INTEGRATED CIRCUITS LABORATORY

STANFORD ELECTRONICS LABORATORIES
DEPARTMENT OF ELECTRICAL ENGINEERING
STANFORD UNIVERSITY · STANFORD, CA 94305

SUPREM-IV USERS MANUAL

By

Mark E. Law

Conor S. Rafferty

Robert W. Dutton

December 1988

Prepared under

Defense Advanced Projects Agency Contract DAAL 01-86-K0101 and

Semiconductor Research Corporation Contract SRC 85-08-062

SUPREM-IV USERS MANUAL

By

Mark E. Law

Conor S. Rafferty

Robert W. Dutton

December 1988

Prepared under

Defense Advanced Projects Agency Contract DAAL 01-86-K0101 and

Semiconductor Research Corporation Contract SRC 85-08-062

Integrated Circuits Laboratory

Stanford Electronics Laboratories

Stanford University, Stanford, California

SUPREM IV

Level: Shell

Author: Mark E. Law

Version: 3.4 11/21/88 13:17:52

DESCRIPTION

This section describes the use of the manual and the conventions used to denote the parameters and choices associated with each statement. The program accepts input in a fashion similar to SUPREM III and PISCES II. The input is made up of a series of statements. It is important to bear in mind that the parser is case sensitive. FOO is not the same as foo.

Each statement can have parameters which are of the form:

parameter_name = value

Value is either a real number expression or a character string. There are some parameters that are booleans. Booleans recognize true and false as the valid forms of values. If the value is left off a boolean parameter, it is set to true.

The manual describes these parameters in the following form:

param = <n> .... a real valued parameter
param = <c> .... a string valued parameter
param .... a boolean parameter

Choices are denoted by putting the parameters in parenthesis and separating them by vertical bars.

(par1 | par2)

This states that either par1 or par2 may be specified but not both.

Anything enclosed in brackets '[ ]' are optional parameters and aren't necessary. Almost everything in the program has some sort of default value, so almost all parameters are optional. In case of doubt, never rely on the defaults.

Floating point parameters can be specified as expressions involving numbers, numerical constants, the operators +,-,*,/,^, and the functions listed below.

Functions

abs	absolute value
active	active portion of the specified dopant
erf	error function
erfc	complimentary error function
exp	exponential
gradx	computes the rough gradient in the x direction
grady	computes the rough gradient in the y direction
log	logarithm
log10	logarithm base 10
mat1@mat2	returns the y value of the interface between mat1 and mat2 along a vertical slice at the given location.
scale	scales the value given by the maximum value
sqrt	square root
xfn	takes y and z and finds an x to match
yfn	takes x and z and finds an y to match
zfn	takes x and y and finds an z to match

If an expression contains spaces, it should be enclosed in parenthesis.

EXAMPLES

Par1=<n>

Par1 is a required numeric valued option.

Par1=(4.0 * exp(-2.0 / (8.62e-5 * 1173.0)))

Par1 is a required numeric valued option, assigned a real number expression.

[Par2=<c>]
Par2 is an optional character variable.

BUGS

Bugs in the documentation?

SEE ALSO

everything(), select(2) for how to choose a z variable, printf(2) for further examples of expressions

SUPREM IV

Level: Shell

Author: Mark E. Law

Version: 3.4 11/21/88 13:22:58

DESCRIPTION

The shell is the user interface to the SUPREM IV program. The shell was designed to be similar to CSH and allow some of the same actions. The shell is a command line interpreter. It reads commands from the user and executes the commands with the arguments listed.

The shell executes commands from the system wide model file to set up coefficients. It then executes commands from the file .supremrc before providing a user prompt. This allows start up commands to be executed before the session begins. The .supremrc file is looked for in the current directory and the user's home directory.

COMMANDS

A simple command is a sequence of words, the first of which specifies the command to be executed. Sequences of commands may be separated by semicolons (;), and are then executed sequentially. A command may be executed in background mode by following the last entry with an ampersand (&) character.

Any sequence may be placed in braces { } to form a simple command.

Built-In Commands

Built-in commands are executed within the shell. The build-in commands are:

define

define "name(args) wordlist"

The first form prints all aliases. The final form assigns the specified wordlist as the macro of name. Every occurrence of name in the input stream is replaced by the wordlist associated with it. Optionally, a definition can contain arguments which will be mapped when the macro is substituted. Name will only be substituted for when is separated by blank space or shell special characters. For more information on macros and arguments, see define(1).

foreach name (val1, val2, val3) commands end

Name is set to each of the values contained in the parenthesis consecutively and the commands are executed. This allows a command to be run several times with different parameters. The name is substituted in the commands using the macro features. See for(1).

help command

Help without any arguments will list all of the available commands. The help "command" will give information on the listed command name. The help command provides a list of parameters and short description of each of the parameters.

set "noexecute | prompt"

This command allows the parameters listed to be changed. Interactive and noexecute are described in the non-built-in command section below. Prompt is the string which appears when the shell is ready to accept input. Set with no input lists the current settings.

source "filename"

The source command allows the file filename to be opened and the contents of the file are placed in the input stream of the shell to be parsed.

undef "name"

This command allows name to be removed from the macro table.

unset "noexecute"

This command allows the noexecute modes to be turned off.

Non-Built-In Command Execution

If a command is entered that is not a built-in, the program checks its parse tables for that command. If the command is found, that parameters are passed to the command for execution.

If the noexecution flag is set, the command will execute only checks on the parameters to see if they are legal. It will exit without performing any action other that the check. This mode is useful for debugging long scripts of input.

Macro Substitution

A command line is scanned, it is parsed into distinct words and each word is checked to see if it is in the macro table. If it is, then the text which is the macro for that command is placed in the input stream. To disable macro substitution, the % character can be used at the beginning of the line. That line will not be macro expanded. Selective macro evaluation can be achieved by disabling macros (with %) and putting $ before the names which are desired to be evaluated. That is, use $name (or ${name} in cases of ambiguity). The intent is to provide shell variables in a simple way.

If a macro is found, substitution is performed of the macro body for the macro name. Any arguments are paired up and they are substituted as well. Macro arguments may not be other macros.

Command Line Parsing

The shell splits input lines into words at blanks and tabs. The following exceptions (parser metacharacters) are considered separate words:

&	ampersand;
;	semicolon;
>	greater-than sign;
{	left brace;
}	right brace;
%	percent sign.

These characters have special meaning to the shell and are described elsewhere.

Input/Output

The standard output of a command may be redirected with the following syntax: > *name*. The file *name* is used as standard output. If the file does not exist then it is created; if the file exists, it is appended to. The entire output of the sequence redirected is sent to the file.

Background

Any sequence of commands may be backgrounded by using the & character. These commands will be executed using a fork and exec call. They will work on the current state of the data space but will not modify the copy you have. The copy will begin execution immediately and return a prompt to the user without waiting for the copy to finish.

EXAMPLES

foo

execute command foo with no parameters.

foo val=1.0 file=name

execute command foo with parameter string val=1.0 file=name.

foo; bar

execute command foo followed by command bar

foo > name

execute command foo and sends it output into file name.

{foo; bar} > name

execute commands foo and bar and send their output to file name.

foo &

execute command foo in the background.

BUGS

Numeric expressions are handled by an *ad hoc* parser, separate and different from the shell yacc parser. If a complicated expression gives unexpected results, parenthesize.

SEE ALSO

everything()

DEFINE

define – define strings for command line substitution

SYNOPSIS

define [macro_name macro_body]

SUPREM IV

Level: Shell Built In

Author: Mark E. Law

Version: 3.2 9/28/88

DESCRIPTION

The define command can be used for command line substitution. The name macro_name will be entered into a table with the macro body. Any time the macro_name appears on a command line as a separate token, the macro_body is substituted. The macro_name can also appear following a '$' which will force macro substition. To concatenate a macro with another string, ${macro_name} should be used. This allows long often used sequences to be entered by the user once. It is is similar to CSH aliasing, except that it takes place anywhere on a command line.

Define without any arguments will list the current definitions. To undefine a name, use the undef command. If the user desires to turn off all macros in a line, the % character can be used. A % will turn off macro expansion from the % character to the end of a line.

EXAMPLES

%define bounds xmin=0.0 xmax=5.0 ymin=0.0 ymax=20.0

This define will substitute all future occurrences of the string bounds with the list of mins and maxs.

%define q quit

This will allow the user to exit suprem with a q instead of typing the quit command.

echo ${q} $q q

If q has been defined as a macro equil to quit, all three instance will be expanded, and the output will look like:

quit quit quit

BUGS

The expansion routine does not recursively expand macros. It finds a token, and expands it once and then moves on. For example,

```
%define foo bar
%define froboz foo
froboz
```

will expand froboz to foo and no further.

SEE ALSO

undef()

FOR

foreach – command looping facility

SYNOPSIS

[foreach | for] name (list)
commands
end

SUPREM IV

Level: Shell Built in

Author: Mark E. Law

Version: %I 9/28/88

DESCRIPTION

The for command allows the user to specify loops in the input. Either for or foreach is acceptable. Name will take on the values in the list consecutively until the there are no values left. The body will be executed once for each value in the list. Values in the list can be separated by commas or spaces. Name is set to the value in the list using the define feature of the shell. Replacement of name in the body of the commands is done using the macro features that are part of the normal shell.

The list is a set of strings separated by commas or spaces. It can also be one of the following numerical operators:

start to end step val

where start is a numerical start value, end is the last value, and val is the size of step to take between them.

EXAMPLES

```
foreach string ( hello, goodbye, aufwiedersehn, goodnight )
echo string
end
```

The command echo string will be executed four times. String will be set to the values "hello", "goodbye", "aufwiedersehn", and "goodnight" consecutively. This will produce output that will look something like:

```
hello
goodbye
aufwiedersehn
goodnight
```

```
foreach val ( 1.0 to 10.0 step 0.5 )
echo val
end
```

This will increment val from 1.0 to 10.0 in steps of 0.5. The inner body of the loop will be executed 19 times.

BUGS

Any bugs in the define code and macro handlers will probably show up here.

SEE ALSO

define

HELP

help – online quick info facility

SYNOPSIS

help [command_name]

SUPREM IV

Level: SHELL

Author: Mark E. Law

Version: 1.1 11/21/88

DESCRIPTION

Help examines the command name and lists the parameters of the specified command and short description of each. This is useful to quickly jog the memory of the articular parameter name desired for a command. There is no extended description given. If no command name is given, the help command lists all known commands.

BUGS

Some command parameters do not have a description. Also, the output should be piped through more ala the man command.

SEE ALSO

unix command - man(1)

MAN

man – online help facility for SUPREM IV

SYNOPSIS

man [command_name]

SUPREM IV

Level: SHELL

Author: Mark E. Law

Version: 3.2 9/28/88

DESCRIPTION

Man examines the command name and attempts to find a manual page for that command. The manual pages are piped through more for user viewing.

Manual pages live in the man subdirectory of the home directory of SUPREM IV. The first location checked for manual pages is that given by the environment variable MANDIR. If this fails, it checks in a default location defined in the program. The system administrator should have set this for your system.

BUGS

The pipe to the more(1) program is slow, ways to speed this and still make use of the more program need to be investigated. The man command also needs to have the exact name typed out or no help will be printed. This behavior is a little restrictive.

SEE ALSO

unix command - more(1)

SET

set -- set various shell parameters

SYNOPSIS

set [(interactive | noexecute | prompt=<c>]

SUPREM IV

Level: Shell Built In

Author: Mark E. Law

Version: 3.2 9/28/88 16:48:57

DESCRIPTION

This command allows the user to turn on several useful shell parameters. The command unset allows the same parameters to be turned off.

interactive

The interactive option automatically invokes the interactive command editor on every command if set true. Any command line parameters are used as defaults for the table editor. The command editor does not exist yet. Therefore this variable should be off.

noexecute

The noexecute flag will put all entered commands into a check only mode. If this flag is on, commands will only check for legality of arguments and will no take any action on the arguments.

prompt The prompt parameter takes the remainder of the line and uses it as the prompt for the rest of the session. The default prompt is SUPREM4>.

BUGS

The boolean variables really ought to understand a parameter so they can be set on or off.

The parameters all have to be typed in full. The parser does not understand shortened versions.

SEE ALSO

unset()

SHELL

shell – command line interpreter

SYNOPSIS

shell

SUPREM IV

Level: Shell

Author: Mark E. Law

Version: 3.1 9/14/87 11:04:20

DESCRIPTION

The shell is the user interface to the SUPREM IV program. The shell was designed to be similiar to CSH and allow some of the same actions. The shell is a command line interpreter. It reads commands from the user and executes the commands with the arguments listed.

The shell executes commands from the file .supremrc before providing a user prompt. This allows start up commands to be executed before the session begins. The .supremrc file is looked for in the current directory.

COMMANDS

A simple command is a sequence of words, the first of which specifies the command to be executed. Sequences of commands may be separated by semicolons (;), and are then executed sequentially. A command may be executed in background mode by following the last entry with an ampersand (&) character.

Any sequence may be placed in braces { } to form a simple command.

Built-In Commands

Built-in commands are executed within the shell. The build-in commands are:

define

define "name(args) wordlist"

The first form prints all aliases. The final form assigns the specified wordlist as the macro of name. Every occurence of name in the input stream is replaced by the wordlist associated with it. Optionally, a definition can contain arguments which will be mapped when the macro is subsituted. Name will only be subsituted for when is sepacrated by blank space or shell special characters. For more information on macros and arguments, see define(1).

foreach name (val1, val2, val3) commands end

Name is set to each of the values contained in the parenthesis consecutively and the commands are executed. This allows a command to be run several times with different parameters. The name is substituted in the commands using the macro features.

set

set "interactive | noexecute | prompt"

This command allows the parameters listed to be chenged. Interactive and noexecute are described in the non built in command section below. Prompt is the string which appears when the shell is ready to accpet input. Set with no input lists the current settings.

source "filename"

The source command allows the file filename to be opened and the contents of the file are placed in the input stream of the shell to be parsed.

undef "name"

This command allows name to be removed from the macro table.

unset "interactive | noexecute"

This command allows these modes to be turned off.

Non-Built-In Command Execution

If a command is entered that is not a built in, the program checks its parse tables for that command. If the command is found, that parameters are passed to the command for execution.

If the interactive flag is set or there were no parameters given, the program executes the command in interactive mode. In interactive mode, the user is presented with a table form of the command parameters to be edited.

If noexecution flag is set, the command will execute only checks on the parameters to see if they are legal. It will exit without performing any action other that the check. This mode is useful for debugging long scripts of input.

Macro Substitution

A command line is scanned, it is parsed into distinct words and each word is checked to see if it is in the macro table. If it is, then the text which is the macro for that command is placed in the input stream. Macro substition stops on any line passed a % (percent sign).

If a macro is found, substitution is performed of the macro body for the macro name. Any arguments are paired up and they are substituted as well. Macro arguments may not be other macros.

Command Line Parsing

The shell splits input lines into words at blanks and tabs. The following exceptions (parser metacharacters) are considered separate words:

&	ampersand;
;	semicolon;
>	greater-than sign;
{	left brace;
}	right brace;
%	percent sign.

These characters have special meaning to the shell and are described elsewhere.

Input/Output

The standard output of a command may be redirected with the following syntax: > *name* The file name is used as standard output. If the file does not exist then it is created; if the file exists, it is truncated, and its previous contents are lost. The entire output of the sequence redirected is sent to the file.

Background

Any sequence of commands may be backgrounded by using the & character. These commands will be executed using a fork and exec call. The will work on the current state of the data space but will not modify the copy you have. The copy will begin execution immediately and return a prompt to the user without waiting for the copy to finish.

EXAMPLES

foo

excute command foo with no parameters.

foo val=1.0 file=name

execute command foo with parameter string val=1.0 file=name.

foo; bar

execute command foo followed by command bar

foo > name

execute command foo and sends it output into file name.

{foo; bar} > name

execute command foo and bar and send their output to file name.

foo &

execute command foo in the background.

BUGS

Immense number.

SEE ALSO

everything()

SOURCE

source – execute commands from the specified file

SYNOPSIS

source filename

SUPREM IV

Level: Shell Built In

Author: Mark E. Law

Version: 3.3 11/21/88 13:14:44

DESCRIPTION

The source command allows users to read commands from a command file. Commands are read from the file until an end of file mark is found. This is especially useful for backgrounding large groups of commands.

The source command places the file in the current input stream. Sources can be nested until the limit of open file descriptors (system dependent ~ 20). Several files are sourced at the beginning of the session. The first is a system wide model file which contains defaults for the coefficients. The second and third are the home directory file by the name of .supremrc and a local directory .supremrc. These files allow the user to specify his own start up information and defaults.

BUGS

No known bugs.

UNDEF

undef – undefine previously defined macros

SYNOPSIS

undef [macro_name]

SUPREM IV

Level: Shell Built In

Author: Mark E. Law

Version: 3.2 9/28/88 16:50:59

DESCRIPTION

The undef command can be used to turn off previously defined macro. Macro_name and its expansion are deleted from the macro table. This command is similar to the CSH unalias command.

BUGS

The command is open to macro substitution. The sequence:

```
define macro this is a macro
undef macro
```

will attempt to undefine the macro "this". To be able to undefine any macro, the % must be the first character on the line.

SEE ALSO

define()

UNSET

unset -- unset various shell parameters

SYNOPSIS

unset [(interactive | noexecute)]

SUPREM IV

Level: Shell Built In

Author: Mark E. Law

Version: 9/28/88 3.2 16:50:41

DESCRIPTION

This command allows the user to turn off several useful shell parameters. The command set allows the same parameters to be turned off.

interactive

The interactive option when off has no effect on program execution. Commands typed with no parameters receive the interactive editor, commands with parameters are taken as is.

noexecute

The noexecute flag when off sets all commands to normal mode. They parse and examine parameters and will execute the command if they have legal data.

BUGS

The boolean variables really ought to understand a parameter so they can be set on or off. This would make this command superfluous.

The parameters all have to be typed in full. The parser does not understand shortened versions.

SEE ALSO

set()

SUPREM IV

Level: Commands

Version: 3.6 12/7/88 07:49:57

DESCRIPTION

This is a brief introduction to the commands and their actions in SUPREM IV. These are commands that usually have some action associated with them, as opposed to the models section commands which just set coefficients.

There are several groups of commands in this section. The first are commands which are used for i/o of data. The second are simulation commands which are either the (meat and potatoes - Greater Midwest) or (corn beef and cabbage - Ireland) of SUPREM IV. The third are commands which are primarily for the post processing. The remainder are bunched in the aptly named miscellaneous category.

Data Input and Output Commands

boundary

This command allows the user to specify lines which are exposed to gas in a rectangular grid.

initialization

This command allows the user to set up the initial grid and specify background concentrations.

line This command allows the user to position x and y grid lines for a rectangular mesh.

profile This command allows the user to read in ascii data file of depth and doping data.

region The region command allows the user to specify which sections of the rectangular mesh are which material.

structure

The structure command allows the user to read and write mesh and solution values. This is the main i/o of data to and from the program.

Simulation Commands

etch This command allows the user to etch layers.

deposit This command allows the user to deposit layers.

diffuse This command allows the user to specify a time temperature step.

implant This command allows the user to model an implant with either a gaussian or a pearsonIV distribution.

method This command allows the user to pick the numerical options for solving the equations.

stress This command computes the thermal elastic stresses.

Post Processing Commands

color This command is similar to the contour card except that it allows area fill between two contours of the fill of a specific material. It is usually used with the plot.2d statement.

contour This command allows the user to plot an isoconcentration line of the selected variable. It is usually used with the plot.2d statement.

label This allows the user to place a label on the plot at a given location.

option This command allows the user to pick the type of graphics device to use, specify graphics hard copy files, or specify the program's verbosity level.

plot.1d This command allows the user to plot the selected variable in one dimensional cross sections through device.

plot.2d This command allows the user to plot the outline and/or grid lines in the two dimensional mesh. It is very useful for setting up the display for the contours.

plot.3d This command lets the user plot a three dimensional bird's eye view of the device and selected variable.

print.1d This prints the information that a plot.1d would draw.

select This command allows a variable to be chosen as the z coordinate for the plot command to follow.

viewport
This allows plotting into subsets of the terminal screen.

Miscellaneous Commands

cpulog This command instructs the program to dump cpu statistics whenever it feels like it.

echo This command prints a string.

pause This command waits for the user to input a command or a simple return.

printf This command passes each white space separated token to the expression parser.

BUGS

There are probably too many bugs to be listed here.

SEE ALSO

See the other command descriptions for more detail.

BOUNDARY

boundary – Specify a surface type.

SYNOPSIS

boundary

reflecting | exposed | backside
xlo = <string>
ylo = <string>
xhi = <string>
yhi = <string>

SUPREM IV

Level: Commands

Author: Conor S. Rafferty

Version: 3.3 9/29/88 16:19:12

DESCRIPTION

This statement is used to specify what conditions to apply at each surface in a rectangular mesh.

reflecting | exposed | backside

At present, three surface types are recognized. **Exposed** surfaces correspond to the top of the wafer. Materials are only deposited on exposed surfaces (so if a deposit does nothing you know where to start looking). Impurity predeposition also happens at exposed surfaces, as does defect recombination and generation.

Backside surfaces roughly correspond to a nitride- or oxide-capped backside. Defect recombination and generation happen here.

Reflecting surfaces correspond to the sides of the device, and are also good for the backside if defects are not being simulated.

The default for **all** surfaces is reflecting.

xlo, ylo, xhi, yhi

The bounds of the rectangle being specified. The <string> value should be one of the tags created in a preceding line statement.

EXAMPLES

boundary exposed xlo=left xhi=right ylo=surf yhi=surf
boundary backsid xlo=left xhi=right ylo=back yhi=back

These lines specify that the top of the mesh is the exposed surface, and that the bottom is the backside.

BUGS

The top surface should probably default to exposed, and the bottom to backside. We took a poll, though, and decided it was safer that our users knew about the existence of boundary codes.

COLOR

color -- perform color fill

SYNOPSIS

color [color=<n>] [min.value=<n>] [max.value=<n>]
[(silicon | oxide | oxynitr | nitr | poly | alumin | photore)]

SUPREM IV

Level: Commands

Author: Mark E. Law

Version: 3.2 9/29/88

DESCRIPTION

The color statement allows the user to perform color fills on either isoconcentration bands in the selected variable or on materials.

color This allows the color to be specified for this color fill. The default is one. Different colors have different meanings to different terminals. It is advised that the user check any gplot documentation to determine the real effect of this on the terminal of interest.

min.value, max.value
These parameters specify two isoconcentration contours. The color specified will be drawn between these two contours.

(silicon | oxide | oxynitr | nitr | poly | alumin | photore)
These parameters indicate that the specifed material should all be drawn the given color.

EXAMPLES

color min.v=10 max.v=12 color=3
This command will draw a band of color 3 between the isoconcentration lines at 10.0 and 12.0.

BUGS

This should probably check to make sure a plot.2d has been done. Standard Disclaimer about how gplot bugs will show up in this command.

SEE ALSO

plot.2d(2), for(1), gplot(5), plotcap(5)

CONTOUR

contour – plot contours in the selected variable on a two d plot

SYNOPSIS

contour [line.type=<n>] [value=<n>] [symb=<n>]

SUPREM IV

Level: Commands

Author: Mark E. Law

Version: 3.2 9/29/88

DESCRIPTION

The contour statement draws a contour in the selected variable (see the select statement) at the value specified. Value must be specified in the range of the computed variable. For instance, if plotting log boron, value should be in the range 10 to 20 and not 1e10 to 1e20. This statement assumes a plot.2d has been specified and the screen has been set up for plotting a two dimensional picture. If this has not been done, the routine will probably produce garbage on the screen.

line.type

This allows the line type to be specified for this contour. The default is one. Different line types have different meanings to different terminals. It is advised that the user check any gplot documentation to determine the real effect of this on the terminal of interest.

value This floating parameter expresses the value that the contour line should be plotted at. If boron had been selected, a value of 1e16 would produce a line of constant boron concentration of 1e16 per cm cubed.

symb This integer parameter indicates that the contour line should be drawn with symbols on the contour line. The integer value indicates different symbols. For a list of symbols, please refer to the high2 library documentation.

EXAMPLES

contour val=1e10 line.type=2

This command will draw a line with line.type 2 at a isoconcentration of 1.0e10.

BUGS

This should probably check to make sure a plot.2d has been done. It is conceivable that this statement could produce floating point exceptions when a plot.2d has not been done. This is, to put it midly, annoying. Standard Disclaimer about how gplot bugs will show up in this command.

SEE ALSO

plot.2d(2), for(1), gplot(5), plotcap(5)

CPULOG

cpulog – log the cpu usage summary to a file

SYNOPSIS

cpu [log] [cpufile=<c>]

SUPREM IV

Level: COMMANDS

Author: Mark E. Law

Version: 3.2 9/29/88

DESCRIPTION

The cpulog statement allows the user to get a summary of cpu usage by the program. The user can select this information to appear on his terminal or have it sent to standard out. Most of the cpu intensive routines will report to the user the time used.

log This option specifies whether to start or stop cpu usage logging. A true value enables cpu usage logging, false turns it off. The default is true.

cpufile This string parameter allows the user to select a file name for the output of cpu usage. The default is to stdout.

EXAMPLES

cpu log

This command enables cpu reporting to the screen.

cpu log file=foo

This enables the cpu statistics reporting and stores them in file foo.

BUGS

The cpu usage statistics of an algorithm are only as complete as the programmer who implemented the algorithm. The times can be no more accurate than a sixtieth of a second (except on the Cray X-MP/4, where 9.5 nanonseconds is the relevant constant).

DEPOSIT

deposit – deposit a layer

SYNOPSIS

deposit

silicon | oxide | oxynitr | nitride | poly | photores | alumin
thickness = <n> [divisions = <n>]
[none | arsenic | antimony | boron | phosphor]
[conc=<n>] [space=<n>]

SUPREM IV

Level: Commands

Author: Conor S. Rafferty/Mark E. Law

Version: 3.3 9/29/88 08:44:45

DESCRIPTION

This statement provides a primitive deposition capability. Material is deposited on the "exposed" surface of the structure, and its upper surface becomes the the new exposed surface.

silicon | oxide | oxynitr | nitride | poly | photores | alumin
: The material to be deposited.

thickness = <n>
: The thickness in microns.

divisions = <n>
: The number of vertical grid spacings in the layer, default 1.

none | arsenic | antimony | boron | phosphor
: The type of doping in the deposited layer.

conc : This floating point parameter specifies the level of the doping in the deposited layer.

space : This floating point parameter represents the average spacing between points along the outside edge of the deposit.

EXAMPLES

deposit oxide thick=0.02
: This deposits silicon dioxide 200 angstroms thick on the surface of the silicon.

BUGS

Only uniform grid in the deposited material.
Occasionally makes grid errors in following the surface.
The offset profile is not based on any physical model.

DIFFUSE

diffuse – Run a time temperature step on the wafer and calculate oxidation and diffusion of impurities

SYNOPSIS

diffuse

time=<n> temperature=<n>
[(dryo2 | weto2 | nitrogen | ammonia | argon |
| antimony | arsenic | boron | phosphorus)]
[gas.conc=<n>] [pressure=<n>]
[gold.surf] [continue]
[movie=<c>] [dump=<n>]

SUPREM IV

Level: Commands

Author: Mark E. Law - Diffusion code
Conor S. Rafferty - Oxidation code

Version: 3.5 11/21/88

DESCRIPTION

The diffuse statement allows the user to specify a diffusion or oxidation step. Any impurities present in the wafer are diffused. If the wafer is exposed to a gas, a predeposition or oxidation can be performed. This statement represents the core simulation capability of SUPREM-IV. Users can expect that this statement will take a lot of CPU time.

The parameters for oxidation and diffusivities can be found on the statements in the models section of the manual. Refer to either the oxide statement or any of the impurity statements to determine how to change coefficients of the models.

time This parameter represents the amount of time the wafer will spend in the furnace in minutes.

temp This parameter allows specification of the furnace temperature in degrees centigrade.

antimony, arsenic, boron, phosphorus
The above switched booleans of gas types allow the user to specify the type of impurity that may be in the gas stream. The value of gas.conc applies to these gas types. Only one gas type may be specified per diffusion step.

dryo2 weto2 nitrogen ammonia argon
The above switched booleans of gas types allow the user to specify the type of gas present in the furnace during the diffusion step. These gas types are not affected by the gas.conc parameter. Only one gas type may be specified per diffusion step. There is currently no difference between nitrogen, argon, and ammonia.

gas.conc
This allows the user to specify the gas concentration of an impurity during predepositions, in atoms per centimeter cubed.

pressure
This is the partial pressure of the active species, in atmospheres. It defaults to 1 for both wet and dry oxidation.

gold.surf
This floating point parameter specifies the gettered gold concentration value as a sink. It is used for the gettering model described on the gold command.

continue
This specifies a continuing diffusion. Do not reset the total time to 0, do not reset the analytic oxide thickness to the default value of "initial" on the oxide card, do not initialize the defects, do not collect $200.

movie This character string parameter should be made up of SUPREM-IV commands. They will be

executed at the beginning of each time step. It is typically used to produce plots of a variable as a function of time during the diffusion process. Any legal SUPREM-IV command can be specified in the string. The variable time is set by the movie routine. Consequently, time will be expanded as a macro for use in plot labels or print outs.

dump This parameter instructs the simulator to output files on after every dumpth time step. The files will be structure files and will be readable with the structure statement. The names will be of the form s.time, where time is the current total time of the simulation.

