Revista de Matemática: Teoría y Aplicaciones 2011 **18**(2) : 231–248
cimpa – ucr issn: 1409-2433

PARALLELIZATION OF A QUANTUM-CLASSIC HYBRID MODEL FOR NANOSCALE SEMICONDUCTOR DEVICES

PARALELIZACIÓN DEL MODELO HÍBRIDO CLÁSICO-CUÁNTICO PARA UN DISPOSITIVO SEMICONDUCTOR MOSFET NANOMÉTRICO

Oscar Salas* Piero Lanucara† Paola Pietra‡
Sergio Rovida§ Giovanni Sacchi¶

Received: 23 Feb 2010; Revised: 24 Nov 2010;
Accepted: 10 Dec 2010

---

*Department of Mathematics, Universidad Nacional, Heredia, Costa Rica. E-Mail: oscar.salas@unipv.it

†Consortium for the Applications of Super-Computing for Universities and Research (CASPUR), c/o Università La Sapienza, P. Also Moro 2, 00185 Roma, Italy. E-Mail: lanucara@caspur.it

‡Institute of Applied Mathematics and Information Technologies, C.N.R., via Ferrata 1, 27100 Pavia, Italy. E-Mail: pietra@imati.cnr.it

§Institute of Applied Mathematics and Information Technologies — CNR, Pavia, Italy. E-Mail: rovida@imati.cnr.it

¶Institute of Applied Mathematics and Information Technologies — CNR, Pavia, Italy. E-Mail: gianni.sacchi@imati.cnr.it

Abstract

The expensive reengineering of the sequential software and the difficult parallel programming are two of the many technical and economic obstacles to the wide use of HPC. We investigate the chance to improve in a rapid way the performance of a numerical serial code for the simulation of the transport of a charged carriers in a Double-Gate MOSFET. We introduce the Drift-Diffusion-Schrödinger-Poisson (DDSP) model and we study a rapid parallelization strategy of the numerical procedure on shared memory architectures.

Keywords: Parallelization; Shared memory paradigm; Schrödinger equation; Drift-Difusion system; Subband model; Nanotransistor.

Resumen

El transformar un software secuencial en uno paralelo, es costoso y difcil, lo cual constituye solo dos de los muchos obstculos tcnicos y econmicos que se tienen que enfrentar cuando se desea hacer uso de sistemas HPC. En este trabajo investigamos la posibilidad de mejorar de forma rpida y eficiente el desempeo de un cdigo numrico secuencial que se encarga de realizar la simulacin del comportamiento y transporte de un flujo de electrones en un dispositivo semiconductor MOSFET doble puerta y de escala nanomtrico. Se introduce el modelo Drift-Diffusion-Schrödinger-Poisson (DDSP) y se estudia una estrategia de paralelizacin rpida del procedimiento numrico, ptimo especficamente para arquitecturas a memoria compartida.

Palabras clave: Paralelezación; Paradigma de Memoria Compartida; Ecuación de Schrödinger; Sistemas Drift-Difusion; Modelo Subband; Nanotubos.

Mathematics Subject Classification: 65N55.

1 Introduction

The ongoing progress of industrial semiconductor device technologies during last fifty years has been focused on minimizing the size of electronic components. Nowadays nanometric devices can be used in a very large number of integrated circuits, resulting in remarkable performances improvement. In this task, modeling and numerical simulations play an important role in the determination of the limit size of a nanoscale device, as well as in the design of new devices.

We present the development of the parallel implementation of a code for numerical simulations of the electron transport in nanoscale semiconductor devices. We are specifically interested in a nanoscale Double-Gate MOSFET (Metal Oxide Semiconductor Field Effect Transistor), which is a very small structure Si/SiO_2.

When the sizes become so small ($< 100nm$), quantum phenomena, like interferences, confinement, and tunnelling effect occur. Therefore, it is necessary to develop new models able to take into account these phenomena. We introduce a quantum modelization in the direction perpendicular to the transport, and then we add a fluid model in the other directions. This mathematical model is named Drift-Diffusion-Schrödinger-Poisson(DDSP) [4], and in order to solve this system the Scharfetter-Gummel [8, 3] scheme is used.

Taking into account that the natural parallelism of the algorithm is achieved in the computation of the eigenvalues and eigenfuctions of the Schrödinger operator, we initially decide to implement a naive parallelism using OpenMP programming environment [2].

In Section 2 we present an overview of the model under consideration. Section 3 describes the numerical scheme and the iterative procedure used for the simulation. Section 4 is devoted to the parallelization strategy and finally the numerical experiments for a nanoscale double gate MOSFET are presented.

