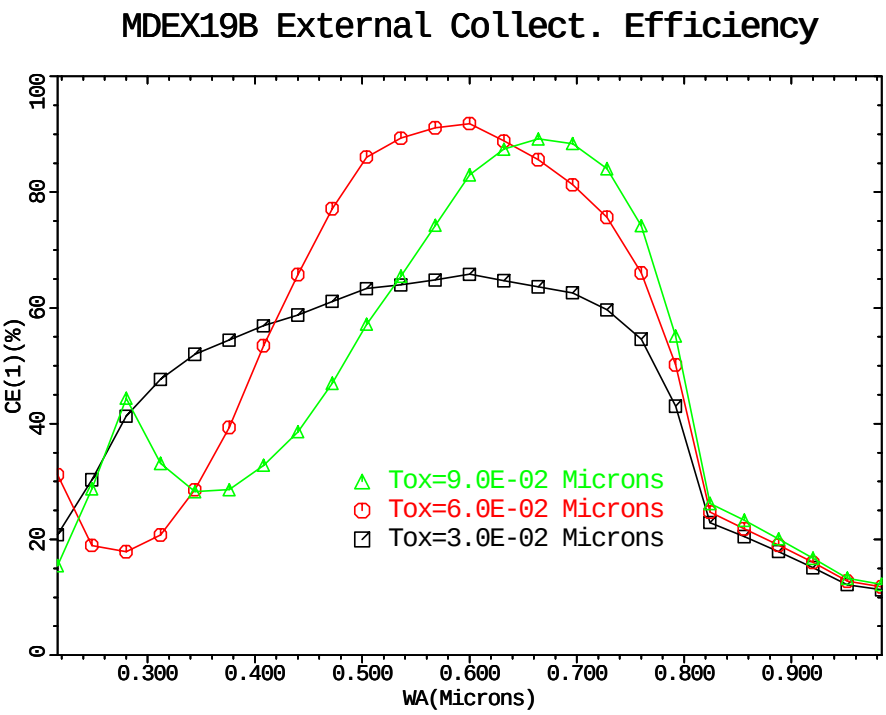


Figure 16-11 External collection efficiency plot generated in lines 21 through 27 in file *mdex19b.inp*, Figure 16-6

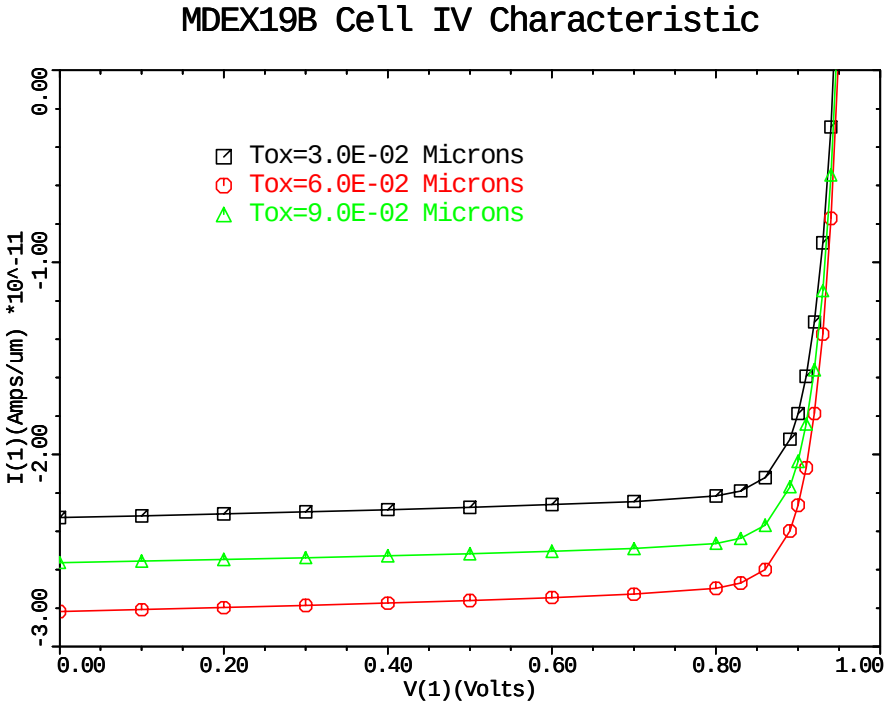


Figure 16-12 Cell IV plot generated in lines 29 through 35 in file *mdex19b.inp*, Figure 16-6

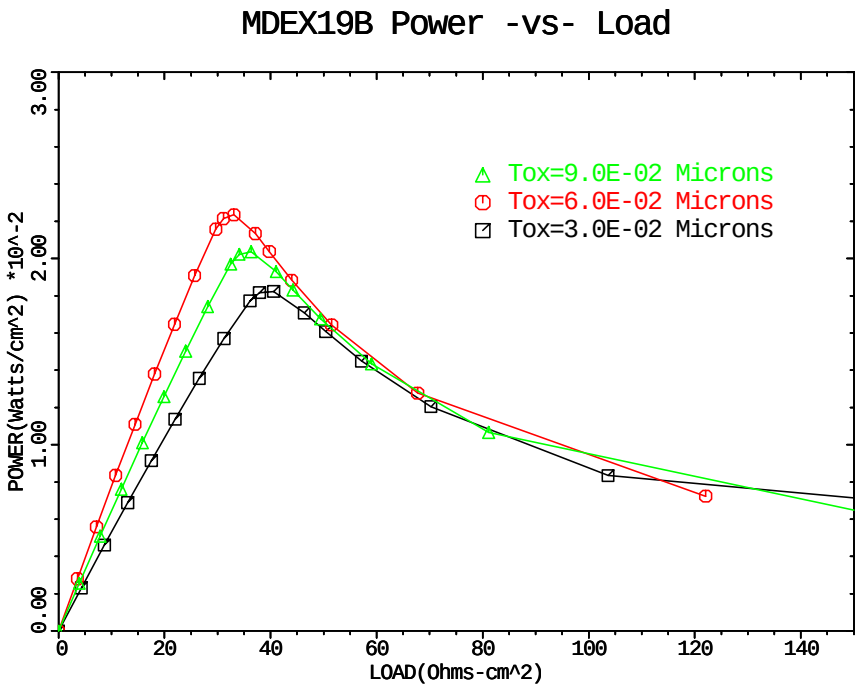


Figure 16-13 Power vs. load plot generated in lines 39 through 45 in file *mdex19b.inp*, Figure 16-6

Simulation of a GaAs Photodetector

In this example the use of the Medici OD-AAM is illustrated by performing some analysis on a GaAs photo detector. GaAs photo detectors have potential uses in a variety of optoelectronic integrated circuits (OEIC) because they are fast, have low background noise, and good off-state isolation.

Structure Generation and Solution Creation

The input file *mdex20a* creates the simulation structure and simulates the operation of the device when exposed to light with a wavelength of 0.82 microns and a flux of 1e20 photons/sec/cm^2. The purpose of the simulation is to calculate the photocurrent when the device is illuminated for 40pS so that the transit time can be determined. The photo response is also determined when the light is modulated using an AC signal under various bias conditions. In this example, the semi insulating layer on which the active region is formed is ignored for convenience.

Referring to Figure 16-14 which shows the simulation input file *mdex20a*, lines 2 through 8 generate the simulation structure which consists of a GaAs block 5 microns wide and 3 microns deep with a uniform N-type doping of 1e14 /cm^3. Additional specifications are as follows:

- Ohmic contacts are placed on the top of the structure at the left and right corners.

- The models used are specified at line 10.
- Since recombination effects are important, both **CONSRH** and **AUGER** are specified.
- At line 11, **FLDMOB**=2 specifies that the GaAs type mobility model (with negative differential velocity) is used.
- Line 12 specifies that both electrons and holes are solved for (since majority and minority carrier effects are important in optoelectronic devices).
- A plot of the device structure can be seen in Figure 16-15.

Steady State Solutions and Photogeneration

Lines 16 to 24 calculate a series of bias points without illumination. These are the dark current values for each particular bias value. The solutions themselves are saved in the files *PDC_03*, *PDC_05*, *PDC_1*, and *PDC_5*. The values of the drain current at each bias are in turn saved in assigned variables I003, I005, I01, and I05 for use in subsequent **EXTRACT** statements.

Line 26 specifies photogeneration and ray tracing.

- Light is directed straight down onto the surface of the device since the default value of **ANGLE** is 90 degrees and the light source is located 5 microns above the device at **Y.ORG**=-5.
- The wavelength is 0.82 microns and the photon flux is 1e20/cm^2/sec.
- The time dependence is specified as **PULSE** type and gives a constant pulse which starts at 10pS and stops at 50pS.

Lines 27 through 30 create a rayplot and a plot of photogeneration contours for the structure. These are shown in Figures 16-16 and 16-17.

Transient Simulations

At line 31, a loop is entered that calculates solutions at the 4 bias voltages (0.3, 0.5, 1.0 5.0) Volts. Line 30 loads the steady state solutions (calculated previously). These solutions set up the initial conditions for the transient.

The **EXTRACT** statement at line 36 calculates the actual photocurrent (IPHOT) by subtracting the dark current (@I005 … @I01) from the terminal current (@I(DRAIN)).

This new user-defined current is stored in the log file along with the normal terminal current. The **SOLVE** at line 37 calculates the transient solution, with an initial time step size of 2pS and a stopping time of 400pS.

Small-Signal AC Analysis

Lines 39 to 43 calculate the frequency dependent small-signal optical gain of the device by specifying **LMODU** on the **SOLVE** statement.

- The steady state solution is read back in at line 39.
- The frequency is swept from 1MHz to 100Ghz in 3 steps per decade.
- The light modulation magnitude is specified as 1e-4 W/cm^2 using **LSS**

.

```
 1... TITLE     Example 20A:  GaAs PHOTOCONDUCTIVE DETECTOR
 2... MESH
 3... X.MESH    WIDTH=5  H1=0.2
 4... Y.MESH    DEPTH=3  H1=0.2  H2=0.5
 5... REGION    GAAS
 6... ELECTR    NAME=SOURCE  TOP  X.MAX=0.5
 7... ELECTR    NAME=DRAIN   TOP  X.MIN=4.5
 8... PROFILE   N-TYPE  N.PEAK=1E14  UNIF
 9... PLOT.2D   GRID  SCALE  TITLE=" GaAs Photoconductor Grid"  FILL

10... MODELS    FLDMOB  CONMOB  CONSRH  AUGER
11... MOBIL     GAAS  FLDMOB=2

12... SYMB      CARR=2  NEWTON
13... METHOD    ITLIM=30
14... SOLVE     OUT.FILE=PDC_0

15... COMMENT   Solve and save W/O illumination (Dark current)
16... SOLVE     V(DRAIN)=0.3  OUT.FILE=PDC_03
17... EXTRACT   NAME=I003  NOW  EXP=@I(DRAIN)
18... SOLVE     V(DRAIN)=0.5  OUT.FILE=PDC_05
19... EXTRACT   NAME=I005  NOW  EXP=@I(DRAIN) CLEAR
20... SOLVE     V(DRAIN)=1    OUT.FILE=PDC_1
21... EXTRACT   NAME=I01  NOW  EXP=@I(DRAIN)  CLEAR
22... SOLVE     V(DRAIN)=5    OUT.FILE=PDC_5
23... EXTRACT   NAME=I05  NOW  EXP=@I(DRAIN)  CLEAR
24... EXTRACT   CLEAR

25... COMMENT   Photogeneration :
  ... +         Pulse excitation, start at 10p, width=40psec.
26... PHOTOGEN  RAYTRACE  X.ORG=2.  Y.ORG=-5.  WAVELENG=0.82  FLUX=1e20
  ... +         T0=10E-12  TPD=40E-12  PULSE  N.INTEG=12
27... PLOT.2D   RAYPLOT FILL
28... VECTOR    RAYTRACE
29... PLOT.2D
30... CONTOUR   PHOTOGEN FILL

31... LOOP      STEP=4
32...   ASSIGN    NAME=BIAS  C1=03  C2=05  C3=1  C4=5
33...   COMMENT   With illumination : transient analysis
34...   LOAD      IN.FILE="PDC_"@BIAS""
35...   LOG       OUT.FILE="PDC_T"@BIAS""
36...   EXTRACT   NAME=IPHOT  EXP=@I(DRAIN)-@I0@BIAS   AT.BIAS
37...   SOLVE     DT=2e-12  TSTOP=400e-12
38...   LOG       CLOSE

39...   LOAD      IN.FILE="PDC_"@BIAS""
40...   LOG       OUT.FILE="LM_"@BIAS""
41...   COMMENT   Perform small signal analysis
42...   SOLVE     L.MODU LSS=1e-4 FREQ=1.E6 FSTEP=2.154435 NFSTEP=15 MULT
43...   LOG       CLOSE
44...   L.END
```

Figure 16-14 Output of simulation input file *mdex20a*

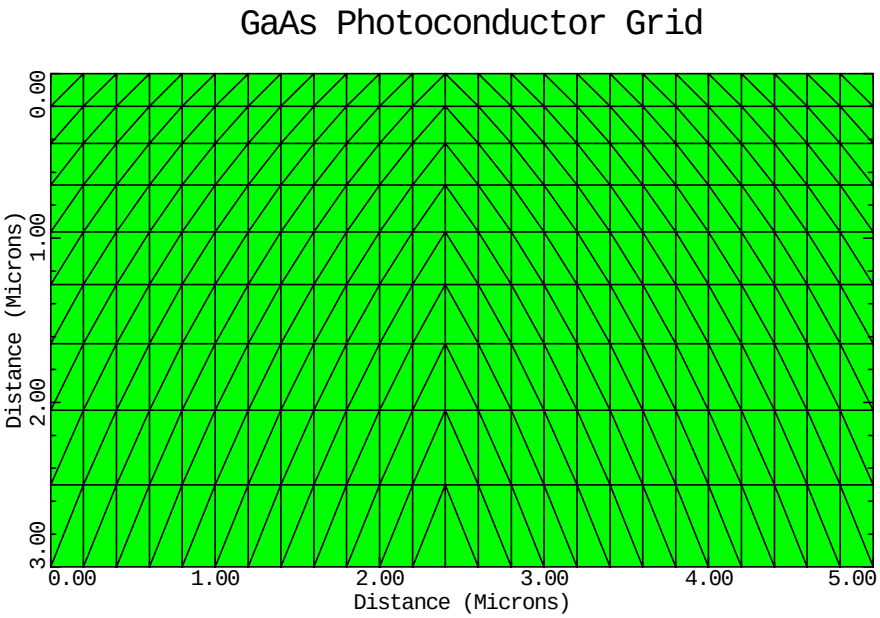


Figure 16-15 GaAs photodetector structure

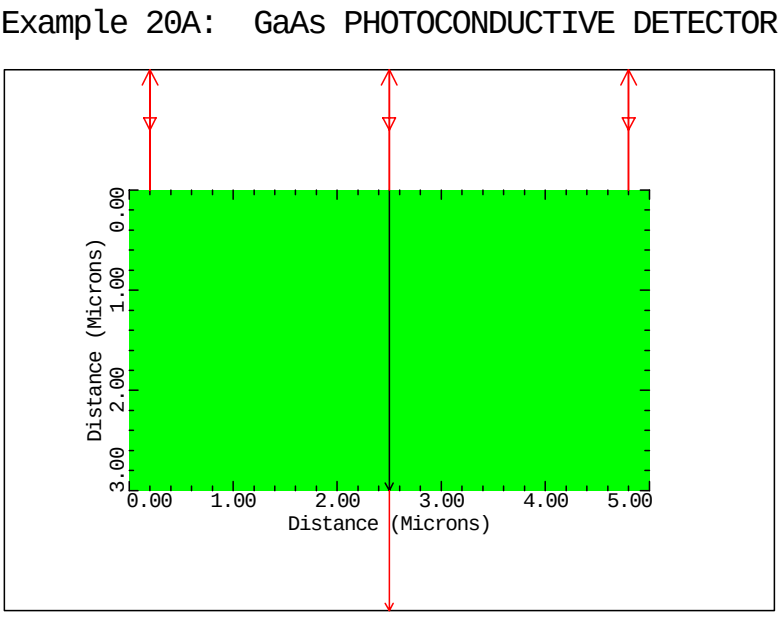


Figure 16-16 Simple rayplot showing the top illumination of the structure

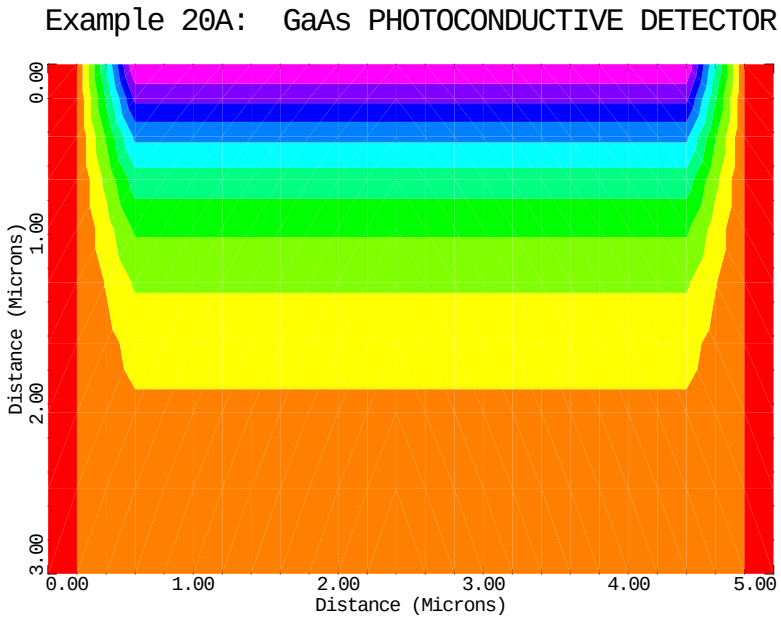


Figure 16-17 Photogeneration contours in the detector as a result of illumination

Examining Results

The second file of this example *mdex20b* (Figure 16-18) generates some I-V plots of the simulation results.

The first 10 lines generate a plot of the total drain current as a function of time. This plot is shown in Figure 16-19.

Lines 12 to 19 generate a plot of the photo current (IPHOT) which was previously obtained by subtracting the dark current from the total terminal current which is shown in Figure 16-20.

