

R. Brown, J. Hayes, and T. Mudge. Rapid prototyping & evaluation of high-performance computers. Proc. Conf. Experimental Research in Computer Systems, NSF Experimental Systems, Ed. L. Snyder, Washington DC, June 1996, pp. 159-168.

Rapid Prototyping & Evaluation of High-Performance Computers

Rich Brown, John Hayes, Trevor Mudge

Advanced Computer Architecture Lab

The University of Michigan

Conference on Experimental Research in Computer Systems

NSF Experimental Systems, June 21 - 22, 1996

Abstract

This report discusses our work with an emulator based on field programmable gate array technology. This technology has made possible the construction of hardware emulators capable of emulating systems having a million gates at clock speeds of 1 MHz and more. Thus it is now possible to run realistic workloads on prospective designs and measure the results. There are three areas that these new emulators can be applied to: 1) validation; 2) concurrent hardware and software development; and 3) rapid prototyping of design alternatives. The report outlines the technology of emulators discussing strengths and weaknesses. It discusses our experiments with a second generation emulator. Concluding remarks address the future potential of this technology.

1. Introduction

The dramatic growth in integration levels has resulted in integrated circuits that routinely contain over a million gates. This has had a major impact on the way in which computers and digital systems in general are designed and implemented. Computer-aided design is essential and success in the marketplace depends on the ability to produce novel, complex, computer-based designs extremely fast. Short time-to-market is the key to competitiveness in the industry, placing a premium on short, but high-quality design-and-verify cycles.

These developments are likely to intensify with integration levels doubling every two years and device speed increasing almost as quickly. This doubling is good news and bad news for the computer designer. The good news is, of course, that much more sophisticated architectures are possible. The bad news is that the difficulty of validating and evaluating these new architectures increases with their level of sophistication. In particular, validating designs is consuming 30-40% of the total manpower required to complete a typical design. What is worse is that this fraction is growing. Furthermore, the competitiveness of a new computer design depends more than ever on having evaluated a sufficiently broad range of architectural alternatives and implementations before freezing a specification. When these requirements of validation and evaluation are coupled with time-to-market pressures, it is clear that significant breakthroughs are needed in validation and rapid prototyping environments.

The new technology of field programmable gate arrays (FPGA's) promises to help in solving these problems. This technology has made possible the construction of hardware emulators capable of emulating target systems having millions of gates at clock speeds of 1 MHz and more. Thus it is now possible to run realistic workloads on prospective designs and measure the results. In particular there are three areas that these new emulators can be applied to:

1. Validation
2. Concurrent hardware and software development
3. Rapid prototyping of design alternatives

Until now, the successful commercial applications of this new emulation technology has been in the first two areas, however, we believe it also has the potential to allow computer designers to quickly experiment with novel ideas and to obtain accurate evaluations of their performance. By using the rapid prototyping capabilities of emulation, the design of future computer systems can be placed on a more scientific footing.

This report discusses our experiences with a second generation hardware emulator — the Quickturn Enterprise system, acquired as part of NSF Grant in Experimental systems in 1993 [1]. The authors are all associated with the Advanced Computer Architecture Laboratory in the EECS Department of The University of Michigan. This lab has been in existence since 1985 and currently includes eight faculty and 60 graduate students. As the name suggests, the

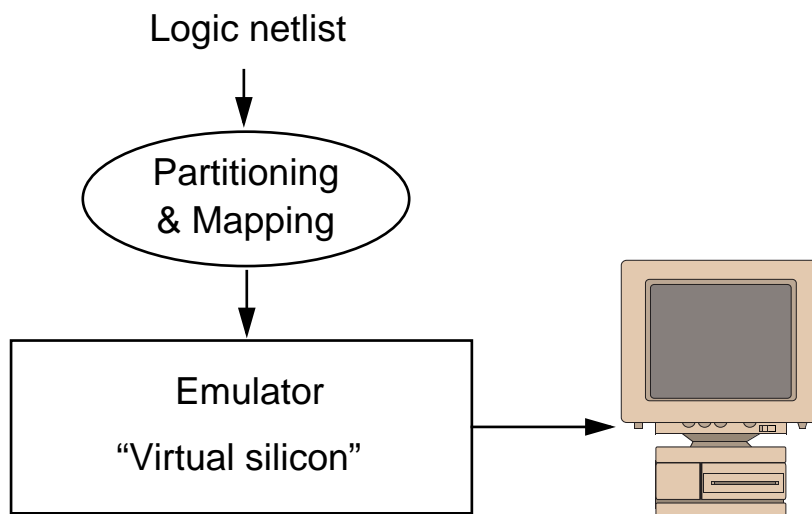
members of the lab are engaged in research into various aspects of computer architecture, reliable computing, testing, performance evaluation, logic design and CAD for computers. The lab has a history of building a variety of experimental digital systems, and as such provided a suitable situation to experiment with new validation and prototyping technologies.

