Parallelization of Monte Carlo Simulation for submicron MOSFET on Hypercube Multiprocessors

Chiang-Sheng Yao, Satoshi Sugino*and Robert. W. Dutton

Integrated Circuits Laboratory
AEL 203, Stanford University, Stanford, CA 94305-4055

Parallelized Monte Carlo device analysis for submicron MOSFET has been investigated on distributed memory parallel processors. In the parallelization algorithm, especially communication issues between many CPUs, access timing of nodes to the host's memory are involving of critical concern.

A device design method using Monte Carlo simulation for sub-micron MOSFET is becoming a realistic tool even for device engineers [1], [2] (Fig 1). However, still the huge computational time is one of the main draw-backs in this method. Certainly, the time required for many calculations might be too long for daily engineering use, in spite of several speedup techniques having made on the method recently. Though both the windowed method and coupling with Drift Diffusion simulation have already been used for the Monte Carlo analysis, it takes several hours even for a high end work station (Table 1). On the other hand, parallel processing has a possibility of driving down not only the computational time but also cost.

In the present work, a parallel algorithm of MC has been implemented into BEBOP [3] and developed on Intel iPSC/860 (Table 2). The speedup performance of the parallelization is measured using Amdahl's Law [4] (Appendix A).

In our MC experiment, when more than 10000 electrons are used, we found that more than 90% of the time was spend in the parallelized part of all, which is corresponding to the calculation of movement and scattering of particles (Fig. 2). The remaining parts, for example, the poisson solver and statistics routines, are not considered to be parallelized. The ideal speedup for 31 nodes is expected to be nearly 12 times faster compared to the case of one i860 node and 11 times faster compared to the case of SUN4/490 sparc server. Instead of using the host computer with of an i386, one of the 32 nodes in the hypercube machine is used instead for a loading host program.

At this moment we have only parallelized the portion for calculating trajectories of particles, since it takes almost 63% of the total elapsed time. The next time comsuming part of the code is for the calculation of scattering events. According to previous work [5], only 2% of the elapsed time is spent on message passing between nodes. Our results also support the conclusion that the parallel computing method is very attractive for Monte Carlo device analysis. As future work, the portion of code for scattering also should be parallelized, especially when the code has to handle the so called "full band structure", because it requires many calculations for interpolation of data.

References

[1] F.Venturi et al., IEEE Trans.on CAD, vol.8,p.380, 1989.

[2] S.Sugino et al., IEDM Tech. Dig., p459, 1990.

[3] "*BEBOP*"- MONTE CARLO SIMULATOR (University of Bologna.)

[4] Johon L Hennessy and David A Patterson, "Computer Architecture A Quantitative Approach", Morgan Kaufmann Publishers, Inc., 1990.

[5] Thesis of Doreen Yining Cheng, at Stanford University. 1988.

$^{-1*}$ visiting scholar from Matsushita Electric Works, 1048 Kadoma, Osaka 571 Japan

