More and more developers of design automation applications are exploiting parallel processing as a means of accelerating CAD applications. Many design automation algorithms partition well in a parallel environment. This, together with the declining cost of high-performance microprocessors, has made parallel processing a viable resource for developers of CAD applications. Hancock and DasGupta (see box below) provide good introductory reading on parallel processing and how it impacts design automation.

This special issue of IEEE Design & Test is devoted to parallel processors for design automation acceleration. The articles focus on those under development or in the prototype stage, while my guest editorial surveys those that are commercially available.

**Surveys and tutorials.**


IS THERE A "BEST" MACHINE?

Developers of CAD applications for parallel processing can choose between special-purpose and general-purpose machines. Special-purpose machines have to run off a general-purpose host, which performs all the support functions, such as preparing the model, controlling the simulation, and analyzing the results. Developers need to take this additional pre- and post-processing into consideration when evaluating the effectiveness of using special-purpose hardware.

The most widely used special-purpose parallel machines are for logic simulation. Other machines have been developed for such applications as routing, design-rule checking, and circuit simulation, but these are not as successful. One reason is that these tasks are done only once in the design cycle, while logic simulation is an iterative task. Second, the algorithm and problem domain are not as stable as those in logic simulation.

Thus, overall, special-purpose parallel machines for applications other than logic simulation may not be useful for very long. On the positive side, they can provide near optimal performance in special applications.

In contrast, general-purpose hardware accelerates a wider class of problem domain. The availability of affordable parallel machines in recent years has triggered a high level of interest in parallel applications. Many of these machines are built using low-cost general-purpose microprocessors. As a rule, the supporting software for programming applications is on the machine.

Parallel machine configurations range from two or more processors sharing a common bus to networks of processors and memory. Control programs supporting these environments range from special-purpose operating systems to hybrids of Unix. To support the development of applications, general-purpose parallel machines usually provide languages that incorporate constructs for message send/receive or process creation/deletion.

Even logic simulation can be done effectively on general-purpose machines. If the logic simulation done on special-purpose parallel hardware can be adapted and programmed onto general-purpose parallel machines, the same piece of hardware can be used to speed up the entire simulation process—from model preparation to results analysis. Partitioning may be required to allocate portions of the circuit to particular processors, but this function can also be implemented on the machine itself.

The simulation algorithm can be programmed on a general-purpose parallel machine to handle models of different levels of abstraction for mixed-level simulation. It offers the flexibility of algorithm changes without requiring any changes to the hardware. The same piece of hardware can be used to accelerate other applications besides logic simulation.

SPECIAL-PURPOSE ACCELERATORS

Special-purpose accelerators have been built for many major design functions, including logic, fault, and circuit simulation and physical design. Blank gives a good survey of special-purpose hardware.

LOGIC/FAULT SIMULATION

Special-purpose parallel hardware such as the Zycad LE series, the Yorktown Simulation Engine, NEC HAL, and Fujitsu SP are specifically designed to accelerate logic simulation. These machines reportedly can simulate many millions of logic gates per second. They run off a general-purpose host machine that performs all support functions, such as model preparation, simulation control, and results analysis.

The high performance level of logic simulation accelerators is due to the duplication of identical units of special-purpose logic dedicated to logic simulation. Each processor is usually pipeline to exploit the low-level parallelism in the simulation algorithms. The processors are often tightly coupled by dedicated interconnection networks.

To support communication among processors, the Zycad LE series uses a high-speed common bus and a control processor; the YSE, a crosspoint switch; the NEC HAL, a multistage network; and the Fujitsu SP, a two-level-hierarchy shared bus and a control processor.

Among special-purpose parallel hardware designed specifically to accelerate concurrent fault simulation is the Zycad FE series. A recent addition to the Zycad family of accelerators is the Zycad FE with the Hyperfault feature. Hyperfault is a front-end that computes the tests before the fault evaluator performs the simulation. With Hyperfault, the FE can achieve up to 90% test coverage depending on the complexity of the circuits used.

