# Parallel Shared Computational Prototyping Environment (ParaSCOPE) Using Scalable Libraries

Principal Investigators: Robert W. Dutton, Kincho H. Law, and Peter Pinsky

Contract Number: DABT63-95-C-0090
Project Number: HJ1500-5205-0724

March 29, 2000

## Executive Summary

The ParaSCOPE project has developed and demonstrated new concepts in creating a Shared COmputational Prototyping Environment (i.e. SCOPE). The key contributions include: common procedural interfaces (API) and shared information about wafer representations (SWR); use of these interfaces and services in creating a heterogeneous Technology-CAD demonstration vehicle; ability to rapidly prototype partial differential equations (PDE) for TCAD applications, primarily in the area of novel device design; use of scripting language for portable representation of the PDEs; demonstration of the ParaSCOPE modules in computational prototyping of advanced electronic devices. Results of the advanced modeling capabilities are presented in this report, along with representative examples of how the PDE-scripting approach allows rapid prototyping. Specific examples related to silicon on insulator (SOI) simulations–both at the device and higher functional block levels–have been demonstrated. Collaborative efforts involving university, government lab and industrial participants and using ParaSCOPE modules have demonstrated the power of the overall capabilities to implement and test totally new device formulations on a quick-turn-around basis. The project has had additional benefits in creating new knowledge, as reflected in PhD theses of several graduate students and an extensive set of publications–both aspects are reflected in the appendices of the report.

# 1 Objective

This project is targeted at demonstration of a rapid computational prototyping environment that enables the design of advanced semiconductor devices based upon tools that support parallel computation and using flexible scripting approaches to system description implementation. In addition to integrating and parallelizing portions of key underlying 3D Technology Computer-Aided Design (TCAD) tools for process and device modeling of IC technologies, applications of the resulting capabilities for a new computational prototyping platform–PROPHET–have been demonstrated.

# 2 Approach

A layered tool integration strategy is employed; encapsulation at the application level is used to provide simplified tool integration and improve software interoperability and portability. Where appropriate, individual applications within this framework exploit the parallelism inherent in the problem domains in order to address increasing computational complexity in 3D simulations. Examples benchmarking feasibility, in support of defense and industrial IC applications, have been tested and validated to the degree possible. Commercialization of component software technology was promoted.

# 3 Results

## 3.1 Supporting Framework Developments Under ParaSCOPE

ParaSCOPE's heterogeneous tool integration strategy relies upon having a sufficient, yet minimal, information model for representation of the geometric, field, and material properties for a semiconductor wafer. This semiconductor wafer representation (SWR) has been defined and implemented as part of previous work on the SPRINT-CAD project [13]. Current wafer state representation issues are now primarily being addressed under other DARPA(ETO)-funded activities in the Composite CAD Program [12], targeted primarily at radio frequency (RF) MEMS applications of interest to DoD. In addition to this information model, a unified control access model of an increasingly complex set of tools (and

their associated interactions) became necessary. Namely, the migration of tools into a heterogeneous computing environment required consistency in the application-level view of the tools [27].

Tcl (Tool Control Language) interface calls and wrappers of core capabilities such as grid and geometry servers as well as application-domain tools have become the dominant method for tool integration. In the area of device specification, the system currently incorporates level set modeling techniques [14] as well as traditional solid modeling methods for prototype construction. An object repository, written in C/C++, has been developed to serve as a layer of abstraction for data interchange among different modules. The repository works in the following way: individual system modules communicate with the object repository, either registering new objects or retrieving existing ones. When two modules need to share data, they do so via these repository transactions, thereby avoiding direct communication. In this way, the repository mechanism facilitates communication between modules while maintaining independence among them. The prototype system employs this mechanism to manage interactions between solid modeling methods, level set surface movement capabilities, and visualization routines.

The Tcl extension language is used to control each module and its repository transactions from high-level user scripts. By exporting the functionality of several components to a unified Tcl environment, script-level control of module interactions and large numbers of intermediate objects is enabled. Furthermore, because Tcl is an interpreted language, it provides intuitive, responsive access to the underlying modeling methods, and supports rapid modifications by end-users without the need to rebuild the environment.

### 3.2 Application-Driven Development and Testing of ParaSCOPE Modules

While primary efforts have been devoted to completion of the software modules, there was ongoing progress on the application side. In addition to the highly configurable and extensible PDE solver modules, ALAMODE [15] and PROPHET [16], an extensible environment for geometric modeling has also been developed. Geometry modeling plays a key and essential role in micro-electro-mechanical systems (or MEMS) and modeling of interconnects.

Both areas have yielded important results based on use of ParaSCOPE modules. Results from each of these geometry-based modeling areas are discussed briefly in the following. More detailed presentation of results involving PROPHET are given in Section 5.

Level set methods for geometry construction have become a major factor in this research project, both in terms of application pull within the diverse potential uses for the ParaSCOPE modules and in redesigning new algorithmic concepts (including new opportunities to use parallelism). One DoD-oriented application is the use of geometry modeling (including the level set work) in prototyping and framework support of MEMS modeling – work that has directly supported the Composite CAD (DARPA/ETO) Project. Specifically, the geometric complexity of MEMS devices comes from extensive use of deposition and etching of layers, processing steps that result in surfaces that are neither ideal nor well-represented by simple geometry modelers. The structures that have been prototyped with advanced level set methods and server modules, developed in large part under this contract, provide realistic physical prototypes [26]. These modules continue to be integrated and applied within the ongoing applications and demonstrations under the Composite CAD Program. The most recent demonstration was given at the October 5-7, 1999, DARPA PI (Composite CAD) Meeting in Agoura Hills, California.

Technology developed under this project has been applied to the area of VLSI interconnects as well. This application demonstration has been transitioned into one of the newly formed (and industry-supported) SIA/FRC in the area of interconnect technologies. As the chip size increases, the delay and adverse coupling caused by R/L/C (parasitics) of global interconnects become major design concerns. The correct circuit simulation depends on the accuracy of those extracted R/L/C based on the layout and processing technology. There are two essential steps in achieving the required accuracy of parasitics extraction: the construction of geometric objects representing the interconnects and electrical simulation performed on these objects to extract parasitics. The platform and modules developed under this research project are well positioned for this application of interconnect analysis. In fact, an example of capacitance extraction for a 6T SRAM cell was quickly demonstrated [1] using geometry-based physical modeling for construction of complex bit and word lines, level set method for surface mesh and third-party tools (Fastcap and Fasthenry from MIT) for

electrical parameter (in this case capacitance) extraction. The same packaged approach, meaning use of a suite of tools, has also been used for the inductance extraction of interconnect in planar digital circuit technology [9] and bonding wire in RF device/circuit [2] . In the latter case, a fast way of constructing 3D objects is proposed.

## 3.3 Specific Module Development Efforts

### 3.3.1 Geometry Modeling

Geometric modeling capabilities are central to the task of building computational prototypes, especially where traditional approaches to geometric construction have proved inadequate for simulation of systems due to both complexity of the models and underlying technology dependences. Modeling of etching and deposition processes has been a particularly challenging area in terms of geometric modeling. Using level set methods [18, 14], such processes can be modeled with greater ease and simplicity compared to alternative methods that attempt to track surface evolution [19]. However, to date there continue to be bottlenecks and performance penalties in level set implementation due to the need for high resolution grid near moving boundaries and development of efficient mechanisms that integrate the general level set material interface tracking scheme with solid modeling functions. Among the many modeling issues to be addressed, this work focused on grid management, efficient data structures, and the use of parallel techniques for unstructured grids. Compact data structures for localized level set grids, which also reduce computation, have been developed in the context of the system architecture outlined above. Due to the ending of the ParaSCOPE project, a full scale demonstration and testing of these algorithmic aspects for the geometry work was not completed. However, as noted above under applications, the DARPA supported Composite CAD program has become the recipient of results from this project and is the venue for further testing.

Work continues in enhancing the solid modeling capabilities available through the integrated Tcl-based interface. Methods for constructing, querying, and performing Boolean operations on discrete geometries based on the Shapes modeling kernel (from XOX Inc.) have been integrated into the solid modeling module and are now being evaluated in the context of MEMS applications. Progress continues (under Composite CAD) in creating a

common interface to provide generic access, where appropriate, to multiple solid modeling kernels so that users can invoke specific modelers through generic commands. This solid modeling interface mirrors the abstraction provided by the sparse matrix interface also developed as part of this project, which encapsulates iterative and direct matrix solution capabilities.

A minimal (prototype) library of visualization scripts has been developed based on the cross-platform, publicly available Visualization Toolkit (Vtk). These scripts are integrated along with the level set and solid modeling modules, such that objects for visualization are retrieved from the shared data repository.

These efforts continue to move the research prototypes, using PROPHET and ALAMODE as demonstration vehicles, towards the overall goal of reusable modules for computational prototyping. The documentation and examples of typical model development and rapid prototyping for new physical systems are included in subsequent sections.

### 3.3.2 Solvers and Parallelization

The initial scope of this project included heterogeneous solver modules–ALAMODE (Stanford) and PROPHET (Bell Labs)–with ambitious goals to demonstrate a common (standard) parallel computation infrastructure. In the context of DARPA redirecting its efforts away from parallel computing and based on coordinating discussions with program managers Dr. Robert Parker and Sonny Maynard, the scope of this subtask was scaled down and redirected towards making the solver capabilities more broadly suited for rapid model prototyping of new applications, especially ones of interest to DoD. Hence, the following discussion briefly highlights areas of enhanced solver capabilities and the applications that can now be addressed based on these enhancements.

In quarterly progress reports, the accomplishments in creating a parallel solver infrastructure for PDE-based numerical applications were discussed. Namely, a common parallel linear solver library, supporting both iterative (PETSc [20]) and direct (DMF [21]) packages, was prototyped and integrated into ALAMODE. Based on the MPI message-passing library and the METIS graph-decomposition library, matrices were decomposed and handed to separate processors so that the linear solution can be carried out in parallel. This library

was delivered to IBM as an additional beta test site and integrated into their FIELDAY device simulation tool. Although some limited scalability has been demonstrated by only distributing the linear solution, to approach optimal scalability would also require decomposition of the mesh so that the matrix evaluation and assembly could be carried out in parallel. Due to the shift in focus from parallelization toward rapid prototyping, mesh decomposition and parallel evaluation/assemble were not implemented in ALAMODE.