EXAMPLES

diffuse time=30 temp=1000 boron gas.conc=1.e20

This statement specifies that a 1000 degree, 30 minute boron predeposition is to be performed.

diffuse time=30 temp=1000 dryo2

This statement instructs the simulator to grow oxide for 30 minutes in a dry oxygen ambient.

diffuse time=30 temp=1000 movie="sel z=log10(bor); plot.1d x.v=1.0;"

This command instructs SUPREM-IV to do a 30 minute 1000 degree diffusion. Before each time step, the current boron concentration will be plotted along 1.0 micron into the wafer. (See select, plot.1d)

diffuse time=30 temp=1100 movie="
sel z=doping
printf Time is: time yfn(0.0, 0.0) sil@oxi(0.0)"

This command instructs SUPREM-IV to do a 30 minute 1000 degree diffusion. Before each time step, the total net active doping concentration is chosen as the plot variable. The printf statement will then print the string "Time is:", the current time of the diffusion in seconds, the y value along a vertical slice at x = 0.0 where the doping is equal to zero (a.k.a. the junction depth), and the location of the silicon oxide interface in the vertical slice at x = 0.0.

BUGS

There can still be occasional convergence problems. If these occur when defects are being computed, they can usually be fixed by performing a complete LU decomposition instead of a partial. Use the "method full" command to achieve this.

The models implemented in the diffusion and oxidation are involved in ongoing research. Disagreements between simulation and experiment is possible. This can be due to the differences in processing conditions between the lab where the original experiment was performed and your lab, or due to model shortcomings.

SEE ALSO

antimony(3), arsenic(3), boron(3), cesium(3), interst(3), material(3), method(3), oxide(3), phosphorus(3), trap(3), vacancy(3)

ECHO

echo – a string printer and desk calculator

SYNOPSIS

echo [string]

SUPREM IV

Level: SHELL

Author: Mark E. Law

Version: 3.3 9/29/88 10:28:40

DESCRIPTION

This command merely prints the string given to it. This is useful for placing comments in an output file. The command attempts to parse the string to a legal real number if possible. It has a regular expression parser built-in. This allows echo to be used as a desk calculator.

EXAMPLES

echo jimmin at the jimjam frippin at the frotz

This will send the string "jimmin at the jimjam frippin at the frotz" to standard out.

echo (2^3^4)

A.K.A. 4096

echo (15.0 - 12.0 * exp(4.0 - 2.0 / 6.0))

This will print the result of the arithmetic expression, which is of course, 8.373.

BUGS

No known bugs.

SEE ALSO

unix command - more(1)

ETCH

etch – etch a layer

SYNOPSIS

etch

[silicon | oxide | oxynitr | nitride | poly | photores | alumin]
[left | right | start | continue | done | dry | all]
[x=<n>] [y=<n>] [thick=<n>]
[p1.x=<n>] [p1.y=<n>] [p2.x=<n>] [p2.y=<n>]

SUPREM IV

Level: Commands

Author: Conor S. Rafferty/Mark E. Law

Version: 3.5 11/21/88 13:14:24

DESCRIPTION

This statement provides an etching capability. Material is etched from the structure, and the underlying regions are marked as the "exposed" surface. The etch shape can be described in several diferent ways. The user must specify the final shape of the region to be etched, there is no capability at present to simulate the etch region shape.

[silicon | oxide | oxynitr | nitride | poly | photores | alumin]
The material to be etched. If a material is specified, only that material is etched even if other materials lie within the etch region. If no material is specified, all materials in the etch region are removed.

[left | right]
These provide the user with a quick means of doing trapazoidal shaped etches. The etch region is to the left/right of the line specified by the coordinates given in p1.x,p1.y and p2.x,p2.y.

[start | continue | done]
This allows the user to specify an arbitrarily complex region to be etched. Several lines can be combined to specify the several points that make up the region. See the examples.

[dry] This parameter indicates that the etch shape should be a replica of the current surface, but straight down "thickness" microns.

[all] All of the specified material is removed.

[x=<n>] [y=<n>]
These parameters allow the user to specify point in the start/continue/done mode of etch region specification.

[p1.x=<n>] [p1.y=<n>] [p2.x=<n>] [p2.y=<n>]
These parameters allow the spcification of a line for left/right etching.

[thick=<n>]
This parameter allows specification of a thickness for the dry etch type.

EXAMPLES

etch nitride left p1.x = 0.5 p1.y=0.0 p2.x=0.5 p2.y=-1.0
This command etches all the nitride left of a vertical line at 0.5.

```
etch oxide start x=0.0 y=0.0
etch continue x=1.0 y=0.0
etch continue x=1.0 y=1.0
etch done x=0.0 y=1.0
```

This etches the oxide in the square defined at (0,0), (1,0), (1,1), and (1,0).

```
etch dry thick=0.1
```

This etch will find the exposed surface, lower it straight down 0.1 microns and this line will be the new surface.

BUGS

Uses no physical model.
Occasionally produces invalid grids.
Usually produces sub-simulation quality grid around corners.
This code is sensitive to grid placement. It often helps to prepare the substrate by having a vertical grid line at the foot of the etched part of a deposited layer. Otherwise a fascinating ripped nylon mesh is generated.
The code currently does *not* distinguish between surface and subsurface materials. "Etch silicon left p1.x=0.2" will etch *all* the silicon to the left of 0.2, including the substrate. Use the start/cont/end sequence to work around this.

IMPLANT

implant – perform a implant

SYNOPSIS

implant

[(antimony | arsenic | boron | bf2 | cesium | phosphorus | silicon)]
[gauss | pearson]
dose=<n> energy=<n>
[damage] [max.damage=<n>]
[range=<n>] [std.dev=<n>]
[gamma=<n>] [kurtosis=<n>]

SUPREM IV

Level: Commands

Author: Mark E. Law & Michael J. Eldredge

Version: 3.4 11/21/88

DESCRIPTION

This statement is used to simulate an implantation. There are two different types of analytic model available. The first implements a simple Gaussian. The second uses the moments data file to compute a Pearson-IV distribution. These models are only as good as the data in the files. The file data can be overridden by directly specifying the appropriate values on the implant command line.

antimony, arsenic, boron, bf2, cesium, phosphorus, silicon
: These parameters specify the impurity to be implanted. Default data for the range statistics exists only for antimony, arsenic, boron, bf2, and phosphorus.

gauss, pearson
: These parameters specify which type of distribution is being used. A gaussian distribution is the default.

dose
: This parameter allows the user to specify the dose of the implant. The units are in cm^{-2}.

energy
: This parameter specifies the implant energy in eV.

damage
: This parameter indicates that a computation of the damage due to the implant should be performed. The data is from Hobler and Selberherr [1] and exist only for antimony, arsenic, boron, and phosphorus.

max.damage
: The maximum amount of damage to that makes sense to calculate. Clearly damage above the crystal density is absurd.

[range=<n>] [std.dev=<n>] [gamma=<n>] [kurtosis=<n>]
: These parameters allow the user to override the table values. The full set of data has to be given. It is not possible, for instance, to override only the range parameter. The gamma and kurtosis variables only have meaning for the Pearson-IV distribution.

EXAMPLES

implant phosph dose=1e14 energy=100 pearson
: This statement specifies that a 100eV implant of phosphorus was done with a dose of 1.0e14. The pearson model is to be used for the distribution function.

implant phos dose=1e14 range=0.1 std.dev=0.02 gauss
: This statement specifies an implant of phosphorus was done with a dose of 1.0e14. The gaussian model is to be used for the distribution function. The range and standard deviation are specified in microns instead of using table values.

implant phosph dose=1e14 energy=100 pearson damage
: This statement specifies that a 100eV implant of phosphorus was done with a dose of 1.0e14. The pearson model is to be used for the distribution function. The damage from this implant is

calculated and stored as point defects.

BUGS

The implementation works by fitting a temporary rectangular mesh to the substrate, doing the calculation on the rectangular mesh, and interpolating back. For completely irregular grids, e.g. those generated by **tri**, the size of the rectangular mesh overflows memory.

REFERENCES

1. G. Hobler and S. Selberherr, "Two-Dimensional Modeling of Ion Implantation Induced Point Defects," *IEEE Transactions on Computer Aided Design*, vol. 7 No. 2, p. 174, February, 1988.

INITIALIZE

initialize – setup grid, background doping levels

SYNOPSIS

initialize
[infile=<c>]
(antimony | arsenic | boron | phosphorus | gold)
[conc=<n>]
[orientation=<n>]
[line.data] [interval.r]
[scale] [flip.y]

SUPREM IV

Level: Commands

Author: Conor S. Rafferty / Mark E. Law

Version: 3.4 10/5/88 15:14:19

DESCRIPTION

The initialize statement sets up the mesh from either a rectangular specification or a from a previous structure file. The statement also initializes the background doping concentration.

infile The infile parameter selects the file name for reading. The file must contain a previously saved structure, see the structure command. A generated grid using tri may also be input at this point. If no filename is specified, the program will finish generating a rectangular mesh.

antimony, arsenic, boron, phosphorus, gold
These parameters specify the type of impurity that forms the background doping.

conc This parameter allows the user to specify the background concentration in cm^-3.

orientation=<n>
This parameter allows the user to specify the substrate orientation. Only 100, 110 and 111 are recognized.

line.data
This boolean parameter indicates that the locations of mesh lines should be listed.

interval.r
This parameter is the maximum ratio between the distances of adjoining mesh lines. The default is 1.5.

scale Parameter to scale the size of an incoming mesh. The default is 1.0.

flip.y This boolean parameter indicates if the mesh should be flipped around the x axis. If is sometimes useful for reading grids prepared with **tri**.

EXAMPLES

init infile=foo
This command reads in a previously saved structure in file foo.

initialize conc=1e15 boron
This command finishes a rectangular mesh and sets up with boron doping of 10e15 cm^-3.

BUGS

There are no known bugs.

SEE ALSO

bound, line, region, structure, skel, tri

LABEL

label – Put labels on a picture

SYNOPSIS

label

[x=<n> y=<n>] [label=<string>]
[symb=<n>] [line.type=<n>]

SUPREM IV

Level: Commands

Author: Mark E. Law & CSR

Version: 3.3 9/29/88 16:33:56

DESCRIPTION

The label command allows the user to put random pieces of text anywhere on a SUPREM4 picture.

x y Where to put the label, in microns. If not specified, a graphics cursor will pop up on the screen and the mouse or locator can be used to point where the label is wanted, if a mouse or locator is supported by the graphics hardware. In this case some extra information will be printed about the location of the cursor, including region and material number.

label What string to print.

symb line

These parameters will draw a short line of the specified linetype or a symbol of the specified number in front of the text. Useful for labelling contours, for instance.

EXAMPLES

label label="Arsenic" line=3
label label="Phosphorus" line=4

These commands will put two labels on the plot at a position specified by the mouse.

label label="This is Interesting, eh?" x=10.0 y=10.0

This will place the string, centered, at the location 10, 10.

BUGS

Standard Disclaimer about gplot software bugs.

SEE ALSO

plot.1d(2), plot.2d(2), plot.3d(2), for(1), gplot(5), plotcap(5)

LINE

line – Specify a mesh line

SYNOPSIS

line

x | y
location = <value>
[spacing = <value>]
[tag = <string>]

SUPREM IV

Level: Commands

Author: Conor S. Rafferty

Version: 3.2 9/29/88

DESCRIPTION

This statement is used to specify the position and spacing of mesh lines. All line statements should come before region and boundary statements, which should in turn be followed by an initialize statement.

Only flat structures can be specified with line / region / boundary statements. To create the grid for a more complicated structure, use the interactive grid generator skel(1).

x | y A mesh line is either horizontal or vertical. The x coordinate increases from left to right. The y coordinate increases into the substrate, that is, from top to bottom. People who learned analytic geometry with the y axis pointing up seem to find this confusing.

location The location along the chosen axis, in microns.

spacing The local grid spacing, in microns. SUPREM-IV will add mesh lines to the ones given according to the following recipe. Each user line has a spacing, either specified by the user or inferred from the nearest neighbor. These spacings are then smoothed out so no adjacent intervals will have a ratio greater than 1.5. New grid lines are then introduced so that the line spacing varies geometrically from the user spacing at one end of each interval to that at the other. The example below might alleviate the confusion you must be experiencing.

tag Lines can be labeled for later reference by boundary and region statements. The label is any word you like.

EXAMPLES

```
line x loc=0 spa=1        tag=left
line x loc=1 spa=0.1
line x loc=2 spa=1        tag=right

line y loc=0 spa=0.02     tag=surf
line y loc=3 spa=0.5      tag=back
```

There are 3 user-specified x lines and 2 user y lines. Taking the x lines for illustration, there is finer spacing in the center than at the edges. After processing, SUPREM-IV produces a mesh with x lines at 0.0, 0.42, 0.69, 0.88, 1.0, 1.12, 1.31, 1.58, 2.0. Around the center, the spacing is 0.12, approximately what was requested. At the edge, the spacing is 0.42, because that was as coarse as it could get without having an interval ratio greater than 1.5. If the interval ratio was allowed to be 9, say, then we would have got one interval of 0.9 and one interval of 0.1 on each side. In this example, specifying a spacing of 1 at the edges was redundant, because that's what the spacing of the user lines was anyway.

BUGS

It's hard to guess how many lines are going to be generated in each interval.
Coarse spacings are sometimes rounded down by the smoothing algorithm in an undesirable way, creating too many grid points.

Automatic regrid has been broken since oxidation introduced moving boundaries. Therefore the initial mesh specification is quite important to the success of the simulation. Use ballpark numbers for the profiles, or coarse-grid simulations, to get a feel for the required spacings in the initial and final profile.

METHOD

method -- select numerical methods and models for diffusion and oxidation

SYNOPSIS

method
[(vacancies | interstitial | arsenic | phosphorus
| antimony | boron | oxidant | velocity | traps
| gold | psi | cesium)]
[rel.error=<n>] [abs.error=<n>]
[init.time=<n>] [(trbdf | formula)]
[min.fill] [min.freq=<n>]
[(gauss | cg)] [back=<n>] [blk.itlim=<n>]
[(time | error | newton)]
[(diag | knot | full.fac)]
[(fermi | two.dim | steady | full.cpl)]
[(erfc | erfg | erf1 | erf2 | vertical | compress | viscous)]
[grid.oxide=<n>] [skip.sil] [oxide.gdt=<n>] [redo.oxide=<n>]
[oxide.early] [oxide.late] [oxide.rel]
[poisson]

SUPREM IV

Level: Commands

Author: Mark E. Law Conor S. Rafferty

Version: 3.8 12/6/88 17:17:11

DESCRIPTION

This statement sets the flags for selection of algorithm for various mathematical solutions. The statement also allows the user to select the complexity desired for the diffusion and oxidation models. Appropriate defaults for the numerical parameters are included in the model start up file so users should only have to specify the type of model desired for the diffusion and oxidation. The numerical methods used in SUPREM-IV for solution of the diffusion equations are described in more detail in Law and Dutton [1].

vacancies interstitial arsenic phosphorus antimony boron oxidant velocity traps gold psi

These options allow the user to specify a single impurity. The error bound parameters are specific for each impurity. If error bounds are listed, a impurity must also be selected.

rel.err This parameter indicates the precision with which the impurity block must be solved. In general, the error will be less that half of the indicated error since the program is conservative in its error estimates. The defaults are 0.01 for all impurities except the potential which is solved to 0.001.

abs.err This parameter indicates the size of the absolute value of the error. For the dopants, the absolute error defaults to 1.0e9. For defects, the absolute error defaults to 1.0e5. For the potential, the error defaults to 1.0e-6.

init.time

This parameter specifies the size of the initial time step. It defaults to 0.1 seconds. A routine to estimate the initial time step is being worked on.

(trbdf | formula)

These parameters specify the method of time integration to be used. The trbdf option indicates a combination trapezoidal rule/backward difference should be used. The error is estimated using Milne's device. The formula method allows the user to specify the time step directly as a function of time (t), previous time step (dt) and grid time (gdt). This option is primarily for testing. The trbdf method is the default. The time step methods have been taken from literature, see Yeager and Dutton [2] and Bank *et al.* [3].

min.fill, min.freq=<n>

This option allows the user to specify a minimum fill. It defaults to true. This is a highly

recommended option since it can reduce the matrix sizes by a factor of two or more. The size of the matrix is proportional to speed. It defaults true. Min.freq is a parameter which controls the frequency of min fill reorderings. It is only partially implemented and will have no effect on the calculation.

(gauss | cg)
: These parameters allow the user to specify the type of iteration performed on the linear system as a whole. Gauss specifies that this iteration should be Gauss-Seidel, cg specifies that a conjugate residual should be used. The default is cg. The gauss option is considerably slower, it should be deleted in the future. The CG algorithm is described in Elman [4].

back
: This integer parameter specifies the number of back vectors that can be used in the cg outer iteration. The default is three, and the maximum that can be used is five. The higher the number, the better the convergence and the more memory used.

blk.itlim
: The maximum number of block iterations that can be taken. The block iteration will finish at this point independent of convergence.

(time | error | newton)
: These parameters specify the frequency with which the matrix should be refactored. The time parameter specifies that the matrix should be refactored twice per time step. This option takes advantage of the similarity in the matrix across a time integration. The error option indicates the matrix should be refactored whenever the error in that block is decreasing. The newton option will force the refactorization at every newton step. The default is time.

(diag | knot | full.fac)
: These parameters specify the amount of fill to be included in the factorization of the matrix. Full.fac indicates that the entire amount of fill is to be computed. The diag option indicates that the only the diagonal blocks should be factored in the matrix. The knot option is currently unimplemented. The diag option is the default, although under certain conditions (one dimensional stripes) full will perform better.

(fermi | two.dim | steady | full.cpl)
: These parameters specify the type of defect model to be used. The fermi option indicates that the defects are assumed to be a function of the fermi level only. The two.dim option indicates that a full time dependent transient simulation should be performed. The steady option indicates that the defects can be assumed to be in steady state. The full.cpl option indicates that the full coupling between defects and dopants should be included. The default is fermi.

(erfc | erfg | erf1 | erf2 | vertical | compress | viscous)
: These are oxide model parameters. The erfc option indicates that a simple error function approximation to a bird's beak shape should be used. The erf1 and erf2 models are analytic approximations to the bird's beak from the literature (see the oxide card). The erfg model chooses whichever of erf1 or erf2 is most appropriate. The vertical model indicates that the oxidant should be solved for and that the growth is entirely vertical. The compress model treats the oxide as a compressible liquid. The viscous model treats the oxide as an incompressible viscous liquid. Real oxide is believed to be incompressible, but the compressible model is a lot faster. It's an accuracy-speed tradeoff.

grid.oxide=<n>
: This parameter is the desired thickness of grid layers to be added to the growing oxide. The value is specified in microns. It represents the desired thickness of grid layers to be added to the growing oxide, in microns. It has a side effect on time steps (see oxide.gdt).

skip.sil
: This boolean parameter indicates whether or not to compute stress in the silicon. The default is false. The calculation can only performed when the viscous oxide model is used. The silicon substrate is treated as an elastic solid subject to the tractions generated by the oxide flow. It should not be turned on indiscriminantly, since the silicon grid is generally much larger than the oxide

grid, and the stress calculation correspondingly more expensive.

oxide.gdt=<n>

The time step may be limited by oxidation as well as diffusion; the oxide limit is specified as a fraction (oxide.gdt) of the time required to oxidize the thickness of one grid layer (grid.oxide).

redo.oxide=<n>

To save time, the oxide flow field need not be computed every time the diffusion equation for impurities is solved. The parameter redo.oxide specifies the percentage of the time required to oxidize the thickness of one grid layer which should elapse before resolving the flow field. Usually redo.oxide is much less than oxide.gdt, which is an upper bound on how long the solution should wait. It is mainly intended to exclude solving oxidation at each and every one of the first few millisecond time steps when defects are being tracked.

oxide.early=<n> oxide.late=<n> oxide.rel=<n>

These parameters are not normally modified by users. They relate to internal numerical mechanisms, and are described only for completeness. A node whose spacing decreases proportionally by more than oxide.late is marked for removal. If there any nodes being removed, then in addition all nodes greater than oxide.early are removed. The default values are oxide.early=0.5 and oxide.late=0.9. For earlier node removal (less obtuse triangles), try oxide.late=0.3 and oxide.early=0.1. It makes no sense, but does no harm, to have oxide.early greater than oxide.late. The oxide.rel parameter is the solution tolerance in the nonlinear viscous model. The default is 1e-6 (that is, a 1e-4 percentage error in the velocities). It can be increased for a faster solution, but a value greater than about 1e-3 will probably give meaningless results.

EXAMPLES

method arsen rel.err=0.01 abs.err=1.0e9

This card indicates that the arsenic equation should be solved with a relative error of 1% and concentrations below 1.0e9 can be ignored.

method min.fill cg back=3 init.ti=0.1 trbdf fermi vert skip.sil=f

This card specifies that minimum fill reordering should be done and that the entire system should be solved using a conjugate residual technique with three back vectors. The initial time step should be 0.1 seconds and time should be integrated using the trbdf model. The fermi model should be assumed for the defects and the vertical model for the oxide growth. The stresses and potential and the silicon should not be computed.

BUGS

It is unclear if the numbering scheme provided by min.fill is optimal. More research needs to be done here, especially for band ordering on vector machines. The only need for user intervention here should be to change the oxidation and defect models.

REFERENCES

1. M.E. Law and R.W. Dutton, "Verification of Analytic Point Defect Models using SUPREM-IV," *IEEE Transactions on Computer Aided Design*, p. 181, Feb., 1987.

2. H.R. Yeager and R.W. Dutton, "An Approach to Solving Multi-Particle Diffusion Exhibiting Nonlinear Stiff Coupling," *IEEE Transactions on Electron Devices*, vol. ED-32, pp. 1964-1976, Oct. 1985.

3. R.E. Bank, W.M. Coughran, Jr., W. Fichtner, E.H. Grosse, D.J. Rose, and R.K. Smith, "Transient Simulation of Silicon Devices and Circuits," *IEEE Trans. on Electron Devices*, vol. ED-32, p. 1992, Oct 1985.

4. H.C. Elman, "Preconditioned Conjugate Gradient Methods for Nonsymmetric Systems of Linear Equations," *Research Report* , vol. 203, April, 1981.

OPTION

option – set terminal options

SYNOPSIS

option

[term=<c>]
[on.save] [off.save] [stop.save]
[file.save=<c>]
[quiet | normal | chat | barf]

SUPREM IV

Level: COMMANDS

Author: Mark E. Law

Version: 3.4 9/29/88 16:55:19

DESCRIPTION

The option statement allows the user to specify the type of terminal they are using for plotting. The default is in the environment variable DEFPDEV. If this environment variable is not set, it tries to use the environment variable TERM. If this is not set, it uses a system wide default (see how to set up your system in the plotcap documentation). This command allows the user to override these and select a plot device from within the program.

The save options allow the user to specify output files for dplot. The dplot software should also have been released from Stanford at this time.

terminal

This string variable allows the user to name the terminal type they are using. For example, hp2623, 2623, and HP2623, all refer to the Hewlett Packard terminal model number 2623.

file.save

This parameter opens the file name for graphics save. It also starts the saving of the graphics.

on.save, off.save, stop.save

These parameters control the save to file. The stop.save parameter ends the saving and closes the file. The off.save pauses the saving for a period and the saving can be restarted with on.save.

quiet, normal, chat, barf

These parameters determine the amount of information that is output to the user about errors, cpu times, behavior of the algorithms, etc. The default is normal. The chat and barf modes are mainly useful for debugging and would not normally be chosen by any user in their right mind.

EXAMPLES

option term=tekc

The terminal being used is a color tektronix 4107 terminal.

option file.save = foo

Start saving graphics information in the file foo.

BUGS

It is sometimes hard to figure out what a particular terminal is referred to as in the gplot entry tables. The documentation (i.e. this document) should contain a helpful list of supported devices and their names.

SEE ALSO

gplot(5), plotcap(5)

PAUSE

pause – wait and execute command

SYNOPSIS

pause

SUPREM IV

Level: COMMAND

Author: Conor S. Rafferty

Version: 1.1 9/29/88 09:28:35

DESCRIPTION

This command will stop the program and ask the user for input. The user may type any legal SUPREM-IV command string to be executed. This is useful for breaking up long input scripts to check the status, or placing in the movie parameter of a diffuse command to check the progress of the diffusion.

EXAMPLES

pause

The program will stop, print:
Type <RETURN> to continue, or a command to be excuted:
and will wait for the user to respond.

BUGS

No known bugs.

PLOT.1D

plot.1d – plot a one dimensional cross section

SYNOPSIS

plot.1d

[(x.value=<n> | y.value=<n>)]
[silicon | oxide | nitride ...] [/exposed | /backside | /reflect | /silicon | /oxide | /nitride ...]
[boundary] [axis] [clear]
[line.type=<n>] [symb=<n>]
[x.max=<n>] [x.min=<n>] [y.max=<n>] [y.min=<n>]
arclength

SUPREM IV

Level: Commands

Author: Mark E. Law Conor S. Rafferty

Version: 3.4 9/29/88

DESCRIPTION

The plot.1d statement allows the user to plot cross sections vertically or horizontally through the device. The statement has options to provide for initialization of the graphics device and plotting of axis. The statement can optionally draw vertical lines whenever a material boundary is crossed. The vertical axis is the variable that was selected (see the select statement). The plot can have limits given to it so that only a portion of the entire device will be shown, or more than one variable can be conveniently plotted.

The options and their action are described below:

x.value, y.value

This parameter specifies that the cross section is to be done at a constant value as specified by this parameter. This parameter is in microns. Only one of x.value or y.value can be specified for a given device. Specifying x.value will produce a vertical slice through the device and y.value will specify a horizontal slice. An easy way to remember is that the cross section is taken at the constant value specified.

silicon | oxide | nitride ... /silicon | /oxide | /nitride

In addition to constant x or y cross-sections, a one-dimensional plot can be generated along an interface. The interface lies between material 1, named without a "/", and material 2, named with a "/". For example, "plot.1 oxide /silicon" shows the values in the oxide at the oxide/silicon interface. Thus "plot.1 oxide /silicon" will usually show something different from "plot.1 silicon /oxide". The backside, reflecting or exposed surfaces of a material can be specified with the appropriate parameter. The ordinate on the axis is the x coordinate of the points along the interface. If the optional parameter arclength is set, the ordinate is instead the arc length along the boundary from the leftmost point on the curve. The leftmost point itself has ordinate equal to its x coordinate in the mesh.

boundary — This parameter specifies that any material boundaries that are crossed should be drawn in as vertical lines on the plot. It defaults false.

axis — This parameter specifies that an axis should be drawn. If this parameter is set false and clear is false, the plot will be drawn on a top of the previous axis. It defaults true.

clear — The clear parameter specifies whether the graphics screen should be cleared before the the graph is drawn. If true, the screen is cleared. It defaults true.

line.type — The line.type parameter controls the line type or color that the cross sectional line is drawn in. The axis is always drawn in line.type=1. The parameter defaults to 1.

symb — This parameter allows the user to specify a symbol type to be drawn on the cross sectional line. Each point is drawn with the specified symbol. Symb=0 means no symbol should be drawn. It defaults to 0.

x.max, x.min — These parameters allow the user to specify the range of the x axis. The units are in microns and the default is the length of the device in the appropriate direction of the cross section.

y.max, y.min — These parameters allow the user to specify a range for the plot variable. The values have to be in units of the selected variable. If log of the boron concentration was selected, the values for this should be 16 and 10 to see the concentrations between 1e16 and 1e10.

EXAMPLES

plot.1d x.v=1.0 symb=1 axis clear

This command will clear the screen, draw a set of axes, and draw the data cross section at x=1.0 micron. Each point will be drawn with symbol 1.

plot.1d x.v=2.0 axis=f clear=f

This command draws a cross section at x=2.0 microns on the previous set of axis.

plot.1d sil /oxi

Plots the selected variable along the silicon interface facing oxide.

BUGS

The plot software uses gplot, any gplot bugs are likely to show up in this command. The command also, therefore, supports all of the devices available to gplot. If the wrong device is selected, nothing will happen on the screen. If you are lucky.

If arclength is set, and the interface between two materials comprises several disconnected pieces, the end of one arc is joined incorrectly to the start of the next.

If arclength is set, the interface between materials is usually ordered left-to-right, but it is possible to confuse the routine, and get right-to-left.

There is no way to specify a line at a fixed distance from an interface.

There is no simple way to follow the "upper" surface of a layer if it has several different layers on top of it.

