SUPREM-IV.GS

Two Dimensional Process Simulation for Silicon and Gallium Arsenide

Edited by

S. E. Hansen and M. D. Deal

Integrated Circuits Laboratory
Stanford University
Stanford, California 94305

Table of Contents

Acknowledgments

This manual, like the SUPREM-IV.GS program itself, is the work of many people. The original SUPREM-IV manual was primarily written by Mark E. Law with contributions by Conor Rafferty, Goodwin Chin, and Chih-Chuan Lin. The modifications and additions needed for SUPREM-IV.GS were done by Michael D. Deal and Stephen E. Hansen.

Section	**Contributors**
Introduction	M. Deal, S. E. Hansen
Adding SUPREM 3.5's GaAs Models ...	M. Deal, S. E. Hansen
User's Reference Manual	M. E. Law, C. S. Rafferty, G. Chin, C. C. Lin, S. E. Hansen
Examples	M. E. Law, C. S. Rafferty, M. Deal, S. E. Hansen

Introduction

SUPREM-IV.GS is the first version of SUPREM-IV which models GaAs and its dopants in addition to modeling silicon fabrication technology. SUPREM-IV is an advanced 2D process simulator originally developed for submicron silicon structures. It provides cross-sections of arbitrary device structures based on physical models for ion implantation, diffusion, oxidation and annealing. It is designed to interface with other programs that accurately simulate etching and deposition of thin films on the semiconductor surface, although it includes basic models for these processes as well. The program structure is designed to handle moving boundaries that commonly occur during oxidation or during deposition or etching of thin films. SUPREM-IV can model stress gradients produced by overlaying films. The implantation, diffusion annealing and oxidation models are point defect based and are believed to be the most advanced models used in any process simulation program. While the oxidation capability will not be needed for simulating GaAs structures, all the other modules can be directly applied to compound materials.

In version 9305 of SUPREM-IV.GS, we started with version 9130 of SUPREM-IV and incorporated most of SUPREM 3.5’s 1D GaAs models and parameters. This involved modifying SUPREM-IV so that it recognizes a new semiconductor material and new dopants, and allows for new expressions and parameters for the GaAs dopants in terms of the various fabrication processes.

SUPREM 3.5 was developed to provide 1D concentration-depth profiles through arbitrary cross-sections of GaAs devices for the specific processing conditions being simulated. SUPREM 3.5 is primarily designed for modeling processes used to make simple ion implanted MESFET and JFET structures in semi-insulating GaAs, with or without buried p-layers. The main processes modeled are ion implantation (into bare GaAs or through a silicon nitride or oxide cap) and the activation anneal (which includes dopant diffusion and activation). The dopants modeled are: Si, Se, Ge, and Sn (the 4 common n-type dopants); and Be, Mg, Zn, and C (the 4 common p-type dopants). Also modeled is the diffusion of non-implanted dopants, such as those incorporated during MBE or MOCVD GaAs growth, which sometimes have different effective diffusion coefficients than implanted dopants. Incorporating GaAs and SUPREM 3.5's models and parameters into SUPREM-IV is only the first step. The simulator must be able to take into account effects such as implant damage, substrate stoichiometry, and defects, all of which are important in GaAs processing. More advanced models and parameters must be developed to model such things as time-dependent diffusion, uphill diffusion, and time-dependent activation and other "anomalous" phenomena, most of which are believed to be related to point defects and their interactions with dopants. In addition, 2D effects during implantation, diffusion, etc. must be modeled. It is planned that a version of SUPREM-IV.GS that models many of these effects and phenomena be released in 1994.

Even without those advanced models, a version of SUPREM-IV with GaAs and SUPREM 3.5's 1D parameters will be very useful. For one, it will allow one to use SUPREM-IV's point defect and 2D capabilities to help develop the more advanced models for GaAs. Secondly, there are many cases where simple SUPREM 3.5 type models are quite adequate to model the fabrication process and having 2D capability will be very useful.

The following briefly summarizes what has been added to SUPREM-IV to make this version 9305 of SUPREM-IV.GS. GaAs and eight GaAs dopants have been added to SUPREM-IV. The Pearson-IV implantation parameters have been included. The electron or hole dependent diffusion coefficients have been put in for these dopants. Other parameters for GaAs, such as segregation coefficients, intrinsic carrier concentrations, and defect energy levels, have also been added. In addition, compensation mechanisms for dopant activation and different diffusivities for implanted versus grown-in dopants have been added. (These two capabilities were not in the silicon-only versions of SUPREM-IV and were added especially for GaAs. However, they may be useful for silicon modeling as well.) SUPREM-IV.GS can now simulate implantation, film deposition and etching, and dopant diffusion for GaAs, as well as all the silicon capabilities.

In the following sections, the addition of GaAs and its SUPREM 3.5 process models to SUPREM-IV is discussed in more detail.

Adding SUPREM 3.5's GaAs Models and Parameters to SUPREM-IV

Adding a New Semiconductor

The first task was to add a new semiconductor material to SUPREM-IV, namely GaAs. This was a non-trivial task as SUPREM-IV had been designed with the implicit assumption that only those materials used in silicon device processing would be used. Every file had to be carefully examined for those places that dealt with material specification. The properties of GaAs, in particular the intrinsic carrier concentration as a function of temperature, had to be added. In SUPREM 3.5, we had used an extended expression for this from Blakemore's analysis [1], which included temperature dependencies of the bandgap, the effective density of states at the conduction and valence bands, and the effective masses. In order to fit this into the form used in SUPREM-IV (*i.e.* $n_i = n_{i.0} \times T^{n_{i.Pow}} \times e^{-(n_{i.E}/(kT))}$), the values from the Blakemore expression were fit to this kind of equation for the temperature range of 500˚C to 1000˚C. From this, we determined that $n_{i.0}$ = 4.0×10^{16}, $n_{i.Pow}$ = 1.5, and $n_{i.E}$ = 0.905.

Adding New Dopants

The eight dopants in SUPREM 3.5 were added to SUPREM-IV. These are the four n-type, Si, Se, Ge, and Sn, and the four p-type, Be, Mg, Zn, and C. As with the addition of GaAs as a new material, the addition of new impurities required careful examination of each routine, but since the original authors have had to add additional impurities this task was more straightforward. One consequence of its bias towards silicon processing was the assumption that materials and impurities could be unambiguously specified on the same input statement. So as not to confuse Si the dopant in GaAs with Si the semiconductor, the designation isilicon (for “impurity silicon”) is used for the former in command line designations. In SUPREM 3.5, we include an algorithm based on energy/mass scaling to simulate the implantation of any element in the periodic table in GaAs. To do this, all elements of the periodic table must be included in the impurity or dopant set in SUPREM-IV. We may want to add this to SUPREM-IV eventually, but in this first version it is not done.

GaAs Implantation Models and Parameters

SUPREM 3.5 models implantation by using Pearson IV distributions, just as in SUPREM-IV. The four Pearson IV moments (projected range, straggle, skewness, and kurtosis) were experimentally determined for four dopants in GaAs (Si, Se, Be, and Mg) as functions of energy. It was previously determined that theoretical calculations of the as-implanted profiles, such as by TRIM (a monte-carlo technique) or BTE (a Boltzmann transport equation method) do not adequately model implantation of ions in crystalline solids such as Si or GaAs. These calculations assume an amorphous substrate and hence do not predict the tails in the profiles due to ion channeling, nor do they estimate the projected range adequately. Based on the extensive experimentation and modeling of four of the dopants in

GaAs, tables of Pearson-IV moments versus implant energy were constructed. Then based on empirically determined mass/energy scaling relationships, tables for the other four dopants, as well as for all elements in GaAs, were built. (These have been shown to very accurately model implantation in GaAs). SUPREM-IV also models implantation using Pearson-IV moment/energy tables. The tables for the eight dopants in GaAs have been added to SUPREM-IV’s implantation data files. In addition, the experimentally determined Gaussian parameters for implanting dopants in or through capping material (silicon nitride and oxide) have been copied from SUPREM 3.5 into SUPREM-IV.

SUPREM-IV also includes a fifth moment, the lateral straggle, in order to model the implantation profile in two dimensions. This moment was not determined for SUPREM 3.5, but in order to put some first-order numbers in for GaAs dopants, an algorithm based on TRIM results was developed. Even though TRIM does not predict the absolute values very well for GaAs, it does adequately predict trends and relationships. From tables of Pearson IV moments produced by TRIM for the dopants in GaAs, it is seen that ratio of the lateral straggle to normal straggle is fairly constant with implant energy for each ion, and that this ratio scales with the log of the ion mass. Assuming that this ratio is valid (as opposed to the actual values of the moments), we calculated this ratio for each ion and applied it to the experimentally determined normal straggles. From this, we determined that for each energy and dopant, the ratio of the lateral straggle (the fifth Pearson IV moment) to the normal straggle (the second Pearson IV moment) equals 1.83 - 0.525log(ion mass). The lateral straggles for each dopant and energy were thus calculated and added to SUPREM-IV for the eight dopants in GaAs.

SUPREM 3.5 includes implant profile dependencies on dose and cap type and thickness. In general, we had experimentally determined that the tails decreased: (1) as the implant dose increased and (2) when the implants were done through oxide or nitride caps (in addition to changing the projected range). Both these phenomena are presumably due to decreasing the ion channeling, either by randomizing the ion trajectory or by decreasing the channels. This is modeled in SUPREM 3.5 by adding an algorithm which modifies the second Pearson-IV moment as a function of dose or cap. SUPREM 3.5 is able to do this because the complete processing history is part of the structure information file. There is no capability for this in SUPREM-IV, and will have to be added later.

SUPREM-IV.GS now allows one to specify the implant angle. Experimental determination of implant parameters for GaAs showed that the channeling was minimized for implant angles of greater than or equal to 10˚ to the normal, and the default parameters correspond to that condition. For now, specifying and angle of less than 10˚ will not increase the channeling in the simulations (that will be added later), but “shadowing” is now simulated for non-zero degree implants.

GaAs Diffusion Models and Parameters

SUPREM-IV models dopant diffusion according to the following equation:

$$\frac{\partial C_A}{\partial t} = \nabla \left[D_v C_A \frac{C_V}{C_V^*} \nabla \left(C_A \frac{C_V}{C_V^*} \frac{n}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \left(C_A \frac{C_I}{C_I^*} \frac{n}{n_i} \right) \right] \quad \textbf{(EQ 1)}$$

where C is concentration, D is diffusivity, V is vacancy, I is interstitial, and the “star” denotes equilibrium values. For p-type dopants, the electron concentration, n, is replaced by the hole concentration, p. It is assumed that the diffusion occurs via interstitials or vacancies, or a combination of both. The full expressions for D_V and D_I, which are dependent on the fermi-level and on the fraction of interstitial-assisted diffusion (F_i), as compared to that which diffuses via vacancies, are given in Table 1. We have

TABLE 1 Equation used in SUPREM-IV to model diffusion, based on interstitial and/or vacancy mechanisms. Also given is the expression for a generic dopant used in GaAs.

The diffusion equation solved by SUPREM-IV

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

for n-type dopants. For p-type dopants, replace the $\frac{n}{n_i}$ by $\frac{p}{n_i}$.

Generic Dopant:

$$D_I = F_i \times \left[\begin{array}{c} D_x + D_m \left(\frac{n}{n_i}\right) + D_p \left(\frac{p}{n_i}\right) + D_{MM} \left(\frac{n}{n_i}\right)^2 + D_{PP} \left(\frac{p}{n_i}\right)^2 \\ + D_{MMM} \left(\frac{n}{n_i}\right)^3 + D_{PPP} \left(\frac{p}{n_i}\right)^3 \end{array} \right]$$

$$D_V = (1 - F_i) \times \left[\begin{array}{c} D_x + D_m \left(\frac{n}{n_i}\right) + D_p \left(\frac{p}{n_i}\right) + D_{MM} \left(\frac{n}{n_i}\right)^2 + D_{PP} \left(\frac{p}{n_i}\right)^2 \\ + D_{MMM} \left(\frac{n}{n_i}\right)^3 + D_{PPP} \left(\frac{p}{n_i}\right)^3 \end{array} \right]$$

Where F_i = fraction of Interstitialcy or interstitial-assisted diffusion and $(1\text{-}F_i)$ = fraction of vacancy assisted diffusion.

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

previously determined that the four n-type dopants in GaAs diffuse via a Ga vacancy mechanism ($F_i = 0$) and that the p-type dopants diffuse via a

Ga interstitial mechanism ($F_i = 1$) [1-4]. We have also shown that the kick-out (or substitutional-interstitial diffusion mechanism), traditionally used to model p-type diffusion in GaAs, results in virtually identical governing diffusion equations as used for interstitial diffusers in SUPREM-IV for Si [4]. We have demonstrated that SUPREM-IV's equation can be used to successfully model the anomalous diffusion behavior of Mg in GaAs when excess defects produced by implant damage are included. For these reasons, it has been a relatively simple task to incorporate SUPREM 3.5's simple non-defect models and parameters into the SUPREM-IV framework. Basically, the expressions in SUPREM 3.5 were made to be the D_i and D_V terms for p-type dopants and n-type dopants, respectively [See the 1990 and 1992 Stanford Technical Reports, "Process Simulators for Silicon VLSI and High Speed GaAs Devices," by Plummer, *et al.* for more discussion on these values]. Two simple modifications to SUPREM 3.5's expressions were made. One was to include a n_i term in the expressions for the p-type dopants to fit the SUPREM-IV framework. In SUPREM 3.5, the diffusivity expressions were derived based on the kick-out methodology and were not normalized to the intrinsic carrier concentration. This modification involved changing the pre-exponential term and the exponential term (since n_i is a function of temperature) of the diffusivities. The other modification was due to the fact that the original n-type diffusivities were derived without an electric field effect. Since an electric field effect is built into SUPREM-IV's diffusion equation, the numbers had to be re-derived, since the effect can increase the total diffusion by up to a factor of two. All eight dopants were tested in SUPREM-IV (without any excess interstitials or vacancies) to ensure that their diffusion behavior for inert conditions is the same as SUPREM 3.5 predicts.

SUPREM 3.5 has two sets of diffusivities for Mg in GaAs, one for temperatures below 825˚C and one for temperatures above 825˚C. For now we use only the higher temperature values. Also, the diffusivity for grown-in Zn has been modified due to recent re-analysis of the literature data.

The equations and parameters for the eight dopants in regards to diffusion in GaAs are shown in Table 2 for the n-type dopants and in Table 3 for the p-type dopants.

SUPREM 3.5 had two sets of diffusivities for dopants in GaAs: one for dopants that were implanted, and one for dopants that were incorporated during growth, or “grown-in.” For p-type dopants, the implanted values are generally higher than the grown-in; for n-types the implanted values are generally lower than the grown-in. These differences are believed to be due to different point defect conditions caused by the introduction of the dopants and the subsequent diffusion. For this first version of SUPREM-IV.GS, we have used the SUPREM 3.5 convention of using different values of D for the two cases (in the future, physically correct models which take into account the different point defect conditions will be used). SUPREM-IV.GS keeps track if a given dopant has been implanted or not, and uses the appropriate value. (It assumes that if the dopant is not implanted, it is grown-in. Gas source diffusions would therefore use the grown-in values). If you want to use the implanted value for cases where the dopant is not implanted in the input file (*e.g.* when a solid-source diffusion is done, or when a SIMS file is used as the initial dopant profile), then one should do a very low dose implant (*i.e*. below the background level) of that dopant so the program thinks that all of it has been implanted.

The SUPREM 3.5 default numbers for dopant diffusivity in cap material (silicon oxide or nitride) used in GaAs technology have been added to SUPREM-IV, as have the segregation and transport values across interfaces. Rather arbitrary default numbers for interstitial and vacancy diffusivity, segregation, and transport parameters have also been added. These will be updated when more reliable numbers are determined.

We have also added a generic dopant to SUPREM-IV.GS, whose diffusivity expression is shown in Table 1. This includes many n and p terms in the expression and can be used to test other dopants and diffusivities for

TABLE 2 Expressions for diffusivities of n-type dopants in GaAs used in SUPREM-IV.GS.

	IMPLANTED	**GROWN-IN**
Silicon (Si)	$D_V = (1-F_i)D_X$ $D_I = F_i D_X$ $F_i = 0$ $D_{X.0} = 1.8\times10^{-9}\mathrm{cm}^2\mathrm{sec}^{-1}$ $D_{X.E} = 1.6e\mathrm{V}$	$D_V = (1-F_i)D_{MM}\left(\frac{n}{n_i}\right)^2$ $D_I = F_i D_{MM}\left(\frac{n}{n_i}\right)^2$ $F_i = 0$ $D_{MM.0} = 33\mathrm{cm}^2\mathrm{sec}^{-1}$ $D_{MM.\mathrm{E}} = 3.9e\mathrm{V}$
Selenium (Se)	$D_V = (1-F_i)D_X$ $D_I = F_i D_X$ $F_i = 0$ $D_{X.0} = 1.8\times10^{-8}\mathrm{cm}^2\mathrm{sec}^{-1}$ $D_{X.E} = 1.6e\mathrm{V}$	$D_V = (1-F_i)D_{MM}\left(\frac{n}{n_i}\right)^2$ $D_I = F_i D_{MM}\left(\frac{n}{n_i}\right)^2$ $F_i = 0$ $D_{MM.0} = 1.2\mathrm{cm}^2\mathrm{sec}^{-1}$ $D_{MM.E} = 3.6e\mathrm{V}$
Tin (Sn)	$D_V = (1-F_i)D_{MM}\left(\frac{n}{n_i}\right)^2$ $D_I = F_i D_{MM}\left(\frac{n}{n_i}\right)^2$ $F_i = 0$ $D_{MM.0} = 9.6\times10^{1}\mathrm{cm}^2\mathrm{sec}^{-1}$ $D_{MM.E} = 3.9e\mathrm{V}$	same as implanted
Germanium (Ge)	$D_V = (1-F_i)D_{MM}\left(\frac{n}{n_i}\right)^2$ $D_I = F_i D_{MM}\left(\frac{n}{n_i}\right)^2$ $F_i = 0$ $D_{MM.0} = 1.1\times10^{-3}\mathrm{cm}^2\mathrm{sec}^{-1}$ $D_{MM.E} = 2.9e\mathrm{V}$	same as implanted

TABLE 3

Expressions for diffusivities of p-type dopants in GaAs used in SUPREM-IV.GS

	IMPLANTED	GROWN-IN
Beryllium (Be)	$D_V = (1 - F_i) D_P \left(\frac{p}{n_i}\right)$ $D_I = F_i D_P \left(\frac{p}{n_i}\right)$ $F_i = 1$ $D_{P.0} = 2.1 \times 10^{-6} \mathrm{cm}^2 \mathrm{sec}^{-1}$ $D_{X.E} = 1.74 eV$	$D_V = (1 - F_i) D_P \left(\frac{p}{n_i}\right)$ $D_I = F_i D_P \left(\frac{p}{n_i}\right)$ $F_i = 1$ $D_{PP.0} = 2.1 \times 10^{-8} \mathrm{cm}^2 \mathrm{sec}^{-1}$ $D_{PP.E} = 1.74 eV$
Magnesium (Mg)	$D_V = (1 - F_i) D_P \left(\frac{p}{n_i}\right)$ $D_I = F_i D_P \left(\frac{p}{n_i}\right)$ $F_i = 1$ $D_{P.0} = 6.1 \times 10^{-2} \mathrm{cm}^2 \mathrm{sec}^{-1}$ $D_{P.E} = 2.8 eV$	$D_V = (1 - F_i) D_P \left(\frac{p}{n_i}\right)$ $D_I = F_i D_P \left(\frac{p}{n_i}\right)$ $F_i = 1$ $D_{P.0} = 6.8 \times 10^{-3} \mathrm{cm}^2 \mathrm{sec}^{-1}$ $D_{P.E} = 2.8 eV$
Zinc (Zn)	$D_V = (1 - F_i) D_{PP} \left(\frac{p}{n_i}\right)^2$ $D_I = F_i D_{PP} \left(\frac{p}{n_i}\right)^2$ $F_i = 1$ $D_{PP.0} = 1.2 \times 10^{2} \mathrm{cm}^2 \mathrm{sec}^{-1}$ $D_{PP.E} = 3.9 eV$	$D_V = (1 - F_i) D_{PP} \left(\frac{p}{n_i}\right)^2$ $D_I = F_i D_{PP} \left(\frac{p}{n_i}\right)^2$ $F_i = 1$ $D_{PP.0} = 15.6 \mathrm{cm}^1 \mathrm{sec}^{-1}$ $D_{PP.E} = 3.9 eV$
Carbon (C)	$D_V = (1 - F_i) D_X$ $D_I = F_i D_X$ $F_i = 1$ $D_{X.0} = 1.0 \times 10^{-16} \mathrm{cm}^2 \mathrm{sec}^{-1}$ $D_{X.E} = 0.0 eV$	same as implanted

subsequent model development. (The default type of the generic dopant is an acceptor.)

Since some dopants in GaAs are amphoteric and can be either donors or acceptors under certain conditions, we have added the capability of changing the type of the impurity from its default to either donor or acceptor. This is an impurity specification option.

GaAs Solubility and Activation Models

In SUPREM-IV one can specify the solubility of a dopant in a semiconductor as a function of temperature. Above that concentration, the dopant is electrically inactive and immobile (the diffusivity is effectively zero). Below that concentration, the dopant is electrically active and mobile. In GaAs, for relatively low concentration levels of n-type dopants, not all the dopants appear to be electrically active, yet they are all mobile. One way to model this is to assume that below the solubility level, some dopants may be inactive but all are mobile. A better way of explaining this, as is done by Walukiewicz [5] and others recently, is that all or most are mobile and electrically active, but there is some compensation mechanism (due to the production of negatively charged Ga vacancies, for example) which reduces the net donor concentration. In SUPREM 3.5 the first method was used, and empirical formulae were determined which calculated the active fraction of each n-type dopant as a function of concentration and temperature. (Arsenic clustering in silicon was modeled in a somewhat similar manner in SUPREM-IV.)

We have decided to model dopant activation for GaAs in SUPREM-IV using the second method. This is believed to be the more physically correct model for the phenomena and will result in the compensation ratio, or fraction inactive, to be based on the total doping, which is what is actually observed, and not on each dopant individually, which is what is done in SUPREM 3.5.

Until a defect-based compensation model is incorporated into SUPREM-IV, a semi-empirical model will be used. The original algorithm in SUPREM 3.5, in addition to having the shortcomings mentioned above, was based on limited data in the literature and resulted in errors at high concentrations and temperatures. An expression for the compensation effect has been determined by fitting recent modeling results of Vanasupa [6] with an analytical equation, which is a function of net doping concentration, $(N_D - N_A)$, and temperature. (Vanasupa’s numbers are based upon both experimental data and a specific compensation mechanism, compensation by amphoteric pairs in her case, but the numbers would apply to any compensation mechanism.) The equation is:

$$(N_D - N_A)_{\text{effective}} = \frac{(2.20 - 0.00117T)\,(N_D - N_A)}{1 + (N_D - N_A)/(1.25\times 10^{18})} \qquad \textbf{(EQ 2)}$$

where T is in degrees Kelvin. The compensation factor will be applied to the net doping concentration calculation to get an effective net doping concentration.

The expression for compensation is based on the equilibrium activation numbers of Vanasupa [6]. She also determined that there is a time dependent, kinetic part to the activation process, important for short time activation anneals. This will also have to be included in future models for dopant activation in GaAs.

The solubility values for dopants in GaAs are not well known. Until good numbers are obtained, default values of $1\times 10^{19}\,\text{cm}^3$ will be used for all dopants.

GaAs Defect Parameters

SUPREM 3.5 does not consider point defects in its implantation, diffusion, or activation models. As discussed above, one of the prime motives for incorporating GaAs into SUPREM-IV is to make use of SUPREM-IV's point defect capabilities, especially in regards to the effect of point defects on dopant diffusion and activation. For the initial version of SUPREM-IV for GaAs, we want to include some default numbers for the defect parameters so we can start our development of future point-defect based models for diffusion and activation.

SUPREM-IV only considers one type of interstitial and one type of vacancy. In GaAs there are two types of each: Ga and As. Eventually we will have to add both types to SUPREM-IV, but for now we will only assume one type. (We have shown that for modeling diffusion of both n-type and p-type dopants in GaAs, it is not a bad assumption to consider only the Ga sublattice (see reference [4] for example), so this limitation may not hinder us too much.)

The total number of defects and the distribution of each charge type is calculated by SUPREM-IV as a function of temperature and fermi-level. Using the energy levels of the variously charged vacancies and interstitials calculated by Baraff and Schluter [7] and by Jansen *et al*. [8], the relative probability of each charge state was calculated using the Shockley-Last equations [9]. These numbers are given in Table 4. Since triply charged interstitials and vacancies are important in GaAs, and SUPREM-IV previously only included neutral through doubly charged defects, new parameters had to be added.

Rather arbitrary default numbers for defect recombination (both surface recombination and bulk I/V recombination) have been put in for Ga interstitials and vacancies

.

TABLE 4 Parameter values for Ga defect distributions used in SUPREM-IV

$$C_I^* = C_{\text{star}} \frac{\text{neu} + \text{neg}\left(\frac{n}{n_i}\right) + \text{dneg}\left(\frac{n}{n_i}\right)^2 + \text{tneg}\left(\frac{n}{n_i}\right)^3 + \text{pos}\left(\frac{p}{n_i}\right) + \text{dpos}\left(\frac{p}{n_i}\right)^2 + \text{tpos}\left(\frac{p}{n_i}\right)^3}{\text{neu} + \text{neg} + \text{dneg} + \text{tneg} + \text{pos} + \text{dpos} + \text{tpos}}$$

$$\text{neu} = \text{neu.0} \times \exp\left(\text{neu.E} / (kT)\right)$$

Gallium interstitials:		Gallium vacancies:	
neu.0 = 1.0	neu.E = 0.0	neu.0 = 1.0	neu.E = 0.0
neg.0 = 0.0	neg.E = 0.0	neg.0 = 0.05305	neg.E = -0.588
pos.0 = 0.053	pos.E = -0.788	pos.0 = 0.0	pos.E = 0.0
dneg.0 = 0.0	dneg.E = 0.0	dneg.0 = 0.00281	dneg.E = -0.856
dpos.0 = 0.0	dpos.E = 0.0	dpos.0 = 0.0	dpos.E = 0.0
tneg.0 = 0.0	tneg.E = 0.0	tneg.0 = 1.489×10^{-4}	tneg.E = -0.899
tpos.0 = 0.0	tpos.E = 0.0	tpos.0 = 0.0	tpos.E = 0.0

References

1. J.S. Blakemore, "Intrinsic density n_i(T) in GaAs: Deduced from band gap and effective mass parameters and derived independently from Cr acceptor capture and emission coefficients," *J. Appl. Phys,* **53**, 520 (1982).
2. M.D. Deal, S.E. Hansen, and T.W. Sigmon, "SUPREM 3.5 - Process Modeling of GaAs Integrated Circuit Technology," *IEEE Trans. CAD.*, **9**, 939 (1989).
3. K.H. Lee, D.A. Stevenson, and M.D. Deal, "Diffusion Kinetics of Si in GaAs and Related Defect Chemistry," *J. Appl. Phys.*, **68**, 4008 (1990).
4. H.G. Robinson, M.D. Deal, P.B. Griffin. G. Amaratunga, D.A. Stevenson, and J.D. Plummer, "Modeling Uphill Diffusion of Mg Implants in GaAs Using SUPREM-IV," *J. Appl. Phys.*, **71**, 2615 (1992).
5. W. Walukiewicz, "Electron Scattering By Native Defects in Uniformly and Modulation Doped Semiconductor Structures," *Mat. Res. Soc. Proc.,* **163**, 845 (1990).