A more recent entry into the marketplace is Silicon Solutions' MACH 1000 logic/fault simulator. It has a bus-oriented system architecture similar to that of the LE series and can accommodate up to eight simulation processors and a control processor. Each simulation processor is constructed using a four-stage pipeline and has a simulation capacity of up to 64,000 modeling elements.
The peak performance of each simulation processor is 500,000 events per second. This translates into performance that is hundreds of times faster than a VAX 11/780. Depending on the size of circuit simulated and the job balance, however, the average performance of each processor is more likely to be from 200,000 to 250,000 events per second, according to Robert Matthews, marketing manager of Zycad/Silicon Solutions. A four-processor configuration increases the system's performance from 500,000 to 1 million events per second.

As is the case with other simulation accelerators, each processor has a limited set of instructions to compute logic functions on only four-input/one-output primitives. A table lookup is used to locate the logic function for evaluation.

**Circuit Simulation**

Like logic simulation accelerators, circuit simulation accelerators duplicate special-purpose logic, except in this case it is dedicated to circuit simulation. Special-purpose hardware for circuit simulation is beginning to appear in the marketplace. For example, Simucad is marketing a circuit simulation accelerator, Powerspice, based on the SPICE2 algorithm. It is designed around the Sequent Balance multiprocessor system, and with equivalent software, it reportedly runs 20 times faster than the VAX 11/780.

**Physical Accelerators**

Hardware engines have been designed, built, and demonstrated for such physical design areas as wiring and design-rule checking. Silicon Solutions markets a hardware design-rule checker, called Fast-Mask. The standard system consists of a 68010 front-end and up to four 68020 sub-systems sitting on a common Q-bus. The 68010 performs I/O servicing and scheduling activities between the four processors.

The peak performance of each sub-system is two times that of a 12-MHz 68010. In practice, given a reasonably complex job stream, a four-processor system is about 7.0 to 7.5 times faster than the 68010, on average. More than four processors may be used, but experience shows that four processors fit most problem sizes without causing bus contention in the front-end processor.

**General-Purpose Parallel Hardware**

General-purpose parallel computing systems have been available for many years, but their high cost (millions of dollars) have priced them out of the general scientific community. In recent years, more affordable microprocessor-based parallel systems, with an average cost of between $100,000 and $500,000, have been appearing. These less expensive systems have generated a lot of interest from disciplines with computationally intensive tasks.

General-purpose engines can have two architectural configurations: shared memory and distributed memory. A shared-memory system consists of two or more processors sharing a common bus and memory. A distributed-memory system consists of a network of processors, and memory that is distributed as local memory to each processor.

**Shared-Memory Systems**

Sequent Computer Systems' Balance, Alliant Computer Systems' FX series, and Encore Computer's Multimax are examples of systems composed of multiple processors sharing a common bus and memory. The Balance and Multimax are built with off-the-shelf microprocessors, while the FX series uses a proprietary microprocessor. All are programmable, either in C or Fortran, and provide library support for shared-memory access (semaphores, barriers) and process scheduling (fork, join).

Both the Sequent and Alliant systems have limited ability to automatically compile Fortran DO loops across multiple processors. This feature enables programs to run on parallel hardware while remaining compatible with existing computer systems. These systems, in general, are flexible enough to support both multiple jobs running in parallel and one job running on multiple processors. The result is high multiuser throughput and efficient use of processors.

In both research and industry, many CAD applications have been developed on these systems. The University of California, Berkeley, is conducting research into circuit simulation and simulated-annealing-based placement as applications for the Balance.

Teradyne is marketing an accelerator, called DataServer, for their Lasar logic, fault, and timing simulator. DataServer is designed around the Balance. The simulation tasks run on optimized simulation coprocessors, while other general-purpose tasks, such as I/O, model compilation, and pattern compilation, are performed on general-purpose processors.

The simulator allows jobs to run in parallel on multiple processors, while nonsimulation tasks run simultaneously on general-purpose processors. It also supports multiuser access to the system. The simulator is reported to be 15 to 25 times faster than the VAX 11/780, depending on the number of users and the nature of the jobs, according to Robert Hotchkiss, DataServer project manager.