The migration to the PETSc package, in concert with a common interface for both direct and iterative parallel linear solver access by heterogeneous tools provided an important step forward. Under a prior DARPA contract, SPRINT-CAD, the application-driven power of several classes of parallel solvers had been demonstrated [21, 22]. This work had also shown the essential trends in the problem scalability. Nonetheless, the physical complexity of the PDE-based models, including both the number of equations and their nonlinear coupling, had not been pushed as hard as the parallelization.

Hence, the following results from the ParaSCOPE projects shifted the emphasis towards model complexity and the rapid prototyping of new PDE-based systems. An essential feature of models developed for semiconductor applications is the ability of solving different equations for different variables in different material regions. The boundary conditions for variables across the material interface or at the surface of the simulation region are flexible and robust. Specifically, the following device simulation capabilities have been prototyped and demonstrated using the PROPHET module:

1. Enhancements in physical models for CMOS scaling such as inclusion of quantum mechanical (QM) effects and support for heterostructures are of major benefit to the mainstream electronics and DoD emerging applications in high-speed and wireless communications applications. The results of this project have demonstrated an excellent integration strategy for coupling both the QM effects of channel confinement and new formulations for tunneling current. This later model development was done jointly with Dr. Mario Ancona of NRL and has resulted in several journal and conference publications (e.g., see [8]).

2. Electrothermal simulation capabilities that combine the solution of thermal diffusion

equation and semiconductor equations–including multi-layer heterostructures–have been implemented and tested. Lattice temperature dependence of physical parameters such as the bandgap and carrier mobility are taken into consideration. Examples demonstrated to date include the electrothermal simulation of 3D SOI MOSFET and thermal simulation of a full chip made of SOI technology (of direct relevance to the SIA/FRC) [10].

3. On the framework side, many improvements has been made on user interface issues including: input scripting, syntax checking, and warnings about improper specification of device structure such as the electrodes. Graphical postprocessing of simulation results has been improved, based on public domain software, to provide 1D (xmgr), 2D (xgraph), and 3D (OpenDX) plotting capabilities.

# 4 Technology Transition of Results

The above demonstrations in support of rapid prototyping using PROPHET have dominately emphasized device modeling applications, including new challenges in interconnects and related substrate parasitic effects. Section 5 goes into a more extended technical demonstration of results. The ALAMODE module was also used and applied in the context of the overall ParaSCOPE projects. As noted above in the summary discussion of parallelism, ALAMODE (as well as PROPHET) was modified to interface with the parallel linear solvers (direct and iterative) and limited benchmark testing was presented in previous quarterly reports. However, there were no major breakthroughs or benchmarks that surpassed previous achievements under the SPRINT-CAD project. By contrast, the ALAMODE module did break new ground in process simulation of extremely complex physical systems.

Overall documentation of the the ALAMODE (A LAyered MOdel Development Environment) module is contained in the Ph.D. thesis of Dr. Daniel Yergeau [17]. Again, owing to the redirection of DARPA interests away from both parallel computing and microelectronics technology design, the ParaSCOPE emphasis was shifted towards the PROPHET module, specifically because of greater application-pull relevance to DoD. However, it is useful for completeness to mention sample landmark accomplishments based on using ALAMODE as

a rapid prototyping tool.

Specifically, the scaling of MOS devices requires design of thermal processing cycles to meet an ever narrowing window of simultaneous and opposing requirements of minimizing dopant diffusion to reduce junction depth while making enough of the dopant electrically active to obtain the desired electrical performance from the device. For the past half dozen years, the sessions devoted to this topic at IEDM have reflected intense interest in thermal process modeling in support of MOS scaling. The physical effects involved in dopant diffusion and activation during thermal processing are very complex, and can require PDE-based models composed of tens or even more of tightly coupled PDEs with complex boundary conditions to be solved in order to simulate the diffusion and activation of dopants during a thermal cycle. Several technical contributions, presented at IEDM, use ALAMODE as an essential tool for model development and validation [23, 24]. In both cases, the flexibility to easily input extreme sets of PDE's for new models and the numerical robustness in achieving useful and accurate results gives excellent testimony to the ParaSCOPE goal of rapid prototyping. The interested reader is referred to the Ph.D. thesis [17] as well as the ALAMODE User's Guide and examples available through web access [25].

Geometry modeling and the overall framework that supports the ParaSCOPE computational approach continue to be used and transferred in support of new applications, particularly in MEMS simulation and modeling as part of the DARPA (ETO)–sponsored Composite CAD program. The light weight "tool control" environment (using the Tcl language) with associated "wrappers" facilitates the ability to move forwards toward Composite CAD goals of software inter-operability.

In the context of new government- and industry-funded initiatives, the PROPHET technology (including the underlying solvers, key to efficiency for large, 3D problems) is finding growing applications. Example areas that are of special relevance to DoD include the following: heterostructure simulations (that could include opto-electronics), modeling of thermal effects related to reliability, and even totally new interests in application of TCAD modules to modeling biological systems [28]. Moreover, in the wireless communications domain, new technologies of relevance include material systems such as SiGe, SOI, and ongoing com-

pound material devices activities are now being addressed using the PROPHET simulation tool.

As noted in the PROPHET example related to quantum mechanical device effects [8] and the ALAMODE example related to process technology scaling[23], the ParaSCOPE goals of providing a Shared Computational Prototyping Environment have been in large part realized. Namely, these modules have supported a diverse and geographically distributed set of technical collaborations in achieving major advances in technical understanding of increasingly complex physical systems. The joint efforts with NRL are especially interesting and important. The work of Dr. Mario Ancona at NRL and industry co-workers, previously at ARO, had pushed forward semiconductor modeling theory but lacked the powerful computational infrastructure of PROPHET that was needed to demonstrate the power and applicability of the density gradient (DG) model. This collaborative work, involving team members from Stanford, Bell Labs, NASA/Ames, and NRL, has broken new ground in modeling and demonstrated major benefits and impact to future scaling.

There are several ongoing aspects of technology transition of the simulation tools and models coming from this project, both in support of further collaborative efforts and in moving the technology beyond the research stage. The ALAMODE module is quite useful for specific sets of applications as illustrated above and both the documentation and binaries are provided through the web (`http://www-tcad.stanford.edu/tcad/programs/alamode.html`). . The PROPHET module, initially developed by Bell Labs, has generated sufficient academic and industrial interest for follow-on development that both paths are being pursued. Under the auspice of NSF within both the NCSA (`http://www.ncsa.uiuc.edu/`) and Computational Electronics (`http://www.ceg.uiuc.edu/descartes.htm`) communities, there are ongoing efforts to enhance and utilize the modeling capabilities for collaborative research–the theme of both the NSF centers cited above. On the industrial side, while there is no formal transition plan to report as a conclusion, Bell Labs provides both a direct end user licensing agreement for PROPHET and has also commercially licensed the tool for further development and productization.

# 5 Script-based Simulation Using PROPHET

**Authors: Dr. Zhiping Yu and Dr. Daniel Yergeau**

PROPHET is an excellent example of an emerging class of simulation tools: software that permits end users to compactly and simply specify the models to be simulated. In the case of PROPHET, the models are specified as systems of partial differential equations (PDE's) which can be input to the simulator in a script form by assembling a description of the PDE system out of equation-building components (operators) drawn from a library. The scripting paradigm provides an intermediate implementation mechanism between traditional model implementation techniques, such as coding of FORTRAN or C "element" routines to be integrated into a PDE solver framework (e.g. ProFLEX [29]), and general PDE manipulation environments, such as Mathematica, which are too inefficient to be used for numerical simulation of complex models on real-world structures. The scripting approach used in PROPHET insulates the model developer from programming details associated with discretization and linearization, eliminating the need for programming skills and expertise in numerical analysis, a mixture of skills that most model developers do not possess. Furthermore, because the reusable operators in the library bind directly to efficiently coded compiled code, there is little to no performance penalty compared to equivalent, hand-coded models.

This section presents and discusses the detailed implementation of several advanced TCAD device models. The rapid prototyping of these models was made possible by the development of the infrastructure and operator library performed under ParaSCOPE.

## 5.1 Simulation of Bulk MOSFET

This example shows the simulation of bulk MOS device with a source/drain (S/D) extension. The device structure is shown in Fig. 1 and has a poly gate length of 0.6 $\mu$m and a channel length of 0.4 $\mu$m. The junction depth is about 70 nm for the extension and 120 nm for the S/D junctions. Other device structure parameters can be found in the figure.

The simulated terminal $I-V$ characteristics are shown in Fig. 2. Note that in the present form of the code, the carrier mobility degradation due to the transverse electric field has not

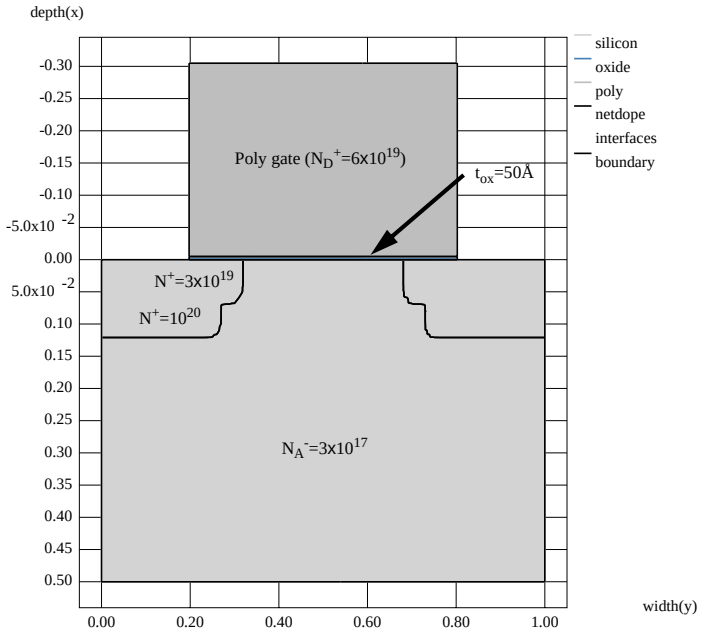



Figure 1: Device structure for a MOSFET with poly gate

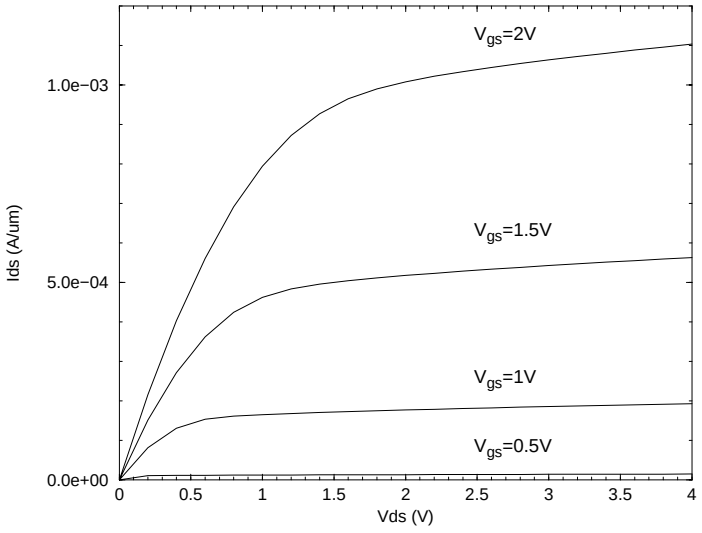



Figure 2: Simulated $I_d - V_{ds}$ characteristics under various gate bias

been implemented, so the current value may not be accurate. The same situation applies to the SOI example to be discussed in the next section.