2 The DDSP model

When the electrons are extremely confined, as it happens in the semiconductor nanometric devices, the energy is divided in partially quantified energy levels. These levels are described by the eigenvalues of the partial Hamiltonian in the confinement direction, and are usually named subbands in physics literature [4]. We name z and x the confinement and the transport directions respectively. The particles are affected by an electrostatic potential $V(t, x, z)$ which depends on the time variable t and spatial variables (x, z).

In z direction the confinement forces a quantization of the system and for this reason the Hamiltonian partial differential equations $-\frac{1}{2}\partial_z^2 + V$ must have a discrete spectrum.

We assume now that the confinement area is bound to the following values $z = 0$ and $z = \ell$. Therefore the z variable belongs the domain $(0, \ell)$. In the system x is considered as a parameter. Furthermore we consider the system to be static (i.e. not depending on t).

In the sub-band decomposition approach the system is viewed as a statistical mixture of eigenstates of the Schrödinger operator in the confined direction.

The occupation number of each state is given by a statistic function: for Boltzmann statistics it is $\exp\left(\frac{\epsilon_F - \epsilon}{k_B T_L}\right)$, for Fermi-Dirac statistics it is, $1/\left[1+\exp\left(\frac{\epsilon - \epsilon_F}{k_B T_L}\right)\right]$.

In these expressions ϵ is the energy of the considered state, k_B is the Boltzmann constant, T_L is the lattice temperature and ϵ_F is the so-called Fermi energy which, at zero temperature, represents the threshold between occupied and unoccupied states [9, 13].

In the confined direction, the system is assumed to be at equilibrium with a local Fermi level ϵ_F which depends on the transport variable x. At a position (x, z), the particle density $N(x, z)$ for Boltzmann statistics is given by

$$N(x,z) = \sum_{k=1}^{+\infty} e^{\beta(\epsilon_F(x) - \epsilon_k(x))} |\chi_k(x,z)|^2, \tag{1}$$

where $\beta = 1/(k_B T_L)$ and $(\chi_k, \epsilon_k)_{k\geq 1}$ is the complete set of eigenfunctions and eigenvalues of the Schrödinger operator in the z variable:

$$\begin{cases} -\dfrac{\hbar^2}{2}\dfrac{d}{dz}\left(\dfrac{1}{m^*(z)}\dfrac{d}{dz}\chi_k\right) + U\chi_k = \epsilon_k \chi_k, \\ \chi_k(x,\cdot) \in H_0^1(0,\ell), \qquad \displaystyle\int_0^\ell \chi_k\, \chi_{k'}\, dz = \delta_{kk'} \end{cases} \tag{2}$$

In equation (2) $\hbar$ is the reduced Planck constant and m^* the effective mass. Moreover, U is the potential energy defined by $U = -eV$, where e is the elementary charge and V denotes the self-consistent electrostatic potential, solution of the Poisson equation

$$\mathrm{div}_{\,x,z}(\varepsilon_R(x,z)\nabla_{x,z} V) = \frac{e}{\varepsilon_0}(N - N_D). \tag{3}$$

$\varepsilon_R(x, z)$ denotes the relative permittivity, ε_0 the permittivity constant in vacuum and $N_D(x, z)$ is the prescribed doping density.

In the transport direction(s), we consider a purely classical transport in the diffusive regime. We can consider some description levels, according to the physical contest: kinetic or fluid. The macroscopic magnitudes determines the transport and can be calculated considering the Vlasov equation (if there are no collisions) or Boltzmann equation from kinetic point of view, and using a Drift-diffusion system from fluid model point of view. We now have a coupled system consisting in one of the prior equations, the sub-bands model (2), and the Poisson equation (3) .

We are interested in the transport in a semiconductor device where the collisions are the main responsible for the motion. The transport equation we are going to consider is the following stationary Drift-Diffusion equation:

$$\begin{aligned} -\mathrm{div}_x\, J(x) &= 0, && (4)\\ J(x) &= \mathbb{D}(\nabla_x N_s(x) + \beta N_s(x)\nabla_x U_s(x)), && (5) \end{aligned}$$

where N_s is the surface density, $\mathbb{D}$ denotes the diffusion coefficient $\mathbb{D} = \mu k_B T_L$ for a constant mobility μ and the effective energy U_s is given by

$$U_s = -k_B T_L \log\left(\sum_{k=1}^{+\infty} e^{-\beta\epsilon_k}\right). \qquad (6)$$

Vlasov-Schrödinger-Poisson system is studied in [1] and Boltzmann-Schrödinger-Poisson system is studied in [12].

