Massively Parallel Computation for Monte Carlo Device Simulation

A. Hiroki, S. Odanaka, and A. Goda
VLSI Technology Research Laboratory, Semiconductor Research Center
Matsushita Electric Industrial Co., Ltd.

1. Introduction

The Monte Carlo device simulation plays an important role in better understanding very small device physics [1], [2]. However, the huge computation time is one of the main drawback to use the Monte Carlo method for physical-based device design. This paper describes a new parallel algorithm and performance of Monte Carlo device simulations. A parallel version of a 2D Monte Carlo simulator : BEBOP [3] is developed using a new-type massively parallel computer : ADENART-256 [4],[5]. The load balancing in the parallel computation is further investigated. The ADENART allows the high-speed parallel computing of Monte Carlo device simulations. The computation time is 59 times faster than that on an EWS.

2. Parallel Algorithm for Monte Carlo Device Simulation

Fig. 1 shows a flowchart of the Monte Carlo device simulation. The parallelized part of the program involves calculations for the free - flight of particles, the particle behavior at the boundary, and the momentum and energy after scattering event, which take more than 90% of the total computation time. Parallel computation for such calculation procedures is achieved by assigning each processor the task of handling several particles and maintaining the load balancing. The approach is different between massively parallel computers [6]. Fig. 2 shows parallel processing for particles using the ADENART. The ADENART is a MIMD computer having 256-PEs (Processing Element). As shown in Fig.3, the processor array can take two positions alternatively : row and column positions. The network is designed to transfer from the data array shared among processors at the row position to that at the column position (or vice versa). The position x, momentum p, and other physical valuables of each particle are defined by the direction valuables written by the two-dimensional array x(/ i /, j) and p(/ i /, j). i corresponds to the processor ($i = 1, ..., 256$) and j to the order number in each processor. In this case, the ADENART allows the parallel computing over i and a serial computation is performed along j.

3. Results and Discussions

Fig. 4 demonstrates parallel performance of Monte Carlo device simulation using the ADENART. The simulated device is a 0.25 μm n-MOSFET. The number of particles used here was 10,000. The result is also compared with the computation time on an EWS : Solbourne - series 5 (The performance is almost identical to SUN 4/490). The computation time for the free - flight of particles, which takes 81% of the total computation time, is greatly reduced to 1/71 of that on the EWS. The real performance reaches to 123 MFLOPS. The computation times for the boundary and the scattering routines are reduced to 1/13 and 1/12 of that on the EWS, respectively. As a result, the computation speed of the parallel version of BEBOP using the ADENART is 59 times faster than that on the EWS.

Moreover, the load balancing in the parallel computation is investigated. To maintain the load balancing, the particles are sorted according to the three events of free-flight, boundary, and scattering. Then the particles at each event are evenly distributed to the processors. Fig. 5 compares the results calculated with and without sorting the particles at each iteration. It is found that the computation time for the free-flight is almost identical in both methods. This means that all particles for the free-flight are evenly distributed all over the processors using the method shown in Fig. 2. For the boundary and scattering routines, the load balancing is improved. In fact, the computation times are 89% and 53% of those calculated without sorting the particles, respectively. However, this improvement of the load balancing has little impact on reducing the total computation time, when the high-speed computation of these routines is achieved.

Acknowledgements

The authors wish to thank Dr. T. Takemoto and Dr. M. Inoue for their encouragements. They also wish to thank Prof. T. Nogi of Kyoto University for parallel computation, Prof. E. Sangiorgi of University of Bologna, and Prof. R. W. Dutton of Stanford University for Monte Carlo simulation.

References

[1] M. F. Fischetti and S. E. Laux, Phys. Rev. B. vol 38, pp. 9721-9745, 1988.
[2] F. Venturi et al., IEEE Trans. CAD, vol. 10, pp. 1276-1286, 1991.
[3] "BEBOP" -- MONTE CARLO SIMULATOR (University of Bologna).
[4] T. Nogi, "Parallel Computation," in Pattern and Waves - Qualitative analysis of nonlinear differential equations - , North - Holland,Amsterdam, pp. 279-318, 1986.
[5] K.Kaneko et al., "Processing element design for a parallel computer," IEEE MICRO, vol.10, no.2, pp.26-38, 1990.
[6] S. Sugino, C.-S. Yao, and R. W. Dutton, Proceedings of SISDEP, 1991.