We also compared the CPU time for using both direct and iterative linear solution methods. The results are: 428 sec. for a direct factorization and 174 sec. for the iterative method. The number of variables solved is $3 \times 2651 = 7953$. For each simulation, $V_{gs} = 2$V and $V_{ds}$ is ramped up from 0 to 4V with bias increment of 0.2V.

## 5.2 Electrothermal Simulation of SOI MOSFET

This example shows the electrothermal simulation of an SOI MOSFET. In addition to the electrical variables, $\psi$, $n$, and $p$, the lattice temperature temperature, $T_L$, is also solved. Thus, in the semiconductor region of the structure, there are four solution variables whereas in the insulating layers only $\psi$ and $T_L$ are to be solved. These four variables are governed by Shockley semiconductor equation set, which consists of Poisson's and carrier continuity equations, and thermal diffusion equation (see Section 5.2.2).

### 5.2.1 The Structure

The structure of the device under simulation is shown in Fig. 3. This SOI $n$MOSFET has a surface channel length of $0.4\,\mu$m (distance between S/D tips, i.e., extensions). The gate length is $0.44\,\mu$m. The source and drain regions fill the entire thickness of the silicon thin film (active region), and near the bottom of the film (next to the buried oxide layer), the source and drain are separated by $0.7\,\mu$m in order to minimize the off-state leakage current. The silicon substrate has a thickness of around $2\,\mu$m and is doped with $p-$type impurity to $5 \times 10^{14}\,\mathrm{cm}^{-3}$. The buried oxide layer on top of the substrate has a thickness of $0.2427\,\mu$m. The active silicon thin film on top of the buried oxide layer has a thickness of $0.08\,\mu$m (800Å) and is located at $X = 0$ to $0.08\,\mu$m. The gate oxide thickness is $0.01\,\mu$m or 100Å. The geometry of the device is defined in the statements of `grid` and `deposit` of file `soi2d.pf`.

The simulation region is discretized using two meshes: one generated from PROPHET script (Fig. 4.a), and the other is from a mesh generator, CAMINO, developed at Stanford (Fig. 4.b) [11].

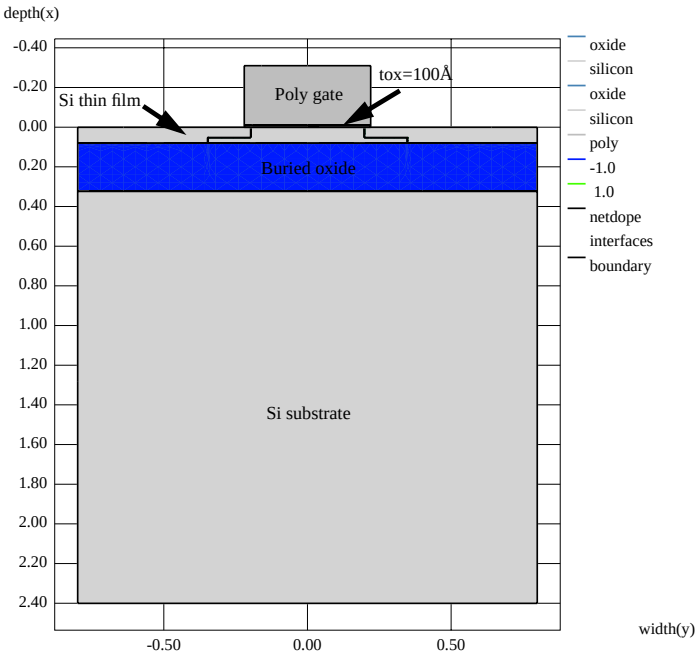
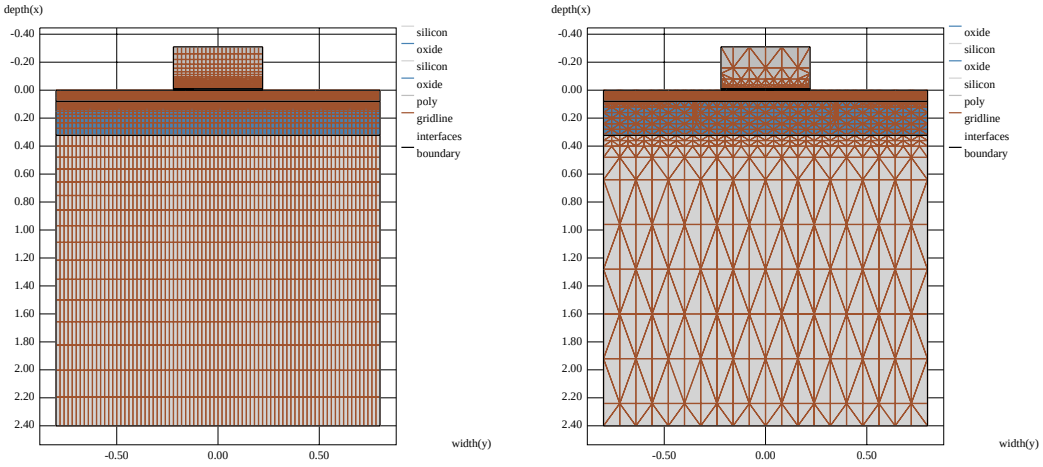



Figure 3: SOI device structure simulated.




(a) Mesh from PROPHET script (# of nodes: 5702).

(b) Mesh from CAMINO (# of nodes: 2499).

Figure 4: Mesh for SOI device

### 5.2.2 The Physical System

The physical system consists of electrothermal equations, simplified to steady state. The electrical part of the analysis is governed by Poisson's eq. and carrier continuity eqs. with physical parameters, such as mobility and bandgap, that depend on the lattice temperature. The thermal part of the analysis is governed by the thermal diffusion eq. and its interaction with electrical part is through a heat generation source modeled by the Joule heat ($\boldsymbol{J}\cdot\boldsymbol{E}$), where $J$ is carrier current density and $E$ is the electric field.

The complete set of PDEs for the steady state is as follows:

$$\begin{aligned}
\nabla\cdot(-\epsilon\nabla\psi) &= q\left(p-n+N_D^+-N_A^-\right) && (1)\\
0 &= -\nabla\cdot\boldsymbol{F}_n - r && (2)\\
0 &= -\nabla\cdot\boldsymbol{F}_p - r && (3)\\
0 &= -\nabla\cdot(-\kappa\nabla T_L) + \boldsymbol{J}\cdot\boldsymbol{E} && (4)
\end{aligned}$$

where Eq. (1) is Poisson's equation. The dielectric constant $\epsilon$ may be the function of space, dependent upon the material and possibly the mole fraction if the material is a ternary/quaternary compound semiconductor such as $\mathrm{Ga}_x\mathrm{In}_{1-x}\mathrm{As}$ (see latter section). Eqs. (2-3) are electron and hole continuity equations, respectively, where $\boldsymbol{F}_c$ is the carrier flux (subscript $c$ for either $n$ or $p$) and $r$ is the net recombination rate (i.e., the recombination minus generation rate) of electron-hole pairs (hence the same $r$ for both electron and hole continuity equations). Eq. (4) is the thermal diffusion equation at the steady state, in which the thermal conductivity $\kappa$ is the function of both material and lattice temperature $T_L$[1]. The heat generation source in this equation is due to the Joule heat only, which is equal to $\boldsymbol{J}\cdot\boldsymbol{E}$ where the current density $\boldsymbol{J} = q(\boldsymbol{F}_p - \boldsymbol{F}_n)$.

The carrier flux according to the drift-diffusion (DD) transport model is expressed as

$$\boldsymbol{F}_c = D_c\nabla c \mp \mu_c c\boldsymbol{E} = D_c\nabla c \pm \mu_c c\nabla\psi \qquad (5)$$

where the top sign in $\mp$ or $\pm$ is for electrons ($c = n$) and bottom one for holes ($c = p$).

These equations are automatically discretized and linearized by PROPHET. Although PROPHET provides multiple geometric discretizations for the mathematical divergence

[1] $\kappa(T_L) = \kappa_0\,(T_L/300K)^{-\alpha}$. For details, refer to [32].

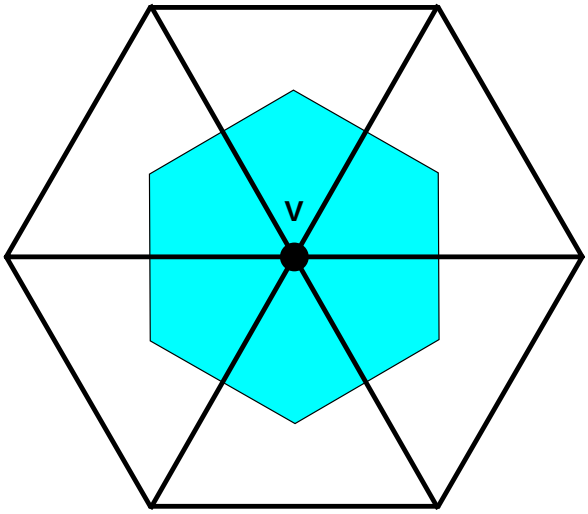



Figure 5: Finite box cell surrounding node $V$ in a triangular mesh. Closed integral is estimated using segments defining the shaded area.

operator ($\nabla\cdot$), the finite box discretization [30] is the preferred discretization technique because of its efficiency, robustness, and ability to handle unstructured meshes. The finite box discretization (FBM) eliminates the divergence operator by using Green's theorem or the divergence theorem to convert the strong form of the PDE shown above into a closed integral.

$$\oint_C \boldsymbol{F}\cdot\boldsymbol{n}ds = \int\int_R \boldsymbol{\nabla}\cdot\boldsymbol{F}dA \tag{6}$$

The integral is then estimated by evaluation of the flux through segments defining a finite box surrounding a mesh node (see Figure 5).