Lines 23 to 30 generate a plot of the normalized gain as a function of frequency (Figure 16-21). The normalization is performed by the **EXTRACT** statements at lines 21 and 22 which divide the gain by its value at 1Mhz.

```
 1... TITLE    GaAs Photoconductive Dectector
 2... COMMENT  Transient W illumination : Time vs. Contact current

 3... PLOT.1D  IN.FILE=PDC_T03  X.AXIS=TIME  Y.AXIS=I(DRAIN)
  ... +        TOP=2.e-6 BOT=0  SYM=1
  ... +        TITLE="GaAs PC type detector : Transient"
 4... PLOT.1D  IN.FILE=PDC_T05  X.AXIS=TIME  Y.AXIS=I(DRAIN)  UNCH
  ... +        SYM=2  COL=2
 5... PLOT.1D  IN.FILE=PDC_T1   X.AXIS=TIME  Y.AXIS=I(DRAIN)  UNCH
  ... +        SYM=3  COL=3
 6... PLOT.1D  IN.FILE=PDC_T5   X.AXIS=TIME  Y.AXIS=I(DRAIN)  UNCH
  ... +        SYM=4  COL=4
 7... LABEL    LABEL="V(DRAIN)=0.3V"  Y=.5e-6 SYMB=1
 8... LABEL    LABEL="V(DRAIN)=0.5V"  COL=2  SYMB=2
 9... LABEL    LABEL="V(DRAIN)= 1V"   COL=3  SYMB=3
10... LABEL    LABEL="V(DRAIN)= 5V"   COL=4  SYMB=4

11... COMMENT  Transient W illumination : Time vs. Photo current

12... PLOT.1D  IN.FILE=PDC_T03  X.AXIS=TIME  Y.AXIS=IPHOT TOP=4.e-7
  ... +        TITLE="GaAs PC type detector : Photocurrent"  SYM=1
13... PLOT.1D  IN.FILE=PDC_T05  X.AXIS=TIME  Y.AXIS=IPHOT  UNCH
  ... +        SYM=2  COL=2
14... PLOT.1D  IN.FILE=PDC_T1   X.AXIS=TIME  Y.AXIS=IPHOT   UNCH
  ... +        SYM=3  COL=3
15... PLOT.1D  IN.FILE=PDC_T5   X.AXIS=TIME  Y.AXIS=IPHOT   UNCH
  ... +        SYM=4  COL=4
16... LABEL    LABEL="V(DRAIN)=0.3V"  SYMB=1
17... LABEL    LABEL="V(DRAIN)=0.5V"  COL=2 SYMB=2
18... LABEL    LABEL="V(DRAIN)= 1V"   COL=3  SYMB=3
19... LABEL    LABEL="V(DRAIN)= 5V"   COL=4  SYMB=4

20... COMMENT  Gain
21... EXTRACT  NAME=Gain0  EXP=@IS(DRAIN)  COND="@LF=1e6"
22... EXTRACT  NAME=Gain   EXP="20*LOG10(@IS(DRAIN)/@Gain0)"
  ... +        UNIT=dB
23... PLOT.1D  IN.FILE=LM_03   X.AXIS=LF  Y.AXIS=Gain    X.LOG  TOP=1
  ... +        SYM=1  TITLE="Normalized Gain vs Frequency"
24... PLOT.1D  IN.FILE=LM_05  X.AXIS=LF  Y.AXIS=Gain   X.LOG  UNCH
  ... +        SYM=2  COL=2
25... PLOT.1D  IN.FILE=LM_1  X.AXIS=LF  Y.AXIS=Gain   X.LOG  UNCH
  ... +        SYM=3  COL=3
26... PLOT.1D  IN.FILE=LM_5  X.AXIS=LF  Y.AXIS=Gain   X.LOG  UNCH
  ... +        SYM=4  COL=4
27... LABEL    LABEL="V(DRAIN)=0.3V"  SYMB=1 Y=-10
28... LABEL    LABEL="V(DRAIN)=0.5V"  COL=2  SYMB=2
29... LABEL    LABEL="V(DRAIN)= 1V"   COL=3  SYMB=3
30... LABEL    LABEl="V(DRAIN)= 5V"   COL=4  SYMB=4
```

Figure 16-18 Output of the simulation input file *mdex20b*

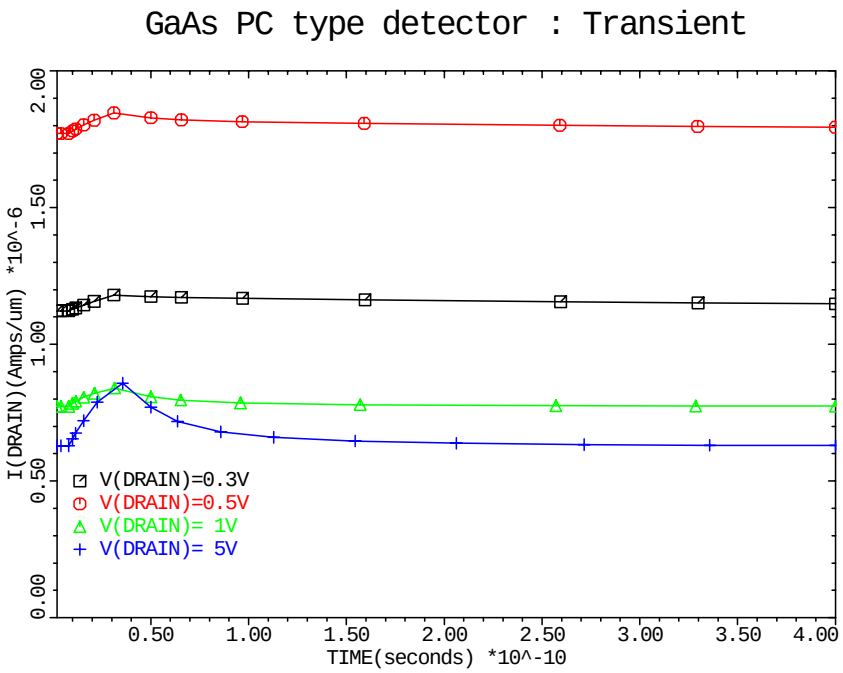


Figure 16-19 Drain current plot generated in lines 3 through 10 file *mdex20b*, Figure 16-18

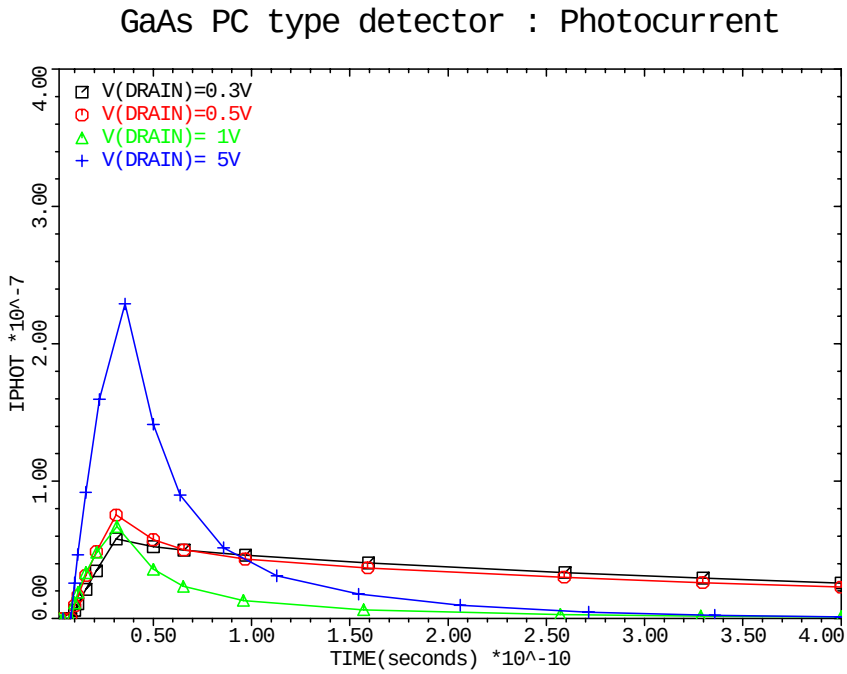


Figure 16-20 Photocurrent plot generated in lines 12 through 19 of file *mdex20b*, Figure 16-18

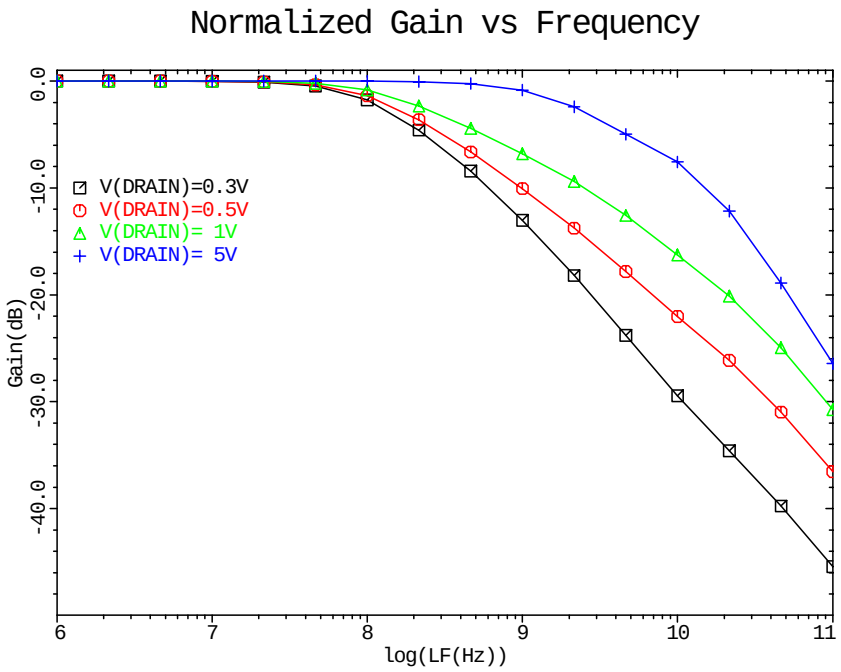


Figure 16-21 Optical gain plots generated in lines 23 through 30 of file *mdex20b*, Figure 16-18

Anisotropic Material Examples

Example Specifications

The Anisotropic Material Advanced Application Module (AM-AAM) is an extension to Medici which allows simulation of advanced anisotropic material properties within semiconductor materials. These effects are important in certain semiconductor materials such as silicon carbide. This chapter presents examples which illustrate the use of the AM-AAM.

Anisotropic Block Example

This example simulates a block of material with strongly anisotropic material properties. While no known material has such strongly anisotropic properties, the example is useful in that it illustrates the use of the AM-AAM and produces easily understood results.

Device Structure and Simulation

Figure 17-1 shows the input file *mdex22*. Lines 2 to 9 generate the structure which consists of a uniformly doped block of silicon with a small electrode on top and a flat electrode all along the bottom.

Tensor Mobility

Line 12 specifies the anisotropic electron mobility tensor=(0.1,0.9,1.0). This tensor is a dimensionless vector which multiplies the normal electron mobility (as calculated using the currently-selected mobility models).

Solution

Since no mobility models were selected in this example, the default isotropic mobility for electrons is 1000. The anisotropic model, therefore, gives the final mobilities as 100 in the X direction and 900 in the Y direction. Line 13 solves

with 0.1 Volts on the top electrode. Lines 16 and 17 generate a plot of current flow lines which is shown in Figure 17-2. This process is repeated 2 more times with mobility tensors of (1,1,1) and (0.9,0.1,1).

Plots

Compare the three plots generated with the 3 different tensors (Figure 17-2).

- In the figure on the left (0.1,0.9,1) (with the lowest X component of mobility) the current spreads out slowly in the X direction and the current must travel half way down the device before the current vectors align themselves with the Y axis.
- In the figure on the right (0.9,0.1,1), however, the current spreads rapidly across the top of the device and then abruptly turns and travels straight down parallel to the Y axis.

  This is to be expected since the mobility in the X direction is high and the current would initially rather travel horizontally in the high mobility direction.
- The center figure (1,1,1) corresponds to normal isotropic mobility.

```
 1... TITLE      MDEX22 Anisotropic Material Example
 2... MESH
 3... X.MESH     WIDTH=3 H1=0.1
 4... Y.MESH     WIDTH=3 H1=0.1
 5... REGION     SILICON
 6... REGION     OXIDE Y.MAX=0.1
 7... ELECTRODE  NAME=TOP  X.MIN=1 X.MAX=2 Y.MAX=0.1
 8... ELECTRODE  NAME=BOT  Y.MIN=2.9
 9... PROFILE    CONC=1e16 N.TYPE UNIF
10... $
11... SYMB       NEWT CARR=2
12... ANISOTROP  MU.N=(0.1,0.9,1.0) SIL
13... SOLVE      V(TOP)=0.1
14... PLOT.2D    ^MARKS ^LABELS TITLE="MDEX22 Anisotropic Material"
15... LABEL      Y=0.15 LABEL="Current Flow with Various Mobility Tensors"
  ... +          C.SIZ=0.38
16... PLOT.2D    X.LENG=4  FILL TITLE="(0.1,0.9,1)" SCALE ^CLEAR Y.OFF=5
17... CONTOUR    FLOW
18... $
19... ANISOTROP  MU.N=(1.0,1.0,1.0) SIL
20... SOLVE      V(TOP)=0.1
21... PLOT.2D    X.OFF=8  X.LENG=4 FILL ^CLEAR TITLE="(1,1,1)" SCALE
  ... +          Y.OFF=5
22... CONTOUR    FLOW
23... $
24... ANISOTROP  MU.N=(0.9,0.1,1.0) SIL
25... SOLVE      V(TOP)=0.1
26... PLOT.2D    X.OFF=14 X.LENG=4 FILL ^CLEAR TITLE="(0.9,0.1,1)"
  ... +          Y.OFF=5  SCALE
27... CONTOUR    FLOW
```

Figure 17-1 Output for the simulation input file *mdex22*

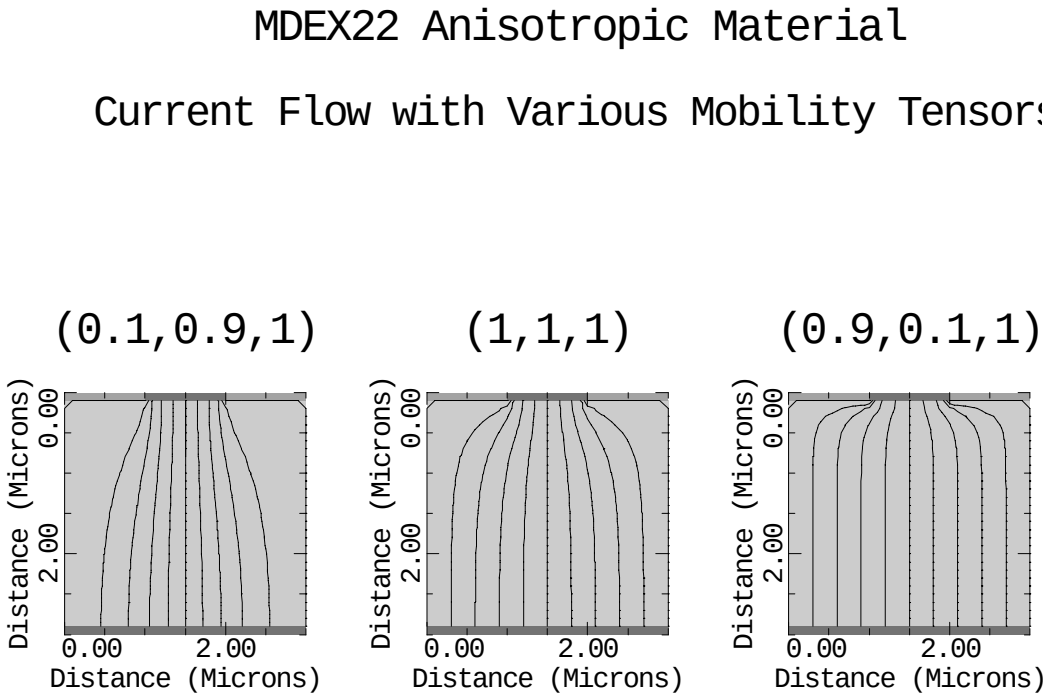


Figure 17-2 Current density vectors with various mobility tensors from file *mdex22*, Figure 17-1

Appendix A: Template Files

Introduction

Included with Medici are several template (**CALL**) files that automatically create, and apply biases to standard MOS and BIPOLAR structures. After you assign values to a few structure dimensions, the template will take care of the entire mesh generation process. Plots and printed information are provided to help verify the structure specification.

The impurity profiles used in these structures may be specified analytically or may come from the *Avant!* TCAD's process simulation programs—TMA SUPREM-3 or TSUPREM-4. These templates make it easy to simulate standard MOS and BIPOLAR structures. The templates are written in standard Medici input language so advanced users can copy and customize the templates to model unusual structures.

Available Templates

Brief descriptions of the presently available templates are given below. The following sections describe the templates in more detail and discuss how to use them. Examples of using the templates are given in Chapter 8.

Template	Description
mosdef0	Defines default values for all parameters and device dimensions used in the MOS templates. This file must be called prior to calling *mosstr0* or *mosstr1*.
mosstr0	Creates simulation structures for MOS devices. Both standard and LDD devices may be created. This template works well with both short- and long-channel devices.

Template	Description
mosstr1	Creates simulation structures for MOS devices. Both standard and LDD devices may be created. This template is well suited for doping profiles read from files and works best with short-channel devices.
mosgat0	Generates a family of gate characteristics (drain current versus gate voltage) for MOS devices and plots the results. This template may be used to bias any four-terminal MOS structure—it is not restricted to structures created by *mosstr0* or *mosstr1*.
mosdrn0	Generates a family of drain characteristics (drain current versus drain voltage) for MOS devices and plots the results. This template may be used to bias any four-terminal MOS structure—it is not restricted to structures created by *mosstr0* or *mosstr1*.
bipdef0	Defines default values for all parameters and device dimensions used in the bipolar templates. This file must be called prior to calling *bipstr0*.
bipstr0	Creates simulation structures for bipolar junction transistors.
bipgum0	Generates the Gummel characteristics for bipolar devices. The output consists of plots of collector and base current versus base-emitter voltage, current gain versus collector current, and cutoff frequency versus collector current. This template may be used to bias any three terminal bipolar structure—it is not restricted to structures created by *bipstr0*.

MOS Templates

This section describes the MOS templates which are available: *mosdef0*, *mosstr0*, *mosstr1*, *mosgat0*, and *mosdrn0*.

MOS Default Value File: *mosdef07*

The standard MOS structure created by the MOS template files is illustrated in Figure A-1. The templates can be used to create structures with or without LDD profiles. In addition to the dimensions shown, you must provide information about the impurity profiles and biases to be applied to the device.