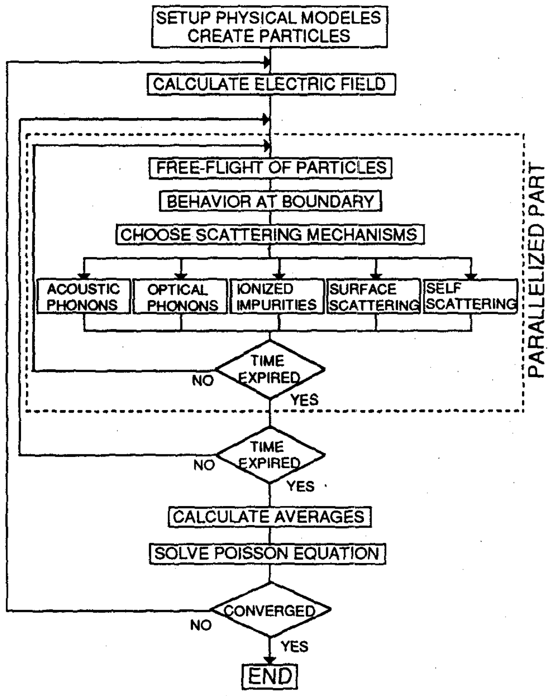

Fig.1 Flowchart of the Monte Carlo device simulation. The parallelized part involves calculations for the free-flight of particles, the particle behavior at the boundary, and the scattering event.

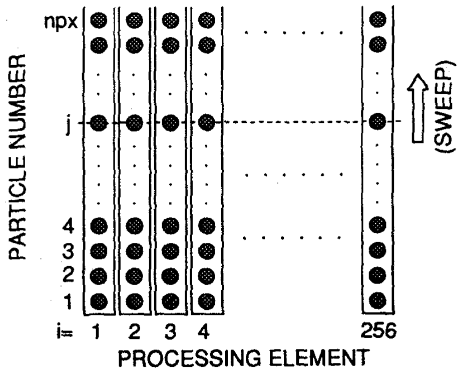

Fig.2 Parallel processing for particles. All particles are evenly distributed all over the processors.

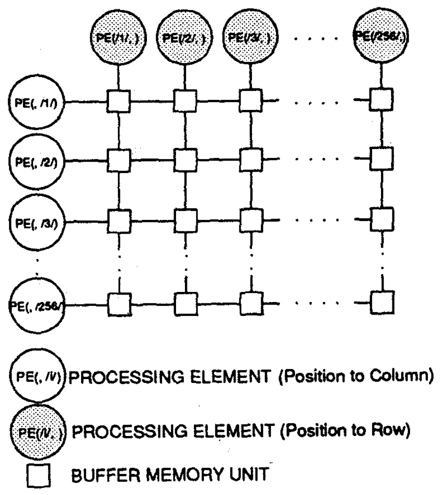

Fig.3 Logical system architecture of the ADENART. The processor array can take two positions alternatively.

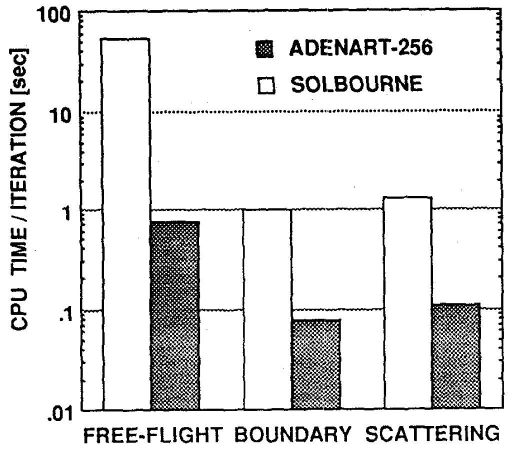

Fig.4 Parallel performance of Monte Carlo simulation using the ADENART. The computation speed using the ADENART is 59 times faster than using an EWS : Solbourne-series 5. The performance of this EWS is almost identical to SUN 4/490.

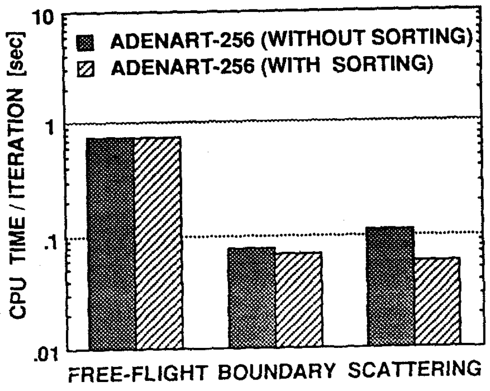

Fig.5 The load balancing is investigated. The computation time is compared to that with sorting the particles at each iteration.