6. L.S. Vanasupa, M.D. Deal, and J.D. Plummer, “Modeling Activation of Implanted Si in GaAs,” *J. Electrochem. Soc.*, **138**, 2134 (1991).
7. G.A. Baraff and M. Schluter, “Electronic Structure, Total Energies, and Abundances of the Elementary Point Defects in GaAs,” *Phys. Rev. Lett.*, **55**, 1327 (1985).
8. R.W. Jansen, D.S. Wolde-Kidane, and O.F. Sankey, “Energetics and Deep Levels of Interstitial Defects in the Compound Semiconductors GaAs, AlAs, ZnSe, and ZnTe,” *J. Appl. Phys.*, **64**, 2415 (1988).
9. W. Shockley and J.T. Last, “Statistics of the Charge Distribution for a Localized Flaw in a Semiconductor,” *Phys. Rev.*, **107**, 392 (1957).

User’s Reference Manual

INTRODUCTION

This manual describes SUPREM-IV.GS, a version of the process simulator SUPREM-IV extended to both GaAs and Silicon process simulation. This manual is not intended as a tutorial guide on how to use SUPREM-IV.GS, instead it is more of a reference to the commands and capabilities. The manual is divided into four sections. The first section (this section) provides an introduction and describes the conventions used in the rest of the manual. Next is a section describing the commands that are part of the interactive shell. The third section describes the commands of SUPREM-IV.GS that perform actions. The final section documents the model commands, most of which are used to set parameters. In that section the physics associated with SUPREM-IV.GS models is described and referenced. Following the manual sections are a series of SUPREM-IV.GS examples, which can be used to learn the program. The example input decks are provided on the source code tape. A new user may want to start in that section.

CONVENTIONS

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, *i.e.* 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, which recognize true and false as special strings. 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 numerical valued parameter
param = <string>	a string valued parameter
param	a boolean parameter

Choices are denoted by putting the parameters in parenthesis and separating them by vertical bars. For example, **(par1 | par2)** 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.

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=<string> **]**

Par2 is an optional character string variable.

SEE ALSO

select for how to choose a **z** variable, **printf** for further examples of expressions.

The SUPREM-IV.GS Shell

DESCRIPTION

The shell is the user interface to the program. The shell was designed to be similar to Unix's® **csh**(1) shell 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 Command Summary

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

define
define name wordlist
define name**(** args **)** wordlist

The first form prints all aliases. The second and third forms assign 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, as shown in the third form, a definition can contain arguments which will be mapped when the macro is substituted. name will only be substituted for when it is separated by blank space or shell special characters.

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. name is substituted in the commands using the macro features.

help command

help without any arguments will list all of the available commands. The command “help command_name” will give information on the listed command name. When used in the second style, the **help** command provides a list of parameters and short description of each of the parameters.

set (echo | noexecute | prompt string **)**

This command allows the parameters listed to be changed. **echo** causes the command buffer to be echoed. **noexecute** is 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 removes the macro name from the macro table.

unset (echo | noexecute)

This command turns off the **echo** or **noexecute** modes.

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 **noexecute** flag is set, the command will only check 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 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. There is a limit on the number of open files. It is not suggested that redirection be nested in a loop.

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.

COMMAND DEFINE

Define strings for command line substitution.

SYNOPSIS

define [macro_name macro_body **]**

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 a macro substitution. 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 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 equal 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

The **undef** command.

COMMAND FOR, FOREACH

Command looping facility.

SYNOPSIS

```
[ foreach | for ]  name ( list )
        commands
end
```

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

The **define** command.

COMMAND HELP

Online quick info facility.

SYNOPSIS

help [command_name **]**

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 particular 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**(1) àla the **man** command.

SEE ALSO

The **man** command.

COMMAND MAN

Online help facility for SUPREM-IV.

SYNOPSIS

man [command_name **]**

DESCRIPTION

man examines the command name and attempts to find a manual page for that command. The manual pages are piped through **more**(1) 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** 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

The UNIX command – **more**(1).

COMMAND SET

Set various shell parameters.

SYNOPSIS

set [(echo | noexecute | prompt string **)]**

DESCRIPTION

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

echo

The **echo** parameter causes the command buffer to be echoed to the standard output device. This is not necessary when commands are being entered interactively but useful when reading commands from a file.

noexecute

The **noexecute** flag will put all entered commands into check only mode. If this flag is on, commands will only check for the legality of their arguments and not act upon them.

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

SEE ALSO

The **unset** command.

COMMAND SOURCE

Execute commands from the specified file.

SYNOPSIS

source filename

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. **source** commands 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.

COMMAND UNDEF

Undefine previously defined macros.

SYNOPSIS

undef [macro_name **]**

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**(1) 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

The **define** command.

COMMAND UNSET

Unset various shell parameters.

SYNOPSIS

unset [(echo | noexecute)]

DESCRIPTION

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

echo

Turning off the **echo** option turns off echoing the command buffer. This is usually the desired setting (and the default) when entering commands interactively as opposed to when commands are taken from a file.

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

SEE ALSO

The **set** command.

Commands

DESCRIPTION

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

There are several groups of commands in this section. The first are commands which are used for the input or output of data. The second are simulation commands which are the core of SUPREM-IV.GS. The third are commands which are primarily for 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.

mode

This command allows the user to select a one or two dimensional simulation.

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 method of inputting and outputting 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 Pearson-IV 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

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 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.

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.

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

See the individual command descriptions.

SEE ALSO

See the other command descriptions for more detail.

COMMAND BOUNDARY

Specify a surface type.

SYNOPSIS

```
boundary
    ( reflecting | exposed | backside )
    xlo = <string> ylo = <string>
    xhi = <string> yhi = <string>
```

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 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.

COMMAND CONTOUR

Plot contours in the selected variable on a two-dimensional plot.

SYNOPSIS

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

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 10^{10} to 10^{20}. 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 1. Different line types have different meanings to different graphics devices.

value

This floating parameter expresses the value that the contour line should be plotted at. If boron had been selected, a value of 10^{16} would produce a line of constant boron concentration of 10^{16} cm^{-3}.

symb

This integer parameter indicates that the contour line should be drawn with symbols on the contour line. The integer value indicates different symbols.

print

This boolean parameter instructs the **contour** command to print the line segments coordinates rather than plot them. This is useful for exporting data to other programs and plot software.

label

The command will attempt to place a text string on the contour in an ideal location to label the value.

EXAMPLES

```
contour val=1e10
```

This command will draw a line at a isoconcentration of 1×10^{10}.

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 mildly, annoying. Plot software bugs will show up in this command.

SEE ALSO

The **plot.2d** and **for** commands.

COMMAND CPULOG

Log the cpu usage summary to a file.

SYNOPSIS

cpu [log] [cpufile=<string> **]**

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 the standard output. 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 the standard output.

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 the reporting on the machine the code is running on. This is typically a sixtieth of a second, but may vary. (For example, the Cray X-MP/4 uses 9.5 nanoseconds).

COMMAND DEPOSIT

Deposit a layer.

SYNOPSIS

```
deposit
        ( silicon | oxide | oxynitr | nitride | poly | photores |
                alumin | gaas )
        thickness = <n> [ divisions = <n> ]
        [ none | arsenic | antimony | boron | phosphor |
                beryllium | magnesium | selenium | isilicon |
                tin | germanium | zinc | carbon | generic]
        [ conc=<n> ] [ space=<n> ]
        [ file=<string> ]
```

DESCRIPTION

This statement provides a primitive deposition capability. Material is deposited on the “exposed” surface of the structure, and its upper surface becomes the new exposed surface. This code is now based on the generalized triangulation package originally written by Conor Rafferty as a stand-alone program, **tri**. This tends to be significantly more robust than the grid generation algorithm in earlier versions of SUPREM-IV (*e.g.* 8912). There have also been several enhancements in the handling of the offset line. It is now possible, for instance, to fill a trench completely.

silicon | oxide | oxynitr | nitride | poly | photores | alumin | gaas

The material to be deposited.

thickness

The thickness in microns.

divisions

The number of vertical grid spacings in the layer, default 1.

none | arsenic | antimony | boron | phosphor | beryllium | magnesium | selenium | isilicon | tin | germanium | zinc | carbon | generic

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.

file

This string parameter specifies the name of a file that contains a string that contains the coordinates associated with the deposit surface. This file could be generated using a topography simulator such as SAMPLE.

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.

SEE ALSO

The **etch** statement.

COMMAND 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 |
                beryllium | magnesium | selenium | isilicon |
                tin | germanium | zinc | carbon | generic ) ]
        [ gas.conc=<n> ] [ pressure=<n> ]
        [ gold.surf ] [ continue ]
        [ movie=<string> ] [ dump=<n> ]
```

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, beryllium, magnesium, selenium, silicon, tin, germanium, zinc, carbon, generic

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 preepositions, in atoms per cubic centimeter.

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 command, and do not initialize the defects.

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 n’th 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˚C, 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 at 1000˚C in a dry oxygen ambient.

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

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

```
diffuse time=30 temp=1100
        movie="select 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˚C diffusion. Before each time step, the total net active doping concentration is chosen as the plot **z** 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 (*i.e.* the junction), 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

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **generic**, **germanium**, **interst**, **magnesium**, **material**, **method**, **oxide**, **phosphorus**, **selenium**, **silicon**, **tin**, **trap**, **vacancy**, and **zinc** commands.

COMMAND ECHO

A string printer and desk calculator.

SYNOPSIS

echo [string **]**

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)

```
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

The UNIX command – **more**(1).

COMMAND ETCH

Etch a layer.

SYNOPSIS

```
etch
     [ silicon | oxide | oxynitr | nitride | poly | photores |
          alumin | gaas ]
     [ 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> ]
     [ file=<string> ]
```

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 different 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, gaas

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 trapezoidal 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, y

These parameters allow the user to specify point in the **start** / **continue** / **done** mode of etch region specification.

p1.x, p1.y, p2.x, p2.y

These parameters allow the specification of a line for **left** / **right** etching.

thick

This parameter allows specification of a thickness for the dry etch type.

file

This string parameter specifies the name of a file that contains a string that contains the coordinates associated with the etched surface. This file could be generated using a topography simulator such as SAMPLE.

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 0.1µm and this line will be the new surface.

BUGS

Uses no physical model.

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. Contrary to what some commercial vendors maintain, the mesh produced is the best possible one to solve PDE's given the constraints. However, It is usually better to place a line at the etch locations.

The code currently does distinguish between surface and subsurface materials. "Etch silicon left p1.x=0.2" will etch the silicon to the left of 0.2µm, including the substrate. Use the **start** / **cont** / **end** sequence to work around this.

COMMAND IMPLANT

Perform ion implantation.

SYNOPSIS

```
implant
        ( antimony | arsenic | boron | bf2 | cesium | phosphorus |
                beryllium | magnesium | selenium | isilicon | tin |
                germanium | zinc | carbon | generic )
        [ gauss | pearson ]
        dose=<n> energy=<n>
        [ damage ] [ max.damage=<n> ]
        [ range=<n> ] [ std.dev=<n> ]
        [ gamma=<n> ] [ kurtosis=<n> ]
        [ angle=<n> ]
```

DESCRIPTION

This statement is used to simulate ion implantation. There are two different types of analytic models available. The first implements a simple Gaussian distribution. 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, beryllium | magnesium | selenium | isilicon | tin | germanium | zinc | carbon

These parameters specify the impurity to be implanted. Default data for the range statistics exists only for antimony, arsenic, boron, BF_2, phosphorus, beryllium, magnesium, selenium, isilicon, tin, germanium, zinc, and carbon.

gauss, pearson

These parameters specify which type of distribution is being used. A Pearson-IV distribution is the default.

dose

This parameter allows the user to specify the dose of the implant. The units are in atoms/cm^2.

energy

This parameter specifies the implant energy in KeV.

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 the crystal that makes sense to calculate. Clearly damage above the crystal density is absurd.

range, std.dev, gamma, kurtosis

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.

angle

This parameter allows the user to specify the angle normal to the substrate that the impurity was implanted at.

EXAMPLES

implant phosph dose=1e14 energy=100 pearson

This statement specifies that a 100KeV implant of phosphorus was done with a dose of $1\times10^{14}\text{cm}^{-2}$. The Pearson-IV 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 $1\times10^{14}\text{cm}^{-2}$. 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 100KeV implant of phosphorus was done with a dose of $1\times10^{14}\text{cm}^{-2}$. The Pearson-IV model is to be used for the distribution function. The damage from this implant is calculated and stored as point defects.

BUGS

Very high angle implants may have non-physical profiles near the device edges.

Occasionally problems are encountered on undercut surfaces and other reentrant shapes. The algorithm really wasn't intended for these types of shapes.

REFERENCES

1. G. Hobler and S. Selberherr, “Two-Dimensional Modeling of Ion Implantation Induced Point Defects,” *IEEE Trans. on CAD*, **7**(2), p. 174, 1988.

COMMAND INITIALIZE

Setup grid, background doping levels.

SYNOPSIS

```
initialize
        [ infile=<string> ]
        [ ( antimony | arsenic | boron | phosphorus | gold |
                gallium | beryllium | magnesium | selenium |
                isilicon | tin | germanium | zinc | carbon | generic ) ]
        [ conc=<n> ]
        [ orientation=<n> ]
        [ line.data ] [ scale ] [ flip.y ]
```

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, beryllium, boron, carbon, generic, germanium, gold, magnesium, phosphorus, selenium, isilicon, tin, zinc

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

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.

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** or other grid generation programs.

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 1×10^{15} cm^{-2}.

BUGS

There are no known bugs.

SEE ALSO

The **bound**, **line**, **region**, **structure** statements.

COMMAND LABEL

Put labels on a plot.

SYNOPSIS

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

DESCRIPTION

The **label** command allows the user to put random pieces of text anywhere on a SUPREM-IV plot.

x, y

Where to put the label, in microns.

label

What string to print.

symb, line.type

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

There are no known bugs.

SEE ALSO

The **plot.1d**, **plot.2d**, and **for** statements.

COMMAND LINE

Specify a mesh line location.

SYNOPSIS

line (x | y | z) location = <n> **[spacing =** <n>**] [tag =** <string>**]**

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**.

x | y | z

A mesh line is either horizontal, vertical, or into the screen. 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. The **z** coordinate is supported, somewhat. Rectangular grid generation in three dimensions is the only three dimensional part of the program. It is anticipated that the next release will be fully one, two, and three dimensional.

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 tagged with a label 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 three user-specified **x** lines and two 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, and 2.0µm. Around the center, the spacing is 0.12, approximately what was requested. At the edge, the spacing is 0.42µm, 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 0.9µm and one interval of 0.1µm on each side. In this example, specifying a spacing of 1µm 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 re-grid has been broken since oxidation introduced moving boundaries. Therefore the initial mesh specification is quite important to the success of the simulation. Use “best guess” numbers for the profiles, or coarse-grid simulations, to get a feel for the required spacings in the initial and final profile.

COMMAND METHOD

Select numerical methods and models for diffusion and oxidation.

SYNOPSIS

```
method
    [ ( vacancies | interstitial | arsenic | phosphorus |
        antimony | boron | gallium | oxidant | velocity |
        traps | gold | psi | cesium | beryllium | magnesium |
        selenium | isilicon | tin | germanium | zinc | carbon |
        generic ) ]
    [ 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 | 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=<n> ] [ oxide.late=<n> ] [ oxide.rel=<n> ]
    [ gloop.emin=<n> ] [ gloop.emax=<n> ] [ gloop.imin=<n> ]
```

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, gallium, oxidant, velocity, traps, gold, psi, beryllium, magnesium, selenium, isilicon, tin, germanium, zinc, carbon, generic

These options allow the user to specify a single impurity. The error bound parameters are specific for each impurity. If error bounds are listed, an 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 10^{9}. For defects, the absolute error defaults to 10^{5}. For the potential, the error defaults to 10^{-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

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. **min.freq** is a parameter which controls the frequency of **min fill** reordering. 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 by Elman [4].

back

This integer parameter specifies the number of back vectors that can be used in the cg outer iteration. The default is 3, and the maximum that can be used is 5. 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 re-factored. The **time** parameter specifies that the matrix should be re-factored 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 re-factored whenever the error in that block is decreasing. The **newton** option will force the re-factorization at every newton step. The default is time.

diag, 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 **diag** option is the default, although under certain conditions (one dimensional) **full.fac** may 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** statement). 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 *vs.* speed trade-off.

grid.oxide

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 indiscriminately, since the silicon grid is generally much larger than the oxide grid, and the stress calculation correspondingly more expensive.

oxide.gdt

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

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, oxide.late, oxide.rel

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 10^{-6} (that is, a 10^{-4} percentage error in the velocities). It can be increased for a faster solution, but a value greater than about 10^{-3} will probably give meaningless results.

gloop.emin, gloop.emax, gloop.imin

This group of parameters controls loop detection during grid manipulation. The loop detection checks for intrusions and extrusions in the boundary. The intrusion-fixing algorithm is triggered by angles greater than **gloop.imin**. The default value is **gloop.imin** = 170˚. A larger value means that more extreme intrusions can develop and increases the possibility of a tangled grid. A smaller value leads to earlier intrusion-fixing; too small a value will lead to inaccuracy due to premature intervention. Similar concerns apply to the other parameters. The values are a compromise between safety and accuracy. The extrusion-fixing algorithm is always triggered by angles greater than **gloop.emax**. Again, the default is 170˚. In addition, it be triggered by lesser extrusions, anything greater than **gloop.emin**, if the extrusion is a single-triangle glitch in the boundary. The default value is **gloop.emin** = 120˚. None of these parameters should be less than 90˚, because then the rectangular edges of the simulation space would be smoothed out!

EXAMPLES

```
method arsenic rel.err=0.01 abs.err=1.0e9
```

This command indicates that the arsenic equation should be solved with a relative error of 1% and concentrations below 1×10^9 can be ignored.

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

This command 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 Trans. on CAD*, **7**(2), p. 181, 1988.
2. H.R. Yeager and R.W. Dutton, “An Approach to Solving Multi-Particle Diffusion Exhibiting Nonlinear Stiff Coupling,” *IEEE Trans. on Elec. Dev.*, **32**(10), p. 1964-1976, 1985.
3. R.E. Bank, W.M. Coughran, W. Fichtner, E.H. Grosse, D.J. Rose and R.K. Smith, “Transient Simulation of Silicon Devices and Circuits,” *IEEE Trans. on Electron Devices*, **ED-32**, p. 1992, 1985.
4. H.C. Elman, Preconditioned Conjugate Gradient Methods for Nonsymmetric Systems of Linear Equations, Yale University Research Report, 1981.

COMMAND MODE

Set the dimensionality of the simulator.

SYNOPSIS

mode [one.dim | two.dim]

DESCRIPTION

The **mode** statement determines the dimensionality of the simulator. The internal data structure is changed slightly to support true one dimensional mode operation when one dimension is selected. Two dimensions is the default, and is the original operation mode.

one.dim, two.dim

These parameters set the simulator dimensionality.

BUGS

None.

COMMAND OPTION

option – Set options.

SYNOPSIS

option [quiet | normal | chat | barf]

DESCRIPTION

The **option** statement used to be considerably more complicated. The plot software now comes configured for X workstations. This command specifies only the verbosity of the simulator.

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.

BUGS

None – but the stuff printed isn't always what is needed.

COMMAND PAUSE

Wait and execute command.

SYNOPSIS

pause

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 executed:
```

and will wait for the user to respond.

BUGS

No known bugs.

COMMAND PLOT.1D

Plot a one dimensional cross section.

SYNOPSIS

```
plot.1d
        [ (x.value=<n> | y.value=<n>) ]
        [ exposed | backside | reflect | silicon | oxide |
                nitride | poly | oxynitride | aluminum |
                photoresist | gaas | gas ]
        [ /exposed | /backside | /reflect | /silicon | /oxide |
                /nitride | /poly | /oxynitride | /aluminum |
                /photoresist | /gaas | /gas ]
        [ boundary ] [ axis ] [ clear ]
        [ symb=<n> ]
        [ x.max=<n> ] [ x.min=<n> ] [ y.max=<n> ] [ y.min=<n> ]
        [ arclength ]
```

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 axes. 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

These parameters specify that the cross section is to be done at a constant value as specified by the parameter. This parameter value 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. In one dimensional mode these parameters do not need to be 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 an initial "/", and material 2, named with a leading "/". For example, "plot.1d oxide /silicon" shows the values in the oxide at the oxide/silicon interface. Thus "plot.1d oxide /silicon" will usually show something different from "plot.1d 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 the axes should be drawn. If this parameter is set false and clear is false, the plot will be drawn on a top of the previous axes. It defaults true.

clear

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

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. **symb** = 1 and higher will add one of the predefined symbols to the plot.

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 10^{16} and 10^{10}.

arclength

If this parameter is set, the ordinate is 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. This parameter only has meaning when used with a plot along an interface.

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 = 1.0μm. 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 = 2.0μm on the previous set of axes.

```
plot.1d sil /oxi
```

Plots the selected variable along the silicon interface facing oxide.

BUGS

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

The **select** statement.

COMMAND PLOT.2D

Plot a two dimensional xy picture.

SYNOPSIS

```
plot.2d
        [ x.max=<n> ] [ x.min=<n> ] [ y.min=<n> ] [ y.max=<n> ]
        [ clear ] [ fill ] [ boundary ] [ grid ] [ axis ]
        [ vornoi ] [ diamonds ]
        [ stress ] [ vmax=<n> ] [ vleng=<n> ]
        [ flow ]
```

DESCRIPTION

The **plot.2d** statement allows the user to prepare 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 xy space of the device, label it, draw material boundaries, and draw stress and flow vectors. If in one dimensional mode, it will return and print an error message.

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.

clear

The **clear** parameter specifies whether the graphics screen should be cleared before 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 axes around the plot. If **axis** is true, the graph will be drawn with labeled axes. This parameter defaults true.

line.grid

The **line.grid** parameter controls the line type or color that the axes 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 tessellation to be drawn.

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. 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/cm^2), 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 axes.

plot.2d bound x.min=2 x.max=5 clear=false

This command will draw the material interfaces and x axis between 2.0 and 5.0μm 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 5.0μm. The boundaries will be drawn with line type 2, no axes, and will draw the grid points.

BUGS

None

SEE ALSO

The **contour** and **select** statements.

COMMAND 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=<string> ]
```

DESCRIPTION

The **print.1d** command 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

These parameters specify that the cross section is to be done at a constant value as specified by the value of the parameter. **x.value** specifies a vertical cross section of the device, and **y.value** a horizontal slice. The parameter value is in microns. Only one of **x.value** or **y.value** can be specified for a given device. In one dimensional mode, it is not necessary to specify a position.

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 an initial “/”, and material 2, named with a leading “/”. For example, “print.1d oxide /silicon”. Thus “print.1d oxide /silicon” will usually show something different from “print.1d 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. If the user is unfamiliar with the C programming language, it might be 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 1µm between the top of the mesh and the 3.0µm 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

The **select**, **plot.1d**, and **printf** statements.

COMMAND PRINTF

A string printer and desk calculator.

SYNOPSIS

print [string **]**

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 parenthesis 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

The **select** statement for choosing z variables,
The section on conventions for a description of the real number parser,
The **diffuse** statement for examples of usage in the **movie** parameter.

COMMAND PROFILE

Read a one dimensional doping profile.

SYNOPSIS

```
profile
        infile=<string>
        ( antimony | arsenic | boron | phosphorus | gallium |
                interstitial | vacancy | beryllium | carbon |
                germanium | selenium | isilicon | tin |
                magnesium | zinc | generic )
```

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, *e.g.* SUPREM-III, or measured data. 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. A grid must be present for this command.

infile

The **infile** parameter selects the file name for data.

antimony, arsenic, boron, phosphorus, intersticial, vacancy, beryllium, magnesium, selenium, isilicon, tin, germanium, zinc, carbon, generic

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

EXAMPLES

profile 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

The **initialize** and **structure** statements.

COMMAND REGION

Specify a mesh region.

SYNOPSIS

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

DESCRIPTION

This statement is used to specify the material of rectangles in a rectangular mesh. **Region** statements should follow **line** statements. Every element 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 | gaas

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 silicon (from the example on the **line** page).

BUGS

None.

SEE ALSO

The **line**, **boundary**, and **init** statements.

COMMAND SELECT

Select the plot variable for the post-processing routines.

SYNOPSIS

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

DESCRIPTION

The **select** statement is used to specify the variable for display in the **contour** statement, **plot.1d** statement, **print.1d** statement, and **plot.2d** statement. In short, all of the post-processing 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 and 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:

antimony	antimony concentration
arsenic	arsenic concentration
beryllium	beryllium concentration
boron	boron concentration
carbon	carbon concentration
cesium	cesium concentration

ci.star	equilibrium interstitial concentration
cv.star	equilibrium vacancies concentration
doping	net active concentration
electrons	electron concentration
germanium	germanium
generic	generic impurity concentration
gold	gold concentration
interstitial	interstitial concentration
isilicon	silicon impurity concentration
magnesium	magnesium concentration
ni	intrinsic electron concentration
oxygen	oxygen concentration
phosphorus	phosphorus concentration
psi	potential
selenium	selenium concentration
Sxx, Sxy, Syy	components of stress in rectangular coordinates
tin	tin concentration
trap	unfilled interstitial trap concentration
vacancy	vacancy concentration
x	x coordinates
x.v	x velocity
y	y coordinates
y.v	y velocity
zinc	zinc concentration

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:

abs	absolute value
active	active portion of the specified dopant
erf	error function
erfc	complimentary error function
exp	exponential
gradx	differentiates the argument with respect to x
grady	differentiates the argument with respect to y
log	logarithm
log10	logarithm base 10
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 to 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 5.0×10^{14}.

```
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 post-processing commands e.g **plot.1d**, **plot.2d**, **print.1d**, **contour**, etc.

COMMAND STRESS

Calculate elastic stresses.

SYNOPSIS

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

DESCRIPTION

The **stress** command 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 1.4×10^{10} when deposited uniformly.

BUGS

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

SEE ALSO

The **material** statement.

COMMAND STRUCTURE

Read/write the mesh and solution information.

