PERFORMANCE ANALYSIS FOR HIGH-PRECISION INTERCONNECT SIMULATION

R. Heinzl, M. Spevak, P. Schwaha, T. Grasser and S. Selberherr

△Christian Doppler Laboratory for TCAD in Microelectronics at the Institute for Microelectronics
○Institute for Microelectronics, Technical University Vienna,
Gußhausstraße 27-29/E360, A-1040 Vienna, Austria
E-mail: {heinzl|schwaha|spevak|grasser|selberherr}@iue.tuwien.ac.at

KEYWORDS
Mesh generation, error estimation, mesh quality, high performance computing, programming paradigms

ABSTRACT
This work analyzes the performance of high-precision interconnect simulation tools on refined meshes with guaranteed accuracy. On the one hand, the integrated circuits are subject to an ongoing miniaturization which results in ever increasing computing power. On the other hand, the simulation of these integrated circuits demands more sophisticated simulation methodologies such as better resolution of geometrical features or more complex surface topography. We show that modern microprocessor architectures and memory hierarchies impose performance limits on the simulation time.

INTRODUCTION
Down-scaling of integrated circuits to the deep sub-micron regime and beyond increases the influence of interconnects on circuit behavior drastically. Parasitic effects are becoming more and more important as devices get faster and line widths smaller. These effects become the limiting factor for further improvements of circuit speed. An essential step in technology computer aided design (TCAD) is the optimization of these parameters, which demands vast amounts of computer resources, CPU-time and memory. Therefore the performance of current computer systems is essential for an optimal simulation flow. The overall performance of computer systems with a given set of applications depends on numerous factors and can not be attributed to only the speed of the central processing unit (CPU). Among the most important factors is the connection of the CPU to the computers main memory [6]. In the early days of computers the employed memories were faster or at least comparable in speed to the CPU. Naturally the focus of the evolution of CPUs was to increase their processing speed. This goal was greatly to the CPU. Naturally the focus of the evolution of CPUs deployed memories were faster or at least comparable in speed main memory [6]. In the early days of computers the em- fant of these parameters is the connection of the CPU to the computers main memory [6]. In the early days of computers the employed memories were faster or at least comparable in speed to the CPU. Naturally the focus of the evolution of CPUs was to increase their processing speed. This goal was greatly to the CPU. Naturally the focus of the evolution of CPUs deployed memories were faster or at least comparable in speed main memory [6]. In the early days of computers the em-
programming accomplishes both, a general solution for most of the application scenarios and highly specialized code parts for minor, but also important, scenarios without sacrificing performance [2, 11]. This has already been demonstrated in the field of numerics and yields figures comparable to Fortran [13, 18], the previously undisputed candidate for this kind of calculations.

Based on these techniques we developed a high performance simulation engine based on the SAP tools [15]. With template meta-programming [1], the functional specification can be used very similarly to the original mathematical formulation, as can be seen in this work. Due to this new programming technique and the corresponding evaluation at compile time the calculation associated with the specified equations is highly optimized by the compiler and thereby ensures excellent run-time performance often superior to highly hand-optimized code. In our case C++ was the language of choice, because currently no other language offers sufficient support for all the necessary programming techniques to enable the required level of abstraction.

Our own investigations in the field of compiler optimization and compiler comparison has shown significant differences in optimization behavior and run-time performance of modern programming techniques [8].

### INTERCONNECT MODEL

Our interconnect simulation tools use the finite element method to discretize the partial differential equations resulting in a system of equations that eventually has to be linearized, and thereafter solved with a preconditioned conjugate gradient algorithm [10]. To give a glimpse on details we consider a typical problem of forming the equation system.

The problem we consider is posed in the following way:

\[
\mathcal{L}\Psi := \text{div}(-\varepsilon \text{grad}(\Psi)) - \varrho = 0 \quad \text{in } \Omega \tag{1}
\]

\[
\Psi - \Psi_D = 0 \quad \text{on } \partial \Omega, \tag{2}
\]

where \( \varepsilon \) denotes the (isotropic) permittivity of the considered domain, which is assumed to be constant in an element of the tessellation. Due to the weak formulation using Galerkin finite elements [19] weighting coefficients for the local element matrices have to be derived for tetrahedra:

\[
g_1^e = \frac{\varepsilon K_{11}^e + K_{22}^e + K_{33}^e}{\text{detJ}} \tag{3}
\]

where \( K \) is the Jacobian matrix of the node points of the tetrahedra. Due to operator overloading different mathematical structures such as scalars, vectors, and even matrices can be handled using identical notation. With this transformation into code results in the following code snippet:

```c++
double g1 = epsilon*(K11*K11 + K21*K21 + K31*K31)/detJ;
```

To assemble the system matrix, a local element matrix has to be assembled: \( S_e \) stands for the local element stiffness matrix which is derived by:

\[
S_e = g_1 S_1 + g_2 S_2 + g_3 S_3 + g_4 S_4 + g_5 S_5 + g_6 S_6; \tag{4}
\]

The corresponding C++ code reads:

\[
S_1 - S_6 \text{ means the linear form function matrices and } g_1 \text{ - } g_6 \text{ are calculated at the nodes of the tetrahedra in the global coordinate system [4].}
\]

### PERFORMANCE ANALYSIS

It should be noted that for high precision simulations it is essential to model the simulation domain as exactly as possible. The accuracy and efficiency of a finite element and finite volume simulation strongly depends on the quality of the tessellation of the domain. As a consequence we introduced a comprehensive solid modeling and mesh generation and adaptation approach [7].