SEE ALSO

option(2), select(2), gplot(5), plotcap(5)

PLOT.2D

plot.2d – plot a two dimensional xy picture

SYNOPSIS

plot.2d

[x.max=<n>] [x.min=<n>] [y.min=<n>] [y.max=<n>]
[zoom.in | zoom.out | pan]
[clear] [fill]
[boundary] [grid] [axis]
[line.grid] [line.bound]
[vornoi] [diamonds]
[stress] [vmax=<n>] [vleng=<n>] [line.com = <n>] [line.ten = <n>]
[flow]

SUPREM IV

Level: Commands

Author: Mark E. Law (Based on PISCES I)

Version: 3.4 9/28/88 15:33:58

DESCRIPTION

The plot.2d statement allows user to prepare for a two dimensional xy plot. This routine does not plot any solution values, however, it can prepare the screen for isoconcentration lines to be plotted. The statement can draw the x y space of the device, label it, and draw material boundaries.

x.min, x.max
: These parameters allow the user to specify a subset of the x axis to be plotted. The parameters will default to the current device limits. The units are in microns.

y.min, y.max
: These parameters allow the user to specify a subset of the y axis to be plotted. The parameters will default to the current device limits. The units are in microns.

[zoom.in | zoom.out | pan]
: This group of parameters overides the numeric bounds (x.min...y.max). If zoom.in is specified, the user is prompted in the graphics window for two corners of a box. The area inside the box is used as the plot window in the next drawing. The box is modified to maintain the proper aspect ratio unless "fill" (see below) is set to false. If zoom.out is specified, a window twice as large as the current window in both x and y is used. If pan is specified, the cursor should be pointed to the area of interest. The next window will have the same bounds as the current window but will be centered around the cursor position. The zoom.in and pan commands both require a device which can be queried for cursor location.

clear
: The clear parameter specifies whether the graphics screen should be cleared before the the graph is drawn. If true, the screen is cleared. It defaults to true.

fill
: The fill parameter specifies that the device should be drawn with the proper aspect ratio. If fill is false, the device will be drawn with the proper aspect ratio. When true, the device will be expanded to fill the screen. It defaults to false.

boundary
: This parameter specifies that the device outline and material interfaces should be drawn. They will be drawn with the color specified by the line.bound parameter. It defaults to off.

grid
: The grid parameter specifies that the numerical grid the problem was solved on should be drawn. The color will be with the color will be with the line.grid parameter. It defaults to off.

axis
: The axis parameter controls the drawing and labeling of a axes around the plot. If axis is true, the graph will be drawn with labeled axis. This parameter defaults true.

line.grid

The line.grid parameter controls the line type or color that the axis and grid will be drawn in. The parameter defaults to 1.

line.bound
: This parameter specifies the line color to be used for the material boundaries. If boundary is not specified, this parameter is unimportant.

vornoi
: This parameter specifies that the vornoi tesselation to be drawn. Anyone interested in finding out why someone would want to draw the vornoi tesselation, please mail to suprem4@oasis.stanford.edu.

diamonds
: This option will cause the plot to draw small diamonds out at each point location. Actually crosses are now plotted but the parameter name has not yet been changed, just to confuse the opposition.

stress
: This parameter plots the principal stresses throughout the structure. If you've never heard of the principal stress, don't worry, you don't have to plot them. Vectors are drawn along the two principal axes of the stress tensor at each point.

vleng
: The length of the vectors is scaled so that the maximum stress has length vleng, specified in microns.

line.com, line.ten
: The sign of the stress is indicated by the color of the vectors drawn. line.com is the color for compressive stress, line.ten is the color for tensile stress.

vmax
: If vmax is specified (in dynes/cm2), that stress will have length vleng, and higher stresses will be omitted.

flow
: Plots little arrows representing the velocity at each point. The vleng and vmax parameters help out here too.

EXAMPLES

plot.2d grid
: This command will draw the triangular grid and axis.

plot.2d bound x.min=2 x.max=5 clear=false
: This command will draw the material interfaces and axis between 2.0 and 5.0 microns and will not clear the screen first.

plot.2d bound line.bound=2 diamonds y.max=5 axis=false
: This command will draw the material interfaces with a maximum value of y equal to 5.0 microns. The boundaries will be drawn with line type 2, no axis, and will draw the grid points.

BUGS

The plot software uses gplot, any gplot bugs are likely to show up in this command. The command also, therefore, supports all of the devices available to gplot. If the wrong device is selected, nothing will happen on the screen. If you're lucky.

SEE ALSO

color(2), contour(2), option(2), select(2), gplot(5), plotcap(5)

PLOT.3D

plot.3d – plot a three dimensional picture

SYNOPSIS

plot.3d

[clear] [axis]
[bound] [line.b=<n>]
[grid] [line.g=<n>]
[azimuth=<n>] [elevation=<n>]
[num.cntr=<n>] [line.type=<n>]
[x.min=<n>] [x.max=<n>] [y.min=<n>] [y.max=<n>] [z.min=<n>] [z.max=<n>]

SUPREM IV

Level: Commands

Author: Mark E. Law
Zvonko Fazarinc (Prototype)

Version: 3.3 11/21/88 13:14:31

DESCRIPTION

The plot.3d statement allows the user to plot a bird's eye view of a three dimensional wire diagram of the selected data. The routine interpolates a series of cross section lines and plots them with the given view point parameters. Axes can be drawn and labeled if so desired by the user.

This routine is not extremely robust. Care has to be exercised in selecting parameters. Azimuths outside the range of 30 to 60 degrees can cause severe problems. Users should try to stay within 30 to 60, 120 to 150, 210 to 240, and 300 to 330. Values outside that range of azimuths are apt to create very unappealing plots at best and core dumps at worse.

clear The clear parameter specifies whether the graphics screen should be cleared before the the graph is drawn. If true, the screen is cleared. It defaults true.

axis This parameter is currently a no-op. It looks good here, though.

boundary
Whether or not to outline the boundary underneath the bird's eye view. Defaults to true.

line.b The line.b parameter controls the line type or color that the boundary is drawn in. The parameter defaults to 2.

grid Whether or not to outline the grid of the device underneath the bird's eye view. Defaults to false.

line.g The line.g parameter controls the line type or color that the grid is drawn in. The parameter defaults to 2.

azimuth This floating point parameter governs the angle from which the graph is viewed in the xy plane. 45 degrees is viewing with the maximum ends of the x and y axes closest to the viewer. This parameter has a limited legal range. Any value within 15 degrees of 45, 135, 225, and 315 degrees is probably all right. Outside this range, the algorithm degrades rapidly, and will segment fault, barf, and die at 0, 90, 180, and 270 degrees. The parameter defaults to 45 degrees.

elevation
This floating point parameter governs the angle above the xy plane the viewer is situated. If an angle of 0 is specified, the viewpoint is in the xy plane. Positive angles indicate above the xy plane and negative below it. All angles between -90 and 90 are legal values.

num.cntr
This floating point parameter governs the number of lines that will be drawn to make up the surface. It governs the level of resolution of the plot. One axis will have this number of lines, the other will be drawn with a number calculated from it dependent on the viewing azimuth. The number defaults to 20, but this may be insufficient for some data sets.

line.type

The line.type parameter controls the line type or color that the contours that describe the data are drawn in. The parameter defaults to 1.

x.min, x.max

These floating point parameters allow the user to specify a subset of the x axis to be plotted. The parameters will default to the current device limits. The units are in microns.

y.min, y.max

These floating point parameters allow the user to specify a subset of the y axis to be plotted. The parameters will default to the current device limits. The units are in microns.

z.min z.max

These floating point parameters allow the user to specify a subset of the z axis to be plotted. The parameters will default to the minimum and maximum values of the selected variable. The units depend on the selected variable.

EXAMPLES

plot.3d clear num.c=30 elev=90

This will plot the device as viewed from straight above. There will be 30 contour lines selected.

plot.3d azim=30 elev=60

This will plot the bird's eye view plot from 60 degrees up the horizon and 30 degrees off the x axis.

BUGS

The plot software uses gplot, any gplot bugs are likely to show up in this command. The command also, therefore, supports all of the devices available to gplot. If the wrong device is selected, nothing will happen on the screen. I consider it a bug to not support all the possible viewing angles. It also seems to interpolate funny onto the cross section lines at the boundary. Non-rectangular areas are padded out with zero values, so the final surface is still rectangular. The hidden line algorithm sometimes hides the wrong lines. Also note the strong caveats expressed above about the range of azimuths that are legal.

SEE ALSO

plot.2d(2), for(1), gplot(5), plotcap(5), select(2)

PRINT.1D

print.1d – print values along a one dimensional cross section

SYNOPSIS

print.1d
[(x.value=<n> | y.value=<n>)]
[silicon | oxide | nitride ...] [/exposed | /backside | /reflect | /silicon | /oxide | /nitride ...]
[arclength]
[layers]
[x.min=<n>] [x.max=<n>]
[format=<s>]

SUPREM IV

Level: Commands

Author: Mark E. Law Conor S. Rafferty

Version: 3.4 9/29/88 17:01:34

DESCRIPTION

The print.1d card allows the user to print the values along cross sections through the device. It is also possible to integrate along a specified line. The value printed is the value that has been selected, see select.

x.value, y.value
: This parameter specifies that the cross section is to be done at a constant value as specified by the value of this parameter. X.value specifies a vertical cross section of the device, and y.value a horizontal slice. This parameter is in microns. Only one of x.value or y.value can be specified for a given device.

silicon | oxide | nitride ...
: In addition to constant x or y cross-sections, a one-dimensional print can be specified along one side of an interface. The interface lies between material 1, named without a "/", and material 2, named with a "/". For example, "print.1 oxide /silicon". Thus "print.1 oxide /silicon" will usually show something different from "print.1 silicon /oxide". The backside, reflecting or exposed surfaces of a material can be specified with the appropriate parameter.

arclength
: This parameter only has meaning for printing along the interface. The ordinate printed is the arc length along the boundary from the leftmost point on the curve, in microns, if arclength is chosen. Otherwise the x value of the interface location is printed. The leftmost point has ordinate equal to its x coordinate in the mesh.

layers
: This option instructs that the selected plot variable should integrated in each material that is crossed. The integrated value and material width is reported. Zero crossings of the variable are treated the same as material interfaces. This option imitates the SUPREM-III command, "print layers", and is probably most useful when doping is the selected variable.

x.min, x.max
: These parameters allow the user to specify the limits of the print region. Only values between these two will be displayed.

format
: This allows the user to change the print format for the variable. It uses standard C format expressions (e.g. 8.2f), minus the % sign (guess why). If the user is unfamiliar with the C programming language, it is best to not experiment.

EXAMPLES

print.1d x.val=1.0 x.max=3.0
: This command will print the selected value at x equal to one micron between the top of the mesh and the 3.0 micron point.

print.1d y.val=0 layers

This will print the integrated value selected in each material layer crossed by a horizontal slice at depth of 0.0.

print.1d sil /oxi

This will print the selected variable along the silicon side of the silicon oxide interface.

BUGS

If the interface between two materials comprises several disconnected pieces, the end of one arc is joined incorrectly to the start of the next.
The interface between materials is usually ordered left-to-right, but it is possible to confuse the routine and get right-to-left.
There is no way to specify a line at a fixed distance from an interface.
There is no simple way to follow the "upper" surface of a layer if it has several different layers on top of it.

SEE ALSO

select(2), plot.1d(2), printf(2)

PRINTF

printf -- a string printer and desk calculator

SYNOPSIS

printf [string]

SUPREM IV

Level: SHELL

Author: Mark E. Law

Version: 1.1 9/29/88 09:28:45

DESCRIPTION

This command is similar to the echo command. The only difference is that it splits its arguments up into white space separated pieces and passes each piece separately to the real number parser. Both quotes and paranthesis can be used to print spaces. This command is most useful in connection with the movie parameter of the diffuse command.

EXAMPLES

printf jimmin at the jimjam frippin at the frotz

This will send the string "jimmin at the jimjam frippin at the frotz" to the standard output.

printf (15.0 - 12.0 * exp(4.0 - 2.0 / 6.0))

This will print the result of the arithmetic expression, which is of course, 8.373.

printf (15.0 * exp (-2.0 / (8.62e-5 * 1173.0)) (12.0 * sqrt(2.0))

This will print:
3.853e-8 16.97

printf "The Time is: " ${time}

Inside the movie parameter of the diffuse command, time is a variable that is set to the current time in the diffusion cycle. Consequently, if this is used there, the command will print:
The Time is: 0.1

printf xfn(0.0, 0.0) yfn(0.0, 0.0) zfn(0.0, 0.0)

This will print the x location where y and z are both zero, the y location where x and z are both zero, and the z location where x and y are both zero.

BUGS

No known bugs.

SEE ALSO

select(2) - for choosing z variables,
Intro(0) - for a description of the real number parser,
diffuse(2) - for examples of usage in the movie parameter.

PROFILE

profile -- read a one dimensional doping profile

SYNOPSIS

initialize

infile=<c>

(antimony | arsenic | boron | phosphorus)

SUPREM IV

Level: Commands

Author: Mark E. Law

Version: 1.1 10/5/88 11:18:18

DESCRIPTION

The profile allows the user to specify a file of one dimensional doping data. This command allows the user to read data from another program, *i.e.* SUPREM-III, or measured data. doping concentration. The new doping is added to any current doping that may exist. The file should have two columns of numbers, depth and concentration. The file should have the depth in microns, and the concentration in cm^{-3}. Any lines which do contain two values are ignored as comments.

infile The infile parameter selects the file name for data.

antimony, arsenic, boron, phosphorus

These parameters specify the type of impurity that forms the doping.

EXAMPLES

init infile=foo phos

This command reads in a doping file named foo, containing a phosphorus doping profile.

The file foo, for example, could look like:

```
#pulse doped buried phosphorus profile
0.2 1.0e5
0.21 1.0e19
0.30 1.0e19
0.31 1.0e5
```

Any area outside the specified doping layer is unchanged.

BUGS

There are no known bugs.

SEE ALSO

initialize, structure

REGION

region – Specify a mesh region

SYNOPSIS

region

(silicon | oxide | nitride | poly | gas | oxynitr | photores | aluminum)
xlo = <string>
ylo = <string>
xhi = <string>
yhi = <string>

SUPREM IV

Level: Commands

Author: Conor S. Rafferty

Version: 3.4 11/21/88 13:14:34

DESCRIPTION

This statement is used to specify the material of rectangles in a rectangular mesh. Region statements should follow line statements. Every triangle must be given some material, so at least one region statement is required for each rectangular mesh.

silicon | oxide | nitride | poly | gas | oxynitr | photores | aluminum
The material being specified.

xlo, ylo, xhi, yhi
The bounds of the rectangle being specified. The <string> value should be one of the tags created in a preceding line statement.

EXAMPLES

region silicon xlo=left xhi=right ylo=surf yhi=back
This line would make the entire mesh, from the example on the line page, silicon.

BUGS

If the user does not specify enough regions to describe the material of every part of the grid, it is sometimes not detected until a subsequent command is executed. This can be baffling.

SELECT

select – select the plot variable for the postprocessing routines

SYNOPSIS

select

[z=<expr>]
[label=<string>] [title=<string>]
[temp=<n>]

SUPREM IV

Level: Commands

Author: Mark E. Law

Version: 3.3 9/29/88 17:04:53

DESCRIPTION

The select statement is used to specify the variable for display in the contour statement, plot.1d statement, print.1d statement, plot.2d statement, plot.3d statement. In short, all of the postprocessing commands. No data will be available for any of these statements unless this statement is specified first. Only one variable can be selected at any one time, each select statement overrides any previous statements.

z This parameter accepts a vector expression of several different vector variables for the z parameter. The operators *,/,+,-,^ all work in the expected way. The vector variables are listed below:

Vector Variables

antimony	antimony concentration
arsenic	arsenic concentration
boron	boron concentration
cesium	cesium concentration
ci.star	equilibrium interstitial concentration
cv.star	equilibrium vacancies concentration
doping	net active concentration
electrons	electron concentration
gold	gold concentration
interstitial	interstitial concentration
ni	intrinsic electron concentration
oxygen	oxygen concentration
phosphorus	phosphorus concentration
psi	potential
Sxx, Sxy, Syy	components of stress in rectangular coordinates
trap	unfilled interstitial trap concentration
vacancy	vacancy concentration
x	x coordinates
x.v	x velocity
y	y coordinates
y.v	y velocity

Many of these are self explanatory. Potential is computed using charge neutrality. The electron concentration is computed from the potential using Boltzmann statistics.

There also several function that are available for the user. These are:

Vector Functions

abs	absolute value
active	active portion of the specified dopant
erf	error function
erfc	complimentary error function
exp	exponential

gradx	numerically differentiates the argument with respect to x location
grady	numerically differentiates the argument with respect to y location
log	logarithm
log10	logarithm base 10
mat1@mat2	returns the y value of the interface between mat1 and mat2 along a vertical slice at the given location.
scale	scales the value given by the maximum value
sqrt	square root

label This is the string that appears on the y axis in a 1d plot. The default is the select string.

title This is the string printed in large letters across the top of the plot. Default is the version number of the program.

temp The temperature at which expressions are evaluated. It defaults the last temperature of diffusion.

EXAMPLES

select z=log10(arsen)
Choose as the plot variable the base 10 logarithm of the arsenic concentration.

select z=(phos - 5.0e14)
Choose as the plot variable the phosphorus concentration minus a constant profile of 5.0e14.

select z=(phos - 1.0e18 * exp (y * y / 1.0e-8))
Choose as the plot variable the difference between the phosphorus and an analytic profile.

select z=(inter * vacan - ci.star * cv.star)
Choose the excess vacancy interstitial product as the plot variable.

BUGS

The version number needs to get updated more frequently.

SEE ALSO

all the other postprocessing commands.

STRESS

stress – calculate elastic stresses

SYNOPSIS

stress [temp1=<n> temp2=<n>] [nel=<n>]

SUPREM IV

Level: Suprem

Author: Conor S. Rafferty

Version: 1.2 9/29/88 17:06:56

DESCRIPTION

The stress card allows the user to calculate stresses due to thin film intrinsic stress or thermal mismatch.

temp1 temp2

These are the initial and final temperatures for calculating thermal mismatch stresses.

nel This is the number of nodes per triangle to use. Currently only 6 or 7 are allowed. 6 nodes is faster than 7 and usually gives adequate results, so 6 is the default.

EXAMPLES

```
material nitride intrin.sig=1.4e10
stress
```

This calculates the stresses in the substrate and film arising from a nitride layer which has an intrinsic stress of 1.4e10 when deposited uniformly.

BUGS

The correct boundary conditions for a thermal mismatch problem are hard to set up.

SEE ALSO

material(3)

STRUCTURE

structure – read/write the mesh and solution information

SYNOPSIS

structure
[(infile=<c> | outfile=<c>)]
[pisces=<c>] [show]
[mirror] [left] [right]

SUPREM IV

Level: Commands

Author: Conor S. Rafferty / Mark E. Law

Version: 3.3 11/21/88 13:14:18

DESCRIPTION

The structure statement allows the user to read and write the entire mesh and solution set. The data saved is from the current set of solution and impurity values.

infile, outfile
: These parameters specify the name of the file to be read or written. An existing file will be overwritten.

pisces
: This is how SUPREM-IV talks to PISCES. The file created is a pisces "geometry" file and is read with the "mesh infile=xxx geom" statement in PISCES (Sept. 87 version). Electrodes are generated at polysilicon and aluminum interfaces with either silicon or oxide, and at the backside. The electrode numbering is by SUPREM-IV region number, and is printed out. Materials that PISCES does not recognize are stripped from the output file. This includes polysilicon and aluminum layers.

show
: The numbering of electrodes is displayed by plotting the structure and labeling each electrode node with its electrode number.

mirror left right
: Mirrors the grid around its left or right boundary. Useful for turning half-a-MOSFET simulations into PISCES grids. The default is to reflect around the right axis.

EXAMPLES

structure infile=foo
: This command reads in a previously saved structure in file foo.

structure pisces=baz
: Writes a pisces file

BUGS

If no backside was specified for the initial structure, there is no way to get an electrode on the back side.
The "show" option echos a lot of nonsense while drawing the structure.
Beware of rounding errors when mirroring. If the left or right boundary is not smooth to within 0.1 angstroms, some points will be duplicated. Maybe I should make the tolerance a user-adjustable parameter?

SEE ALSO

initialize

VIEWPORT

viewport – select a subset of the plotting surface

SYNOPSIS

viewport

[x.min=<n>] [x.max=<n>] [y.min=<n>] [y.max=<n>]

SUPREM IV

Level: Suprem

Author: Mark E. Law

Version: 3.2 9/29/88 08:42:20

DESCRIPTION

This command allows the user to specify a subset of the current plotting device to plot on. It works with all the plot calls. The viewport remains in the current state until it is reset with a subsequent viewport call.

x.min, x.max

These floating point parameters allow the user to specify a subset of the x axis of the plot surface to be selected. The values are specified in the range of zero to one, with zero being the axis minimum and one being the axis maximum

y.min, y.max

These floating point parameters allow the user to specify a subset of the y axis of the plot surface to be selected. The values are specified in the range of zero to one, with zero being the axis minimum and one being the axis maximum

BUGS

The clear options of the various plot commands clear the whole screen, not just the current viewport area. The viewport command does not take effect until the next axis is drawn.

SEE ALSO

gplot(5), plotcap(5)

SUPREM IV

Level: Models

Version: 3.5 12/7/88 08:10:08

DESCRIPTION

This section contains the commands used to change model parameters and coefficients. The parameters are described briefly on the statement descriptions. The file for the initialization of all the constants is human readable. Unfortunately, this means that SUPREM IV is slow starting up because it has to parse the file.

The model statements can be broken up by diffusion impurity. All the parameters concerning arsenic diffusion, segregation, and clustering are on the same statement. The oxide statement contains some material information as well as parameters for oxidant diffusion.

BUGS

The cards implement the best physics we have worked out to date. However, that is a day to day proposition.

SEE ALSO

See the other model statement descriptions for more detail. Most of the model statements will include references to the relevant published papers. Please see these Stanford Ph.D. theses for more information.
Peter Griffin - Defect Models and Diffusion, Experiments
Mark Law - Defect Models and Diffusion, Numerical Techniques
Conor Rafferty - Oxidation Models and Stress

ANTIMONY

antimony – set the coefficients of antimony kinetics

SYNOPSIS

antimony
(silicon | oxide | oxynit | nitride | gas | poly)
[Dix.0=<n>] [Dix.E=<n>]
[Dim.0=<n>] [Dim.E=<n>]
[Fi = <n>]
[ss.clear] [ss.temp=<n>] [ss.conc=<n>]
[(/silicon | /oxide | /oxynitr | /nitride | /gas | /poly)]
[Seg.0=<n>] [Seg.E=<n>] [Trn.0=<n>] [Trn.E=<n>]

SUPREM IV

Level: Models

Author: Mark E. Law

Version: 3.2 9/30/88

DESCRIPTION

This statement allows the user to specify values for coefficients of antimony diffusion and segregation. The diffusion equation for antimony is:

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_T \frac{C_V}{C_V^*}\nabla\ln\left(C_T \frac{C_V}{C_V^*}\frac{n}{n_i}\right) + D_I C_T \frac{C_I}{C_I^*}\nabla\ln\left(C_T \frac{C_I}{C_I^*}\frac{n}{n_i}\right)\right]$$

where C_T is the total chemical concentration of antimony, C_V and C_I are the vacancy and interstitial concentration, C_V^* and C_I^* are the equilibrium vacancy and interstitial concentration, D_V and D_I are the diffusivities with vacancies and interstitials, and n and n_i refer to the electron concentration and the intrinsic concentration respectively. [1] The diffusivities are given by:

$$D_V = (1-Fi)\left[Dix + Dim\frac{n}{n_i}\right]$$

$$D_I = Fi\left[Dix + Dim\frac{n}{n_i}\right]$$

Dix and *Dim* are described in greater detail below.

The segregation at material interfaces is computed using the following expression:

$$flux = Trn\left[C_1 - \frac{C_2}{Seg}\right]$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the *Seg* and *Trn* terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly

These allow the specification of parameters for that material. Only one of these can be specified per statement. The parameters specified in that statement will apply in the material listed. These parameters specify which material is material 1 for the segregation terms.

Dix.0, Dix.E

These floating point parameters allows the specification of the antimony diffusing with neutral defects. Dix.0 is the pre-exponential constant and Dix.E is the activation energy. Dix.0 defaults to 0.214 centimeters squared per second in silicon, and Dix.E defaults to 3.65eV in silicon [2]. The Dix term is calculated using:

$$Dix = Dix.0 \exp\left(-\frac{Dix.E}{kT}\right)$$

Dim.0, Dim.E

These floating point parameters allows the specification of the antimony diffusing with singly negative defects. Dim.0 is the pre-exponential constant and Dim.E is the activation energy. Dim.0 defaults to 15.0 centimeters squared per second in silicon, and Dim.E defaults to 4.08eV in silicon [2]. The Dim term is calculated using:

$$Dim = Dim.0 \exp\left(-\frac{Dim.E}{kT}\right)$$

Fi This parameter allows the specification of the fractional interstitialcy. This value indicates whether antimony diffuses through interaction with interstitials or vacancies. The value of Fi defaults to 0.05 [3].

ss.clear This parameter clears the currently stored solid solubility data.

ss.temp, ss.conc

These parameters add a single temperature solid solubility concentration point to those already stored. The default values are [4]:

Solid Solubility Data

800°C	$2.3 \cdot 10^{19}$
900°C	$3.0 \cdot 10^{19}$
1000°C	$4.0 \cdot 10^{19}$
1100°C	$4.8 \cdot 10^{19}$
1250°C	$6.2 \cdot 10^{19}$

/silicon, /oxide, /oxynit, /nitride, /gas, /poly

These parameters specify material 2. Only one of the these parameters can be specified at one time.

Seg.0, Seg.E

These parameters allow the computation of the equilibrium segregation concentrations. The formula used to calculate seg from material 1 to material 2 is:

$$seg = Seg.0 \exp\left(-\frac{Seg.E}{kT}\right)$$

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/s. The formula used to calculate trn is

$$trn = Trn.0 \exp\left(-\frac{Trn.E}{kT}\right)$$

EXAMPLES

antimony silicon Dix.0=0.214 Dix.E=3.65

This command changes the neutral defect diffusivity in silicon.

antimony silicon /oxide Seg.0=30.0 Trn.0=1.66e-7

This command will change the segregation parameters at the Silicon - Silicon Dioxide interface. The silicon concentration will be 30.0 times the oxide concentration in equilibrium.

BUGS

As far as the implemented models are physically correct, there are no known bugs.

REFERENCES

1. D. Mathiot and J.C. Pfister, "Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects," *Journal of Applied Physics*, vol. 55 No. 10, p. 3518, May 15, 1984.

2. R.B. Fair, "Concentration Profiles of Diffused Dopants in Silicon," in *Impurity Doping Processes in Silicon*, ed. F. F. Y. Yang, p. 315, North-Holland, New York, 1981.

3. P. Fahey, G. Barbuscia, M. Moslehi, and R.W. Dutton, "Kinetics of Thermal Nitridation Processes in the Study of Dopant Diffusion Mechanisms in Silicon," *Appl. Phys. Lett.*, vol. 46 (8), p. 784, April 1985.

4. F. A Trumbore, "Solid Solubilities of Impurity Elements in Germanium and Silicon," *Bell System Tech Journal*, Jan. 1960.

SEE ALSO

arsenic, boron, phosphorus, intersititial, vacancy

ARSENIC

arsenic – set the coefficients of arsenic kinetics

SYNOPSIS

arsenic

(silicon | oxide | oxynit | nitride | gas | poly)
[Dix.0=<n>] [Dix.E=<n>]
[Dim.0=<n>] [Dim.E=<n>]
[Fi = <n>]
[Ctn.0=<n>] [Ctn.E=<n>]
[(/silicon | /oxide | /oxynitr | /nitride | /gas | /poly)]
[Seg.0=<n>] [Seg.E=<n>] [Trn.0=<n>] [Trn.E=<n>]

SUPREM IV

Level: Models

Author: Mark E. Law

Version: 3.2 9/30/88

DESCRIPTION

This statement allows the user to specify values for coefficients of arsenic diffusion and segregation. The diffusion equation for arsenic is:

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_A \frac{C_V}{C_V^*}\nabla\ln\left[C_A \frac{C_V}{C_V^*}\frac{n}{n_i}\right] + D_I C_A \frac{C_I}{C_I^*}\nabla\ln\left[C_A \frac{C_I}{C_I^*}\frac{n}{n_i}\right]\right]$$

where C_T is the total chemical concentration of arsenic, C_A is the total electrically active concentration of arsenic, C_V and C_I are the vacancy and interstitial concentration, C_V^* and C_I^* are the equilibrium vacancy and interstitial concentration, D_V and D_I are the diffusivities with vacancies and interstitials, and n and n_i refer to the electron concentration and the intrinsic concentration respectively. [1] The diffusivities are given by:

$$D_V = (1-Fi)\left[Dix + Dim\frac{n}{n_i}\right]$$

$$D_I = Fi\left[Dix + Dim\frac{n}{n_i}\right]$$

Dix and *Dim* are described in greater detail below.