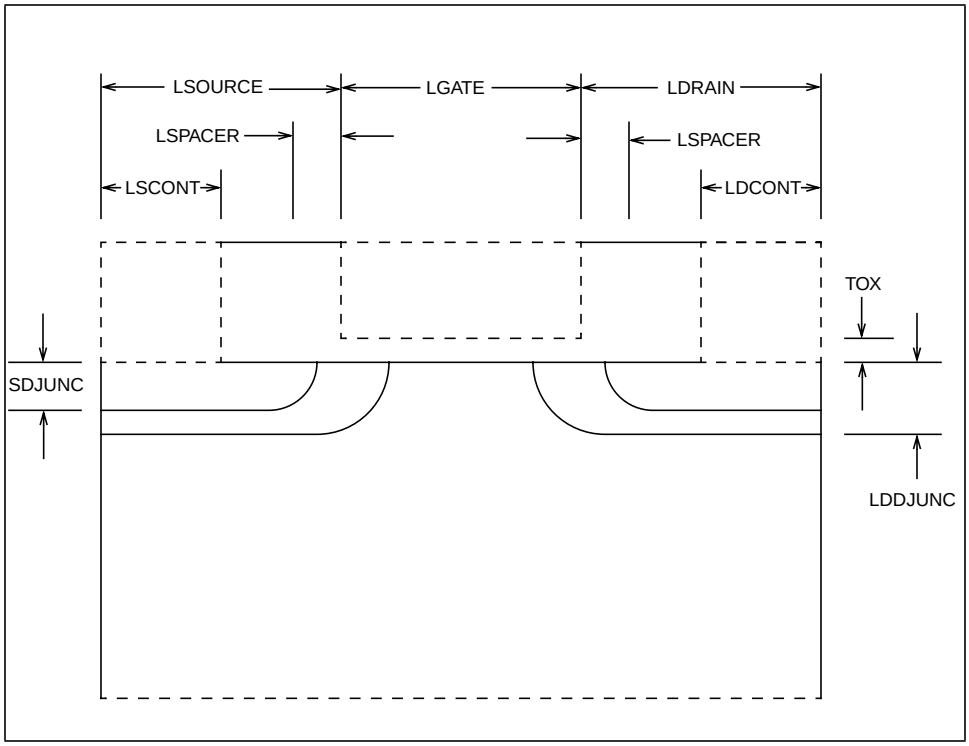


Figure A-1 MOS structure and definition used by the MOS templates

Default Value File

Definitions and default values for all of the parameters and dimensions required by the templates are given in the Default Value File, *mosdef0* (Figures A-2 and A-3). The Default Value File should be called before calling the Structure Definition Files *mosstr0* or *mosstr1*.

```
$-------------------
$  mosdef0
$-------------------
TITLE     TMA MEDICI Default Value File for MOS Structures
COMMENT   Structure Definitions
+    LGATE    = gate length (microns)
+    LSOURCE  = distance from left device edge
+               to gate edge (microns)
+    LSCONT   = length of source contact (microns)
+    LDRAIN   = distance from right device edge
+               to gate edge (microns)
+    LDCONT   = length of drain contact (microns)
+    LSPACER  = spacer thickness (microns)
+    TOX      = gate oxide thickness (microns)

ASSIGN    NAME=LGATE     N.VALUE=1.0
ASSIGN    NAME=LSOURCE   N.VALUE=1.0
ASSIGN    NAME=LSCONT    N.VALUE=0.5
ASSIGN    NAME=LDRAIN    N.VALUE=1.0
ASSIGN    NAME=LDCONT    N.VALUE=0.5
ASSIGN    NAME=LSPACER   N.VALUE=0.2
ASSIGN    NAME=TOX       N.VALUE=0.0250

COMMENT   Doping Information
+    TRANTYPE = transistor type (NMOS or PMOS)
+    PROFTYPE = profile type (ANALYTIC, SUPREM3, TSUPREM4,
+               or SUPRA)
+    LATD     = source/drain and LDD lateral diffusion factor
$  Analytic Profile Parameters
+    NSUB     = substrate doping (#/cm^3)
+    VTTYPE   = doping type for threshold adjust implant (N or P)
+    VTPEAK   = peak doping for threshold adjust implant (#/cm^3)
+    VTCHAR   = characteristic length
+               for threshold implant (microns)
+    SDPEAK   = peak doping for source/drain (#/cm^3)
+    SDJUNC   = junction depth for source/drain (microns)
+    LDDPEAK  = peak doping for lightly doped drain (#/cm^3)
+    LDDJUNC  = junction depth for lightly doped drain (microns)

$  Profile File Parameters
+    CHFILE   = SUPREM-3 output file containing channel doping
+    SDFILE   = SUPREM-3 output file containing source/drain dop.
+    LDDFILE  = SUPREM-3 output file containing LDD doping
+    2DFILE   = TSUPREM-4 or SUPRA file containing 2D doping
+    X.OFFSET = x-offset for profile read from 2DFILE (microns)
+    Y.OFFSET = y-offset for profile read from 2DFILE (microns)

ASSIGN    NAME=TRANTYPE  C.VALUE=NMOS
ASSIGN    NAME=PROFTYPE  C.VALUE=ANALYTIC
ASSIGN    NAME=LATD      N.VALUE=0.80

ASSIGN    NAME=NSUB      N.VALUE=3E15
ASSIGN    NAME=VTTYPE    C.VALUE=P
ASSIGN    NAME=VTPEAK    N.VALUE=2E16
ASSIGN    NAME=VTCHAR    N.VALUE=0.25
ASSIGN    NAME=SDPEAK    N.VALUE=1E20
ASSIGN    NAME=SDJUNC    N.VALUE=0.25
```

Figure A-2 First part of template file *mosdef0* listing parameters and their default values

```
ASSIGN    NAME=LDDPEAK   N.VALUE=2E18
ASSIGN    NAME=LDDJUNC   N.VALUE=0.35
ASSIGN    NAME=CHFILE    C.VALUE=S3CHAN
ASSIGN    NAME=SDFILE    C.VALUE=S3SD
ASSIGN    NAME=LDDFILE   C.VALUE=S3LDD
ASSIGN    NAME=2DFILE    C.VALUE=TS4PROF
ASSIGN    NAME=X.OFFSET  N.VALUE=0.
ASSIGN    NAME=Y.OFFSET  N.VALUE=0.

COMMENT   Grid Spacings, Ratio, Maximum Voltage
+    CHANSP   = vertical grid spacing in the channel (microns)
+    JUNCSP   = grid spacing at junctions (microns)
+    RATIO    = grid spacing ratio
+    VDBMAX   = maximum drain-substrate reverse bias (volts)

ASSIGN    NAME=CHANSP    N.VALUE=.0125
ASSIGN    NAME=JUNCSP    N.VALUE=.0250
ASSIGN    NAME=RATIO     N.VALUE=1.4
ASSIGN    NAME=VDBMAX    N.VALUE=5

COMMENT   Model, File, and Graphics Information
+    MODELS   = physical models to use during solutions
+    QSS      = gate oxide fixed interface charge (#/cm^2)
+    FILE     = prefix for output file names
+    SAVE     = TRUE if solution files are saved; otherwise FALSE
+    DEVICE   = graphics output device (X, SUN, etc.)
ASSIGN    NAME=MODELS
+         C.VALUE="CONMOB FLDMOB PRPMOB CONSRH AUGER BGN"
ASSIGN    NAME=QSS       N.VALUE=0
ASSIGN    NAME=FILE      C.VALUE="MOS"
ASSIGN    NAME=SAVE      C.VALUE="FALSE"
ASSIGN    NAME=DEVICE    C.VALUE="DEFAULT"

COMMENT   Electrode Name Assignments
ASSIGN    NAME=DRAIN     C.VALUE=Drain
ASSIGN    NAME=GATE      C.VALUE=Gate
ASSIGN    NAME=SOURCE    C.VALUE=Source
ASSIGN    NAME=SUBSTRAT  C.VALUE=Substrate

COMMENT   Biasing
+       VD0      = initial drain bias (volts)
+       VDSTEP   = drain bias step size (volts)
+       NDSTEP   = number of drain bias steps
+       VG0      = initial gate bias (volts)
+       VGSTEP   = gate bias step size (volts)
+       NGSTEP   = number of gate bias steps
+       VB0      = initial substrate bias (volts)
+       VBSTEP   = substrate bias step size (volts)
+       NBSTEP    = number of substrate bias steps

ASSIGN    NAME=VD0       N.VALUE=0.0
ASSIGN    NAME=VDSTEP    N.VALUE=0.2
ASSIGN    NAME=NDSTEP    N.VALUE=1
ASSIGN    NAME=VG0       N.VALUE=0.0
ASSIGN    NAME=VGSTEP    N.VALUE=0.2
ASSIGN    NAME=NGSTEP    N.VALUE=1
ASSIGN    NAME=VB0       N.VALUE=0.0
ASSIGN    NAME=VBSTEP    N.VALUE=-1.0
ASSIGN    NAME=NBSTEP    N.VALUE=1
```

Figure A-3 Second part of template file *mosdef0* listing parameters and their default values

Parameter and Dimension Values

If you want to change a parameter value or dimension you may either edit *mosdef0*, or specify the new value using an **ASSIGN** statement after calling *mosdef0*.

For example, the default drawn gate length in *mosdef0* is 1 micron. The following input statements use the template file *mosstr0* to create a 1 micron long NMOS transistor:

```
CALL    FILE=mosdef0
CALL    FILE=mosstr0
```

To create a transistor with a 3 micron gate length, an **ASSIGN** statement is added to override the default value for **LGATE**:

```
CALL    FILE=mosdef0
ASSIGN  NAME=LGATE    N.VALUE=3
CALL    FILE=mosstr0
```

Any other parameter or device dimension defined in *mosdef0* may be changed in a similar way.

Additional Parameters

There are several parameters in *mosdef0* which deserve additional discussion.

Boundary Conditions

The parameters **LDRAIN**, **LSOURCE**, **NSUB**, and **VDMAX** should be chosen carefully to ensure the boundary conditions at the left, right, and bottom of the simulation structure do not adversely affect the results.

If you intend to compare the simulation results with experimental data, **LDRAIN** and **LSOURCE** should be set equal to the distance from the gate edge to the center of the drain or source contacts, respectively, in your test structure layout.

The parameters **NSUB**, the substrate doping concentration, and **VDBMAX**, the maximum drain/substrate reverse bias, are used to calculate the maximum depth of the depletion region under the drain. The structure depth is chosen so the maximum depletion depth does not reach the bottom of the structure. Even if you are going to use impurity profiles from a process simulator it is important to specify **NSUB**.

Source and Drain Profiles

The parameter **LATD** defines the horizontal extent of the source and drain profiles when analytic or TMA SUPREM-3 profiles are used. This parameter can have a significant effect on the effective channel length of short channel transistors. Generally accepted values for **LATD** range from 0.7 to 0.8.

Saved Files

The parameters **FILE** and **SAVE** determine the name and number of saved files. The **FILE** parameter is a prefix used at the beginning of each saved file name. If you are using these templates in a loop, you may want to vary the **FILE** parameter to ensure each set of files has a unique name.

The **SAVE** parameter determines how many solution files are saved. By default, **SAVE** is set to "FALSE" so only a few required files are saved. If **SAVE** is set to "TRUE" then every solution will be saved to a file.

MOS Structure Definition Templates: *mosstr0* and *mosstr1*

Two different structure definition template files, *mosstr0* and *mosstr1*, have been provided. Each template represents a different approach to mesh generation.

Template *mosstr0*

In *mosstr0*, a rectangular mesh is generated with fine mesh spacings located at important locations such as junctions. **ELIMINATE** statements are used to remove unnecessary mesh nodes.

Advantage

The principle advantage of this technique is that the aspect ratio of the mesh can be quite large (the mesh aspect ratio is the horizontal grid spacing divided by the vertical grid spacing). Template *mosstr0* can be used to simulate large structures without creating a prohibitively large number of grid nodes.

Limitation

The principle limitation of this gridding technique is that it requires knowledge of the junction locations to ensure correct placement of the fine grid sections. Even if you are using impurity profiles from a one- or two-dimensional process simulator you must specify the profile parameters **NSUB**, **VTTYPE**, **VTPEAK**, **VTCHAR**, **SDPEAK**, **SDJUNC**, **LDDPEAK**, and **LDDJUNC**.

Template *mosstr1*

In *mosstr1*, a coarse rectangular mesh is generated and then several regrids are performed to add mesh nodes near the junctions and in the channel. When modeling short (3 microns or less) channel length devices using profiles from a one- or two-dimensional simulator, the *mosstr1* template is recommended. For longer channel devices, *mosstr0* should be used.

Advantage

The advantage of this technique is that the junction locations do not have to be known in advance. Because of this *mosstr1* can be used with profiles from any source. The profile parameters, such as **NSUB** and **VTTYPE**, do not have to be specified when using impurity profiles from one- or two-dimensional simulators in .

Limitation

The main limitation is that to avoid the formation of obtuse triangles or mesh *holes*, the aspect ratio of the initial rectangular mesh is set to two. As a result, *mosstr1* does not allocate mesh nodes as efficiently as *mosstr0*—i.e. *mosstr1* structures will probably contain more mesh nodes than *mosstr0* structures.

MOS Bias Templates: *mosgat0* and *mosdrn0*

The MOS Bias Template Files *mosgat0* and *mosdrn0* may be used to simulate the gate and drain characteristics for any four terminal MOS structure.

Required Variables

The following variables must be assigned values prior to calling *mosgat0* or *mosdrn0*:

Variable	Description
TRANTYPE	The transistor type. Valid values are "NMOS" or "PMOS".
MODELS	A list of models to be used.
QSS	The interface fixed charge.
FILE	A prefix used at the beginning of output file names.
SAVE	Determines whether or not solution files are saved: if set to "TRUE" all solutions are saved, if set to "FALSE" only a few essential solutions are saved.
VD0	The initial drain voltage.
VDSTEP	The drain voltage step. Not used in *mosgat0*.
NDSTEP	The number of drain voltage solutions. Not used in *mosgat0*.
VG0	The initial gate voltage.
VGSTEP	The gate voltage step.
NGSTEP	The number of gate voltage solutions.
VB0	The initial substrate voltage.
VBSTEP	The substrate voltage step. Not used in *mosdrn0*
NBSTEP	The number of substrate voltage solutions. Not used in *mosdrn0*.

Default values for all of these variables are defined in *mosdef0*. However, the default biases given in *mosdef0* should not be used when calling *mosgat0* or *mosdrn0* (the default biases are given in *mosdef0* only to prevent undefined variable errors).

Template Examples

This section contains examples of input statements showing the generation of gate and drain IV characteristics for NMOS and PMOS transistors.

NMOS

The following input statements can be used to generate a set of gate and drain characteristics for an NMOS transistor:

```
$ Specify transistor type, models, etc.  These are the
+ default values normally set in mosdef0.
```

```
ASSIGN  NAME=TRANTYPE  C.VAL="NMOS"
ASSIGN  NAME=MODELS    C.VAL="CONMOB FLDMOB PRPMOB CONSRH AUGER BGN"
ASSIGN  NAME=QSS       N.VAL=0
ASSIGN  NAME=FILE      C.VAL="MOS"
ASSIGN  NAME=SAVE      C.VAL="FALSE"
ASSIGN  NAME=DEVICE    C.VAL="DEFAULT"

$-----------------------------------------------------------
$ Specify the bias conditions for the gate characteristics
$-----------------------------------------------------------

$ Set the drain bias to 0.1 volts.
ASSIGN  NAME=VD0       N.VAL=0.1

$ Step the gate bias from 0 volts to 5 volts
+ in 0.2 volt steps.
ASSIGN  NAME=VG0       N.VAL=0
ASSIGN  NAME=VGSTEP    N.VAL=0.2
ASSIGN  NAME=NGSTEP    N.VAL=(5/0.2)+1

$ Perform a gate sweep at two different substrate biases,
+ Vb=0 and Vb=-1 volts.
ASSIGN  NAME=VB0       N.VAL=0
ASSIGN  NAME=VBSTEP    N.VAL=-1.0
ASSIGN  NAME=NBSTEP    N.VAL=2

$ Perform the gate sweeps.
CALL  FILE=mosgat0

$-----------------------------------------------------------
$ Specify the bias conditions for the drain characteristics
$-----------------------------------------------------------

$ Step the drain bias from 0 volts to 5 volts
+ in 0.2 volt steps.
ASSIGN  NAME=VD0       N.VAL=0
ASSIGN  NAME=VDSTEP    N.VAL=0.2
ASSIGN  NAME=NDSTEP    N.VAL=(5/0.2)+1

$ Perform the drain sweeps with gate biases
+ of 1, 2, 3, 4, and 5 volts.
ASSIGN  NAME=VG0       N.VAL=1
ASSIGN  NAME=VGSTEP    N.VAL=1
ASSIGN  NAME=NGSTEP    N.VAL=5

$ Perform the drain sweeps with the substrate grounded.
ASSIGN  NAME=VB0       N.VAL=0
ASSIGN  NAME=VBSTEP    N.VAL=0
ASSIGN  NAME=NBSTEP    N.VAL=1

$ Perform the drain sweeps.
CALL  FILE=mosdrn0
```

PMOS

The following input statements can be used to generate a set of gate and drain characteristics for a PMOS transistor:

```
$ Specify transistor type, models etc.
ASSIGN  NAME=TRANTYPE  C.VAL="PMOS"
ASSIGN  NAME=MODELS    C.VAL="CONMOB FLDMOB PRPMOB CONSRH AUGER BGN"
ASSIGN  NAME=QSS       N.VAL=0
ASSIGN  NAME=FILE      C.VAL="PMOS"
ASSIGN  NAME=SAVE      C.VAL="FALSE"
ASSIGN  NAME=DEVICE    C.VAL="DEFAULT"
```

```
$-------------------------------------------------------------
$ Specify the bias conditions for the gate characteristics
$-------------------------------------------------------------

$ Set the drain bias to -0.1 volts.
ASSIGN  NAME=VD0       N.VAL=-0.1

$ Step the gate bias from 0 volts to -5 volts
+ in -0.2 volt steps.
ASSIGN  NAME=VG0       N.VAL=0
ASSIGN  NAME=VGSTEP    N.VAL=-0.2
ASSIGN  NAME=NGSTEP    N.VAL=(5/0.2)+1

$ Perform a gate sweep at two different substrate biases,
+ Vb=0 and Vb=+1 volts.
ASSIGN  NAME=VB0       N.VAL=0
ASSIGN  NAME=VBSTEP    N.VAL=1.0
ASSIGN  NAME=NBSTEP    N.VAL=2

$ Perform the gate sweeps.
CALL  FILE=mosgat0

$-------------------------------------------------------------
$ Specify the bias conditions for the drain characteristics
$-------------------------------------------------------------

$ Step the drain bias from 0 volts to -5 volts
+ in -0.2 volt steps.
ASSIGN  NAME=VD0       N.VAL=0
ASSIGN  NAME=VDSTEP    N.VAL=-0.2
ASSIGN  NAME=NDSTEP    N.VAL=(5/0.2)+1

$ Perform the drain sweeps with gate biases
+ of -1, -2, -3, -4, and -5 volts.
ASSIGN  NAME=VG0       N.VAL=-1
ASSIGN  NAME=VGSTEP    N.VAL=-1
ASSIGN  NAME=NGSTEP    N.VAL=5

$ Perform the drain sweeps with the substrate grounded.
ASSIGN  NAME=VB0       N.VAL=0
ASSIGN  NAME=VBSTEP    N.VAL=0
ASSIGN  NAME=NBSTEP    N.VAL=1

$ Perform the drain sweeps.
CALL  FILE=mosdrn0
```

Single Carrier Solutions

The template files, *mosgat0* and *mosdrn0*, perform single carrier solutions.

