

Efficient Signal and Power Integrity Analysis Using Parallel Techniques

Tao Su (tsu@sigrity.com), Xiaofeng Wang (xfwang@sigrity.com),
Zhengang bai (walsebai@sigrity.com), Venkata Vennam (vvprakash@sigrity.com)

Sigrity, Inc.
4675 Stevens Creek Blvd., Suite #130, Santa Clara, CA 95051
Phone: 408-260-9344. Fax: 408-260-9342

Abstract: This paper describes the application of parallel algorithms for analyzing the electronic packages and PCB's in a fast and efficient manner. The effectiveness of multithreading at various stages of the analysis has been illustrated. The application of distributed computation to further accelerate the solution process is demonstrated.

Key words: Multi-threading, Distributed computation, Signal integrity, Power integrity

I. Introduction

Current advances in computer technology are rapidly bringing what was once thought to be impossible to within the reach of every day users. Today, we see computer simulations replacing experimentation in the field of package and board analysis. Dramatic improvements in the performance and reductions in cost of performing large scale simulations have made these changes feasible. With the advent of dual-core and quad-core machines, and the powerful 64-bit platforms, the electrical package analysis can be performed more efficiently, with corresponding reductions in the turn-around time.

This paper applies the multithreading and the distributed computation to the package/board analysis. The advantages of applying parallel algorithms to various stages of the analysis flow have been demonstrated with the help of several examples. Critical issues that ultimately control the efficiency have been discussed.

II. Multithreading

Multithreading is the capability of being able to use multiple threads of execution simultaneously in an application. When an application is run, it is called a process and each of these processes contains at least one thread (or many concurrent threads). A typical flow for the electrical package analysis is presented in Fig.1. The electrical package analysis tools can take advantage of multithreading at various stages of the flow.

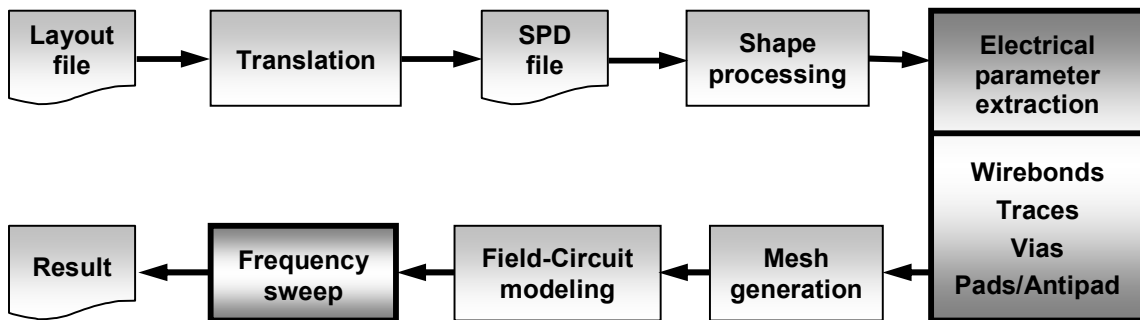


Fig.1. Electrical package analysis flow.

It is important to identify the most time consuming stages, and target them for parallel simulation. For example, in the preprocessing stage, the electrical characterization of traces, coupled lines, vias, pads, etc. can be done in parallel by using multiple threads. In addition, one needs to distinguish between logical and physical processors on the machine, and be intelligent enough to make efficient use of them. A physical processor is the actual CPU, while the Hyper-Threading technology makes it appear as two logical processors. It only provides thread-level parallelism on each physical processor by pipelining certain parts of instruction evaluation. As a result, one doesn't gain much by Hyper-Threading, especially when the parallel operations are similar in nature.