The segregation at material interfaces is computed using the following expression:

$$flux = Trn\left[C_1 - \frac{C_2}{Seg}\right]$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the *Seg* and *Trn* terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly

These allow the specification of parameters for that material. Only one of these can be specified per statement. The parameters specified in that statement will apply in the material listed. These parameters specify which material is material 1 for the segregation terms.

Dix.0, Dix.E

These floating point parameters allows the specification of the arsenic diffusing with neutral defects. Dix.0 is the pre-exponential constant and Dix.E is the activation energy. Dix.0 defaults to 8.0 centimeters squared per second in silicon, and Dix.E defaults to 4.05eV in silicon [2]. The Dix term is calculated using:

$$Dix = Dix.0 \exp\left[-\frac{Dix.E}{kT}\right]$$

Dim.0, Dim.E

These floating point parameters allows the specification of the arsenic diffusing with singly negative defects. Dim.0 is the pre-exponential constant and Dim.E is the activation energy. Dim.0 defaults to 12.8 centimeters squared per second in silicon, and Dim.E defaults to 4.05eV in silicon [2]. The Dim term is calculated using:

$$Dim = Dim.0 \exp\left[-\frac{Dim.E}{kT}\right]$$

Fi This parameter allows the specification of the fractional interstitialcy. This value indicates whether arsenic diffuses through interaction with interstitials or vacancies. The value of Fi defaults to 0.20 [3].

Ctn.0, Ctn.E

These parameters specify the clustering coefficients for arsenic. The parameters are respectively the pre-exponential coefficient and the activation energy. The total cluster coefficient is:

$$Ctn = Ctn.0 \exp\left(\frac{Ctn.E}{kT}\right)$$

The active concentration, C_A is calculated using:

$$C_T = C_A\left[1 + \left[Ctn\frac{n}{n_i}\right]^2 \frac{C_V}{C_V^*}\right]$$

Physically, this represents clustering of arsenic with doubly negative vacancies. The default parameters are Ctn.0 = 5.9e-24 and Ctn.E=0.60eV in silicon [2].

/silicon, /oxide, /oxynit, /nitride, /gas, /poly

These parameters specify material 2. Only one of the these parameters can be specified at one time.

Seg.0, Seg.E

These parameters allow the computation of the equilibrium segregation concentrations. The formula used to calculate seg from material 1 to material 2 is:

$$seg = Seg.0 \exp\left[-\frac{Seg.E}{kT}\right]$$

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/s. The formula used to calculate trn is

$$trn = Trn.0 \exp\left[-\frac{Trn.E}{kT}\right]$$

EXAMPLES

arsenic silicon Dix.0=8.0 Dix.E=4.05

This command changes the neutral defect diffusivity in silicon.

arsenic silicon /oxide Seg.0=30.0 Trn.0=1.66e-7

This command will change the segregation parameters at the Silicon - Silicon Dioxide interface. The silicon concentration will be 30.0 times the oxide concentration in equilibrium.

BUGS

As far as the implemented models are physically correct, there are no known bugs.

REFERENCES

1. D. Mathiot and J.C. Pfister, "Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects," *Journal of Applied Physics*, vol. 55 No. 10, p. 3518, May 15, 1984.

2. R.B. Fair, "Concentration Profiles of Diffused Dopants in Silicon," in *Impurity Doping Processes in Silicon*, ed. F. F. Y. Yang, p. 315, North-Holland, New York, 1981.

3. P. Fahey, G. Barbuscia, M. Moslehi, and R.W. Dutton, "Kinetics of Thermal Nitridation Processes in the Study of Dopant Diffusion Mechanisms in Silicon," *Appl. Phys. Lett.*, vol. 46 (8), p. 784, April 1985.

SEE ALSO

arsenic, boron, phosphorus, intersititial, vacancy

BORON

boron – set the coefficients of boron kinetics

SYNOPSIS

boron

(silicon | oxide | oxynit | nitride | gas | poly)
[Dix.0=<n>] [Dix.E=<n>]
[Dip.0=<n>] [Dip.E=<n>]
[Fi = <n>]
[ss.clear] [ss.temp=<n>] [ss.conc=<n>]
[(/silicon | /oxide | /oxynitr | /nitride | /gas | /poly)]
[Seg.0=<n>] [Seg.E=<n>] [Trn.0=<n>] [Trn.E=<n>]

SUPREM IV

Level: Models

Author: Mark E. Law

Version: 3.2 9/30/88

DESCRIPTION

This statement allows the user to specify values for coefficients of boron diffusion and segregation. The diffusion equation for boron is:

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_A \frac{C_V}{C_V^*}\nabla\ln\left[C_A \frac{C_V}{C_V^*}\frac{p}{n_i}\right] + D_I C_A \frac{C_I}{C_I^*}\nabla\ln\left[C_A \frac{C_I}{C_I^*}\frac{p}{n_i}\right]\right]$$

where C_T is the total chemical concentration of boron, C_A is the total electrically active concentration of boron, C_V and C_I are the vacancy and interstitial concentration, C_V^* and C_I^* are the equilibrium vacancy and interstitial concentration, D_V and D_I are the diffusivities with vacancies and interstitials, and p and n_i refer to the hole concentration and the intrinsic electron concentration respectively. [1] The diffusivities are given by:

$$D_V = (1-Fi)\left[Dix + Dip\frac{p}{n_i}\right]$$

$$D_I = Fi\left[Dix + Dip\frac{p}{n_i}\right]$$

Dix and *Dip* are described in greater detail below.

The segregation at material interfaces is computed using the following expression:

$$flux = Trn\left[C_1 - \frac{C_2}{Seg}\right]$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the *Seg* and *Trn* terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly

These allow the specification of parameters for that material. Only one of these can be specified per statement. The parameters specified in that statement will apply in the material listed. These parameters specify which material is material 1 for the segregation terms.

Dix.0, Dix.E

These floating point parameters allows the specification of the boron diffusing with neutral defects. Dix.0 is the pre-exponential constant and Dix.E is the activation energy. Dix.0 defaults to 0.28 centimeters squared per second in silicon, and Dix.E defaults to 3.46eV in silicon [2]. The Dix term is calculated using:

$$Dix = Dix.0 \exp\left(-\frac{Dix.E}{kT}\right)$$

Dip.0, Dip.E

These floating point parameters allows the specification of the boron diffusing with singly positive defects. Dip.0 is the pre-exponential constant and Dip.E is the activation energy. Dip.0 defaults to 0.23 centimeters squared per second in silicon, and Dip.E defaults to 3.46eV in silicon [2]. The Dip term is calculated using:

$$Dip = Dip.0 \exp\left(-\frac{Dip.E}{kT}\right)$$

Fi This parameter allows the specification of the fractional interstitialcy. This value indicates whether boron diffuses through interaction with interstitials or vacancies. The value of Fi defaults to 0.94 [3].

ss.clear This parameter clears the currently stored solid solubility data.

ss.temp, ss.conc

These parameters add a single temperature solid solubility concentration point to those already stored. The default values are [4]:

Solid Solubility Data

800°C	$3.4499 \cdot 10^{19}$	1050°C	$1.5331 \cdot 10^{20}$
825°C	$4.1291 \cdot 10^{19}$	1075°C	$1.7263 \cdot 10^{20}$
850°C	$4.9027 \cdot 10^{19}$	1100°C	$1.9356 \cdot 10^{20}$
875°C	$5.7777 \cdot 10^{19}$	1125°C	$2.1613 \cdot 10^{20}$
900°C	$6.7615 \cdot 10^{19}$	1150°C	$2.4041 \cdot 10^{20}$
925°C	$7.8610 \cdot 10^{19}$	1175°C	$2.6643 \cdot 10^{20}$
950°C	$9.0832 \cdot 10^{19}$	1200°C	$2.9423 \cdot 10^{20}$
975°C	$1.0435 \cdot 10^{20}$	1225°C	$3.2387 \cdot 10^{20}$
1000°C	$1.1922 \cdot 10^{20}$	1250°C	$3.5536 \cdot 10^{20}$
1025°C	$1.3552 \cdot 10^{20}$	1275°C	$3.8876 \cdot 10^{20}$

/silicon, /oxide, /oxynit, /nitride, /gas, /poly

These parameters specify material 2. Only one of the these parameters can be specified at one time.

Seg.0, Seg.E

These parameters allow the computation of the equilibrium segregation concentrations. The formula used to calculate seg from material 1 to material 2 is:

$$seg = Seg.0 \exp\left(-\frac{Seg.E}{kT}\right)$$

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/s. The formula used to calculate trn is

$$trn = Trn.0 \exp\left(-\frac{Trn.E}{kT}\right)$$

EXAMPLES

boron silicon Dix.0=0.28 Dix.E=3.46

This command changes the neutral defect diffusivity in silicon.

boron silicon /oxide Seg.0=1126.0 Seg.E=0.91 Trn.0=1.66e-7

This command will change the segregation parameters at the Silicon - Silicon Dioxide interface. The silicon concentration will be 30.0 times the oxide concentration in equilibrium.

BUGS

As far as the implemented models are physically correct, there are no known bugs.

REFERENCES

1. D. Mathiot and J.C. Pfister, "Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects," *Journal of Applied Physics*, vol. 55 No. 10, p. 3518, May 15, 1984.

2. R.B. Fair, "Concentration Profiles of Diffused Dopants in Silicon," in *Impurity Doping Processes in Silicon*, ed. F. F. Y. Yang, p. 315, North-Holland, New York, 1981.

3. P. Fahey, G. Barbuscia, M. Moslehi, and R.W. Dutton, "Kinetics of Thermal Nitridation Processes in the Study of Dopant Diffusion Mechanisms in Silicon," *Appl. Phys. Lett.*, vol. 46 (8), p. 784, April 1985.

4. A. Armigliato, D. Nobili, P. Ostoja, M. Servidori, and S. Solmi, "Solubility and Precipitation of Boron in Silicon and Supersaturation Resulting by Thermal Predeposition," *Journal of Applied Physics*.

SEE ALSO

antimony, arsenic, phosphorus, intersititial, vacancy

INTERSTITIAL

interstitial – set coefficients of interstitial kinetics

SYNOPSIS

interstitial

(silicon | oxide | poly | oxynitr | nitride | gas)
[D.0=<n>] [D.E=<n>]
[Kr.0=<n>] [Kr.E=<n>]
[Cstar.0=<n>] [Cstar.E=<n>]
[ktrap.0=<n>] [ktrap.E=<n>]
[neu.0=<n>] [neu.E=<n>]
[neg.0=<n>] [neg.E=<n>] [dneg.0=<n>] [dneg.E=<n>]
[pos.0=<n>] [pos.E=<n>] [dpos.0=<n>] [dpos.E=<n>]
[(/silicon | /oxide | /poly | /oxynitr | /nitride | /gas)]
[time.inj] [growth.inj] [recomb]
[Ksurf.0=<n>] [Ksurf.E=<n>] [Krat.0=<n>] [Krat.E=<n>]
[Kpow.0=<n>] [Kpow.E=<n>]
[vmole=<n>] [theta.0=<n>] [theta.E=<n>]
[Gpow.0=<n>] [Gpow.E=<n>]
[A.0=<n>] [A.E=<n>] [t0.0=<n>] [t0.E=<n>]
[Tpow.0=<n>] [Tpow.E=<n>]
[rec.str=<s>] [inj.str=<s>]
[(antimony | arsenic | boron | phosphorus)]

SUPREM IV

Level: Models

Author: Mark E. Law

Version: 3.3 12/6/88

DESCRIPTION

This statement allows the user to specify values for coefficients of interstitial continuity equation. The statement allows coefficients to be specified for each of the materials. Of course, the meaning of a lattice interstitial in an amorphous region is a philosophical topic. SUPREM-IV has default values only for silicon and the interfaces with silicon. Polysilicon has not been characterized as extensively, and its parameters default to those for silicon.

The interstitials obey a complex diffusion equation that was first described by Mathiot and Pfister [1]. This equation can be written (although with different notation than Mathiot and Pfister):

$$\frac{\partial}{\partial t}\left[C_I - C_{ET} + \sum_{imp,c} k_{AI} C_A C_I\right] = \nabla\left[-J_I - \sum_{imp} J_{AI}\right] - R$$

where C_I is the interstitial concentration, C_{Et} is the number of empty interstitial traps, the first sum is over all impurities A and charge states c, J_I refers to the interstitial flux, J_{AI} refers to the flux of impurity A diffusing with interstitials, and R is all sources of bulk recombination of interstitials. In the two.dim model (see method(2)) the fluxes from the dopants, J_{AI} are ignored. Only in the full.cpl model are the computed and included in the interstitial equation. This model represents the diffusion of all interstitials, paired or unpaired, and can be derived from assuming thermal equilibrium between the species and that the simple pairing reactions are dominant.

The trap reaction was described by Griffin [2]. This model explains some of the wide variety of diffusion coefficients extracted from several different experimental conditions. The trap equation is:

$$\frac{\partial C_{ET}}{\partial t} = -K_T\left[C_{ET}C_I - \frac{e^*}{1-e^*}C_I^*(C_T - C_{ET})\right]$$

where C_T is the total trap concentration, K_T is the trap reaction coefficient, e^* is the equilibrium trap

occupancy ratio. (see Trap(3) for a more complete discussion). Instead of the reaction, the time derivative is used in the interstitial equation because it has better properties in the numerical calculation.

The pair fluxes are the interstitial portions of the flux as described in each of the models for the impurities. (see antimony(3), arsenic(3), boron(3), and phosphorus(3)) The interstitial flux can be written [3, 1]:

$$-J_I = D_I C_I^* \nabla\left[\frac{C_I}{C_I^*}\right]$$

It is important to note that the equilibrium concentration, C_I^* is a function of fermi level [4]. This flux accounts correctly for the effect of an electric field on the charged portion of the defect concentration. The bulk recombination is simple interaction between interstitials and vacancies. This can be expressed:

$$R = K_R\left[C_I C_V - C_I^* C_V^*\right]$$

where K_R is the bulk recombination coefficient, and C_V and C_V^* are the vacancy and vacancy equilibrium concentration, respectively.

The defects obey a flux balance boundary condition, as described by Hu [5]:

$$J_I \cdot \mathbf{n} + \mathrm{K_{surf}}\left[\mathrm{C_I} - \mathrm{C_I^*}\right] = \mathrm{G}$$

where $\mathbf{n}$ is the surface normal, K_{surf} is the surface recombination constant, and G is the generation, if any, at the surface. This expression has been used with success on several different surface types.

In all cases for the parameters, the value can be determined by using a simple singly activated model:

$$X = X.0 \exp\left[-\frac{X.E}{KT}\right]$$

(silicon | oxide | poly | oxynitr | nitride | gas)
These parameters allow the specification of the material for which the parameters apply.

D.0, D.E
These floating point parameters are used to specify the diffusion coefficient of the interstitials. The units are in cm^2 / s. The default values are 600.0 cm^2 / s and 2.44eV [6].

Kr.0, Kr.E
These floating point parameters allow the specification of the bulk recombination rate $cm^{-3}s^{-1}$. The bulk recombination value was extracted using experiments from Packan [7], and Guerrero *et al.* [8]. The values that best fit the data are $3.16 \cdot 10^{-6}$ $cm^{-3}s^{-1}$ and 2.44eV.

Cstar.0, Cstar.E
These parameters allow the specification of the total equilibrium concentration of interstitials in intrinsically doped conditions. The default values are $5.0 \cdot 10^{22}$ cm^{-3} and 2.36eV [6].

ktrap.0, ktrap.E
These values allow the specification of the trap reaction rate. At present, it is very difficult to extract exact values for these parameters. The default values assume that the trap reaction is limited by the interstitial concentration. The trap coefficient is essentially infinity.

neu.0, neu.E, neg.0, neg.E, dneg.0, dneg.E, pos.0, pos.E, dpos.0, dpos.E
These values allow the user to specify the relative concentration of interstitials in the various charge states (neutral, negative, double negative, positive, double positive) under intrinsic doping conditions. In intrinsic conditions, the concentration of doubly negative interstitials is given by:

$$C_I^{=} = \frac{Cstar \cdot dneg}{neu + neg + dneg + pos + dpos}$$

where Cstar is the total equilibrium concentration in intrinsic doping conditions. In extrinsic conditions, the total equilibrium concentration is given by [4]:

$$C_I^* = Cstar \frac{neu + neg\frac{n}{n_i} + dneg\frac{n^2}{{n_i}^2} + pos\frac{p}{n_i} + dpos\frac{p^2}{{n_i}^2}}{neu + neg + dneg + pos + dpos}$$

There is very little data for the interstitial charge states, it is matter of ongoing research. The default values are chosen based on an analysis of Giles [9]. These parameters take on a different meaning when an impurity is specified. This is described later in this document under the description of the impurity parameters.

(/silicon | /oxide | /poly | /oxynitr | /nitride | /gas)

These parameters allow the specification of the interface for which the boundary condition parameters apply. For example, parameters about the oxide silicon effect on interstitials should be specified by listing silicon /oxide. Only one of these can be specified at a time.

time.inj, growth.inj, recomb

These parameters specify the type of reactions occurring at the specified interface. The time.inj parameter means that a time dependent injection model should be chosen. The growth.inj parameter ties the injection to the interface growth velocity. The recomb parameter indicates a finite surface recombination velocity.

Ksurf.0, Ksurf.E, Krat.0, Krat.E, Kpow.0, Kpow.E

These parameters allow the specification of the surface recombination velocity. The formula used in computing the actual recombination velocity is:

$$K_s = Ksurf\left[Krat\left(\frac{v_i}{B/A}\right)^{Kpow} + 1\right]$$

where v_i is the interface velocity, and B/A is the Deal Grove oxide growth coefficient. (see oxide(3)). The full model applies only to oxide, since it is the only interface that can grow in SUPREM-IV. This formulation allows an inert interface to have different values than a growing interface. For inert interfaces, (gas, oxynitride, nitride, poly) Ksurf is the only important parameter.

vmole, theta.0, theta.E, Gpow.0, Gpow.E

These parameters allow the specification of generation that is dependent on the growth rate of the interface. The formula is:

$$G = \theta \cdot vmole \cdot v_i \left(\frac{v_i}{B/A}\right)^{Gpow}$$

Theta is the fraction of silicon atoms consumed during growth that are reinjected as interstitials. Once again, this model only makes since for oxide since it is currently the only material which can have a growth velocity.

A.0, A.E, t0.0, t0.E, Tpow.0, Tpow.E

These parameters allow an injection model with a fairly flexible time dependency. The equation for the injection is :

$$G = A\left(t + t0\right)^{Tpow}$$

where t is the time in the diffusion in seconds. This can be used to represent the injection of interstitials from an oxynitride layer.

rec.str, inj.str

These two string parameters are useful for experimenting with new models for injection or recombination at interfaces. Three macros are defined for use, t the time in seconds and x and y the coordinates. If these are specified, they are used in place of any other model. For example,

interst silicon /oxide inj.str = (10.0e4 * exp(t / 10.0))

describes an injection at the silicon oxide interface that exponentially decays in time.

(antimony | arsenic | boron | phosphorus)

These parameters allow the user to specify the a dopant. If a dopant is specified, the user can then specify the equilibrium reaction coefficient using the neutral, negative, double, and positive parameters. For example, to specify the value of K sub {BI sup 0}, the reaction coefficient for boron and neutral interstitials:

interst silicon boron neu.0=1.0e-18 neu.E=0.0

All of these coefficients are next to impossible to determine experimentally, and the default for them is zero. This basically states that the number of impurity interstitial pairs is negligible compared to the number of free interstitials.

EXAMPLES

inter silicon Di.0=5.0e-7 D.E=0.0 Cstar.0=1.0e13 Cstar.E=0.0

This statement specifies the silicon diffusion and equilibrium values for interstitials.

inter silicon /oxide growth vmole=5.0e22 theta.0=0.01 theta.E=0.0

This parameter specifies the oxide - silicon interface injection is to be computed using the oxide growth velocity and with 1% of consumed silicon injected as interstitials.

inter silicon /nitride Ksurf.0=3.5e-3 Ksurf.E=0.0 Krat.0=0.0

This specifies that the surface recombination velocity at the nitride silicon interface is 3.5e-3 cm/s.

interst silicon neu.0=1.0 neg.0=1.0 pos.0=0.0 dneg.0=0.0 dpos.0=0.0 interst silicon neu.E=0.0 neg.E=0.0 pos.E=0.0 dneg.E=0.0 dpos.0=0.0

This specifies that there are equal numbers of negative and neutral charged interstitials in intrinsic doping.

BUGS

There are no known bugs in this routine. However, the models used here are involved in ongoing research, and may by out of data at any time. Further, experimental verification is difficult. Some of the default parameters may not be appropriate for any process line other than Stanford's. Many of the parameters have unknown dependencies on stress, temperature, starting silicon material, stacking fault density, etc. High concentration phosphorus diffusion is promising with these models, but the coefficients have not been extracted to allow good fits to experiment.

REFERENCE

1. D. Mathiot and J.C. Pfister, "Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects," *Journal of Applied Physics*, vol. 55 No. 10, p. 3518, May 15, 1984.

2. P.B. Griffin and J.D. Plummer, "Process Physics Determining 2-D Impurity Profiles in VLSI Devices," *International Electron Devices Meeting*, p. 522, Los Angeles, Dec., 1986.

3. M.E. Law, "Two Dimensional Numerical Simulation of Dopant Diffusion in Silicon," PhD dissertation, Department of Electrical Engineering, Stanford University, Jan. 1988.

4. W. Shockley and J.T. Last, "Statistics of the Charge Distribution for a Localized Flaw in a Semiconductor," *Phys. Rev.*, vol. 107 No. 2, p. 392, July 15, 1957.

5. S.M. Hu, "On Interstitial and Vacancy Concentrations in Presence of Injection," *J. Appl. Phys.*, vol. 57, p. 1069, 1985.

6. G.B. Bronner and J.D. Plummer, "Gettering of Gold in Silicon: A Tool for Understanding the Properties of Silicon Interstitials," *Journal of Applied Physics*, vol. 61 No. 12, p. 5286, June 15, 1987.

7. P. Packan, *Private Communication.*

8. E. Guerrero, W. Jungling, H. Potzl, U. Gosele, L. Mader, M. Grasserbauer, and G. Stingeder, "Determination of the Retarded Diffusion of Antimony by SIMS measurements and numerical simulations," *Journal Electrochemical Society* , p. 2182, Oct., 1986.

9. M.D. Giles, "Defect Coupled Diffusion at High Concentrations," *IEEE Transactions on Computer Aided Design*, To be Published.

SEE ALSO

antimony(3), arsenic(3), boron(3), phosphorus(3), trap(3), vacancy(3)

MATERIAL

material – set the coefficients of some materials

SYNOPSIS

interstitial
(silicon | oxide | poly | oxynitr | nitride | photores | aluminu)
[(wet | dry)]
[Ni.0=<n>] [Ni.E=<n>] [Ni.Pow=<n>]
[eps=<n>]
[visc.0=<n> visc.E=<n> visc.x=<n>]
[Young.m=<n> Poiss.r=<n>]
[lcte=<c>]
[intrin.sig=<n>]

SUPREM IV

Level: Models

Author: MEL, CSR

Version: 3.2 10/5/88

DESCRIPTION

This statement allows the user to specify values for coefficients of the intrinsic concentration and relative permittivitty for all the materials.

silicon oxide oxynitr nitri poly photores aluminu
: These parameters indicate the material type that the remainder of the parameters apply in.

wet | dry
: These parameters indicate whether the parameters apply to in wet or dry oxides. When oxide is specified, it also necessary to specify the type of oxide.

Ni.0 Ni.E Ni.Pow
: These parameters specify the dependencies of the intrinsic electron concentration as a function of temperature.

$$n_i = Ni.0 \exp(\frac{Ni.E}{KT}) \; T^{Ni.pow}$$

Ni.0 is the pre-exponential term, Ni.E is the activation energy, and Ni.pow is the power dependence of temperature. The default in silicon is Ni.0 = 3.9e16, Ni.E=0.605 eV, and Ni.Pow = 1.5.

eps
: This parameter allows the specification of the relative permittivity of the different materials.

visc.0 visc.E visc.x
: These are the parameters specifying viscosity. visc.0 is the pre-exponential coefficient, visc.E is the activation energy. visc.x is the incompressibility factor. Normally this is left at the value of 0.499 (0.5 is infinitely incompressible), but if it is desired to allow more compressibility, it can be lowered.

Young.m Poiss.r
: Young.m is the material's Young's modulus, in dynes/cm2. Poiss.r is the material's Poisson ratio.

lcte
: This is an expression giving the linear coefficient of thermal expression as a function of temperature, called T in the expression. It is given as a fraction, not as a percentage.

intrin.sig
: This parameter specifies the initial uniform stress state of a material such as a thin film of nitride deposited on the substrate. It can be specified as a function of temperature by using an expression and the variable T. (see examples)

EXAMPLES

material silicon eps = 11.9

This statement specifies the silicon relative permittivity.

material nitride lcte=(3e-6 + 2*1e-10 * T) intrin.sig=1.4e10 Young.m=3e12

This statement gives the thermal expansion coefficient of nitride as a function of absolute temperature T. Thus at 0K the coefficient is .0003%/K. The initial stress in the nitride film is 1.4e10 dynes/cm2 and the Young's modulus is 3e12 dynes/cm2.

BUGS

There is limitation in the kind of temperature dependence Ni can take on. It is limited to expressions of the form above. It should be done in the same fashion as the coefficient of expansion.

SEE ALSO

oxide(3), stress(2), diffuse(2)

OXIDE

oxide – Specify oxidation coefficients

SYNOPSIS

oxide

orientation
dry | wet
[lin.l.0 lin.l.e lin.h.0 lin.h.e l.break l.pdep]
[par.l.0 par.l.e par.h.0 par.h.e p.break p.pdep]
[ori.dep ori.fac]
[thinox.0 thinox.e thinox.l]
[hcl.pc hclT hclP hcl.par hcl.lin]
[baf.dep baf.ebk baf.pe baf.ppe baf.ne baf.nne baf.k0 baf.ke]
[stress.dep Vc Vr Vd Vt Dlim]
[gamma]
[silicon | oxide | nitride | poly | gas]
[/silicon | /oxide | /nitride | /poly | /gas]
[alpha | henry.coeff | theta]
[diff.0 | diff.e | seg.0 | seg.E | trn.0 | trn.E]
[initial | spread | mask.edge]
[erf.q erf.delta erf.lbb erf.h nit.thick]

SUPREM IV

Level: Commands

Author: Conor S. Rafferty

Version: 3.3 9/29/88 17:11:33

DESCRIPTION

All parameters relating to oxidation, and a few more besides, are specified here.

Five oxide models are supported in this release: 1) an error-function fit to bird's beak shapes, 2) a parametrized error-function model from the literature 3) a model where oxidant diffuses and the oxide boundaries moves vertically at a rate determined by the local oxidant concentration 4) a compressible viscous flow model, similar to 3 but in addition the new oxide formed at each time step forces the overlying oxide (and nitride) to flow 5) an incompressible viscous flow model.

The error function is by far the fastest. It should be used for uniform oxidation, and is the most reliable for semi-recessed oxidation. It requires fitting data for the lateral spread of the bird's beak.
The parameterized model is by Guillemot et al., IEEE Transactions on Electron Devices, ED-34, May 1987. The bird's beak length and nitride lifting were measured as a function of process conditions.
The diffusion model has no fitting parameters, but is only accurate when the growth is reasonably vertical - less than say 30 degrees interface angle.
The compressible model is more accurate, but requires more computer time. It is still cheaper than the incompressible model, which uses many more grid points. The incompressible model is required whenever accurate stresses are desired.

Most coefficients need to know whether wet or dry oxidation is intended, and some need to know what the substrate orientation is.

orientation

The substrate orientation to which the coefficients specified apply. Required for orientation factor (see below) and thin oxide coefficients.

dry | wet

The type of oxidation to which the specified coefficients apply. Required for everything except for the one-dimensional coefficients and the volume ratio.

[lin.l.0 lin.l.e lin.h.0 lin.h.e l.break l.pdep]

The linear rate coefficients (B/A). A doubly activated Arrhenius model is assumed. l.break is the temperature breakpoint between the lower and higher ranges, in degrees Celsius. lin.l.0 is the prefactor in microns/min, and lin.l.e is the activation energy in eV for the low temperature range. lin.h.0 and lin.h.e are the corresponding high temperature numbers. l.pdep is the exponent of the pressure dependence. The value given is taken to apply to <111> orientation and later adjusted by ori.fac according to the substrate orientation present.

[par.l.0 par.l.e par.h.0 par.h.e p.break p.pdep]

The parabolic rate coefficients (B). Like the linear rate coefficients, but different.

[ori.dep ori.fac]

The numerical coefficient ori.fac is the ratio of B/A on the specified orientation to that on the <111> orientation. The boolean parameter ori.dep (defaults true) specifies whether the local orientation at each point on the surface should be used to calculate B/A. If it is false, the substrate orientation is used at all points.