The *TRANTYPE* variable is used to determine whether electron or hole solutions are performed. *TRANTYPE* does not, however, control the polarity of the applied biases. Bias values should be chosen carefully so junctions are not forward biased. Since only single carrier solutions are calculated, breakdown analysis using the impact ionization model cannot be performed using these templates.

Note that variables *NGSTEP*, *NDSTEP*, and *NBSTEP* specify the number of solutions to be performed for the gate, drain, and substrate, respectively, not the num-

ber of steps. (This is slightly different than the **SOLVE** statement **NSTEP** parameter.) To be sure your bias sweeps do not end one step short of the desired final bias, choose a value equal to

```
NSTEP = 1 + (FINAL BIAS - INITIAL BIAS)/(BIAS STEP).
```

Bipolar Templates

This section describes the bipolar templates which are available: *bipdef0*, *bipstr0*, and *bipgum0*.

Bipolar Default Value File: *bipdef0*

The standard bipolar structure created by the bipolar template file is illustrated in Figure A-4. The template can be used to create structures with or without a buried layer in the collector or an extrinsic base diffusion. In addition to the dimensions shown, you must provide information about the impurity profiles and biases to be applied to the device.

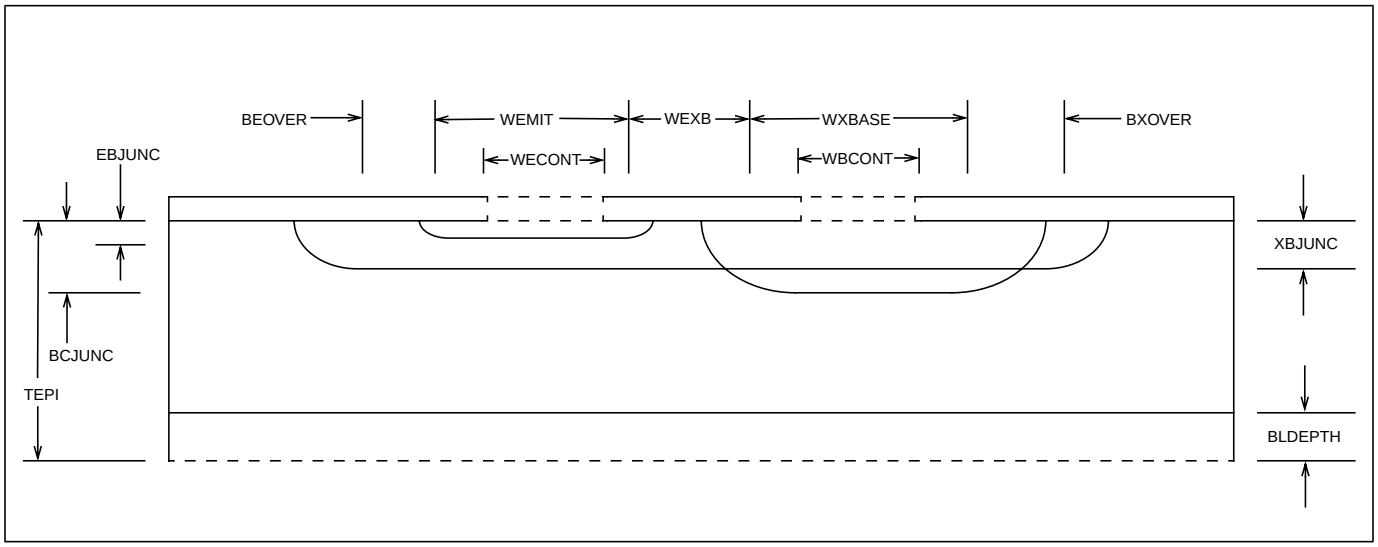


Figure A-4 Bipolar structure and definitions used by the bipolar templates

Default Value File

Definitions and default values for all of the parameters and dimensions required by the templates are given in the Default Value File, *bipdef0* (Figures A-5 and A-6) The Default Value File should be called first, before calling the Structure Definition File *bipstr0*.

```
$-------------------
$  bipdef0
$-------------------
TITLE     TMA MEDICI Default Value File for BJT Structure

COMMENT   Structure Definitions
+    WEMIT    = emitter width (microns)
+    WECONT   = emitter contact width (microns)
+    WXBASE   = extrinsic base width (microns)
+    WBCONT   = base contact width (microns)
+    WEXB     = emitter to extrinsic base distance (microns)
+    BEOVER   = base diffusion edge
+               to emitter diffusion edge (microns)
+    BXOVER   = base diffusion edge
+               to x-base diffusion edge (microns)
+    TEPI     = structure depth (epi thickness) (microns)

ASSIGN    NAME=WEMIT     N.VALUE=1.0
ASSIGN    NAME=WECONT    N.VALUE=1.0
ASSIGN    NAME=WXBASE    N.VALUE=1.0
ASSIGN    NAME=WBCONT    N.VALUE=1.0
ASSIGN    NAME=WEXB      N.VALUE=0.5
ASSIGN    NAME=BEOVER    N.VALUE=0.5
ASSIGN    NAME=BXOVER    N.VALUE=0.5
ASSIGN    NAME=TEPI      N.VALUE=2.0

COMMENT   Doping Information
+    TRANTYPE = transistor type (NPN or PNP)
+    PROFTYPE = profile type (ANALYTIC, SUPREM3, TSUPREM4,
+                             or SUPRA)
+    LATD     = lateral diffusion factor
$  Analytic Profile Parameters
+    NEPI     = epitaxial layer doping (#/cm^3)
+    BLPEAK   = peak doping for buried layer (#/cm^3)
+    BLDEPTH  = depth of buried layer (microns)
+    BPEAK    = peak doping for intrinsic base (#/cm^3)
+    YBPEAK   = distance from surface
+               to peak base doping (microns)
+    BCJUNC   = base-collector junction depth (microns)
$    XBPEAK   = peak doping for extrinsic base (#/cm^3)
+    XBJUNC   = extrinsic base-collector junction depth (microns)
+    EPEAK    = peak doping for emitter (#/cm^3)
+    EBJUNC   = emitter-base junction depth (microns)

$  Profile File Parameters
+    EPIFILE  = SUPREM-3 output file containing
+               epi and buried layer
+    BFILE    = SUPREM-3 output file containing
+               intrinsic base doping
+    XBFILE   = SUPREM-3 output file containing
+               extrinsic base doping
+    2DFILE   = TSUPREM-4 or SUPRA file containing 2D doping
+    X.OFFSET = x-offset for profile read from 2DFILE (microns)
+    Y.OFFSET = y-offset for profile read from 2DFILE (microns)
```

Figure A-5 First part of file *bipdef0* listing available parameters and their defaults

```
ASSIGN    NAME=TRANTYPE  C.VALUE=NPN
ASSIGN    NAME=PROFTYPE  C.VALUE=ANALYTIC
ASSIGN    NAME=LATD      N.VALUE=0.80

ASSIGN    NAME=NEPI      N.VALUE=1E16
ASSIGN    NAME=BLPEAK    N.VALUE=1E19
ASSIGN    NAME=BLDEPTH   N.VALUE=0.5
ASSIGN    NAME=BPEAK     N.VALUE=4E17
ASSIGN    NAME=YBPEAK    N.VALUE=0.0
ASSIGN    NAME=BCJUNC    N.VALUE=0.40
ASSIGN    NAME=XBPEAK    N.VALUE=5E19
ASSIGN    NAME=XBJUNC    N.VALUE=0.45
ASSIGN    NAME=EPEAK     N.VALUE=1E20
ASSIGN    NAME=EBJUNC    N.VALUE=0.10

ASSIGN    NAME=EPIFILE   C.VALUE=S3EPI
ASSIGN    NAME=BFILE     C.VALUE=S3BASE
ASSIGN    NAME=XBFILE    C.VALUE=S3XBASE
ASSIGN    NAME=2DFILE    C.VALUE=TS4PROF
ASSIGN    NAME=X.OFFSET  N.VALUE=0.
ASSIGN    NAME=Y.OFFSET  N.VALUE=0.

COMMENT   Grid Spacings, Ratio, Maximum Voltage
+    EBSP     = grid spacing at emitter-base junction (microns)
+    BCSP     = grid spacing at collector-base junction (microns)
+    RATIO    = grid spacing ratio
+    VCBMAX   = maximum collector-base reverse bias (volts)

ASSIGN    NAME=EBSP      N.VALUE=.0125
ASSIGN    NAME=BCSP      N.VALUE=.0250
ASSIGN    NAME=RATIO     N.VALUE=1.5
ASSIGN    NAME=VCBMAX    N.VALUE=3

COMMENT   Solution and File Information
+    MODELS   = physical models to use during solutions
+    FILE     = prefix for output file names
+    SAVE     = TRUE if solution files are saved; otherwise FALSE
+    DEVICE   = graphics output device (X, SUN, etc.)

ASSIGN    NAME=MODELS    C.VALUE="CONMOB FLDMOB CONSRH AUGER BGN"
ASSIGN    NAME=FILE      C.VALUE="BJT"
ASSIGN    NAME=SAVE      C.VALUE="FALSE"
ASSIGN    NAME=DEVICE    C.VALUE="DEFAULT"

COMMENT   Electrode Name Assignments
ASSIGN    NAME=COLLECT   C.VALUE=Collector
ASSIGN    NAME=BASE      C.VALUE=Base
ASSIGN    NAME=EMITTER   C.VALUE=Emitter

COMMENT   Biasing
+       VC0      = initial collector bias (volts)
+       VCSTEP   = collector bias step size (volts)
+       NCSTEP   = number of collector bias steps
+       VB0      = initial base bias (volts)
+       VBSTEP   = base bias step size (volts)
+        NBSTEP   = number of base bias steps

ASSIGN    NAME=VC0       N.VALUE=3.0
ASSIGN    NAME=VCSTEP    N.VALUE=1.0
ASSIGN    NAME=NCSTEP    N.VALUE=1
ASSIGN    NAME=VB0       N.VALUE=0.0
ASSIGN    NAME=VBSTEP    N.VALUE=0.1
ASSIGN    NAME=NBSTEP    N.VALUE=10
```

Figure A-6 Part 2 of file *bipdef0* listing available parameters and defaults

Parameter and Dimension Values

If you want to change a parameter value or dimension you may either edit *bipdef0*, or specify the new value using an **ASSIGN** statement after calling *bipdef0*.

For example, the default emitter stripe width in *bipdef0* is 1 micron. The following input statements use the template file *bipstr0* to create a bipolar transistor with a 1 micron emitter stripe width:

```
CALL    FILE=bipdef0
CALL    FILE=bipstr0
```

To create a transistor with a 3 micron emitter stripe width, an **ASSIGN** statement is added to override the default value for *WEMIT*:

```
CALL    FILE=bipdef0
ASSIGN  NAME=WEMIT    N.VALUE=3
CALL    FILE=bipstr0
```

Any other parameter or device dimension defined in *bipdef0* may be changed in a similar way.

Additional Parameters

There are several parameters in *bipdef0* which need additional discussion.

Simulation Structure Dimensions

The parameters **TEPI**, **NEPI**, and **VCBMAX** are used to determine the dimensions of the simulation structure.

TEPI is the depth of the structure. If a buried layer is not used, then **TEPI** should be deep enough that the collector-base depletion region does not extend to the bottom of the structure.

The epitaxial layer impurity concentration, **NEPI**, and the maximum collector-base reverse bias, **VCBMAX**, are used to calculate the distance from the base profile to the left and right edges of the simulation structure.

The intent is to create a structure wide enough that the collector-base depletion region does not extend to the structure boundaries. Even if the impurity profiles used come from a one- or two-dimensional process simulator, **NEPI** must be specified to ensure the structure is large enough.

Self-Aligned Bipolar Structures

To allow modeling of self-aligned bipolar structures, the width of the diffused emitter stripe, **WEMIT**, may be set equal to the emitter contact width **WECONT**. Similarly, the width of the extrinsic base diffusion, **WXBASE**, may be set equal to the base contact width, **WBCONT**. Setting the contact widths greater than the diffusion widths will result in an error.

Specifying small, or zero, values for the emitter to extrinsic base distance, **WEXB**, the base-emitter overlap distance, **BEOVER**, or the base to extrinsic base overlap distance, **BXOVER**, will also generate an error.

Lateral Diffusion

The lateral diffusion factor, **LATD**, determines the horizontal extent of the extrinsic and intrinsic base profiles and the emitter profile. These horizontal extents may not be specified independently. It may be appropriate to specify different values of **LATD**, depending on the operating conditions you are interested in simulating.

For example, when simulating the normal operating characteristics of the transistor a value of **LATD** should be chosen which best represents the lateral emitter profile. But, when simulating the collector-base breakdown behavior of the device the lateral base profile is more important than the lateral emitter profile and **LATD** should be specified accordingly. Generally accepted values of **LATD** range between 0.7 and 0.8.

Saved Files

The parameters **FILE** and **SAVE** determine the name and number of saved files. The **FILE** parameter is a prefix used at the beginning of each saved file name. If you are using these templates in a loop, you may want to vary the **FILE** parameter to ensure each set of files has a unique name.

The **SAVE** parameter determines how many solution files are saved. By default, **SAVE** is set to FALSE so only a few required files are saved. If **SAVE** is set to TRUE then every solution will be saved to a file.

Grid and Mesh Spacing

The parameters **EBSP**, **BCSP**, and **RATIO** determine the grid spacing at the emitter-base and base-collector junctions and the rate at which the mesh spacing can increase, respectively. The emitter-base grid spacing is critical for accurate modeling of bipolar gain and should not be changed.

The vertical mesh spacing under the emitter, between the emitter-base and base-collector junctions is important when modeling early voltage. Specifying a smaller value of **RATIO** may improve early voltage prediction at the expense of larger mesh counts and simulation times.

Bipolar Structure Definition Template: *bipstr0*

The bipolar structure created by *bipstr0* begins as a rectangular mesh with fine mesh spacings located at important locations such as junctions. **ELIMINATE** statements are used to remove unnecessary mesh nodes.

Advantage

The principle advantage of this technique is that the aspect ratio of the mesh can be quite large. (The mesh aspect ratio is the horizontal grid spacing divided by the vertical grid spacing.) Because of this *bipstr0* can be used to simulate large structures without using a prohibitively large number of grid nodes.

Limitation

The principle limitation of this gridding technique is that it requires knowledge of the junction locations to ensure correct placement of the fine grid sections. Even if impurity profiles from a one- or two-dimensional process simulator are used, the profile parameters **NEPI**, **BLPEAK**, **BLDEPTH**, **BPEAK**, **YPEAK**, **BCJUNC**, **XBPEAK**, **XBJUNC**, **EPEAK**, and **EBJUNC** must still be specified.

Base and Emitter Contacts

By default, *bipstr0* uses neutral (ohmic) base and emitter contacts. Polysilicon base and emitter contacts may be approximated by specifying a finite recombination velocity on a **CONTACT** statement after calling *bipstr0*. For example, the following input statements add a polysilicon emitter to the default bipolar structure:

```
CALL     FILE=bipdef0
CALL     FILE=bipstr0
CONTACT  NUM=EMITTER  SURF.REC  VSURFN=1.07  VSURFP=1.04  WORKF=0
```

Bipolar Bias Template: *bipgum0*

The bipolar bias template file, *bipgum0*, may be used to simulate the Gummel characteristics of any three terminal bipolar structure. The following variables must be assigned prior to calling *bipgum0*:

Variable	Description
MODELS	A list of models to be used.
FILE	A prefix used at the beginning of output file names.
SAVE	Determines whether or not solution files are saved: if set to "TRUE" all solutions are saved, if set to "FALSE" only a few essential solutions are saved.
TIFFILE	If set to "TRUE" saved solution files are in TIF format; if set to "FALSE" saved solution files are in binary format.
VC0	The collector voltage.
VB0	The initial base voltage.
VBSTEP	The base voltage step.
NBSTEP	The number of base voltage solutions.

Default values for all of these variables are defined in *bipdef0*. However, the default biases given in *bipdef0* should not be used when calling *bipgum0* (the default biases are given in *bipdef0* only to prevent undefined variable errors).

Examples

This section contains examples detailing Gummel characteristics of the NPN and PNP bipolar transistors.

NPN

The following input statements can be used to simulate the Gummel characteristics of an NPN transistor:

```
$ Set the collector bias to 3 volts.
ASSIGN  NAME=VC0  N.VALUE=3.0

$ Step the base voltage from 0 to 1 volts
+ in 0.1 volt increments.
ASSIGN  NAME=VB0     N.VALU=0.0
ASSIGN  NAME=VBSTEP  N.VALU=0.1
ASSIGN  NAME=NBSTEP  N.VALU=11

$ Perform the base sweep.
CALL  FILE=bipgum0
```

PNP

The following input statements can be used to simulate the Gummel characteristics of a PNP transistor:

```
$ Set the collector bias to -3 volts.
ASSIGN  NAME=VC0  N.VALUE=-3.0

$ Step the base voltage from 0 to -1 volts
+ in -0.1 volt increments.
ASSIGN  NAME=VB0     N.VALU=0.0
ASSIGN  NAME=VBSTEP  N.VALU=-0.1
ASSIGN  NAME=NBSTEP  N.VALU=11

$ Perform the base sweep.
CALL  FILE=bipgum0
```

Cutoff Frequency

In addition to calculating and plotting the Gummel characteristics and current gain of the transistor, *bipgum0* also estimates the cutoff frequency, F_t, as a function of collector current using the following approximation:

$$F_t = \frac{G_{cb}}{2\pi C_{bb}} \qquad \text{Equation A-1}$$

where

- G_{cb} is the collector-base conductance
- C_{bb} is the base capacitance

At each base bias, an AC small-signal solution is obtained at a fixed frequency (1 MHz) with the AC signal applied to the base.