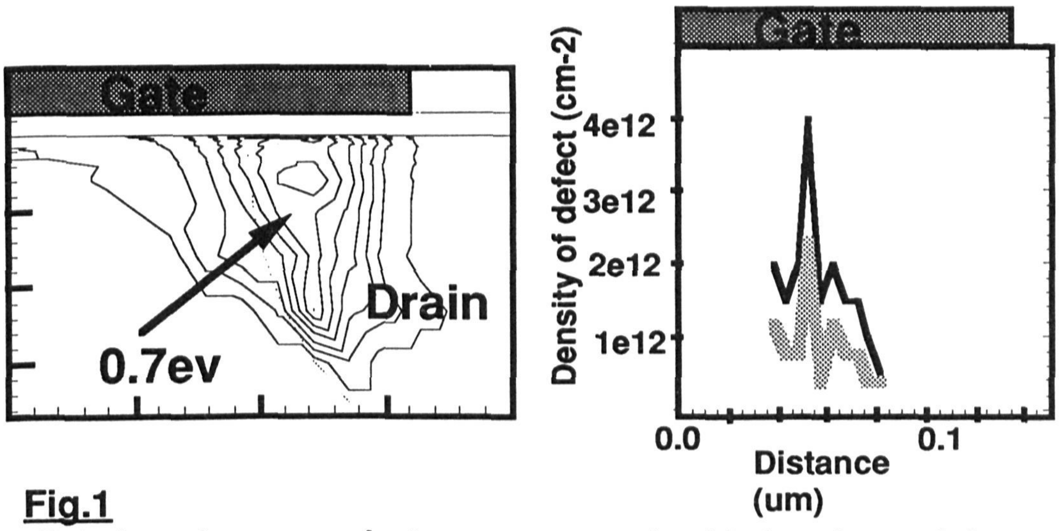

Fig.1
Relation of average electron energy and oxide interfaece defects

Left : Average electron energy around drain.
Right: Associated interface defects at the Si-SiO2 interface.

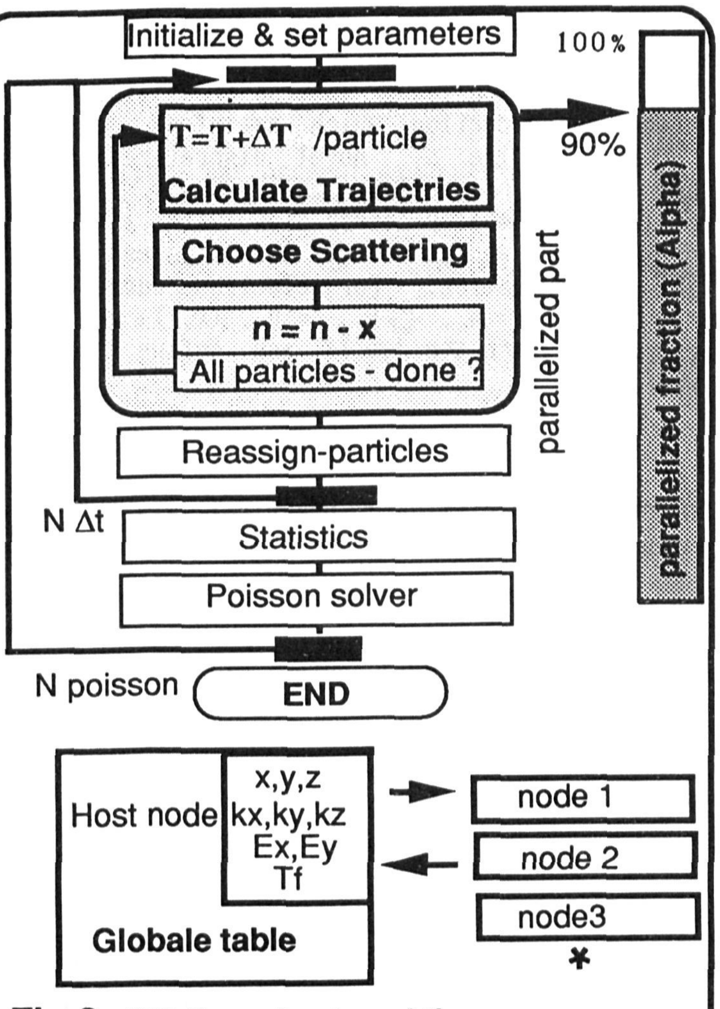

Fig.2 MC flow chart and the parallelization algorithm

Table 1 MC benchmark

Machine	Elapsed Time
sun sparc 1+	669min/bias
sun4/490	401min/bias

Table 2 Intel iPSC/860

processor	i860(64bit)x32
Memory	16Mbyte/node

Relative performance /node

i860 is **42%** faster than Sparc1+

sun4/490 is **43%** faster than i860

Amdhal's Law (appendix A)

$$\textbf{speedup} = \frac{1}{(1-\alpha) + \dfrac{\alpha}{\textbf{Nnode}}}$$

$\alpha =$ **Parallelized fraction of MC**

Nnode=Number of CPU in the cube