SYNOPSIS

structure
[(infile=<string> **| outfile=**<string>**)]**
[pisces=<string> **] [show] [backside.y=**<n> **]**
[mirror] [left] [right]
[imagetool=<string> **]**
[x.min=<n> **| x.max=**<n> **| y.min=**<n> **| y.max=**<n> **|**
z.min=<n> **| z.max=**<n> **| pixelx=**<n> **| pixely=**<n> **|**
nxfac=<n> **| nyfac=**<n> **| mode=**<n> **]**
[simpl=<string> **] [header=**<string> **]**

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 PI-

SCES (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.

backside.y

Used in conjunction with the **pisces** parameter, the substrate (assumed to be silicon) will be etched from the location specified by the argument to backside.y to the bottom of the wafer. A backside contact extending across the substrate in the x-direction will be placed at this y-value.

mirror left
mirror 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.

imagetool

This string parameter specifies the name of the binary raster file that can be used by NCSA programs such as XImage. The entire structure will be plotted unless clipped by the window parameters **x.min**, **x.max**, **y.min**, and **y.max**. In addition the size of the raster image is controlled by the parameters **pixelx** and **pixely**. Z values are derived using the **select** command. All z values will be plotted unless clipped with **z.min** and **z.max**. For details on the parameters associated with imagetool files, see below.

x.min

This float parameter is minimum x-value in microns used in generating the raster image. **x.min** defaults to the left edge of the entire structure.

x.max

This float parameter is maximum x-value in microns used in generating the raster image. **x.max** defaults to the right edge of the entire structure.

y.min

This float parameter is minimum y-value in microns used in generating the raster image. **y.min** defaults to the top edge of the entire structure.

y.max

This float parameter is maximum y-value in microns used in generating the raster image. **y.max** defaults to the bottom edge of the entire structure.

z.min

This float parameter is the minimum z-value used for generating the raster image. The z array is created using the select command. If the actual z-value is less than **z.min**, the value will be given the same color as **z.min**.

z.max

This float parameter is the maximum z-value used for generating the raster image. The z array is created using the select command. If the actual z-value is greater than **z.max**, the value will be given the same color as **z.max**.

pixelx

This integer parameter is the number of pixels in the x-direction for the raster image. The default value is 400 pixels.

pixely

This integer parameter is the number of pixels in the y-direction for the raster image. The default value is 200 pixels.

nxfac

This integer parameter is the amount of linear interpolation in the x-direction to be done on the rectangular grid that will be used to create the raster image. The default value of 1 gives the most resolution at the cost of higher CPU usage. A higher value will give a slightly coarser picture but will take less CPU time.

nyfac

This integer parameter is the amount of linear interpolation in the y-direction to be done on the rectangular grid that will be used to create the raster image. The default value of 1 gives the most resolution at the cost of higher CPU usage. A higher value will give a slightly coarser picture but will take less CPU time.

mode

This integer parameter is a special flag (0 or 1) useful for plotting doping concentrations. It assumes that the follow **select** statement was issued:

```
select z=sign(bor-phos-ars)*log10(abs(bor-phos-ars))
```

The colors are then mapped between -14 and 14. This compresses the color spectrum to allow more colors in the region of interest.

clear

This integer parameter clears a global counter which counts frame numbers is association with the **movie** parameter of the **diffuse** command.

count

This integer parameter increments the global counter. When used with the **movie** parameter of the **diffuse** command, a sequence of image files can be generated whose last 3 characters are numbers. Files in this type of sequence can be easily animated using NCSA XImage.

simpl

This string parameter generates a file in the SIMPL-2 format. This is necessary to incorporate SUPREM-IV information into the SIMPL-IPX system. Another parameter, **header**, must also be specified.

header

This string parameter is the name of the file used to describe the cross section to be described. This is the first few lines from a SIMPL-2 file. Usually these files are automatically generated when using SUPREM-IV within the SIMPL-IPX environment.

EXAMPLES

structure infile=foo

This command reads in a previously saved structure in file foo.

structure pisces=baz show backside.y=5.0

Writes a PISCES-II compatible file. The locations and number of the electrodes will be displayed. In addition the silicon substrate will be removed for y > 5.0 and a backside contact will be placed at y = 5.0.

```
structure imagetool=foo x.min=-1.0 x.max=1.0
          y.min=-1.0 y.max=1.0 pixelx=300 pixely=150
```

Writes an binary raster file for XImage of size 300 by 150 pixels. Only data in the window -1.0µm ≤ x ≤ 1.0µm, -1.0µm ≤ y ≤ 1.0µm is shown in the image.

```
structure simpl=foo.cross header=headerfile
```

Writes a SIMPL-2 ASCII file using cross section information from file headerfile.

BUGS

The **show** option echoes 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 0.1Å, some points will be duplicated.

SEE ALSO

The **initialize** statement.

Models

DESCRIPTION

This section contains the commands used to change model parameters and coefficients. The parameters and equations they belong to are described briefly on the statement descriptions. References are provided, where necessary, to help clarify the physics involved in the simulator.

The file for the initialization of all the constants is human readable (usually called modelrc). Unfortunately, this means that SUPREM-IV.GS can be slow in starting up because it has to parse the file. Local modification of the file is encouraged, because many of the parameters are areas of active research and can not be guaranteed, other parameters may be dependent upon the equipment used locally.

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

These commands 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 University Ph.D. theses for more information.

Paul Fahey – Defect Models, Fractional Interstitialcy,
Peter Griffin – Defect Models and Diffusion, Experiments,
Mark Law – Defect Models and Diffusion, Numerical Techniques,
Paul Packan – Oxidation Enhanced Diffusion Experiments,
Conor Rafferty – Oxidation Models and Stress.
Heyward Robinson – Defect Models and Diffusion.
Emily Allen – Defect Models and Diffusion.
Jeff Murray – Defect Models and Diffusion.
Chris Pass– Defect Models and Diffusion.
Linda Vanasupa – Defect Models and Activation.
M. D. Deal – Defect Models and Diffusion.

COMMAND ANTIMONY

Set the coefficients of antimony kinetics.

SYNOPSIS

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

DESCRIPTION

This statement allows the user to specify values for coefficients of antimony diffusion and segregation. The diffusion equation for antimony is [1,2]:

$$\frac{\partial C_T}{\partial t} = \nabla \left[D_V C_T \frac{C_V}{C_V^*} \nabla \ln\left(C_T \frac{C_V}{C_V^*} \frac{n}{n_i} \right) + D_I C_T \frac{C_I}{C_I^*} \nabla \ln\left(C_T \frac{C_I}{C_I^*} \frac{n}{n_i} \right) \right] \quad \textbf{(EQ 3)}$$

where C_T is the total chemical concentration of antimony, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium 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. The diffusivities are given by:

$$D_V = (1 - F_i)\ (D_X + D_M \frac{n}{n_i}) \quad \textbf{(EQ 4)}$$

$$D_I = F_i\,(D_X + D_M \frac{n}{n_i}) \quad \textbf{(EQ 5)}$$

D_X and D_M are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1 \frac{C_2}{M_{12}}\right) \quad \textbf{(EQ 6)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow 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 cm^2/sec in silicon, and **Dix.E** defaults to 3.65 eV in silicon [3]. D_X is calculated using a standard Arrhenius relationship.

Dim.0, Dim.E

These floating point parameters allow 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 cm^2/sec in silicon, and **Dim.E** defaults to 4.08 eV in silicon [3]. D_M is calculated using a standard Arrhenius relationship.

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 [4].

implanted, grown.in

Specifies whether the **Dix**, **Dim**, or **Fi** coefficients apply to implanted or grown-in antimony. If neither is specified then a specified parameter applies to both.

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 [5]:

ss.temp	ss.conc
800˚C	$2.3\times10^{19} cm^{-3}$
900˚C	$3.0\times10^{19} cm^{-3}$
1000˚C	$4.0\times10^{19} cm^{-3}$
1100˚C	$4.8\times10^{19} cm^{-3}$
1200˚C	$6.2\times10^{19} cm^{-3}$

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

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. M_{12} is calculated using a standard Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. T_r is calculated using a standard Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default antimony is a donor.

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 Seg.E=0.0
```

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. M.E. Law and J.R. Pfiester, "Low Temperature Annealing of Arsenic/ Phosphorus Junctions," *IEEE Trans. on Elec. Dev.,* **38**(2), p. 278, 1991.
2. D. Mathiot and J.C. Pfister, "Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects," *J. Appl. Phys.*, **55**(10), p. 3518, 1984.
3. R.B. Fair, "Concentration Profiles of Diffused Dopants in Silicon," Impurity Doping Processes in Silicon, Yang ed., 1981, North-Holland, New York.
4. 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.*, **46**(8), p. 784, 1985.
5. F.A. Trumbore, "Solid Solubilities of Impurity Elements in Germanium and Silicon," Bell System Tech Journal, 1960.

SEE ALSO

The **arsenic**, **boron**, **phosphorus**, **interstitial**, and **vacancy** statements.

COMMAND ARSENIC

Set the coefficients of arsenic kinetics.

SYNOPSIS

```
arsenic
        ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
        [ Dix.0=<n> ] [ Dix.E=<n> ] [ Dim.0=<n> ] [ Dim.E=<n> ]
        [ Fi = <n> ]
        [ implanted ] [ grown.in ]
        [ Ctn.0=<n> ] [ Ctn.E=<n> ]
        [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
        [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
                /gaas ) ]
        [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
        [ ( donor | acceptor ) ]
```

DESCRIPTION

This statement allows the user to specify values for coefficients of arsenic diffusion and segregation. The diffusion equation for arsenic is [1,2]:

$$\frac{\partial C_T}{\partial t} = \nabla \left[D_V C_A \frac{C_V}{C_V^*} \nabla \ln\left(C_A \frac{C_V}{C_V^*} \frac{n}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \ln\left(C_A \frac{C_I}{C_I^*} \frac{n}{n_i} \right) \right] \quad \textbf{(EQ 7)}$$

where C_T and C_A are the total chemical and active concentrations of arsenic, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium 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. The diffusivities are given by:

$$D_V = (1 - F_i)\ (D_X + D_M \frac{n}{n_i}) \qquad \textbf{(EQ 8)}$$

$$D_I = F_i\,(D_X + D_M \frac{n}{n_i}) \qquad \textbf{(EQ 9)}$$

D_X and D_M are described in greater detail below.

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

$$\text{flux} = T_r \left(C_1 \frac{C_2}{M_{12}} \right) \qquad \textbf{(EQ 10)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and Tr terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow 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 0.214 cm^2/sec in silicon, and **Dix.E** defaults to 3.65 eV in silicon [3]. D_X is calculated using a standard Arrhenius relationship.

Dim.0, Dim.E

These floating point parameters allow 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 15.0 cm^2/sec in silicon, and **Dim.E** defaults to 4.08 eV in silicon [3]. D_M is calculated using a standard Arrhenius relationship.

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 [4].

implanted, grown.in

Specifies whether the **Dix**, **Dim**, or **Fi** coefficients apply to implanted or grown-in arsenic. If neither is specified then a specified parameter applies to both.

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 obeys a standard Arrhenius relationship. The active concentration, C_A is calculated using:

$$C_T = C_A \left[1 + C_{\mathrm{tn}} \left(\frac{n}{n_i}\right)^2 \frac{C_V}{C_V^*} \right] \qquad \textbf{(EQ 11)}$$

Physically, this represents clustering of arsenic with doubly negative vacancies. The default parameters for **Ctn.0** and **Ctn.E** in silicon are 5.9×10^{-24} and 0.60 eV, respectively[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 [5]:

ss.temp	ss.conc
800˚C	$1.60 \times 10^{20} \mathrm{cm}^{-3}$
900˚C	$1.63 \times 10^{20} \mathrm{cm}^{-3}$
1000˚C	$2.14 \times 10^{20} \mathrm{cm}^{-3}$
1100˚C	$5.00 \times 10^{20} \mathrm{cm}^{-3}$

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

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. M_{12} is calculated using a standard Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. T_r is calculated using a standard Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default arsenic is a donor.

EXAMPLES

```
arsenic silicon Dix.0=8.0 Dix.E=3.65
```

This command changes the neutral defect diffusivity in silicon.

```
arsenic silicon /oxide Seg.0=30.0 Trn.0=1.66e-7 Seg.E=0.0
```

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. M.E. Law and J.R. Pfiester, “Low Temperature Annealing of Arsenic/Phosphorus Junctions,” *IEEE Trans. on Elec. Dev.*, **38**(2), p. 278, 1991.
2. D. Mathiot and J.C. Pfister, “Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects,” *J. Appl. Phys.*, **55**(10), p. 3518, 1984.
3. R.B. Fair, “Concentration Profiles of Diffused Dopants in Silicon,” Impurity Doping Processes in Silicon, Yang ed., 1981, North-Holland, New York.
4. 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.*, **46**(8), p. 784, 1985.
5. D. Nobili, A. Carabelas, G. Celotti, and S. Solmi, Journal of the Electrochemical Society, **130**, p 922, 1983.

SEE ALSO

The **antimony**, **boron**, **phosphorus**, **interstitial**, and **vacancy** statements.

COMMAND BERYLLIUM

Set the coefficients of beryllium kinetics.

SYNOPSIS

```
beryllium
    ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
    [ Dix.0=<n> ] [ Dix.E=<n> ]
    [ Dip.0=<n> ] [ Dip.E=<n> ] [ Dipp.0=<n> ] [ Dipp.E=<n> ]
    [ Fi = <n> ]
    [ implanted ] [ grown.in ]
    [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
    [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
        /gaas ) ]
    [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
    [ ( donor | acceptor ) ]
```

DESCRIPTION

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

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_A \frac{C_V}{C_V^*}\nabla \ln\left(C_A \frac{C_V}{C_V^*}\frac{p}{n_i}\right) + D_I C_A \frac{C_I}{C_I^*}\nabla \ln\left(C_A \frac{C_I}{C_I^*}\frac{p}{n_i}\right)\right] \quad \text{(EQ 12)}$$

where C_T and C_A are the total chemical and active concentrations of beryllium, C_A is the total electrically active concentration of beryllium, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium value, 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. The diffusivities are given by:

$$D_V = (1 - F_i)\left(D_X + D_p\left(\frac{p}{n_i}\right) + D_{PP}\left(\frac{p}{n_i}\right)^2\right) \tag{EQ 13}$$

$$D_I = F_i\left(D_X + D_p\left(\frac{p}{n_i}\right) + D_{PP}\left(\frac{p}{n_i}\right)^2\right) \tag{EQ 14}$$

D_X , D_P, and D_{PP} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1 \frac{C_2}{M_{12}}\right) \tag{EQ 15}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of D_X, the beryllium diffusivity with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. **Dix.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dix.E** defaults to 0.0 eV in gallium arsenide. D_X is calculated using a standard Arrhenius relationship.

Dip.0, Dip.E

These floating point parameters allow the specification of D_P, the beryllium diffusivity with singly positive defects. **Dip.0** is the pre-exponential constant and **Dip.E** is the activation energy. **Dip.0** defaults to 2.1×10^{-6} cm^2/sec for implanted beryllium in gallium arsenide and 2.1×10^{-6} cm^2/sec for grown-in beryllium in gallium arsenide, and **Dip.E** defaults to 1.74 eV for both in gallium arsenide [1, 2, 3]. D_P is calculated using a standard Arrhenius relationship.

Dipp.0, Dipp.E

These floating point parameters allow the specification of D_{PP}, the beryllium diffusivity with doubly positive defects. **Dipp.0** is the pre-exponential constant and **Dipp.E** is the activation energy. **Dipp.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dipp.E** defaults to 0.0 eV in gallium arsenide. D_{PP} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether beryllium diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 1.0.

implanted, grown.in

Specifies whether the **Dix**, **Dip**, **Dipp**, or **Fi** coefficients apply to implanted or grown-in beryllium. If neither is specified then a specified parameter applies to both.

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:

ss.temp	ss.conc
700˚C	1.0×10^{19} cm^{-3}
1250˚C	1.0×10^{19} cm^{-3}

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

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 segregation constant follows an Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. The transfer coefficient follows an Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default, beryllium is an acceptor.

EXAMPLES

```
beryllium gaas implanted Dip.0=2.2e-6 Dip.E=1.76
```

This command changes the positive defect diffusivity for implanted beryllium in gallium arsenide.

```
beryllium gaas /nitride Seg.0=1126.0 Seg.E=0.91
          Trn.0=1.66e-7
```

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be half the nitride concentration in equilibrium at 1100˚C.

BUGS

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

REFERENCES

1. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1990.
2. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1991.
3. M.D. Deal and H.G. Robinson, “Diffusion of Implanted Beryllium in P and N Type GaAs,” *Appl. Phys. Lett.* **55**, 1990.

SEE ALSO

The **antimony**, **arsenic**, **boron**, **carbon**, **generic**, **germanium**, **interstitial**, **magnesium**, **phosphorus**, **selenium**, **isilicon**, **tin**, **vacancy**, and **zinc** statements.

COMMAND BORON

Set the coefficients of boron kinetics.

SYNOPSIS

```
boron
        ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
        [ Dix.0=<n> ] [ Dix.E=<n> ] [ Dip.0=<n> ] [ Dip.E=<n> ]
        [ Fi = <n> ]
        [ implanted ] [ grown.in ]
        [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
        [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
                /gaas ) ]
        [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
        [ ( donor | acceptor ) ]
```

DESCRIPTION

This statement allows the user to specify values for coefficients of boron diffusion and segregation. The diffusion equation for boron is [1,2]:

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_A \frac{C_V}{C_V^*} \nabla \ln\left(C_A \frac{C_V}{C_V^*} \frac{p}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \ln\left(C_A \frac{C_I}{C_I^*} \frac{p}{n_i} \right) \right] \quad \text{(EQ 16)}$$

where C_T and C_A are the total chemical and active concentrations of boron, C_A is the total electrically active concentration of boron, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium value, 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. The diffusivities are given by:

$$D_V = (1 - F_i)\,(D_X + D_P \frac{p}{n_i}) \qquad \textbf{(EQ 17)}$$

$$D_I = F_i\,(D_X + D_P \frac{p}{n_i}) \qquad \textbf{(EQ 18)}$$

D_X and D_P are described in greater detail below.

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

$$\text{flux} = T_r \left(C_1 \frac{C_2}{M_{12}} \right) \qquad \textbf{(EQ 19)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of D_X, the boron diffusivity with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. **Dix.0** defaults to 0.28 cm^2/sec in silicon, and **Dix.E** defaults to 3.46 eV in silicon [3]. D_X is calculated using a standard Arrhenius relationship.

Dip.0, Dip.E

These floating point parameters allow the specification of D_P, the boron diffusivity with singly positive defects. **Dip.0** is the pre-exponential constant and **Dip.E** is the activation energy. **Dip.0** defaults to 0.23 cm^2/sec in silicon, and **Dip.E** defaults to 3.46 eV in silicon [3]. D_P is calculated using a standard Arrhenius relationship.

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 0.8 [4].

implanted, grown.in

Specifies whether the **Dix**, **Dip**, or **Fi** coefficients apply to implanted or grown-in boron. If neither is specified then a specified parameter applies to both.

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 [5].

ss.temp	ss.conc	ss.temp	ss.conc
800˚C	$3.45\times10^{19}\text{cm}^{-3}$	1050˚C	$1.53\times10^{20}\text{cm}^{-3}$
825˚C	$4.13\times10^{19}\text{cm}^{-3}$	1075˚C	$1.73\times10^{20}\text{cm}^{-3}$
850˚C	$4.90\times10^{19}\text{cm}^{-3}$	1100˚C	$1.94\times10^{20}\text{cm}^{-3}$
875˚C	$5.78\times10^{19}\text{cm}^{-3}$	1125˚C	$2.16\times10^{20}\text{cm}^{-3}$
900˚C	$6.76\times10^{19}\text{cm}^{-3}$	1150˚C	$2.40\times10^{20}\text{cm}^{-3}$
925˚C	$7.86\times10^{19}\text{cm}^{-3}$	1175˚C	$2.66\times10^{20}\text{cm}^{-3}$
950˚C	$9.08\times10^{19}\text{cm}^{-3}$	1200˚C	$2.94\times10^{20}\text{cm}^{-3}$
975˚C	$1.04\times10^{20}\text{cm}^{-3}$	1225˚C	$3.24\times10^{20}\text{cm}^{-3}$
1000˚C	$1.19\times10^{20}\text{cm}^{-3}$	1250˚C	$3.55\times10^{20}\text{cm}^{-3}$
1025˚C	$1.35\times10^{20}\text{cm}^{-3}$	1275˚C	$3.89\times10^{20}\text{cm}^{-3}$

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

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 segregation constant follows an Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. The transfer coefficient follows an Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default boron is an acceptor.

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 half the oxide concentration in equilibrium at 1100˚C.

BUGS

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

REFERENCES

1. M.E. Law and J.R. Pfiester, “Low Temperature Annealing of Arsenic/Phosphorus Junctions,” *IEEE Trans. on Elec. Dev.*, **38**(2), p. 278, 1991.
2. D. Mathiot and J.C. Pfister, “Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects,” *J. Appl. Phys.*, **55**(10), p. 3518, 1984.
3. G.P. Barbuscia, G. Chin, R.W. Dutton, T. Alvarez and L. Arledge, “Modeling of Polysilicon Dopant Diffusion for Shallow Junction Bipo-

lar Technology," International Electron Devices Meeting, San Francisco, p. 757, 1984.

4. P.A. Packan and J.D. Plummer, "Temperature and Time Dependence of B and P Diffusion in Si During Surface Oxidation," *J. Appl. Phys.*, **68**(8), 1990.
5. 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

The **antimony**, **arsenic**, **interstitial**, **phosphorus**, and **vacancy** statements.

COMMAND CARBON

Set the coefficients of carbon kinetics.

SYNOPSIS

```
carbon
    ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
    [ Dix.0=<n> ] [ Dix.E=<n> ]
    [ Dip.0=<n> ] [ Dip.E=<n> ] [ Dipp.0=<n> ] [ Dipp.E=<n> ]
    [ Fi = <n> ]
    [ implanted ] [ grown.in ]
    [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
    [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
          /gaas ) ]
    [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
    [ ( donor | acceptor ) ]
```

DESCRIPTION

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

$$\frac{\partial C_T}{\partial t} = \nabla \left[D_V C_A \frac{C_V}{C_V^*} \nabla \ln\left(C_A \frac{C_V}{C_V^*} \frac{p}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \ln\left(C_A \frac{C_I}{C_I^*} \frac{p}{n_i} \right) \right] \qquad \text{(EQ 20)}$$

where C_T and C_A are the total chemical and active concentrations of carbon, C_A is the total electrically active concentration of carbon, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium value, 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. The diffusivities are given by:

$$D_V = (1 - F_i)\left(D_X + D_p\left(\frac{p}{n_i}\right) + D_{PP}\left(\frac{p}{n_i}\right)^2\right) \qquad \textbf{(EQ 21)}$$

$$D_I = F_i\left(D_X + D_p\left(\frac{p}{n_i}\right) + D_{PP}\left(\frac{p}{n_i}\right)^2\right) \qquad \textbf{(EQ 22)}$$

D_X , D_P, and D_{PP} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1 \frac{C_2}{M_{12}}\right) \qquad \textbf{(EQ 23)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of D_X, the carbon diffusivity with neutral defects. **Dix.0** is the pre-exponential constant and

Dix.E is the activation energy. **Dix.0** defaults to 1.0×10^{-16} cm^2/sec in gallium arsenide, and **Dix.E** defaults to 0.0 eV in gallium arsenide [1, 2]. D_X is calculated using a standard Arrhenius relationship.

Dip.0, Dip.E

These floating point parameters allow the specification of D_P, the carbon diffusivity with singly positive defects. **Dip.0** is the pre-exponential constant and **Dip.E** is the activation energy. **Dip.0** defaults to 0.0 cm^2/sec in gallium arsenide and **Dip.E** defaults to 0.0 eV in gallium arsenide. D_P is calculated using a standard Arrhenius relationship.

Dipp.0, Dipp.E

These floating point parameters allow the specification of D_{PP}, the carbon diffusivity with doubly positive defects. **Dipp.0** is the pre-exponential constant and **Dipp.E** is the activation energy. **Dipp.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dipp.E** defaults to 0.0 eV in gallium arsenide. D_{PP} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether carbon diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 1.0.

implanted, grown.in

Specifies whether the **Dix**, **Dip**, **Dipp**, or **Fi** coefficients apply to implanted or grown-in carbon. If neither is specified then a specified parameter applies to both.

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:

ss.temp	ss.conc
700˚C	1.0×10^{19}cm^{-3}
1250˚C	1.0×10^{19}cm^{-3}

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

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 segregation constant follows an Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. The transfer coefficient follows an Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default, carbon is an acceptor.

EXAMPLES

```
carbon gaas implanted Dip.0=2.2e-6 Dip.E=1.76
```

This command changes the positive defect diffusivity for implanted carbon in gallium arsenide.

