USING GRID COMPUTING IN PARALLEL ELECTRONIC CIRCUIT SIMULATION

Bojan Anđelković, Marko Dimitrijević, Vančo Litovski

Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia, phone: +381 18 529321, e-mail: (abojan, marko, vanco)@elfak.ni.ac.yu

This paper presents the concept of a parallel electronic circuit simulator running in the Grid infrastructure. The simulator implements new algorithm that parallelizes equation formulation for nonlinear analog elements to reduce long simulation runtimes of complex electronic circuits at transistor level. Simulations execute on a Beowulf cluster using MPI for communication between workstations. Moreover, the paper describes the development of Grid interface for the parallel simulator. It enables a designer to perform demanding simulations in the Grid environment on distant shared computer resources. The interface provides submission of simulation jobs together with simulation models, simulation run on a distant computer cluster and retrieval of simulation results.

Keywords: Parallel circuit simulation, Grid, Beowulf cluster

1. INTRODUCTION

The simulation process of modern complex integrated circuits involves intensive calculations requiring significant processing power, large operating memory, as well as big storage capacity. These requirements pertain to the fact that a large number of ordinary (and, potentially, partial) nonlinear differential equations have to be solved for long running excitations. Therefore, simulation runtimes are very long. One possibility to reduce such long runtimes is to parallelize the simulation algorithm and use parallel computers to execute simulations. In this approach complex calculations necessary during the simulation process can be distributed over different workstations/processors and performed simultaneously.

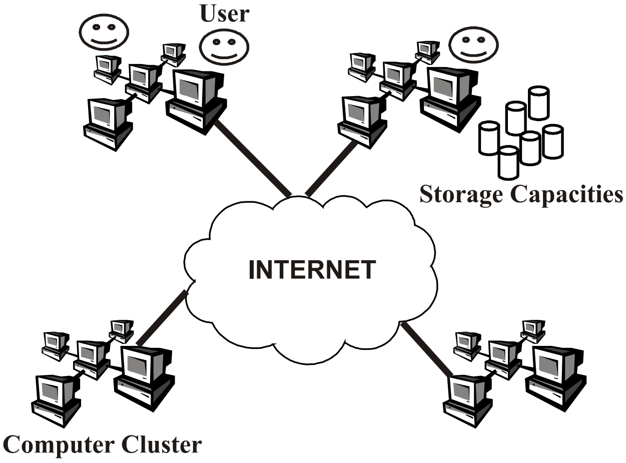

Fig. 1 A Grid structure

The development of low-cost personal computers and gigabit LAN network connections offers a possibility for implementation of inexpensive distributed multiprocessor systems such as computer clusters [1]. Furthermore, the growth of Internet and WAN links of great capacity and speed led to development of the computational Grid (Fig. 1).

It is a highly heterogonous and geographically distributed computing system consisting of interconnected shared computer resources that users can utilize for their demanding tasks.

2. Parallel Electronic Circuit Simulation on the Grid

2.1 Parallel simulator implementation

Complex mixed-signal electronic circuits at transistor level are modeled using algebraic equations and nonlinear Ordinary Differential Equations (ODE), in order to simulate their behavior in time domain. ODEs are discretized and that generates sets of nonlinear algebraic equations. The system of nonlinear equations is solved iteratively with the help of linearization i.e. by application of Newton methods [2]. At every iteration and time instant the matrix entries of the system of linear equations have to be recalculated. Having in mind the number of matrix entries, the number of iterations and the number of time instants, it is necessary to provide an immense computational effort. In order to reduce long simulation runtimes the calculation of matrix entries and equation formulation for nonlinear circuit elements should be parallelized.

Such parallelization of equation formulation process is implemented in the simulator Alecsis [3]. It is a mixed-signal and mixed-domain simulator with proprietary hardware description language AleC++ [4] developed for modeling and simulation of complex systems containing different kinds of devices and subsystems. The developed simulator with parallel simulation capability is called *pAlecsis* (Parallel Analog and Logic Electronic Circuits Simulation System) [5]. It executes on a Beowulf cluster using MPI [6] (Fig. 2).