The FBM requires a further discretization to evaluate fluxes through each of these segments, i.e., the flux between two mesh nodes aligned with a mesh edge. The discretization of these fluxes must be treated very carefully by the simulator because standard flux discretizations are not stable for fluxes containing diffusive and convective terms, such as is the case with the carrier continuity equations. For these equations, using a standard simple flux discretization, such as finite differences, would result in an unacceptable numerical instability when the potential drop across the edge exceeds, say, two times of thermal volt-

age. The usual stabilization technique used in the computational fluid dynamics (CFD) community applies "upwinding," which adds a stabilizing artificial diffusion to the equation to compensate for the strong advective flux based on the element's size. However, the upwinding approach to stabilization has not been very successful for the coupled device equations. A more elegant flux discretization method, which maintains both accuracy and stability, is the so-called Scharfetter-Gummel (SG) scheme and is almost universally used in any device simulation. In the SG discretization, the flux along a mesh edge from points 1 to 2 is computed using

$$F_{1\to 2} = -\frac{D_c}{h}[c_2 B(\pm\Delta\psi/V_t) - c_1 B(\mp\Delta\psi/V_t)] \tag{7}$$

where $V_t = kT_L/q$ is the thermal voltage, $D_c = V_t\mu_c$, $\Delta\psi = \psi_2 - \psi_1$, and $B$ is the Bernoulli function which is defined as follows and has the following asymptotic behavior:

$$B(x) = \frac{x}{e^x - 1} = \begin{cases} 1 & x \to 0 \\ 0 & x \to \infty \\ -x & x \to -\infty \end{cases} \tag{8}$$

Again the top sign of $\pm$ and $\mp$ in Eq. (7) is for electrons and bottom one for holes.

Since no carriers exist in the oxide region of the device, the carrier continuity equations do not need to be solved in that region. The boundary condition at the $Si/SiO_2$ interface is that the electrostatic potential, $\psi$, and lattice temperature, $T_L$, are continuous while the normal flux for carriers vanishes, i.e., is equal to zero, across the interface. The boundary conditions for the device are as follows:

1. At the electrode, the Ohmic contact boundary condition is applied to the electrical part of the system, meaning that both the charge neutrality and thermal equilibrium conditions are imposed on $\psi$, $n$, and $p$. The Fermi-level for both types of carriers at the contact is determined by the applied bias.

2. If no thermal contact is specified separately, the electrode is also used as the Dirichlet boundary condition for $T_L$.

3. On all other surfaces of the device, the zero normal flux for solution variables ($\psi$, $n$, $p$, and $T_L$) is assumed. This type of boundary condition is also called the natural boundary condition.

### 5.2.3 PROPHET Implementation

Before the entire electrothermal system is implemented in PROPHET, the issue involving implementation of $\boldsymbol{J} \cdot \boldsymbol{E}$ has to be addressed. In order to use the operators available in PROPHET, the following mathematical transformation is done to take advantage of the fact that $\nabla \cdot \boldsymbol{J} = 0$ at the steady state.

$$\begin{aligned} \boldsymbol{J} \cdot \boldsymbol{E} &= \boldsymbol{J} \cdot (-\nabla\psi) = -\nabla \cdot (\psi \boldsymbol{J}) + \psi \nabla \cdot \boldsymbol{J} \\ &= -\nabla \cdot (\psi \boldsymbol{J}) = -\nabla \cdot [q\psi(\boldsymbol{F}_p - \boldsymbol{F}_n)] \end{aligned} \tag{9}$$

Thus Eq. (4) becomes

$$0 = -\nabla \cdot (-\kappa \nabla T_L) - \nabla \cdot [q\psi(\boldsymbol{F}_p - \boldsymbol{F}_n)] \tag{10}$$

The first step to implement the above physical system, Eqs. (1-4), in a PROPHET script is to move all the terms in the equations to the same side, resulting in a system where every equation is set equal to zero. Then, one needs to find a combination of a geometric term (`geoterm`) and a physical term (`phyterm`) to be used for each term in the equation. The available geometric terms and physical terms are listed in [31].

The result is as follows:

$$\underbrace{-\nabla\cdot}_{\substack{\text{ndiv\_fbm.}}} \underbrace{(\epsilon\nabla\psi)}_{\substack{\text{diffusion}\\ \text{(epsilon,psi}\\ \text{|psi)}}} + \underbrace{q\left(n - p + N_A^- - N_D^+\right)}_{\substack{\text{volume.nscd(}\\ \text{electrons,}\\ \text{holes,}\\ \text{netdope}\\ \text{|psi)}}} = 0 \tag{11}$$

$$\underbrace{-\nabla\cdot}_{\substack{\text{ndiv\_fbm.}}} \underbrace{(-\boldsymbol{F}_n)}_{\substack{\text{ncflux(}\\ \text{psi,}\\ \text{electrons,}\\ \text{tl,nmob0}\\ \text{|electrons)}}} + \underbrace{r}_{\substack{\text{volume.srh(}\\ \text{electrons,}\\ \text{holes,ni,}\\ \text{taun,taup}\\ \text{|electrons)}}} = 0 \tag{12}$$

$$\underbrace{-\nabla\cdot}_{\substack{\text{ndiv\_fbm.}}}\ \underbrace{(-\boldsymbol{F}_p)}_{\substack{\text{ncflux(}\\ \text{psi,holes,}\\ \text{tl,pmob0}\\ \text{|holes)}}}\ +\ \underbrace{r}_{\substack{\text{volume.srh(}\\ \text{electrons,}\\ \text{holes,ni,}\\ \text{taun,taup}\\ \text{|holes)}}}\ =\ 0 \tag{13}$$

$$\underbrace{-\nabla\cdot}_{\substack{\text{ndiv\_fbm.}}}\ \underbrace{(\kappa\nabla T_L)}_{\substack{\text{diffusion(}\\ \text{kappa,}\\ \text{tl|tl)}}}\ +\ \underbrace{-\nabla\cdot}_{\substack{\text{ndiv\_fbm.}}}\ \underbrace{[-q\psi(\boldsymbol{F}_p-\boldsymbol{F}_n)]}_{\substack{\text{npsij(psi,}\\ \text{electrons,holes,}\\ \text{tl,nmob0,}\\ \text{pmob0|tl)}}}\ =\ 0 \tag{14}$$

where the phrase under the lateral brace is the corresponding PROPHET script implementation for the term braced. Each scripted term (i.e., phrase) consists of four parts:

- The word before the dot (`.`) is the `geoterm`.
- The word after the dot is the `phyterm`.
- The arguments in the parentheses before the vertical bar (`|`) are the input parameters to the `phyterm`.
- The argument(s) in the parentheses after the vertical bar represents the equation(s) to which the entire term is applied. The equation is represented by the corresponding solution variable (i.e., `sysvar`) for which the equation governs (e.g., `psi`, $\psi$, for the Poisson's equation and `electrons`, $n$, for electron continuity equation, etc.).

To help remember the mnemonic names of the operators, here are some explanations:

- `ndiv_fbm`: negatively signed divergence operator in FBM implementation.
- `nscd`: negative space charge density
- `ncflux`: negative carrier flux
- `npsij`: negatively signed product of $\psi$ and $\boldsymbol{J}$

The input script for defining the electrothermal analysis of SOI structure is listed below. The exact meaning of the `geoterms` and `phyterms` used in this system are discussed in documentation available from `http://www-tcad.stanford.edu/~prophet/`.

```
system name=electrotherm
+ sysvars=psi,electrons,holes,tl
+ term0=dirichlet.device_dirichlet_h(netdope,ni,nc,nv|psi,electrons,holes)\
@{silicon/source,silicon/drain,silicon/substrate,oxide/gate}
+ term1=ndiv_fbm.lapflux(psi|psi)@{silicon,oxide}
+ term2=nodal.nscd(electrons,holes,netdope|psi)@{silicon,oxide}
+ term3=ndiv_fbm.ncflux(psi,electrons,tl,nmob0,nin,edge|electrons)@{silicon}
+ term4=ndiv_fbm.ncflux(psi,holes,tl,pmob0,nip,edge|holes)@{silicon}
+ term5=dirichlet.therm_dirichlet(tl|tl)@\
{silicon/source,silicon/drain,silicon/substrate,oxide/gate}
+ term6=ndiv_fbm.diffusion(kappa,tl|tl)@{silicon,oxide}
+ term7=ndiv_fbm.npsij(psi,electrons,holes,tl,nmob0,pmob0|tl)@{silicon}
+ term8=constraint.continuity(psi|psi)@{oxide/silicon}
+ term9=constraint.continuity(tl|tl)@{oxide/silicon}
+ nterm=10
```

### 5.2.4 Solution Method

The 2D device structure is discretized using a tensor product mesh, i.e., a regular rectangular mesh without the terminated mesh line in the interior of the simulation region. For electrothermal system, there are four solution variables for each node (in PROPHET, a node might be different from the grid or called point if a grid falls on material interface, in which case more than one node share the same point): $\psi, n, p$ and $T_L$.

For sufficiently large 2D and virtually all 3D simulations, using an iterative algorithm for the linear equation solution is usually less costly than a direct factorization, both in solution time and memory usage. In PROPHET script the iterative linear equation solver can be selected and tuned by setting the database entries as shown in the following statements:

```
dbase prefix=library/math/systems/default_numerical_parameters
```

```
dbase create name=method sval=iterative
dbase create name=accel sval=bicg
dbase create name=precon sval=ilu
dbase create name=maxfil ival=4
```

where the database entry `method` specifies either `iterative` or `direct` and for iterative solution method there are several parameters for selection and tuning of the interation and preconditioner to use for iterative linear solution. The `accel` database entry selects which iteration to use among:

- `bicg` – stabilized biconjugate gradient
- `cg` – conjugate gradient
- `cgs` – stabilized conjugate gradient
- `gmre` – generalized minimal residual.

For the asymmetric and poorly conditioned Jacobian matrices obtained from discretization of the device equations, only `bicg` and `gmre` are recommended. The `precon` database entry selects the preconditioner to be applied. The only recommended preconditioner is `ilu`, an incomplete lower/upper factorization. This preconditioner is tunable through the `maxfil` parameter, which specifies the amount of fill-in to permit in each row. With `maxfil=0`, only original sparse structure of the matrix will be used for the incompletely factored LU preconditioner. Increasing the value for `maxfil` does provide a better preconditioning and, hence, a more robust and better quality iterative linear solution. The computational and memory costs also increase with increasing `maxfil`, and there is usually an optimal value that can only be determined experimentally for a given mesh dimensionality and element type and the system of partial differential equations being solved.

