Solid-State Electronics 199 (2023) 108487

Contents lists available at ScienceDirect

Solid State Electronics

journal homepage: www.elsevier.com/locate/sse

Development of an ensemble Monte Carlo simulator for high-power semiconductor devices with self-consistent electromagnetism and GPU implementation ☆

S.J. Cooke [a,*], M.G. Ancona [b]

[a] Electronics Science and Technology Division, Naval Research Laboratory, Washington, DC 20375, USA
[b] Department of Electrical and Computer Engineering, Florida State University, Tallahassee, FL 32310, USA

ARTICLE INFO

Keywords:
Electron transport
Electromagnetism
Monte Carlo simulation
Particle-in-cell simulation

ABSTRACT

The ensemble Monte Carlo (EMC) method is a powerful and widely used technique for modeling carrier transport in semiconductors in which one can easily take into account band structure and a variety of scattering mechanisms. In the usual implementations of EMC, the semiclassical electron dynamics is coupled self-consistently to the electrostatic field and this suffices for most devices. However, for high power, high current semiconductor device applications the electrostatic approximation is often not adequate and the full electromagnetic (EM) equations must be solved in fully-coupled fashion. We report on the development of a fully self-consistent, time-domain, EM-EMC simulator for two- and three-dimensional device simulations in which we also address a well known drawback of these methods, their computational intensiveness, by producing a GPU implementation.

1. Introduction

High-power, high-current semiconductor devices, such as fast opening switches or high-power amplifiers, often exhibit effects such as strong magnetic fields, filamentation and instability that are more often associated with vacuum plasmas [1] and high current-density charged-particle beams. In those applications, a robust numerical method that is widely used to model the complex interaction dynamics between electromagnetic fields and classical charged particle distributions is the electromagnetic particle-in-cell (EM-PIC) method. This time-domain technique solves simultaneously Maxwell's equations for the electric and magnetic fields (represented across the cells of a spatial computational grid) and the classical equations of motion for a particle distribution in phase space (represented by the positions and momenta of a particle ensemble). These two systems of equations are coupled self-consistently via the Lorentz forces acting on the charged particle ensemble and the resulting electromagnetic current density distribution driving the EM field solution. In this paper we use the EM-PIC method as the basis to create a fully self-consistent electromagnetic ensemble Monte Carlo (EM-EMC) model for semiclassical electron dynamics in semiconductors for high-power device simulation. Because the EMC method incorporates the microscopic physics, the EM-EMC method can be particularly useful for predicting the potential of new semiconductor materials for high-power device applications.

A convenient starting point for our development of a particle-based simulator of solid-state plasma processes is an existing 3D EM-PIC simulation framework called Neptune [2], which was created for modeling high-power, relativistic, vacuum electronic devices. This code already includes a GPU implementation for computational efficiency and provides a flexible scripting language interface to customize device geometry and operating parameters during simulations.

Solving Maxwell's equations in place of Poisson's equation introduces a number of challenges. While Poisson's equation is defined in terms of electric potential and charge density distributions, Maxwell's time-dependent equations fundamentally use electromagnetic fields and currents. Neptune's EM-PIC algorithm preserves the charge continuity equation exactly so that Poisson's equation remains satisfied implicitly for the duration of the simulation. The character of the field equation solution has also changed from elliptic to hyperbolic. The time integration of electromagnetic fields in Neptune uses an explicit leapfrog scheme related to the Finite Difference Time Domain (FDTD) method, for which the maximum time step is constrained by the Courant–Friedrich–Lewy (CFL) condition for stability. For typical dimensions and particle velocities found in semiconductor simulations

☆ The review of this paper was arranged by Francisco Gamiz.

* Corresponding author.
E-mail address: simon.cooke@nrl.navy.mil (S.J. Cooke).

https://doi.org/10.1016/j.sse.2022.108487

Available online 31 October 2022
0038-1101/Published by Elsevier Ltd.

this leads to time steps much shorter than when using a Poisson solver, but at a lower computational cost per time step due to locality in the discrete equations. In the present implementation, the particle time step is taken to be the same as the electromagnetic time step and, because of this, the implementation of the scattering process in our model differs from that in conventional EMC implementations [3,4]. Rather than computing a time-of-flight until the next scattering event takes place, the probabilities of each scattering mechanism are evaluated independently each time step.

2. EM-EMC algorithm

The EM-EMC algorithm follows initially a similar scheme to the EM-PIC method that is implemented in the Neptune simulation code. An outline of the formulation and implementation are provided in this section.