[thinox.0 thinox.e thinox.l]

Coefficients for the thin oxide model proposed by Massoud. thinox.0 is the prefactor in microns/min, thinox.e is the activation energy in eV, and thinox.l is the characteristic length in microns.

[hcl.pc hclT hclP hcl.par hcl.lin]

The numerical parameter hcl.pc is the percentage of HCl in the gas stream. It defaults to 0. The HCl dependence of the linear and parabolic coefficients is obtained from a look-up table specified in the model file. The rows of the table are indexed by HCl percentage. The row entries can be specifed with the parameter hclP, which is an array of numerical values, surrounded by double quotes and separated by spaces or commas. The columns are indexed by temperature. The column entries can be specifed with the parameter hclT, which is an array of numerical values, surrounded by double quotes and separated by spaces or commas. The dependence of B/A can be specified with the parameter hcl.lin, which is an array of numerical values, surrounded by double quotes and separated by spaces or commas. The number of entries in hcl.lin must be the product of the number of entries in hclP and hclT. The dependence of B can be specified with the parameter hcl.par, which is an array of numerical values, surrounded by double quotes and separated by spaces or commas. The number of entries in hcl.par must be the product of the number of entries in hclP and hclT.

[baf.dep baf.ebk baf.pe baf.ppe baf.ne baf.nne baf.k0 baf.ke]

These parameters relate to the doping dependence of the oxidation rate. The doping dependence is turned on if baf.dep is true. The linear rate coefficient, B/A, is assumed to depend on the Fermi level as follows. (Ho & Plummer, J. Electrochem. Soc., Sep 1979, p 1516).

$$B/A = (B/A)_0\left[1 + K([V]/[V]_0 - 1)\right]$$

$[V]$ is the total concentration of vacancies and K is an activated coefficient with prefactor baf.k0 and activation energy baf.ke. The remaining parameters above serve to compute $[V]$. The total concentration of vacancies $[V]$ is the sum of the concentrations in each charge state. The concentration in the single positive charge state is related to the Fermi level through $[V sup +] / [V sup 0] = (n sub i / n) sqrt f exp (- baf.pe / kT)$. The concentration in the double positive charge state is $[V sup ++] / [V sup 0] = (n sub i / n) f exp (- baf.ppe / kT)$, in the negative state $[V sup -] / [V sup 0] = (n / n sub i) sqrt f exp (- baf.ne / kT)$, and in the double negative state $[V sup =] / [V sup 0] = (n / n sub i) sup 2 f exp (- baf.nne / kT)$. The factor f derives from the change in bandgap with temperature. If the bandgap is written $E sub g = E sub g0 - beta T$, then f is $exp(beta / k)$ and is specified as baf.ebk. Defaults for all the values are given in the model file.

[stress.dep Vc Vr Vd Vt Dlim]

These parameters control the stress dependence of oxidation, which is only calculated under the viscous model. The parameter stress.dep turns on the dependence. The parameter Vc is the

activation volume of viscosity. The parameter Vr is the activation volume of the reaction rate with respect to normal stress. The parameter Vt is the activation volume of the reaction rate with respect to tangential stress. The parameter Vd is the activation volume of oxidant diffusion with respect to pressure. The parameter Dlim is the maximum increase of diffusion permitted under tensile stress.

[gamma]
The parameter gamma is the oxidant surface energy in ergs/cm2, which controls reflow.

[alpha]
The volume expansion ratio between material 1 and material 2. In a nutshell, alpha oxide /silicon is 2.2, and alpha x /y is 1.0 for everything else (until someone tells us otherwise).

[silicon | oxide | nitride | poly | gas]
What material 1 is.

[/silicon | /oxide | /nitride | /poly | /gas]
What material 2 is.

[henry.coeff | theta]
Henry's coefficient is the solubility of oxidant in material 1 at one atmosphere, in 1/cm3. Theta is the number of oxygen atoms incorporated in a cubic centimeter of oxide. **Note:** Don't change these unless you really know what you are doing. Change the Deal-Grove coefficients instead.

[diff.0 | diff.e | seg.0 | seg.E | trn.0 | trn.E]
The diffusion coefficients of oxidant in material 1, and the boundary coefficients ("transport" and "segregation") from material 1 to material 2. See the note above. For the record, diff.0 is the diffusivity prefactor in cm2/s, diff.e is the energy in eV. The transport coefficient represents the gas-phase mass-transfer coefficient in terms of concentrations in the solid at the oxide-gas interface, the chemical surface-reaction rate constant at the oxide-silicon surface, and a regular diffusive transport coefficient at other interfaces. The segregation coefficient is 1 at the oxide-gas interface, infinity at the oxide-silicon interface, and represents a regular segregation coefficient at other interfaces.

[initial]
The thickness of the existing oxide at the start of oxidation, default 2 nm. If the wafer is bare, an oxide layer of this thickness is deposited before oxidation begins.

[spread | mask.edge]
These coefficients are only used in the error-function approximation to a bird's beak shape. Spread is the relative lateral to vertical extension, defaults to 1. It's a fitting parameter to make erfc birds look realistic. Mask.edge is the position of the mask edge in microns, and defaults to negative infinity. Oxide grows to the right of the mask edge.

erf.q erf.delta erf.lbb erf.h nit.thick
This group of parameters all apply to the "erfg" model. See Guillemot et al. for their interpretation.

erf.q erf.delta
The delta and q parameters for the "erfg" model. These probably do not need to be changed, but they're available if you want them.

erf.lbb The erf.lbb parameter is the length of the bird's beak. It can be specified as an expression in Eox (the field oxide thickness (um)), eox (the pad oxide thickness (um)), Tox (the oxidation temperature (Kelvin)), and en (the nitride thickness (um)). The published expression can be found in the models file. Specifying "erf.lbb=Eox" for instance would give a lateral spread equal to the field thickness, similar to the Hee-Gook Lee model with a spread of 1.

erf.h The erf.h parameter is the ratio of the nitride lifting to the field oxide thickness. (It corresponds to the Guillemot "H" parameter except that it is normalized to the field oxide thickness). Again it is specified as an expression of Eox, eox, Tox, en.

nit.thick

The nitride thickness to substitute for the parameter "en".

EXAMPLES

Look at the models file for a huge example.

BUGS

In increasing order of severity:

+ If a required parameter is omitted, like orientation when a linear rate coefficient is being specified, the statement is ignored without warning.

+ The volume expansion data should be on the material statement.

+ Oxidant in materials other than oxide is allowed to diffuse and segregate, but its concentration is then ignored (no oxynitridation, for instance). The diffusion and transport coefficients in oxide to gas and silicon are derived from the Deal-Grove coefficients, so these parameters are ignored if read from input statements.

+ The non-analytic oxide models don't account for the thin oxide correction in dry oxygen.

+ The analytic models use the "thickness" of the oxide to compute the growth rate. In addition, the "erfg" model uses the nitride thickness. These values are NOT inferred from the structure. Instead, the nitride thickness is taken from the user on the oxide card, and the oxide thickness is computing by adding the total oxide grown to the initial oxide thickness specified on the oxide card. If the structure at the start of diffusion has more than the default 20 angstroms of native oxide on it, that thickness must be specified by the user. Beware of this when continuing a diffusion by any means (e.g. after reading in a previous structure). A "diffuse" followed by "diffuse continue=t" will however do the right thing.

PHOSPHORUS

phosphorus – set the coefficients of phosphorus kinetics

SYNOPSIS

phosphorus

(silicon | oxide | oxynit | nitride | gas | poly)
[Dix.0=<n>] [Dix.E=<n>]
[Dim.0=<n>] [Dim.E=<n>]
[Fi = <n>]
[ss.clear] [ss.temp=<n>] [ss.conc=<n>]
[(/silicon | /oxide | /oxynitr | /nitride | /gas | /poly)]
[Seg.0=<n>] [Seg.E=<n>] [Trn.0=<n>] [Trn.E=<n>]

SUPREM IV

Level: Models

Author: Mark E. Law

Version: 3.3 11/21/88

DESCRIPTION

This statement allows the user to specify values for coefficients of phosphorus diffusion and segregation. The diffusion equation for phosphorus is:

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_T \frac{C_V}{C_V^*}\nabla\ln\left[C_T \frac{C_V}{C_V^*}\frac{n}{n_i}\right] + D_I C_T \frac{C_I}{C_I^*}\nabla\ln\left[C_T \frac{C_I}{C_I^*}\frac{n}{n_i}\right]\right]$$

where C_T is the total chemical concentration of phosphorus, C_V and C_I are the vacancy and interstitial concentration, C_V^* and C_I^* are the equilibrium vacancy and interstitial concentration, D_V and D_I are the diffusivities with vacancies and interstitials, and n and n_i refer to the electron concentration and the intrinsic concentration respectively. [1] The diffusivities are given by:

$$D_V = \left[Dvx + Dvm\frac{n}{n_i} + Dvmm\frac{n^2}{n_i^2}\right]$$

$$D_I = \left[Dix + Dim\frac{n}{n_i} + Dimm\frac{n^2}{n_i^2}\right]$$

The terms in these equations are described in greater detail below.

The segregation at material interfaces is computed using the following expression:

$$flux = Trn\left[C_1 - \frac{C_2}{Seg}\right]$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the Seg and Trn terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly

These allow the specification of parameters for that material. Only one of these can be specified per statement. The parameters specified in that statement will apply in the material listed. These parameters specify which material is material 1 for the segregation terms.

Dix.0, Dix.E

These floating point parameters allows the specification of the phosphorus diffusing with neutral interstitials. Dix.0 is the pre-exponential constant and Dix.E is the activation energy. Dix.0 defaults to 3.85 centimeters squared per second in silicon, and Dix.E defaults to 3.66eV in silicon [2,3]. The Dix term is calculated using:

$$Dix = Dix.0 \exp\left(-\frac{Dix.E}{kT}\right)$$

Dim.0, Dim.E

These floating point parameters allows the specification of the phosphorus diffusing with singly negative interstitials. Dim.0 is the pre-exponential constant and Dim.E is the activation energy. Dim.0 defaults to 4.44 centimeters squared per second in silicon, and Dim.E defaults to 4.00eV in silicon [2, 3]. The Dim term is calculated using:

$$Dim = Dim.0 \exp\left(-\frac{Dim.E}{kT}\right)$$

Dimm.0, Dimm.E

These floating point parameters allows the specification of the phosphorus diffusing with doubly negative interstitials. Dimm.0 is the pre-exponential constant and Dimm.E is the activation energy. Dimm.0 defaults to 4.44 centimeters squared per second in silicon, and Dimm.E defaults to 4.00eV in silicon [2, 3]. The Dimm term is calculated using:

$$Dimm = Dimm.0 \exp\left(-\frac{Dimm.E}{kT}\right)$$

Dvx.0, Dvx.E

These floating point parameters allows the specification of the phosphorus diffusing with neutral vacancies. Dvx.0 is the pre-exponential constant and Dvx.E is the activation energy. Dvx.0 defaults to 0.0 centimeters squared per second in silicon, and Dvx.E defaults to 0.0eV in silicon [2, 3]. The Dvx term is calculated using:

$$Dvx = Dvx.0 \exp\left(-\frac{Dvx.E}{kT}\right)$$

Dvm.0, Dvm.E

These floating point parameters allows the specification of the phosphorus diffusing with singly negative vacancies. Dvm.0 is the pre-exponential constant and Dvm.E is the activation energy. Dvm.0 defaults to 0.0 centimeters squared per second in silicon, and Dvm.E defaults to 0.0eV in silicon [2, 3]. The Dvm term is calculated using:

$$Dvm = Dvm.0 \exp\left(-\frac{Dvm.E}{kT}\right)$$

Dvmm.0, Dvmm.E

These floating point parameters allows the specification of the phosphorus diffusing with doubly negative vacancies. Dvmm.0 is the pre-exponential constant and Dvmm.E is the activation energy. Dvmm.0 defaults to 0.0 centimeters squared per second in silicon, and Dvmm.E defaults to 0.0eV in silicon [2, 3]. The Dvmm term is calculated using:

$$Dvmm = Dvmm.0 \exp\left(-\frac{Dvmm.E}{kT}\right)$$

ss.clear This parameter clears the currently stored solid solubility data.

ss.temp, ss.conc

These parameters add a single temperature solid solubility concentration point to those already stored. The default values are [4]:

Solid Solubility Data

850°C $2.7943 \cdot 10^{20}$

900°C $3.1585 \cdot 10^{20}$
1000°C $3.3981 \cdot 10^{20}$
1100°C $3.7943 \cdot 10^{20}$

/silicon, /oxide, /oxynit, /nitride, /gas, /poly
These parameters specify material 2. Only one of the these parameters can be specified at one time.

Seg.0, Seg.E
These parameters allow the computation of the equilibrium segregation concentrations. The formula used to calculate seg from material 1 to material 2 is:

$$seg = Seg.0 \exp\left(-\frac{Seg.E}{kT}\right)$$

Trn.0, Trn.E
These parameters allow the specification of the transport velocity across the interface given. The units are in cm/s. The formula used to calculate trn is

$$trn = Trn.0 \exp\left(-\frac{Trn.E}{kT}\right)$$

EXAMPLES

phosphorus silicon Dix.0=3.85 Dix.E=3.85
This command changes the neutral interstitial diffusivity in silicon.

phosphorus silicon /oxide Seg.0=30.0 Trn.0=1.66e-7
This command will change the segregation parameters at the Silicon - Silicon Dioxide interface. The silicon concentration will be 30.0 times the oxide concentration in equilibrium.

BUGS

As far as the implemented models are physically correct, there are no known bugs.

REFERENCES

1. D. Mathiot and J.C. Pfister, "Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects," *Journal of Applied Physics*, vol. 55 No. 10, p. 3518, May 15, 1984.

2. R.B. Fair, "Concentration Profiles of Diffused Dopants in Silicon," in *Impurity Doping Processes in Silicon*, ed. F. F. Y. Yang, p. 315, North-Holland, New York, 1981.

3. P. Fahey, G. Barbuscia, M. Moslehi, and R.W. Dutton, "Kinetics of Thermal Nitridation Processes in the Study of Dopant Diffusion Mechanisms in Silicon," *Appl. Phys. Lett.*, vol. 46 (8), p. 784, April 1985.

4. D. Nobili, A. Armigliato, M. Finetti, and S. Solmi, "Precipitation as the Phenomenon Responsible for the Electrically Inactive Phosphorus in Silicon," *J. Appl. Phys.*, vol. 53 No. 3, pp. 1481 - 1491, March 1982.

SEE ALSO

arsenic, boron, phosphorus, intersititial, vacancy

TRAP

trap – set coefficients of interstitial traps

SYNOPSIS

trap

(silicon | oxide | poly | oxynitr | nitride | gas | aluminum | photores)
[enable]
[total=<n>]
[frac.0=<n>] [frac.E=<n>]

SUPREM IV

Level: Models

Author: Mark E. Law

Version: 1.1 12/6/88

DESCRIPTION

This statement allows the user to specify values for coefficients of the interstitial traps. The statement allows coefficients to be specified for each of the materials. SUPREM-IV has default values only for silicon. Polysilicon has not been characterized as extensively, and its parameters default to those for silicon.

The trap reaction was described by Griffin [1]. This model explains some of the wide variety of diffusion coefficients extracted from several different experimental conditions. The trap equation is:

$$\frac{\partial C_{ET}}{\partial t} = -K_T\left[C_{ET}C_I - \frac{e^*}{1-e^*}C_I^*(C_T - C_{ET})\right]$$

where C_T is the total trap concentration, C_I and C_I^* are the interstitial and equilibrium interstitial concentration, K_T is the trap reaction coefficient, e^* is the equilibrium trap occupancy ratio. Instead of the reaction, the time derivative is used in the interstitial equation because it has better properties in the numerical calculation.

The equation is derived from the reaction:

$$I + T \leq > IT$$

In equilibrium, the reaction will balance. Therefore :

$$K_T C_I^* C_{ET}^* = K_r(C_T - C_{ET}^*)$$

where K_T and K_r are the forward and reverse rates, respectively. e^* is defined to be C_{ET}^*/C_T, the equilibrium number of empty traps divided by the total traps.

(silicon | oxide | poly | oxynitr | nitride | gas)
These parameters allow the specification of the material for which the parameters apply.

enable This indicates that material specified has a finite number of traps.

total This floating point parameter is used to specify the total number of traps, in cm^{-3}. The default for silicon is $5{\cdot}10^{17}$ cm^{-3}. This value is appropriate for Czochralski silicon material.

frac.0, frac.E
These floating point parameters allow the specification of the equilibrium empty trap ratio, e^*.

$$e^* = frac.0\ exp(-\frac{frac.E}{kT})$$

The default for these parameters is 3.6e5 and 1.598eV for the preexponential and activation energy.

EXAMPLES

trap silicon total=5.0e17 frac.0=0.5 frac.E=0.0 enable
This statement turns on interstitial traps and sets the total to $5{\cdot}10^{17}$ and the fraction to a half.

BUGS

There are no known bugs in this model. However, the model used here is involved in ongoing research, and may by out of data at any time. Further, experimental verification is difficult. Some of the default parameters may not be appropriate for any process line other than Stanford's. The trap concentration will depend on the thermal history of the wafer, starting material, stress and temperature.

REFERENCE

1. P.B. Griffin and J.D. Plummer, "Process Physics Determining 2-D Impurity Profiles in VLSI Devices," *International Electron Devices Meeting*, p. 522, Los Angeles, Dec., 1986.

SEE ALSO

interstitial(3), vacancy(3)

VACANCY

vacancy – set coefficients of vacancy kinetics

SYNOPSIS

vacancy

(silicon | oxide | poly | oxynitr | nitride | gas)
[D.0=<n>] [D.E=<n>]
[Kr.0=<n>] [Kr.E=<n>]
[Cstar.0=<n>] [Cstar.E=<n>]
[neu.0=<n>] [neu.E=<n>]
[neg.0=<n>] [neg.E=<n>] [dneg.0=<n>] [dneg.E=<n>]
[pos.0=<n>] [pos.E=<n>] [dpos.0=<n>] [dpos.E=<n>]
[(/silicon | /oxide | /poly | /oxynitr | /nitride | /gas)]
[time.inj] [growth.inj] [recomb]
[Ksurf.0=<n>] [Ksurf.E=<n>] [Krat.0=<n>] [Krat.E=<n>]
[Kpow.0=<n>] [Kpow.E=<n>]
[vmole=<n>] [theta.0=<n>] [theta.E=<n>]
[Gpow.0=<n>] [Gpow.E=<n>]
[A.0=<n>] [A.E=<n>] [t0.0=<n>] [t0.E=<n>]
[Tpow.0=<n>] [Tpow.E=<n>]
[rec.str=<s>] [inj.str=<s>]
[(antimony | arsenic | boron | phosphorus)]

SUPREM IV

Level: Models

Author: Mark E. Law

Version: 3.3 12/6/88

DESCRIPTION

This statement allows the user to specify values for coefficients of vacancy continuity equation. The statement allows coefficients to be specified for each of the materials. Of course, the meaning of a lattice vacancy in an amorphous region is a philosophical topic. SUPREM-IV has default values only for silicon and the interfaces with silicon. Polysilicon has not been characterized as extensively, and its parameters default to those for silicon.

The vacancies obey a complex diffusion equation that was first described by Mathiot and Pfister [1]. This equation can be written (although with different notation then Mathiot and Pfister):

$$\frac{\partial}{\partial t}\left[C_V + \sum_{imp,c} k_{AV^c} C_A C_{V^c}\right] = \nabla\left[-J_V - \sum_{imp} J_{AV}\right] - R$$

where C_V is the vacancy concentration, the first sum is over all impurities A and charge states c, J_V refers to the vacancy flux, J_{AV} refers to the flux of impurity A diffusing with vacancies, and R is all sources of bulk recombination of vacancies. In the two.dim model (see method(2)) the fluxes from the dopants, J_{AV} are ignored. Only in the full.cpl model are the computed and included in the vacancy equation. This model represents the diffusion of all vacancies, paired or unpaired, and can be derived from assuming thermal equilibrium between the species and that the simple pairing reactions are dominant.

The pair fluxes are the vacancy portions of the flux as described in each of the models for the impurities. (see antimony(3), arsenic(3), boron(3), and phosphorus(3)) The vacancy flux can be written [2, 1]:

$$-J_V = D_V C_V^* \nabla\left(\frac{C_V}{C_V^*}\right)$$

It is important to note that the equilibrium concentration, C_V^* is a function of fermi level [3]. This flux accounts correctly for the effect of an electric field on the charged portion of the defect concentration. The bulk recombination is simple interaction between vacancies and vacancies. This can be expressed:

$$R = K_R\left[C_I C_V - C_I^* C_V^*\right]$$

where K_R is the bulk recombination coefficient, and C_I and C_I^* are the vacancy and vacancy equilibrium concentration, respectively.

The defects obey a flux balance boundary condition, as described by Hu [4]:

$$J_V \cdot \mathbf{n} + K_{surf}\left[C_V - C_V^*\right] = G$$

where $\mathbf{n}$ is the surface normal, K_{surf} is the surface recombination constant, and G is the generation, if any, at the surface. This expression has been used with success on several different surface types.

In all cases for the parameters, the value can be determined by using a simple singly activated model:

$$X = X.0 \exp\left[-\frac{X.E}{KT}\right]$$

(silicon | oxide | poly | oxynitr | nitride | gas)

These parameters allow the specification of the material for which the parameters apply.

D.0, D.E

These floating point parameters are used to specify the diffusion coefficient of the vacancies. The units are in cm^2 / s. The default values are 0.1 cm^2 / s and 2.00eV [5].

Kr.0, Kr.E

These floating point parameters allow the specification of the bulk recombination rate $cm^{-3}s^{-1}$. The bulk recombination value was extracted using experiments from Packan [6], and Guerrero *et al.* [7]. The values that best fit the data are $3.16 \cdot 10^{-6}$ $cm^{-3}s^{-1}$ and 2.44eV.

Cstar.0, Cstar.E

These parameters allow the specification of the total equilibrium concentration of vacancies in intrinsically doped conditions. The default values are $2.0 \cdot 10^{23}$ cm^{-3} and 2.0eV [5].

neu.0, neu.E, neg.0, neg.E, dneg.0, dneg.E, pos.0, pos.E, dpos.0, dpos.E

These values allow the user to specify the relative concentration of vacancies in the various charge states (neutral, negative, double negative, positive, and double positive) under intrinsic doping conditions. In intrinsic conditions, the concentration of doubly negative vacancies is given by:

$$C_V^{=} = \frac{Cstar \cdot dneg}{neu + neg + dneg + pos + dpos}$$

where Cstar is the total equilibrium concentration in intrinsic doping conditions. In extrinsic conditions, the total equilibrium concentration is given by [3]:

$$C_V^* = Cstar\frac{neu + neg\frac{n}{n_i} + dneg\frac{n^2}{{n_i}^2} + dpos\frac{p}{n_i} dpos\frac{p^2}{{n_i}^2}}{neu + neg + dneg + pos + dpos}$$

There is very little data for the vacancy charge states, it is matter of ongoing research. The default values are chosen based on experiments and analysis of Watkins [8]. These parameters take on a different meaning when an impurity is specified. This is described later in this document under the description of the impurity parameters.

(/silicon | /oxide | /poly | /oxynitr | /nitride | /gas)

These parameters allow the specification of the interface for which the boundary condition parameters apply. For example, parameters about the oxide silicon effect on vacancies should be specified by listing silicon /oxide. Only one of these can be specified at a time.

time.inj, growth.inj, recomb

These parameters specify the type of reactions occurring at the specified interface. The time.inj parameter means that a time dependent injection model should be chosen. The growth.inj

parameter ties the injection to the interface growth velocity. The recomb parameter indicates a finite surface recombination velocity.

Ksurf.0, Ksurf.E, Krat.0, Krat.E, Kpow.0, Kpow.E

These parameters allow the specification of the surface recombination velocity. The formula used in computing the actual recombination velocity is:

$$K_s = Ksurf\left[Krat\left(\frac{v_i}{B/A}\right)^{Kpow} + 1\right]$$

where v_i is the interface velocity, and B/A is the Deal Grove oxide growth coefficient. (see oxide(3)). The full model applies only to oxide, since it is the only interface that can grow in SUPREM-IV. This formulation allows an inert interface to have different values than a growing interface. For inert interfaces, (gas, oxynitride, nitride, poly) Ksurf is the only important parameter.

vmole, theta.0, theta.E, Gpow.0, Gpow.E

These parameters allow the specification of generation that is dependent on the growth rate of the interface. The formula is:

$$G = \theta \cdot vmole \cdot v_i \left(\frac{v_i}{B/A}\right)^{Gpow}$$

Theta is the fraction of silicon atoms consumed during growth that are reinjected as vacancies. Once again, this model only makes since for oxide since it is currently the only material which can have a growth velocity.

A.0, A.E, t0.0, t0.E, Tpow.0, Tpow.E

These parameters allow an injection model with a fairly flexible time dependency. The equation for the injection is :

$$G = A\left(t + t0\right)^{Tpow}$$

where t is the time in the diffusion in seconds. This can be used to represent the injection of vacancies from an oxynitride layer.

rec.str, inj.str

These two string parameters are useful for experimenting with new models for injection or recombination at interfaces. Three macros are defined for use, t the time in seconds and x and y the coordinates. If these are specified, they are used in place of any other model. For example,

vacancy silicon /oxide inj.str = (10.0e4 * exp(t / 10.0))

describes an injection at the silicon oxide interface that exponentially decays in time.

(antimony | arsenic | boron | phosphorus)

These parameters allow the user to specify the a dopant. If a dopant is specified, the user can then specify the equilibrium reaction coefficient using the neutral, negative, double, and positive parameters. For example, to specify the value of K sub {BV sup 0}, the reaction coefficient for boron and neutral vacancies:

vacancy silicon boron neu.0=1.0e-18 neu.E=0.0

All of these coefficients are next to impossible to determine experimentally, and the default for them is zero. This basically states that the number of impurity vacancy pairs is negligible compared to the number of free vacancies.

EXAMPLES

vacan silicon Di.0=5.0e-7 D.E=0.0 Cstar.0=1.0e13 Cstar.E=0.0

This statement specifies the silicon diffusion and equilibrium values for vacancies.

vacan silicon /oxide growth vmole=5.0e22 theta.0=0.01 theta.E=0.0

This parameter specifies the oxide - silicon interface injection is to be computed using the oxide

growth velocity and with 1% of consumed silicon injected as vacancies.

vacan silicon /nitride Ksurf.0=3.5e-3 Ksurf.E=0.0 Krat.0=0.0

This specifies that the surface recombination velocity at the nitride silicon interface is 3.5e-3 cm/s.

vacancy silicon neu.0=1.0 neg.0=1.0 pos.0=0.0 dneg.0=0.0 dpos.0=0.0 vacancy silicon neu.E=0.0 neg.E=0.0 pos.E=0.0 dneg.E=0.0 dpos.E=0.0

This specifies that there are equal numbers of negative and neutral charged vacancies in intrinsic doping.

BUGS

There are no known bugs in this routine. However, the models used here are involved in ongoing research, and may by out of data at any time. Further, experimental verification is difficult. Some of the default parameters may not be appropriate for any process line other than Stanford's. Many of the parameters have unknown dependencies on stress, temperature, starting silicon material, stacking fault density, etc. High concentration phosphorus diffusion is promising with these models, but the coefficients have not been extracted to allow good fits to experiment.

REFERENCE

1. D. Mathiot and J.C. Pfister, "Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects," *Journal of Applied Physics*, vol. 55 No. 10, p. 3518, May 15, 1984.

2. M.E. Law, "Two Dimensional Numerical Simulation of Dopant Diffusion in Silicon," PhD dissertation, Department of Electrical Engineering, Stanford University, Jan. 1988.

3. W. Shockley and J.T. Last, "Statistics of the Charge Distribution for a Localized Flaw in a Semiconductor," *Phys. Rev.*, vol. 107 No. 2, p. 392, July 15, 1957.

4. S.M. Hu, "On Interstitial and Vacancy Concentrations in Presence of Injection," *J. Appl. Phys.*, vol. 57, p. 1069, 1985.

5. T.Y. Tan and U. Gosele, "Point Defects, Diffusion Processes, and Swirl Defect Formation in Silicon," *Appl. Phys.*, vol. A37 No. 1, p. 1, 1985.

6. P. Packan, *Private Communication.*

7. E. Guerrero, W. Jungling, H. Potzl, U. Gosele, L. Mader, M. Grasserbauer, and G. Stingeder, "Determination of the Retarded Diffusion of Antimony by SIMS measurements and numerical simulations," *Journal Electrochemical Society* , p. 2182, Oct., 1986.

8. G.D. Watkins, "EPR Studies of the Lattice Vacancy and Low Temperature Damage Processes in Silicon," in *Lattice Defects in Semiconductors 1974*, ed. F.A. Huntley, Inst. Phys. Conf. Ser. 23, London, 1975.