There are two loops in solving a nonlinear equation system if the iterative method is preferred. An outer Newton-Raphson (NR) iteration loop which updates the linear equations (through change of Jacobian matrix and the residue) every time the solution is updated, and the inner loop which solves the linear equation iteratively. The parameter `maxNewton` is to

specify the maximum number of iterations in the outer loop for solving *nonlinear* equations. The default value is 10. When using the iterative method in solving the linear equations, this number usually needs to be increased (the outer loop, i.e., the NR iterations, often has a slower convergence rate once the iterative method is used for linear solver) relative to the direct method in solving the linear equations. In this example, it is set to 50. The maximum count in inner iterations (for solving linear eq.) is specified through parameter `maxIts` and the default value is 250 (for 2D and 3D simulations).

The convergence criterion for NR convergence is set via parameter `NewtonUpd` with default value of $10^{-5}$. The update norm is a measure relative change in values compared the value at the same node after the update is applied. There is also a second criterion which essentially puts an upper limit for the residue of the function before the iterations are considered converged.

The output from the execution of this example is a `.pas` file which saves the device structure and simulation results in a format which can be

1. reloaded into PROPHET for further simulation (using `load pas=file.pas` where `file` is the name of the file excluding the extension `.pas`)

2. converted to `.ucd` (for AVS's Unstructured Cell Data) or `.dx` (for IBM's Data Explorer) formats for detailed visualization and examination of the computed solutions.

### 5.2.5 Simulation Results for 2D SOI MOSFET

The simulated $I_d$ vs. $V_{ds}$ for $V_{gs} = 2$ and 3,V using both iterative and direct linear equation solvers for this SOI device is shown in Fig. 6. It can be seen that the results from using the direct solver is not as smooth. The exact reason is still under investigation.

## 5.3 1D MOS Capacitance Simulation

This example is to show how one can apply bias to a MOS capacitor by solving the Poisson's eq. only for finding the charge distribution within the current capabilities of PROPHET.

Several features of PROPHET used in this example are:

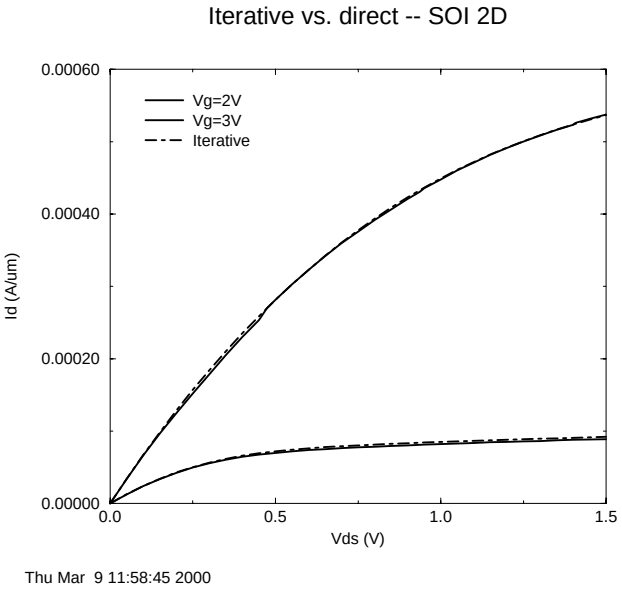



Figure 6: Simulated $I_d - V_{ds}$ for $V_{gs} = 3\,\mathrm{V}$ of SOI MOSFET

1. Use of M4 macro to define `foreach` facility and its use for ramping up the quasi-Fermi level which is equivalent to the bias.

2. The quasi-Fermi level for carriers is specified through a database entry `qf.fixed`.

The structure has a $p-$substrate ($N_A = 3 \times 10^{18}\,\mathrm{cm}^{-3}$), a gate oxide of thickness 34 Å, and an $n^+$ poly gate of active doping concentration $N_D^+ = 5 \times 10^{19}\,\mathrm{cm}^{-3}$.

The goal of the analysis is to find the capacitance in the deep accumulation region ($V_g = -4\,\mathrm{V}$) using quasi-steady state analysis, i.e., $C = \Delta Q/\Delta V$ where $\Delta V = 0.01\,\mathrm{V}$. To make sure that a converged solution can be found at the destined bias ($V_g = -4\,\mathrm{V}$), the bias is gradually applied from $V_g = 0$ towards $V_g = -4\,\mathrm{V}$ with initial increment of $0.025\,\mathrm{V}$ and eventually an increment of as big as $1\,\mathrm{V}$. This bias ramp-up is realized by the use of

```
foreach('vg', (0.05,0.1,0.15,0.2,0.25,0.3,0.35,0.4,0.45,0.5,0.55,0.6,\
0.7,0.8,0.9,1,1.5,2,3,4),
dbase prefix=/library/physics/poly/electrons
dbase create name=qf.fixed rval=-vg
```

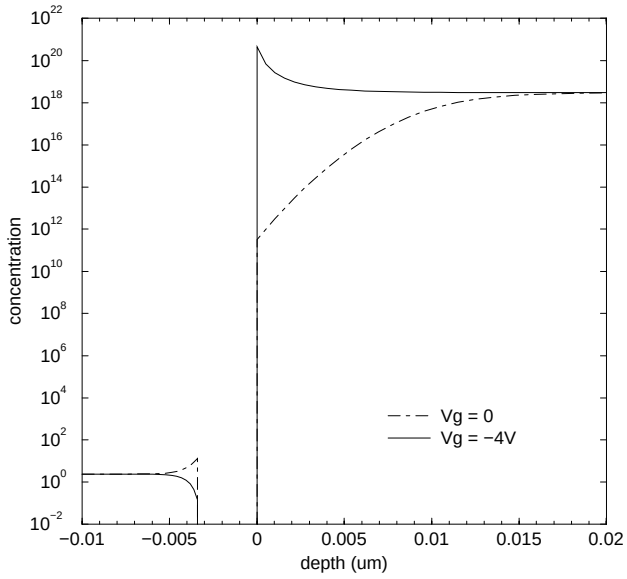



Figure 7: Simulated hole distribution at thermal equilibrium (dot-dash line) and at deep accumulation region ($V_g = -4\,\mathrm{V}$).

```
dbase prefix=/library/physics/poly/holes
dbase create name=qf.fixed rval=-vg
bias system=poisson elec=gate voltage=-vg
)
```

There is a post-processing procedure to extract the charge in the poly gate region (or in the substrate) based on the solutions, which are stored in file using statement of

```
dump sol outf=1dcap.sol
```

The simulated hole distribution at two different biases ($V_g = 0$ and $-4\,\mathrm{V}$, respectively) are shown in Fig. 7.

### 5.4 Heterostructure Simulation – $\mathbf{Ga_{0.47}In_{0.53}As}$–InP *pn* Diode

This example shows the simulation of heterostructure devices and in particular a $Ga_xIn_{1-x}As$-InP *pn* diode. The bandgap at the room temperature for $Ga_{0.47}In_{0.53}As$ is 0.72 eV and that

for InP is 1.347 eV. So this is a narrow bandgap material on $p-$side of the diode and wide bandgap on the $n-$side (refer to Fig. 8.b).

### 5.4.1 Physical System

For a heterostructure, all material parameters become function of material composition (in terms of mole fraction $x$ in this case).

We now look at the dependence of dielectric constant, $\epsilon$, on material property and composition.

$$\epsilon(x) = \epsilon_0 \frac{1 + 2\left[x\frac{\epsilon_1 - 1}{\epsilon_1 + 2} + (1-x)\frac{\epsilon_2 - 1}{\epsilon_2 + 2}\right]}{1 - x\frac{\epsilon_1 - 1}{\epsilon_1 + 2} - (1-x)\frac{\epsilon_2 - 1}{\epsilon_2 + 2}} \tag{15}$$

where $\epsilon_0 = 8.854 \times 10^{-14}$ F/cm is the dielectric constant in the vacuum, $\epsilon_{1,2}$ are the relative dielectric constants ($\epsilon_r$) for the binary material in the ternary compound when mole fraction $x = 1$ and 0, respectively. So for example, for $Ga_xIn_{1-x}As$, $\epsilon_1$ is that of GaAs ($\epsilon_r = 13.1$) and $\epsilon_2$ of InAs ($\epsilon_r = 14.55$).

We further consider the boundary condition for solution variables to meet at the material interface. First let us consider the simplest case that the variable is continuous across the material interface such as in the case of lattice temperature, $T_L$. The way this boundary condition is implemented in PROPHET is through one "geoterm" (i.e., geometric operator) `constraint` and one "phyterm" (i.e., physical operator), which defines the form of constraint imposed upon the solution variables under question. The geoterm dictates how the program assembles the equation for the solution variables concerned in the defined material region(s) including material interface, if so defined (this case). For geoterm `constraint`, the concerned solution variables involve two nodes, one for each material, at the same physical location across the interface. The program will assemble a control-volume equation (for finite box method, or FBM) for one of the two variables (during the space discretization, the program will designate one of the node variables associated with a point as the independent variable and the remaining node variables are related to this independent variable through the "constraint" as defined by the phyterm associated with the geoterm `constraint`. The phyterm defines an algebraic constraint among solution variables across the interface and is realized in a functional form. Note that, although the function form shares the syntac-

tic format for a regular flux expression, the `constraint` geoterm does not treat it as a flux-based across the interface, such as is the case with the `interface` geoterm.

For the lattice temperature, we simply have a continuity relationship

$$f = T_{L1} - T_{L2} = 0, \tag{16}$$

and can use the `continuity` phyterm in conjunction with `constraint`. For electrostatic potential across the material interface, one must apply the continuity of vacuum potential across the interface to obtain

$$f(\psi_1, \psi_2) = \psi_2 - \psi_1 - \frac{1}{2}\left[V_{t2} \ln \frac{N_{C2}}{N_{V2}} - V_{t1} \ln \frac{N_{C1}}{N_{V1}} + \frac{1}{q}(E_{g2} - E_{g1})\right] - \frac{1}{q}(\chi_2 - \chi_1) \tag{17}$$

where $V_t = kT_L/q$ is the thermal voltage, $N_C$ and $N_V$ are effective density of states for conduction and valence bands, respectively, $E_g$ is the bandgap, and $\chi$ is the electron affinity. The subscripts 1 and 2 represent quantities on each side of the interface.