CALCULATE statements are used to calculate F_t from the resulting conductance and capacitance values.

Appendix B: *Avant!* TCAD Graphics

Specifying the Graphics Device

The **DEVICE** parameter on **PLOT.1D**, **PLOT.2D**, and **PLOT.3D** statements specifies the graphics device name. This name determines which graphics device is used for graphical output. The value of the **DEVICE** parameter must match one of the valid graphics device names defined by the plot device definition file *mdpdev* (see Chapter 1, “Plot Device Definition File—mdpdev” on page 1-14). The valid graphics device names are described in “Graphics Devices” on page B-2.

Driver Subroutine

The driver subroutine used for graphical output is determined by comparing the value of the **DEVICE** parameter with the **NAME** entry in each line of the plot device definition file *mdpdev* (see “Plot Device Definition File” on page B-9). The driver used corresponds to the first line for which the **DEVICE** parameter most closely matches the **NAME** entry. If the value of the **DEVICE** parameter is “DEFAULT”, the driver used corresponds to the first line which contains an asterisk (*) as the first nonblank character.

The default graphics device defined by the file *mdpdev* can be changed by moving the asterisk to the beginning of another line in the file. Because the default graphics device may be redefined at your site, you should check the file *mdpdev*, rather than this Appendix, to determine the default graphics device.

The default value for the **DEVICE** parameter is “DEFAULT”. This default may be overridden by setting the *DEFPDEV* environment variable to an alternate graphics device name before executing Medici.

Graphics Devices

This section contains descriptions of the graphics devices and graphics software libraries supported by Medici.

Direct Device Drivers

The direct device drivers generate graphics by producing FORTRAN formatted output. The **LU** entry in the plot device definition file *mdpdev* (see "Plot Device Definition File" on page B-9) controls whether the driver output is sent to your terminal. Output is sent to the terminal if **LU** is 0. No terminal output is generated if **LU** is negative.

Driver Output

The driver output may also be sent simultaneously to an output file. Output is sent to the file specified by the **PLOT.OUT** parameter on **PLOT.1D**, **PLOT.2D**, and **PLOT.3D** statements if this parameter is specified. Otherwise, if the **DF** entry is "T" in the plot device definition file *mdpdev*, output is sent to the formatted plot output file <*base*>.*dplt* (see Chapter 1, "Formatted Plot Output Files" on page 1-10). The descriptions below indicate the default destination for the output produced by each direct device driver.

Available Drivers

The following direct device drivers are available:

TEK4010 (4010)

Tektronix 4010-series and compatible graphics terminals. This driver generates character sequences to directly control the device. It supports monochrome output. Output is sent to your terminal by default.

TEK4100 (4100)

Tektronix 4100-series and compatible graphics terminals. This driver generates character sequences to directly control the device. It supports 14-color output and filled polygons. Output is sent to your terminal by default.

HP7550 (7550, L/HP7550, L/7550)

Hewlett-Packard 7510, 7550, 7570, 7586, 7595, 7596, and compatible graphics plotters. This driver generates character sequences to directly control the plotter using HPGL and device control instructions. It supports 8-color output and filled polygons with landscape orientation. Output is sent to the formatted plot output file <*base*>.*dplt* by default.

P/HP7550 (P/7550)

Same as **HP7550** with portrait orientation.

HP7475 (7475, L/HP7475, L/7475)

Hewlett-Packard 7470, 7475, 7580, 7585, and compatible graphics plotters. This driver generates character sequences to directly control the plotter using HPGL and device control instructions. It supports 8-color output with landscape orientation. Output is sent to the formatted plot output file <*base*>.*dplt* by default.

P/HP7475 (P/7475)

Same as **HP7475** with portrait orientation.

POSTSCRIPT (P/POSTSCRIPT)

Adobe Systems Postscript graphics printers. This driver generates character sequences to directly control the device. It supports 14-level gray scale monochrome output and filled polygons with portrait orientation. Output is sent to the formatted plot output file <*base*>.*dplt* by default.

L/POSTSCRIPT Same as **POSTSCRIPT** with landscape orientation.

C/POSTSCRIPT (CP/POSTSCRIPT) Adobe Systems Postscript color graphics printers. This driver generates character sequences to directly control the device. It supports 14-color output and filled polygons with portrait orientation. Output is sent to the formatted plot output file <*base*>.*dplt* by default.

CL/POSTSCRIPT Same as **C/POSTSCRIPT** with landscape orientation.

XTERM X-Windows "xterm" graphics window. This driver generates character sequences to plot in the graphics window of the X-windows "xterm" terminal emulator program. It supports monochrome output. Output is sent to your terminal by default.

HP2648 (2648) Hewlett-Packard 2648 graphics terminals. This driver generates character sequences to directly control the device using binary relative and absolute formats. It supports monochrome output. Output is sent to your terminal by default.

HP2623 (2623) Hewlett-Packard 2623 and compatible graphics terminals. This driver generates character sequences to directly control the device using binary relative and absolute formats. It supports monochrome output. Output is sent to your terminal by default.

REGIS Digital Equipment ReGIS and compatible graphics terminals. This driver generates character sequences to directly control the device. It supports 8-color output with a dark background. Output is sent to your terminal by default.

I/REGIS Same as **REGIS** with a white background.

SELANAR Selanar HiRez and compatible graphics terminals. This driver generates character sequences to directly control the device. It supports monochrome output. Output is sent to your terminal by default.

CANON (P/CANON) Canon LBP-8II laser printer with full resolution. This driver generates character sequences to directly control the device. It supports monochrome output and filled polygons with portrait orientation and full resolution (requires 1536 Kbytes of RAM for full page output). Output is sent to the formatted plot output file <*base*>.*dplt* by default.

L/CANON Same as **CANON** with landscape orientation.

H/CANON (HP/CANON)

Canon LBP-8II laser printer with half resolution. This driver generates character sequences to directly control the device. It supports monochrome output and filled polygons with portrait orientation and half resolution (only requires 512 Kbytes of RAM for full page output). Output is sent to the formatted plot output file <*base*>.*dplt* by default.

HL/CANON Same as **H/CANON** with landscape orientation.

FORMAT Formatted plot output file. This driver generates one FORTRAN formatted line containing the subroutine arguments (*X, Y, IPEN*) for each call to the driver subroutine. The *X* and *Y* arguments are multiplied by **XPIX** and **YPIX**, respectively, obtained from the plot device definition file *mdpdev*. It supports 14-color output and filled polygons. Output is sent to the formatted plot output file <*base*>.*dplt* by default.

BINARY Binary plot output file. This driver generates one FORTRAN unformatted (binary) record containing the subroutine arguments (*X, Y, IPEN*) for each call to the driver subroutine. The *X* and *Y* arguments are written as real FORTRAN variables and the *IPEN* argument is written as an integer FORTRAN variable. It supports 14-color output and filled polygons. Output is sent to the binary plot output file <*base*>.*bplt* by default.

Graphics Software Library Drivers

The graphics software library drivers generate graphics output by performing FORTRAN subroutine calls to subroutines in graphics software libraries. These graphics software libraries are not provided by *Avant!* TCAD.

Enabling

A graphics software library driver is enabled for use by performing the following steps:

1. Replace all occurrences of "C*" with two spaces in the FORTRAN source code file for the driver.

   This enables calls to graphics software library subroutines and related FORTRAN statements. For example, the source code file TGPD19 must be modified to enable the **IGL/PLOT-10** driver.

2. Recompile the FORTRAN source code file for the driver.

3. Relink Medici with the graphics software library included in the link.

Output

Output may be generated on those graphics devices which are supported by the graphics software library. The supported graphics devices and the methods used to specify the output device and obtain output will depend on the specific graphics software library and the options chosen during its installation. Consult with your computer systems administrator for assistance in the use of the graphics software library drivers.

Available Drivers

The following graphics software library drivers are available:

CALCOMP **CALCOMP** graphics software library. This driver makes calls to library subroutines. It supports monochrome output.

TCS/PLOT-10 Tektronix PLOT 10 Terminal Control System (TCS) graphics software library. This driver makes calls to library subroutines. It supports 8-color output.

IGL/PLOT-10 Tektronix PLOT 10 Interactive Graphics Library (IGL) graphics software library. This driver makes calls to library subroutines. It supports 8-color output and filled polygons with a black background.

I/IGL/PLOT-10 Same as **IGL/PLOT-10** with a white background.

DI-3000 Precision Visuals DI-3000 graphics software library. This driver makes calls to library subroutines. It supports 8-color output and filled polygons with a normal background.

I/DI-3000 Same as DI-3000 with a complement-of-normal background.

DISSPLA Integrated Software Systems DISSPLA graphics software library. This driver makes calls to library subroutines. It supports 8-color output and filled polygons.

PLXY-11 Digital Equipment PLXY-11 graphics software library. This driver makes calls to library subroutines. It supports monochrome output.

GDDM IBM GDDM graphics software library for graphics terminals and auxiliary graphics plotters. This driver makes calls to library subroutines. It supports 8-color output and filled polygons.

GKS Graphics Kernel System (GKS) graphics software library for graphics terminals. This driver makes calls to library subroutines. It supports 8-color output and filled polygons.

Special Device Drivers

The special device drivers generate graphics output by using system features which are only available for some computer systems. Some or all of these drivers are not provided with versions of Medici for which the necessary system features are not available. The descriptions below indicate the computer systems for which each special device driver is available.

Available Drivers

The following special device drivers are available:

APOLLO Apollo GPR frame mode window. This driver generates character sequences and pipes them to the *tmaplot* program, which is executed by the driver. It supports 8-color output and filled polygons with a white background. The *tmaplot* program generates a GPR graphics window in which graphical output is produced. Multiple graphics images are retained. The display window can be controlled with the following keys:

- *downward vertical scroll*: display next page
- *upward vertical scroll*: display previous page
- *exit* or *abort*: exit

This driver and the *tmaplot* program are only provided with versions of Medici for the Unix operating system. An Apollo version of the *tmaplot* program must be executed on the local Apollo computer which controls the monitor display.

I/APOLLO Same as **APOLLO** with a black background.

SUN SunView graphics window. This driver generates character sequences and pipes them to the *tmaplot* program, which is executed by the driver. It supports 14-color output and filled polygons with a white background. The *tmaplot* program generates a SunView graphics window in which graphical output is produced. Multiple graphics images are retained. The display window can be controlled with the following keys and mouse buttons:

- *left mouse button*: display next page

- *right mouse button*: display previous page
- *z*: Zoom—magnify the image by a factor of 2 and center it at the cursor location
- *Z*: Unzoom—demagnify the image by a factor of 2 and center it at the cursor location
- *p*: Pan—center the image at the cursor location
- *r*: Reset—restore the image to its initial state
- *u*: Undo—undo the previous Zoom, Unzoom, Pan, or Reset operation
- *w*: print the cursor location in image coordinates
- *d*: use *replot* to convert the current image to a formatted plot file using the device specified by the *TMAPLOT_REPLOT* environment variable (the default is "FORMAT")
- *control-C*: exit

This driver and the *tmaplot* program are only provided with versions of Medici for the Unix operating system. A Sun version of the *tmaplot* program which supports SunView graphics must be executed on the local Sun computer which controls the monitor display.

I/SUN Same as **SUN** with a black background.

X X-Windows graphics window. This driver generates character sequences and pipes them to the *tmaplot* program, which is executed by the driver. It supports 14-color output and filled polygons with a white background. The *tmaplot* program generates a X-Windows graphics window in which graphical output is produced. Multiple graphics images are retained.

Display Window with Control Panel

If *tmaplot* is executed on a Sun Sparc system *and* the *TMAPLOT_XLIB* environment variable is not set, the display window is controlled through a control panel. Complete help information is provided as part of the user interface. The help information is only available if the directory where the file *studio_view.info* is located is included in the directory list set in the *HELPPATH* environment variable. The display window can also be controlled with the following keys:

- *f*: display next page
- *b*: display previous page
- *w*: print the cursor location in image coordinates
- *d*: use *replot* to convert the current image to a formatted plot file using the device specified by the *TMAPLOT_REPLOT* environment variable (the default is "FORMAT")

Display Window without Control Panel

If *tmaplot* is executed on any system other than a Sun Sparc *or* the *TMAPLOT_XLIB* environment variable is set, no control panel is available. The display window can be controlled with the following keys and mouse buttons:

- *left mouse button*: display next page
- *right mouse button*: display previous page
- *z*: Zoom—magnify the image by a factor of 2 and center it at the cursor location
- *Z*: Unzoom—demagnify the image by a factor of 2 and center it at the cursor location
- *p*: Pan—center the image at the cursor location
- *r*: Reset—restore the image to its initial state
- *u*: Undo—undo the previous Zoom, Unzoom, Pan, or Reset operation
- *w*: print the cursor location in image coordinates
- *d*: use *replot* to convert the current image to a formatted plot file using the device specified by the *TMAPLOT_REPLOT* environment variable (the default is "FORMAT")
- *control-C*: exit

This driver and the *tmaplot* program are only provided with versions of Medici for the Unix operating system.

I/X Same as **X** with a black background.

X60 Same as **X** with the addition of 41 shades of gray corresponding to color indices 20 through 60.

I/X60 Same as **X60** with a black background.

GPR Apollo GPR frame mode window. This driver makes calls to GPR library subroutines. It supports 8-color output and filled polygons with a white background. Multiple graphics images are retained. The display window can be controlled with the following keys:

- *downward vertical scroll*: display next page
- *upward vertical scroll*: display previous page
- *exit* or *abort*: exit

This driver is only provided with versions of Medici for Apollo computer systems.

I/GPR Same as **GPR** with a black background.

Plot Device Definition File

Medici contains 30 subroutines (TGPD01-TGPD30) which are used to generate graphical output. These subroutines either control graphics devices directly or provide indirect control through graphics software libraries.

mdpdev File

The plot device definition file *mdpdev* contains information which controls use of the driver subroutines. It associates each graphics device name with a driver subroutine through a unique driver subroutine index. See "Specifying the Graphics Device" on page B-1 for a description of how to specify the graphics device name.

Syntax

The plot device definition file *mdpdev* may contain blank lines which are ignored. Lines in the file which contain a slash (/) as the first nonblank character are ignored and can be used to document the file.

Graphics Device Characteristics

Lines in the plot definition file define the characteristics for a single graphics device and contain the following 11 columns of information (see the listing of *mdpdev* at the end of this section):

NAME—The character string defining the graphics device name.

The driver subroutine used for graphical output is the line in the file which exactly matches the graphics device name specified by the **DEVICE** parameter on the **PLOT.OUT** parameter on statements **PLOT.1D**, **PLOT.2D**, and **PLOT.3D**. The name may not contain blanks.

"Graphics Devices" on page B-2 contains a description of the graphics devices and graphics software libraries supported by Medici.

DV—The graphics driver subroutine index.

The value of **DV** determines which driver subroutine will be used for graphical output. Each driver subroutine is assigned a unique index (i.e. **DV**=1 corresponds to the subroutine TGPD01, **DV**=30 corresponds to the subroutine TGPD30). **DV** should be an integer in the range 1 to 30.

XSIZE—The horizontal size of the plotting area in centimeters.

The value of **XSIZE** should be the horizontal physical size of the output device to obtain correct character sizes and aspect ratios. **XSIZE** should be a positive floating point number containing a decimal point.

YSIZE—The vertical size of the plotting area in centimeters.

The value of **YSIZE** should be the vertical physical size of the output device to obtain correct character sizes and aspect ratios. **YSIZE** should be a positive floating point number containing a decimal point.

XPIX—The number of horizontal display pixels per centimeter.

The value of **XPIX** is only used for devices where display addressing is based on pixels. **XPIX** should be a non-negative floating point number containing a decimal point.

YPIX—The number of vertical display pixels per centimeter.

The value of **YPIX** is only used for devices where display addressing is based on pixels. **YPIX** should be a non-negative floating point number containing a decimal point.

LU—Controls whether graphics is output to a logical unit number or to the user's terminal.

- If **LU** is positive, it is the FORTRAN logical unit number to which the graphics output will be sent. Medici *does not* explicitly perform a FORTRAN OPEN statement to associate this logical unit number with a file, but it will be associated automatically with a file by some computer systems.
- If **LU** is 0, graphics output is directed to the user's terminal using logical unit number PLLUNM which is initialized in subroutine COMMN.
- If **LU** is negative, graphics output is not sent to a logical unit number or to the user's terminal, but may still be sent to <*base*>*.bplt* and <*base*>*.dplt*. **LU** should be an integer.

BF—Indicates whether output is to be sent to the binary plot output file <*base*>*.bplt* in addition to the output device. **BF** should be "T" for true or "F" for false.

If **BF** is "T", then output is sent to <*base*>*.bplt*. <*base*>*.bplt* is a FORTRAN unformatted (binary) file with each line containing the arguments *X* (real), *Y* (real), and *IPEN* (integer) for one call to the driver subroutine. The arguments *X* and *Y* are in units of centimeters.

This output is *not* in the same format as the output sent to the output device. The output to <*base*>*.bplt* is the standard output produced by the **BINARY** graphics device driver (**DV**=16). The value of **BF** is independent of the value of **DF**. Output may be sent to either, both, or none of the files <*base*>*.bplt* and <*base*>*.dplt*.

DF—Indicates whether output is to be sent to the formatted plot output file <*base*>*.dplt* in addition to the output device. **DF** should be "T" for true or "F" for false.

If **DF** is "T", then output is sent to <*base*>*.dplt*. <*base*>*.dplt* is a FORTRAN formatted file containing the character sequences which control the graphics device.

This file may be output to the graphics device to reproduce the graphical output. This can be accomplished with the *cat* command for Unix operating systems and with the *type* command for IBM/CMS and VAX/VMS operating systems.

Output has the following limitations and special characteristics:

- <*base*>.*dplt* is only available for the direct device drivers and some special device drivers (**DV**=1, 2, 3, 4, 5, 6, 8, 9, 10, 15, 28, 29, and 30)
- <*base*>.*dplt* is not available for the **BINARY** graphics device driver or the graphics software library drivers (**DV**=16, 17, 18, 19, 20, 21, 22, 23, 24)

The value of **DF** is independent of the value of **BF**. Output may be sent to either, both, or none of the files <*base*>.*bplt* and <*base*>.*dplt*.

SCRI controls up to 10 adjacent feature flags. Each flag should be "T" for true or "F" for false.