Figure 1 presents the example structure under investigation with a coarse mesh for the following performance analysis. In order to obtain sufficiently accurate results, the mesh size typically has to be in the order of \( 10^4 \) to some \( 10^5 \) nodes.

For a rigorous analysis we evaluate three different implementations in C++ and compare them to a hand-optimized Fortran 77 implementation on different computer architectures. The first implementation is based on the GNU GCC `valarray` data-type which is a standardized data-structure representing a mathematical vector. This data-type has shown excellent performance on different computer architectures with recent compilers. Secondly, we utilize the Blitz++ [18] library, which introduced high performance calculation comparable to Fortran 77 directly in C++. Lastly, a naive C++ implementation is used that creates two temporary objects, one for the addition and one for the assignment. As a consequence all elements have to be accessed three times.

The tests were performed on four different computer systems:

<table>
<thead>
<tr>
<th>CPU type</th>
<th>Clock speed</th>
<th>RAM</th>
<th>Compiler</th>
<th>MFLOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pentium 4</td>
<td>2.8 GHz</td>
<td>2 GB</td>
<td>GCC 4.0.2</td>
<td>2510.9</td>
</tr>
<tr>
<td>AMD64</td>
<td>2.2 GHz</td>
<td>2 GB</td>
<td>GCC 3.4.4</td>
<td>3543.0</td>
</tr>
<tr>
<td>IBM P655</td>
<td>8x1.5 GHz</td>
<td>64 GB</td>
<td>GCC 4.0.2</td>
<td>16361.7</td>
</tr>
<tr>
<td>G5</td>
<td>4x2.5 GHz</td>
<td>8 GB</td>
<td>GCC 4.0.0</td>
<td>24434.0</td>
</tr>
</tbody>
</table>

Figures 2-5 compare these different approaches on different hardware architectures. The y-axes is labeled with million operations per second. The vector addition consists of 3 operations, two additions and one assignment.
For vector lengths smaller than $10^4$, cache hits reveal the full computation power of the CPU, longer vectors show the limits imposed by memory bandwidth. The poor performance of naive C++ code is indeed remarkable.

Based on these observations of restrictions due to the limited bandwidth, we illustrate the influence of a problem’s size on the overall finite element. We therefore investigate our test structure with different levels of refinement. By resolving a three-dimensional simulation domain, the number of points easily exceeds the critical threshold and thereby leads to severe problems caused by memory bandwidth restrictions (Figure 6).

Investigations of parallelization attempts on multi-processor machines (G5) show that the inner loop of the finite element assembly cannot be parallelized easily. On the one side, the update mechanisms of the element matrices may require access to the same part of memory simultaneously, which could be avoided by a different assembly scheme, e.g. node-based assembly. On the
other hand, the inner loops are compiled very efficiently and only bounded by memory bandwidth. Tests with four CPUs have shown, that parallel assembly does not speed up the total assembly process at all. Restrictions resulting from memory bandwidth completely negate any benefit due to parallelization.

CONCLUSION

Although the observed performance issues are presented for the field of interconnect simulation, the main findings are certainly transferable to other areas, such as process and device simulation. Memory bandwidth is the limiting factor as we have seen from our benchmarks. In summary, highly expressive code in C++ on different platforms and computer architectures does not show any abstraction penalty, where naive C++ code does not perform well. Regarding parallelization, current memory links hardly provide enough bandwidth to accommodate the throughput required to satisfy the computational performance of multiple cores.

REFERENCES


BIOGRAPHIES

RENÉ HEINZL studied electrical engineering at the Technische Universität Wien. He joined the Institute for Microelectronics in November 2003, where he is currently working on his doctoral degree. In April 2005 he achieved first place at the doctoral competition at the EEIFCT in Brno. His research interests include process simulation, solid modeling, and adaptive mesh generation for TCAD with special emphasis on three-dimensional applications.

PHILIPP SCHWAHA studied electrical engineering at the Technische Universität Wien. He joined the Institute for Microelectronics in June 2004, where he is currently working on his doctoral degree. His research activities include circuit and device simulation, device modeling, and software development.

MICHAEL SPEVAK studied electrical engineering at the Technische Universität Wien. He joined the Institute for Microelectronics in December 2004, where he is currently working on his doctoral degree.

TIBOR GRASSER received the Ph.D. degree in technical sciences, and the “venia docendi” in microelectronics from the Technische Universität Wien in 1999, and 2002, respectively. He is currently employed as an Associate Professor at the Institute for Microelectronics. Since 1997 he has headed the Minimos-NYV development group, working on the successor of the highly successful MiniMOS program. In 2003 he was appointed head of the Christian Doppler Laboratory for TCAD in Microelectronics, an industry-funded research group embedded in the Institute for Microelectronics. His current scientific interests include circuit and device simulation and device modeling.

SIEGFRIED SELBERHERR received the Ph.D. degree in technical sciences from the Technische Universität Wien in 1981. Since that time he has been with the Technische Universität Wien as professor. Dr. Selberherr has been holding the “venia docendi” on “Computer-Aided Design” since 1984. As of 1988 he has been chair professor of the Institute for Microelectronics. From 1998 to 2005 he served as Dean of the “Fakultät für Elektrotechnik und Informationstechnik” at the Technische Universität Wien. His current topics of interest are modeling and simulation of problems for microelectronics engineering.