2.1. EM-PIC time-domain model

Maxwell's time-dependent equations, in macroscopic form, describe the evolution of electromagnetic fields in a volume containing regions of dielectric and/or permeable materials:

$$\frac{d}{dt}\vec{B} = -\nabla\times\vec{E} \qquad \vec{E} = \varepsilon^{-1}\vec{D}$$
$$\frac{d}{dt}\vec{D} = \nabla\times\vec{H} - \vec{J} \qquad \vec{H} = \mu^{-1}\vec{B}$$

The numerical solution first discretizes the fields on a Cartesian spatial grid according to the Yee scheme: components of electric fields ($\vec{E}\&\vec{D}$) are associated with grid cell edges and of the magnetic fields ($\vec{B}\&\vec{H}$) with cell faces. This ensures that all of the discrete spatial finite-difference terms are correctly centered. Discretization in time follows an explicit *leapfrog* time-integration scheme with a fixed time step, Δt: the electric and magnetic fields are defined at alternate half time-steps and are updated sequentially to complete a full time integration step. This scheme is stable when $c\Delta t$ is sufficiently small to satisfy the CFL condition, related to the spatial grid cell size. The implementation in Neptune parallelizes each field update step with respect to the cells of the 3D grid.

In both the EM-EMC and EM-PIC methods, a set of simulation particles is introduced to represent the distribution of charges in phase space. Each simulation particle carries a relative weight, which scales its charge, mass, energy and momentum relative to the physical particle that it represents. For each particle in an ensemble, its motion in an electromagnetic field is governed by the following general scheme:

$$\frac{d}{dt}\vec{x} = \vec{v} \qquad \vec{v} = \nabla_{\vec{p}}\,\mathcal{E}(\vec{p})$$
$$\frac{d}{dt}\vec{p} = \vec{F} \qquad \vec{F} = q(\vec{E}(\vec{x}) + \vec{v}\times\vec{B}(\vec{x}))$$

The dependence of velocity on the particle momentum is defined in terms of an energy–momentum relation, $\mathcal{E}(\vec{p})$. For relativistic particles moving in vacuum this relation is:

$$\mathcal{E}^2 = c^2\left(\vec{p}\cdot\vec{p} + m^2c^2\right) \quad \Rightarrow \quad \vec{v} = \frac{1}{\sqrt{1+\frac{\vec{p}\cdot\vec{p}}{m^2c^2}}}\frac{\vec{p}}{m}$$

The numerical formulation uses the same leapfrog time-integration scheme as for the fields, with the same time step, and with position and momentum defined on alternate half time steps. The formulation is implicit due to the velocity and field terms in the Lorentz force equation, but may be made explicit using the method of Boris. The implementation in Neptune computes time-step updates to the momentum and position for each particle in parallel across the available GPU or CPU cores.

To complete the EM-PIC formulation, the current density distribution is calculated from the particle ensemble during the particle update step:

$$\vec{J}(\vec{x}) = \sum_i q_i\vec{v}_i\,\delta^3(\vec{x}-\vec{x}_i)$$

This then couples the motion of charged particles back to the electromagnetic fields in the next time step via the current density source term. This source term is discretized such that the discrete current exactly conserves the charge represented by the particles as they move through the simulation volume.

2.2. Semiconductor band model

To adapt the model to semiclassical carrier dynamics in semiconductor materials, we replace the energy–momentum relation in Neptune with one representing the semiconductor band structure, with the momentum related to the wavevector, $\vec{p} = \hbar\vec{k}$.

In our initial implementation of the EM-EMC method we have implemented a single-band, non-parabolic Kane model for electrons:

$$\mathcal{E}(1+\alpha\mathcal{E}) = \frac{\vec{p}\cdot\vec{p}}{2m^*} = \mathcal{E}_0 \quad \Rightarrow \quad \vec{v} = \frac{1}{\sqrt{1+4\alpha\mathcal{E}_0}}\frac{\vec{p}}{m^*}$$

where α determines the non-parabolic term, and m^* is the effective mass. The remainder of the EM-PIC model carries over to the EM-EMC model.

2.3. Scattering

The model that we use for scattering of charge carriers in semiconductor materials, due to acoustical and optical phonon interactions and impurities, is similar to that used in conventional electrostatic EMC methods but the implementation must be adapted for the EM-EMC method.