Recently, Endot implemented an architectural simulation tool, called n.2p, on the Balance. On a 12-processor system, the simulator yielded slightly better than eight times the performance of a single processor.

The Alliant FX series currently supports a wide range of industry CAD applications, including circuit simulation with HSPICE; logic/fault simulation with Silos, Verilog, and CADAT; analog/digital simulation with Salt; test generation with Testgrade; design verification with Dracula; process simulation with Suprem; and device simulation with Pisces. Unfortunately, many of these applications do not use the full parallel processing capability of the system. However, a parallel version of HSPICE is available, and parallel versions of Pisces and Suprem are under development.

BBN Laboratories' Butterfly has an architecture with elements of shared memory and message passing. The Butterfly interconnects up to 256 processors over a circuit-switching network that directly connects many processors, without many point-to-point connections. The system implements the best from both local and global memory concepts by having all memory modules distributed across every processor node. At the same time, processors are tightly coupled, allowing them to directly reference memory anywhere in the system.
DISTRIBUTED-MEMORY SYSTEMS

Intel’s iPSC, NCUBE’s model, Floating Point Systems’ T series, and the Connection Machine from Thinking Machines are examples of a system that is composed of a network of processors, and memory that is distributed as local memory to each processor. Topologies for connecting these processor-memory elements vary.

The iPSC, NCUBE, and T series use a hypercube interconnect topology. The processor-memory elements are mapped onto an n-dimensional geometric figure called a hypercube. The longest path across a hypercube is n links. For example, in a 3-D hypercube, each processor is linked to its three nearest neighbors. Each processor node independently executes its own sequence of instructions and accesses its own local memory. Processors communicate via message passing to access nonlocal memory. All three systems support C and Fortran, with library primitives for message passing. There has been some research into the development of CAD applications on the iPSC and NCUBE, for example, device simulation, switch-level simulation, placement, and routing.

The Connection Machine is a more elaborate example of a hypercube system. The machine is a large array of 65,536 one-bit processors, each processor associated with 4096 bits of local memory. The basic replicated unit of the system is a custom-built IC carrying 16 small processors and a device for routing communications. The 16 processors on each chip are connected by a switch that effects a direct connection between any pair of processing units. Up to 4096 of these chips are interconnected to form a 12-D hypercube.

Unlike the iPSC and the Balance, the Connection Machine is a data-parallel machine. Every processor simultaneously executes the same instruction on data objects stored in its local memory. For instance, the machine can be set up for circuit simulation by assigning each device and each node to a separate processor-memory element.

All devices compute their parameters in parallel. Data is sent in parallel to nodes, and the nodes compute their voltages in parallel. Data is propagated by movement along pointers, which represent connectivity. The machine’s routing hardware automatically establishes the necessary communication paths.

The Connection Machine uses a conventional host that provides all the infrastructure for program development and communication with other computers. It can be viewed as a memory extension of the host, in which partitions of the memory have their own units that execute parallel operations. The host can be either a VAX or a Lisp machine. Extensions to C and Lisp are provided to support the parallel programming model of the Connection Machine, such as special constructs for defining parallel data objects and pointers for non-local-memory references.

A circuit simulator has been implemented on the system that permits simulation of up to 1 million devices. Actual simulation shows that up to 64,000 devices can be simulated within a few minutes. Not all the capabilities of the system were used in this algorithm, however.

Besides circuit simulation, such other computationally intensive CAD applications as 3-D process and device simulation and simulated-annelling-based placement have been implemented on the Connection Machine. A parallel version of the Timberwolf annealing program computed the placement of a 6000-part design in less than two hours using 32,000 processors. An equivalent job is estimated to take 400 hours on a VAX 11/780.

There is ongoing research into design rule checking, routing, switch-level simulation, and the development of a memory-resident design database.

THEME ARTICLES

For this special issue, I have chosen three articles that illustrate the uses of parallel processing in the acceleration of CAD applications (see also “D&T Roundtable” and “D&T Conferences,” IFIP Workshop on CAD Engines,” for a discussion on the prospects of parallel processing for CAD applications).