If we define the repartition function $\mathcal{Z}$ as

$$\mathcal{Z}(x) = \sum_{k=1}^{+\infty} e^{-\beta\epsilon_k(x)}, \qquad (7)$$

then, we easily obtain from (1) and (2) that the surface density satisfies

$$N_s(x) = \int_0^\ell N(x,z)\,dz = e^{\beta\epsilon_F}\mathcal{Z}(x).$$

Therefore, $\epsilon_F(x)$ in (1) can be written in terms of N_s and N_s can be chosen as unknown in the model, then we have

$$N(x,z) = \frac{N_s(x)}{\mathcal{Z}(x)} \sum_{k=1}^{+\infty} e^{-\beta\,\epsilon_k(x)} |\chi_k(x,z)|^2. \qquad (8)$$

If we introduce the Slotboom variable u defined by

$$u(x) = e^{\beta\epsilon_F} = \frac{N_s(x)}{\mathcal{Z}(x)}, \qquad (9)$$

then, we get easily that the drift-diffusion equation (4)-(5) reads

$$-\text{div}_x(\mathbb{D}\mathcal{Z}(x)\nabla_x u(x)) = 0. \tag{10}$$

The drift-diffusion equation can be derived from kinetic theory when the mean free path is small compared to the system length scale [11, 6].

The unknowns of the overall system are the surface density $N_s(x)$, the eigenenergies $\epsilon_k(x)$, the eigenfunctions $\chi_k(x,z)$ and the electrostatic potential $V(x,z)$. If we assume that the electrostatic potential V is given, then a diagonalization of the one dimensional Schrödinger operator (2) provides the eigenvalues and eigenvectors (ϵ_k, χ_k). The effective energy U_s can be computed from (6). This allows us to obtain the surface density N_s by solving the drift-diffusion problem (4) - (5). Thus, we have all the knowledge to find the density N in (8) and therefore to compute a new potential thanks to the resolution of Poisson equation (3).

The DDSP model studies the coupled system consisting in the equations (4), (2) and (3) using the expression (6) and (8). The transport variable x belongs to the domain $(0,L)$, then the studied domain is $\Omega = [\,0,L\,]\times[\,0,\ell\,]$. In order to complete this system suitable boundary conditions must be imposed (see Section 3).

3 Numerical implementation

In this work we are interested in a nanoscale Double-Gate MOSFET, which is a very small structure Si/SiO_2. The device,shown in Fig.1, consists of two highly doped regions (N^+) near the Ohmic contacts (denoted by *source* and *drain*) and an active region, called channel, with lower doping (N).

This MOSFET has two gates, insulated from the channel by layers of silicon dioxide SiO_2.

We assume invariance in the y direction (infinite boundary conditions), so that the problem is studied in a (x,z)-domain.

The device occupies a region of a 2-D domain denoted by $\Omega = [\,0,L\,]\times[\,0,\ell\,]$.

We recall the stationary DDSP system used for the simulation, taking into account the presence of the oxide of silicon. As in Section 2, we will use the notation $\beta = 1/k_BT_L$.

Find $N_s(x)$, $(\epsilon_k(x), \chi_k(x,z))$, for $k \geq 1$, and $V(x,z)$ such that

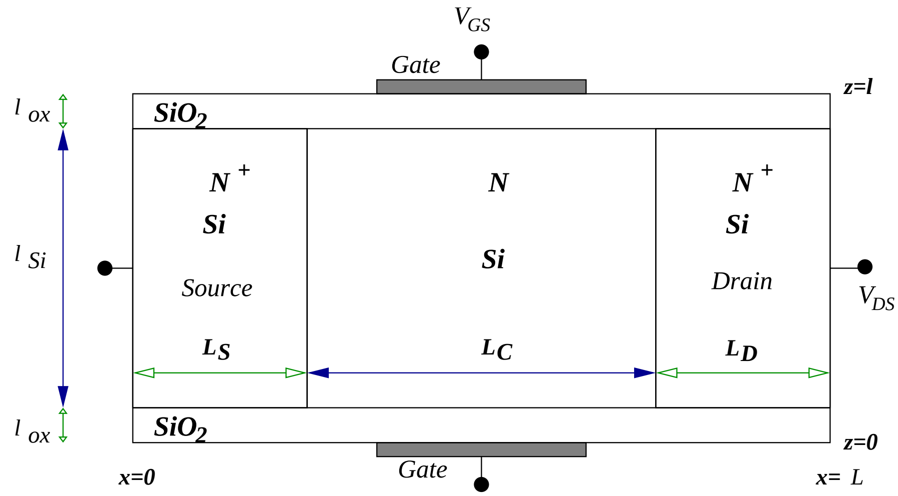

Figure 1: Schematic representation of the modeled device.