The constraints for electrons and holes can be expressed as follows. Assuming that the quasi-Fermi levels for electrons and holes, respectively, are continuous across the interface, then one has

$$f(n_1, n_2) = -\psi_1 + V_{t1} \ln \frac{n_1}{n_{i1}} + \psi_2 - V_{t2} \ln \frac{n_2}{n_{i2}} \tag{18}$$

$$f(p_1, p_2) = -\psi_1 - V_{t1} \ln \frac{p_1}{n_{i1}} + \psi_2 + V_{t2} \ln \frac{p_2}{n_{i2}} \tag{19}$$

where $n_i$ is the intrinsic carrier concentration, which is equal to

$$n_i = \sqrt{N_C N_V} e^{-E_g/(2kT_L)} \tag{20}$$

Note that in the above equations (Eqs. (17-19)), except of the solution variables, $\psi$, $n$, and $p$, all other parameters are material parameters and thus remain constant during the solution process.

The net recombination rate, $r$, in Eqs. (2-3), can be the combination of different recombination mechanisms such as Shockley-Read-Hall (SRH), Auger, and radiative recombination, each of which can be expressed as the function of carrier concentrations.

### 5.4.2 The PROPHET Implementation

Now we look at the implementation of the argument for the divergence, such as $-\nabla\cdot(\epsilon\nabla\psi)$. If $\epsilon$ is a constant, i.e., not a function of space, then one can use the phyterm called `lapflux` which takes only one input parameter, i.e., the solution variable such as $\psi$. The dielectric constant $\epsilon$ will be retrieved from the database depending on the individual region. If instead, $\epsilon$ is the function of space as in the case for compound material, one should use the phyterm of `diffusion(epsilon,psi|psi)`. This operator is also named "drift". The reason for the name of drift for this particular flux term is not obvious and is explained here. The phyterm `drift(a,b|a/b)`, where `a/b` means either `a` or `b`, always performs the operation of gradient on `b` ($\nabla b$) but depending on whether $a$ or $b$ is the solution variable (indicated by the symbol beyond "`|`") it can represent either a diffusion flux (with `(a,b|b)`) where $a$ will be the diffusivity, which may be a position-dependent variable, or a drift flux (with `(a,b|a)`) where $a$ is the solution variable such as $n$ in the case of $\boldsymbol{F}_{n,drift} = \mu_n n\nabla\psi$ which can be represented by `drift(n,psi|n)`. The coefficient $\mu_n$ will be retrieved from the database using $a$ as a tag (i.e., lead). $\mu_n$ will have all the dependence as $n$ would have, such as the region dependence.

In summary, for the diffusion flux, if the diffusivity is only a material parameter (no region-dependency, no-position dependency) then one can use the phyterm `lapflux(d1|s1)` which has only one input parameter and one output parameter and `d1` and `s1` are for the same solution variable. But if the diffusivity is a position-dependent variable, one must use `diffusion(d1,d2|s1)`. If `d2` and `s1` are the same, then `d1` represents the diffusivity. On the other hand, if `d1` and `s1` are the same, the operator represents a drift term with mobility retrieved based on material, species (such as $n$ or $p$), and region.

### 5.4.3 Post-Processing the Simulation Data

PROPHET provides some mechanisms for users to post-process the simulation results. In principle, all `tmpvar`s and the mathematical expressions based on these variables can be saved and plotted using `field` variable. Here we show how it can be done using the example of a GaInAs-InP *pn* diode.

First one needs to instruct the program to save the `tmpvar` in the script file (normally

these temporary variables are discarded once their mission is completed):

```
dbase create name=/options/keeptmpvar ival=1
```

All `tmpvar` as defined in `tmpvars=` are then accessible to the users. In this example, those physical quantities as defined by `tmpvar` are: $\epsilon$ (`epsilon`), $E_g$ (`eg`), $\chi$ (`affinity`), $m_n^*, m_p^*$ (`mn, mp`), $N_C, N_V$ (`nc, nv`), $n_i$ (`ni`), $\kappa$ (`kappa`), $\mu_{n,0}, \mu_{p,0}$ (low field mobility, `nmob0, pmob0`), and $\tau_n, \tau_p$ (`taun, taup`).

Now we will see how to construct and display/save quantities based on the above physical parameters. An obvious yet significant application would be the drawing of the band diagram:

$$E_C = -q\psi + \frac{1}{2}\left(E_g + kT_L \ln \frac{N_C}{N_V}\right) \quad (21)$$

$$E_V = E_C - E_g \quad (22)$$

where units of eV are used for energy-related quantities. This is implemented in the script file as

```
field set=ec value="0.5*(eg+k*tl*log(nc/nv))-psi"
field set=ev value="ec-eg"
```

To plot, use

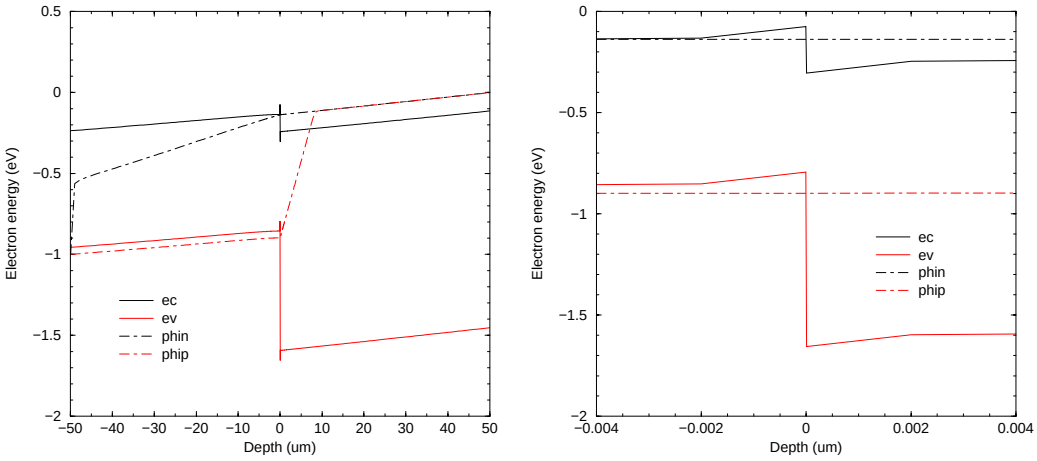
```
graph elem=ec log=0
graph elem=ev log=0 ax=f
```

The band diagram is shown in Fig. 8.

### 5.5 Cylindrical Coordinates and Surface Recombination Velocity

This example shows the photogeneration simulation without applying a bias. The structure is a cylindrical column with the principal axis being the $x-$axis. The radius of the cylinder is 700 $\mu$m and the height for the silicon part of the structure is 10 $\mu$m. On top of the silicon substrate there is an oxide layer of thickness 10 Å.

There are several unique features for this example of simulation:




(a) Band diagram at $V_d = 1\,\mathrm{V}$. Right half is $p-$ and left $n-$type.

(b) Detailed band diagram around the heterojunction at $V_d = 1\,\mathrm{V}$.

Figure 8: Simulation results for hetero-diode

1. Cylindrical coordinate is specified through statement

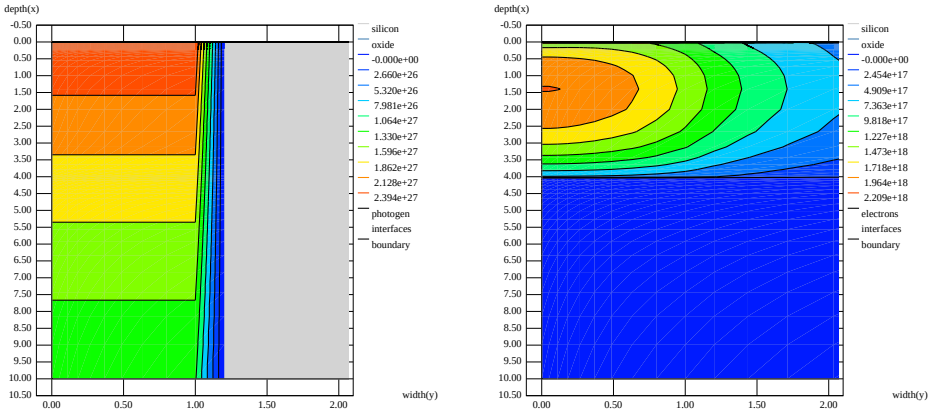
   ```
   dbase create name=/options/cylindrical sval=X
   ```

   The principal axis can be specified by assigning letter "X" (for x-axis), "Y" (for y-axis), and "Z" (for z-axis) to `sval=`. Note that both 1D and 2D (but not 3D) simulation are qualified for constructing a cylindrical coordinate system.

2. There exists carrier flow from silicon to oxide at the $Si/SiO_2$ interface through the specification of surface recombination velocity, or SRV, via the combination of geo and phy terms `interface.srv`.

3. The mobility model used is from Klaassen of Philips Lab, which includes the effect of carrier-carrier scattering.

4. To facilitate repeated simulation of the same structure for different physical material parameters (used, e.g., in optimization), some parameters are defined through




(a) 2D photo-generation contour plot for strength of $9.55 \times 10^5\,\mathrm{cm}^{-3}$.

(b) 2D electron contour plot under the same photo-generation source

Figure 9: Cylindrical structure under photogen

`define()` facility, such as in `define(srvS0, 1e4)` to define the SRV for both carriers.

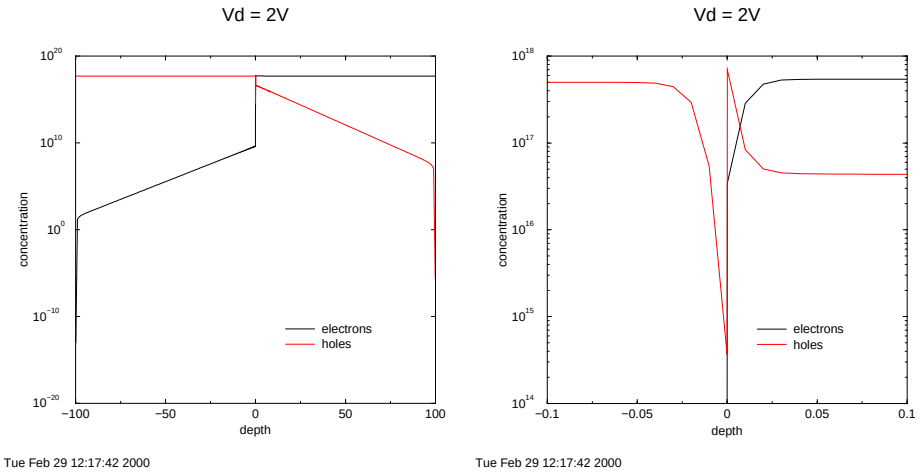
5. The photogeneration (`photogen`) is included in the carrier continuity equations as carrier generation source, which is specified through `field` statement. Since during the simulation, the strength of the photogeneration source is swept from weak to strong, the `foreach` facility is used.