Parallel equation formulation is implemented using master-slave algorithm [7]. In this algorithm calculation of matrix entries for nonlinear circuit elements per time and per iteration is distributed to multiple cluster nodes (slave nodes) and they are calculated simultaneously. At the same time master node calculates matrix entries for specific number of nonlinear elements, as well as for constant and linear time dependent elements. In order to minimize communication between cluster nodes, necessary data structures for all elements of the circuit are generated on all nodes simultaneously during compilation of the simulation model. In that way all cluster nodes have the information necessary to generate matrix contributions for all elements. To achieve equal load of all nodes, each node of the cluster performs equation formulation and calculation of matrix entries for equal number of nonlinear circuit elements. After generation of entries (for all elements) on one slave, they are sent to the master node using appropriate MPI routines for exchanging data (Fig. 2). When the master node receives matrix entries from all slaves, it flushes them to the circuit matrix and performs one iterative simulation step. In order to enable calculation of matrix entries on slave nodes, the master node should send to the slaves vectors of solutions of the system of equations for the two past time instants and previous iteration (denoted by *vp1*, *vp2* and *vi*, respectively in Fig. 2). Appropriate MPI routines for transferring data are used to send and receive these vectors.

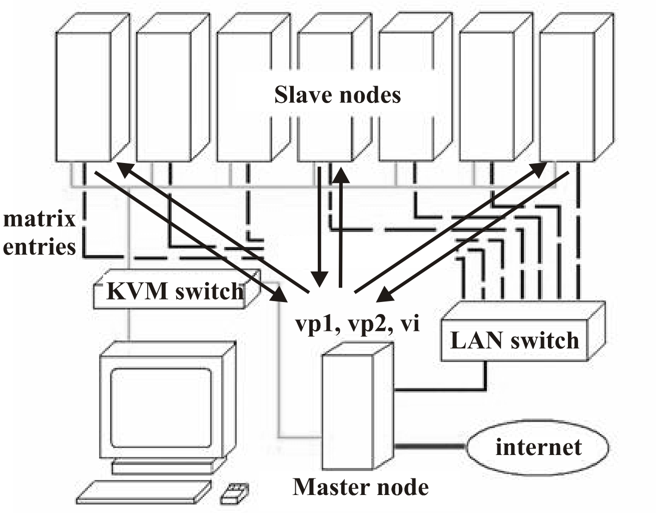

Fig. 2 Parallel simulation in pAlecsis on a Beowulf cluster

Performance analysis of the implemented parallel simulator can be found in [5].

2.2 Using Grid computing in parallel simulation

Fig. 3 shows the structure of a grid application. In order to enable a designer to run parallel simulations on the Grid resources, it is necessary to develop appropriate Grid interface for the simulator. Such interface should provide submission of simulation jobs together with simulation models, simulation run on a distant computer cluster and retrieval of simulation results.

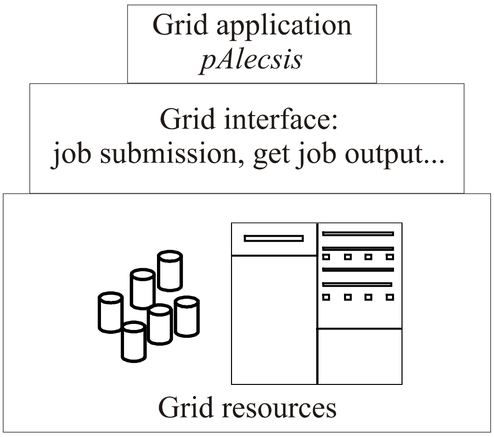

Fig. 3 Basic structure of a Grid application

Implementation of the *pAlecsis* simulator that enables to perform parallel simulations using the Grid infrastructure is shown in Fig. 4. The access point to the Grid is the User Interface (UI). This can be any machine connected to the Grid where the user has a personal account and where their user certificate is installed. From an UI, a user can be authenticated and authorized to use the Grid resources, submit simulation tasks (called jobs in Grid terminology) for execution, cancel jobs, retrieve the output of finished jobs, and show the status of submitted jobs. In order to execute simulations using Grid resources, the user should submit a simulation job from the UI to a Resource Broker (RB). Simulation jobs are described in the Job Description Language (JDL) [8]. Fig. 5 shows a part of the JDL file for simulations in *pAlecsis* on the Grid. The job description specifies the file containing the model for simulation in AleC++ (*circuit.ac* in Fig. 5) and, this file is, together with *pAlecsis* executable, initially copied to a Resource Broker. This set of files is called the *Input Sandbox*.