```
carbon gaas /nitride Seg.0=1126.0 Seg.E=0.91 Trn.0=1.66e-7
```

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be half the nitride concentration in equilibrium at 1100˚C.

BUGS

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

REFERENCES

1. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1990.
2. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1991.

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **generic**, **germanium**, **interstitial**, **magnesium**, **phosphorus**, **selenium**, **isilicon**, **tin**, **vacancy**, and **zinc** statements.

COMMAND GENERIC

Set the coefficients for a generic impurity's kinetics.

SYNOPSIS

```
generic
        ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
        [ Dix.0=<n> ] [ Dix.E=<n> ]
        [ Dim.0=<n> ] [ Dim.E=<n> ]
        [ Dimm.0=<n> ] [ Dimm.E=<n> ]
        [ Dimmm.0=<n> ] [ Dimmm.E=<n> ]
        [ Dip.0=<n> ] [ Dip.E=<n> ]
        [ Dipp.0=<n> ] [ Dipp.E=<n> ]
        [ Dippp.0=<n> ] [ Dippp.E=<n> ]
        [ Fi = <n> ]
        [ implanted ] [ grown.in ]
        [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
        [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly | gaas ) ]
        [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
        [ ( donor | acceptor ) ]
```

DESCRIPTION

This statement allows the user to specify values for coefficients of a generic impurity's diffusion and segregation. The diffusion equation for the generic impurity is:

$$\frac{\partial C_T}{\partial t} = \nabla \left[D_V C_A \frac{C_V}{C_V^*} \nabla \ln\left(C_A \frac{C_V}{C_V^*} \frac{p}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \ln\left(C_A \frac{C_I}{C_I^*} \frac{p}{n_i} \right) \right] \quad \textbf{(EQ 24)}$$

where C_T and C_A are the total chemical and electrically active concentrations of the generic impurity, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium value, 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. The diffusivities are given by:

$$D_V = (1 - F_i) \begin{bmatrix} D_X + D_M \left(\frac{n}{n_i}\right) + D_{MM} \left(\frac{n}{n_i}\right)^2 + D_{MMM} \left(\frac{n}{n_i}\right)^3 \\ + D_p \left(\frac{p}{n_i}\right) + D_{PP} \left(\frac{p}{n_i}\right)^2 + D_{PPP} \left(\frac{p}{n_i}\right)^3 \end{bmatrix} \quad \textbf{(EQ 25)}$$

$$D_I = F_i \begin{bmatrix} D_X + D_M \left(\frac{n}{n_i}\right) + D_{MM} \left(\frac{n}{n_i}\right)^2 + D_{MMM} \left(\frac{n}{n_i}\right)^3 \\ + D_p \left(\frac{p}{n_i}\right) + D_{PP} \left(\frac{p}{n_i}\right)^2 + D_{PPP} \left(\frac{p}{n_i}\right)^3 \end{bmatrix} \quad \textbf{(EQ 26)}$$

D_X, D_M, D_{MM}, D_{MMM}, D_P, D_{PP}, and D_{PPP} are described in greater detail below.

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

$$\text{flux} = T_r \left(C_1 \frac{C_2}{M_{12}} \right) \quad \textbf{(EQ 27)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and Tr terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of D_X, the generic impurity's diffusivity with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. **Dix.0** defaults to 0.0 cm^2/sec and **Dix.E** defaults to 0.0 eV. D_X is calculated using a standard Arrhenius relationship.

Dim.0, Dim.E

These floating point parameters allow the specification of D_M, the generic impurity's diffusivity with singly positive defects. **Dim.0** is the pre-exponential constant and **Dim.E** is the activation energy. **Dim.0** defaults to 0.0 cm^2/sec and **Dim.E** defaults to 0.0 eV. D_M is calculated using a standard Arrhenius relationship.

Dimm.0, Dimm.E

These floating point parameters allow the specification of D_{MM}, the generic impurity's diffusivity with doubly positive defects. **Dimm.0** is the pre-exponential constant and **Dimm.E** is the activation energy. **Dimm.0** defaults to 0.0 cm^2/sec and **Dimm.E** defaults to 0.0 eV. D_{MM} is calculated using a standard Arrhenius relationship.

Dimmm.0, Dimmm.E

These floating point parameters allow the specification of D_{MMM}, the generic impurity's diffusivity with triply positive defects. **Dimmm.0** is the pre-exponential constant and **Dimmm.E** is the activation energy. **Dimmm.0** defaults to 0.0 cm^2/sec and **Dimmm.E** defaults to 0.0 eV. D_{MMM} is calculated using a standard Arrhenius relationship.

Dip.0, Dip.E

These floating point parameters allow the specification of D_P, the generic impurity's diffusivity with singly positive defects. **Dip.0** is the pre-exponential constant and **Dip.E** is the activation energy. **Dip.0** defaults to 0.0 cm^2/sec and **Dip.E** defaults to 0.0 eV. D_P is calculated using a standard Arrhenius relationship.

Dipp.0, Dipp.E

These floating point parameters allow the specification of D_{PP}, the generic impurity's diffusivity with doubly positive defects. **Dipp.0** is the pre-exponential constant and **Dipp.E** is the activation energy. **Dipp.0** defaults to 0.0 cm^2/sec and **Dipp.E** defaults to 0.0 eV. D_{PP} is calculated using a standard Arrhenius relationship.

Dippp.0, Dippp.E

These floating point parameters allow the specification of D_{PPP}, the generic impurity's diffusivity with triply positive defects. **Dippp.0** is the pre-exponential constant and **Dippp.E** is the activation energy. **Dippp.0** defaults to 0.0 cm^2/sec and **Dippp.E** defaults to 0.0 eV. D_{PPP} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether generic diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 0.0.

implanted, grown.in

Specifies whether the **Dix**, **Dim**, **Dimm**, **Dimmm**, **Dip**, **Dipp**, **Dippp**, or **Fi** coefficients apply to implanted or grown-in generic. If neither is specified then a specified parameter applies to both.

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:

ss.temp	ss.conc
700˚C	1.0×10^{19}cm^{-3}
1250˚C	1.0×10^{19}cm^{-3}

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

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 segregation constant follows an Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. The transfer coefficient follows an Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. There is no default set so the type must be set before using the generic impurity.

EXAMPLES

```
generic gaas implanted Dip.0=2.2e-6 Dip.E=1.76
```

This command changes the positive defect diffusivity for an implanted generic impurity in gallium arsenide.

```
generic gaas /nitride Seg.0=1126.0 Seg.E=0.91 Trn.0=1.66e-7
```

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be half the nitride concentration in equilibrium at 1100˚C.

BUGS

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

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **germanium**, **interstitial**, **magnesium**, **phosphorus**, **selenium**, **isilicon**, **tin**, **vacancy**, and **zinc** statements.

COMMAND GERMANIUM

Set the coefficients of germanium kinetics.

SYNOPSIS

```
germanium
        ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
        [ Dix.0=<n> ] [ Dix.E=<n> ] [ Dim.0=<n> ] [ Dim.E=<n> ]
        [ Dimm.0=<n> ] [ Dimm.E=<n> ] [ Fi = <n> ]
        [ implanted ] [ grown.in ]
        [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
        [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
                /gaas ) ]
        [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
        [ ( donor | acceptor ) ]
```

DESCRIPTION

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

$$\frac{\partial C_T}{\partial t} = \nabla \left[D_V C_A \frac{C_V}{C_V^*} \nabla \ln\left(C_A \frac{C_V}{C_V^*} \frac{n}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \ln\left(C_A \frac{C_I}{C_I^*} \frac{n}{n_i} \right) \right] \quad \textbf{(EQ 28)}$$

where C_T and C_A are the total chemical and active concentrations of germanium, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium 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. The diffusivities are given by:

$$D_V = (1 - F_i)\left(D_X + D_M\left(\frac{n}{n_i}\right) + D_{MM}\left(\frac{n}{n_i}\right)^2\right) \qquad \text{(EQ 29)}$$

$$D_I = F_i\left(D_X + D_M\left(\frac{n}{n_i}\right) + D_{MM}\left(\frac{n}{n_i}\right)^2\right) \qquad \text{(EQ 30)}$$

D_X, D_M, and D_{MM} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1 \frac{C_2}{M_{12}}\right) \qquad \text{(EQ 31)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of the germanium diffusing with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. **Dix.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dix.E** defaults to 0.0 eV in gallium arsenide. D_X is calculated using a standard Arrhenius relationship.

Dim.0, Dim.E

These floating point parameters allow the specification of the germanium diffusing with singly negative defects. **Dim.0** is the pre-exponential constant and **Dim.E** is the activation energy. **Dim.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dim.E** defaults to 0.0 eV in gallium arsenide. D_M is calculated using a standard Arrhenius relationship.

Dimm.0, Dimm.E

These floating point parameters allow the specification of the germanium diffusing with doubly negative defects. **Dimm.0** is the pre-exponential constant and **Dimm.E** is the activation energy. **Dimm.0** defaults to 1.1×10^{-3} cm^2/sec in gallium arsenide, and **Dimm.E** defaults to 2.9 eV in gallium arsenide [1,2]. D_{MM} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether germanium diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 0.0.

implanted, grown.in

Specifies whether the **Dix**, **Dim**, **Dimm**, or **Fi** coefficients apply to implanted or grown-in germanium. If neither is specified then a specified parameter applies to both.

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

ss.temp	ss.conc
700˚C	1.0×10^{19}cm^{-3}
1250˚C	1.0×10^{19}cm^{-3}

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

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. M_{12} is calculated using a standard Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. T_r is calculated using a standard Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default, germanium is an donor.

EXAMPLES

```
germanium gaas implanted Dimm.0=2.1e-3 Dimm.E=3.1
```

This command changes the doubly negative defect diffusivity for implanted germanium in silicon.

```
germanium gaas /nitride Seg.0=30.0 Trn.0=1.66e-7
Seg.E=0.0
```

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be 30.0 times the nitride concentration in equilibrium.

BUGS

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

REFERENCES

1. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1990.
2. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1991.
3. E.L. Allen, J.J. Murray, M.D. Deal, J.D. Plummer, K.S.Jones, and W.S. Robert, “Diffusion of Ion-Implanted Group IV Dopants in GaAs,” *J. Electrochem. Soc.*, **138**, p. 3440, 1991.

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **generic**, **interstitial**, **magnesium**, **phosphorus**, **selenium**, **isilicon**, **tin**, **vacancy**, and **zinc** statements.

COMMAND INTERSTITIAL

Set coefficients of interstitial kinetics.

SYNOPSIS

```
interstitial
        ( silicon | oxide | poly | oxynitr | nitride | gas | gaas )
        [ 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> ] [ tneg.0=<n> ] [ tneg.E=<n>]
        [ pos.0=<n> ] [ pos.E=<n> ] [ dpos.0=<n> ] [ dpos.E=<n> ]
        [ tpos.0=<n> ] [ tpos.E=<n> ]
        [ ( /silicon | /oxide | /poly | /oxynitr | /nitride | /gas |
                /gaas ) ]
        [ time.inj ] [ growth.inj ] [ recomb ] [ segregation ]
        [ 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> ]
        [ Seg.E=<n> ] [ Seg.0=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
```

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, gallium arsenide, 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 which can be written [1]:

$$\frac{\partial (C_I - C_{\mathrm{ET}})}{\partial t} = \nabla \left(-J_I - \sum_{\mathrm{imp}} J_{AI}\right) - R \qquad \textbf{(EQ 32)}$$

where C_I is the interstitial concentration, C_{ET} is the number of empty interstitial traps, 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) the fluxes from the dopants, J_{AI}, are ignored. Only in the **full.cpl** model are they 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 [1].

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:

$$\frac{\partial C_{\mathrm{ET}}}{\partial t} = -K_T \left[C_{\mathrm{ET}} C_I - \frac{e^*}{1 - e^*} C_I^* (C_T - C_{\mathrm{ET}}) \right] \qquad \textbf{(EQ 33)}$$

where C_T is the total trap concentration, K_T is the trap reaction coefficient, e^* is the equilibrium trap occupancy ratio. (see trap 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, arsenic, boron, and phosphorus) The interstitial flux can be written [1]:

$$-J_I = D_I C_I^* \nabla \frac{C_I}{C_I^*} \qquad \textbf{(EQ 34)}$$

It is important to note that the equilibrium concentration, C_I^* 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 interstitials and vacancies. This can be expressed:

$$R = K_R (C_I C_V - C_I^* C_V^*) \qquad \textbf{(EQ 35)}$$

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 [4]:

$$J_I \vec{n} + K_I (C_I - C_I^*) = g \qquad \textbf{(EQ 36)}$$

where $\vec{n}$ is the surface normal, K_I 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.

silicon, oxide, poly, oxynitr, nitride, gaas, 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/sec. The default values are $1 \times 10^6\,cm^2$/sec and 3.22 eV [5] for Si and $1 \times 10^{-12}\,cm^2$/sec and 0.0 eV for GaAs.

Kr.0, Kr.E

These floating point parameters allow the specification of the bulk recombination rate in cm^3/sec. The default is 1.4 cm^3/sec and 3.99 eV [6] for Si and $1 \times 10^{14}\,cm^3$/sec and 0.0 eV for GaAs.

Cstar.0, Cstar.E

These parameters allow the specification of the total equilibrium concentration of interstitials in intrinsically doped conditions. The default values are $3.1 \times 10^{19}\,cm^{-3}$ and 1.58 eV [5] for Si and $1 \times 10^{14}\,cm^{-3}$ and 0.0 eV for GaAs.

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. A diffusion limited model has been assumed [6] and results in defaults of $1.1 \times 10^{-2}\,cm^3$/sec and 3.22 eV for Si, the same values are used for GaAs.

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

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

$$C_I^{=} = \frac{C_{\mathrm{Star}}\,\mathrm{dneg}}{\mathrm{neu} + \mathrm{neg} + \mathrm{dneg} + \mathrm{tneg} + \mathrm{pos} + \mathrm{dpos} + \mathrm{tpos}} \quad \text{(EQ 37)}$$

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

$$C_I^{*} = C_{\mathrm{Star}} \frac{\mathrm{neu} + \mathrm{neg}\frac{n}{n_i} + \mathrm{dneg}\left(\frac{n}{n_i}\right)^2 + \mathrm{tneg}\left(\frac{n}{n_i}\right)^3 + \mathrm{pos}\frac{p}{n_i} + \mathrm{dpos}\left(\frac{p}{n_i}\right)^2 + \mathrm{tpos}\left(\frac{p}{n_i}\right)^3}{\mathrm{neu} + \mathrm{neg} + \mathrm{dneg} + \mathrm{tneg} + \mathrm{pos} + \mathrm{dpos} + \mathrm{tpos}} \quad \text{(EQ 38)}$$

There is very little data for the interstitial charge states; it is matter of ongoing research. The default values for Si are chosen based on an analysis of Giles [7]. See Table 4 in the section “Adding SUPREM 3.5’s GaAs Models and Parameters to SUPREM-IV” for GaAs values.

/silicon, /oxide, /poly, /oxynitr, /nitride, /gaas, /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:

$$K_I = K_{\text{Surf}}\left(K_{\text{rat}}\left(\frac{v_{\text{ox}}}{B/A}\right)^{K_{\text{pow}}} + 1\right) \qquad \text{(EQ 39)}$$

where v_{ox} is the interface velocity, and B/A is the Deal Grove oxide growth coefficient. (see **oxide**). 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) K_{Surf} 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:

$$g = \theta V_{\text{mole}} v_{\text{ox}}\left(\frac{v_{\text{ox}}}{B/A}\right)^{G_{\text{pow}}} \qquad \text{(EQ 40)}$$

Theta, θ, is the fraction of silicon atoms consumed during growth that are reinjected as interstitials. V_{mole} is the lattice density of the consumed material. Once again, this model only makes sense 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:

$$g = A\,(t + t_0)^{T_{\mathrm{pow}}} \qquad \text{(EQ 41)}$$

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 decays exponentially in time.

Seg.0, Seg.E

These parameters allow the computation of the equilibrium segregation concentrations. The segregation constant follows an Arrhenius relationship. This is for use in help supporting future model development, and is currently not in use.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. The transfer coefficient follows an Arrhenius relationship. This is for use in help supporting future model development, and is currently not in use.

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 sil /oxi 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 3.5×10^{-3}cm/sec.

```
inter sil neu.0=1.0 neg.0=1.0 pos.0=0.0
        dneg.0=0.0 dpos.0=0.0
inter sil 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 date at any time. Further, experimental verification is difficult. 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. M.E. Law and J.R. Pfiester, “Low Temperature Annealing of Arsenic/Phosphorus Junctions,” *IEEE Trans. on Elec. Dev.*, **38**(2), p. 278, 1991.
2. P.B. Griffin and J.D. Plummer, “Process Physics Determining 2-D Impurity Profiles in VLSI Devices,” International Electron Devices Meeting, Los Angeles, p. 522, 1986.
3. W. Shockley and J.T. Last, “Statistics of the Charge Distribution for a Localized Flaw in a Semiconductor,” *Phys. Rev.*, **107**(2), p. 392, 1957.
4. S.M. Hu, “On Interstitial and Vacancy Concentrations in Presence of Injection,” *J. Appl. Phys.*, **57**, p. 1069, 1985.
5. C. Boit, F. Lau and R. Sittig, “Gold Diffusion in Silicon by Rapid Optical Annealing,” *Appl. Phys. A*, **50**, p. 197, 1990.
6. M.E. Law, “Parameters for Point Defect Diffusion and Recombination,” *IEEE Trans. on CAD*, Accepted for Publication, Sept., 1991.
7. M.D. Giles, “Defect Coupled Diffusion at High Concentrations,” *IEEE Trans. on CAD*, **8**(4), p. 460, 1989.
8. G.A. Baraff and M. Schluter, “Electronic Structure, Total Energies, and Abundances of the Elementary Point Defects in GaAs,” *Phys. Rev. Lett.*, **55**, 1327 (1985).
9. R.W. Jansen, D.S. Wolde-Kidane, and O.F. Sankey, “Energetics and Deep Levels of Interstitial Defects in the Compound Semiconductors GaAs, AlAs, ZnSe, and ZnTe,” *J. Appl. Phys.*, **64**, 2415 (1988).

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **generic**, **germanium**, **magnesium**, **phosphorus**, **selenium**, **isilicon**, **tin**, **vacancy**, and **zinc** statements.

COMMAND MAGNESIUM

Set the coefficients of magnesium kinetics.

SYNOPSIS

```
magnesium
     ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
     [ Dix.0=<n> ] [ Dix.E=<n> ]
     [ Dip.0=<n> ] [ Dip.E=<n> ] [ Dipp.0=<n> ] [ Dipp.E=<n> ]
     [ Fi = <n> ]
     [ implanted ] [ grown.in ]
     [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
     [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
          /gaas ) ]
     [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
     [ ( donor | acceptor ) ]
```

DESCRIPTION

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

$$\frac{\partial C_T}{\partial t} = \nabla \left[D_V C_A \frac{C_V}{C_V^*} \nabla \ln\left(C_A \frac{C_V}{C_V^*} \frac{p}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \ln\left(C_A \frac{C_I}{C_I^*} \frac{p}{n_i} \right) \right] \quad \text{(EQ 42)}$$

where C_T and C_A are the total chemical and active concentrations of magnesium, C_A is the total electrically active concentration of magnesium, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium value, 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. The diffusivities are given by:

$$D_V = (1 - F_i)\left(D_X + D_p\left(\frac{p}{n_i}\right) + D_{PP}\left(\frac{p}{n_i}\right)^2\right) \qquad \textbf{(EQ 43)}$$

$$D_I = F_i\left(D_X + D_p\left(\frac{p}{n_i}\right) + D_{PP}\left(\frac{p}{n_i}\right)^2\right) \qquad \textbf{(EQ 44)}$$

D_X, D_P, and D_{PP} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1\frac{C_2}{M_{12}}\right) \qquad \textbf{(EQ 45)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of D_X, the magnesium diffusivity with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. **Dix.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dix.E** defaults to 0.0 eV in gallium arsenide. D_X is calculated using a standard Arrhenius relationship.

Dip.0, Dip.E

These floating point parameters allow the specification of D_P, the beryllium diffusivity with singly positive defects. **Dip.0** is the pre-exponential constant and **Dip.E** is the activation energy. **Dip.0** defaults to 6.1×10^{-2} cm^2/sec for implanted magnesium in gallium arsenide and 6.8×10^{-3} cm^2/sec for grown-in magnesium in gallium arsenide, and **Dip.E** defaults to 2.8 eV for both in gallium arsenide [1,2,3]. D_P is calculated using a standard Arrhenius relationship.

Dipp.0, Dipp.E

These floating point parameters allow the specification of D_{PP}, the magnesium diffusivity with doubly positive defects. **Dipp.0** is the pre-exponential constant and **Dipp.E** is the activation energy. **Dipp.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dipp.E** defaults to 0.0 eV in gallium arsenide. D_{PP} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether magnesium diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 1.0.

implanted, grown.in

Specifies whether the **Dix**, **Dip**, **Dipp**, or **Fi** coefficients apply to implanted or grown-in magnesium. If neither is specified then a specified parameter applies to both.

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:

ss.temp	ss.conc
700˚C	$1.0\times10^{19}\text{cm}^{-3}$
1250˚C	$1.0\times10^{19}\text{cm}^{-3}$

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

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 segregation constant follows an Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. The transfer coefficient follows an Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default, magnesium is an acceptor.

EXAMPLES

```
magnesium gaas implanted Dip.0=2.2e-6 Dip.E=1.76
```

This command changes the positive defect diffusivity for implanted magnesium in gallium arsenide.

```
magnesium gaas /nitride Seg.0=1126.0 Seg.E=0.91
Trn.0=1.66e-7
```

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be half the nitride concentration in equilibrium at 1100˚C.

BUGS

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

REFERENCES

1. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1990.
2. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1991.
3. H.G. Robinson, M.D. Deal, and D.A. Stevenson, “Hole Dependent Diffusion of Implanted Mg in GaAs,” *Appl. Phys. Lett.* **58**, p. 2801, 1990.
4. H.G. Robinson, “Diffusion of P-Type Dopants in GaAs,” *Stanford University Ph.D. Thesis, 1992.*

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **generic**, **germanium**, **interstitial**, **phosphorus**, **selenium**, **isilicon**, **tin**, **vacancy**, and **zinc** statements.

COMMAND MATERIAL

Set the coefficients of some materials.

SYNOPSIS

```
material
        ( silicon | oxide | poly | oxynitr | nitride | photores |
                aluminum )
        [ ( 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=<string> ]
        [ intrin.sig=<n> ]
        [ act.a=<n> ] [ act.b=<n> ] [ ( n.type | p.type ) ]
```

DESCRIPTION

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

silicon | oxide | oxynitride | nitride | poly | photores | aluminu

These parameters indicate the material that the remainder of the parameters apply to.

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.

$$n_i = N_{i.0} e^{(-N_{i.\mathrm{E}}/kT)} T^{N_{i.\mathrm{pow}}} \quad \textbf{(EQ 46)}$$

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** = $3.9 \times 10^{16} \mathrm{cm}^3$, **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/cm^2. **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).

act.a, act.b

GaAs activation modeling includes a compensation mechanism. This is modeled in SUPREM-IV.GS using the following expression:

$$(N_D - N_A)_{\text{effective}} = \frac{A\,(N_D - N_A)}{1 + (N_D - N_A)/B} \qquad \text{(EQ 47)}$$

Where A and B in the above expression are set by the **act.a** and **act.b** parameters. **act.a** and **act.b** may be expressions in terms of T, the temperature in degrees C. The values of A and B are dependent upon the local dominant impurity type so that one of the **p.type** or **n.type** parameters must be specified when specifying **act.a** or **act.b**.

n.type, p.type

When setting the GaAs activation coefficients using the **act.a** and **act.b** parameters, one of the **n.type** or **p.type** parameters must be specified to indicate the type of region that they are to apply to.

EXAMPLES

```
material silicon eps = 11.9
```

This statement specifies the silicon relative permittivity.

```
mate nitride lcte=(3e-6+2*1e-10*T)
                  intrin.sig=1.4e10 You=3e12
```

This statement gives the thermal expansion coefficient of nitride as a function of absolute temperature T. Thus at 0˚K the coefficient is 0.0003%/˚K. The initial stress in the nitride film is 1.4×10^{10}dynes/cm^2 and the Young's modulus is 3×10^{12}dynes/cm^2.

BUGS

There is limitation in the kind of temperature dependence n_i 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

The **oxide**, **stress**, and **diffuse** statements.

COMMAND OXIDE

Specify oxidation coefficients.

SYNOPSIS

```
oxide
        orientation= ( 111 | 110 | 100 )
        ( dry | wet )
        [ lin.l.0=<n> ] [ lin.l.e=<n> ] [ lin.h.0=<n> ] [ lin.h.e=<n> ]
        [ l.break=<n> ] [ l.pde=<n> ]
        [ par.l.0=<n> ] [ par.l.e=<n> ] [ par.h.0=<n> ] [ par.h.e=<n> ]
        [ p.break=<n> ] [ p.pdep=<n> ]
        [ ori.dep ] [ ori.fac=<n> ]
        [ thinox.0=<n> ] [ thinox.e=<n> ] [ thinox.l=<n> ]
        [ hcl.pc=<n> ] [ hclT=<string> ] [ hclP=<string> ]
        [ hcl.par=<string> ] [ hcl.lin=<string> ]
        [ baf.dep ] [ baf.ebk=<n> ] [ baf.pe=<n> ] [ baf.ppe=<n> ]
        [ baf.ne=<n> ] [ baf.nne=<n> ] [ baf.k0=<n> ] [ baf.ke=<n> ]
        [ stress.dep ] [ Vc=<n> ] [ Vr=<n> ] [ Vd=<n> ] [ Vt=<n> ]
        [ Dlim=<n> ]
        [ gamma=<n> ] [ alpha=<n> ]
        [ henry.coeff=<n> | theta=<n> ]
        [ ( silicon | oxide | nitride | poly | gaas | gas ) ]
        [ ( /silicon | /oxide | /nitride | /poly | /gaas | /gas ) ]
        [ diff.0=<n> ] [ diff.e=<n> ] [ seg.0=<n> ] [ seg.E=<n> ]
        [ trn.0=<n> ] [ trn.E=<n> ]
        [ initial=<n> ] [ spread=<n> | mask.edge=<n> ]
        [ erf.q=<n> ] [ erf.delta=<n> ] [ erf.lbb=<n> ] [ erf.h=<n> ]
        [ nit.thick=<n> ]
```

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 parameterized error-function model from the literature; 3) a model where oxidant diffuses and the oxide boundaries move 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 [1]. 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˚ 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 μm/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 [2]. **thinox.0** is the prefactor in μm/min, **thinox.e** is the activation energy in eV, and **thinox.l** is the characteristic length in μm.

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 specified 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 specified 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 [3].

$$B/A = (B/A)_0[1 + K([V]/[V]_0 - 1)] \quad \text{(EQ 48)}$$

$[V]$ 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 (see vacancies).

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/cm^2, 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 | gaas | gas

What material 1 is.

/silicon | /oxide | /nitride | /poly | /gaas | /gas

What material 2 is.

henry.coeff | theta

Henry's coefficient is the solubility of oxidant in material 1 at one atmosphere, in cm^{-3}. **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 cm^2/sec, **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 (μm)), eox (the pad oxide thickness (μm)), Tox (the oxidation temperature (Kelvin)), and en (the nitride thickness (μm)). The published expression can be found in the models file. Specifying “erf.llbb=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 command, and the oxide thickness is computing by adding the total oxide grown to the initial oxide thickness specified on the oxide command. 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.

References

1. N. Guillemot, G. Pananakakis and P. Chenevier, “A New Analytical Model of the ``Bird's Beak",” *IEEE Transactions on Electron Devices*, **ED-34**, 1987.
2. H.Z. Massoud, J.D. Plummer and E.A. Irene, “Thermal Oxidation of Silicon in Dry Oxygen Growth Rate Enhancement in the Thin Regime,” *J. Electrochem. Soc.*, **132**, p. 2685, 1985.
3. C.P. Ho and J.D. Plummer, “Si/SiO2 Interface Oxidation Kinetics: A Physical Model for the Influence of High Substrate Doping Levels, Parts I and II,” *J. Electrochem. Soc.*, 126, p. 1523, 1979.

COMMAND PHOSPHORUS

Set the coefficients of phosphorus kinetics.

