High Performance Computing Aspects in Semiconductor Process Simulation

Andreas Hössinger[1], Paul Manstetten[3], Georgios Diamantopoulos[2],
Michael Quell[2], and Josef Weinbub[2]
[1]Silvaco Europe Ltd, United Kingdom
[2]Christian Doppler Laboratory for High Performance TCAD at the
[3]Institute for Microelectronics, TU Wien, Austria
andreas.hoessinger@silvaco.com

In recent years the gain in computing power was no longer related to a significant increase in processor core performance but was mainly related to an increase in the number of processor cores available within a CPU and to an increase in the number of CPUs available within a workstation. Nowadays the objective of any commercial software development is to make best use of this parallel computing power and obviously this also applies to semiconductor process simulation. In the past, semiconductor process simulation mainly dealt with the simulation of ion implantation and doping diffusion. With deeper penetration of 3D process technology all simulation steps dealing with the creation and modification of the geometry have received a lot more attention. The wide variety of technological applications and analysis requirements has also extended the need for an extensive hierarchy of simulation models or more precisely for simulation algorithms, because in those process simulation steps, different levels of model complexity, means different algorithms or even different data representations. Usually any such algorithms and any sub-algorithms therein requires a different approach towards optimally facilitating high performance simulation hardware.

In this work we would like to illustrate this for the case of etching simulation (embedded into a full process flow) and we would like to demonstrate recent approaches and achievements to better facilitate multi-core computing architecture. In contrast to the simulation of doping diffusion where the performance dominating steps are always the equation assembler and the linear solver, an etching process simulation has to be broken into much smaller pieces to identify the performance critical sub-modules and it is usually not only one of them which dominates. By breaking the simulation flow of an etching simulation into its main sub-modules one can also identify the various possible levels modeling for that process step. Level of modeling means that some of the sub-modules are just approximated and not simulated.

The choice of the modeling level determines the overall performance gain achievable by improving the performance of a single sub-module. Usually the requirement is to optimize the performance of all sub-modules separately, whereby every sub-module uses different and independent algorithms. The basic sub-module list for an etching simulation is:

(a) Transient loop

1. Reactor scale particle transport
2. Extraction of near surface and surface properties
3. Particle transport from the reactor scale to the feature scale and within
4. Interaction of surface with incoming particles
5. Extraction of surface velocity
6. Transient surface motion and interaction with volume properties

(b) Formation of final topology and volume data stage

We present how we have significantly improved the performance of (a-3) ("Particle transport from the reactor scale to the feature scale and within") by replacing an implicit representation of the surface, for modeling the interaction of the particles with the surface. Despite the multi-core scaling of the original implementation was already very good, we managed to further improve the total performance of that module by replacing the implicit surface representation with an explicit surface representation and by using efficiently parallel radiosity algorithms for ray surface interaction within the flux integrator [1][2]. Despite introducing overhead for implicit to explicit surface conversion the performance gain of Victory Process in typical application cases is up to a factor of 7, while maintaining the multi-core scalability.

We also managed to significantly improve performance of (a-6) ("Transient surface motion and interaction with volume properties") by developing a mesh hierarchy based parallelization and by fine tuning the data representation, of level-set re-distancing and level-set velocity extension [3][4]. Despite the inherently serial nature of some of these problems, we obtain a performance gain of Victory Process in typical application cases up to a factor of 3, as well as decent multi-core scalability.

[1] P. Manstetten *et al.*, Procedia Computer Science **108**, 245 (2017).

[2] P. Manstetten *et al.*, in Proc. SISPAD (2017).

[3] G. Diamantopoulos *et al.*, Advances in Computational Mathematics (2019), in press.

[4] G. Diamantopoulos *et al.*, in Proc. ICCSA (2017).