Currently, the following four feature flags, as identified by their column headings, are recognized:

- **S** (flag 1): "T" means characters are produced with built-in character fonts. This feature is available for three device drivers (**DV**=2, 3, 4).
- **C** (flag 2): "T" means color output is produced. If this flag is "F", monochrome output is produced. This feature is available for fifteen device drivers (**DV**=2, 3, 4, 8, 15, 16, 18, 19, 20, 21, 23, 24, 28, 29, 30).
- **R** (flag 3): "T" means output is rotated 90 degrees from the default orientation. This feature is useful for generating both portrait and landscape orientation output on hardcopy devices. Although this feature is available for all device drivers, it is useful mainly for three device drivers (**DV**=3, 4, 10).
- **I** (flag 4): "T" means the foreground and background colors are reversed (inverted). This feature is available for six device drivers (**DV**=8, 19, 20, 28, 29, 30).

MPLY—The maximum number of vertices supported by the device for the specification of filled polygons.

- Values of **MPLY** less than 9999 in *mdpdev* are maximum values which are determined by limitations imposed by graphics devices or by graphics software libraries.
- Values of **MPLY** less than 3 disable the filling of polygons.
- **MPLY** should be an integer less than 10000.

Defaults

The default graphics device is defined by the first line in the file *mdpdev* which contains an asterisk (*) as the first nonblank character. The default graphics device should normally be defined as the graphics device which is most commonly available to the users of Medici.

Multiple Lines and Entries

Multiple lines may be present in the file *mdpdev* for the same value of **DV** (e.g. **NAME**=**TEK4100** and **NAME**=**4100** both refer to **DV**=2) with different entries for

NAME, **XSIZE**, **YSIZE**, **XPIX**, **YPIX**, **LU**, **BF**, **DF**, **SCRI**, or **MPLY**. This allows a single driver subroutine to be used for multiple physical devices having different sizes, output characteristics, and methods of connection to the computer system.

Editing

You can modify the existing lines or add new lines to change the characteristics of the available graphics devices. The plot device definition file is in standard text format and can be modified by any text editor.

mdpdev Listing

The following is a listing of the plot device definition file *mdpdev*.

```
9040
/ The first line in this file is a revision code used to determine whether
/ the file format is consistent with the program.
/ Lines beginning with "/" represent comments and are ignored.
/ All other lines represent device definitions (one per line) containing
/ 11 fields, separated by spaces and/or tabs.
/
/ More information regarding this file is provided in the Appendix titled
/ "TCAD Graphics" in the User's Manual.
/
/ The default device can be identified by preceding a device definition
/ line with "*".  This device will be used if the value of the DEVICE
/ parameter in input statements is "DEFAULT". The DEVICE parameter is given
/ the value "DEFAULT" if it is not specified. Only one such default device
/ should be identified. If more than one default device is identified, a
/ warning message will be issued and the first default device will be used.
/
/ The fields in each device definition line have the following meanings:
/
/ NAME   character:device name specified with the DEVICE parameter in input
/ statements (abbreviations such as POST are recognized)
/ DV     integer: index of the device driver - each index corresponds to a
/       FORTRAN subroutine name containing the index (e.g. DV=8 for TGPD08)
/ XSIZE  float:  horizontal size of the graphics output in cm
/ YSIZE  float:  vertical size of the graphics output in cm
/        XSIZE and YSIZE should match the physical size of the display
/        device to achieve proper aspect ratios and character sizes
/ XPIX   float:  number of horizontal pixels per cm
/ YPIX   float:  number of vertical pixels per cm
/        XPIX and YPIX are used for devices which are addressed by pixel
/ LU     integer:  controls the destination for output of character
/        sequences which directly control the graphics device as follows:
/          =0:  output to terminal
/          <0:  no output to terminal
/          >0:  output to the logical unit specified by this field
/ BF     T or F:  T means a binary plot file is generated by default
/ DF     T or F:  T means a device plot file is generated by default
/        This file contains character sequences which control the device
/ SCRI   T or F:  up to 10 adjacent logical flags (each must be T or F)
/        The following four flags are currently recognized:
/          S (flag 1):  T means characters produced with built-in symbols
/          C (flag 2):  T means color is supported (F means monochrome)
/          R (flag 3):  T means output is rotated 90 degrees
/          I (flag 4):  T means foreground/background colors are inverted
/ MPLY   integer:  maximum number of vertices supported for filled polygons
/
/NAME             DV  XSIZE  YSIZE   XPIX   YPIX  LU BF DF  SCRI    MPLY
 TEK4010           1  23.95  17.96  171.0  171.0   0  F  F  FFFF       0
 4010              1  23.95  17.96  171.0  171.0   0  F  F  FFFF       0
 TEK4100           2  23.95  17.96  171.0  171.0   0  F  F  TTFF    9999
 4100              2  23.95  17.96  171.0  171.0   0  F  F  TTFF    9999
 HP7550            3  25.40  19.60  400.0  400.0  -1  F  T  TTFF     100
```

```
7550              3  25.40  19.60  400.0  400.0  -1  F  T  TTFF     100
L/HP7550          3  25.40  19.60  400.0  400.0  -1  F  T  TTFF     100
L/7550            3  25.40  19.60  400.0  400.0  -1  F  T  TTFF     100
P/HP7550          3  19.60  25.40  400.0  400.0  -1  F  T  TTTF     100
P/7550            3  19.60  25.40  400.0  400.0  -1  F  T  TTTF     100
HP7475            3  25.40  19.60  400.0  400.0  -1  F  T  FTFF       0
7475              3  25.40  19.60  400.0  400.0  -1  F  T  FTFF       0
L/HP7475          3  25.40  19.60  400.0  400.0  -1  F  T  FTFF       0
L/7475            3  25.40  19.60  400.0  400.0  -1  F  T  FTFF       0
P/HP7475          3  19.60  25.40  400.0  400.0  -1  F  T  FTTF       0
P/7475            3  19.60  25.40  400.0  400.0  -1  F  T  FTTF       0
POSTSCRIPT        4  19.05  25.40  118.1  118.1  -1  F  T  TFFF     100
P/POSTSCRIPT      4  19.05  25.40  118.1  118.1  -1  F  T  TFFF     100
L/POSTSCRIPT      4  25.40  19.05  118.1  118.1  -1  F  T  TFTF     100
C/POSTSCRIPT      4  19.05  25.40  118.1  118.1  -1  F  T  TTFF     100
CP/POSTSCRIPT     4  19.05  25.40  118.1  118.1  -1  F  T  TTFF     100
CL/POSTSCRIPT     4  25.40  19.05  118.1  118.1  -1  F  T  TTTF     100
XTERM             5  20.48  15.36  200.0  200.0   0  F  F  FFFF       0
HP2648            6  25.40  12.70  28.34  28.34   0  F  F  FFFF       0
2648              6  25.40  12.70  28.34  28.34   0  F  F  FFFF       0
HP2623            6  21.50  16.40  23.81  23.78   0  F  F  FFFF       0
2623              6  21.50  16.40  23.81  23.78   0  F  F  FFFF       0
NULL              7  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
REGIS             8  20.32  11.17   39.3   39.3   0  F  F  FTFF       0
I/REGIS           8  20.32  11.17   39.3   39.3   0  F  F  FTFT       0
SELANAR           9  21.56  16.42  190.0  190.0   0  F  F  FFFF       0
CANON            10  19.05  25.40  118.1  118.1  -1  F  T  FFFF      10
P/CANON          10  19.05  25.40  118.1  118.1  -1  F  T  FFFF      10
L/CANON          10  25.40  19.05  118.1  118.1  -1  F  T  FFTF      10
H/CANON          10  19.05  25.40  59.05  59.05  -1  F  T  FFFF      10
HP/CANON         10  19.05  25.40  59.05  59.05  -1  F  T  FFFF      10
HL/CANON         10  25.40  19.05  59.05  59.05  -1  F  T  FFTF      10
NULL             11  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
NULL             12  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
NULL             13  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
NULL             14  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
FORMAT           15  25.40  25.40  500.0  500.0  -1  F  T  FTFF    9999
BINARY           16  25.40  25.40    0.0    0.0   0  T  F  FTFF    9999
CALCOMP          17  25.40  27.94    0.0    0.0   0  F  F  FFFF       0
TCS/PLOT-10      18  25.40  19.50    0.0    0.0   0  F  F  FTFF       0
IGL/PLOT-10      19  25.40  19.50    0.0    0.0   0  F  F  FTFF     100
I/IGL/PLOT-10    19  25.40  19.50    0.0    0.0   0  F  F  FTFT     100
DI-3000          20  25.40  25.40    0.0    0.0   0  F  F  FTFF     100
I/DI-3000        20  25.40  25.40    0.0    0.0   0  F  F  FTFT     100
DISSPLA          21  25.40  25.40    0.0    0.0   0  F  F  FTFF     100
PLXY-11          22  25.40  27.94    0.0    0.0   0  F  F  FFFF       0
GDDM             23  25.40  25.40    0.0    0.0   0  F  F  FTFF    9999
GKS              24  25.40  25.40    0.0    0.0   0  F  F  FTFF     100
NULL             25  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
NULL             26  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
GPR              27  16.40  16.40   30.5   30.5   0  F  F  FTFF    9999
I/GPR            27  16.40  16.40   30.5   30.5   0  F  F  FTFT    9999
APOLLO           28  16.40  16.40   30.5   30.5  -1  F  F  FTFF    9999
I/APOLLO         28  16.40  16.40   30.5   30.5  -1  F  F  FTFT    9999
SUN              29  20.32  15.24   33.0   33.0  -1  F  F  FTFF    9999
I/SUN            29  20.32  15.24   33.0   33.0  -1  F  F  FTFT    9999
*X               30  20.32  15.24  500.0  500.0  -1  F  F  FTFF    9999
I/X              30  20.32  15.24  500.0  500.0  -1  F  F  FTFT    9999
X60              30  20.32  15.24  500.0  500.0  -1  F  F  FTFFFFFFFT 9999
I/X60            30  20.32  15.24  500.0  500.0  -1  F  F  FTFTFFFFFT 9999
```

Unique Operating System Specifications

This section details special specifications found in particular operating systems.

Non-Unix

For versions of Medici for operating systems other than Unix the lines for **DV**=27, **DV**=28, **DV**=29, and **DV**=30 are replaced with the following:

```
NULL       27  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
NULL       28  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
NULL       29  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
NULL       30  25.40  25.40    0.0    0.0   0  F  F  FFFF       0
```

Non-Apollo

For versions of Medici for computer systems other than Apollo the lines for **DV**=27 are replaced with the following:

```
NULL       27  25.40   25.40     0.0     0.0   0  F  F  FFFF       0
```

Default Graphics Device

The default graphics device is indicated here as **X** because that is the most standard device available for a wide variety of computer systems. For some computer systems another default graphics device, such as **TEK4010**, **GPR**, **APOLLO**, or **SUN**, may be defined by moving the asterisk (*) to the beginning of another line. Since the default graphics device may be modified at your site, you should check the file *mdpdev*, rather than this Appendix, to determine the default graphics device.

Appendix C: Version 2000.2 Enhancements

Enhancements to Medici Version 2000.2

Enhancements have been made to Medici version 2000.2 relative to Medici version 1999.4 that include:

- Implementation of a self-consistent direct tunneling model
- Addition of valence band electron tunneling to the direct tunneling model
- Implementation of the Lucent mobility model
- Implementation of a non-local impact ionization model
- New models for compound semiconductors
- An improved IC-CAP interface

These enhancements are described in the sections that follow.

Self-Consistent Direct Tunneling Model

The large gate current produced by direct tunneling through aggressively scaled gate oxides can alter both the on- and off- state characteristics of state-of-the art MOSFETs. As shown by example in Figure C-1, on-state characteristics are altered by the sourcing of electrons from the source and drain to supply the tunneling current. To accurately model these coupled phenomena, Medici now contains a self-consistent implementation of direct tunneling to compliment the post-processing model introduced in version 1999.4. The self-consistent model is solved as part of the electron and hole continuity equations and provides an efficient means of determining the impact of direct tunneling on terminal characteristics. The self-consistent direct tunneling model is activated by specifying the new **DT.CUR** parameter on the **MODELS** statement. A new parameter on the **METHOD**

statement, **DT.JACOB**, is also provided in order to turn the calculation of the jacobian terms for direct tunneling on and off. This can be used to speed up simulations under certain conditions. An example of using self-consistent direct tunneling is given in "Direct Tunneling Analysis in a N-MOSCAP" on page 4-39.

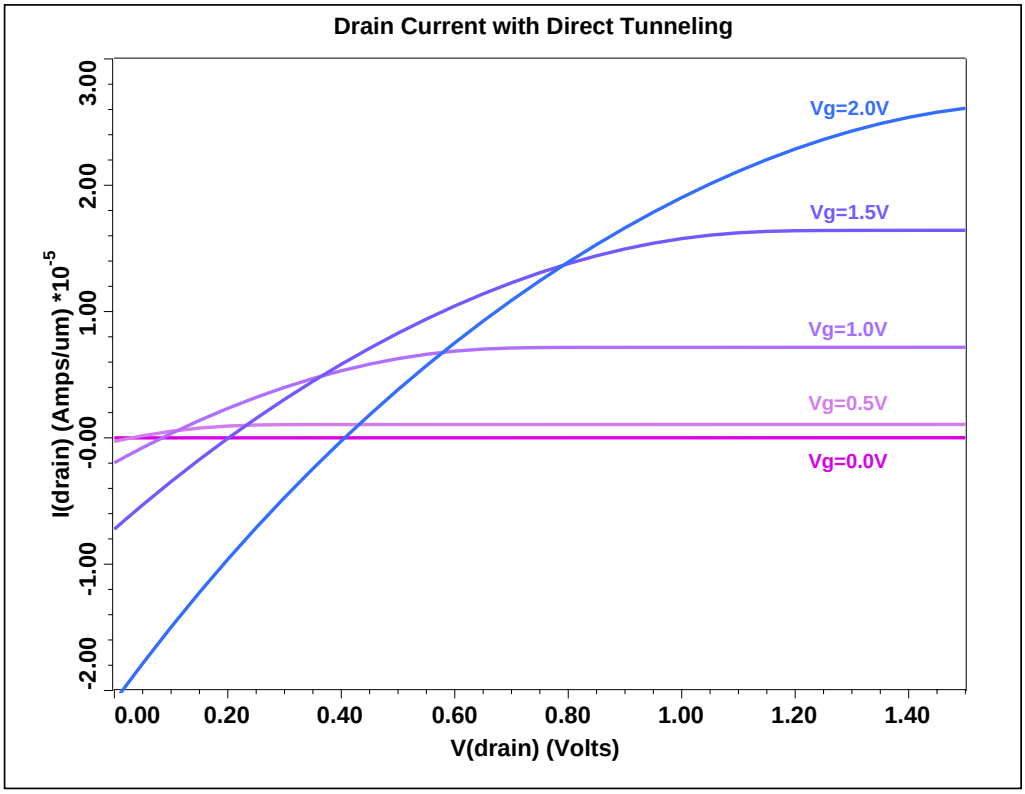


Figure C-1 Drain current in a long channel (10µm), thin gate (15Å) N-MOSFET computed using the self-consistent direct tunneling model. Large direct tunneling draws electrons out of the drain, drastically distorting the drain curves. After Momose [1].

Conduction and Valence Band Electron Tunneling

The previous version of Medici considered only conduction band electron tunneling (CBET) when modeling direct tunneling. For MOS devices with ultra-thin gate oxides, valence band electron tunneling (VBET) can also produce significant leakage current under certain bias conditions. VBET is due to electrons in the valence band of the substrate tunneling across the gate oxide into the conduction band of the gate. This is in contrast to CBET, in which electrons tunnel from the conduction band of the substrate. In VBET, the tunneling of electrons out of the substrate valence band generates free holes that are swept to the substrate contact. The measurement of the substrate current produced by VBET has been shown to be a promising technique for characterizing gate oxide thicknesses [2]. VBET can be activated as a post-process calculation by using the new **DT.VBET** parameter on the **SOLVE** statement. A similar parameter for activating conduction band electron tunneling, **DT.CBET**, is also provided. VBET and CBET can also be selectively

activated in a self-consistent calculation by specifying **DT.VBET** and **DT.CBET** on the **MODELS** statement in conjunction with **DT.CUR**. Two new material-dependent parameters, **MHH.DT** and **MLH.DT**, can be used to set the tunneling mass of the heavy and light hole bands, respectively.

As an example, Figure C-2 shows the dependence of the substrate current in a poly-gate N-MOSFET produced by VBET for various gate-oxide thicknesses. The current turns on at the applied bias necessary to drop the conduction band of the gate below the valence band of the substrate. At this point, electrons in the substrate valence band tunnel across the gate oxide into the conduction band of the gate, generating free holes that drift to the substrate contact. The current produced by VBET depends exponentially on the gate oxide thickness.

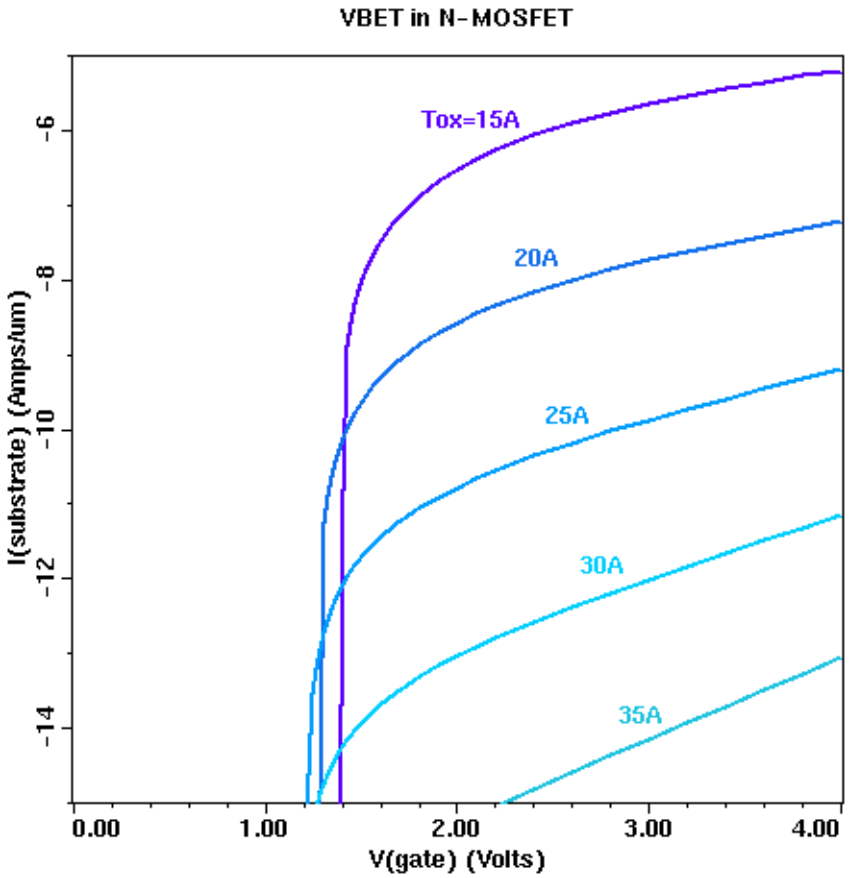


Figure C-2 Substrate current produced by valence band electron tunneling (VBET) in a poly-gate N-MOSFET for various gate-oxide thicknesses. The current turns on at the applied bias necessary to bring the conduction band of the poly gate below the valence band in the substrate.