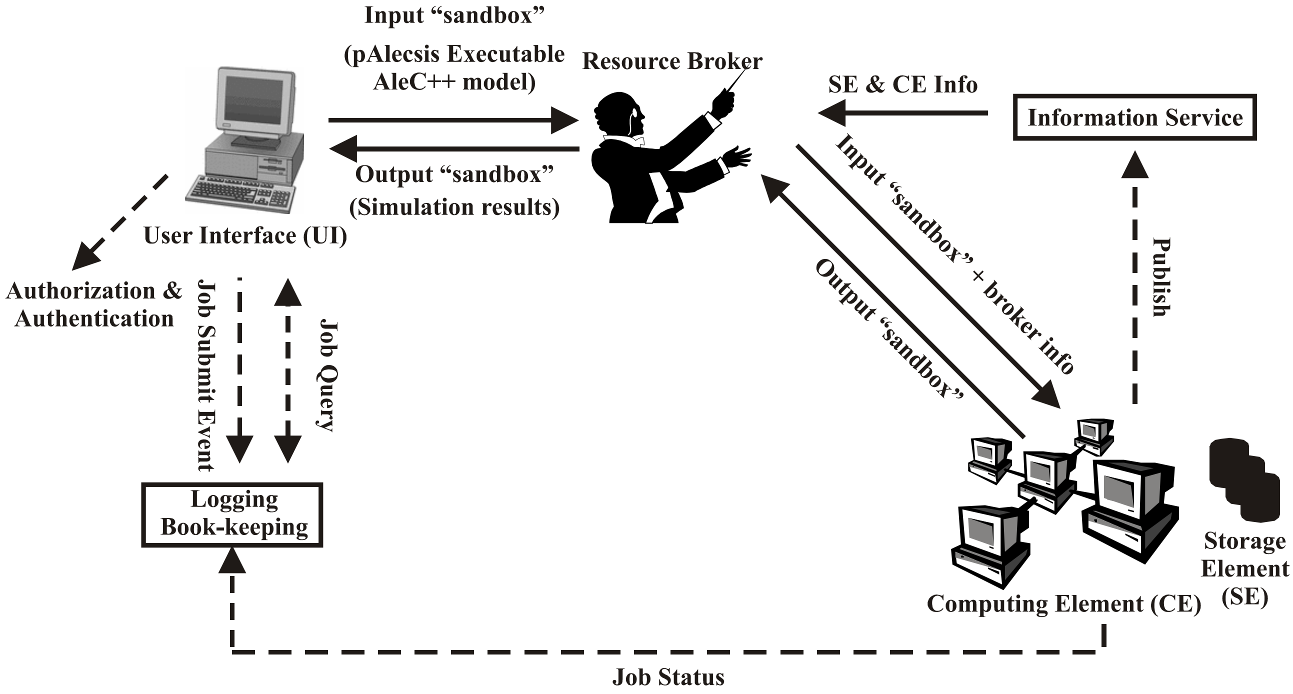

Fig. 4 Parallel simulation in pAlecsis on the Grid

```
# Type of JDL
Type = "Job";

# Type of job
JobType = "MPICH";

# Number of nodes
NodeNumber = 3;

# The executable parameter specifies the executable/command that will be run on the CE
Executable = "MPI_alec.sh";

# Any arguments that are needed by the executable
Arguments = "pAlecsis circuit.ac";
...
# The files to be transferred between the UI and the WN before the job execution
InputSandbox = {"MPI_alec.sh","pAlecsis","circuit.ac"};

# The files to be transferred between the UI and the WN after the job execution
OutputSandbox = {"mpiexec.out","circuit.ar"};
```