Fig. 9 (a) shows the 2D contour plot for photogeneration rate at the center of the cylinder. Note that the strength of photogeneration rate is exponentially decayed from the surface of the structure (i.e., the thin oxide layer) to deep into the substrate. Fig. 9 (b) shows the electron distribution under the same stimulus as in Fig. 9 (a).

## 5.6 Thermionic Emission Through Heterostructure Interface

This example shows how the thermionic emission transport mechanism across the material interface can be incorporated in the heterostructure device simulation.

The simulated carrier distribution at $V_d = 2V$ are shown in Figs. 10.




(a) Electron/hole distribution at $V_d = 2\,\mathrm{V}$. Right half is $p-$ and left $n-$type.

(b) Detailed carrier distribution around the heterojunction at $V_d = 2\,\mathrm{V}$.

Figure 10: Simulation results for hetero-diode with thermionic emission across GaAs/GaInAs interface

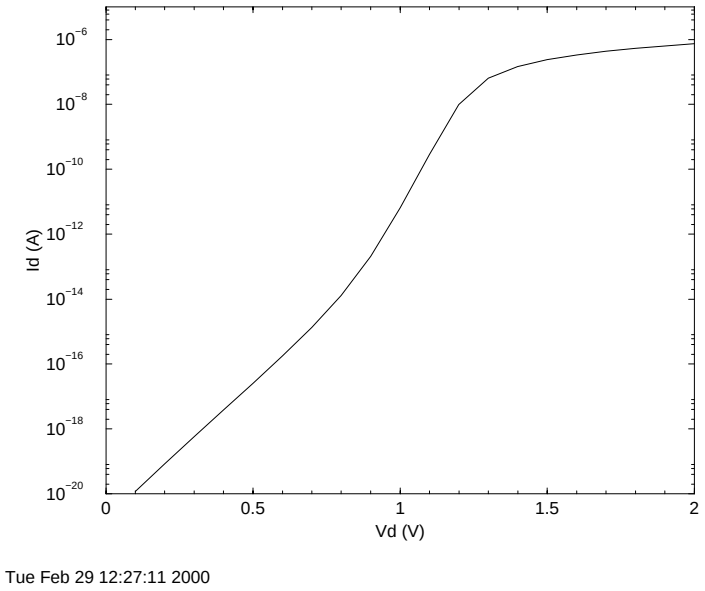



Figure 11: Simulated $I-V$ characteristics with thermionic emission included.

The thermionic emission through a hetero-interface is modeled by an operator (phyterm) called `tiemission`. It is used together with geoterm `interface` as in the following example for electrons:

```
term19=interface.tiemission(electrons,tl,nc,affinity,mn|electrons)@{GaAs/AlGaAs}
```

where the thermionic emission is to occur across the GaAs/AlGaAs interface. The input parameters for `tiemission` are carrier concentration, lattice temperature, effective density of states for electrons, the electron affinity, and the corresponding effective mass. The formulation for computing the electron flux due to the thermionic emission is as follows:

$$F_{1\to 2} = 2A\left(\frac{1}{m^*_{n1}}+\frac{1}{m^*_{n2}}\right)\frac{\overline{T}_L^2}{N_{C1}}\left(c_1 - c_2\frac{N_{C1}}{N_{C2}e^{(\chi_2-\chi_1)/k\overline{T}_L}}\right) \tag{23}$$

where $A$ is the Richardson constant, which is equal to 120 Ampere/cm$^2$-K$^2$ for free electrons and $\overline{T}_L = (T_{L1}+T_{L2})/2$.

The simulated $I-V$ characteristics are shown in Fig. 11.

## 5.7 Full Chip 3D Thermal Simulation

This example solves the thermal diffusion equation only given the distribution of the heat generation source across the chip. The size of the problem, however, is much bigger – the chip dimension is about $1.6 \times 1.9\,\mathrm{cm}^2$ in width and length and the number of distinct heat sources, representing logical functional blocks, is 28. For the simulation, the number of unknowns, i.e., lattice temperature, is 183954, which requires total memory space of around 2G bytes (the sum of available physical memory and swap space).

### 5.7.1 Structure

The chip is made using SOI technology and the functional block-level placement floor-plan is known as an input together with the chip cross-section which shows the thickness for individual layers including substrate and interconnecting/interleaving dielectric layers.

The chip, as simulated, has seven layers, the top and bottom layers are electrically insulating layers with small thermal conductivity (in the example one tenth of that for $Si_3N_4$ is used), served as the thermal resistive layers to model the thermal resistance between the chip and the environment (room temperature). From the actual dimensionality used in the example, the thermal resistance is $0.705\,\mathrm{K/W}$ on each face of the chip.

From the bottom layer up, they are (in sequence): silicon substrate ($500\,\mu$m thick), buried oxide ($0.4\,\mu$m), silicon thin film ($0.18\,\mu$m), 1st dielectric layer ($0.43\,\mu$m), M1 (1st metal, $0.48\,\mu$m, approximated using silicon material), and top thermal resistive layer ($30\,\mu$m thick with thermal resistivity of $0.0014\,\mathrm{W/K\text{-}cm}$) to model the effects of both other interconnect layers and thermal resistance between top of the chip and the surrounding environment. At present, the layer thickness for the top and bottom thermal resistive layers is adjusted in such a way that the simulated temperature in the active layer is in the range of targeted (i.e., correlated to the measured data) temperature rise in the power consumption level specified.

### 5.7.2 Heat Generation Source

The first step in the simulation is to mesh the chip, which we basically follow the placement layout of functional blocks. But, since the background grid before the blocks are placed

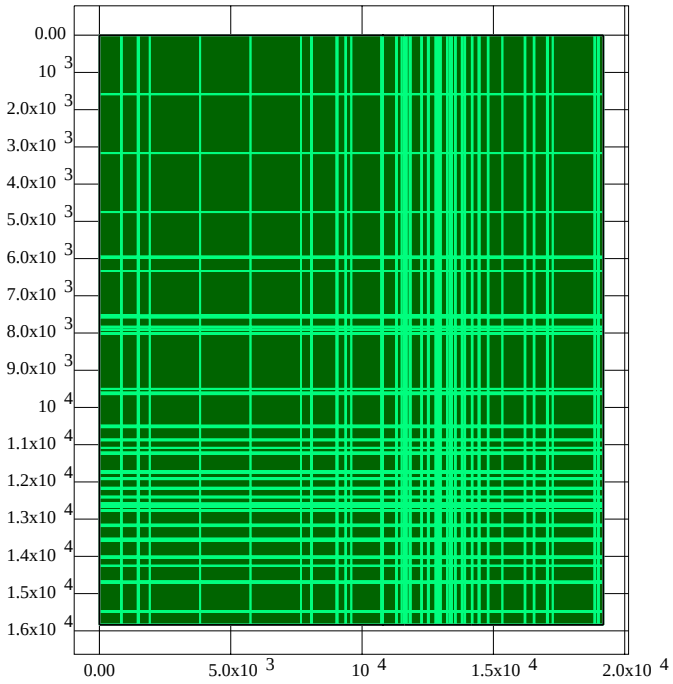



Figure 12: Top view of mesh for the CPU chip under thermal simulation

is rather coarse (in this example we are limited to only place 15 spacings each on both directions of the chip layout), in order to guarantee the heat generation is confined to the designated blocks, each block is encased by a mesh which is 20 $\mu$m wider from the frame of the block. The resulting mesh is shown in Figure 12

The final mesh size in terms of node count is $X \times Y \times Z = 37 \times 107 \times 79 = 303,992$. The power consumption is known for each functional block from the electrical simulation, and, in order to compute the strength (or intensity) of the heat generation source for each block, it is assumed that the heat generation is uniformly distributed in the active silicon thin film layer with the lateral dimension of the block size. For example, for a SOI technology with silicon thin film thickness of 0.18 $\mu$m (this example) and a functional block (`dcacheN` for upper data cache) of size $0.5965 \times 0.7228\,\mathrm{cm}^2$ and consumes power of 2.747 W, the heat generation rate within this block is $2.747/(0.5965 * 0.7228 * 0.18 \times 10^{-4}) = 3.54 \times 10^5\ \mathrm{W/cm}^3$.

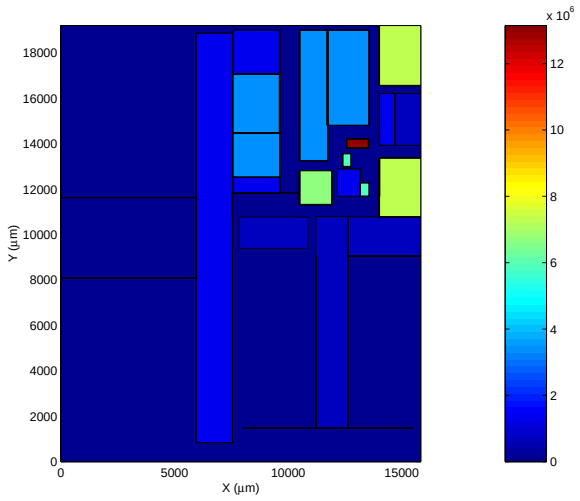



Figure 13: Heat generation source as block distribution across a CPU chip.

There are 28 heat generation blocks and they are specified using `field` card, each having an unique name and with generation rate (in units W/cm$^3$) and location (in 3D) specified. These 28 heat sources are then lumped into a single source (also a `field` called `hg`) which is used in the system definition. The total power consumption from these heat generation blocks is 62 W.

### 5.7.3 Rendering of Simulated Temperature

The simulation result, i.e., the lattice temperature distribution, is saved in a `.pas` file and can be further converted to a `.dx` file (in fact, one can save the `.dx` file directly from the input file which dictates the simulation). This `.dx` file can then be used for 3D rendering using IBM Data Explorer visualization program. We have also developed a customized program to probe the simulation results to generate point, line, and plane data to be plotted using `xmgr` (for 1D line plot) or `Matlab` (for 2D contour plot). For a detailed discussion and results, refer to [10].

The simulated lattice temperature is shown in Fig. 14.

The 3D rendering is shown in Fig. 15.