$$\frac{d}{dx}\left(\mathbb{D}\left(\frac{d}{dx}N_s + N_s\frac{d}{dx}U_s\right)\right) = 0 \qquad \text{in } (0, L), \tag{11}$$

$$\begin{cases} -\dfrac{\hbar^2}{2}\dfrac{d}{dz}\left(\dfrac{1}{m^*(z)}\dfrac{d}{dz}\chi_k\right) + (U + U_c)\chi_k = \epsilon_k\chi_k. \\ \chi_k(x,\cdot) \in H_0^1(0,\ell), \qquad \displaystyle\int_0^\ell \chi_k\,\chi_{k'}\,dz = \delta_{kk'}\,, \end{cases} \tag{12}$$

$$\text{div}_{\,x,z}(\varepsilon_R(x,z)\nabla_{x,z}V) = \frac{e}{\varepsilon_0}(N - N_D) \qquad \text{in } \Omega, \tag{13}$$

where

$$U = -eV\ ; \qquad N(x,z) = \frac{N_s(x)}{\mathcal{Z}(x)}\sum_{k=1}^{+\infty} e^{-\beta\epsilon_k(x)}|\chi_k(x,z)|^2, \tag{14}$$

$$U_s = -k_B T_L \log \mathcal{Z}\ ; \qquad \mathcal{Z}(x) = \sum_{k=1}^{+\infty} e^{-\beta\epsilon_k(x)}. \tag{15}$$

In (12) U_c is a given potential barrier between the silicon and the oxide. Moreover, the diffusion coefficient $\mathbb{D}$ in (11), ϵ_R, ϵ_0, e, N_D in (13), and the electron effective mass m^* in (14) are defined as in Section 2.

At the ohmic source, drain, and gate contacts (denoted by Γ_S, Γ_D, and Γ_G, respectively) Dirichlet boundary conditions are imposed. The remaining part of the boundary (denoted by Γ_N) is considered insulated, and homogeneous Neumann boundary conditions are imposed. Due to the high doping, the drain and the source contacts can be considered as small electron reservoirs in which we assume that the potential does not depend on the transport direction. Therefore, the surface density N_s at the vicinity of the drain and source contacts is assumed to be constant and is then equal to $N^+ \times \ell_{Si}$.

Moreover, the electrostatic potential V equals the sum of the applied voltage and the potential at thermal equilibrium, denoted by $V_b(x_c, z)$, with $x_c = 0$ or $x_c = L$.

In order to find $V_b(x_c, z)$ the following 1-D Schrödinger-Poisson system must be solved on the vertical edge $x = x_c$

$$\begin{cases} -\dfrac{\hbar^2}{2}\dfrac{d}{dz}\left(\dfrac{1}{m^*(z)}\dfrac{d}{dz}\chi_k\right) + (U_b + U_c)\chi_k = \epsilon_k \chi_k. \\ \chi_k(x_c, \cdot) \in H_0^1(0, \ell), \qquad \displaystyle\int_0^\ell \chi_k\, \chi_{k'}\, dz = \delta_{kk'}\,, \end{cases} \tag{16}$$

$$\frac{d}{dz}\left(\varepsilon_R(x_c, z)\frac{d}{dz}V_b\right) = \frac{e}{\varepsilon_0}(N - N_D), \qquad V_b = 0 \quad \text{at } z = 0 \text{ and } z = \ell, \tag{17}$$

where U_b, N, $\mathcal{Z}$ are the same as in (14) and (15) computed for $x = x_c$.

Summarizing, the boundary conditions for system (11) - (13) are

$$N_s = N^+ \times \ell_{S_i} \quad \text{at } x = 0 \text{ and } x = L, \tag{18}$$

$$V(z) = V_b(0, z) \quad \text{on } \Gamma_S, \qquad V(z) = V_b(L, z) + V_{DS} \quad \text{on } \Gamma_D, \tag{19}$$

$$V(x) = V_{GS} \quad \text{on } \Gamma_G, \tag{20}$$

$$\frac{\partial V}{\partial n} = 0 \quad \text{on } \Gamma_N. \tag{21}$$

with V_{DS} Drain-Source voltage and V_{GS} Gate-Source voltage.

We introduce a partition of $[0, L]$ with nodes x_i, $i = 1, \cdots, N_x$, and a partition of $[0, \ell]$ with nodes z_j, $j = 1, \cdots, N_z$. Then, we mesh the domain $[0, L] \times [0, \ell]$ with rectangular triangles using the nodes (x_i, z_j) previously defined. The 1-D Schrödinger equations and the 2-D Poisson equation are discretized with conforming $\mathbb{P}^1$ finite elements, while for the 1D drift-diffusion equation the Scharfetter-Gummel scheme is used (see [8, 3]). From now on, when referring to equations and formulas (11) - (21) we intend their discretized counterpart.

In the solution of DDSP system we can distinguish three different computational phases.

Initialization: The first step for initializing the procedure is the computation of V_b on the source and drain contacts. To this aim a Gummel iteration method, described at the end of this section, is used to solve the 1-D Schrödinger-Poisson system.