Fig. 5 Part of the JDL file for pAlecsis simulations on the Grid

The job description also specifies minimal number of cluster nodes for simulation. The Resource Broker looks for the best available Computing Element (CE) to execute the job. To do so, it interrogates the Information Service, to determine the status of computational and storage resources. A Computing Element, in Grid terminology, is some set of computing resources localized at a site (i.e. a cluster, a computing farm). A CE includes a collection of Worker Nodes (WNs), the cluster nodes where the jobs are run. The choice of CE to which the job is sent is made in a process called match-making, which first selects, among all available CEs, those which fulfill the requirements expressed by the user (MPI support, number of cluster nodes/processors etc.) and which are close to specified input Grid files. The Resource Broker then chooses the CE with the highest rank, a quantity derived from the CE status information which expresses the "goodness" of a CE (typically a function of the numbers of running and queued jobs). Since *pAlecsis* needs MPI support on a Beowulf cluster to run simulations, it is necessary to specify in the job description file that simulations are MPI job type. It informs the Resource Broker to look for CEs with installed MPICH runtime environment. The *Input Sandbox* files are copied from the RB to an available Worker Node of the CE where the job is executed. If the simulation ends without errors, the output files containing the results of the simulation and specified by the user (*circuit.ar* in Fig. 5) in the so called *Output Sandbox* are transferred back to the RB node. At this point, the user can retrieve the output of his simulation to the UI. The file *mpiexec.out* contains information about cluster nodes on the Grid where the simulation executed. Fig. 6 shows a sample file for one such simulation.

```
Master node:
"wn88.ulakbim.gov.tr"

Slave node(s):
"wn68.ulakbim.gov.tr"
"wn87.ulakbim.gov.tr"
```

Fig. 6 Information about used Grid cluster nodes for the simulation

3. Conclusions

This paper describes the development of Grid interface for the parallel electronic circuit simulator. It enables a designer to perform demanding simulation tasks not just on the local Beowulf cluster, but also on distant shared clusters connected to the Grid. The interface provides all necessary features to execute simulations in the Grid environment, such as submission of simulations jobs and models, monitoring of simulation tasks, and retrieval of simulation results.

The development of the *pAlecsis* simulator is a part of SEE-GRID-2 (South-eastern European grid-enabled einfrastructure development 2) project co-funded by the European Commission under the FP6 research program.

4. References

[1] Sterling, T., *Beowulf Cluster Computing with Linux*, MIT Press, 2001.

[2] Litovski, V., and Zwolinski, M., *VLSI Circuit Simulation and Optimization*, Chapman and Hall, London, 1997.

[3] Mrčarica, Ž., Ilić, T., Glozić, D., Litovski, V., and Detter, H., *Mechatronic Simulation Using Alecsis: Anatomy of the Simulator*, Proceedings of Eurosim'95, pp. 651-656, 1995.

[4] Litovski, V., Maksimović, D., and Mrčarica, Ž., *Mixed-Signal Modeling with AleC++: Specific Features of the HDL*, Simulation Practice and Theory 8, 2001, pp. 433-449.

[5] Anđelković, B., Litovski, V., Petković, P., *Implementation and Performance Analysis of Parallel Circuit Simulator on Beowulf Cluster*, Proceedings of the ETRAN 2007 Conference, 2007. (to be published)

[6] Gropp, W., Lusk, E., and Skjellum, A., *Using MPI: Portable Parallel programming with the Message-Passing Interface, second edition*, MIT Press, 1999.

[7] Gropp, W., Lusk, E., and Thakur, R., *Using MPI-2: Advanced Features of the Message-Passing Interface*, MIT Press, 1999.

[8] Burke, S., Campana, S., Peris, A., Donno, F., Lorenzo, P., Santinelli, R., Sciab`a, A., *gLite3 User Guide*, https://edms.cern.ch/file/722398/1.1/gLite-3-UserGuide.pdf, 2007.