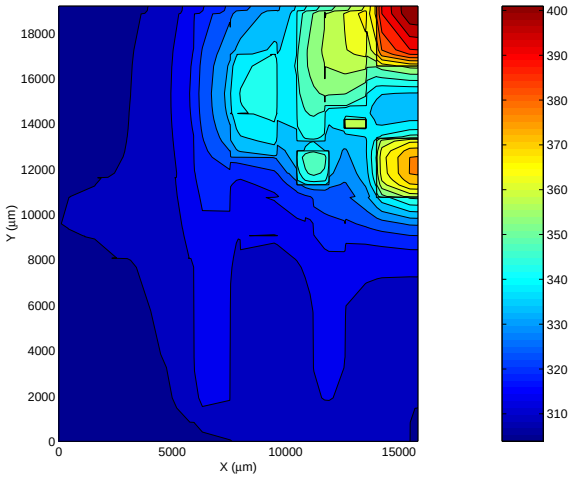
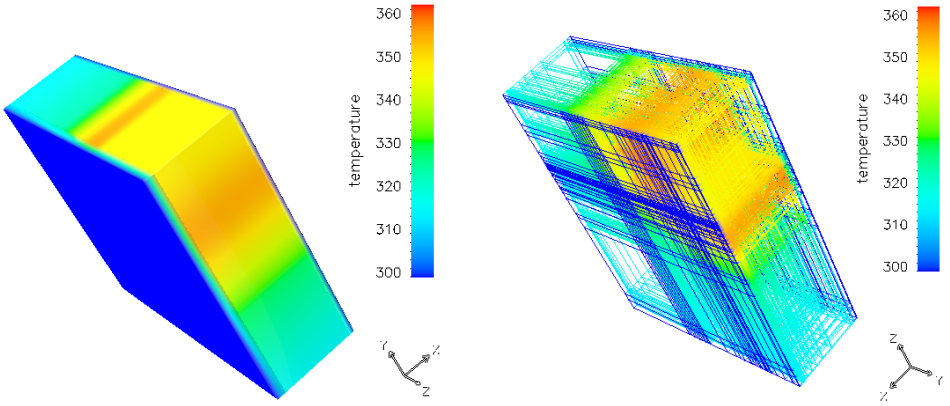



Figure 14: Simulated lattice temperature distribution across the CPU chip.




(a) Solid rendering of temperature distribution. The $X-$axis points from the surface to the back of the substrate.

(b) Wire representation of structural meshing. Note that the orientation is different from figure on the left.

Figure 15: 3D rendering of full chip thermal simulation

## References


[1] X. Qi, S. Shen, Z. Hsiau, Z. Yu, and R.W. Dutton, "Layout-based 3D solid modeling of IC structures and interconnects including electrical parameter extraction," SISPAD '98, p. 61, Lueven, Belgium, Sept. 1998.

[2] X. Qi, P. Yue, T. Arnborg, H.T. Soh, Z. Yu, R.W. Dutton, and H. Sakai, "A Fast 3D Modeling Approach to Parasitcs Extraction of Bonding Wires for RF Circuits," IEDM Technical Digest, pp. 299-302, Dec. 1998.

[3] C.S. Rafferty, Z. Yu, B. Biegel, M.G. Ancona, J. Bude, and R.W. Dutton, "Multi-dimensional quantum effect simulation using a density-gradient model and script-level programming techniques," SISPAD '98, p. 137, Lueven, Belgium, Sept. 1998.*

[4] K.C. Wang, C.A. Taylor, Z.-K. Hsiau, D. Parker, R.W. Dutton, "Level set methods and MR image segmentation for geometric modeling in computational hemodynamics," *Proc 20th Int'l Conf*, IEEE Engineering in Medicine and Biology Society, 1998, pp. 3079-3082.*

[5] K.C. Wang, D. Parker, R.W. Dutton, C.A. Taylor, "Level sets for vascular model construction in computational hemodynamics," Proc ASME Summer Bioengineering Conf, 1999, pp. 721-722.

[6] K.C. Wang, R.W. Dutton, C.A. Taylor, "Improving geometric model construction for blood flow modeling," IEEE Engineering in Medicine and Biology Magazine, vol. 18 no. 6, pp. 33-39, 1999.*

[7] D. Yergeau, K. Taniguchi, K. Lilja, T. Saito, and R. Dutton, "Physical model portability in a heterogeneous TCAD simulator environment," private communication, May 1999.*

[8] M. G. Ancona, Z. Yu, R. W. Dutton, P. J. Vande Voorde, M. Cao, and D. Vook, "Density-gradient analysis of tunneling in MOS structure with ultra-thin oxides," SISPAD '99, p. 235, Kyoto, Japan, Sept. 1999.*

[9] Xiaoning Qi, Bendik Kleveland, Zhiping Yu, S. Simon, Wong, Robert W. Dutton , and Tak Young, "On-chip inductance modeling of VLSI interconnects," ISSCC '00, San Francisco, Feb. 2000.

[10] Zhiping Yu, Dan Yergeau, Robert W. Dutton, Sam Nakagawa, Norman Chang, Shen Lin, and Weize Xie, "Full chip thermal simulation," ISQED (Int'l Symposium of Quality Electronic Design), p. 145 Santa Clara, CA, March, 2000.

[11] Tao Chen, *Automatic Mesh Generation for Semiconductor Process and Device Simulations Using a Generalized Octree Method*, Ph. D. thesis, Stanford University, to be published in 2000.

[12] N.M. Wilson, Z.-K. Hsiau, R.W. Dutton, and P.M. Pinsky, "A Heterogeneous Environment for Computational Prototyping and Simulation Based Design of MEMS Devices", SISPAD '98, September 2-4, 1998, pp. 153-156.*

[13] Z.H. Sahul, K.C. Wang, Z.-K. Hsiau, E.W. McKenna, and R.W. Dutton, "Heterogeneous Process Simulation Tool Integration," IEEE Transactions on Semiconductor Manufacturing, vol. 9, no. 1, pp. 35-58, Feb. 1996.

[14] Z.-K. Hsiau, *Boundary Movement in Semiconductor Etching and Deposition Simulation*, PhD thesis, Stanford University, 1997.†

[15] D.W. Yergeau, E.C. Kan, M.J. Gander, and R.W. Dutton, "ALAMODE: A Layered Architecture for Model Development," SISDEP '95, Sept. 1995, pp. 66-69.*

[16] G. Hobler, C.S. Rafferty, and S. Senkader, "A model of {311} defect evolution based on nucleation theory," SISPAD '97, Sept. 1997, pp. 73-76.

[17] D.W. Yergeau, *A Dial-An-Operator Approach to Simulation of Impurity Diffusion in Semiconductors*, PhD thesis, Stanford University, 1999.†

[18] J.A. Sethian, *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge University Press, 1996.

[19] Z.-K. Hsiau, E.C. Kan, J.P. McVittie, R.W. Dutton, "Physical Etching/Deposition Simulation with Collision-Free Boundary Movement," IEDM Technical Digest, pp.101-104, Dec. 1995.

[20] Satish Balay and William D. Gropp and Lois Curfman McInnes and Barry F. Smith, "Efficient Management of Parallelism in Object Oriented Numerical Software Libraries," Modern Software Tools in Scientific Computing, Birkhauser Press, 1997, pp. 163-202.

[21] B.P. Herndon, *A Methodology for the Parallelization of PDE Solvers: Application to Semiconductor Device Physics*, PhD thesis, Stanford University, 1995.

[22] N.R. Aluru, *Parallel and Stabilized Finite Element Methods for the Hydrodynamic Transport Model of Semiconductor Devices*, PhD thesis, Stanford University, 199

[23] A.D. Lilak, S.K. Earles, K.S. Jones, and M.E. Law, "A Physics-Based Modeling Approach for the Simulation of Anomalous Boron Diffusion and Clustering Behaviors,", IEDM Technical Digest, pp. 493-496, Dec. 1997.*

[24] T. Saito, J. Xia, R. Kim, T. Aoki, H. Kobayashi, Y. Kamakura, and K. Taniguchi, "Experiments and modeling of boron segregation to {311} defects and initial rapid enhanced boron diffusion induced by self-implantation in Si,", IEDM Technical Digest, pp. 497-500, Dec. 1998.*

[25] ALAMODE homepage, `http://www-tcad.stanford.edu/tcad/programs/alamode.html`.

[26] N.M. Wilson, S. Liang, P.M. Pinsky and R.W. Dutton, "A Novel Method to Utilize Existing TCAD Tools to Build Accurate Geometry Required for MEMS Simulation," MSM, San Juan, Puerto Rico, April 1999.*

[27] N.M. Wilson, Z.K. Hsiau, R.W. Dutton and P.M. Pinsky, "A Heterogeneous Environment for Computational Prototyping and Simulation Based Design of MEMS Devices," SISPAD '98, September 2-4, 1998.

[28] K. Hess, U. Ravaioli, N.R. Aluru, M. Gupta, and R.S. Eisenberg, "Simulation of Biological Ionic Channels by Technology Computer-Aided Design," to appear in Science.

[29] T.J. Liszka, W.W. Tworzydlo, J.M. Bass, S.K. Sharma, T.A. Westermann, and B.B. Yavari, "ProPHLEX - An hp-Adaptive Finite Element Kernel For Solving Coupled Systems of Partial Differential Equations in Computational Mechanics", Comput. Methods Appl. Mech. Engrg. 150 (1997) pp. 251-271.

[30] S. Selberherr, *Analysis and Simulation of Semiconductor Devices*, Springer-Verlag, New York, 1984.

[31] C.S. Rafferty and R.K. Smith, "Solving Partial Differential Equations with the `PROHET` simulator," Lucent Technologies Memo, Dec. 5, 1997, available as `http://www-tcad.stanford.edu/~prophet/math.fm.ps`.

[32] Zhiping Yu, et al., PISCES-2ET (manual), 1994, Stanford University.

*Hardcopy version provides these papers as Appendix B

†Thesis copies available on request

# A Graduate students supported under this contract

Kenneth Wang, PhD expected August 2000

Ze-Kai Hsiau, PhD, June 1997

Tao Chen, PhD expected June 2000

Amit Agarwal[1]

Michael Kwong[2]

Dan Yergeau, PhD, Oct. 1999

Roy Lue[3]

Gao-feng Wang[3]

Yi-chang Lu[4]

[1] now employed, Parametric Technology Corp. in San Jose, CA

[2] follow-on, application-pull research supported by Semiconductor Research Corporation (SRC)

[3] follow-on, collaborative support from NASA/Ames Research Center

[4] follow-on, application-pull support from IBM