SYNOPSIS

```
phosphorus
        ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
        [ Dix.0=<n> ] [ Dix.E=<n> ]
        [ Dim.0=<n> ] [ Dim.E=<n> ]
        [ Dimm.0=<n> ] [ Dimm.E=<n> ]
        [ Dvx.0=<n> ] [ Dvx.E=<n> ]
        [ Dvm.0=<n> ] [ Dvm.E=<n> ]
        [ Dvmm.0=<n> ] [ Dvmm.E=<n> ]
        [ implanted ] [ grown.in ]
        [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
        [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
                /gaas ) ]
        [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
        [ ( donor | acceptor ) ]
```

DESCRIPTION

This statement allows the user to specify values for coefficients of phosphorus diffusion and segregation. The diffusion equation for phosphorus is [1,2]:

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_T \frac{C_V}{C_V^*} \nabla \ln\left(C_T \frac{C_V}{C_V^*} \frac{n}{n_i}\right) + D_I C_T \frac{C_I}{C_I^*} \nabla \ln\left(C_T \frac{C_I}{C_I^*} \frac{n}{n_i}\right)\right] \quad \textbf{(EQ 49)}$$

where C_T is the total chemical concentration of phosphorus, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium 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. The diffusivities are given by:

$$D_V = D_{VX} + D_{VM}\frac{n}{n_i} + D_{VMM}\left(\frac{n}{n_i}\right)^2 \quad \text{(EQ 50)}$$

$$D_I = D_{IX} + D_{IM}\frac{n}{n_i} + D_{IMM}\left(\frac{n}{n_i}\right)^2 \quad \text{(EQ 51)}$$

D_{IX}, D_{IM}, D_{IMM}, D_{VX}, D_{VM}, and D_{VMM} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1 - \frac{C_2}{M_{12}}\right) \quad \text{(EQ 52)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of the phosphorus diffusing with interstitial neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. **Dix.0** defaults to 3.85 cm^2/sec in silicon, and **Dix.E** defaults to 3.66 eV in silicon [3]. D_X is calculated using a standard Arrhenius relationship.

Dim.0, Dim.E

These floating point parameters allow the specification of the phosphorus diffusing with singly negative interstitial defects. **Dim.0** is the pre-exponential constant and **Dim.E** is the activation energy. **Dim.0** defaults to 4.44 cm^2/sec in silicon, and **Dim.E** defaults to 4.00 eV in silicon [3]. D_M is calculated using a standard Arrhenius relationship.

Dimm.0, Dimm.E

These floating point parameters allow the specification of the phosphorus diffusing with doubly negative interstitial defects. **Dimm.0** is the pre-exponential constant and **Dimm.E** is the activation energy. **Dimm.0** defaults to 44.2 cm^2/sec in silicon, and **Dim.E** defaults to 4.37 eV in silicon [3]. D_{MM} is calculated using a standard Arrhenius relationship.

Dvx.0, Dvx.E

These floating point parameters allow the specification of the phosphorus diffusing with neutral vacancy defects. **Dvx.0** is the pre-exponential constant and **Dvx.E** is the activation energy. Both of these are 0.0, since phosphorus is an interstitial diffuser [4].

Dvm.0, Dvm.E

These floating point parameters allow the specification of the phosphorus diffusing with singly negative vacancy defects. **Dvm.0** is the pre-exponential constant and **Dvm.E** is the activation energy. Both of these are 0.0, since phosphorus is an interstitial diffuser [4].

Dvmm.0, Dvmm.E

These floating point parameters allow the specification of the phosphorus diffusing with doubly negative vacancy defects. **Dvmm.0** is the pre-exponential constant and **Dvmm.E** is the activation energy. Both of these are 0.0, since phosphorus is an interstitial diffuser [4].

implanted, grown.in

Specifies whether the **Dix**, **Dim**, **Dimm**, **Dvx**, **Dvm**, **Dvmm**, or **Fi** coefficients apply to implanted or grown-in phosphorus. If neither is specified then a specified parameter applies to both.

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 [5]:

ss.temp	ss.conc
800˚C	2.79×10^{20}cm^{-3}
900˚C	3.16×10^{20}cm^{-3}
1000˚C	3.40×10^{20}cm^{-3}
1100˚C	3.80×10^{20}cm^{-3}

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

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. M_{12} is calculated using a standard Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. T_r is calculated using a standard Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default phosphorus is a donor.

EXAMPLES

```
phosphorus silicon Dix.0=3.85 Dix.E=3.65
```

This command changes the neutral defect diffusivity in silicon.

```
phosphorus silicon /oxide Seg.0=30.0 Trn.0=1.66e-7
Seg.E=0.0
```

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. M.E. Law and J.R. Pfiester, “Low Temperature Annealing of Arsenic/ Phosphorus Junctions,” *IEEE Trans. on Elec. Dev.*, **38**(2), p. 278, 1991.
2. D. Mathiot and J.C. Pfister, “Dopant Diffusion in Silicon: A consistent view involving nonequilibrium defects,” *J. Appl. Phys.*, **55**(10), p. 3518, 1984.
3. R.B. Fair, “Concentration Profiles of Diffused Dopants in Silicon,” Impurity Doping Processes in Silicon, Yang ed., 1981, North-Holland, New York.
4. 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.*, **46**(8), p. 784, 1985.
5. F.A. Trumbore, “Solid Solubilities of Impurity Elements in Germanium and Silicon,” Bell System Tech Journal, 1960.

SEE ALSO

The **antimony**, **arsenic**, **boron**, **interstitial**, and **vacancy** statements.

COMMAND SELENIUM

Set the coefficients of selenium kinetics.

SYNOPSIS

```
selenium
        ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
        [ Dix.0=<n> ] [ Dix.E=<n> ] [ Dim.0=<n> ] [ Dim.E=<n> ]
        [ Dimm.0=<n> ] [ Dimm.E=<n> ] [ Fi = <n> ]
        [ implanted ] [ grown.in ]
        [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
        [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
                /gaas ) ]
        [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
        [ ( donor | acceptor ) ]
```

DESCRIPTION

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

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_A \frac{C_V}{C_V^*}\nabla\ln\left(C_A \frac{C_V}{C_V^*}\frac{n}{n_i}\right) + D_I C_A \frac{C_I}{C_I^*}\nabla\ln\left(C_A \frac{C_I}{C_I^*}\frac{n}{n_i}\right)\right] \quad \text{(EQ 53)}$$

where C_T and C_A are the total chemical and active concentrations of selenium, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium 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. The diffusivities are given by:

$$D_V = (1 - F_i)\left(D_X + D_M\left(\frac{n}{n_i}\right) + D_{MM}\left(\frac{n}{n_i}\right)^2\right) \qquad \text{(EQ 54)}$$

$$D_I = F_i\left(D_X + D_M\left(\frac{n}{n_i}\right) + D_{MM}\left(\frac{n}{n_i}\right)^2\right) \qquad \text{(EQ 55)}$$

D_X, D_M, and D_{MM} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1 \frac{C_2}{M_{12}}\right) \qquad \text{(EQ 56)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of the selenium diffusing with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. For implanted selenium in gallium arsenide **Dix.0** defaults to 1.8×10^{-8} cm^2/sec and **Dix.E** defaults to 1.6 eV. For grown-in selenium in gallium arsenide **Dix.0** defaults to 0.0 cm^2/sec and **Dix.E** defaults to 0.0 eV [1,2,3]. D_X is calculated using a standard Arrhenius relationship.

Dim.0, Dim.E

These floating point parameters allow the specification of the selenium diffusing with singly negative defects. **Dim.0** is the pre-exponential constant and **Dim.E** is the activation energy. **Dim.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dim.E** defaults to 0.0 eV in gallium arsenide. D_M is calculated using a standard Arrhenius relationship.

Dimm.0, Dimm.E

These floating point parameters allow the specification of the selenium diffusing with doubly negative defects. **Dimm.0** is the pre-exponential constant and **Dimm.E** is the activation energy. For implanted selenium in gallium arsenide **Dimm.0** defaults to 0.0 cm^2/sec and **Dimm.E** defaults to 0.0 eV. For grown-in selenium in gallium arsenide **Dimm.0** defaults to 1.2 cm^2/sec and **Dimm.E** defaults to 3.6 eV [1,2,3]. D_{MM} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether selenium diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 0.0.

implanted, grown.in

Specifies whether the **Dix**, **Dim**, **Dimm**, or **Fi** coefficients apply to implanted or grown-in selenium. If neither is specified then a specified parameter applies to both.

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

ss.temp	ss.conc
700˚C	1.0×10^{19}cm^{-3}
1250˚C	1.0×10^{19}cm^{-3}

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

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. M_{12} is calculated using a standard Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. T_r is calculated using a standard Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default, selenium is an donor.

EXAMPLES

```
selenium gaas implanted Dimm.0=2.1e-3 Dimm.E=3.1
```

This command changes the doubly negative defect diffusivity for implanted selenium in silicon.

selenium gaas /nitride Seg.0=30.0 Trn.0=1.66e-7 Seg.E=0.0

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be 30.0 times the nitride concentration in equilibrium.

BUGS

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

REFERENCES

1. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1990.
2. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1991.
3. M.D. Deal, S.E. Hansen, and T.W. Sigmon, “SUPREM 3.5 – Process Modeling of GaAs Integrated Circuit Technology,” *IEEE Trans. CAD,* **9**, p. 939, 1989.

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon, generic**, **germanium**, **interstitial**, **magnesium**, **phosphorus**, **isilicon**, **tin**, **vacancy**, and **zinc** statements.

COMMAND iSILICON

Set the coefficients of silicon impurity kinetics.

SYNOPSIS

```
isilicon
      ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
      [ Dix.0=<n> ] [ Dix.E=<n> ] [ Dim.0=<n> ] [ Dim.E=<n> ]
      [ Dimm.0=<n> ] [ Dimm.E=<n> ] [ Fi = <n> ]
      [ implanted ] [ grown.in ]
      [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
      [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
            /gaas ) ]
      [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
      [ ( donor | acceptor ) ]
```

DESCRIPTION

This statement allows the user to specify values for coefficients of silicon impurity diffusion and segregation. The diffusion equation for silicon impurities are:

$$\frac{\partial C_T}{\partial t} = \nabla \left[D_V C_A \frac{C_V}{C_V^*} \nabla \ln\left(C_A \frac{C_V}{C_V^*} \frac{n}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \ln\left(C_A \frac{C_I}{C_I^*} \frac{n}{n_i} \right) \right] \quad \textbf{(EQ 57)}$$

where C_T and C_A are the total chemical and active concentrations of silicon, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium 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. The diffusivities are given by:

$$D_V = (1 - F_i)\left(D_X + D_M\left(\frac{n}{n_i}\right) + D_{MM}\left(\frac{n}{n_i}\right)^2\right) \quad \textbf{(EQ 58)}$$

$$D_I = F_i\left(D_X + D_M\left(\frac{n}{n_i}\right) + D_{MM}\left(\frac{n}{n_i}\right)^2\right) \quad \textbf{(EQ 59)}$$

D_X, D_M, and D_{MM} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1 \frac{C_2}{M_{12}}\right) \quad \textbf{(EQ 60)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of the silicon diffusing with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. For implanted silicon in gallium arsenide

Dix.0 defaults to 1.8×10^{-9} cm^2/sec and **Dix.E** defaults to 1.6 eV. For grown-in silicon in gallium arsenide **Dix.0** defaults to 0.0 cm^2/sec and **Dix.E** defaults to 0.0 eV [1,2,3]. D_X is calculated using a standard Arrhenius relationship.

Dim.0, Dim.E

These floating point parameters allow the specification of the silicon diffusing with singly negative defects. **Dim.0** is the pre-exponential constant and **Dim.E** is the activation energy. **Dim.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dim.E** defaults to 0.0 eV in gallium arsenide. D_M is calculated using a standard Arrhenius relationship.

Dimm.0, Dimm.E

These floating point parameters allow the specification of the silicon diffusing with doubly negative defects. **Dimm.0** is the pre-exponential constant and **Dimm.E** is the activation energy. For implanted silicon in gallium arsenide **Dimm.0** defaults to 0.0 cm^2/sec and **Dimm.E** defaults to 0.0 eV. For grown-in silicon in gallium arsenide **Dimm.0** defaults to 33.0 cm^2/sec and **Dimm.E** defaults to 3.9 eV [1,2,3]. D_{MM} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether silicon diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 0.0.

implanted, grown.in

Specifies whether the **Dix**, **Dim**, **Dimm**, or **Fi** coefficients apply to implanted or grown-in silicon. If neither is specified then a specified parameter applies to both.

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

ss.temp	ss.conc
700˚C	1.0×10^{19} cm^{-3}
1250˚C	1.0×10^{19} cm^{-3}

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

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. M_{12} is calculated using a standard Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. T_r is calculated using a standard Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default, silicon is an donor.

EXAMPLES

```
isilicon gaas implanted Dimm.0=2.1e-3 Dimm.E=3.1
```

This command changes the doubly negative defect diffusivity for implanted silicon in gallium arsenide.

```
isilicon gaas /nitride Seg.0=30.0 Trn.0=1.66e-7 Seg.E=0.0
```

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be 30.0 times the nitride concentration in equilibrium.

BUGS

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

REFERENCES

1. J.D. Plummer et al., "Process Simulators for Si VLSI and High Speed GaAs Devices," *Stanford University Technical Report,* 1990.
2. J.D. Plummer et al., "Process Simulators for Si VLSI and High Speed GaAs Devices," *Stanford University Technical Report*, 1991.
3. E.L. Allen, J.J. Murray, M.D. Deal, J.D. Plummer, K.S.Jones, and W.S. Robert, "Diffusion of Ion-Implanted Group IV Dopants in GaAs," *J. Electrochem. Soc.,* **138**, p. 3440, 1991.
4. J.J. Murray, M.D. Deal, E.A. Allen, D.D. Stevenson, and S. Nozaki, "Modeling Silicon Diffusion in GaAs Using Well Defined Silicon Doped Molecular Beam Epitaxy Structures," *J. Electrochem. Soc.,* **139**, p. 2037, 1992.

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **generic**, **germanium**, **interstitial**, **magnesium**, **phosphorus**, **selenium**, **tin**, **vacancy**, and **zinc** statements.

COMMAND TIN

Set the coefficients of tin kinetics.

SYNOPSIS

```
tin
      ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
      [ Dix.0=<n> ] [ Dix.E=<n> ] [ Dim.0=<n> ] [ Dim.E=<n> ]
      [ Dimm.0=<n> ] [ Dimm.E=<n> ] [ Fi = <n> ]
      [ implanted ] [ grown.in ]
      [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
      [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
            /gaas ) ]
      [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
      [ ( donor | acceptor ) ]
```

DESCRIPTION

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

$$\frac{\partial C_T}{\partial t} = \nabla \left[D_V C_A \frac{C_V}{C_V^*} \nabla \ln\left(C_A \frac{C_V}{C_V^*} \frac{n}{n_i} \right) + D_I C_A \frac{C_I}{C_I^*} \nabla \ln\left(C_A \frac{C_I}{C_I^*} \frac{n}{n_i} \right) \right] \quad \text{(EQ 61)}$$

where C_T and C_A are the total chemical and active concentrations of tin, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium 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. The diffusivities are given by:

$$D_V = (1 - F_i)\left(D_X + D_M\left(\frac{n}{n_i}\right) + D_{MM}\left(\frac{n}{n_i}\right)^2\right) \quad \textbf{(EQ 62)}$$

$$D_I = F_i\left(D_X + D_M\left(\frac{n}{n_i}\right) + D_{MM}\left(\frac{n}{n_i}\right)^2\right) \quad \textbf{(EQ 63)}$$

D_X, D_M, and D_{MM} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1 \frac{C_2}{M_{12}}\right) \quad \textbf{(EQ 64)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of the tin diffusing with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. **Dix.0** defaults to 0.0 cm^2/sec in gallium arsenide, and

Dix.E defaults to 0.0 eV in gallium arsenide. D_X is calculated using a standard Arrhenius relationship.

Dim.0, Dim.E

These floating point parameters allow the specification of the tin diffusing with singly negative defects. **Dim.0** is the pre-exponential constant and **Dim.E** is the activation energy. **Dim.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dim.E** defaults to 0.0 eV in gallium arsenide. D_M is calculated using a standard Arrhenius relationship.

Dimm.0, Dimm.E

These floating point parameters allow the specification of the tin diffusing with doubly negative defects. **Dimm.0** is the pre-exponential constant and **Dimm.E** is the activation energy. **Dimm.0** defaults to 96.0 cm^2/sec in gallium arsenide, and **Dimm.E** defaults to 3.9 eV in gallium arsenide [1,2,3]. D_{MM} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether tin diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 0.0.

implanted, grown.in

Specifies whether the **Dix**, **Dim**, **Dimm**, or **Fi** coefficients apply to implanted or grown-in tin. If neither is specified then a specified parameter applies to both.

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

ss.temp	ss.conc
700˚C	1.0×10^{19}cm^{-3}
1250˚C	1.0×10^{19}cm^{-3}

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

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. M_{12} is calculated using a standard Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. T_r is calculated using a standard Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default, tin is an donor.

EXAMPLES

```
tin gaas implanted Dimm.0=2.1e-3 Dimm.E=3.1
```

This command changes the doubly negative defect diffusivity for implanted tin in silicon.

```
tin gaas /nitride Seg.0=30.0 Trn.0=1.66e-7 Seg.E=0.0
```

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be 30.0 times the nitride concentration in equilibrium.

BUGS

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

REFERENCES

1. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report*, 1990.
2. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report,* 1991.
3. E.L. Allen, J.J. Murray, M.D. Deal, J.D. Plummer, K.S.Jones, and W.S. Robert, “Diffusion of Ion-Implanted Group IV Dopants in GaAs,” *J. Electrochem. Soc.,* **138**, p. 3440, 1991.

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **generic**, **germanium**, **interstitial**, **magnesium**, **phosphorus**, **selenium**, **isilicon**, **vacancy**, and **zinc** statements.

COMMAND 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> ]
```

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:

$$\frac{\partial C_{\mathrm{ET}}}{\partial t} = -K_T\left[C_{\mathrm{ET}}C_I - \frac{e^*}{1-e^*}C_I^*(C_T - C_{\mathrm{ET}})\right] \quad \textbf{(EQ 65)}$$

where C_T is the total trap 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 a simple interaction of an interstitial and a trap. To achieve balance in equilibrium, 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, aluminum, photores

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 $2\times10^{18}\,\mathrm{cm}^{-3}$. 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^*. The default for these parameters is $2.39\times10^{5}\,\mathrm{cm}^{-3}$ and 1.57 eV for the pre-exponential and activation energy for silicon.

EXAMPLES

```
trap silicon total=2.0e18 frac.0=0.5 frac.E=0.0 enable
```

This statement turns on interstitial traps and sets the total to 2×10^{18} 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 date at any time. Further, experimental verification is difficult. 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, Los Angeles, p. 522, 1986.

SEE ALSO

The **interstitial** and **vacancy** statements.

COMMAND VACANCY

Set coefficients of vacancy kinetics.

SYNOPSIS

```
vacancy
    ( silicon | oxide | poly | oxynitr | nitride | gaas | 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> ]
    [ tneg.0=<n> ] [ tneg.E=<n> ]
    [ pos.0=<n> ] [ pos.E=<n> ] [ dpos.0=<n> ] [ dpos.E=<n> ]
    [ tpos.0=<n> ] [ tpos.E=<n> ]
    [ ( /silicon | /oxide | /poly | /oxynitr | /nitride | /gaas |
        /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> ]
```

DESCRIPTION

This statement allows the user to specify values for coefficients of the 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 and gallium arsenide and its interfaces. Polysilicon has not been characterized as extensively, and its parameters default to those for silicon.

The vacancies obey a complex continuity equation which can be written [1]:

$$\frac{\partial C_V}{\partial t} = \nabla\left(-J_V - \sum_{\text{imp}} J_{AV}\right) - R \qquad \text{(EQ 66)}$$

where C_V is the vacancy concentration, 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**) the fluxes from the dopants, J_{AV}, are ignored. Only in the **full.cpl** model are they 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 [1].

The pair fluxes are the vacancy portions of the flux as described in each of the models for the impurities (see the various impurity statements). The vacancy flux can be written [1]:

$$-J_V = D_V C_V^* \nabla \frac{C_V}{C_V^*} \qquad \text{(EQ 67)}$$

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 interstitials and vacancies. This can be expressed:

$$R = K_R (C_I C_V - C_I^* C_V^*) \qquad \textbf{(EQ 68)}$$

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

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

$$J_V \vec{n} + K_V (C_V - C_V^*) = g \qquad \textbf{(EQ 69)}$$

where $\vec{n}$ is the surface normal, K_V 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.

silicon, oxide, poly, oxynitr, nitride, gaas, 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/sec. The default values are 6.34×10^3 cm^2/sec and 3.29 eV [5] for silicon, and 1.0×10^{-12} cm^2/sec and 0.0 eV for GaAs.

Kr.0, Kr.E

These floating point parameters allow the specification of the bulk recombination rate in cm^3/sec. The default is 1.4 cm^3/sec and 3.99 eV [6] for silicon, and 1.0×10^{-18}cm^3/sec and 0.0 eV for GaAs.

Cstar.0, Cstar.E

These parameters allow the specification of the total equilibrium concentration of vacancies in intrinsically doped conditions. The default values are 4.77×10^{18}cm^{-3} and 0.713 eV [5] for silicon, and 4.77×10^{14}cm^{-3} and 0.0 eV for GaAs.

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

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

$$C_I^{=} = \frac{C_{\text{Star}}\,\text{dneg}}{\text{neu} + \text{neg} + \text{dneg} + \text{tneg} + \text{pos} + \text{dpos} + \text{tpos}} \qquad \textbf{(EQ 70)}$$

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

$$C_I^{*} = C_{\text{Star}}\frac{\text{neu} + \text{neg}\,\frac{n}{n_i} + \text{dneg}\,\left(\frac{n}{n_i}\right)^2 + \text{tneg}\,\left(\frac{n}{n_i}\right)^3 + \text{pos}\,\frac{p}{n_i} + \text{dpos}\,\left(\frac{p}{n_i}\right)^2 + \text{tpos}\,\left(\frac{p}{n_i}\right)^3}{\text{neu} + \text{neg} + \text{dneg} + \text{tneg} + \text{pos} + \text{dpos} + \text{tpos}} \qquad \textbf{(EQ 71)}$$

There is very little data for the vacancy charge states; it is matter of ongoing research. The default values for silicon are chosen based on an analysis and measurements of Watkins [7]. See Table 4 in the section "Adding SUPREM 3.5's GaAs Models and Parameters to SUPREM-IV" for GaAs values.

/silicon, /oxide, /poly, /oxynitr, /nitride, /gas, /gaas

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:

$$K_V = K_{\text{Surf}}\left(K_{\text{rat}}\left(\frac{v_{\text{ox}}}{B/A}\right)^{K_{\text{pow}}} + 1\right) \qquad \textbf{(EQ 72)}$$

where v_{ox} is the interface velocity, and B/A is the Deal Grove oxide growth coefficient. (see **oxide**). 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) K_{Surf} 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:

$$g = \theta V_{\mathrm{mole}} v_{\mathrm{ox}} \left(\frac{v_{\mathrm{ox}}}{B/A}\right)^{G_{\mathrm{pow}}} \quad \textbf{(EQ 73)}$$

Theta is the fraction of silicon atoms consumed during growth that are re-injected as vacancies. V_{mole} is the lattice density of the consumed material. Once again, this model only makes sense 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:

$$g = A(t + t_0)^{T_{\mathrm{pow}}} \quad \textbf{(EQ 74)}$$

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

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

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

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 sil /oxi 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 3.5×10^{-3} cm/sec.

```
inter sil neu.0=1.0 neg.0=1.0 pos.0=0.0 dneg.0=0.0
        dpos.0=0.0
inter sil 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 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 date at any time. Further, experimental verification is difficult. 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. M.E. Law and J.R. Pfiester, "Low Temperature Annealing of Arsenic/Phosphorus Junctions," *IEEE Trans. on Elec. Dev.*, 38(2), p. 278, 1991.
2. P.B. Griffin and J.D. Plummer, "Process Physics Determining 2-D Impurity Profiles in VLSI Devices," International Electron Devices Meeting, Los Angeles, p. 522, 1986.
3. W. Shockley and J.T. Last, "Statistics of the Charge Distribution for a Localized Flaw in a Semiconductor," *Phys. Rev.*, **107**(2), p. 392, 1957.
4. S.M. Hu, "On Interstitial and Vacancy Concentrations in Presence of Injection," *J. Appl. Phys.*, **57**, p. 1069, 1985.
5. T.Y. Tan and U. Gösele, "Point Defects, Diffusion Processes, and Swirl Defect Formation in Silicon," *J. Appl. Phys.*, **37**(1), p. 1, 1985.
6. M.E. Law, "Parameters for Point Defect Diffusion and Recombination," *IEEE Trans. on CAD*, Accepted for Publication, Sept., 1991.
7. G.D. Watkins, "EPR Studies of the Lattice Vacancy and Low Temperature Damage Processes in Silicon," Lattice Defects in Semiconductors 1974, Huntley ed., 1975, Inst. Phys. Conf. Ser. 23, London.
8. G.A. Baraff and M. Schluter, "Electronic Structure, Total Energies, and Abundances of the Elementary Point Defects in GaAs," *Phys. Rev. Lett.*, **55**, 1327 (1985).

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **generic**, **germanium**, **interstitial**, **magnesium**, **phosphorus**, **selenium**, **isilicon**, **tin**, and **zinc** statements.

COMMAND ZINC

Set the coefficients of zinc kinetics.

SYNOPSIS

```
zinc
        ( silicon | oxide | oxynit | nitride | gas | poly | gaas )
        [ Dix.0=<n> ] [ Dix.E=<n> ]
        [ Dip.0=<n> ] [ Dip.E=<n> ] [ Dipp.0=<n> ] [ Dipp.E=<n> ]
        [ Fi = <n> ]
        [ implanted ] [ grown.in ]
        [ ss.clear ] [ ss.temp=<n> ] [ ss.conc=<n> ]
        [ ( /silicon | /oxide | /oxynitr | /nitride | /gas | /poly |
                /gaas ) ]
        [ Seg.0=<n> ] [ Seg.E=<n> ] [ Trn.0=<n> ] [ Trn.E=<n> ]
        [ ( donor | acceptor ) ]
```

DESCRIPTION

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

$$\frac{\partial C_T}{\partial t} = \nabla\left[D_V C_A \frac{C_V}{C_V^*}\nabla\ln\left(C_A \frac{C_V}{C_V^*}\frac{p}{n_i}\right) + D_I C_A \frac{C_I}{C_I^*}\nabla\ln\left(C_A \frac{C_I}{C_I^*}\frac{p}{n_i}\right)\right] \quad \text{(EQ 75)}$$

where C_T and C_A are the total chemical and active concentrations of zinc, C_A is the total electrically active concentration of zinc, C_V and C_I are the vacancy and interstitial concentrations, the superscript * refers to the equilibrium value, 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. The diffusivities are given by:

$$D_V = (1 - F_i)\left(D_X + D_p\left(\frac{p}{n_i}\right) + D_{PP}\left(\frac{p}{n_i}\right)^2\right) \qquad \text{(EQ 76)}$$

$$D_I = F_i\left(D_X + D_p\left(\frac{p}{n_i}\right) + D_{PP}\left(\frac{p}{n_i}\right)^2\right) \qquad \text{(EQ 77)}$$

D_X , D_P, and D_{PP} are described in greater detail below.