SEE ALSO

antimony(3), arsenic(3), boron(3), phosphorus(3), trap(3), vacancy(3)

SUPREM IV

Level: Shell

Version: 1.2 12/7/88 11:21:56

DESCRIPTION

The examples are provided to give the users templates of simulation input decks to edit and modify. These examples are intended to provide more detail about the program and allow a user to become sufficient.

The best way to read the examples is to go through the first one to get a feel for the program and the output. Each example builds on the preceeding ones.

SUPREM IV

Level: Examples

Version: 1.2 11/22/88 09:17:50

DESCRIPTION

This example performs a simple anneal of a boron implant. The final structure is saved, and then various post processing is performed. The input file for the simulation is in the "examples/exam1" directory, in the file boron.in

```
#some set stuff
set echo
option quiet

#the x dimension definition
line x loc = 0.0 tag = left
line x loc = 0.25 spacing = 0.25 tag = right

#the vertical definition
line y loc = 0     spacing = 0.02 tag = top
line y loc = 0.50  spacing = 0.02
line y loc = 2.0   spacing = 0.25 tag=bottom

#the silicon wafer
region silicon xlo = left xhi = right ylo = top yhi = bottom

#set up the exposed surfaces
bound exposed  xlo = left xhi = right ylo = top  yhi = top

#calculate the mesh
init boron conc=1.0e14

#the pad oxide
deposit oxide thick=0.075

#the uniform boron implant
implant boron dose=3e14 energy=70 pearson

#plot the initial profile
sel z=log10(boron)
plot.1d x.v=0.1 x.max=2.0 y.min=14.0 y.max=20.0

#the diffusion card
diffuse time=30 temp=1100

#save the data
structure out=boron.str

#plot the final profile
sel z=log10(bor)
plot.1d x.v=0.1 x.max=2.0 cle=f axi=f
```

The first lines in the input deck set some basic SUPREM-IV options.

```
#some set stuff
set echo
option quiet
```

The '#' character is the comment character for SUPREM-IV. The entire line after the '#' is ignored. The echo flag is turned on. This instructs SUPREM-IV to echo input lines after they are typed. This is useful when the output from the simulator is being redirected into a file, so the commands are in the output listing. The second command instructs the simulator to be quiet about what it is doing. The default option is verbose, but this prints far more information than is needed by the novice user.

The next section begins the definition of the mesh to be used for the simulation. The section

```
#the x dimension definition
line x loc = 0.0 tag = left
line x loc = 0.25 spacing = 0.25 tag = right
```

describes the locations of the x lines in the mesh. SUPREM-IV defines x to be the direction across the top of the wafer, and y to be the vertical dimension into the wafer. This simulation is to be done on a one dimensional problem, therefore only two lines are used in the x direction. The first command line instructs SUPREM-IV to locate a mesh line at x equals 0.0μm. This line is 'tagged' to by the name of "left". The tag name is used later in defining regions and boundaries. The second line is tagged "right" and placed at 0.25μm.

The line statement fixes line locations in the mesh as well as the average spacing between lines. In this case, only a pair of mesh lines horizontally is desired, and the spacing is set equal to the distance between the lines. If more mesh lines had been desired, the spacing could have been smaller.

The next section of input describes the location and spacings of the vertical mesh lines.

```
#the vertical definition
line y loc = 0      spacing = 0.02 tag = top
line y loc = 0.50   spacing = 0.02
line y loc = 2.0    spacing = 0.25 tag=bottom
```

The first statement places a mesh line at the top of wafer, y coordinate 0.0μm and tags the line as "top". The spacing is set to 0.02μm. A line is place at 0.5μm which is the expected starting depth of the implanted profile. The spacing is set to 0.02μm. Since the spacing between the first two mesh lines is the same, 0.02μm, there will be mesh lines placed 0.02μm apart between 0.0 and 0.50μm. Finally a line is placed at 2.0μm, which is greater than the depth of the profile after the anneal. The spacing here is set to 0.25μm. In the case where spacings are different in neighboring lines, the spacing is graded between them. The distance between mesh lines near 0.5μm will be 0.02μm and the distance will get progressively closer to 0.25μm as the mesh gets closer to 2.0μm.

The next two sections describe the device starting material and the surfaces which are exposed to gas.

```
#the silicon wafer
region silicon xlo = left xhi = right ylo = top yhi = bottom

#set up the exposed surfaces
bound exposed xlo = left xhi = right ylo = top yhi = top
```

The region statement is used to define the starting materials. In this case the wafer is silicon with no initial masking layers. The silicon area is defined to extend between the lines tagged left and right in the horizontal direction, and top and bottom in the vertical direction. Looking back at the mesh line definition section,

this corresponds to the entire area that had been defined. The initial simulation area is completely silicon.

The bound statement allows the definition of the the front and backsides of the wafer. Any gases on the diffusion statement, depositions, and etches are applied to the surface marked exposed. It is important to define the top of the wafer for SUPREM-IV to correctly simulate these actions. Most mistakes are made by ignoring to define the exposed surface.

The next line informs SUPREM-IV that the mesh has been defined and should be computed.

```
#calculate the mesh
init boron conc=1.0e14
```

This statement computes the locations of lines given the spacings, triangulates the rectangular mesh, and computes geometry information. The initial doping is boron with a concentration of 10^{14} cm^{-3}.

The next line adds a pad oxide to the wafer.

```
#the pad oxide
deposit oxide thick=0.075
```

The deposited oxide is specified to 0.075µm thick. This represents the pad oxide that is grown before the uniform boron implant. Rather than simulate the growth, it is simpler to add a deposited oxide the correct thickness of the pad.

The next statement performs the implant of the boron.

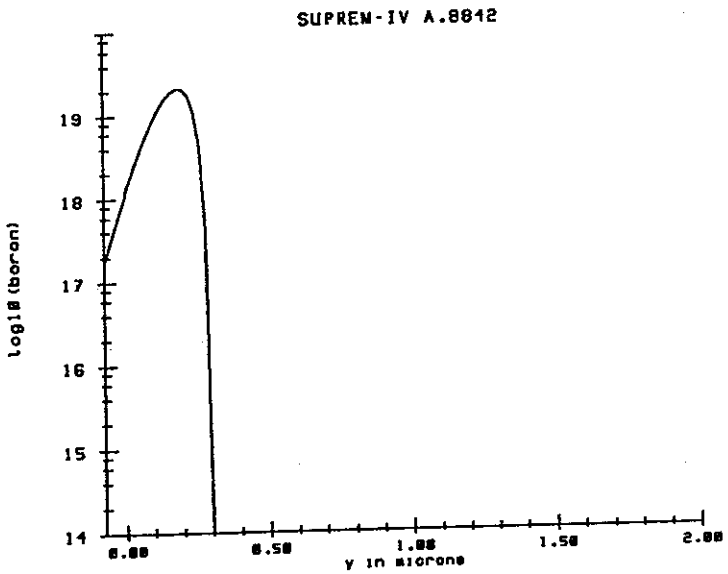
```
#the uniform boron implant
implant boron dose=3e14 energy=70 pearson
```

The implant is modeled with a Pearson-IV distribution. The energy and dose are $3* 10^{14}$ cm^{-2} and 70 Kev respectively. This produces an abrupt boron profile with a junction near 0.3µm.

The next cards choose a plot variable and display it.

```
#plot the initial profile
sel z=log10(boron)
plot.1d x.v=0.1 x.max=2.0 y.min=14.0 y.max=20.0
```

The plot will look like:



The select statement selects the plot variable to be the log base ten of the boron concentration. The plot.1d statement specifies that a cross section should be plotted at a constant x concentration of 0.1μm. The plot should extend between the minimum value of y through 2.0μm. Dimensions on the plot card always refer to the plot axis. In this case, the x axis is the depth dimension, and the y axis is the log of the boron concentration. The minimum and maximum on the y axis are set to 10^{14} and 10^{20}. The y axis values have to be specified in the units of the selected variable, log10 of concentration.

The diffusion card contains the directive to simulate the 30 minute 1100°C drive in and anneal.

```
#the diffusion card
diffuse time=30 temp=1100
```

The default ambient is inert, so no oxide will be grown. The diffuse command will produce the following output:

```
estimated first time step 4.063116e-03
Solving          0 +        0.1 =        0.1,     100%, np 86
Solving        0.1 +     3.1349 =     3.2349,   3134.9%, np 86
Solving     3.2349 +    5.98708 =    9.22198,  190.982%, np 86
Solving    9.22198 +    10.2258 =    19.4478,  170.798%, np 86
Solving    19.4478 +    15.5932 =     35.041,  152.489%, np 86
Solving     35.041 +    22.3125 =    57.3534,  143.091%, np 86
Solving    57.3534 +     30.686 =    88.0395,  137.529%, np 86
Solving    88.0395 +    41.2091 =    129.249,  134.293%, np 86
Solving    129.249 +    54.7286 =    183.977,  132.807%, np 86
Solving    183.977 +    73.7026 =     257.68,  134.669%, np 86
Solving     257.68 +    101.156 =    358.836,  137.249%, np 86
Solving    358.836 +    140.822 =    499.657,  139.212%, np 86
Solving    499.657 +    198.114 =    697.772,  140.684%, np 86
Solving    697.772 +     280.69 =    978.461,  141.681%, np 86
Solving    978.461 +    404.965 =    1383.43,  144.275%, np 86
Solving    1383.43 +    416.574 =       1800,  102.867%, np 86
```

The first line prints the estimate of the initial time step size. The message is informative only. Each of the following lines indicates a time step of the diffusion. The format is the current time + the size of the time step will equal the time after the step. The user can, therfore, determine exactly how far the simulation has proceeded. The next number is the size of the next time step, expressed as a percentage of the current time step. It can be seen for this problem that the time step increases regularly. The final bit of information is the number of points in the mesh, 86 in this example.

The next step is to save the data for further examination.

```
#save the data
structure out=boron.str
```

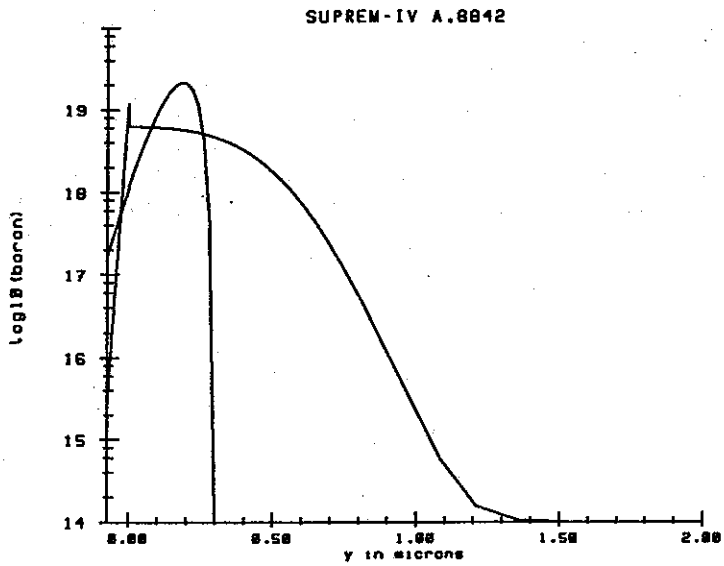
This saves the data in the file boron.str. This file can be read in using the init command or the structure command.

The following script will plot the final boron concentration.

```
#plot the final profile
sel z=log10(bor)
plot.1d x.v=0.1 cle=f axi=f
```

The first statement picks the log base ten of the boron concentration to plot. The plot will be at a constant x value of 0.1μm and will be placed on the earlier plot. The "cle=f axi=f" instructs SUPREM-IV to not clear

and not compute new axes.



The profile shows a deeper junction as expected, as well as segregation into the oxide. The spike at the surface indicates that the surface oxide concentration is about 20 times larger than the silicon surface concentration.

SUPREM IV

Level: Examples

Version: 1.2 11/22/88 09:17:54

DESCRIPTION

This example performs a simple anneal of a boron implant under a dry oxygen ambient. The interstitials are injected by the growing oxide and will enhance the boron diffusivity. This example will produce a deeper boron junction than example 1 because of the interstitial injection. The final structure is saved, and then various post processing is performed. The input file for the simulation is in the "examples/exam2" directory, in the file oed.in.

```
#some set stuff
set echo
option quiet

#the x dimension definition
line x loc = 0.0 tag = left
line x loc = 0.25 spacing = 0.25 tag = right

#the vertical definition
line y loc = 0     spacing = 0.02 tag = top
line y loc = 1.50  spacing = 0.05
line y loc = 5.0   spacing = 0.5
line y loc = 400.0                tag=bottom

#the silicon wafer
region silicon xlo = left xhi = right ylo = top yhi = bottom

#set up the exposed surfaces
bound exposed  xlo = left xhi = right ylo = top  yhi = top

#calculate the mesh
init boron conc=1.0e14

#the pad oxide
deposit oxide thick=0.075

#the uniform boron implant
implant boron dose=3e14 energy=70 pearson

#plot the initial profile
sel z=log10(boron)
plot.1d x.v=0.1 x.ma=2.0 y.mi=14.0 y.max=20.0

#the diffusion card
method two.d init=1.0e-3
diffuse time=30 temp=1100 dry

#save the data
structure out=oed.str

#plot the final profile
sel z=log10(bor)
plot.1d x.v=0.1 cle=f axi=f
```

This example is very similar to example 1. It starts by setting command line echoing and by instructing SUPREM-IV to be relatively quiet about the progress of computation. The x line specification is identical to the first example.

The y line section is considerably different. In this example, we wish to calculate the oxidation enhanced diffusion of a boron layer under a growing oxide. Therefore, the vertical specification needs to be different.

```
#the vertical definition
line y loc = 0      spacing = 0.02 tag = top
line y loc = 1.50   spacing = 0.05
line y loc = 5.0    spacing = 0.5
line y loc = 400.0                 tag=bottom
```

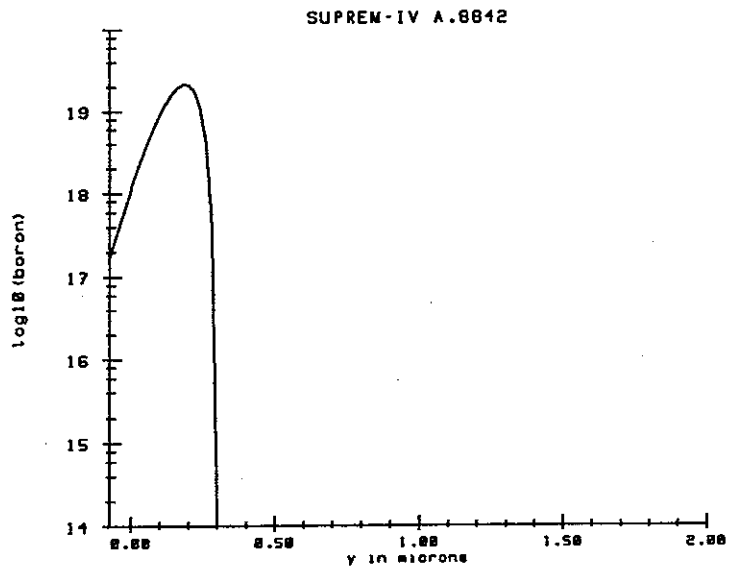
This is similar to the first example. However, we expect the junction to be deeper, therefore the tight grid spacing is maintained out to 1.5μm. The spacing is increased out to a depth of 5μm. The backside is placed at the full thickness of the wafer since the defects can travel great distances into the silicon substrate.

The next series of lines define the region, boundary, and initialize the wafer. A starting oxide is deposited and the boron implant is performed. These steps are all identical to example 1.

The next cards choose a plot variable and display it.

```
#plot the initial profile
sel z=log10(boron)
plot.1d x.v=0.1 x.max=2.0 y.min=14.0 y.max=20.0
```

The plot will look like:



The diffusion card contains the directive to simulate the 30 minute 1100°C drive in and anneal.

```
#the diffusion card
method two.d init=1.0e-3
diffuse time=30 temp=1100 dry
```

The ambient is dry oxygen. The diffuse command will produce the following output:

```
estimated first time step 4.063116e-03
Solving          0 +        0.1 =        0.1,      100%, np 86
Solving        0.1 +     3.1349 =     3.2349,    3134.9%, np 86
Solving     3.2349 +     5.9870 =     9.2219,    190.98%, np 86
Solving     9.2219 +     10.226 =     19.448,    170.80%, np 86
Solving     19.448 +     15.593 =     35.041,    152.49%, np 86
Solving     35.041 +     22.313 =     57.353,    143.09%, np 86
Solving     57.353 +     30.686 =     88.040,    137.53%, np 86
Solving     88.040 +     41.209 =     129.25,    134.29%, np 86
Solving     129.25 +     54.729 =     183.98,    132.81%, np 86
Solving     183.98 +     73.703 =     257.68,    134.67%, np 86
Solving     257.68 +     101.16 =     358.84,    137.25%, np 86
Solving     358.84 +     140.82 =     499.66,    139.21%, np 86
Solving     499.66 +     198.11 =     697.77,    140.68%, np 86
Solving     697.77 +     280.69 =     978.46,    141.68%, np 86
Solving     978.46 +     404.97 =     1383.4,    144.28%, np 86
Solving     1383.4 +     416.57 =       1800,    102.87%, np 86
```

The first thing to notice, in comparison to example 1, is that more time steps are needed. This is for two reasons. First, the defects move faster than the boron and they limit the size of the time step initially. At longer times, the defects begin to approach steady state, and the boron diffusion limits the time step. Since the boron diffusivity is enhanced by the injection of interstitials, the time steps have to be smaller than in example 1 to resolve the faster changes in the boron.

The next step is to save the data for further examination.

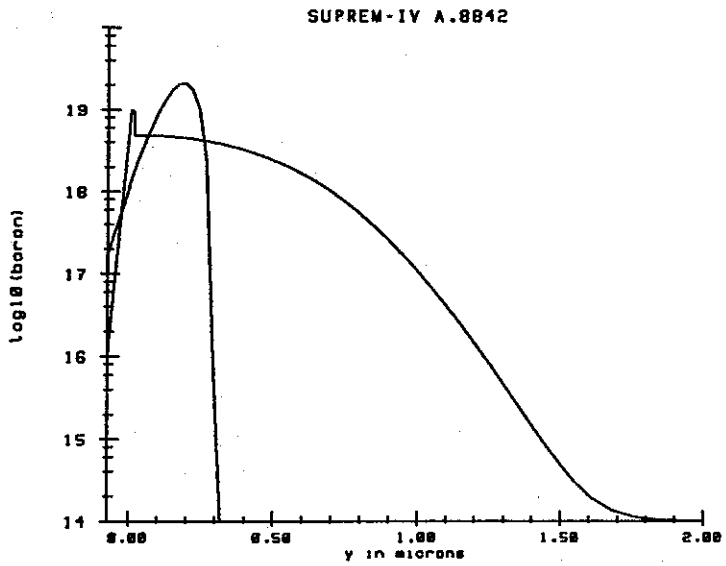
```
#save the data
structure out=boron.str
```

This saves the data in the file boron.str. This file can be read in using the init command or the structure command.

The following script will plot the final boron concentration.

```
#plot the final profile
sel z=log10(bor)
plot.1d x.v=0.1 cle=f axi=f
```

The first statement picks the log base ten of the boron concentration to plot. The plot will be at a constant x value of 0.1μm and will be placed on the earlier plot. The "cle=f axi=f" instructs SUPREM-IV to not clear and not compute new axes.



The profile shows a deeper junction as expected, as well as segregation into the oxide. The spike at the surface indicates that the surface oxide concentration is about 20 times larger than the silicon surface concentration.

SUPREM IV

Level: Examples

Version: 1.1 11/21/88 13:14:59

DESCRIPTION

This example uses the movie command to look at the time dependence of the interstitial concentration resulting from oxide growth. This simulation will be of a silicon membrane 28 μm thick. This will allow the simulation of the effective diffusivity. The input file for the simulation is in the "examples/exam3" directory, in the file oed.in.

```
#some set stuff
option quiet

#the x dimension definition
line x loc = 0.0 tag = left
line x loc = 0.25 spacing = 0.25 tag = right

#the vertical definition
line y loc = 0      spacing = 0.02 tag = top
line y loc = 1.50   spacing = 0.05
line y loc = 5.0    spacing = 0.5
line y loc = 28.0   tag=bottom

#the silicon wafer
region silicon xlo = left xhi = right ylo = top yhi = bottom

#set up the exposed surfaces
bound exposed  xlo = left xhi = right ylo = top  yhi = top
bound backsid  xlo = left xhi = right ylo = bottom  yhi = bottom

#calculate the mesh
init

#the pad oxide
deposit oxide thick=0.02

method init=1.0e-3 two.d

#set up some integration variables
%define otim 0.0
%define oldfci 1.0
%define oldbci 1.0
%define totfci 0.0
%define totbci 0.0

#the diffusion card
%diffuse time=240 temp=1100 dry movie="
    sel z=(inter/ci.star);
    %define newfci zfn(0.0,sil@oxi(0.0)+1.0e-6)
    %define newbci zfn(0.0,28.0)
    %define totfci ${totfci} + 0.5 * (${time}-${otim})*(${newfci}+${oldf
    %define totbci ${totbci} + 0.5 * (${time}-${otim})*(${newbci}+${oldb
    %define den ${time}+1.0e-6
    printf Data ${time}/60.0 (${totfci}/${den}) (${totbci}/${den})
```

```
%define otim ${time}
%define oldfci ${newfci}
%define oldbci ${newbci}"

#save the data
structure out=oed.str
```

This example is very similar to example 2. It starts by instructing SUPREM-IV to be relatively quiet about the progress of computation. The x line specification is identical to the last example.

The y line section is somewhat different. In this example, we wish to calculate the oxidation enhanced diffusion resulting from a growing oxide. Therefore, the vertical specification needs to be different.

```
#the vertical definition
line y loc = 0      spacing = 0.02 tag = top
line y loc = 1.50   spacing = 0.05
line y loc = 5.0    spacing = 0.5
line y loc = 28.0   tag=bottom
```

This is similar to the last example. The backside is placed at the membrane thickness so that we can compute the interstitials at this interface.

The next series of lines define the region, boundary, and initialize the wafer. The are only two differences between this example and the previous one. First is the specification of a backside interface. This will allow the specification of boundary conditions for the defects. Second, no boron implant is performed. This will speed the simulation. A starting oxide is deposited next. The method command specifies the defects should be solved and sets a small initial time step.

The next set of lines initialize some variables that will be used in the diffuse statement.

```
#set up some integration variables
%define otim 0.0
%define oldfci 1.0
%define oldbci 1.0
%define totfci 0.0
%define totbci 0.0
```

In this diffusion, the time average value of the interstitial concentration will be calculated and printed. The variable otim is the old time, oldcfi is the old scaled interstitial concentration at the front, oldbci is the old interstitial concentration at the back, and totfci and totbci are the time averaged values of the scaled interstitial concentration at the front and back respectively.

The diffusion card contains the directive to simulate the 30 minute 1100°C dry oxygen drive in and anneal.

```
#the diffusion card
%diffuse time=30 temp=1100 dry movie="
```

In this example, the movie parameter is used to compute the time average of the interstitial concentration at the front and backside. The entire quoted string for the movie parameter is treated as SUPREM-IV input at the end of every time step. This allows a time history to be developed.

```
sel z=(inter/ci.star);
%define newfci zfn(0.0,sil@oxi(0.0)+1.0e-6)
%define newbci zfn(0.0,28.0)
%define totfci ${totfci} + 0.5 * (${time}-${otim})*(${newfci}+${oldfci})
%define totbci ${totbci} + 0.5 * (${time}-${otim})*(${newbci}+${oldbci})
```

```
%define den ${time}+1.0e-6
printf Data ${time}/60.0 (${totfci}/${den}) (${totbci}/${den})
%define otim ${time}
%define oldfci ${newfci}
%define oldbci ${newbci}"
```

The first command in the movie line is to select the scaled interstitial concentration, C_I/C_I^* as the plot variable. The next line defines the variable newfci to be equal to the value of the z value interpolated at $x = 0mu$ m and $y = 1$ angstrom below the silicon / silicon dioxide interface. This effectively is the surface value of the scaled interstitial concentration. The next line sets newbci to the back side interstitial concentration. The integrated time value at the front and back sides are computed and assigned to the variables totfci and totbci. This time integration uses the trapazoidal rule to compute the values. The time is adjusted by one microsecond in the computation of the average to avoid divide by zero errors. The printf command prints the current time in the diffusion and the front and back side time averaged values. Finally, the variables representing the previous time step values are updated.

The diffuse card should produce output of the form:

```
Data 0 0 0
estimated first time step 1.000000e+37
Solving            0 +        0.001 =        0.001,       100%, np 146
Data 1.66667e-05 1.01264 0.999001
Solving        0.001 +    0.0179525 =    0.0189525,  1795.25%, np 146
Data 0.000315875 1.10615 0.999947
Solving    0.0189525 +    0.0682282 =    0.0871806,  380.049%, np 146
Data 0.00145301 1.24807 0.999989
Solving    0.0871806 +     0.290535 =     0.377716,  425.829%, np 146
Data 0.00629527 1.46482 0.999997
Solving     0.377716 +     0.913082 =       1.2908,  314.276%, np 146
Data 0.0215133 1.72242 1
Solving       1.2908 +      2.49836 =      3.78915,  273.618%, np 146
Data 0.0631525 1.96681 1
Solving      3.78915 +      6.32589 =       10.115,  253.202%, np 146
Data 0.168583 2.16782 1
Solving       10.115 +       14.295 =        24.41,  225.975%, np 146
Data 0.406833 2.31544 1
Solving        24.41 +      34.0394 =      58.4494,  238.122%, np 146
Data 0.974157 2.42726 1
Solving      58.4494 +      81.1766 =       139.626,  238.478%, np 146
Data 2.3271 2.50765 1
Solving       139.626 +      199.404 =       339.03,  245.642%, np 146
Data 5.6505 2.56128 1
Solving       339.03 +      490.628 =      829.658,  246.048%, np 146
Data 13.8276 2.58905 1
Solving      829.658 +       1138.9 =      1968.56,  232.131%, np 146
Data 32.8093 2.58753 1
Solving      1968.56 +      2271.07 =      4239.62,  199.409%, np 146
Data 35.1322 2.58302 1
Solving      2107.93 +      254.309 =      2362.24,  182.473%, np 146
Data 39.3707 2.57032 1
Solving      2362.24 +      1055.62 =      3417.86,  415.093%, np 146
Data 56.9643 2.53371 1
Solving      3417.86 +      2687.85 =       6105.7,  254.623%, np 146
Data 79.6508 2.50023 1
```

```
Solving      4779.05 +      2649.42 =      7428.47,  194.639%, np 144
Data 123.808 2.44842 1
Solving      7428.47 +      3234.85 =      10663.3,  122.096%, np 144
Data 135.691 2.43338 1
Solving      8141.47 +      1223.57 =      9365.03,   171.61%, np 142
Data 156.084 2.40536 1
Solving      9365.03 +      3879.52 =      13244.6,  317.066%, np 142
Data 201.923 2.35689 1
Solving      12115.4 +      2284.63 =        14400,  83.0674%, np 142
Data 240 2.32365 1
```

The lines starting Data show the time in minutes and the front and back side time averaged interstitial concentration. The interstitial concentration ramps up and then levels off. As the simulation continues, the interstitial concentration falls off due to the decreased oxidation rate.

SUPREM IV

Level: Examples

Version: 1.1 11/21/88 13:15:04

DESCRIPTION

This example performs a simple anneal of a boron implant under a dry oxygen ambient. The interstitials are injected by the growing oxide and will enhance the boron diffusivity. This example will use a two dimensional structure to investigate the lateral extent of the oxidation enhanced diffusion behavior. In addition, the two dimensional growth of the oxide will be simulated. The final structure is saved, and then various post processing is performed. The input file for the simulation is in the "examples/exam4" directory, in the file oed.in.

```
#some set stuff
set echo
option quiet

#the x dimension definition
line x loc = -2.0 tag = left
line x loc = 0.0 spacing=0.05
line x loc = 5.0 tag = right

#the vertical definition
line y loc = 0     spacing = 0.02 tag = top
line y loc = 1.50  spacing = 0.05
line y loc = 5.0   spacing = 0.5
line y loc = 400.0                tag=bottom

#the silicon wafer
region silicon xlo = left xhi = right ylo = top yhi = bottom

#set up the exposed surfaces
bound exposed  xlo = left xhi = right ylo = top  yhi = top

#calculate the mesh
init boron conc=1.0e14

#the pad oxide
deposit oxide thick=0.02

#the uniform boron implant
implant boron dose=3e14 energy=70 pearson

#deposit the nitride mask
deposit nitride thick=0.05
etch nitride right p1.x=0.0 p2.x=0.0

#the diffusion card
method init=1.0e-3 two.d
diffuse time=30 temp=1100 dry

#save the data
structure out=oed.str
```

This example is very similar to example 2. It starts by setting command line echoing and by instructing SUPREM-IV to be relatively quiet about the progress of computation. The x line specification is different for the two dimensional device. For this example, an x line is placed at -2μm and another at 5μm for the left and right of the device. The area of most interest is the middle area near where the mask edge will be placed. Consequently, a tighter spacing is specified at the mask edge.