In the electrostatic EMC method, the interval between scattering events may be shorter than the time step between updates to the numerical field solution. Therefore the particle path between field updates is evaluated as a sequence of free flights between scattering events while keeping the field solution fixed. Each free flight interval is evaluated from a random distribution, based on the total scattering probability for all scattering processes, and the actual scattering process is then selected at random based on their relative probabilities [5].

In the EM-EMC method, the CFL condition dictates a much shorter time step between field updates. Consequently, the probability of any scattering event taking place within a single time step is small. Particles may be evolved in time just as in the EM-PIC model, using the same time step as for the fields, except that after each particle momentum update the scattering probabilities for each process during that step are evaluated, and a specific scattering event (or no scattering event) selected according to a random distribution. If a scattering event occurs then the scattered particle parameters are computed exactly as in the electrostatic case and used in the next time step. The implementation in Neptune uses a proven parallel random number generator (PRNG) [6] to ensure a truly random sample is used.

2.4. Initial conditions

Prior to the start of a simulation each cell of the Cartesian grid is assigned a material type according to the user-defined device geometry, which may be vacuum, conductor, insulator or semiconductor. The permittivity and permeability of each cell is evaluated to define the discrete electromagnetic constitutive relations required by the field solution. The electromagnetic fields in the simulation are initialized everywhere to be zero.

The initial distribution of simulation particles is created to represent free carriers present in doped regions of semiconductor, spatially distributed according to the specified doping profile and initialized with a momentum distribution derived from the lattice temperature. The field-free condition implies local charge neutrality, so that each initial free charge effectively has an associated bound charge of the opposite sign fixed at its initial position.

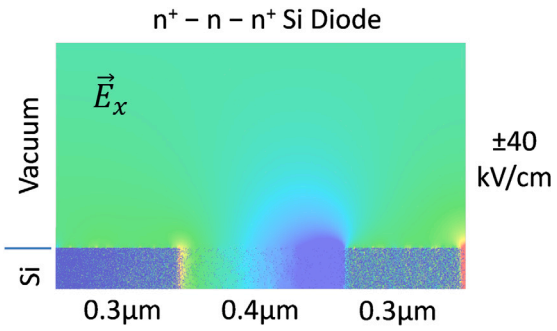

Fig. 1. Longitudinal electric field strength in a quasi-1D n^+—n—n^+ Si diode after steady-state operation is reached (red is +ve, green is zero, blue is −ve). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

2.5. Device simulations

Each simulation begins in a charge-neutral state with no applied voltages, and an ensemble of simulation particles distributed according to the donor density distribution. To introduce voltages to the device contacts, controlled currents are applied between electrodes to transfer charge and thereby generate electric fields in the device, which then interact self-consistently with the simulation particle ensemble until a steady-state solution is reached. Simulation particles are added and removed at the electrodes during the simulation to model Ohmic contacts.

3. Simulation examples

The Neptune implementation of the EM-EMC algorithm can perform parallel calculations using either CPU (multi-core) or GPU (many-core) processors, making maximum use of parallelism in each case to reduce execution times. All simulations were performed with a Supermicro workstation, with dual Intel Xeon Gold 6136 CPUs ($2\times12 = 24$ physical cores; 3.0–3.7 GHz clock rate) and an Nvidia Titan RTX GPU (4608 CUDA cores; 1.35–1.77 GHz clock rate). Single precision floating point calculations were used to maximize performance.

The implementation of the EM-EMC simulator is three-dimensional, but in our initial testing we perform 2D device simulations by constraining the third dimension of the grid to just two grid cells. The Cartesian grids may be non-uniform on each axis, and the simulation time step is computed automatically to ensure stability. The following two examples illustrate simulations of a Silicon diode and a Silicon MESFET.

3.1. Silicon diode

For the first example we model the n^+—n—n^+ Si diode shown in Fig. 1, having conducting electrodes at each end. The donor densities in each of the three regions are $5{\cdot}10^{17}$ cm^{-3}, $2{\cdot}10^{15}$ cm^{-3}, and $5{\cdot}10^{17}$ cm^{-3} respectively. The upper region is vacuum.

Charge is transferred between the electrodes during the simulation to apply 1 V across the diode and the simulation is continued to reach a steady state solution. Particles are injected at the cathode to maintain the local carrier density. Fig. 1 illustrates the final steady state solution, showing the longitudinal electric field component and particle ensemble distributions.