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

$$\text{flux} = T_r\left(C_1\frac{C_2}{M_{12}}\right) \qquad \text{(EQ 78)}$$

where C_1 and C_2 are the concentrations in material 1 and 2 respectively, and the M_{12} and T_r terms are computed using expressions shown below with the parameters of the models.

silicon, oxide, oxynitr, nitride, gas, poly, gaas

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 allow the specification of D_X , the zinc diffusivity with neutral defects. **Dix.0** is the pre-exponential constant and **Dix.E** is the activation energy. **Dix.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dix.E** defaults to 0.0 eV in gallium arsenide. D_X is calculated using a standard Arrhenius relationship.

Dip.0, Dip.E

These floating point parameters allow the specification of D_P, the beryllium diffusivity with singly positive defects. **Dip.0** is the pre-exponential constant and **Dip.E** is the activation energy. **Dip.0** defaults to 0.0 cm^2/sec in gallium arsenide, and **Dip.E** defaults to 0.0 eV in gallium arsenide. D_P is calculated using a standard Arrhenius relationship.

Dipp.0, Dipp.E

These floating point parameters allow the specification of D_{PP}, the zinc diffusivity with doubly positive defects. **Dipp.0** is the pre-exponential constant and **Dipp.E** is the activation energy. **Dipp.0** defaults to 1.2×10^2 cm^2/sec for implanted zinc in gallium arsenide and 15.6 cm^2/sec for grown-in zinc in gallium arsenide, and **Dipp.E** defaults to 3.9 eV for both in gallium arsenide[1,2]. D_{PP} is calculated using a standard Arrhenius relationship.

Fi

This parameter allows the specification of the fractional interstitialcy. This value indicates whether zinc diffuses through interaction with interstitials or vacancies. The value of **Fi** defaults to 1.0.

implanted, grown.in

Specifies whether the **Dix**, **Dip**, **Dipp**, or **Fi** coefficients apply to implanted or grown-in zinc. If neither is specified then a specified parameter applies to both.

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:

ss.temp	ss.conc
700˚C	$1.0 \times 10^{19} \mathrm{cm}^{-3}$
1250˚C	$1.0 \times 10^{19} \mathrm{cm}^{-3}$

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

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 segregation constant follows an Arrhenius relationship.

Trn.0, Trn.E

These parameters allow the specification of the transport velocity across the interface given. The units are in cm/sec. The transfer coefficient follows an Arrhenius relationship.

donor, acceptor

These parameters determine whether the impurity is to be treated as a donor or as an acceptor in a semiconductor material. These parameters are not presently material specific. By default, zinc is an acceptor.

EXAMPLES

zinc gaas implanted Dip.0=2.2e-6 Dip.E=1.76

This command changes the positive defect diffusivity for implanted zinc in gallium arsenide.

zinc gaas /nitride Seg.0=1126.0 Seg.E=0.91 Trn.0=1.66e-7

This command will change the segregation parameters at the gallium arsenide – silicon nitride interface. The gallium arsenide concentration will be half the nitride concentration in equilibrium at 1100˚C.

BUGS

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

REFERENCES

1. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report,* 1990.
2. J.D. Plummer et al., “Process Simulators for Si VLSI and High Speed GaAs Devices,” *Stanford University Technical Report,* 1991.

SEE ALSO

The **antimony**, **arsenic**, **beryllium**, **boron**, **carbon**, **generic**, **germanium**, **interstitial**, **magnesium**, **phosphorus**, **selenium**, **isilicon**, **tin**, and **vacancy** statements.

Examples

INTRODUCTION

This section contains several examples showing the use of SUPREM-IV.GS. The first nine examples are from the pre-gallium arsenide version of the program and show silicon processing exclusively. However, these examples are still useful for users interested in GaAs processing as they exploit many of the program’s features that are common to all aspects of semiconductor process simulation. The remaining examples concentrate on GaAs process simulation. Only those commands that are unique to GaAs simulation are discussed.

EXAMPLE 1 Boron Anneal – 1D

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
mode one.dim

#the vertical definition
line x loc = 0     spacing = 0.02 tag = top
line x loc = 0.50  spacing = 0.02
line x loc = 2.0   spacing = 0.25 tag=bottom

#the silicon wafer
region silicon xlo = top xhi = bottom

#set up the exposed surfaces
bound exposed  xlo = top  xhi = 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
select z=log10(boron)
plot.1d 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
select z=log10(bor)
plot.1d x.max=2.0 cle=f axi=f
```

The first lines in the input deck set some basic SUPREM-IV.GS options.

```
#some set stuff
set echo
option quiet
mode one.dim
```

The '**#**' character is the comment character for SUPREM-IV.GS. The entire line after the '**#**' is ignored. The **echo** flag is turned on. This instructs SUPREM-IV.GS 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 final command specifies that the simulation will be done in one dimension. It is therefore necessary to specify only vertical grid information.

The next section begins the definition of the mesh to be used for the simulation. The section

```
#the vertical definition
line x loc = 0     spacing = 0.02 tag = top
line x loc = 0.50  spacing = 0.02
line x loc = 2.0   spacing = 0.25 tag=bottom
```

describes the locations of the x lines in the mesh. SUPREM-IV.GS, in one dimensional mode, defines x to be the direction into the wafer. In two-dimensional mode, x is across the wafer top and y is the direction into the wafer. This simulation is to be done on a one dimensional problem, therefore lines are only specified in the x direction. The first statement places a mesh line at the top of wafer, x coordinate 0.0µm and tags the line as “top”. The spacing is set to 0.02µm. A line is placed 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 **line** statement fixes line locations in the mesh as well as the average spacing between lines.

The next two sections describe the device starting material and the surfaces which are exposed to gas.

```
#the silicon wafer
region silicon xlo = top xhi = bottom

#set up the exposed surfaces
bound exposed  xlo = top  xhi = 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 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 front and backsides of the wafer. Any gases on the **diffuse**, **deposit**, and **etch** statements are applied to the surface marked exposed. It is important to define the top of the wafer for SUPREM-IV.GS to correctly simulate these actions. Most mistakes are made by ignoring to define the exposed surface.

The next line informs SUPREM-IV.GS 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 $1\times10^{14}\text{cm}^{-3}$.

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
```

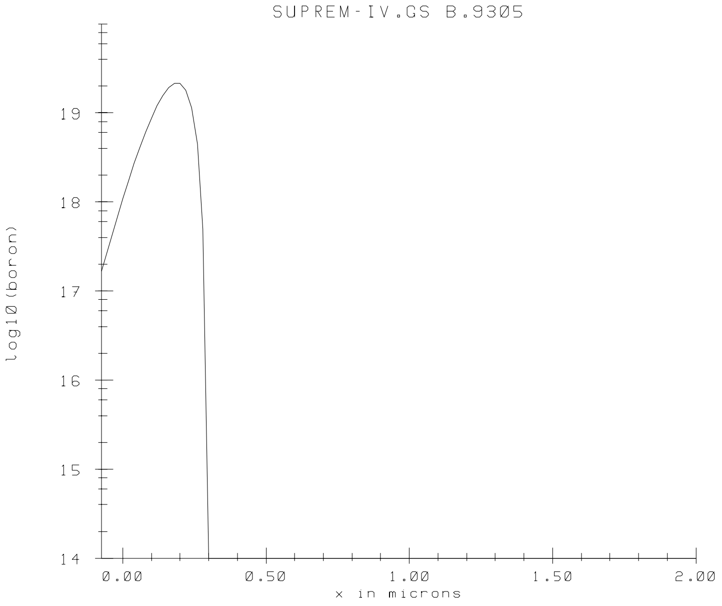
The implant is modeled with a Pearson-IV distribution. The energy and dose are $3\times10^{14}\text{cm}^{-2}$ and 70KeV respectively. This produces an abrupt boron profile with a junction near 0.3µm.

The next commands choose a plot variable and display it.

```
#plot the initial profile
select z=log10(boron)
plot.1d x.max=2.0 y.min=14.0 y.max=20.0
```

The plot will look like Figure 1. The **select** statement selects the plot variable to be the log base ten of the boron concentration. The **plot.1d** statement does not need to specify the location of the cross section in one dimension. The plot should extend between the minimum value of y through 2.0µm. Dimensions on the plot command 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, $\log_{10}$ of concentration.

FIGURE 1 Implanted Boron profile.



The **diffuse** command simulates the 30 minute, 1100˚C 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 -0.000000e+00
Solving       0 +      0.1 =      0.1,    100%, np 43
Solving      0.1 + 0.488986 = 0.588986, 488.986%, np 43
Solving 0.588986 +   2.9165 =  3.50548, 596.438%, np 43
Solving  3.50548 +  5.36005 =  8.86554, 183.784%, np 43
Solving  8.86554 +  8.54232 =  17.4079,  159.37%, np 43
Solving  17.4079 +  3.80138 =  21.2092, 44.5005%, np 43
Solving  21.2092 +   12.795 =  34.0042, 336.588%, np 43
Solving  34.0042 +  15.4141 =  49.4183,  120.47%, np 43
Solving  49.4183 +  22.2639 =  71.6822, 144.439%, np 43
Solving  71.6822 +  28.0168 =   99.699, 125.839%, np 43
Solving   99.699 +  26.4923 =  126.191, 94.5586%, np 43
Solving  126.191 +  41.1269 =  167.318, 155.241%, np 43
Solving  167.318 +   29.629 =  196.947, 72.0429%, np 43
Solving  196.947 +  56.4872 =  253.434, 190.649%, np 43
Solving  253.434 +  60.4027 =  313.837, 106.932%, np 43
Solving  313.837 +  92.0295 =  405.867,  152.36%, np 43
Solving  405.867 +  103.787 =  509.654, 112.776%, np 43
Solving  509.654 +  73.6803 =  583.334, 70.9916%, np 43
Solving  583.334 +  175.487 =  758.821, 238.174%, np 43
Solving  758.821 +  205.108 =   963.93, 116.879%, np 43
Solving   963.93 +   265.71 =  1229.64, 129.546%, np 43
Solving  1229.64 +  376.381 =  1606.02, 141.651%, np 43
Solving  1606.02 +  193.979 =     1800, 51.5381%, np 43
```

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, therefore, 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, 43 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.

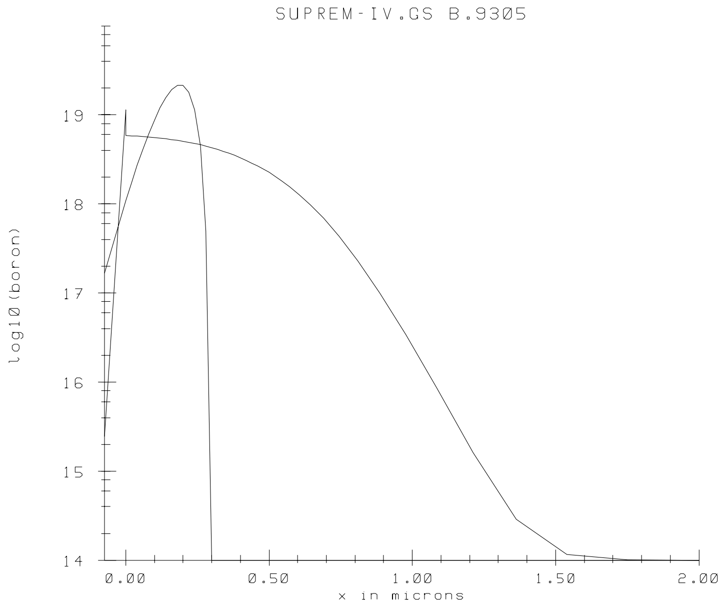
The following script will plot the final boron concentration.

```
#plot the final profile
select z=log10(bor)
plot.1d x.max=2.0 cle=f axi=f
```

The first statement picks the log base ten of the boron concentration to plot. The plot will be placed on the earlier plot. The “cle=f axi=f” instructs SUPREM-IV.GS to not clear and not compute new axes.

Figure 2 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.

FIGURE 2 Comparison of annealed and implanted boron profiles.



EXAMPLE 2 Boron OED – 1D

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
mode one.dim

#the vertical definition
line x loc = 0     spacing = 0.02 tag = top
line x loc = 1.50  spacing = 0.05
line x loc = 5.0   spacing = 0.5
line x loc = 400.0                tag=bottom

#the silicon wafer
region silicon xlo = top xhi = bottom

#set up the exposed surfaces
bound exposed  xlo = top  xhi = 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
```

```
select z=log10(boron)
plot.1d x.ma=2.0 y.mi=14.0 y.max=20.0

#the diffusion card
method init=1.0e-3 two.d
diffuse time=30 temp=1100 dry

#save the data
structure out=oed.str

#plot the final profile
select z=log10(bor)
plot.1d cle=f axi=f
```

This example is very similar to Example 1. It starts by setting command line echoing and by instructing SUPREM-IV.GS to be relatively quiet about the progress of computation. 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 x loc = 0     spacing = 0.02 tag = top
line x loc = 1.50  spacing = 0.05
line x loc = 5.0   spacing = 0.5
line x 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 set of commands choose a plot variable and display it. This plot will be identical to Figure 1 of Example 1.

The **diffuse** command 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 -0.000000e+00
Solving        0 +    0.001 =    0.001,    100%, np 80
Solving    0.001 +0.0028164 =0.0038164,281.644%, np 80
Solving0.0038164 +0.0176591 =0.0214756,627.002%, np 80
Solving0.0214756 + 0.168429 = 0.189905,953.779%, np 80
Solving 0.189905 + 0.813906 =  1.00381,483.233%, np 80
Solving  1.00381 +   2.6438 =  3.64761,324.828%, np 80
Solving  3.64761 +  6.03434 =  9.68195,228.245%, np 80
Solving  9.68195 +  9.78821 =  19.4702,162.208%, np 80
Solving  19.4702 +  14.1452 =  33.6153,144.512%, np 80
Solving  33.6153 +  19.4714 =  53.0868,137.654%, np 80
Solving  53.0868 +  26.1378 =  79.2245,134.236%, np 80
Solving  79.2245 +  35.1156 =   114.34,134.348%, np 80
Solving   114.34 +  38.3539 =  152.694,109.222%, np 80
Solving  152.694 +  44.1769 =  196.871,115.182%, np 80
Solving  196.871 +  63.5629 =  260.434,143.883%, np 80
Solving  260.434 +  84.4433 =  344.877, 132.85%, np 80
Solving  344.877 +    105.6 =  450.477,125.054%, np 80
Solving  450.477 +  78.7206 =  529.198, 74.546%, np 80
Solving  529.198 +  71.7795 =  600.977,91.1826%, np 80
Solving  600.977 +  121.337 =  722.315,169.042%, np 80
Solving  722.315 +   191.01 =  913.324, 157.42%, np 80
Solving  913.324 +  108.262 =  1021.59,56.6788%, np 80
Solving  1021.59 +  193.111 =   1214.7,178.374%, np 80
Solving   1214.7 +  350.262 =  1564.96,181.378%, np 80
Solving  1564.96 +  235.041 =     1800,67.1044%, np 81
```

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 **initialize** command or the structure command.

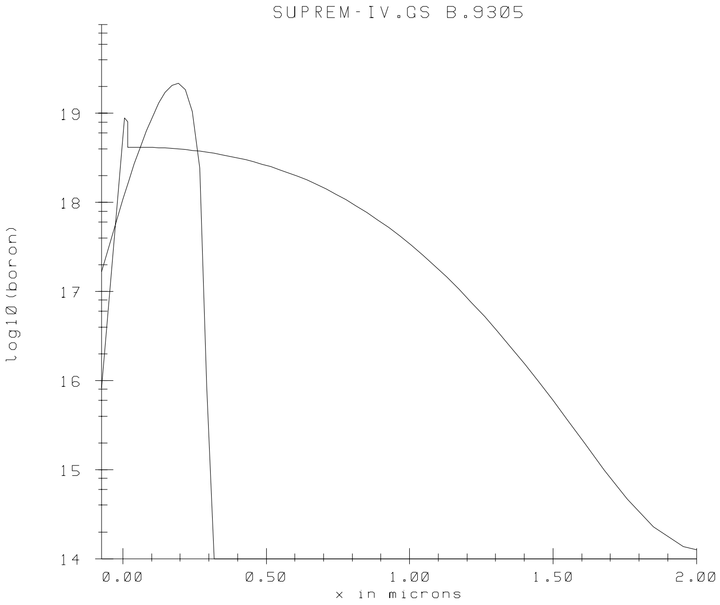
The following script will plot the final boron concentration.

```
#plot the final profile
select z=log10(bor)
plot.1d cle=f axi=f
```

The first command picks the log base ten of the boron concentration to plot. The plot will be placed on the earlier plot. The “cle=f axi=f” instructs SUPREM-IV.GS to not clear and not compute new axes.

The profile in Figure 1 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.

FIGURE 1 Final structure comparing the boron before and after anneal.



EXAMPLE 3 OED Time

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
mode one.dim

#the vertical definition
line x loc = 0     spacing = 0.02 tag = top
line x loc = 1.50  spacing = 0.05
line x loc = 5.0   spacing = 0.5
line x loc = 28.0  tag=bottom

#the silicon wafer
region silicon xlo = top xhi = bottom

#set up the exposed surfaces
bound exposed  xlo = top  xhi = top
bound backsid  xlo = bottom  xhi = bottom

#calculate the mesh
init

#the pad oxide
deposit oxide thick=0.02

#the diffusion card
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 tfci 0.0
%define tbci 0.0
%diffuse time=240 temp=1100 dry movie="
     select z=(inter/ci.star);
     %define newfci zfn(0.0,sil@oxi(0.0)+1.0e-6)
     %define newbci zfn(0.0,27.9)
     %define tfci ${tfci}+0.5*(${time}-${otim})*(${-
newfci}+${oldfci})
     %define tbci ${tbci}+0.5*(${time}-${otim})*(${-
newbci}+${oldbci})
     %define den ${time}+1.0e-6
     printf Data ${time}/60.0 (${tfci}/${den})
(${tbci}/${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.GS to be relatively quiet about the progress of computation. 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 x loc = 0     spacing = 0.02 tag = top
line x loc = 1.50  spacing = 0.05
line x loc = 5.0   spacing = 0.5
line x 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 commands 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 commands 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 tfci 0.0
%define tbci 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 **tfci** and **tbci** are the time averaged values of the scaled interstitial concentration at the front and back respectively.

The **diffuse** command 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.GS input at the end of every time step. This allows a time history to be developed.

```
     select z=(inter/ci.star);
     %define newfci zfn(0.0,sil@oxi(0.0)+1.0e-6)
     %define newbci zfn(0.0,27.99)
     %define tfci ${tfci}+0.5*(${time}-${otim})*(${-
newfci}+${oldfci})
     %define tbci ${tbci}+0.5*(${time}-${otim})*(${-
newbci}+${oldbci})
```

```
      %define den ${time}+1.0e-6
      printf Data ${time}/60.0 (${tfci}/${den})
 (${tbci}/${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 = 0.0µm and y = 1Å 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 trapezoidal 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** command should produce output of the form:

```
estimated first time step 8.740542e-10
Data 0 0 0
Solving        0 +    0.001 =    0.001,    100%, np 73
Data 1.66667e-05 1.14116 0.999001
Solving    0.001 +0.0021718 =0.0031718,217.181%, np 73
Data 5.28635e-05 1.23489 0.999685
Solving0.0031718 +0.0105442 = 0.013716,485.503%, np 73
Data 0.0002286 1.26412 0.999927
Solving 0.013716 +0.0857607 =0.0994767,813.344%, np 73
Data 0.00165794 1.27887 0.99999
Solving0.0994767 + 0.380333 =  0.47981,443.482%, np 73
Data 0.00799683 1.31034 0.999998
Solving  0.47981 +  1.11267 =  1.59248,292.553%, np 73
Data 0.0265413 1.39119 1
Solving  1.59248 +  2.71156 =  4.30404,243.697%, np 73
Data 0.071734 1.5448 1
```

```
Solving  4.30404 +  7.50309 =  11.8071,276.707%, np 73
Data 0.196785 1.81401 1
Solving  11.8071 +  23.0711 =  34.8782,307.488%, np 73
Data 0.581303 2.21682 1
Solving  34.8782 +  52.5225 =  87.4007,227.655%, np 73
Data 1.45668 2.59747 1
Solving  87.4007 +  116.309 =   203.71,221.446%, np 73
Data 3.39517 2.8976 1
Solving   203.71 +  264.827 =  468.537,227.693%, np 73
Data 7.80895 3.09822 1.00005
Solving  468.537 +  629.713 =  1098.25,237.783%, np 73
Data 18.3042 3.19492 1.00143
Solving  1098.25 +  1476.13 =  2574.38,234.413%, np 72
Data 42.9063 3.19724 1.02452
Solving  2574.38 +  1786.28 =  4360.66,121.011%, np 71
Data 72.6777 3.13466 1.09187
Solving  4360.66 +  2319.41 =  6680.07,129.846%, np 71
Data 94.745 3.07212 1.16452
Solving   5684.7 +  2813.18 =  8497.88,212.469%, np 71
Data 141.631 2.96848 1.34316
Solving  8497.88 +  3114.92 =  11612.8,110.726%, np 70
Data 193.547 2.89165 1.53193
Solving  11612.8 +  2787.21 =    14400,89.4793%, np 69
Data 240 2.83454 1.67415
```

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. The output can be redirected to a file and then plotted using any number of x-y plot programs.

EXAMPLE 4 Boron OED – 2D

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

#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 = 1.5   spacing = 0.05
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

#deposit the nitride mask
deposit nitride thick=0.05
etch nitride right p1.x=0.0 p2.x=0.0

#the boron implant
implant boron dose=3e14 energy=40 pearson

#the diffusion card
method 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.GS 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 0.05 µm 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 **diffuse** command 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
```

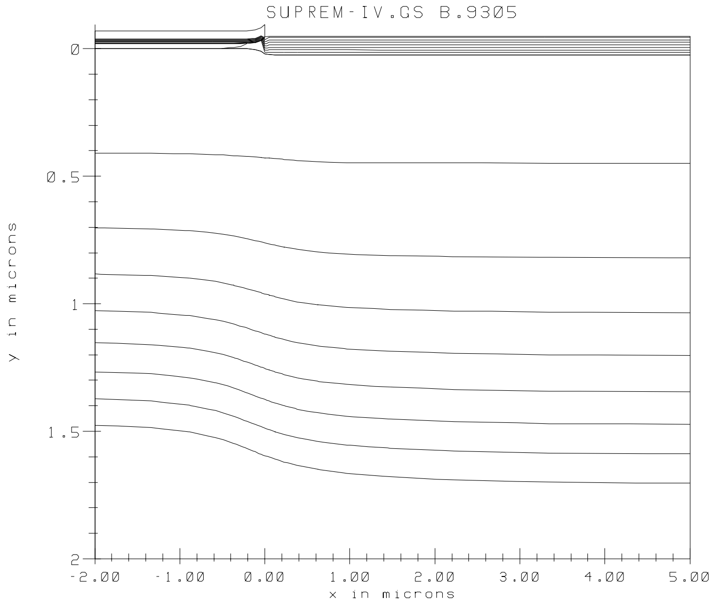
This will read in the file and initialize the data structures of the program.

The final lines will plot the final boron concentration, in two dimensions

```
#plot the final profile
select 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 statement instructs SUPREM-IV.GS 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 **contour** statements are plotted as part of loop. The **foreach** command repeats 9 times and sets the variable **v** to the value specified in the loop. Figure 1 shows that the lateral extent of the OED is well over two microns. There is little lateral difference in the shape of the profile.

FIGURE 1 Boron contours resulting from the implant and anneal.

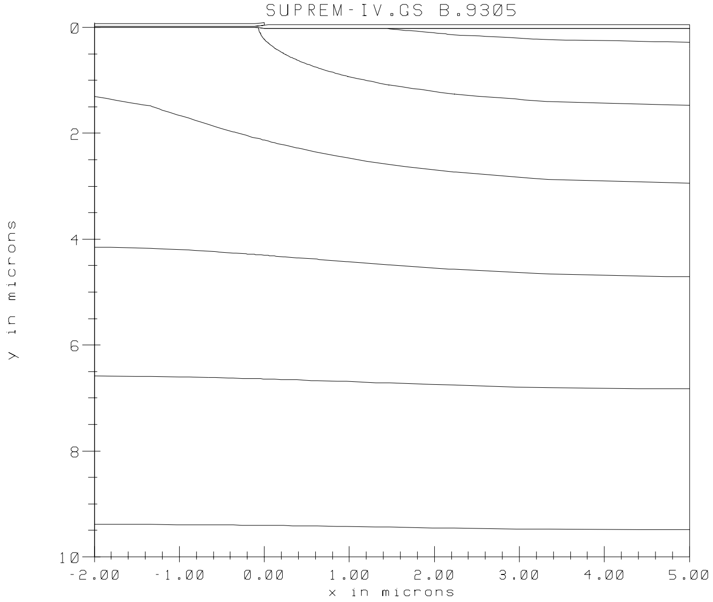


The interstitial concentration can also be plotted. This will verify that there is little lateral decay in the defect profile.

```
#plot the final profile
select 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
```

Figure 2 shows the result. There is more shape to the defects than the boron. However, the profile is fairly flat across the top of the wafer.

FIGURE 2 Interstitial contours resulting from the injection.

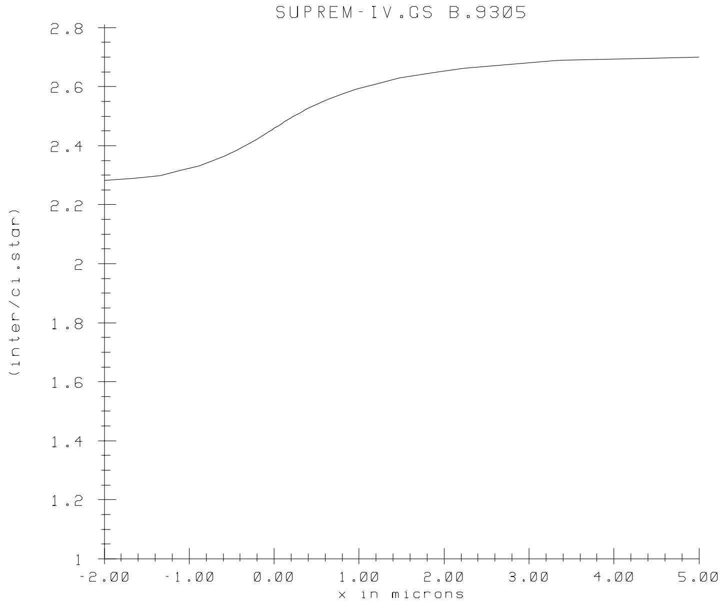


The interstitial concentration can also be plotted in cross section and is shown in Figure 3.