The next series of lines define the y lines, region, boundary, and initialize the wafer. A starting oxide is deposited and the boron implant is performed. These steps are all similar to example 2.

Following the boron implant, a mask is deposited and etched.

```
#deposit the nitride mask
deposit nitride thick=0.05
etch nitride right p1.x=0.0 p2.x=0.0
```

The mask material is specified to be 0.05μm of nitride. This will mask the oxide growth. The nitride is etched off to the right of a vertical line at x equal to 0μm. The oxide will grow on the right and inject interstitials there.

The diffusion card contains the directive to simulate the 30 minute 1100°C drive in and anneal.

```
#the diffusion card
method two.d init=1.0e-3
diffuse time=30 temp=1100 dry
```

The ambient is dry oxygen. This diffusion step will produce output similar to example 2.

The next step is to save the data for further examination.

```
#save the data
structure out=oed.str
```

This saves the data in the file oed.str. This file can be read in using the init command or the structure command.

The following commands are not in an input deck, instead type them in to the simulator using the interactive facilities to plot and check data. The first command will be to read in the stored structure file. Type:

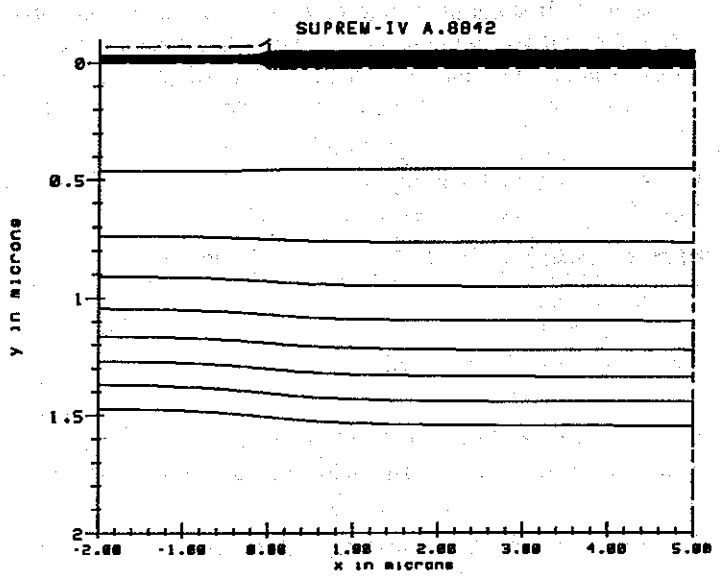
```
init inf=oed.str
```

This will read in the file and initialize the data structures of the program.

The following lines will plot the final boron concentration, in two dimensions

```
#plot the final profile
sel z=log10(bor)
plot.2d bound fill y.max=2.0
foreach v (15.0 to 19.0 step 0.5)
   contour val=v
end
```

The first statement picks the log base ten of the boron concentration to plot. The next line instructs SUPREM-IV to plot the two dimensional outline of the device. The bound parameter asks for the material boundaries to be plotted. The area plotted will extend only down to y equal to 2μm. The contours are plotted as part of loop. The foreach command repeats 9 times and sets the variable v to the value specified in the loop.

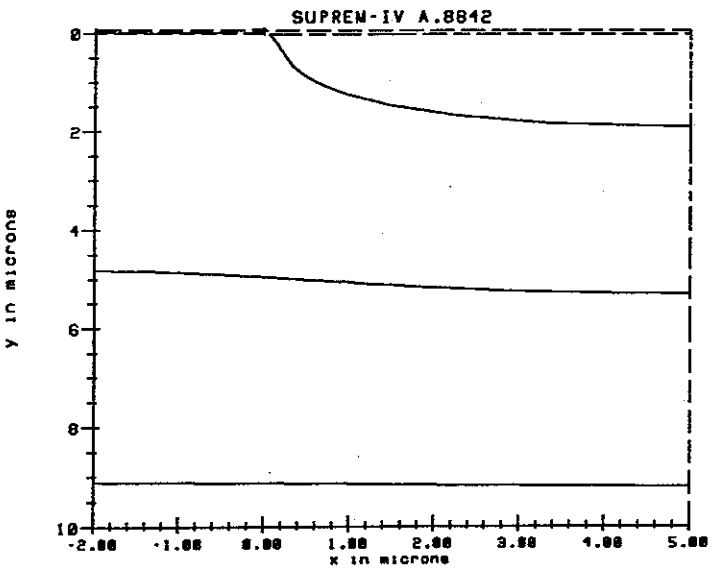


It can be seen that the lateral extent of the oed is well over two microns. There is little lateral difference in the shape of the profile.

The interstitial concentration can also be plotted. This will verify that there is little lateral decay in the defect profile.

```
#plot the final profile
sel z=(inter/ci.star)
plot.2d bound fill y.max=10.0
foreach v (1.25 to 3.0 step 0.25)
   contour val=v
end
```

The plot looks like:

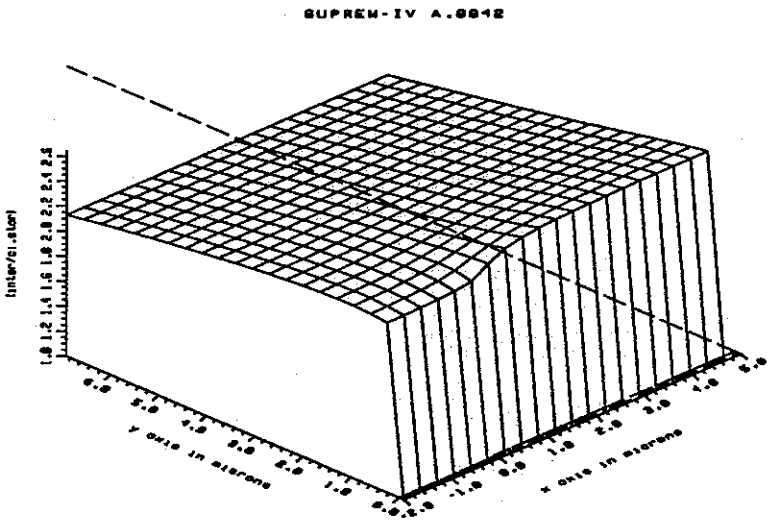


There is more shape to the defects than the boron. However, the profile is fairly flat across the top of the wafer.

A three dimensional bird's eye view plot can be produced with the following command sequence:

```
sel z=(inter/ci.star)
plot.3d y.max=7.0 azim=225
```

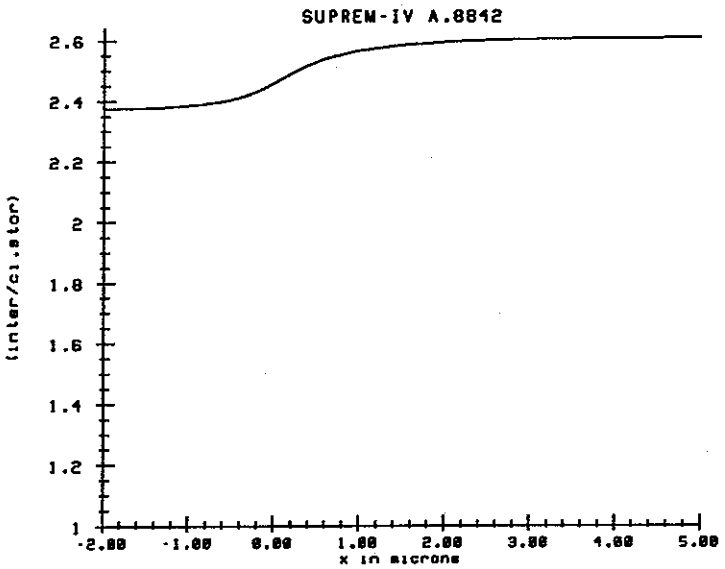
The azimuth parameter rotates the plot around through 225° so that the surface is closest to the viewpoint. The plot is:



The interstitial concentration can also be plotted in cross section.

```
plot.1d y.v=0.5
```

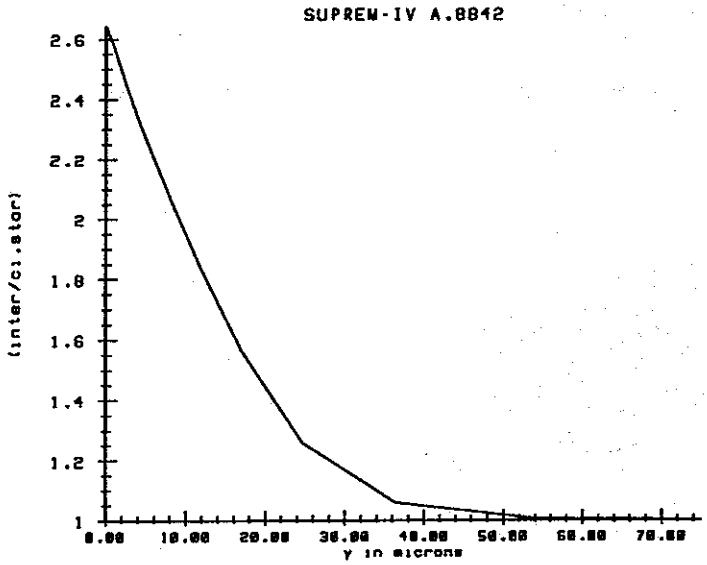
This command will plot the interstitial concentration 0.5μm below the surface. The plot:



The value can also be plotted vertically:

```
plot.1d x.v=5.0 x.ma=75.0
```

This shows the penetration into the bulk of the interstitials. The penetration is limited by both the bulk recombination of the interstitials with vacancies and the trapping mechanism.



SUPREM IV

Level: Examples

Version: 1.1 11/21/88 13:15:06

DESCRIPTION

This example simulates the anneal of a lightly doped drain cross section. This example will produce a cross section appropriate to pass to Pisces for device simulation.

This example is in the directory "example/exam5", and the file whole.s4.

```
set echo
cpu log
#
phos poly /gas Trn.0=0.0
bor poly /gas Trn.0=0.0
phos oxide /gas Trn.0=0.0
bor oxide /gas Trn.0=0.0
#
line x loc=0.0 tag=lft spacing=0.25
line x loc=0.45        spacing=0.03
line x loc=0.75        spacing=0.03
line x loc=1.4         spacing=0.25
line x loc=1.5 tag=rht spacing=0.25

line y loc=0.0 tag=top spacing=0.01
line y loc=0.1          spacing=0.01
line y loc=0.25         spacing=0.05
line y loc=3.0 tag=bot

region silicon xlo=lft xhi=rht ylo=top yhi=bot

bound exposed xlo=lft xhi=rht ylo=top yhi=top
bound backside xlo=lft xhi=rht ylo=bot yhi=bot

init boron conc=1.0e16

#deposit the gate oxide
deposit oxide thick=0.025

#channel implant
implant boron dose=1.0e12 energy=15.0

#deposit the gate poly
deposit poly thick=0.500 div=10 phos conc=1.0e19
#anneal the beast
diff time=10 temp=1000

#etch the poly away
etch poly right p1.x=0.55 p1.y=-0.025 p2.x=0.45 p2.y=-0.55
#anneal this step
diffuse time=30.0 temp=950
struct outf=poly.str

#do the phosphorus implant
implant phos dose=1.0e13 energy=50.0
```

```
#deposit the oxide spacer
deposit oxide thick=0.400 div=10 spac=0.04
#etch the spacer back
etch dry oxide thick=0.420

struct outf=imp2.str

#after etch anneal
method vert
diffuse time=30 temp=950 dry

#implant the arsenic
implant ars dose=5.0e15 energy=80.0

#deposit a cap oxide
deposit oxide thick=0.15 space=0.03
struct outf=imp4.str

#do the final anneal
diffuse time=20 temp=950

struct outf=ldd.str
```

The next section begins the definition of the mesh to be used for the simulation. The section

```
phos poly /gas Trn.0=0.0
bor poly /gas Trn.0=0.0
phos oxide /gas Trn.0=0.0
bor oxide /gas Trn.0=0.0
```

turns off the out diffusion of phosphorus and boron. In this simulation, there are anneals of bare poly. To keep the time step from being limited by the out diffusion behavior, the gas transport is turned off by setting the transport rate preexponential constant to zero.

The next section describes the locations of the x lines in the mesh.

```
line x loc=0.0 tag=lft spacing=0.25
line x loc=0.45        spacing=0.03
line x loc=0.75        spacing=0.03
line x loc=1.4         spacing=0.25
line x loc=1.5 tag=rht spacing=0.25
```

SUPREM-IV defines x to be the direction across the top of the wafer, and y to be the vertical dimension into the wafer. Similar to example 4, the location of the mesh lines is chosen to minimize the spatial error and to represent the location of various etch steps.

The next section of input describes the location and spacings of the vertical mesh lines.

```
#the y dimension
line y loc=0.0 tag=top spacing=0.01
line y loc=0.1         spacing=0.01
line y loc=0.25        spacing=0.05
line y loc=3.0 tag=bot
```

The lines are chosen close together near the surface and increasing away from it. Tight spacing is maintained only near the top surface. Since the structure will be passed to Pisces, a depth of 3.0μm is chosen so that the eventual substrate contact will be deep to not unduly influence the simulation.

The next two sections describe the device starting material and the surfaces which are exposed to gas.

```
#define the starting material
region silicon xlo=lft xhi=rht ylo=top yhi=bot

#define the the exposed surface
bound exposed xlo=lft xhi=rht ylo=top yhi=top
bound backside xlo=lft xhi=rht ylo=bot yhi=bot
```

The region statement is used to define the starting materials. In this case the wafer is silicon with no initial masking layers. The bound statement allows the definition of the the front and backsides of the wafer. Any gasses on the diffusion statement, depositions, and etches are applied to the surface marked exposed. The backside will not influence the process simulation. However, the backside is the location of a contact when the specfication of the electrodes is made for Pisces.

The next line informs SUPREM-IV that the mesh has been defined and should be computed.

```
#the starting wafer doping
init boron conc=1.0e16

#deposit the gate oxide
deposit oxide thick=0.025
```

The first line initializes the starting material. The deposited oxide is specified to 0.025 microns thick. This represents the gate oxide that is grown before the uniform boron implant.

The next statement performs the channel implant of the boron.

```
#channel implant
implant boron dose=1.0e12 energy=15.0 pears
```

The implant is modeled with a Pearson-IV distribution. The energy and dose are $1 * 10^{12}\ cm^{-2}$ and 15 Kev respectively. This produces an abrupt boron profile with a shallow junction.

The next cards define the poly deposition and anneal. The poly deposition is done first, then the heat cycle that follows does the temperature cycle that occurs during actual poly deposition.

```
#deposit the gate poly
deposit poly thick=0.500 div=10 phos conc=1.0e19
#anneal the beast
diff time=10 temp=1000

#etch the poly away
etch poly right p1.x=0.55 p1.y=-0.025 p2.x=0.45 p2.y=-0.55
#anneal this step
diffuse time=30.0 temp=950

struct outf=poly.str
```

The first pair of statements deposit the poly and then provide a temperature step representing the deposition. This temperature step will cause diffusion of the threshold adjust implant. The poly is put down with ten grid layers and doped to concentration of 10^{19} phosphorus. The next statements etch the gate material and perform a poly anneal. The structure is then saved in the file "poly.str"

The phosphorus light implant and spacer definition comes next.

```
#do the phosphorus implant
implant phos dose=1.0e13 energy=50.0

#deposit the oxide spacer
deposit oxide thick=0.400 div=10 spac=0.04
#etch the spacer back
etch dry oxide thick=0.420

struct outf=imp2.str
```

This implant is the the one which will form the lightly doped drain. The next two steps of the full device are to deposit and etch the spacer oxide. The spacer is deposited with 10 divisions, and the curvature is resolved to 0.04μm. The oxide is etched back a thickness of 0.42μm. The dry parameter indicates that the oxide surface is to be dropped straight down. Finally, the structure at this stage is saved.

The oxide is regrown and the phosphorus annealed during the next step.

```
#after etch anneal
method vert
diffuse time=30 temp=950 dry
```

The method card specifies the oxide will grow vertically. No injection of interstitials will be simulated during this oxide growth. The diffuse line specifies that the anneal is to be done for 30 minutes at 950°C and in dry O_2.

The next step implants the heavily doped portion of the drain, and deposits the cap oxide.

```
#implant the arsenic
implant ars dose=5.0e15 energy=80.0

#deposit a cap oxide
deposit oxide thick=0.15 space=0.03
struct outf=imp4.str
```

The arsenic is implanted through the new oxide growth and then a cap oxide is deposited. The structure is saved in the file "imp4.str".

Finally, we want to do the final anneal.

```
#do the final anneal
diffuse time=20 temp=950

struct outf=ldd.str
```

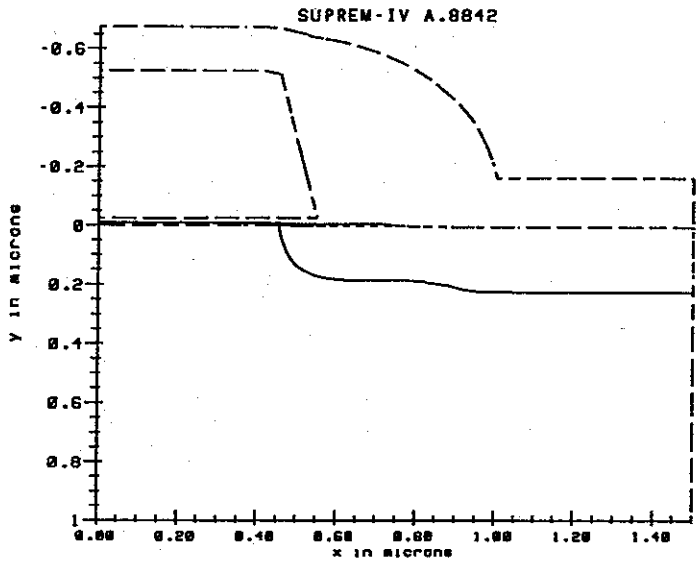
This anneal will be carried out for 20 minutes at 950°C. The structure is saved in the file ldd.str. At this point, any of the structure files can be read back and examined to check the results. Commands and analysis similar to that found in example 4 may be performed.

To just perform a simple check, the following commands should produce a plot.

```
plot.2d bound fill y.max=1.0
sel z=log10(phos+ars)-log10(bor)
contour val=0.0
```

The first command plots the device outline. The second command selects the difference in the logs of the

doping. This tends to produce much smoother contours for plotting purposes. The final command plots the junction location, as shown below.



To prepare a Pisces structure file, the following input deck, found in example/exam5/sup2pis.s4 can be used.

```
#make the contact hole
etch oxide right p1.x=1.4

#put down the aluminum and etch off
deposit alum thick=0.1
etch alum left p1.x=1.4

#remove extra grid nodes to save Pisces compute time
etch start x=-0.5 y=-0.1
etch cont  x=1.6 y=-0.1
etch cont  x=1.6 y=-1.0
etch done  x=-0.5 y=-1.0

#reflect the structure
struct mirror left

#save it in Pisces format
struct pisc=ldd.mesh
```

The first line etches the oxide to form the drain contact hole. Aluminum is deposited and etched to form contact to the drain. Everything more than 0.1μm above the surface silicon is removed. This material is uninteresting to Pisces since it can not solve for anything more than the silicon substrate. The poly and oxide would just be wasted nodes. This etch uses the polygon specification for illustration. Any number of points forming a polygon can be specified. The material inside this polygon will be removed. The structure is reflected around the left edge. This doubles the number of nodes and forms the complete device with both source and drain contacts.

The structure is saved in Pisces format in the file ldd.mesh. This command also reports the location of contacts in the mesh for Pisces.

```
Electrode 1: xmin    -0.550 xmax     0.550 ymin    -0.100 ymax    -0.025
Electrode 2: xmin     1.400 xmax     1.500 ymin    -0.100 ymax     0.005
```

```
Electrode 3: xmin   -1.500 xmax   -1.400 ymin   -0.100 ymax    0.005
Electrode 4: xmin   -1.500 xmax    1.500 ymin    3.000 ymax    3.000
```

From this information, the gate is electrode number 1, the source and drain are 2 and 3, and the backside contact is number 4. The file ldd.mesh can be read by the latest release of pisces and is a "geom" type file.

SUPREM IV

Level: Examples

Version: 1.1 11/21/88 13:15:09

DESCRIPTION

This example compares the oxide thickness grown for different orientations. The input file for the simulation is in the "examples" directory, in the file "oxcalib.s4".

```
#---some set stuff
option quiet
unset echo

foreach O (100 110 111)
echo "---------- Orientation < O > --------------------"

#---the minimal mesh
line x loc = 0.0 tag = left
line x loc = 1.0 tag = right

line y loc = 0.0 tag = top
line y loc = 1.0 tag=bottom

region silicon xlo = left xhi = right ylo = top yhi = bottom

bound exposed  xlo = left xhi = right ylo = top  yhi = top

init ori=O

#---the oxidation step
meth     grid.ox=0.03
diffuse time=30 temp=900 wet

select z=y*1e8 label="Depth(A)"; print.1d x.v=0 form=8.0f

end
```

The first two lines keep the program as quiet as possible. The minimum output is printed, and the input is not echoed at all.

The loop

```
foreach O (100 110 111)
...
end
```

runs its body three times for three different orientations.

The first 7 lines of the loop body define the simplest mesh possible. It has exactly four lines (one rectangle), and a front side.

The method card chooses the oxide grid spacing to be 0.03 microns, knowing that 30 minutes at 900°C will grow about 0.1 microns. The oxidation step chooses the vertical model by default. The vertical model solves the Deal-Grove diffusion equation for oxidant, and uses the calculated velocities to move the oxide vertically everywhere. For the purposes of thickness on a planar or near-planar structure, the accuracy is good enough. Grid is added automatically to the oxide as it grows, by default at increments of 0.1μm. The

time step is initially 0.1s; subsequently it is automatically chosen to add no more than ¼ of the grid increment per time step.

When the diffusion is done, the select statement fills the "z vector" with the y (vertical) locations of the points, scaled to angstroms. The print statement then prints the y displacements along the left side of the mesh. The form parameter rounds off the thickness to the nearest angstrom. The output is

```
"---------- Orientation < 100 > --------------------"
estimated first time step 1.000000e+37
Solving            0 +          0.1 =          0.1,      100%, np 6
Solving          0.1 +       301.28 =       301.38,   301280%, np 6
Solving       301.38 +       301.283 =       602.663,  100.001%, np 6
Solving       602.663 +      309.771 =       912.433,  102.817%, np 6
Solving       912.433 +      318.027 =       1230.46,  102.665%, np 6
Solving       1230.46 +      326.295 =       1556.75,    102.6%, np 8
Solving       1556.75 +      243.245 =          1800,  74.5477%, np 8
y coordinate in um          Depth(A) Material
       -0.04404786       -440  oxide
       -0.00202714        -20  oxide
        0.03503988        350  oxide
        0.03503988        350  silicon
        0.99999997      10000  silicon
        0.99999997      10000  silicon
"---------- Orientation < 110 > --------------------"
estimated first time step 1.000000e+37
Solving            0 +          0.1 =          0.1,      100%, np 6
Solving          0.1 +       215.553 =       215.653,  215553%, np 6
Solving       215.653 +      215.557 =       431.209,  100.002%, np 6
Solving       431.209 +      224.045 =       655.254,  103.938%, np 6
Solving       655.254 +      232.212 =       887.466,  103.645%, np 6
Solving       887.466 +      240.401 =       1127.87,  103.527%, np 8
Solving       1127.87 +      248.601 =       1376.47,  103.411%, np 8
Solving       1376.47 +      256.809 =       1633.28,  103.302%, np 8
Solving       1633.28 +      166.724 =          1800,  64.9213%, np 10
y coordinate in um          Depth(A) Material
       -0.05755641       -576  oxide
       -0.01580480       -158  oxide
        0.02408379        241  oxide
        0.04629701        463  oxide
        0.04629701        463  silicon
        0.99999997      10000  silicon
        0.99999997      10000  silicon
"---------- Orientation < 111 > --------------------"
estimated first time step 1.000000e+37
Solving            0 +          0.1 =          0.1,      100%, np 6
Solving          0.1 +       179.762 =       179.862,  179762%, np 6
Solving       179.862 +      179.766 =       359.628,  100.003%, np 6
Solving       359.628 +      188.254 =       547.882,  104.722%, np 6
Solving       547.882 +       196.36 =       744.242,  104.306%, np 6
Solving       744.242 +      204.498 =        948.74,  104.144%, np 8
Solving        948.74 +      212.649 =       1161.39,  103.986%, np 8
Solving       1161.39 +      220.812 =        1382.2,  103.839%, np 8
Solving        1382.2 +      228.986 =       1611.19,  103.702%, np 10
Solving       1611.19 +      188.814 =          1800,  82.4564%, np 10
y coordinate in um          Depth(A) Material
```

```
-0.06573780      -657  oxide
-0.02416809      -242  oxide
 0.01549940       155  oxide
 0.05311484       531  oxide
 0.05311484       531  silicon
 0.99999997     10000  silicon
 0.99999997     10000  silicon
```

The thickness is the difference between the top and bottom readings for the oxide, namely 790A, 1039A and 1188A for the <100>, <110> and <111> orientations respectively.

SUPREM IV

Level: Examples

Version: 1.1 11/21/88 13:15:11

DESCRIPTION

This example shows the evolution of an oxide profile during semi-recessed oxidation. It uses the Deal-Grove model to calculate the oxide growth rate at the silicon/oxide interface, and treats the deformation of the oxide and nitride as viscous incompressible flow. The input file for the simulation is in the "examples" directory, in the file "fullrox.s4".

```
# --Recessed LOCOS cross section: recess 0.3 um, grow 0.54um (0.23 + 0.31)--
option quiet

#------------Substrate mesh definition
line y loc=0  spac=0.05 tag=t
line y loc=0.6 spac=0.2
line y loc=1            tag=b

line x loc=-1   spac=0.2 tag=l
line x loc=-0.2 spac=0.05
line x loc=0  spac=0.05
line x loc=1  spac=0.2 tag=r

region silicon xlo=l xhi=r ylo=t yhi=b
bound  expo    xlo=l xhi=r ylo=t yhi=t

init or=100

#-----------Anisotropic silicon etch
etch    silicon left p1.x=-0.218 p1.y=0.3 p2.x=0 p2.y=0

#----------Pad oxide and nitride mask
deposit oxide thick=0.02
deposit nitride thick=0.1
etch    nitride left p1.x=0
etch    oxide   left p1.x=0

plot.2 g line.g=1 b line.b=3

#----------Field oxidation
meth compr

plot.2 b y.mi=-0.5 line.b=2

diffuse tim=90 tem=1000 weto2 movie="plot.2 b clear=f axis=f line.b=1"
stru    outf=fc.mesh

plot.2 b flow vleng=0.1
```

The first line

```
option quiet
```

asks for as little output as possible.

The grid definition comes next. First the vertical lines are defined:

```
line y loc=0 spac=0.05 tag=t
line y loc=0.6 spac=0.2
line y loc=1 tag=b
```

Ninety minutes in 1000°C steam grows about 0.54 of oxide on <100> silicon. In the process, 0.24 of silicon is consumed. The second y line is around the expected final depth of the oxide, and there is one more line at 1 to round out the grid.

```
line x loc=-1    spac=0.2 tag=l
line x loc=-0.25 spac=0.05
line x loc=0     spac=0.05
line x loc=1     spac=0.2 tag=r
```

The x lines run from -1 to 1 for symmetry, with the mask edge at 0. The extra refinement around 0.25 is to prepare for the silicon etch, which will be at an angle of 54° and therefore has a lateral extent of $0.3/\tan 54° \simeq 0.218$.

```
region silicon xlo=l xhi=r ylo=t yhi=b
bound  expo    xlo=l xhi=r ylo=t yhi=t
```

The region statement identifies the entire area as silicon substrate. It refers to the tags defined on the x and y lines. These tags are used to label lines uniquely so that new lines can be added or subtracted easily without renumbering. The boundary statement identifies the top surface as being exposed. (SUPREM-IV does not assume the top is exposed.) Layer depositions, oxidations and impurity predepositions only happen on "exposed" surfaces, so this statement must not be omitted.

```
init ori=100
```

The init card causes the initial rectangular mesh to be generated. The substrate orientation is <100>.

```
#----------Anisotropic silicon etch
etch    silicon left p1.x=-0.218 p1.y=0.3 p2.x=0 p2.y=0
```

The silicon substrate is etched. The etch statement removes all silicon found lying to the left of the line joining the points (-0.218, 0.3) and (0,0).

```
#----------Pad oxide and nitride mask
deposit oxide thick=0.02
deposit nitride thick=0.1
etch    nitride left p1.x=0
etch    oxide   left p1.x=0
plot.2 grid bound
```

The pad oxide is put down, then nitride is deposited on top and patterned. The patterning is presumed to remove the underlying pad oxide also. The plot command shows the structure before oxidation (Figure 7.1).