The following table summarizes statistics for the 2D diode simulation:

Grid size:	$400\times97\times2$ cells
Simulated time:	3.5 ps
Time step, Δt:	$5.3\cdot10^{-6}$ ps
Particle weight:	0.01
Average particle load:	41,000

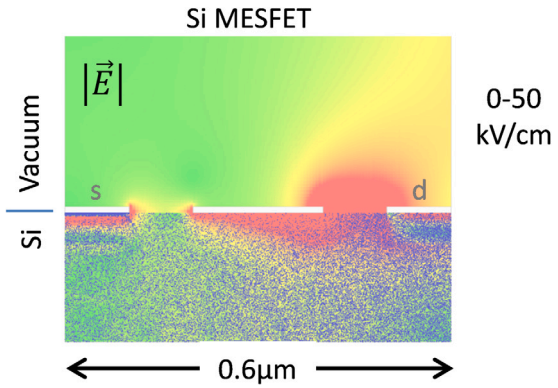

Fig. 2. Electric field magnitude and particle distribution in a 2D Si MESFET simulation after reaching steady-state operation.

Execution time using the GPU implementation was 2.9 min (4608 cores), compared to 20.3 min using the parallel CPU implementation (24 cores). Particle calculations represent approximately 40% of the execution time.

3.2. Silicon MESFET

In our second example we model a three-electrode Si MESFET device geometry, shown in Fig. 2. Source and drain electrode lengths are 0.1 µm, the central gate electrode length is 0.2 µm, with electrodes separated by 0.1 µm. The simulation thickness is 0.01 µm. The donor density in the semiconductor volume is $1\cdot10^{17}$ cm^{-3}, except under the source and drain electrodes where it is $3{\cdot}10^{17}$ cm^{-3} to a depth of 0.05 µm. The upper region is vacuum.

Charge is transferred between the drain and source electrodes to set up a potential difference of 1 V, generating a flow of carriers in the device, while the gate electrode is left floating. Fig. 2 illustrates the final steady state, showing the electric field magnitude and final particle ensemble distributions.

The following table summarizes statistics for the 2D MESFET simulation:

Grid size:	$120\times120\times2$ cells
Simulated time:	6.0 ps
Time step, Δt:	$9.2\cdot10^{-6}$ ps
Particle weight:	0.002
Average particle load:	52,000

Execution time using the GPU implementation was 2.7 min (4608 cores), compared to 26.2 min using the parallel CPU implementation (24 cores).

As a final measurement of the code performance, the same device was simulated in 3D by extruding the 2D model geometry to a simulation thickness of 0.4 µm. The following table summarizes statistics for the 3D MESFET simulation:

Grid size:	$120\times120\times80$ cells
Simulated time:	6.0 ps
Time step, Δt:	$9.2\cdot10^{-6}$ ps
Particle weight:	0.002
Average particle load:	2,090,000

Execution time using the GPU implementation was 163.7 min.

4. Conclusion

We have adapted an existing GPU-accelerated EM-PIC simulation code to perform simulations of charge transport in semiconductor device geometries using an electromagnetic ensemble Monte Carlo algorithm. Initial 2D and 3D simulations of a simple n^+—n—n^+ Si diode and a Si MESFET device geometry show qualitative results consistent

with the expected device operation, and illustrate both the required parameters for stable simulations and the code performance characteristics. The results indicate that the EM-EMC algorithm implemented for a modern GPU processor will be able to provide effective device simulations with acceptable simulation times.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors thank the Office of Naval Research (US) for funding support at the Naval Research Laboratory. Distribution Statement A: Approved for public release, distribution unlimited.

References

[1] Platzman PM, Wolff PA. Waves and interactions in solid state plasmas. 1st ed. Academic Press; 1973.

[2] Cooke SJ, Chernyavskiy IA, Stanchev GM, Levush B, Antonsen Jr. TM. GPU-accelerated large-signal device simulation using the 3D particle-in-cell code 'Neptune'. In: 39th IEEE International Conference on Plasma Science, Edinburgh, UK. 2012, p. 8–12.

[3] Jacobini C, Reggiani L. The Monte Carlo method for the solution of charge transport in semiconductors with applications to covalent materials. Rev Modern Phys 1983;56(3).

[4] Sellier JM. Archimedes – the free Monte Carlo simulator, https://www.gnu.org/software/archimedes/.

[5] Hockney RW, Eastwood JW. Computer simulation using particles. IOP Publishing Ltd.; 1988.

[6] Salmon JK, Moraes MA, Dror RO, Shaw DE. Parallel random numbers: As easy as 1, 2, 3. In: SC'11: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis. Seattle, Washington, USA: ACM; 2011, p. 1–12. http://dx.doi.org/10.1145/2063384.2063405, Article No.: 16.