```
plot.1d y.v=0.5
```

This command will plot the interstitial concentration 0.5µm below the surface.

FIGURE 3 Lateral cross section of the interstitial profile.

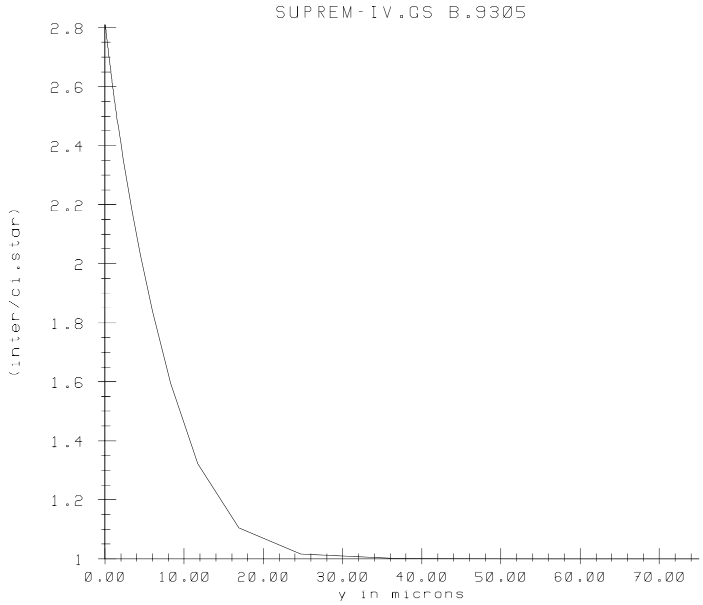


The value can also be plotted vertically, and is shown in Figure 4.

```
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.

FIGURE 4 Vertical cross section of the interstitial profile.



EXAMPLE 5 LDD Cross Section

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-II 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
diff time=10 temp=1000

#etch the poly away
etch poly right p1.x=0.55 p1.y=-0.020 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 spac=0.05

#etch the spacer back
etch dry oxide thick=0.420

struct outf=imp2.str
#after etch anneal
method vert fermi grid.ox=0.0
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 pre-exponential 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.GS 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.

```
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-II, 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.

```
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
```

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 front and backsides of the wafer. Any gasses on the **diffuse**, **deposit**, and **etch** commands 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-II.

The next line informs SUPREM-IV.GS that the mesh has been defined and should be computed.

```
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 0.025µm 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
```

The implant is modeled with a Pearson-IV distribution. The energy and dose are 10^{12}cm^{-2} and 15KeV respectively. This produces an abrupt boron profile with a shallow junction.

The next commands 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
diff time=10 temp=1000

#etch the poly away
etch poly right p1.x=0.55 p1.y=-0.020 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}cm^{-3} 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 spac=0.05

#etch the spacer back
etch dry oxide thick=0.420

struct outf=imp2.str
```

This implant is 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 fermi grid.ox=0.0
diffuse time=30 temp=950 dry
```

The **method** command specifies the oxide will grow vertically. No injection of interstitials will be simulated during this oxide growth. The **diffuse** command specifies that the anneal is to be done for 30 minutes at 950˚C 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
```

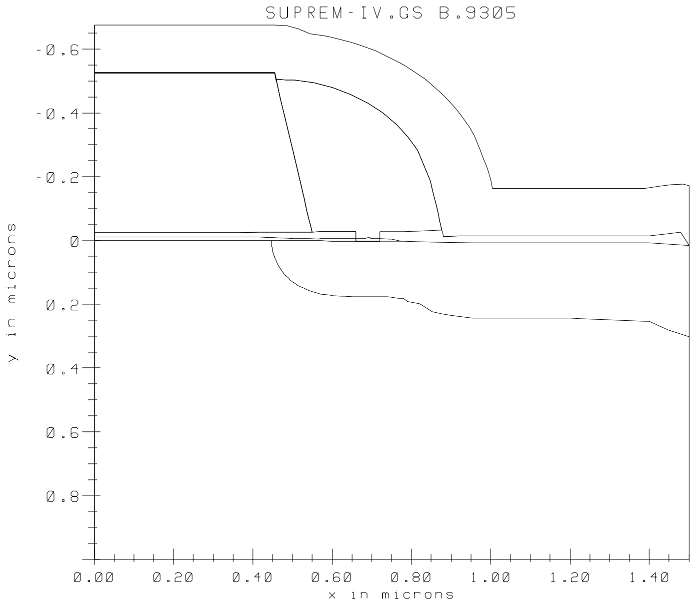
This anneal will be carried out for 20 minutes at 950˚C. The structure is saved in the file ldd.str. At this point, any of the structure files can be read back and examined to check the results. Commands and analysis similar to that found in Example 4 may be performed.

To just perform a simple check, the following commands should produce a plot.

```
plot.2d bound fill y.max=1.0
select 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.

FIGURE 1 Final LDD Device Structure.



To prepare a PISCES-II 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-II 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-II format in the file ldd.mesh. This command also reports the location of contacts in the mesh for PISCES-II.

```
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-II and is a “geom” type file.

EXAMPLE 6 One Dimensional Oxide Growth

DESCRIPTION

This example compares the oxide thickness grown for different orientations. The input file for the simulation is in the “examples/exam6” 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 seven lines of the loop body define the simplest mesh possible. It has exactly four lines (one rectangle), and a front side.

The **method** command chooses the oxide grid spacing to be 0.03µm, knowing that 30 minutes at 900˚C will grow about 0.1µm of oxide. 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 0.1second; subsequently it is automatically chosen to add no more than 14 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 1.286963e-16
Solving        0 +      0.1 =      0.1,      100%,np 6
Solving      0.1 +   302.52 =   302.62,   302520%,np 6
Solving   302.62 +  302.522 =  605.142,  100.001%,np 6
Solving  605.142 +  311.041 =  916.183,  102.816%,np 6
Solving  916.183 +  319.326 =  1235.51,  102.664%,np 6
Solving  1235.51 +  327.623 =  1563.13,  102.598%,np 8
Solving  1563.13 +  236.869 =     1800,  72.2992%,np 8
layer  material        thickness  Integrated
num    type            microns    Depth(A)
```

```
 0     oxide           -0.043     0.000000e+00
 1     oxide           0.000      -3.104514e-04
 2     oxide           0.035      0.000000e+00
 3     silicon         1.000      4.993907e-01
"---------- Orientation < 110 > --------------------"

estimated first time step 4.402759e-16
Solving        0 +      0.1 =      0.1,      100%,np 6
Solving      0.1 +  216.262 =  216.362,   216262%,np 6
Solving  216.362 +  216.266 =  432.629,  100.002%,np 6
Solving  432.629 +  224.785 =  657.413,  103.939%,np 6
Solving  657.413 +   232.98 =  890.393,  103.646%,np 6
Solving  890.393 +  241.199 =  1131.59,  103.528%,np 8
Solving  1131.59 +  249.427 =  1381.02,  103.411%,np 8
Solving  1381.02 +  257.664 =  1638.68,  103.302%,np 8
Solving  1638.68 +  161.318 =     1800, 62.6078%,np 10
layer  material        thickness  Integrated
num    type            microns    Depth(A)
 0     oxide           -0.056     0.000000e+00
 1     oxide           -0.000     -1.296132e-03
 2     oxide           0.046      7.714586e-04
 3     silicon         1.000      4.989346e-01
"---------- Orientation < 111 > --------------------"

estimated first time step 1.269085e-17
Solving        0 +      0.1 =      0.1,      100%,np 6
Solving      0.1 +   180.25 =   180.35,   180250%,np 6
Solving   180.35 +  180.255 =  360.605,  100.003%,np 6
Solving  360.605 +  188.773 =  549.378,  104.726%,np 6
Solving  549.378 +  196.907 =  746.285,  104.309%,np 6
Solving  746.285 +  205.073 =  951.358,  104.147%,np 8
Solving  951.358 +  213.252 =  1164.61,  103.988%,np 8
Solving  1164.61 +  221.444 =  1386.05,  103.841%,np 8
Solving  1386.05 +  229.647 =   1615.7, 103.704%,np 10
Solving   1615.7 +    184.3 =     1800, 80.2534%,np 10
layer  material        thickness  Integrated
num    type            microns    Depth(A)
 0     oxide           -0.065     0.000000e+00
 1     oxide           0.000      -1.962906e-03
 2     oxide           0.053      1.281196e-03
 3     silicon         1.000      4.985963e-01
```

The thickness is the difference between the top and bottom readings for the oxide, namely 780 Å, 1020 Å and 1180 Å for the <100>, <110> and <111> orientations respectively.

EXAMPLE 7 Fully Recessed Oxide Growth

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/exam7” directory, in the file “fullrox.s4”.

```
#Recessed LOCOS cross section: recess 0.3um, grow
0.54um
#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.2d grid bound
```

```
#----------Field oxidation
meth compr

plot.2 bound y.mi=-0.5 line.b=2
diffuse tim=90 tem=1000 weto2 movie="plot.2 b cle=f
axi=f"
stru    outf=fc.mesh

plot.2d bound 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 0.54µm of oxide on <100> silicon. In the process, 0.24µm of silicon is consumed. The second line is around the expected final depth of the oxide, and there is one more line at 1µm to round out the grid.

```
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
```

The x lines run from -1 to 1 for symmetry, with the mask edge at 0. The extra refinement around 0.25µm is to prepare for the silicon etch, which will be at an angle of 54˚ and therefore has a lateral extent of 0.3µm/ $\tan 54^{\circ} \approx 0.218$.

```
region silicon xlo=l xhi=r ylo=t yhi=b
bound  expo    xlo=l xhi=r ylo=t yhi=t
init or=100
```

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.GS 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. The **initialize** command causes the initial rectangular mesh to be generated. The substrate orientation is <100>.

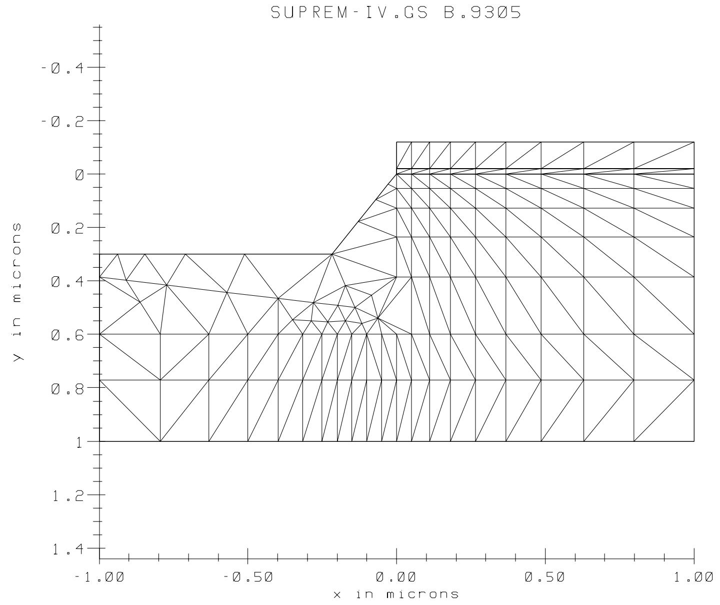
```
#-----------Anisotropic silicon etch
etch   silicon left p1.x=-0.218 p1.y=0.3 p2.x=0 p2.y=0
```

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.2d 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 and is shown in Figure 1.

FIGURE 1 Initial grid structure before oxidation.



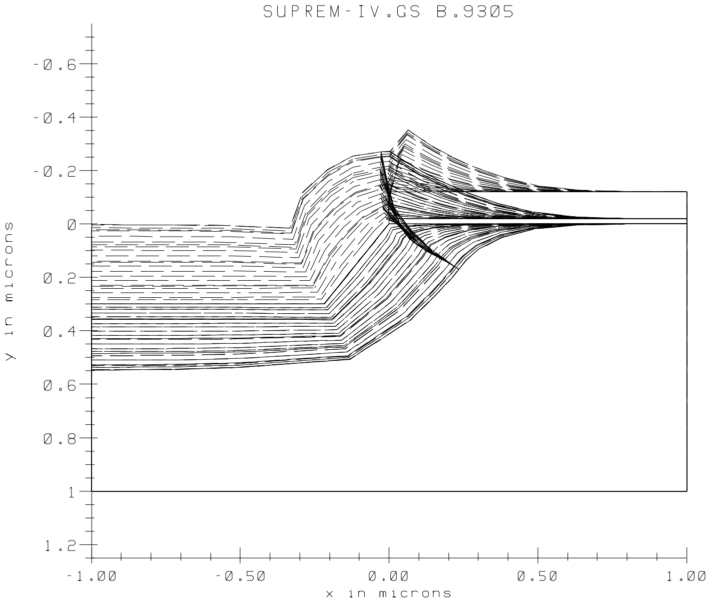
```
#----------Field oxidation
meth compr
```

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.2d bound y.mi=-0.5 line.b=2
diffuse tim=90 tem=1000 weto2 movie="plot.2 b cle=f
axi=f"
```

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 re-scaling the picture. A series of outlines is generated, one from each time step, illustrating the evolution of the oxide profile (Figure 2). The first **plot.2d** bound defined a plotting window large enough so that the subsequent plots without axes would fit inside it.

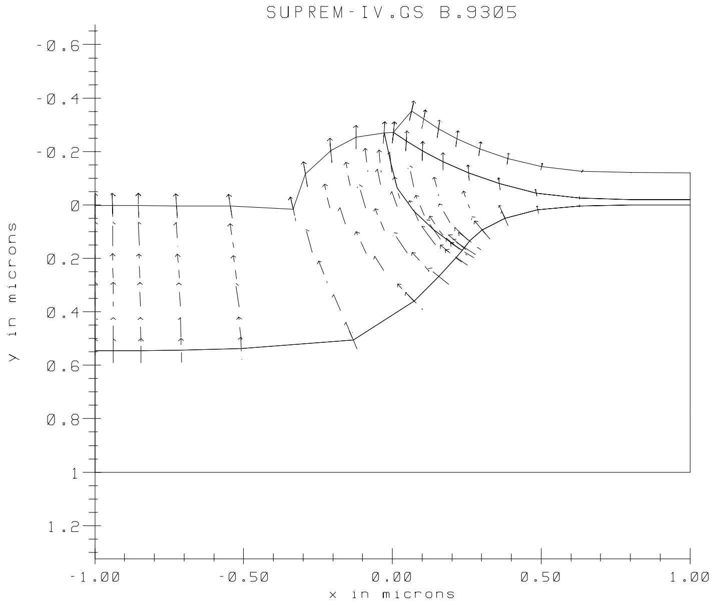
FIGURE 2 Evolution of bird's head profile.



```
plot.2d bound flow vleng=0.1
```

The flow pattern at the last time step can be plotted with the statement above. The parameter **vleng** is the length to draw the longest velocity vector. The other vectors are scaled proportionally. This results in Figure 3. The flow is normal at the interface but becomes more vertically oriented away from it.

FIGURE 3 Flow vectors at the end of oxidation.



EXAMPLE 8 Shear Stress

DESCRIPTION

This example calculates the extent of shear stress in the silicon substrate, generated by a film edge. It is well known that when Si_3N_4 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/exam8 directory under the name. 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 Irene [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 analyzed, starting with a space 10µm by 5µm. 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. 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.

```
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
```

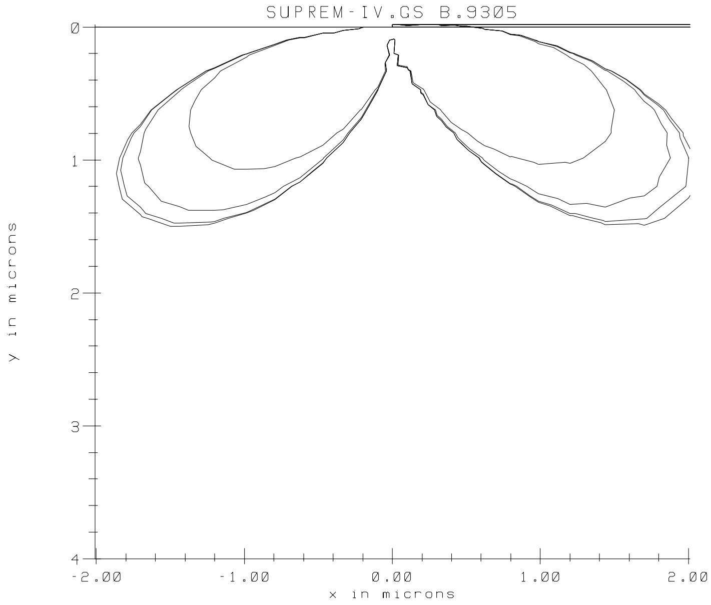
This calculates the stress distribution arising from the nitride initial stress. Stress arising from thermal expansion mismatch would be included by specifying a different **temp2**. Be warned that large thermal steps often bring into play many more complicated phenomena than the simple thermal expansion mismatch analyzed in SUPREM-IV.GS. 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. This corresponds to the σ_{xy} shear force in the plane of the simulation. The value 3×10^7 is considered by Hu[2] 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 σ_{xy} equal to 3×10^7 contour.

```
plot.2  bound  x.mi=-2 x.ma=2 y.ma=4 cl=f
select  z=Sxy
contour val=-3e7
```

The contours of σ_{xy} equal to 3×10^7 in the silicon are shown in Figure 1. The contour is a double lobe because the shear stress is related to the polar components of stress through $\sigma_{xy} = (\sigma_{rr} - \sigma_{\theta\theta})\ \sin 2\theta + \sigma_{r\theta}\ \cos 2\theta\ \ \sigma_{rr}\ \sin 2\theta$. The function $\sin 2\theta$ is at a maximum around 45˚ from the vertical, and is zero at $\theta = 90˚$. Both σ_{rr} and $\cos 2\theta$ 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 ±2µm horizontally, and 1.5µm vertically.

FIGURE 1 Critical Shear Stress Contours.



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.

References

1. E. A. Irene, J. Electronic Mat., **5**(3), p. 287, (1976).

2. S. M. Hu, "Film-edge-induced Stress in Silicon Substrates," *Appl. Phys. Lett*, **32**(1), p. 5, 1978

EXAMPLE 9 Stress Dependent Oxidation

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 two to three 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. 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.GS can predict such second-order effects. The example is given in the file “sdep.s4” in the “examples/exam9” directory.

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 [2].

```
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 0.05µm 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  0.000463 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 loop28  cut  0.5  upd   0.00263 orhs  0.0005139
rhs  0.0002645
Newton loop30  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 proceeding 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 lambda = 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 lambda 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 (lambda = 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 (lambda =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 Figures 1 and 2. The thicker nitride clearly has the effect of reducing the bird's beak.

FIGURE 1 Comparison of LOCOS Shapes with 500Å Nitride and 1500Å Nitride.

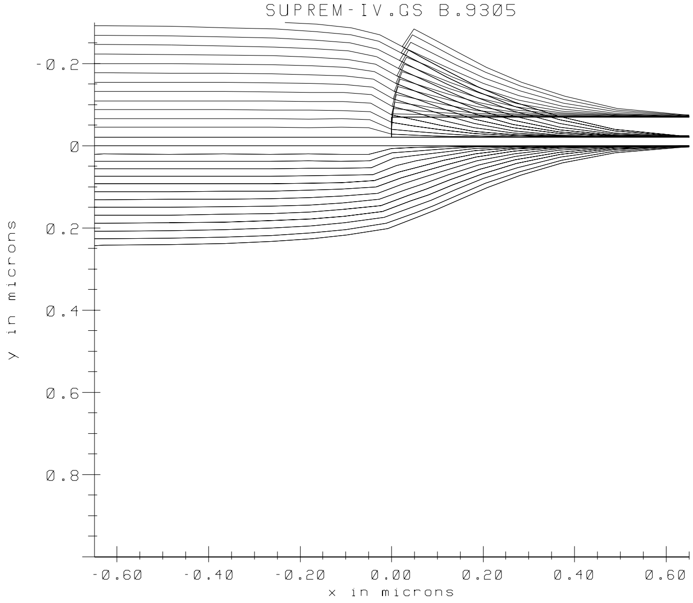
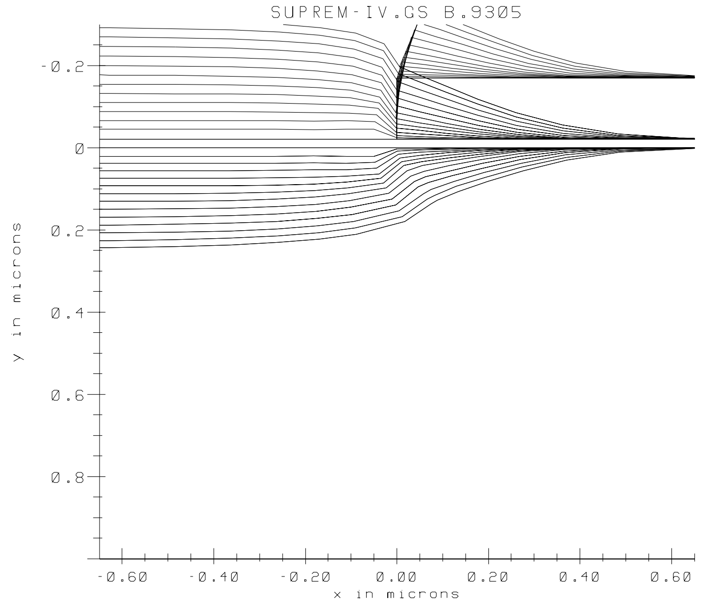


FIGURE 2 Comparison of LOCOS Shapes with 500Å Nitride and 1500Å Nitride.



References

1. 1. N. Guillemot, G. Pananakakis and P. Chenevier, *IEEE Transactions on Electron Devices*, **ED-34**, (1987).
2. 2. D.-B. Kao, J. P. McVittie, W. D. Nix and K. C. Saraswat, “Two-Dimensional Silicon Oxidation Experiments and Theory,” *IEDM Tech. Digest*, 1985,

EXAMPLE 10 GaAs MESFET Gate Simulation – 1D

DESCRIPTION

This example simulates a simple 1D GaAs MESFET structure underneath the gate. This utilizes SUPREM-IV.GS's new true 1D mode. First, the 1D grid is set up and initialized with carbon as a background dopant. Beryllium and silicon are implanted into GaAs. Note that silicon as an impurity is designated as “**isilicon**”. The as-implanted dopant concentration-depth profiles are plotted in 1D in Figure 1. Next an anneal step is done, using the Fermi method, and the diffused beryllium profile is then plotted in the same figure. The hump in the beryllium profile at the beryllium/silicon junction is due to the electric field effect on diffusion. Since the beryllium was implanted, the implanted diffusivity numbers (as opposed to the grown-in numbers) are used.   The input file for the simulation is in the “examples/exam10” directory, in the file example10.in.

```
option quiet
set echo

mode one.dim
line x loc=0.0 spacing=0.01 tag=top
line x loc=1.0 spacing=0.01
line x loc=20 spacing=0.25 tag=bottom

region gaas       xlo=top  xhi=bottom
boundary exposed  xlo=top xhi=top
boundary backside xlo=bottom xhi=bottom

init carbon conc=1e15
implant beryllium dose=2e13 energy=100 pearson
implant isilicon dose=5e13 energy=50 pearson

#beryllium gaas Dip.0=2.1e-8 Dip.E=1.74
beryllium gaas /nitride Seg.0=.5 Seg.E=0

deposit nitride thick=.3
```

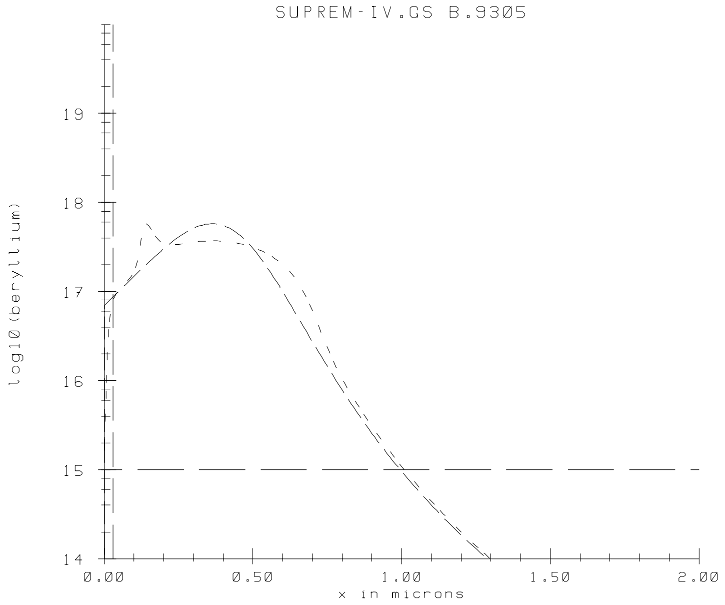
```
     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20
line.type=2
     select z=log10(isilicon)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=3
     select z=log10(carbon)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=4

method fermi init=1e-5
method full.fac
diffuse time=15 temp=800 argon
     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=5

quit
```

If you wish to change the beryllium diffusivity parameters, you would uncomment the line:

```
#beryllium gaas Dip.0=2.1e-8 Dip.E=1.74
```

and put in your own values.

FIGURE 1 Beryllium, Silicon, and Carbon profiles under the MESFET gate.

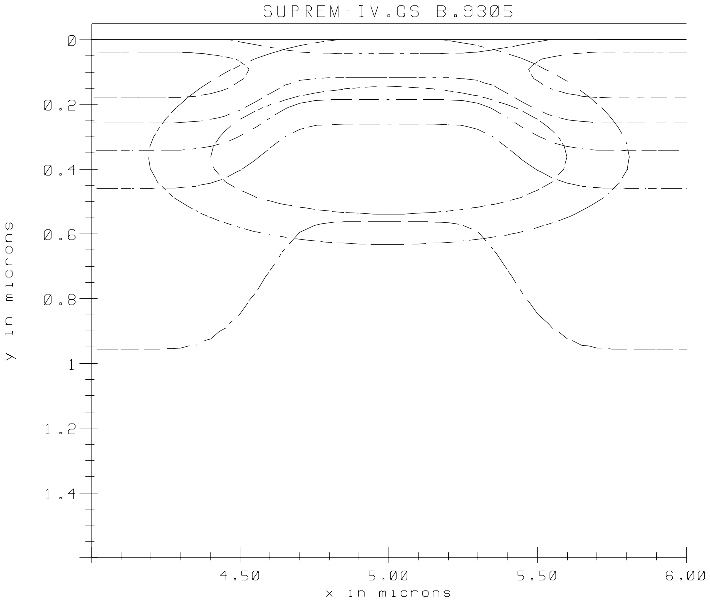


EXAMPLE 11 GaAs Simulation – 2D

This shows how one would do a 2D GaAs simulation. A 2D grid is set up, and silicon and beryllium are implanted in the source, drain and channel regions. Then the doping contours are plotted for the designated concentration levels in Figure 2. In this example, only one half of the structure is simulated, then mirrored before the result is plotted, in order to speed up simulation time. The input file for the simulation is in the “examples/exam11” directory, in the file example11.in.

```
option quiet
set echo

line x loc = 0.0 tag=left   spacing = 0.5
line x loc = 0.50           spacing = 0.5
line x loc = 4.00           spacing = 0.5
line x loc = 4.10           spacing = 0.1
line x loc = 4.20           spacing = 0.05
line x loc = 4.50           spacing = 0.05
line x loc = 5.00 tag=right spacing = 0.05

line y loc=0.0 tag=top      spacing=0.03
line y loc=0.3              spacing=0.03
line y loc=1.0             spacing=0.1
line y loc=3.0  tag=bottom  spacing=0.3

region gaas  xlo=left  xhi=right  ylo=top  yhi=bottom
boundary exposed xlo=left xhi=right ylo=top yhi=top
boundary backside xlo=left xhi=right ylo=bottom yhi=-
bottom
init isilicon conc=3e15

deposit nitride thick=1.0
etch nitride start x=5 y=0.0
etch continue x=5 y=-1.10
etch continue x=4.5 y=-1.10
etch done x=4.5 y=0.0

implant isilicon dose=1.75e12 energy=75 pearson
implant beryllium dose=1e13 energy=100 pearson

etch nitride start x=4.5 y=0.0
etch continue x=4.5 y=-1.10
etch continue x=0 y=-1.10
etch done x=0 y=0.0

deposit nitride thick=1.0
etch nitride start x=4.5 y=0.0
etch continue x=4.5 y=-1.10
etch continue x=0.0 y=-1.10
etch done x=0.0 y=0.0

implant isilicon dose=1e13 energy=100.0 pearson

etch nitride start x=4.5 y=0.0
etch continue x=4.5 y=-1.10
etch continue x=5.0 y=-1.10
etch done x=5.0 y=0.0

deposit nitride thick=.05

structure mirror right

plot.2d bound fill x.min=4.0 x.max=6 y.max=1.6
select z=log10(isilicon)
foreach v (15. to 18.5 step 0.5)
contour val=v line.type=2
end

plot.2d bound fill x.min=4 x.max=6 y.max=1.6 cle=f
axi=f
select z=log10(beryllium)
foreach v (16.5 to 17.0 step .5)
contour val=v line.type=4
end

quit
```

FIGURE 1 Beryllium and Silicon contours in the MESFET source, drain, and channel regions.