Equilibrium state: Secondly, we consider the whole system for zero applied source-drain voltage. Equation (11) does not need to be solved in this case, since we do not have electron transport. Actually, the Slotboom variable $u = N_s/\mathcal{Z}$, solution of the 1-D equation (10), is constant. It is then sufficient to evaluate it on the boundary (for $x = 0$ for instance), where N_s is prescribed. It remains to calculate a 2D Schrödinger-Poisson system, with boundary conditions (19)–(21) and $V_{DS} = 0$.

Out of equilibrium state: Finally, we consider the resolution of the DDSP system when a drain-source voltage V_{DS} is applied. We start from the previously obtained potential and we increment the voltage by steps of $0.02V$.

The iterative procedure for the solution of one DDSP system consists of an iteration on the electrostatic potential and it is summarized in the following steps.

Step 1 - For a given potential V_{old} in the whole domain Ω we solve the eigenvalue problem (12) by diagonalization of the Hamiltonian on each slice of the device ($x = x_i$). Thus, we obtain N_x sets of eigenfunctions $\{\chi_k(x_i, z)\}_{i=1,\cdots,N_x}$ and eigenvalues $\{\epsilon_k\}$.

Step 2 - Next, we compute the effective energy U_s from (15). We are then able to solve the 1D stationary drift-diffusion equation (11) with Dirichlet boundary conditions (18).

Step 3 - We have then all data needed to compute the density N using to the expression (14).

Step 4 - The Poisson equation (3) is solved in the 2-D domain using boundary conditions (19)–(21). The system is solved using the preconditioned conjugate gradient method. A new potential V_{new} is then obtained.

Step 5 - We repeat the four previous steps until $\|V_{new} - V_{old}\|_{L^\infty}$ is sufficiently small.

We conclude with few remarks on implementation aspects.

A) The solution of the highly non-linear coupled Schrödinger-Poisson system is the most delicate step in the procedure described above. If the decoupling procedure is not appropriate, the algorithm fails. On the other hand the use of Newton-Raphson method is very expensive. We use a Gummel iterative scheme [8] and we substitute in the procedure the Poisson equation (3) with

$$-\nabla(\varepsilon_R \nabla V_{new}) + \frac{e}{\varepsilon_0} N(x,z) \frac{V_{new}}{V_{ref}} = \frac{e}{\varepsilon_0}\left(N_D - N(x,z)\left(1 - \frac{V_{old}}{V_{ref}}\right)\right),$$

with $V_{ref} = k_B T_L / e$ (see [10]). This method can be viewed as an approximate Newton method where the Jacobian of the system is replaced by a diagonal matrix.

B) When solving the eigenvalue problems it is not necessary to compute all the N_z modes because of the exponential dependence of U_s on the energy levels ϵ_k's (see (15)). Here we used only the first 12 modes.

C) The solution of one Schrödinger problem on a slice is independent of the others, therefore, the most costly part of the algorithm is fully and easily parallelizable.

4 Parallelization

We describe the parallel implementation of the DDSP procedure, which presents, as already mentioned, a high degree of parallelism when computing the eigenvalues and eigenfunctions of the Schrödinger operator.

We start from an existing serial code developed in Fortran90, using the Sparse Linear Algebra Package (SLAP) version 2.0 (see [7]), and the Linear Algebra Package (LAPACK).

We need to introduce some considerations.

The numerical approximation of the problem is carried out using $\mathbb{P}^1$ finite elements; the mesh sizes are relatively small (ranging from 50×50 degrees of freedom (dof) to 150×150 dof); the number of eigenstates of the Schrödinger operator we calculate is small (12 eigenstates are computed).

Moreover the diagonalization of the Hamiltonian is a real generalized symmetric-definite eigenproblem (in the form $A * x = \lambda * B * x$), and to solve it we use the LAPACK routine `DSYGV` in order to compute all the eigenvalues, and eigenvectors of the operator.

The linear system resulting from the approximation of the 1-D Poisson problem is solved by LAPACK routine `DGTSV` using Gaussian elimination with partial pivoting.

The solution of the final linear system related to the 2-D Poisson problem is performed by the SLAP subroutine `DSICCG`, which implements the Incomplete Cholesky Preconditioned Conjugate Gradient method, with the l_∞ - norm of the residual as stopping criteria and 10^{-8} as fixed tolerance.

In the end the Drift-Diffusion problem is solved as before, using LAPACK subroutine `DGTSV`.

Due to the relatively small size of the numerical problem and taking into account that the natural parallelism of the algorithm is achieved in the computation of the eigenvalues and eigenfuctions of the Schrödinger operator, we initially decide to implement a naive parallelism using the programming environment provided by OpenMP version 2.5 (see [2]).

4.1 OpenMP

The OpenMP Application Program Interface (OpenMP API) provides a model for parallel programming that is portable across shared memory architectures offered by different vendors. The directives, library routines, and environment variables defined in the OpenMP standard allow users to create and manage parallel programs, assuring portability at the same time. The directives extend the C, C^{++} and Fortran base languages with Single Process, Multiple Data (SPMD) constructs, work-sharing constructs, and synchronization constructs, while providing support for the sharing and privatization of data. Compilers that support the OpenMP API often include a command line option to the compiler that activates and allows interpretation of all OpenMP directives.