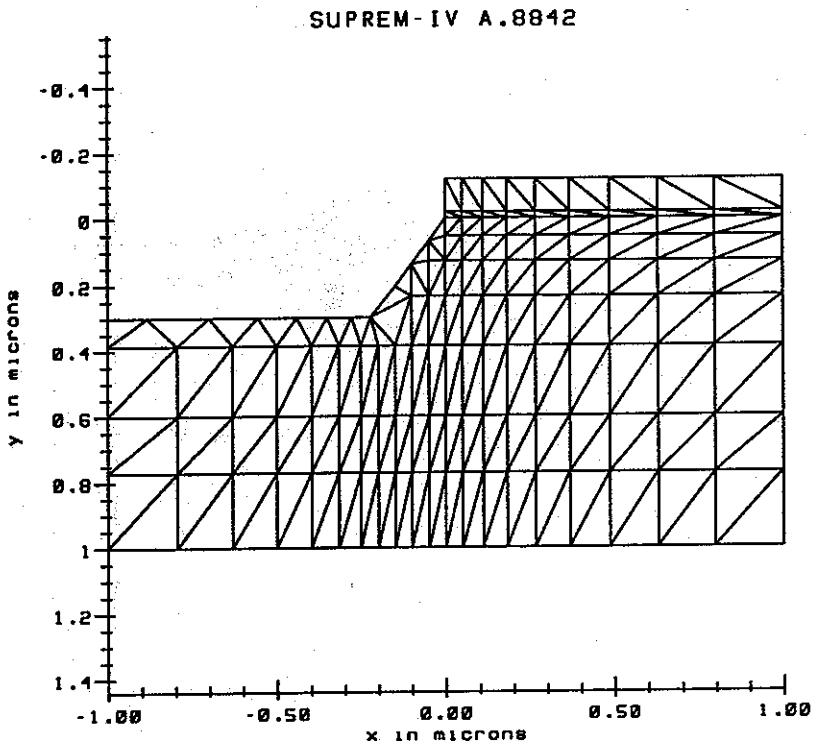


Figure 7.1 - Recessed structure before oxidation

```
#----------Field oxidation
meth compr
```

The compressible flow model is chosen for oxidation. The incompressible model could also have been used, but for this example where the stress is not desired, the compressible model is faster and nearly as accurate.

```
plot.2 b y.mi=-0.5 line.b=2
diffuse tim=90 tem=1000 weto2 movie="plot.2 b clear=f axis=f line.b=1"
```

The diffuse statement is the point of the exercise. For 90 minutes, oxide grows at the silicon interface. The new oxide being formed pushes up the old oxide and nitride layers which cover it, causing the characteristic bird's head profile. The `movie` parameter lists any extra actions to take at each time step. In this case, the boundary is plotted without erasing or rescaling the picture. A series of outlines is generated, one from each time step, illustrating the evolution of the oxide profile (Figure 7.2). The first `plot.2 b` defined a plotting window large enough so that the subsequent plots without axes would fit inside it.

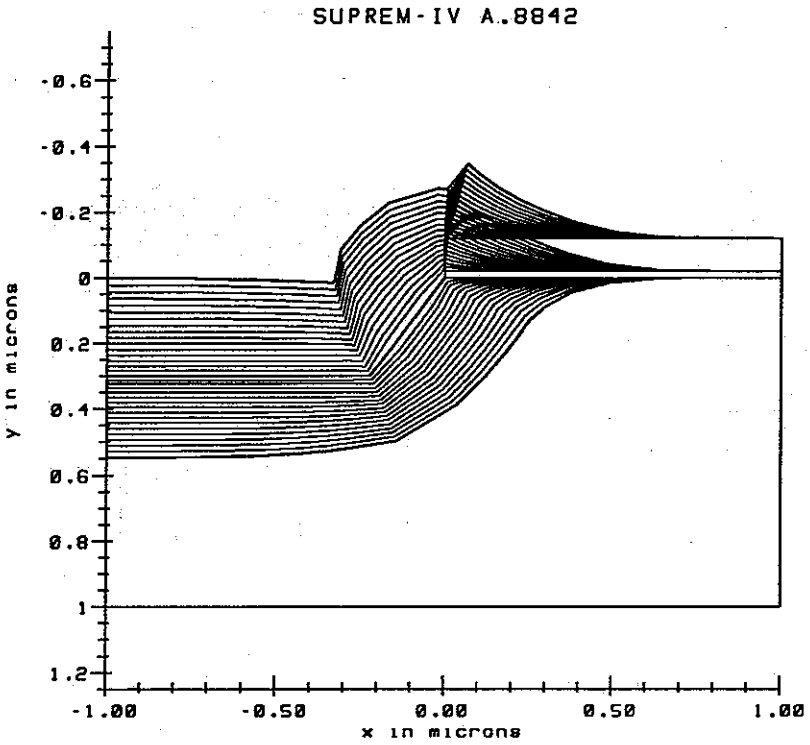


Figure 7.2 - Evolution of bird's head profile

The flow pattern at the last time step can be plotted with the statement

```
plot.2 bound flow vleng=0.1
```

The parameter `vleng` is the length to draw the longest velocity vector. The other vectors are scaled proportionally. This results in figure 7.3. The flow is normal at the interface but becomes more vertically oriented away from it.

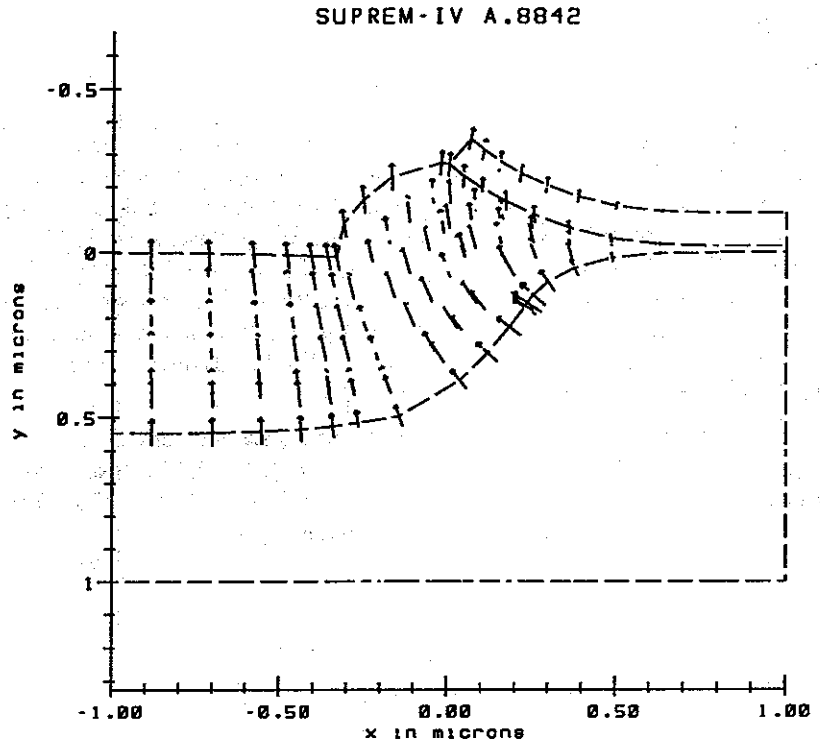


Figure 7.3 - Flow vectors at end of oxidation

SUPREM IV

Level: Examples

Version: 1.1 11/21/88 13:15:12

DESCRIPTION

This example calculates the extent of shear stress in the silicon substrate, generated by a film edge. It is well known that when is deposited on silicon by chemical vapor deposition, a large intrinsic stress is present in the layer. If the film is continuous and sufficiently thin, this stress does not present a problem. The substrate is much thicker (~1000 times) than the film, so the substrate stress is 1000 times less than the film stress. If the film is etched, however, there is a localized stress in the silicon close to the film edge which is of the same order of magnitude as the stress in the film. This large stress can induce dislocations directly, or indirectly through several mechanisms. It can also induce dislocations to glide from implanted regions into masked regions, causing junction leakage. Similar stress patterns are set up by thermal expansion mismatches between the substrate and overlying films.

A simulation of the substrate stress induced by a nitride film is shown below. This example can be found in the "`examples`" directory under the name "`nit-stress.s4`". A <111> substrate is considered with mask edges aligned along the <110> directions. The nitride film is 0.02μm thick. The intrinsic stress in the nitride is assumed to be 1.4×10^{10} dynes/cm^2, as reported in [1]

The input deck for the simulation begins as follows.

```
#
#  nitride on silicon example
foreach SEP ( 10 5 4 2.5 )

line x loc=( - SEP ) tag=l
line x loc=-2 spac=0.3
line x loc= 0 spac=0.1 tag=m
line x loc= 2 spac=0.3
line x loc= ( SEP ) tag=r

line y loc=0 spac=0.1 tag=si
line y loc=2 spac=0.3
line y loc=5 tag=b

region silicon xlo=l xhi=r ylo=si yhi=b
bound  expos   xlo=l xhi=r ylo=si yhi=si
initial ori=111
```

Quite a large piece of substrate is analysed, starting with $10\mu m\times5\mu m$. The foreach loop chooses four different widths to examine the effect of different stripe separations. The structure being simulated has reflecting boundary conditions (no perpendicular displacement) on the left and right sides. Those boundary conditions correspond to a single instance of a repeating pattern of nitride stripes, shown schematically in Figure 8.1. When the stripes are widely separated, there is little interference between the stress field of one stripe and the next, and the results are found to be independent of the stripe separation. At smaller separations, the patterns begin to overlap. This example will show the stress pattern for widely separated stripes and the modifications that occur as the stripes are brought closer together.

The backside of the wafer also has a reflecting boundary condition. Although that approximation is not quite physical, the nitride film primarily exerts a horizontal force on the substrate, the assumption does not seriously affect the result. This can be verified by using different backside thicknesses; 2μm or 100μm gives identical results. The exposed surface is the only surface with free displacements.

The spacing around the film edge is 0.1μm. This could be reduced for more accuracy, at the cost of more cpu time.

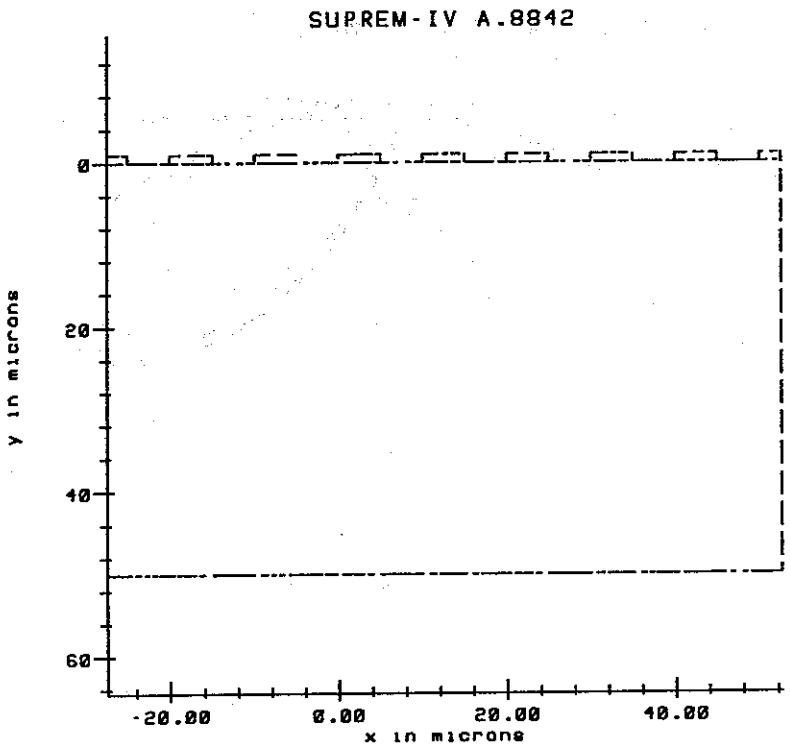


Figure 8.1 - Schematic of Structure

```
deposit nitride thick=0.02 div=2
etch    nitride left p1.x = 0
```

The nitride is deposited directly on silicon and patterned.

```
material    intrin.sig = 1.4e10 nitride
```

The initial nitride stress is specified in dynes/cm^2.

```
stress   temp1=1000 temp2=1000
```

This calculates the stress distribution arising from the nitride initial stress. Stress arising from thermal expansion mismatch would be included by specifying a different `temp2`. Be warned that large thermal steps often bring into play many more complicated phenomena than the simple thermal expansion mismatch analyzed in SUPREM-IV. For instance breakdown of film adhesion or structural change may occur, but are not taken into account in the program.

The principal slip system in silicon is in the <110> direction on {111} planes [2]. This corresponds to the σ_{xy} shear force in the plane of the simulation. The value 3×10^7 is considered by Hu to be the critical shear stress for slip in silicon. Dislocations found in regions where the shear stress is larger than that value will move under the stress field of the nitride film. Therefore when analyzing stress in the substrate, a principal concern is the extent of the $\sigma_{xy} = 3\times10^7$ contour.

```
plot.2  bound  x.mi=-2 x.ma=2 y.ma=4 cl=f
```

```
sel      z=Sxy
contour  val=-3e7
```

The contours of $\sigma_{xy} = 3\times10^7$ in the silicon is shown in Figure 8.2.

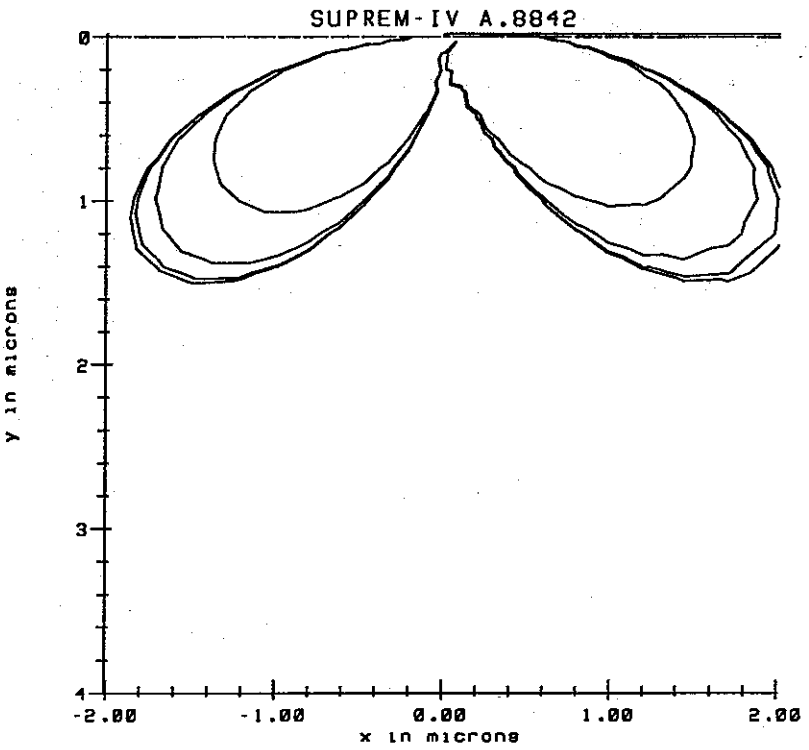


Figure 8.2 - Critical Shear Stress Contours

The contour is a double lobe because the shear stress is related to the polar components of stress through [3]

$$\sigma_{xy} = (\sigma_{rr} - \sigma_{\theta\theta})\sin2\theta + \sigma_{r\theta}\cos2\theta \approx \sigma_{rr}\sin2\theta$$

The function $\sin2\theta$ is at a maximum around 45° from the vertical, and is zero at $\theta = 90°$. Both σ_{rr} and $\cos2\theta$ change sign moving from left to right, so that the sign of σ_{xy} is the same throughout. This means that a dislocation which is being driven under the mask by the stress field will continue to move in that direction after passing the mask edge. However as it passes through the center, the decrease in shear stress may leave it stranded under the mask edge. The figure shows that the area of influence of the nitride film is many times greater than its thickness, about $\pm2\mu m$ horizontally, and 1.5μm vertically.

The largest lobe corresponds to the 10μm stripe. The result for the 5μm stripe lies so closely on top that it cannot be distinguished. Thus either can be a reasonable approximation for infinitely separated stripes. The 4μm and 2.5μm lobes are somewhat smaller. This shows that the shear stress fields from neighboring stripes tend to cancel each other in their overlap areas, reducing the influence of the nitride film somewhat.

1. E.A. Irene, "Residual Stress In Silicon Nitride Films," *J. Electronic Mat.*, vol. 5(3), p. 287, 1976.

2. S.M. Hu, "Film-edge-induced stress in silicon substrates," *Appl. Phys. Lett*, vol. 32, p. 5, 1978.

3. S.P. Timoshenko and J.N. Goodier, *Theory of Elasticity*, McGraw-Hill, New York, 1970.

SUPREM IV

Level: Examples

Version: 1.1 11/21/88 13:15:14

DESCRIPTION

This example shows the use of the stress-dependent oxidation model. Experimental LOCOS profiles are generally of two distinct types [1]. When the nitride mask is more than 2-3 times thicker than the pad oxide, the oxide/silicon interface and the oxide gas surface is kinked. For thinner nitride masks, the shape can be approximately be described by an error-function [2]. The kinks were not observed in the first generation of oxidation simulators, because they result from stress effects on the growth coefficients. Early oxidation programs did not take stress into account and found essentially identical oxide shapes irrespective of nitride thickness. This example shows how the stress-dependent model in SUPREM-IV can predict such second-order effects. The example is given in the file "**sdep.s4**" in the "**examples**" directory.

```
# --LOCOS cross section with stress effects

opt      chat

# --Substrate mesh definition--
line x loc=-1 spac=0.2 tag=l
line x loc=0  spac=0.05 tag=c
line x loc=1  spac=0.2 tag=r

line y loc=0 tag=t
line y loc=1 tag=b

region silicon xlo=l xhi=r ylo=t yhi=b
bound  expo    xlo=l xhi=r ylo=t yhi=t

init ori=100

# --Pad oxide and nitride mask--
deposit oxide thick=0.02 div=1
deposit nitride thick=0.05 div=1
etch    nitride left p1.x=0

#----------Field oxidation
oxide   Vc=300 Vr=30  Vd=25 stress.dep=t
mater   nitride visc.0=1.8e15

meth   viscous oxide.rel=1e-2

plot.2 b  y.mi=-0.3 x.mi=-0.3 x.ma=0.3

diffuse tim=90 tem=1000 weto2  movie="plot.2 b cl=f ax=f line.b=1; stru outf

stru outf=sdep10.str
```

The grid structure and nitride/oxide sandwich is very similar to the fully-recessed oxide example. The substrate grid is as sparse as possible (two lines). The lateral grid is a little coarse to compensate for the increased computation time used in the nonlinear model. The new statements are as follows:

```
oxide stress.dep=t
```

The nonlinear stress-dependent model is turned on. The default is to turn it off since it is much more expensive to run than the linear model. The activation volume for plastic flow **Vc**, for the stress-dependent reaction rate **Vr**, and for the diffusivity **Vd**, are taken from the defaults in the model file. The values used were derived by fitting Kao's cylinder oxidation data [3].

```
mater    nitride visc.0=1.8e15
```

The nitride is given a viscosity ten times that of wet oxide at this temperature. The default of 1e12 is rather low. Experimental data is not available. Ten times the oxide viscosity is a reasonable guess.

```
meth    viscous oxide.rel=1e-2
```

The viscous model is chosen, because only this model takes stress effects into account. (Only the viscous model calculates stress accurately enough to feed back into the coefficients). The relative error criterion is chosen as 1% in the velocities. The default of 10^{-6} is rather tight and requires more CPU time. The **diffuse** statement is as usual.

This input file was executed twice, once with 0.05 µm of nitride and once with 0.15µm of nitride. The output contains the usual features, along with many lines as follows:

```
Newton loop  0  cut         1 upd      0.9914 orhs       3.811 rhs  3.534e-15
Newton loop  1* cut         1 upd   1.322e-12 orhs   3.534e-15 rhs       1e-38
Continuation step #0 to lambda = 0.25 step 0.25
Newton loop  0  cut         1 upd    0.008394 orhs   0.0002229 rhs  0.0003045
Newton loop  0  cut      0.25 upd    0.008394 orhs   0.0002229 rhs   0.000135
Newton loop  2  cut      0.25 upd    0.005944 orhs   0.0001278 rhs   6.63e-05
Newton loop  4  cut       0.5 upd    0.004301 orhs   6.479e-05 rhs  1.638e-05
Newton loop  5* cut         1 upd    0.002137 orhs   1.638e-05 rhs   1.43e-05
Newton loop  7  cut         1 upd   0.0004636 orhs   1.625e-05 rhs       1e-38
Continuation step #0 to lambda = 0.5 step 0.25
Newton loop  0  cut         1 upd      0.1527 orhs      0.0359 rhs    0.04011
...
Newton loop  4  cut     0.031 upd       2.062 orhs     0.01961 rhs     0.0199
Continued too far, backing off.
Continuation step #-1 to lambda = 0.375 step 0.125
Newton loop  0  cut         1 upd     0.06759 orhs     0.02686 rhs    0.02223
Newton loop  2  cut         1 upd      0.1005 orhs     0.02204 rhs    0.02313
...
Newton loop 28  cut       0.5 upd     0.00263 orhs   0.0005139 rhs  0.0002645
Newton loop 30  cut         1 upd   0.0009523 orhs   0.0002647 rhs       1e-38
Continuation step #0 to lambda = 0.5 step 0.125
Newton loop  0  cut         1 upd     0.01423 orhs     0.01075 rhs   0.008121
Newton loop  2  cut         1 upd    0.002104 orhs    0.008299 rhs   0.003507
Newton loop  4  cut         1 upd    0.002404 orhs    0.003543 rhs   0.007114
Newton loop  4  cut      0.25 upd    0.002404 orhs    0.003543 rhs   0.002892
Newton loop  6  cut      0.25 upd   0.0007027 orhs    0.002888 rhs       1e-38
Continuation step #0 to lambda = 1 step 0.5
Newton loop  0  cut         1 upd   0.0005178 orhs    0.002156 rhs       1e-38
```

The nonlinear solver works by proceding first from the linear solution. The linear solution takes exactly two Newton steps. The first has a large update step (**0.9914**), which reduces the error from **3.811** to **3.5e-15**. The second Newton step (**1.3e-12**) then of course is trivial since the solution has been reached. The Newton loop counter is incremented by two whenever a new Jacobian is factorized, and by one when only the error is recalculated.

The stress-dependent viscosity is turned on (continuation step 0 to λ=0.25). The first step is moderately large (**0.008394**) and causes the error to increase (**0.00022** to **0.0003045**). A quarter-step (**cut 0.25**) in that direction is tried, which is found to decrease the error from **0.00022** to **0.0001**. This new position is accepted as worthwhile, and a second Newton step is taken from there. It is found to decrease the error from **0.0059** to **0..00012**. Since this is successful, the cutback factor is increased back to 0.5. The third Newton step (#4) reduces the error by a large factor, so on the subsequent step not only is the length increased back to 1 but the same Jacobian is recycled, indicated by the asterisk. The recycled Jacobian proves not to be effective, since the error only goes from **1.6e-5** to **1.4e-5**. One more Jacobian is factored, and causes an update of **0.0004636**, less than the accuracy criterion. The first nonlinear problem has been solved.

The stress-dependent reaction rate is then turned on (continuation to 0.5). In this case, the Newton process fails. Successively smaller steps along the Newton direction are taken, but at each attempted new position the error is larger than the current location. The smallest step tried is 0.031; the next would be 0.031/4 which is less than 1% of the Newton update and the situation is considered hopeless. The program then backs off by trying an intermediate λ of 0.375. This corresponds to a stress dependence which is only half as strong as the desired dependence. After 15 Newton steps, this weaker problem is solved. The solution is then used as an initial guess for the problem first tried, that with the full stress dependence (λ=0.5). The improved initial guess leads to a successful solution of the problem after 3 Newton steps. Finally, the stress-dependent diffusivity is turned on (λ=1). Since the activation volume for diffusivity was left at 0, nothing of interest happens and the problem is solved after one loop. The total number of Newton loops is therefore about 24, compared to 1 for a linear problem. Thus the nonlinear problem is about 24 times as expensive to solve as a linear problem.

The different oxide shapes are shown in Figure 9.1a and 9.1b.

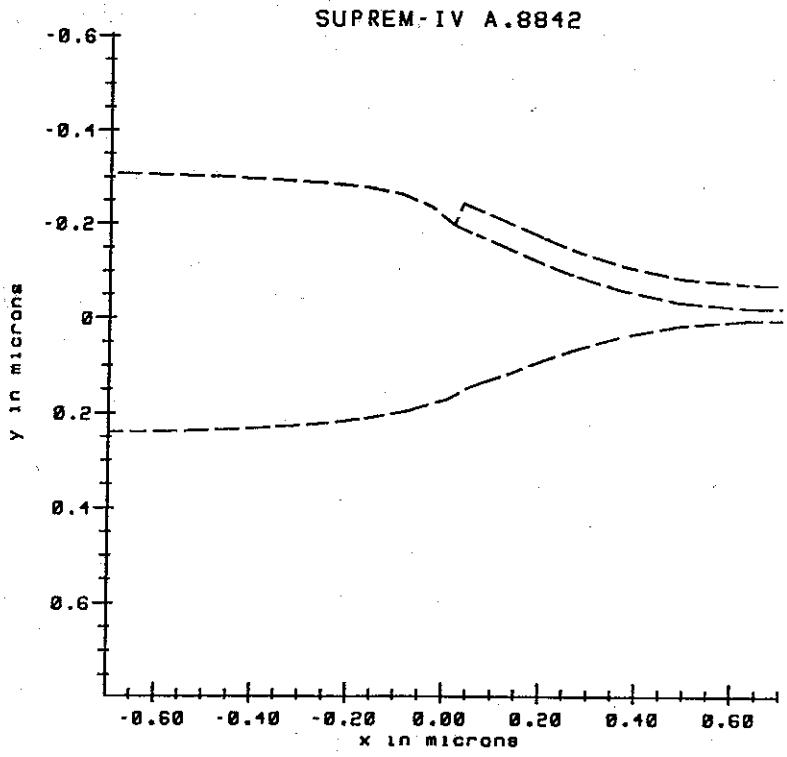


Figure 9.1a - LOCOS with 200Å pad oxide and 500Å nitride

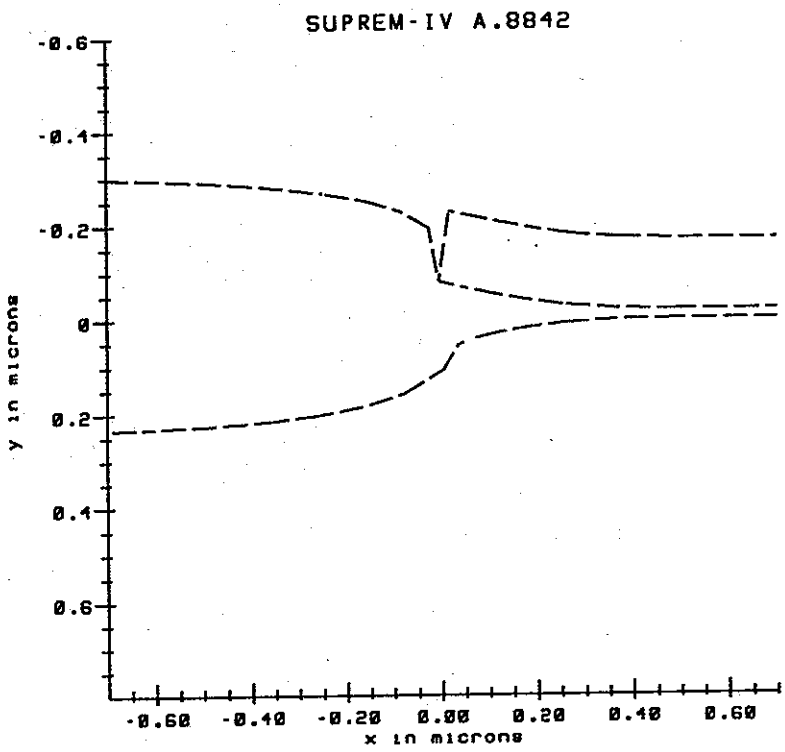


Figure 9.1b - LOCOS with 200Å pad oxide and 1500Å nitride

The thicker nitride clearly has the effect of reducing the bird's beak.

1. N. Guillemot, G. Pananakakis, and P. Chenevier, "A New Analytical Model of the "Bird's Beak"," *IEEE Transactions on Electron Devices*, vol. ED-34, May 1987.

2. A.M-R. Lin, "The Effect of Oxidation on Impurity Diffusion and Stacking Fault Growth in Silicon," PhD dissertation, Department of Electrical Engineering, Stanford University, March 1982.

3. D-B. Kao, J.P. McVittie, W.D. Nix, and K.C. Saraswat, "Two Dimensional Thermal Oxidation of Silicon I. Experiments," *IEEE Transactions on Electron Devices*, May 1987.