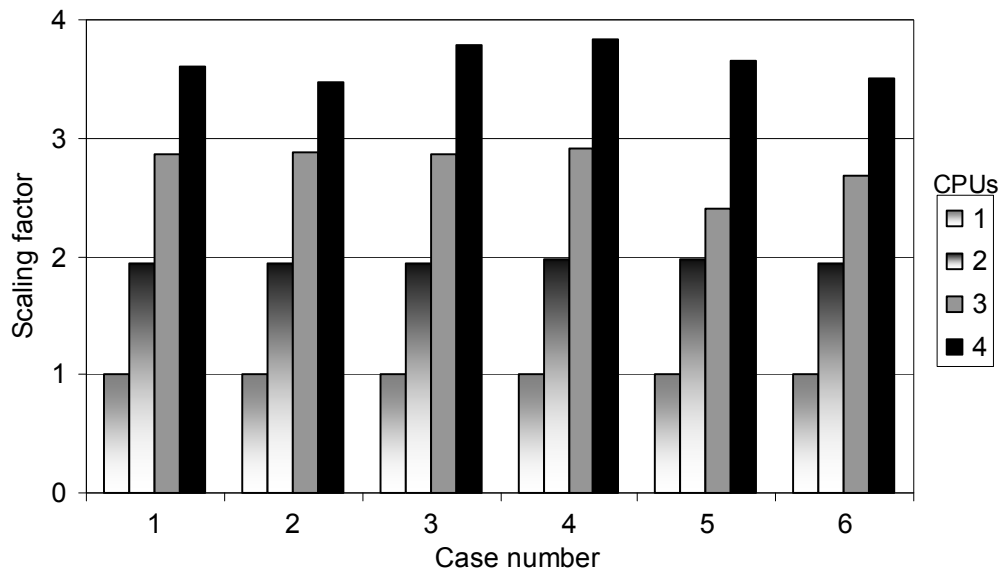


Fig.2. Scaling of CPU time as a function of number of processors for Pad modeling.

This is illustrated with the help of an example, as shown in Fig.2 and 3. The trace and pad characterization stages have been carried out in parallel by using multiple threads. The simulations have been carried out on an EM64T machine with four physical processors running at 3192MHz and having 16GB of physical memory. The computational efficiency is presented as a function of number of processors. It can be seen that good scaling performance has been obtained as a function of CPU numbers, which is an indicator of well balanced loads to the processors. These characterization stages use math kernel libraries that are independently threaded by OpenMP. Care should be taken to balance the threads among the host and the libraries to avoid overloads.

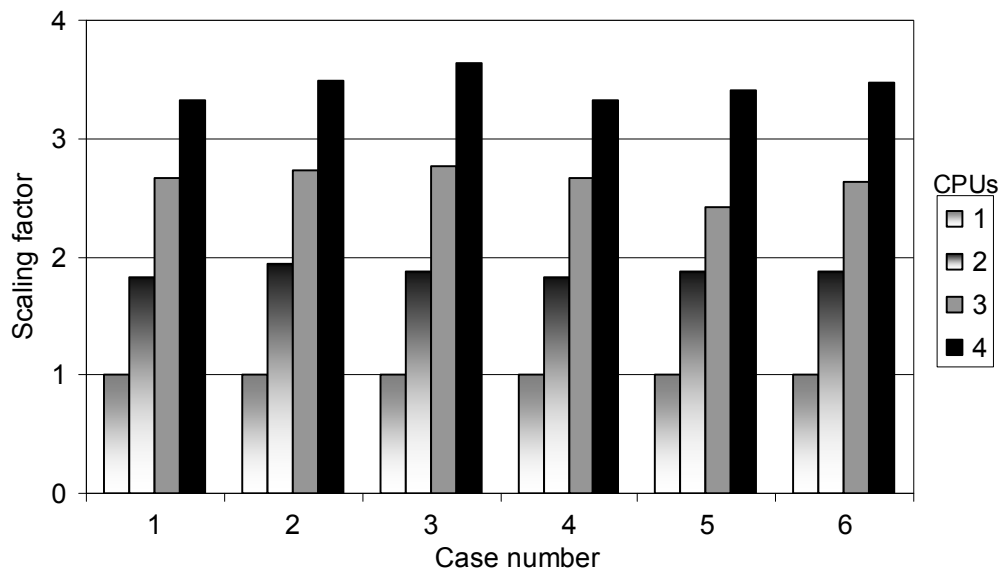


Fig.3. Scaling of CPU time as a function of number of processors for transmission line modeling.