In order to parallelize our code we use the `parallel do` construct which provides a shortcut form for specifying a parallel region that contains a single `do loop`.

As we specified in Sec. 3, in order to decouple the stationary DDSP system the first step consists of computing the eigenvalues of Schrödinger operator (12) on each vertical slice at the point $x = x_i$ of the considered discretization.

The parallelization can be achieved in a rapid natural way adding,as in Fig 4, the `parallel do` directive to the original Fortan code.

The routine `sch1d` computes at each step k the eigenvalues En ($\{\epsilon_k\}$ and the sets of the eigenfunctions xin ($\{\chi_k(x_i, z)\}_{i=1,\cdots,N_x}$ introduced in Section 3.

The construct `parallel do` creates a team of threads which will solve the problem (12) for each k index (i.e. on each vertical slice). The clause

```
!$omp parallel do schedule (dynamic)
!$omp private(g,ge)
!$omp default(shared)
    do  k  = 1, nbpointx-1
          . . .
    allocate(g(1:nbk1), ge(1:nbk1))
      Update of function g = e V , ge = U c
          . . .
    call sch1d ( zp1,ily,g,ge,xin(k+1,:,:),En(k+1,:),zmtbe )
          . . .

      Normalisation of the set of the eigenfunctions
    deallocate(g,ge)
    end do
!$omp end parallel do
```

Figure 2: The Code Flow Chart.

`schedule`, set to `dynamic` orders to the threads, which has completed its job to start with the next free index k.

4.2 Code structure

Figure 3 illustrates the hierarchical organization of the code and the three phases (*Step 1–3*) described in Section 3.

4.3 Code description

The MOSFET simulation code can be divided in the three blocks in Fig. 3. Each one of these steps requires the solution of a quantum-classic hybrid mono-dimensional or bi-dimensional model.

The execution of the code is controlled by a main program which calls the subroutines corresponding to the computational kernels of the procedure. This way we have these main routines:

- `subroutine sch1d` to calculate the Schrödinger problem
- `subroutine pot1d_vdm` to calculate the Poisson 1-D problem
- `subroutine pot2d_vd` to calculate the Poisson 2-D problem
- `subroutine driftdiv_1d` to calculate the Drift-Diffusion 1-D problem

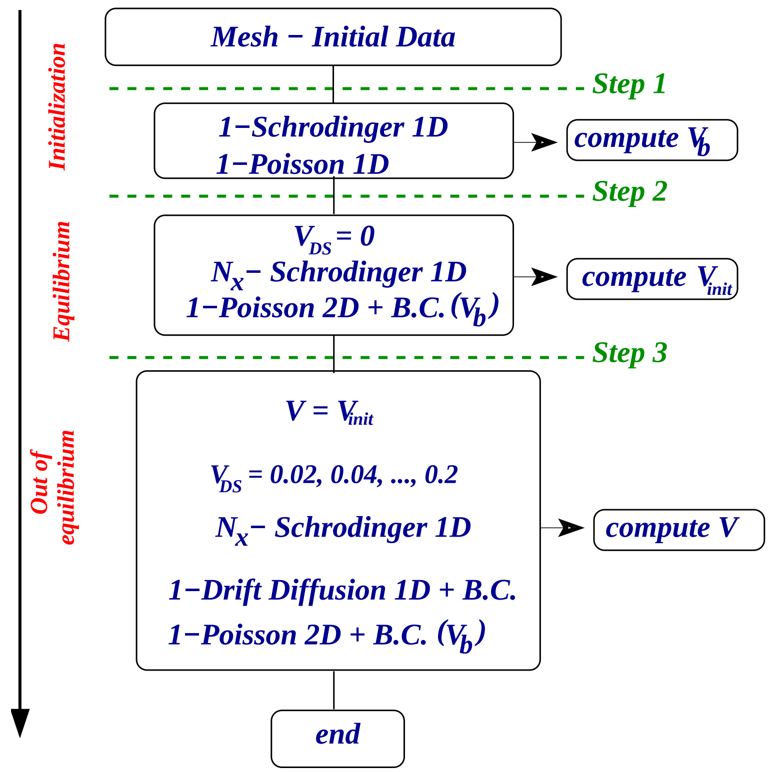

Figure 3: The Code Flow Chart.

Each one of these subroutines respectively implement the approximation method described in the introduction of the present section.

In the main program the iterative Gummel procedure is implemented at each step in order to find the corresponding solution, as described in the Section 3.