Lucent Mobility Model

A mobility model developed by Darwish, et al. [3] that incorporates slightly modified versions of the Philips Unified Mobility model (**PHUMOB**) and the Lombardi Surface Mobility model (**LSMMOB**), as well as accounting for high field effects has been incorporated into Medici. The model is selected with the **LUCMOB** parameter on the **MODELS** statement.

For low longitudinal field, the carrier mobility is given by

$$\mu_S = \left[\frac{1}{\mu_b} + \frac{1}{\mu_{ac}} + \frac{1}{\mu_{sr}}\right]^{-1} \qquad \text{Equation C-1}$$

where

- μ_b is mobility in bulk silicon
- μ_{ac} is mobility degraded by surface acoustical phonon scattering
- μ_{sr} is mobility degraded by surface roughness scattering.

The equations for μ_b are very similar to those used for **PHUMOB**. For electron mobility we have

$$\frac{1}{\mu_{b,n}} = \frac{1}{\mu_{L,n}} + \frac{1}{\mu_{I,n}} \qquad \text{Equation C-2}$$

where

$$\mu_{L,n} = \mathbf{MMXN.UM}\left(\frac{T}{300}\right)^{-\mathbf{TETN.UM}} \qquad \text{Equation C-3}$$

$$\mu_{I,n} = \mu_{N,n}\left(\frac{N_{total}}{N_{I\mathrm{eff},n}}\right)\left(\frac{\mathbf{NRFN.UM}}{N_{total}}\right)^{\mathbf{ALPN.UM}} + \mu_{c,n}\left(\frac{n+p}{N_{I\mathrm{eff},n}}\right) \qquad \text{Equation C-4}$$

and $\mu_{N,n}$, $\mu_{c,n}$ and $N_{I\mathrm{eff},n}$ are given by

$$\mu_{N,n} = \frac{\mathbf{MMXN.UM}^2}{\mathbf{MMXN.UM} - \mathbf{MMNN.UM}}\left(\frac{T}{300}\right)^{3(\mathbf{ALPN.UM})-1.5} \qquad \text{Equation C-5}$$

$$\mu_{c,n} = \frac{\mathbf{MMXN.UM}\cdot\mathbf{MMNN.UM}}{\mathbf{MMXN.UM} - \mathbf{MMNN.UM}}\left(\frac{300}{T}\right)^{0.5} \qquad \text{Equation C-6}$$

$$N_{I\mathrm{eff},n} = N_D + N_A G(P_n)\,. \qquad \text{Equation C-7}$$

For holes, $N_{I\mathrm{eff},p}$ is given by

$$N_{Ieff,p} = N_A + N_D G(P_p)$$ Equation C-8

The functions $G(P_n)$ and $G(P_p)$ and the screening parameters P_n and P_p are described in "Philips Unified Mobility" on page 2-21.

Similar expressions hold for hole mobility. Note that the parameters used in the above expressions are the same as those used with **PHUMOB**.

The expressions for μ_{ac} and μ_{sr} for electrons and holes are very similar to those used with the **LSMMOB**:

$$\mu_{ac,n} = \frac{\mathbf{BN.LUC}}{E_{\perp,n}} + \frac{\mathbf{CN.LUC}\, N_{total}^{\mathbf{EXN4.LUC}}}{(T/300)^{\mathbf{KN.LUC}}\sqrt[3]{E_{\perp,n}}}$$ Equation C-9

$$\mu_{ac,p} = \frac{\mathbf{BP.LUC}}{E_{\perp,p}} + \frac{\mathbf{CP.LUC}\, N_{total}^{\mathbf{EXP4.LUC}}}{(T/300)^{\mathbf{KP.LUC}}\sqrt[3]{E_{\perp,p}}}$$ Equation C-10

$$\mu_{sr,n} = \left(\frac{\mathbf{DN.LUC}}{E_{\perp,n}^{\gamma_n}}\right)$$ Equation C-11

$$\mu_{sr,p} = \left(\frac{\mathbf{DP.LUC}}{E_{\perp,p}^{\gamma_p}}\right)$$ Equation C-12

where

$$\gamma_n = \mathbf{AN.LUC} + \frac{\mathbf{FN.LUC}(n+p)}{N_{total}^{\mathbf{EXN9.LUC}}}$$ Equation C-13

$$\gamma_p = \mathbf{AP.LUC} + \frac{\mathbf{FP.LUC}(n+p)}{N_{total}^{\mathbf{EXP9.LUC}}}$$ Equation C-14

Finally, the total mobility is obtained using the expression:

$$\mu_{n,p} = \frac{2\mu_{S,n,p}}{1+\left[1+\left(\frac{2\mu_{S,n,p}E_{\|,n,p}}{v_{n,p}^{sat}}\right)^2\right]^{1/2}}$$ Equation C-15

Note:

The longitudinal field mobility expression used with the Lucent mobility model (Equation C-15) can also be used as alternative to the Caughey-Thomas expression that is used by default when* FLDMOB *is specified on

the* MODELS *statement. This is accomplished by specifying* FLDMOB=3 *on the* MOBILITY *statement for each region or material where it is desired to use this model.

Default silicon parameters for **LUCMOB** are given in the following table:

Table 3-1 LUCMOB Default Parameters

Parameter	Default	Parameter	Default
AN.LUC	2.58	**AP.LUC**	2.18
BN.LUC	3.61×10^{7}	**BP.LUC**	1.51×10^{7}
CN.LUC	1.70×10^{4}	**CP.LUC**	4.18×10^{3}
DN.LUC	3.58×10^{18}	**DP.LUC**	4.10×10^{15}
FN.LUC	6.85×10^{-21}	**FP.LUC**	7.82×10^{-21}
KN.LUC	1.7	**KP.LUC**	0.9
EXN4.LUC	0.0233	**EXP4.LUC**	0.0119
EXN9.LUC	0.0767	**EXP9.LUC**	0.123

Non-Local Impact Ionization Model

To improve the accuracy of impact ionization calculations for deep submicron semiconductor devices, a non-local model has been implemented. The non-local model is based on the lucky electron model developed by Jungemann, et al. [4].

The model evaluation starts by tracking the potential drop along the current path at each point in the device. Impact ionization generation occurs only after the potential drop has reached a certain threshold. There are two generation models presented in [4], the hard threshold model and the soft threshold model. At present, only the hard threshold model is available. The soft threshold model will be available in the next release of the program.

To use the non-local impact ionization model, **IMPACT.I** should be specified on the **MODELS** statement. A new parameter, **II.NLOC**, is used to control whether the local or non-local model is invoked:

$$\text{II.NLOC} = \begin{cases} -1 & \text{, local model (default)} \\ 0 & \text{, post-processing non-local model} \\ 1 & \text{, self-consistent non-local model, no derivatives} \\ 2 & \text{, self-consistent non-local model, derivatives included} \end{cases}$$

The threshold energy for electron and hole impact ionization is controlled by the parameters **CN.IIGAP** and **CP.IIGAP**, which can be specified on the **MATERIAL** statement. The default values for these parameters are 1, which specifies that the bandgap energy is used as the threshold:

$$\varepsilon_{TH} = \begin{cases} \mathbf{CN.IIGAP} \cdot E_g & \text{, electrons} \\ \mathbf{CP.IIGAP} \cdot E_g & \text{, holes} \end{cases}$$

Despite the inclusion of non-local derivatives when **II.NLOC**=2 is used, the convergence can sometimes be very difficult near breakdown. Therefore, the user should approach the full-blown breakdown simulation with the non-local model with care.

New Models for Compound Semiconductors

Models for permittivity and the conduction and valence band density of states have been updated to include mole fraction dependence, enabling Medici to more accurately simulate the behavior of compound semiconductors, such as $Si_{1-x}Ge_x$. The new models are described below.

Permittivity

The dependence of permittivity on mole fraction is described by the expression

$$\varepsilon(x) = (\mathbf{PERMITTI} + \mathbf{EPS.X1} \cdot x + \mathbf{EPS.X2} \cdot x^2)\,\varepsilon_0 \qquad \text{Equation C-16}$$

where the parameters **PERMITTI**, **EPS.X1**, and **EPS.X2** can be specified on the **MATERIAL** statement and ε_0 is the vacuum permittivity. For backward compatibility, the default values of **EPS.X1** and **EPS.X2** are 0.

Density of States

The dependence of conduction and valence band density of states on mole fraction is given by the expressions

$$N_C(x, T) = N_C(T) \cdot \left[1 + \mathbf{NC.0}\left(\exp\left(\frac{-\mathbf{NC.E}\ x}{kT}\right) - 1\right)\right] \qquad \text{Equation 3-17}$$

$$N_V(x, T) = N_V(T) \cdot \left[1 + \mathbf{NV.0}\left(\exp\left(\frac{-\mathbf{NV.E}\ x}{kT}\right) - 1\right)\right] \qquad \text{Equation 3-18}$$

where the parameters **NC.0**, **NC.E**, **NV.0** and **NV.E** can be specified on the **MATERIAL** statement and $N_C(T)$ and $N_V(T)$ are described in "Bandgap and Effective Density of States" on page 2-8. For backward compatibility, the default values of the new parameters **NC.0**, **NC.E**, **NV.0** and **NV.E** are 0.

IC-CAP Interface Improvements

Medici interfaces to Hewlett Packard's IC-CAP parameter extraction program by converting data contained in Medici log files into IC-CAP data files. The following describes various improvements that have been made to the Medici IC-CAP interface.

Increased Number of Input/ Output Variables

The number of input variables allowed for Medici created IC-CAP data files has been increased from four to five. The number of output variables allowed for Medici created IC-CAP data files has been increased from three to four. The new parameters for these variables are **INP5** and **OUT4** on the **LOG** statement.

User Specified Minimum Current and Voltage Values

The user can now specify values for the minimum recognizable currents and voltages that Medici uses when examining data values in Medici log files. These values can be specified using the **I.MIN** and **V.MIN** parameters on the **LOG** statement. Currents and voltages with absolute value less than these values are considered to be zero. These values are also used as tolerances when comparing currents and voltages.

MDM Format for IC-CAP Data Files

Medici can now generate data files for Hewlett-Packard's IC-CAP program using the "Measured Data Management File Format." To create IC-CAP data files using this format, the **MDM** parameter should be specified on the **LOG** statement in addition to the **ICCAP** parameter.

An example of an IC-CAP data file in the **MDM** format is shown below:

```
!Data management file for Medici generated data
!"id vs vg"
!
BEGIN_HEADER
 ICCAP_INPUTS
  vg      V G GROUND LIN   1  0.00000E+00  2.00000E+00   11
  vb      V B GROUND CON  0.00000E+00
  vd      V D GROUND CON  1.00000E-01
  vs      V S GROUND CON  0.00000E+00
 ICCAP_OUTPUTS
  id      I D GROUND
END_HEADER

BEGIN_DB
 ICCAP_VAR vb        0.00000E+00
 ICCAP_VAR vd        1.00000E-01
 ICCAP_VAR vs        0.00000E+00

 #vg           id
   0.00000E+00  2.61303E-12
   2.00000E-01  6.54914E-10
   4.00000E-01  1.07913E-07
   6.00000E-01  9.79638E-07
   8.00000E-01  2.14085E-06
   1.00000E+00  3.25431E-06
   1.20000E+00  4.28818E-06
   1.40000E+00  5.24665E-06
   1.60000E+00  6.13876E-06
   1.80000E+00  6.97329E-06
   2.00000E+00  7.75782E-06
END_DB
```

LIST Sweep Type Available for Input Variables

By default, Medici will assume that the input variable sweep type is LIN and will look for *start*, *stop*, and *step* values consistent with the data in the log file. However, the user may now indicate that the sweep type is LIST for one or more of the input variables by specifying **LIST1**, **LIST2**, **LIST3**, **LIST4**, and/or **LIST5**.

Note:

The LIST parameters can only be used when writing IC-CAP data files using the* `MDM` *format.

Capacitance, Conductance, and Admittance in IC-CAP Data Files

Medici can now create IC-CAP data files that contain capacitance, conductance, and/or admittance data taken from Medici log files. This is accomplished simply by specifying the desired quantity with one of the output parameters on the **`LOG`** statement (e.g. **`OUT1`**="C(gate,drain)"). Capacitance, conductance, and admittance values can only be written to IC-CAP data files when using the **`MDM`** format.

S-Parameters

S-parameters can now be written to IC-CAP data files, provided that S-parameters were calculated by Medici during an AC small-signal analysis (the **`S.PARAM`** parameter on the **`SOLVE`** statement) and are contained in a Medici log file. This is accomplished by specifying the **`S.PARAM`** parameter instead of the output variables described above. In this case, the **`TERMINAL`** parameter is also required to specify which electrodes are terminal "1" and terminal "2". S-parameters are only allowed when writing data using the **`MDM`** format.

Bug Fixes in Medici Version 2000.2

The following miscellaneous bug fixes have been made to Medici version 2000.2 relative to version 1999.4.

Structure Related Fixes

- Fixed three problems related to the use of **OTHER** quantities that are read in or defined on the **PROFILE** statement:
  - **OTHER** quantities were not being read from insulator regions
  - Interpolation errors would sometimes occur due to roundoff problems
  - **OTHER** quantities were being read incorrectly from **1D.ASC** or **2D.ASC** files.
- Fixed a problem that would cause polysilicon regions to be converted to electrodes after a quadtree regrid.
- Corrected the bounds that the program uses for **ELIMINATE** statements in cases where the user specified bounds are outside the device structure.
- Fixed a problem occurring with periodic boundary conditions when there were only two vertical lines of nodes in the structure (that is, the left and right edges).

Model or Solution Related Fixes

- Fixed a problem that would sometimes produce an incorrect amount of photogeneration during transient simulations.
- Fixed a problem that would sometimes slow down the convergence in simulations involving energy balance equations and the impact ionization model.
- Changed the default evaluation method for direct tunneling to the WKB method (**DT.METH**=2).
- Fixed a problem that would sometimes cause the program to not recalculate gate current quantities for plotting after a change in bias conditions.
- Fixed a problem in the scaling of the post-processed Fowler-Nordheim tunneling current when used in cylindrical coordinates.
- Fixed a problem in the calculation of N_c and N_v used in setting the boundary condition on the potential at an electrode/insulator interface. This problem only appeared when the ambient temperature was changed from 300K and the exponents on N_c and N_v were changed from their default values.
- Fixed a problem that used the wrong temperature in the incomplete ionization model when solving for lattice temperature.

Input/Output Related Fixes

- Fixed a problem that would sometimes cause unusual results to occur when a TIF file was read that contained no material or model information.
- Fixed units of gate current reported by the simulator when in circuit mode.
- Fixed a bug with **PRINT** statement which prevents the printing of virtual node information in a rectangular mesh.

References

[1] H.S. Momose *et al*, "1.5 nm Direct-Tunneling Gate Oxide Si MOSFETs", IEEE TED 43, 1233 (1996).

[2] A. Shanware, J.P. Shiely, H.Z. Massoud, E. Vogel, K. Henson, A. Srivastava, C. Osburn, J.R. Hauser, and J.J. Wortman, "Extraction of the Gate Oxide Thickness of N- and P-Channel MOSFETs Below 20Å from the Substrate Current Resulting from Valence-Band Electron Tunneling," IEDM 1999 Technical Digest, p. 815 (1999).

[3] M.N. Darwish, J.L. Lentz, M.R. Pinto, P.M. Zeitzoff, T.J. Krutsick, and H.H. Vuong, "An Improved Electron and Hole Mobility Model for General Purpose Device Simulation," *IEEE Trans. on Electron Devices*, Vol. 44, No. 9, pp. 1529-1538, Sept., 1997.

[4] C.Jungemann, R.Thoma and W.L.Engl, "A Soft Threshold Lucky Electron Model for Efficient and Accurate Numerical Device Simulation," *Solid-State Electronics*, Vol. 39, No. 7, pp.1079-1086, 1996.

Appendix D: Technology Interchange Format (TIF)

Introduction

The TIF file format provides a framework for seamless and direct integration of heterogeneous TCAD tools. The unified file format provides the ability to store technology simulation data created by various process and device simulators in a single, well-defined and self-documenting form. This includes the following:

- Structure information
- Material properties and model parameters using a library of materials and models
- Mesh information
- Field information (process and device simulation results)
- General table data
- Quantities calculated during a simulation such as currents, voltages, peak temperature, peak electric field, etc.

Current Applications

The TIF format is currently used by *Avant!* TCAD process and device simulation tools. It is an open format specification, which can be used to implement bi-directional interfaces between *Avant!* TCAD's and non-*Avant!* TCAD tools and simulators. The format should be used when integrating new simulators into Taurus WorkBench.

Format

The TIF file consists of an arbitrary number of records. A record has the general form of:

<recname>	<parameters> CR

Where *recname* is a character string that identifies each record type. Generally, any number of records of a particular type can appear in one file. However, because records may point to each other, certain restrictions apply.

Informational Group

The informational group contains data identifying when and by whom the file was written, and is useful for describing the revision history of the file. This entry is also used by the simulators to determine that this is a TIF file and its revision.

h	TIF V1.2.0 <*char* name> <*char* file> <*char* date> <*char* other>
	char **name** Name of the program that created the file. *char* **file** Name of the input file (if any) to the creating program. *char* **date** Date the file was created. *char* **other** Other information

Record Types

A TIF file can contain a number of core records and some optional records. The core records are used by most of the tools/simulators and must be generated when creating a TIF file (see "Core Records" on page D-3). Optional records can be generated by certain tools to store tool-specific data (see "Optional Records" on page D-5). These records are ignored by tools which do not need the information contained in them. The core records are as follows:

Nodal coordinate	c
Edge record	e
Region record	r
Boundary record	b
Interface record	i
Interface boundary record	j
Triangle record	t
Solution record	s
Nodal solution record	n

Note:

Core records must be maintained in a consistent manner and written in the order indicated above.