EXAMPLE 12 Dopant Activation Model Example

Here the SUPREM-IV.GS dopant activation model is illustrated. The input file for the simulation is in the “examples/exam12” directory, in the file example12.in.

```
option quiet
set echo

mode one.dim
line x loc=0.0 spacing=0.02 tag=top
line x loc=0.5 spacing=0.02
line x loc=20 spacing=0.25 tag=bottom

region gaasxlo=top  xhi=bottom
boundary exposed xlo=top xhi=top
boundary backside xlo=bottom xhi=bottom

init beryllium conc=3e17
implant isilicon dose=5e13 energy=100 pearson

#material gaas     p.type  act.a="(2.2 - 0.00117 * T)"
act.b="1.00e21"
#material gaas     n.type  act.a="(2.2 - 0.00117 * T)"
act.b="4.25e18"

deposit nitride thick=.3
method fermi init=1e-5

diffuse time=.001 temp=750 argon
     select z=log10(isilicon)
     plot.1d x.mi=0 x.ma=2 y.mi=14 y.ma=20 line.type=1
     select z=log10(electrons)
     plot.1d x.mi=0 x.ma=2 y.mi=14 y.ma=20 cle=f axi=f
line.type=2
     select z=log10(abs(doping))
     plot.1d x.mi=0 x.ma=2 y.mi=14 y.ma=20 cle=f axi=f
line.type=3

quit
```

EXAMPLE 12: Dopant Activation Model Example

Silicon is implanted into GaAs with a beryllium doped background. Figure 1 shows the 1D plot of this. The default n and p-type activation models are used, commented out in the lines:

```
#material gaas p.type act.a="(2.2 - 0.00117 * T)"
act.b="1.00e21"
#material gaas n.type act.a="(2.2 - 0.00117 * T)"
act.b="4.25e18"
```

Plotted in Figure 1 are the silicon profile, the electron profile, and the net doping profile. The electron profile takes into account the less than 100 percent net n-type dopant activation under the silicon peak, as well as the reduced electron concentration in the bulk due to the beryllium background doping. The abs(doping) profile clearly shows the silicon/beryllium n/p junction.

Figure 2 shows the results when the n-type activation model parameters are changed. In this case, the following line is included:

```
material gaas n.type act.a="(2.2 - 0.00117 * T)"
act.b="4.25e18"
```

Note that the **act.b** parameter has been changed. This increases the net active n-type dopant concentration under the peak of the silicon implant profile.

FIGURE 1

Silicon, electrons, and net doping profiles for a silicon/beryllium MESFET structure. The default activation models are used for silicon.

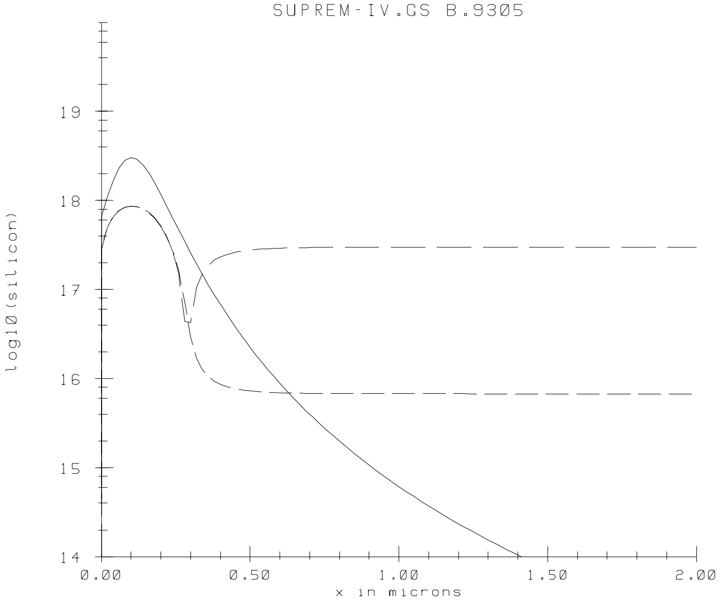
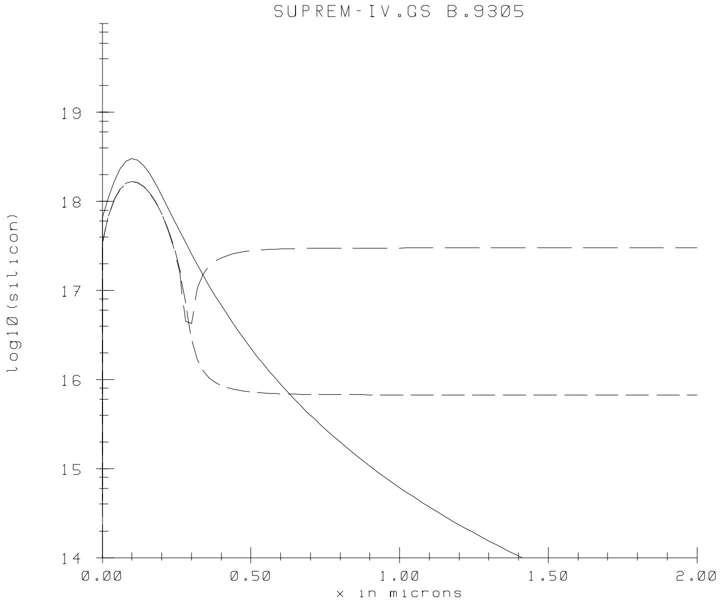


FIGURE 2

Same as Figure 1 except that modified silicon activation numbers are used, resulting in a higher silicon activation.

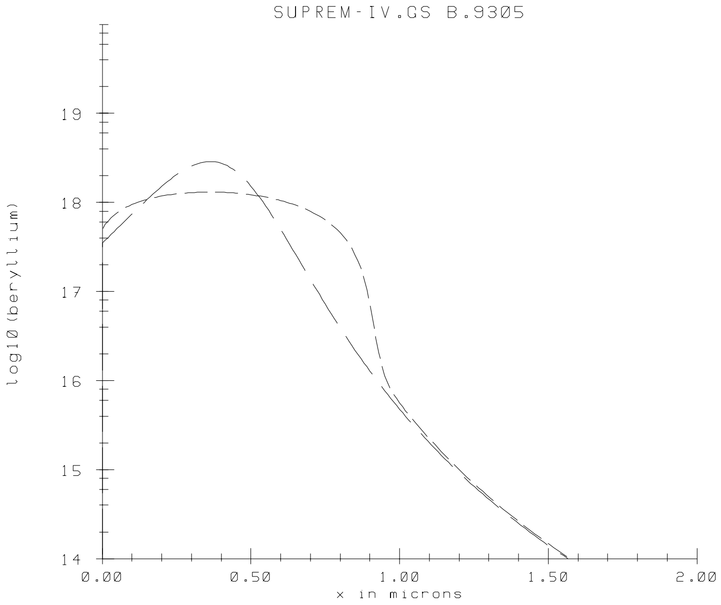


EXAMPLE 13 Implanted and Annealed Beryllium

Examples 13, 14, and 15 illustrate the differences in using implanted vs. grown-in dopants, and how it effects the diffusivity values used. In Example 13, beryllium is implanted in GaAs, and the structure is annealed. The input file for the simulation is in the “examples/exam13” directory, in the file example13.in. The as-implanted and diffused beryllium profiles are shown in Figure 1. The implanted diffusion coefficients are used since there is an “implant beryllium ...” statement.

```
option quiet
set echo

mode one.dim
line x loc=0.0 spacing=0.01 tag=top
line x loc=1.0 spacing=0.01
line x loc=20 spacing=0.25 tag=bottom

region gaas       xlo=top  xhi=bottom
boundary exposed  xlo=top  xhi=top
boundary backside xlo=bottom xhi=bottom

init carbon conc=1e15
implant beryllium dose=1e14 energy=100 pearson

deposit nitride thick=.3

     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20
line.type=4

method fermi init=1e-5
method full.fac
diffuse time=15 temp=800 argon
     select z=log10(beryllium)
     plot.1d x.mi=0 x.ma=2 y.mi=14 y.ma=20 cle=f axi=f
line.typ=2

quit
```

FIGURE 1 Implanted and annealed beryllium using the implanted diffusion coefficients.



EXAMPLE 14 Grown-in and Annealed Beryllium

The input file for this simulation is in the “examples/exam14” directory, in the file example14.in.

```
option quiet
set echo

mode one.dim
line x loc=0.0 spacing=0.01 tag=top
line x loc=1.0 spacing=0.01
line x loc=20 spacing=0.25 tag=bottom

region gaas       xlo=top  xhi=bottom
boundary exposed  xlo=top  xhi=top
boundary backside xlo=bottom xhi=bottom

init beryllium conc=1e15
deposit gaas thick=.5 divisions=100 beryllium 
conc=5e18
deposit gaas thick=.5 divisions=100 beryllium 
conc=1e15
deposit nitride thick=.3

     select z=log10(beryllium)
     plot.1d x.min=-1 x.ma=1 y.mi=14 y.max=20 
line.type=4

method fermi init=1e-5
method full.fac
diffuse time=15 temp=800 argon
     select z=log10(beryllium)
     plot.1d x.min=-1 x.ma=1 y.mi=14 y.max=20 cle=f 
axi=f line.type=2

quit
```

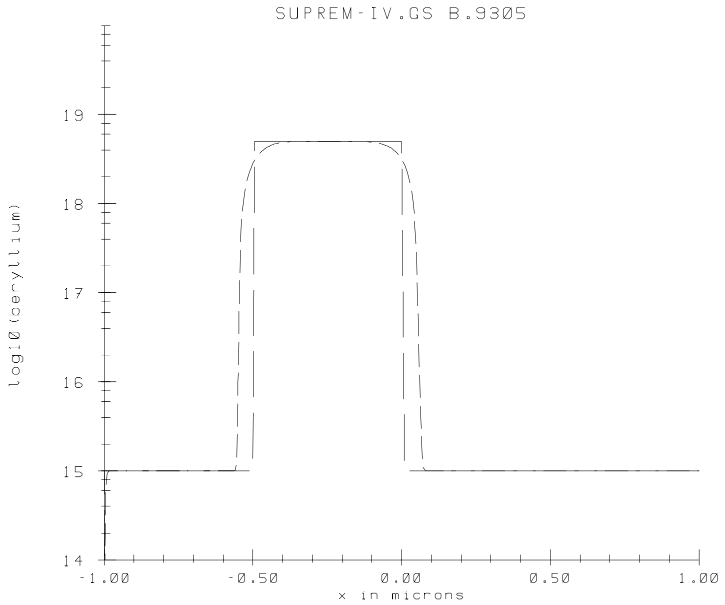
Here, Be is grown into the GaAs by using the **deposit** statements

```
deposit gaas thick=.5 divisions=100 beryllium 
conc=5e18
```

```
deposit gaas thick=.5 divisions=100 beryllium
conc=1e15
```

instead of an **implant** statement. In this case, SUPREM-IV.GS uses the lower “as-grown” diffusivities for beryllium since no implanted beryllium is present. The as-grown and diffused profiles are shown in Figure 5. The extent of beryllium diffusion is seen to be much less in this case compared to the implanted case (Example 13), even though the same anneal conditions are used and the beryllium peak concentrations are comparable (the diffusion is hole, or concentration, dependent).

FIGURE 1 Grown-in and annealed beryllium in GaAs using the grown-in diffusion coefficients.



EXAMPLE 15 Annealing An Implant From Measured (SIMS) Data

In this example, the “implanted” diffusivities are used, even though there is no implant statement. The input file for the simulation is in the “examples/exam15” directory, in the file example15.in.

```
option quiet
set echo

mode one.dim
line x loc=0.0 spacing=0.01 tag=top
line x loc=1.0 spacing=0.01
line x loc=20 spacing=0.25 tag=bottom

region gaas      xlo=top  xhi=bottom
boundary exposed xlo=top  xhi=top
boundary backside xlo=bottom xhi=bottom

init carbon conc=1e15
profile infile=be1 beryllium
implant beryllium dose=1e1 energy=100 pearson
deposit nitride thick=.3

     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20
line.type=4

method fermi init=1e-5
method full.fac
diffuse time=15 temp=800 argon
     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=2

quit
```

The as-implanted profile obtained from SIMS measurements is input from a file (“be1”) by using the **profile** statement:

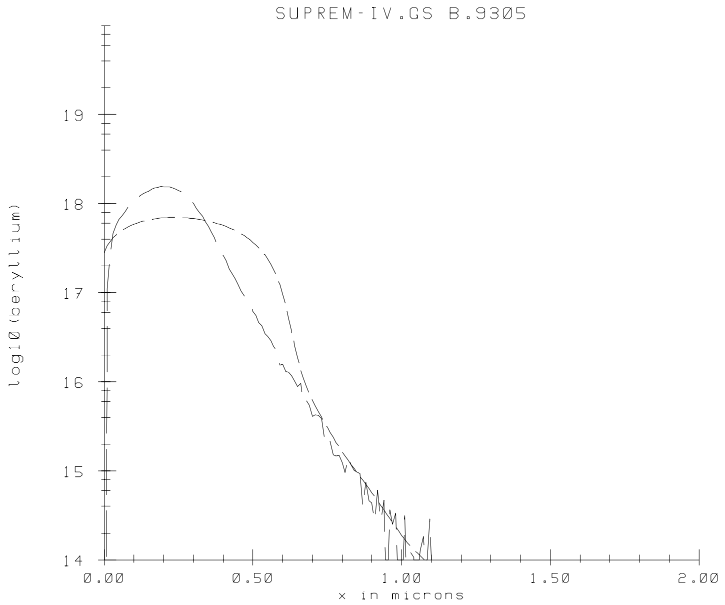
```
profile infile=be1 beryllium
```

File be1 contains the x concentration values for the as-implanted beryllium profile. In order to use the “implanted” diffusivities even though there is no **implant** statement a “dummy” implant is done,

```
implant beryllium dose=1e1 energy=100 pearson
```

in which a very low dose beryllium implant is used. As shown in Figure 1, the extent of diffusion is comparable to that of Example 13, and much more than in Example 14, indicating the higher “implanted” diffusivity is used.

FIGURE 1 Implanted beryllium from SIMS data and SUPREM-IV.GS annealed profile.



EXAMPLE 16 Two.Dim Diffusion With File Input Interstitials

Here the **two.dim** method for diffusion is illustrated. The input file for the simulation is in the “examples/exam16” directory, in the file example16.in.

```
option quiet
set echo

mode one.dim
line x loc=0.0 spacing=0.01 tag=top
line x loc=1.0 spacing=0.01
line x loc=20 spacing=0.25 tag=bottom

region gaas      xlo=top  xhi=bottom
boundary exposed xlo=top  xhi=top
boundary backside xlo=bottom xhi=bottom

init carbon conc=1e15
implant beryllium dose=1e14 energy=100 pearson
profile infile=file1 inter
deposit nitride thick=.3

interstitial gaas D.0=5e-14     D.E= 0.
interstitial gaas Kr.0=1e-18    Kr.E= 0.
interstitial gaas Cstar.0= 1.0e16    Cstar.E= 0.
interstitial gaas  /nitride  Ksurf.0= 1e-3    Ksurf.E=
0.
interstitial gaas neu.0=0 pos.0=1 neu.E=0 neg.0=0
pos.E=0 dneg.0=0
interstitial gaas dpos.0=0 dpos.E=0 neg.E=0 tpos.0=0
tneg.0=0

vacancy gaas D.0= 1e-15      D.E= 0.
vacancy gaas Kr.0=1e-18      Kr.E= 0.
vacancy gaas Cstar.0= 1e16    Cstar.E= 0.
vacancy gaas  /nitride  Ksurf.0= 1e-3     Ksurf.E= 0.
vacancy gaas neu.0=0 pos.0=0 neu.E=0 neg.0=0 pos.E=0
dneg.0=0
vacancy dpos.0=0 dpos.E=0 neg.E=0 tpos.0=0 tneg.0=1
```

```
method full.fac

     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20
line.type=4
     select z=log10(inter)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=2

method two.dim init=1e-5
diffuse time=15 temp=800 argon
     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=4
     select z=log10(inter)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=2

quit
```

In Example 13, the Fermi method for diffusion was used, in which it is assumed that interstitial and vacancy concentrations are at their equilibrium values throughout the simulation. Therefore, the diffusivities are only dependent on temperature and the local doping concentrations (n or p). In Example 14, the **two.dim** method for diffusion is used, in which extrinsic interstitials or vacancies (i.e. added from some source, such as implant damage, or from oxidation in the case of silicon technology) are taken into account in the diffusion of dopants. In this case _I/I*_ or _V/V*_ may not equal one, and these terms are included in the diffusion equations. In this example, the interstitial profile is input by the statement:

```
profile infile=file1 inter
```

where file1 contains the x concentration values for the interstitials that might be caused by implantation damage. Now the statement:

```
method two.dim init=1e-5
```

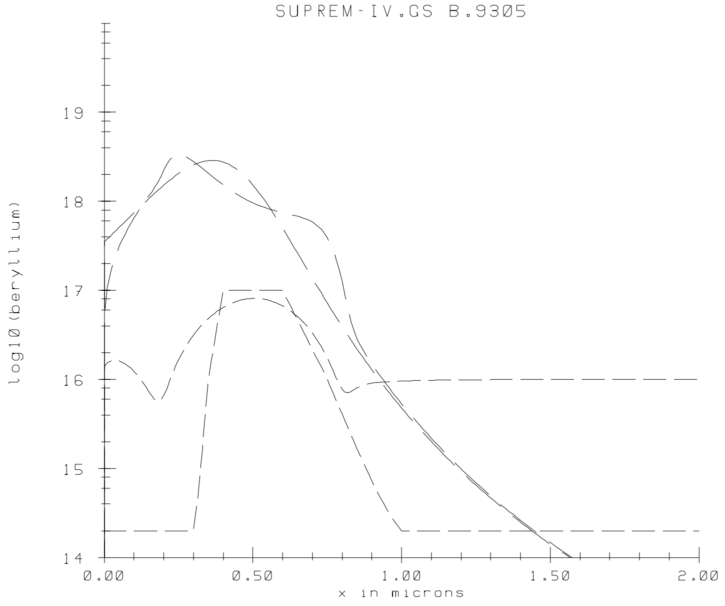
is used, rather than the fermi method statement. Parameters for various interstitial and vacancy properties are also included. For example, only +1 charge state interstitials are used, according to the statements:

```
interstitial gaas neu.0=0 pos.0=1 neu.E=0 neg.0=0
pos.E=0 dneg.0=0
interstitial gaas dpos.0=0 dpos.E=0 neg.E=0 tpos.0=0
tneg.0=0
```

The as-implanted and diffused profiles of beryllium and interstitials are shown in Figure 1. The diffused beryllium profile is much different than that in Example 13, where the fermi method was used (with no added interstitials). The simulation predicts the famous “uphill diffusion” of implanted p-type dopants often observed.

FIGURE 1

Beryllium and interstitial profiles, before and after anneal, using the two.dim method for diffusion. The excess interstitials (above I^*) produce “uphill diffusion” of Beryllium.



EXAMPLE 17 Full.Coupled Diffusion

The input file for the simulation is in the “examples/exam17” directory, in the file example17.in. In this example, the full.coupled method of diffusion is illustrated.

```
option quiet
set echo

mode one.dim
line x loc=0.0 spacing=0.01 tag=top
line x loc=1.0 spacing=0.01
line x loc=20 spacing=0.25 tag=bottom

region gaas      xlo=top  xhi=bottom
boundary exposed xlo=top  xhi=top
boundary backside xlo=bottom xhi=bottom

init carbon conc=1e15
implant beryllium dose=1e14 energy=100 pearson
deposit nitride thick=.3

interstitial gaas D.0=5e-14     D.E= 0.
interstitial gaas Kr.0=1e-18    Kr.E= 0.
interstitial gaas Cstar.0= 1.0e16  Cstar.E= 0.
interstitial gaas  /nitride  Ksurf.0= 1e-3  Ksurf.E= 0.
interstitial gaas neu.0=0 pos.0=1 neu.E=0 neg.0=0
pos.E=0 dneg.0=0
interstitial gaas dpos.0=0 dpos.E=0 neg.E=0 tpos.0=0
tneg.0=0
interstitial gaas beryllium neu.0=0 pos.0=0 neg.0=0
dneg.0=0 dpos.0=0
interstitial gaas beryllium tneg.0=0 tpos.0=0

vacancy gaas D.0= 1e-15      D.E= 0.
vacancy gaas Kr.0=1e-18      Kr.E= 0.
vacancy gaas Cstar.0= 1e16   Cstar.E= 0.
vacancy gaas /nitride  Ksurf.0= 1e-3  Ksurf.E= 0.
vacancy gaas neu.0=0 pos.0=0 neu.E=0 neg.0=0 pos.E=0
dneg.0=0
```

```
vacancy dpos.0=0 dpos.E=0 neg.E=0 tpos.0=0 tneg.0=1
vacancy gaas beryllium neu.0=0 pos.0=0 neg.0=0
dneg.0=0 dpos.0=0
vacancy gaas beryllium tneg.0=0 tpos.0=0

method full.fac
diffuse time=.00001 temp=800 argon
     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20
line.type=4
     select z=log10(inter)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=2

method full.cpl init=1e-5
diffuse time=15 temp=800 continue argon
     select z=log10(beryllium)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=4
     select z=log10(inter)
     plot.1d x.min=0 x.ma=2 y.mi=14 y.max=20 cle=f
axi=f line.type=2

quit
```

Here Example 13 is repeated, but using the full.coupled diffusion method rather than the fermi method. In this method, non-equilibrium levels of interstitials or vacancies can be produced by the diffusion process itself, without adding non-equilibrium levels of defects at the beginning, as was done in Example 16. This is because in the diffusion process, the dopant diffuses as a dopant/defect pair, and defects are carried along with the dopant. This can result in local regions of non-equilibrium concentrations of defects and can in turn effect the dopant diffusion. (If one believes that the “kick-out” mechanism is occurring rather than the pair mechanism, the same effect occurs as the interstitial dopant kicks-out a matrix atom, creating an interstitial.) The **method** statement is now:

```
method full.cpl init=1e-5
```

Note that there is an initial very short **diffuse** statement:

```
diffuse time=.00001 temp=800 argon
```

In this first diffusion step, a short time anneal is done to establish the equilibrium defect concentrations in case you want to plot the initial values of these (since they are temperature dependent). In the second **diffuse** statement:

```
diffuse time=15 temp=800 continue argon
```

the **continue** parameter is used to maintain the initial defect values. Therefore, the initial **diffuse** uses the fermi method, and the defect levels are initially set at their equilibrium values. (Since the interstitials are set at the +1 charge state, the equilibrium interstitial concentration is fermi level dependent and therefore the initial interstitial profile follows the beryllium profile). The main diffusion step uses the **full.cpl** method. The dopant/defect pairing parameters are specified by the following statements.

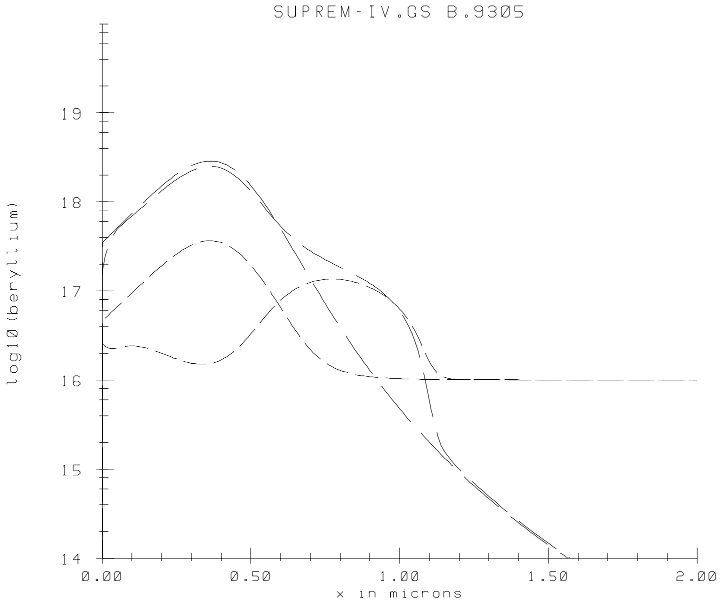
```
interstitial gaas beryllium neu.0=0 pos.0=0 neg.0=0
dneg.0=0 dpos.0=0
interstitial gaas beryllium tneg.0=0 tpos.0=0
```

These are all normally set to 0 if one assumes that the pair concentration is much less than the defect concentration (non-equilibrium levels of defects will still occur, since the defect flux equation in the full-coupled mode still takes into account the extra defects produced by the diffusion process). In Figure 1, the as-implanted and diffused beryllium and interstitial profiles generated are shown. One can see that, compared to Example 13, a kink in the profile occurs due to this defect non-equilibrium effect. If one increases the interstitial diffusivity in the statement:

```
interstitial gaas D.0=5e-14 D.E= 0.
```

from 5×10^{-14} to 1×10^{-11} for example, the interstitials are able to diffuse back to their equilibrium levels everywhere, and normal “Example 13 type” diffusion occurs.

FIGURE 1 Beryllium and interstitial profiles, before and after anneal, using the full.coupled method for diffusion. Excess interstitials created by the pair (or kickout) diffusion mechanism, result in a kink or knee in the beryllium profile.


SUPREM-IV.GS B.9305
log10(beryllium)
x in microns