In the initialization phase we need to solve a mono-dimensional problem consisting of one-Schrödinger equation (16) and one-Poisson 1-D equation (17), in order to find the value $V_b(x_c, z)$ in source and drain areas. This value will be used in order to impose the boundary conditions in the following steps. Notice that it is sufficient to evaluate the problem for $x_c = 0$, and obviously we cannot exploit any kind of parallelism in this phase.

In the second step, or equilibrium phase, the potential difference applied between source and drain is null ($V_{DS} = 0$), therefore we have to solve a system consisting of N_x-Schrödinger equations (12) and one-Poisson 2-D equation (13), with boundary conditions (19, 20,21)- In this step we can exploit, at each iteration, the intrinsic parallelism in the calculation of the N_x-Schrödinger problems. These problems are indipendent and can be calculated at the same time, according to the available number of threads.

In the third step, or out of equilibrium phase, the potential difference applied between source and drain is not null, thus we add to the system considered above the one-Drift-Diffusion 1-D equation (11), with boundary conditions (18). Therefore we exploit the advantages of decoupling the system with the Gummel iterative procedure and the intrinsic parallelism in the calculation of the N_x-Schrödinger problems.

5 Numerical experiments

We consider the model problem described in the Section 3 and we consider the geometry in Fig.1 with the following values in nanometers:

$$\Omega = [\,0, 50\,] \times [\,0, 12\,],\ \ell_{ox} = 2,\ \ell_{Si} = 8,\ L_S = 15,\ L_C = 30,\ L_D = 15.$$

The values of the main physical parameter, used in the computation are shown in Table 1.

$T_L = 300\ K$	$N^+ = 10^{20}\ cm^{-3}$
$e = 1.6^{-19}\ A\ s$	$N = 10^{15}\ cm^{-3}$
$\varepsilon_0 = 8.85^{-12}\ A\ s\ cm^{-1}$	$U_c = 3\ eV$
$\varepsilon_R[Si] = 11.7$	$m_e = 9.10^{-31}\ kg$
$\varepsilon_R[SiO_2] = 3.9$	$m^* = 0.5\ m_e$

Table 1: Physical parameters.

Moreover, we consider $N_x = N_z = 50$ and $N_x = N_z = 150$ (see Sec. 3).

Numerical tests have been carried out on the AMD Opteron cluster at Institute of Applied Mathematics and Information Technologies, CNR in Pavia and on the IBM Power 5 cluster at Inter-University Consortium for the Application of Super-Computing for Universities and Research (CASPUR)in Roma.

The experiments on the AMD Opteron cluster have been performed on a single 4 CPUs node, running one thread per CPU, using the PGI-Compiler 6.2 and the OpenMP API 2.5, ACML 3.6.0 library and SLAP 2.0 package.

The IBM cluster belongs to Series Power 5, and consists of 21 nodes which share 44GB RAM. Each node is provided with 8 processors SP5, each processor with 1.9 GHz clock speed.

The nodes of the cluster are connected to each other with an high speed network, based on IBM High Performance Switch (HPS).

The experiments on IBM Power 5 cluster have been performed on a single 8 CPUs node, running one thread per CPU, using the IBM XL compiler and the OpenMP API 2.5, ESSL 3.2 library and SLAP 2.0 package.

We present the following numerical tests:

Case A - Partial problem (initialization phase + equilibrium phase) **:** In this case we consider a mesh with 22500 dof and we discuss the scalability behaviour obtained running the code on both the IBM Power 5 cluster and the AMD Opteron cluster.

We initially considered the Partial problem in the way to find out the extent of performance improvement when executing the parallel version of our algorithm.

Tables 2-3 show the results obtained increasing the number of threads.

threads	*expected speedup*	*speedup*
1	1.00	1.01
2	1.42	1.41
3	1.65	1.62
4	1.80	1.74
5	1.91	1.84
6	1.98	1.93
7	2.04	1.94
8	2.08	1.97

Table 2: Case A: Speedup and Expected speedup on IBM SP5.

In Tables 2-3 we report in the second column entitled *expected speedup*, the maximum expected speedup evaluated according to Amdahls law, taking into account that the cost of the parallel portion of the run, on ADM and IBM cluster respectively, are equal to 47% or 59% of the total execution time.

threads	*expected speedup*	*speedup*
1	1	1.00
2	1.29	1.25
3	1.44	1.41
4	1.56	1.51

Table 3: Case A: Speedup and Expected speedup on AMD Opteron cluster.

The third column entitled *speedup*, we report the speedup value calculated as the ratio between the sequential wall clock time and the parallel wall clock time.

The *speedup* values are very close to the ones expected The parallel performance obtained must be regarded as optimal and are satisfactory, taking into account the suimplicity of the approach.

In Table 2 we can see that using 8 processors the execution time can be reduced nearly one half.