III. Distributed Computation

In section II, we presented multi-threading on a single machine, which is a shared memory type of architecture. This section deals with distributed computing, which is becoming popular due to the advent of high speed computer networks. The basics flow of distributed computing are illustrated in Fig. 4 as implemented in the package analysis tool.

The host communicates with remote servers through sockets, which is a generalized communicated scheme that is independent of operating system. It controls the frequency band splitting, and creates separate designs for each server. The simulation data is dynamically passed back from servers, and stored at the host. The scheme allows mixing a 32-bit and 64-bit OS, thus allowing flexibility in a corporate environment. Note that each server can access multiple CPUs, if available, of itself.

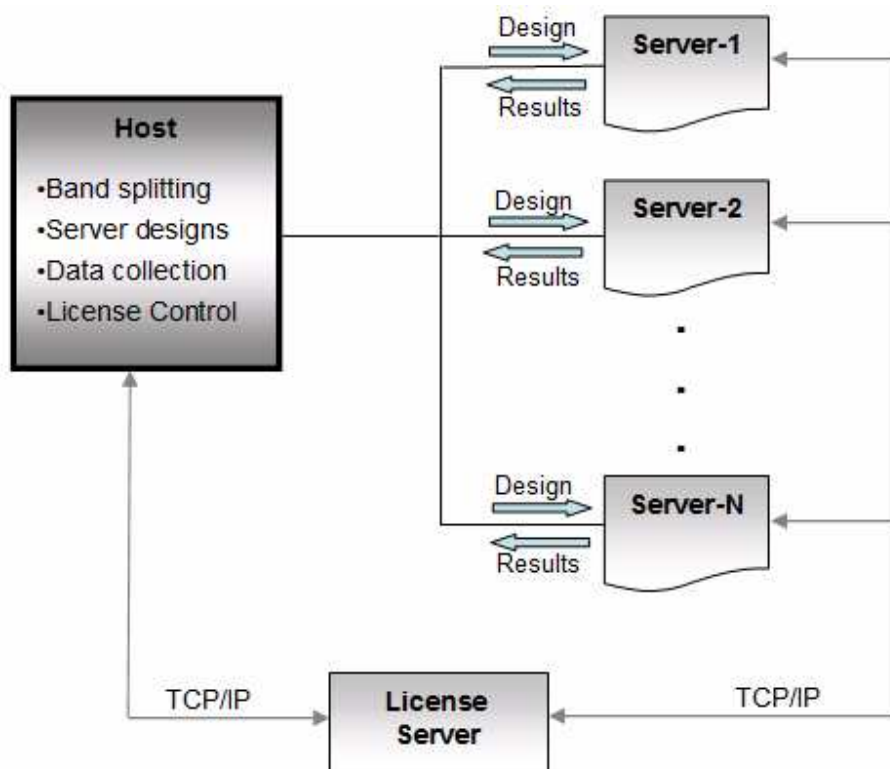


Fig. 4. Schematic of distributed analysis flow.

The scheme has been tested by using a 32-bit machine as host, and two 64-bit machines as servers. When the simulation is done by the host and one server, the total simulation is faster by a factor of 1.88, and when the second server is added, the scaling factor turned out to be 2.65. This includes the time for data exchange, and the network delays.

References:

- [1] Mark Towfiq, An open interface for network programming under Microsoft windows, Version 1.1, 1993.
- [2] Yinghua Lu, Cui Jian and Xueping Yu, "Computational electromagnetic on distributed systems", Proc. Computational Electromagnetics, pp.557-560, 1999.
- [3] Scott J. Norton, and Mark D. Dipasquale, Thread Time: The multithread programming guide, Pentice Hall, Edition 1, 1996.
- [4] Andrew Sohn, Mitsuhsa Sato, Namhoon Yoo, and Jean-Luc Gaudiot, "Effects of multithreading on data and workload distribution for distributed-memory multiprocessors", IEEE Proc. IPPS, pp.116-122, 1996.