All other records are considered optional. Optional records may be grouped together at the end of the TIF file. The order of the optional records with respect to each other and with respect to the core records should not matter.

A device may be thought of to consist of regions and interfaces. Electrodes are also represented as interfaces and regions. "Thin" electrodes are represented only as interfaces. "Thick" electrodes are represented as both regions and interfaces. Regions are in turn composed of triangles and boundary edges. Interfaces are composed of interface edges. Triangles are composed of three coordinate points.

Core Records

Nodal Coordinates Record

c	<*int* index> <*float* x-coord> <*float* y-coord> <*float* h>
	int **index** node index (sequential, starts at 1) *float* **x-coord** X-coordinate of node *float* **y-coord** Y-coordinate of node *float* **h** mesh spacing parameter associated with the node

Note:

Node numbers must be sequential, starting from 1 with no gaps.

Edge Record

e	<*int* index> <*int* point1> <*int* point2> <*int* bcode>
	int **index** edge index (sequential, starts at 1). *int* **point1** index of the starting coordinate node. *int* **point2** index of the ending coordinate node. *int* **bcode** has the value 2 for edges on exposed boundaries, and zero otherwise (important for TSUPREM-4).

Region Record

r	<*int* index> <*char* type> <*char* name>
	int **index** region index (sequential, starts at 1). *char* **type** material type (refers to the material symbol from the material database file. *char* **name** name of the region.

Boundary Record

b	<*int* edge>
	int **edge** index for edge on boundary of a region. The specified edge is part of the region started by a preceding region "r" record.

These entries *must* follow a region "r" entry.

Interface or Electrode Record

i	<*int* index> <*char* type> <*char* name> <*int* region>
	int **index** interface index number (sequential, starts at 1). *char* **type** Interface type (refers to the interface symbol from the material database file). Can specify an electrode. *char* **name** name of the interface. *int* **region** Region index if an entire region is the electrode. (Used for thick, i.e. regional electrodes.) if <=0, the interface is defined by following "j" entries.

Interface Edge Record

j	<*int* edge>
	int **edge** index for edge on an interface

This entry *must* follow an interface (i) entry if and only if the region entry on the preceding interface "i" entry is <=0.

Triangle Record

t	<*int* index> <*int* region> <*int* c1> <*int* c2> <*int* c3> <*int* t1> <*int* t2> <*int* t3>
	int **index** triangle index (sequential, starts at 1) *int* **region** region index of the region the triangle is part of *int* **c1** coordinate index of the triangle node 1 *int* **c2** coordinate index of the triangle node 2 *int* **c3** coordinate index of the triangle node 3 *int* **t1** triangle index of neighbor triangle opposite node c1 *int* **t2** triangle index of neighbor triangle opposite node c2 *int* **t3** triangle index of neighbor triangle opposite node c3

A code of -1024 is used instead of a neighbor triangle index for nodes opposite a reflecting boundary, and -1022 is used instead of a neighbor triangle index for nodes opposite an exposed boundary.

Fields Record (Solution Functions)

s	<*int* n> <*char* name1> <*char* name2> ... <*char* nameN>
	int **n** Number of solution variables to follow. *char* **name1-nameN** Names of the solution variables. Names of solution variables are defined in the solutions database.

These entries are used to store solution quantities which exist on the mesh. Examples are doping, potential, and electron concentrations.

Node Record

n	<*int* index> <*char* material> <*float* val1> ... <*float* valN>
	int **index** coordinate node index to which this node record belongs. *char* **material** material symbol from the materials database to which this node record belongs. Several node records can point to the same coordinate node if multiple material regions join at that node. *float* **val1-valN** values for the solutions at the nodes.

Optional Records

The general convention is that any tool using TIF can create additional records for its own use. Tools which encounter records unknown to them while reading a TIF file ignore those records. Only some of the most important optional records are documented in this specification.

Curve Data Records

The following data records are used for storing the terminal voltages and currents as well as calculated or derived quantities, device dimensions, etc. This information is typically created by a device simulator and used for visualization purposes.

ds	<*char* name>
	char **name** Name of the following data set.

di	<*char* name1> <*char* name2> ... <*char* nameN>
	char **name1-nameN** Names of the data items in the data set.

du	<*char* units1> <*char* units2> ... <*char* unitsN>
	char **units1-unitsN** Units of the data items in the data set.

dd	<*float* value1> <*float* value2> ... <*float* valueN>
	float **value1-valueN** Values of the data items.

Because some of these lines (di, du, dd) can become very long, the length is restricted to about 110 characters. Longer lines are continued over several statements. A line is continued by placing the continuation character "+" at its end.

Three-Dimensional Structure Records

3D extensions to TIF are currently under development and will be published at a later time.

TIF Example

The example TIF file shown below was created by Michelangelo, an interactive TIF editor and visualizer. Note that the example file contains a number of optional Michelangelo-specific records, such as Creator, File, Date, refine, dop2ing, viewport, and biasing. These records are ignored by other tools reading this file.

```
h TIF V1.2.0 TMA Michelangelo V 1.1.4 Fri Jul 22 18:54:35 199
Creator TMA Michelangelo V 1.1.4
File test.tif
Date Fri Jul 22 18:54:35 1994
refine 0 0 1 1  0.1 0.1 0.5 0
doping Asa 0 0 0.6 0.6  1e+20 0 0
viewport 0 1.05689 0 1 0.1 0.1 1
c 1 0.1 0.1 1
c 2 0.1 0.9 1
c 3 0.9 0.9 1
c 4 0.9 0.1 1
c 5 0.1 0.5 1
c 6 0.5 0.9 1
c 7 0.9 0.5 1
c 8 0.5 0.1 1
c 9 0.633333 0.5 1
e 1 1 5 0
e 2 2 6 0
e 3 3 7 0
e 4 4 8 2
e 5 5 2 0
e 6 6 3 0
e 7 7 4 0
e 8 8 1 2
r 1 Si Region1
b 5
b 1
b 8
b 4
b 7
b 3
b 6
b 2
biasing 1 0 0 0
t 1 1 3 6 9 4 3 -1024
t 2 1 1 8 5 6 -1024 -1022
t 3 1 9 7 3 -1024 1 8
t 4 1 9 6 5 5 6 1
t 5 1 2 5 6 4 -1024 -1024
t 6 1 5 8 9 7 4 2
t 7 1 8 4 9 8 6 -1022
t 8 1 7 9 4 7 -1024 3
s 3   Asa Net Total
n 3 Si  1.000000e-12 1.000000e-12 1.000000e-12
n 6 Si  1.000000e-12 1.000000e-12 1.000000e-12
n 9 Si  1.000000e-12 1.000000e-12 1.000000e-12
n 1 Si  1.000000e+20 1.000000e+20 1.000000e+20
n 8 Si  1.000000e+20 1.000000e+20 1.000000e+20
n 5 Si  1.000000e+20 1.000000e+20 1.000000e+20
n 7 Si  1.000000e-12 1.000000e-12 1.000000e-12
n 2 Si  1.000000e-12 1.000000e-12 1.000000e-12
n 4 Si  1.000000e-12 1.000000e-12 1.000000e-12
n 4 Ambient  0.000000e+00 0.000000e+00 0.000000e+00
n 8 Ambient  1.000000e+20 1.000000e+20 1.000000e+20
n 1 Ambient  1.000000e+20 1.000000e+20 1.000000e+20
```

Figure D-1 TIF file created by Michelangelo

Appendix E: Medici and STUDIO Command Editor

Introduction

Beginning with Medici V2.0, the use of arbitrary electrode names has been allowed. This makes it possible to refer to applied voltages and current using meaningful names that are representative of the simulated structure. For example, a MOSFET structure might specify biases similar to this:

```
SOLVE    V(Gate)=3.0  V(Drain)=5.0  V(Substrate)=-5.0
```

Restrictions

The additional flexibility that is now allowed in the Medici input language, however, imposes some restrictions when using the STUDIO Command Editor to create or modify Medici input files.

Bias or current specifications on the **SOLVE** statement using arbitrary electrode names can only be created or used when the Command Editor is in *text mode*. In *command mode*, a specification using an arbitrary electrode name will result in a **parameter name unknown** error. The remainder of this appendix describes the restrictions in more detail.

Creating Input Files with the STUDIO Command Editor

This section details creating input files with the STUDIO Command Editor. The following topics are discussed:

- Command mode
- Text mode
- **parameter name unknown** error message

Command Mode

When creating input files in *command mode*, you are restricted to using numbers for the electrode names for bias and current specifications on the **SOLVE** statement. This is consistent with the usage in versions of the program prior to V2.0. That is, only specifications such as the following will be allowed:

```
SOLVE    V1=3.0  V4=5.0  V2=-5.0
```

Text Mode

If it is necessary to refer to an electrode that has an arbitrary name on the **SOLVE** statement, this can be accomplished by first choosing **View as Text** from the **View** menu to switch the Command Editor into *text mode*. Once the program is in *text mode*, it is possible to edit the input file to enter the appropriate specification. For example, the following line could be created in *text mode*:

```
SOLVE    V(Whatever)=3.2  I(Dog)=1.2e-4  V(Poppyseed)=-3.8
```

Parameter Error Message

After making modifications to a **SOLVE** statement in *text mode*, it is possible to switch back to *command mode* to do further editing. When the switch is made, the Command Editor will issue an error message stating **parameter name unknown** for the various specifications involving electrode names (such as **V(Whatever)** in the above example). Although these parameters will not appear in *command mode*, they still actually exist as long as no editing is performed on the **SOLVE** statement that contains the parameter.

To see the parameters, you can either switch back to *text mode* or temporarily change the line containing it to a comment (by selecting **Comment** from the **Edit** menu).

Reading Existing Input Files into the STUDIO Command Editor

The restrictions described in the previous section also apply to existing input files that are loaded into the Command Editor.

Specifically, if an input file is loaded that contains **SOLVE** statements containing references to named electrodes, the Command Editor will issue error messages stating **parameter name unknown**.

As described in the previous section, the unknown parameters will not appear in *command mode*, but still in fact exist and can be seen by switching to *text mode*. See the previous section for more details regarding editing input files from the Command Editor.

If an input file is loaded and **SOLVE** statements in the input file only contain references to numbered electrodes (such as **V1**, **V3**, **I7**, etc.), then there are no limitations for editing this file from the Command Editor.

Appendix F: Limitations of IBM SP2 Versions of Medici

The IBM SP2 version of Medici allows you to take the advantage of the computational power offered by this parallel computer systems. Due to the enormous effort needed to make fully parallel a comprehensive device simulation program like Medici, there are certain features that are not yet available in the present IBM SP2 version of Medici.

Energy Balance and Lattice Temperature Equations

Only one such equation may be solved in the present IBM SP2 version of Medici. In addition, only the coupled Newton method is available for the solution method when such an equation is solved.

Flowline Plots

Specifying **`FLOWLINE`** in a **`CONTOUR`** statement is not available in the present IBM SP2 version of Medici.

Linear Matrix Solution in Medici

Iterative solutions are not available in the present IBM SP2 version of Medici.

Glossary

This glossary contains terms frequently used in the *Medici User's Manual.* Refer to the Index for more information about individual terms. A list of acronyms is included as the last section in the Glossary.

A

ABC mesh (Automatic Boundary Conforming) gridding algorithm capable of automatically creating a mesh in such a way that grid lines conform to region boundaries

abrupt heterojunction A heterojunction with abrupt change of the bandgap

acceptor scattering Mechanism of Coloumbic carrier scattering on ionized acceptors

acceptor states States in forbidden energy gap near the valence band

acoustic phonon scattering Mechanism of carrier scattering involving acoustic phonons

anisotropic mobility Direction-dependent mobility

B

back-lit When light enters from the back side of the wafer, away from the side with the contacts and diffusions

back surface electrode Electrode at bottom of the substrate

band-to-band recombination Direct recombination between valence and conductive bands

band-to-band tunneling Phenomenon of a valence band electron tunneling through the forbidden energy gap to the conduction band (leaving behind a hole)

Boltzmann statistics Carrier statistics for non-degenerate semiconductors

box method A discretization method where each equation is integrated over a small volume enclosing each node (Voronoi volume), yielding nonlinear algebraic equations for the unknown variables.

breakdown walk-out A phenomenon where the breakdown voltage of a MOSFET is increased with time due to the trapping of carriers within the oxide and at the interface of the device

bulk mobility Mobility of carriers in the bulk, away from surface without any surface scattering

C

carrier velocity overshoot Non-monotonic dependence of carrier velocity from the electric field

carrier-carrier scattering Scattering mechanism due to Coulombic interaction between the carriers

Cartesian coordinate system A coordinate system that utilizes perpendicular coordinate axes (x and y)

Caughey-Thomas expression Describes mobility reduction due to the component of electric field parallel to current flow

charged impurity scattering Carrier scattering mechanism due to Coulombic interaction with ionized impurity

Chynoweth law A model for impact ionization

continuation method A method where voltage-current steps are automatically selected to trace complex I-V curves

continuity equations Partial differential equations that govern the carrier concentrations for electrons and holes

Coulombic scattering Carrier scattering mechanism due to interaction of charged particles

current boundary condition A boundary condition where a fixed current is assigned to a terminal

current density Current per unit area

cylindrical coordinates A coordinate system that utilizes radial and height coordinate axes (r and z).

D

decoupled method A method where one set of variables is held fixed while another set is solved for

Dirichlet boundary condition A boundary condition where a constant value is assigned to a boundary

distributed resistance Resistance, distributed over the area of a contact

donor states States in forbidden energy gap near the conductive band

drift diffusion A standard approach for carrier transport simulation, assuming that the transport mechanisms are drift in the electric field and gradient-driven diffusion

E

electron-hole scattering Scattering mechanism due to Coloumbic interaction of electrons and holes

energy balance equations Device equations that govern the carrier temperatures for electrons and holes

F

fast traps A type of trap where the trap occupation function adjusts instantaneously to the carrier concentrations.

floating electrode An electrode in memory cells that is completely surrounded by insulating material

floating region A semiconductor region which is not connected to any electrodes

Fowler-Nordheim tunneling A mechanism of field-driven carrier tunneling through insulators

G

Gaussian (or Gauss) elimination Algorithm for eliminating unknowns when using a direct method for the solution of a linear system of equations

graded heterojunction A heterojunction with a gradual change of the bandgap

Gummel's method Sequential (decoupled) solution of a system of equations

H

hot carrier injection Injection of energetic (hot) carriers into insulators

I

impact ionization A mechanism of generation of electrons and holes by ionizing an atom due to impact by an energetic carrier

impurity freeze-out A mechanism of impurity deactivation due to reduction of the carriers at low temperature

impurity profiles Distribution of acceptor and donor impurities in semiconductors

intrinsic carrier concentration Carrier concentration in an undoped semiconductor

intrinsic Fermi energy Fermi level in an undoped semiconductors

ionization integral An integral involving impact ionization coefficients along an electric field line, which is used to estimate the amount of carrier multiplication occurring in a device

isothermal simulation Simulation under the assumption of constant temperature throughout the device

J

Jacobian matrix A matrix containing derivatives of the device equations with respect to the device variables at each node of the structure

L

lattice heat equation Heat transfer equation for the semiconductor lattice temperature

lattice scattering Carrier scattering mechanism with the lattice atoms

M

majority carrier concentration Concentration of electrons in n-type regions and holes in p-type regions

mesh aspect ratio Horizontal grid spacing divided by the vertical grid spacing.

minority carrier concentration Concentration of electrons in p-type regions and holes in n-type regions

N

Neumann boundary condition A boundary condition imposed on flux

Newton damping An approach to improve convergence of the Newton iterations

Newton's method Simultaneous (coupled) solution of a system of equations

Newton-Raphson method A modification of Newton's method with fewer updates of the Jacobian matrix to speed up the solution

O

Ohmic contacts Contacts that are implemented as simple Dirichlet boundary conditions

P

permittivity tensor Representation of permittivity for an anisotropic material

preconditioning Process of transforming a matrix to an approximate identity matrix.

Q

quadtree mesh generation The quadtree mesh used by Medici is generated and refined using an external mesh generator. This mesh generator makes use of two support files in addition to the primary mesh file used by Medici.

S

saturated drift velocity Carrier velocity at high parallel electric fields

Schottky contacts Contacts defined by a work function of the electrode metal and an optional surface recombination velocity.

thermal runaway A usually destructive phenomena caused by lattice heating

transverse field effects Mobility reduction due to high transverse fields

trap energy level Energy level of a trap in the bandgap

trap-assisted tunneling Tunneling with intermediate trapping

V

virtual nodes A separate data structure associated with a node at heterojunctions. The program accounts separately for the different properties associated with different materials that meet at the same point in space.

Acronyms

ABC	Automatic Boundary Conforming Mesh
AM-AAM	Anisotropic Material Advanced Application Module
BDF	Backward Differentiation Formula
Bi-CGSTAB	Bi-Conjugate Gradient stabilized Method
BJT	Bipolar Junction Transistor
CA-AAM	Circuit Analysis Advance Application Module
CCD	Charge Coupled Device
CMOS	Complementary Metal-Oxide Semiconductor
DOS	Density of States
DRAM	Dynamic Random Access Memory
EEPROM	Electrically Erasable Programmable Read Only Memory
EPROM	Erasable Programmable ROM
ESD	Electrostatic Discharge
HBT	Heterojunction Bipolar Transistor
HD-AAM	Heterojunction Device Advanced Application Module
HEMT	High Electron Mobility Transistor
ICCG	Incomplete Cholesky Conjugate Gradients
ILUCGS	Incomplete LU Conjugate Gradient Square Method
JFET	Junction Field-Effect Transistors
LDD	Lightly Doped Drain
LT-AAM	Lattice Temperature Advanced Application Module
MESFET	Metal-Semiconductor Field-Effect Transistor
MOSFET	Metal-Oxide Semiconductor Field-Effect Transistor
N-FET	N-Field Effect Transistor
NMOS	N-channel MOSFET
OD-AAM	Optical Device Advanced Application Module
PD-AAM	Programmable Device Advanced Application Module
PDE	Partial Differential Equation
PMOS	P-channel MOSFET

SCR	Silicon Controlled Rectifier
SEU	Single-Event Upset
SOI	Silicon On Insulator
SOR	Successive Over-Relaxation
SRAM	Static Random Access Memory
SRH	Shockley-Read-Hall Recombination
TC-AAM	Trapped Charge Advanced Application Module
TFT	Thin Film Transistor

Index

Symbols

Numerics

A

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y