Case B - Full problem (see Fig, 2) **:** We solve the complete problem with a mesh of bot 2500 and 22500 dof and we discuss the results obtained in terms of the total wall clock execution time on the AMD Opteron cluster.

Table 4 shows the results obtained varying the number of threads. We remark that each node of the AMD Opteron cluster has 4 CPU and that we execute up to 4 threads concurrently.

	Case B	**dof=2500**	**Case B**	**dof=22500**
threads	*total time*	*Schr/iter*	*total time*	*Schr/iter*
1	52.7	0.09	349.3	0.75
2	44.1	0.05	271.6	0.39
3	41.1	0.02	245.8	0.26
4	39.7	0.02	230.8	0.20

Table 4: Case B: Wall clock time in *secs* using 2500 d.o.f.

In Table 4 the first column, entitled *threads*, contains the number of threads used. The columns entitled *total time*, we report the total wall clock time spent by the algorithm. The columns entitled *Schr/iter* reports the wall clock time for the solution of the N_x Schrödinger equations per a single Gummel iteration.

An analysis of the values reported in column *Schr/iter* shows that the time decreases, as expected, proportionally to the number of threads used. This confirms, also for the full problem, the good scalability of our implementation. Referring to Table 4 we notice that the times per iteration remain unchanged when three or four threads are used. This is probably due to the overhead in the management of the threads which is not offset by the size of the computational problem. The results on the scalability described above are confirmed by the reduction of the *total times* in Table 4.

Comparing theses results and computing, as for case A, the expected speedup and the speedup obtained, one can get values which are very close to the ones reported in Table 3. This observation confirms that the chosen parallelization strategy is a good choice also for the treatment of the Full Problem.

6 Conclusions

In this work we focused on a parallelization strategy for the code used in the numerical simulation of a nanoscale double-gate MOSFET.

We were interested in an initial approach leading us to optimally exploit the structure of the original existing code. This way the use of OpenMP library is an ideal approach, since the code can be executed, once the directives have been implemented in it, on both shared memory multiple core PCs and single core PCs, without reingeneering any part of it.

Considering the results obtained with the parallel version of the code, we regard this as a good starting point to upgrade this implementation to possible future models considering both quantum effects along the transport direction or 3-D extension of the present model.

Other parallelization approaches, exploiting inter-node parallelism, are still to be evaluated. A complete algorithm reingeneering and deep modification of the complex data structures of the sequential code would be required. This operation must be carefully evaluated both in terms of the resources used and of the performance improvements one can obtained.

References

[1] Ben Abdallah, N.; Mhats, F. (2004) "On a Vlasov–Schrödinger–Poisson model", *Comm.Partial Differential Equations* **29**(1-2): 173–206.

[2] OpenMP Architecture Review Board. *OpenMP Application Program Interface. Version 2.5*, May 2005.

[3] Brezzi, F.; Marini, L.D.; Pietra, P. (1987) "Méthodes d'éléments finis mixtes et schéma de Scharfetter–Gummel", *C.R. Acad. Sci. Paris Sér. I* **305**: 599–604.

[4] Davies, J.H. (1998) *The Physics of Low Dimensional Semiconductors.* Cambridge University Press.

[5] Degond, P.; Levermore, C.D.; Schmeiser, C. (2004) "A note on the Energy-Transport limit of the semiconductor Boltzmann equation, Transport in transition regimes", *IMA Vol. Math. Appl.*, Springer, **135**: 137–153.

[6] Golse, F.; Poupaud, F. (1992) "Limite fluide des équations de Boltzmann des semiconducteurs pour une statistique de Fermi–Dirac", *Asymptotic Analysis* **6**: 135–169.

[7] Greenbaum, A. (1986) "Routines for Solving Large Sparse Linear Systems (SLAP)". Lawrence Livermore Nat. Laboratory, Livermore Computing Center, USA.

[8] Gummel, H.K. (1964) "A self-consistent iterative scheme for one-dimensional steady state transistor calculations", *IEEE Trans. on Elec Dev.* **11**(10): 455–465.

[9] Nier, F. (1990) "A stationary Schrödinger–Poisson system arising from the modelling of electronic devices", *Forum Math.* **2**(5):489–510.

[10] Pietra, P.; Vauchelet N. (2007) "Modeling and simulations of the diffusive transport in a nanoscale double-gate MOSFET", Technical Report 10PV07/10/0, IMATI-CNR, Pavia.

[11] Poupaud, F. (1981) "Diffusion approximation of the linear semiconductor Boltzmann equation: analysis of boundary layers", *Asymptotic Analysis* **4**: 293–317.

[12] Vauchelet, N. (2006) *Modélisation Mathématique du Transport Diffusif de Charges Partiellement Quantiques.* Ph.D. thesis, Université Paul Sabatier, Toulouse.

[13] Vinter, B.; Weisbuch, C. (1991) *Quantum Semiconductor Structures.* Academic Press.