The first article, “Multipurpose Parallelism for VLSI CAD on the R3P,” by Frederica Darema and Gregory Pister, describes a multipurpose experimental parallel system that is being developed at IBM T.J. Watson Research Center. Short for research parallel processing prototype, the R3P is a tightly coupled system with an architecture similar to that of the Butterfly. It supports development of programs written in C and Fortran.

Several VLSI CAD applications, such as circuit simulation, circuit simulation, and placement by simulated annealing, have been programmed to run on the R3P. Placement by simulated annealing is the topic of this article. The article demonstrates that if the number of processors equals 10% to 20% of the objects being placed, an efficiency of 80% is possible.

The second article, “A Multiprocessor-Based Programmable Accelerator,” by Pramita Agrawal et al. describes an accelerator whose architecture lies somewhere between that of a general-purpose parallel machine and a special-purpose parallel machine. MARS (short for microprogrammable accelerator for rapid simulations) can efficiently implement a wide range of computationally complex algorithms.

MARS achieves high performance by using highly pipelined and parallel architecture. Each processor contains custom hardware to accelerate operations on data structures commonly used for simulation. Flexibility is achieved through custom-designed microprogrammable and reconfigurable processing elements.

While MARS can accelerate many graph-related problem solutions, its architecture is ideally suited for performing event-driven simulation. The anticipated performance from MARS while programmed as a logic simulator is about 1 million gate evaluations per second.
The third article, "An ISMA Lee Router Accelerator," by Thomas Ryan and Edwin Rogers proposes the use of a family of application-specific iterative state machine arrays for design automation acceleration. The simplicity of the basic ISMA cells allows unique engine behavior to be realized by user-specific hardware.

Problems suited for ISMA acceleration are those that can be represented on a regular bit plane and contain inherent parallelism. Placement, routing, design-rule checking, and switch-level circuit simulation fall into this category.

To demonstrate the ISMA's application to design automation acceleration, the design of a low-cost ISMA chip for two-layer Lee routing is described. It is estimated to route 1000 two-point nets of length 2500\(\alpha\) over a (2048\(\times\))\(^2\) surface in under half a minute.

T

here are many ways to use parallelism for the acceleration of CAD applications. The choice between special-purpose parallel hardware and general-purpose (or multipurpose) parallel hardware is usually a matter of economics. General-purpose parallel systems, however, offer the most flexibility and the widest applicability.

With the availability of low-cost, high-performance general-purpose parallel processors, more and more designers are using these machines for the global acceleration of CAD applications. These general-purpose systems may someday accelerate a wider class of CAD applications and provide a more integrated approach to accelerating the design cycle.

To effectively use these systems designers will need new algorithms and techniques for mapping CAD applications onto parallel machines, especially algorithms that distribute the load evenly among processors and minimize the overhead from synchronizing activities.

While much has been accomplished in the development of parallel hardware, very little progress has been made in the development of tools for programming parallel systems. Good programming tools are needed to assist developers in designing and implementing applications that will use parallel hardware effectively. For example, optimizing compilers are needed to customize the code to the target hardware. The quality of the programming tools has a direct impact on the usability of the parallel system and hence the development cost of CAD applications.

Acknowledgments

I thank all the authors and reviewers who contributed to this issue. I am also grateful for the help and guidance from Vishwani Agrawal, D&I's editor-in-chief, and Akihiko Yamada. A special thank-you to my husband for his support and encouragement.

Last but not least, I thank Mentor Graphics Corp. for the use of its facilities.

References


Kit Yoke Tham is a senior software engineer with Mentor Graphics Corp., working on logic, fault, and timing simulation. Her current interests include parallel simulation techniques, VLSI CAD tools, hardware description languages, and expert systems for CAD. Previously, she was with Intel Corp.'s Corporate CAD Group and Intel Scientific Computers. Tham holds a PhD, an MSc, and a BSc in computer science from the University of Manchester, England. She is a member of the Computer Society, the IEEE, and ACM, as well as an editorial board member of the Computer Society Press (IEEE). Her address is Mentor Graphics Corp., SW Creekside Pl., Beaverton, OR 97005-7191.