This report is organized as follows. The next section provides an outline of FPGA-based emulators and discusses how these emulators fit into the development cycle of a new system. Section 3 discusses two recent experiences and current projects. Section 4 concludes with some remarks on the technology and its future outlook.

2. FPGA-based Emulation

2.1. Background

As noted earlier, the introduction of FPGA technology has led to the re-emergence of emulation. The idea is simply to assemble a large number of regularly interconnected FPGA chips into a box and provide software that can map the netlist of a digital system to be emulated onto the FPGAs. The resulting configuration of FPGA logic can emulate the logical functioning of the target system and can be run at some appreciable fraction of the intended speed of the target system. FPGA-based emulation creates a logically equivalent machine. This new approach to emulation differs from the traditional idea of emulation in which the microcode of a general purpose computer can be changed to mimic a different computer. In this approach the emulation takes place at the RTL level of microprogramming.



The steps in emulation are shown above. Emulation on FPGA-based systems is orders of magnitude faster than the traditional alternative, software simulation on a general purpose computer. In particular, emulation speed, while only a fraction of actual speed, is still fast enough to exercise the target system far more extensively than a pure software simulation could. Emulation clock speeds of several MHz are not unusual. For very high performance systems (several hundred MHz) this represents a slow down of about 100. For, more modest systems with much slower clocks real-time or close to real-time emulation is possible. This technology goes by the industry name of “computer aided prototyping.”

To illustrate the speed advantage of emulation consider the example of a 50 K gate digital system intended to operate at 50 MHz with 10% gate activity. The table below com-

Technique	Wall clock		
	1 ms	1 s	1 hr
Simulator @ 5 Hz	33 min	23 days	230 yrs
Accelerator @ 500 Hz	33 s	9 hrs	3.8 yrs
Emulator @ 1 MHz	50 ms	50 s	50 hr

pares software simulation, software simulation aided by a hardware accelerator, and emulation (figures are for the Quickturn Enterprise System). If the 10% gate activity were increased the two simulation based methods would slow accordingly. With emulation performance does not degrade with gate activity.

The last few years have seen a number of prominent systems companies adopt emulation as part of their design strategy. Examples include: IBM, HP, Intel, and NCR, and they have used the technology for system validation and concurrent hardware/software development of ASICs, controllers, embedded computers, mainframe channels, and microprocessors. A notable examples was the use of a group of Quickturn systems to emulate the Intel Pentium. Claims of reducing the time to market by six months were reported in the trade press [2]. Intel engineers were able to boot DOS and execute third party applications before first silicon. Booting an operating system is an exacting test of the correctness of the hardware that is

difficult to match with simulation. Nevertheless, it is worth noting the notorious divide error of the Pentium was not detected.

2.2. Emulator Architecture

As we have noted, the enabling technology for emulators is the FPGA, in particular the RAM-based Xilinx family of parts. These FPGAs are organized as a two dimensional array of hundreds of programmable cells interconnected with a programmable grid of wires and switch boxes. The individual cell logic consists of combinational logic blocks and clocked flip-flops. A typical cell might have two 4 input 4 variable functions or one 9 variable function and 2 flip-flops. The interconnection pattern of the blocks and their functionality is controlled by a reconfiguration RAM. Typical density for a single FPGA part is several thousand gates, although, in most applications, only a fraction are usable.

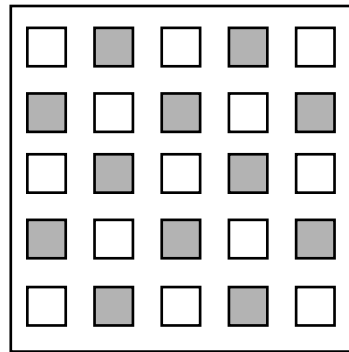
The first generation emulators assembled systems that were essentially boards full of FPGAs mounted in a chassis. The FPGAs on a board were hard-wired into a grid. Implementing the netlist of a target system is a two step process:

1. Partition the logic into components that can be realized within an FPGA
2. Placing and routing these components among the grid of FPGAs in the emulator

The first step is handled by logic synthesis software provided by the manufacturer of the FPGAs. The place and route software is provided by the manufacturer of the emulator.

Experience with first generation emulators showed two significant weaknesses: difficulty implementing memory, and difficulty producing efficient place and route algorithms for the grid of FPGAs. Implementing memory using logic gates in FPGAs is inefficient: single memory cells can use up a complete FPGA cell. This problem was solved by providing reconfigurable memory boards made principally from SRAM. These simplified the implementation of a variety of memory structures like caches and queues. This simple augmentation to the original idea of using only FPGAs allows a dramatic increase in the size of systems that can be emulated. To appreciate why this is so, it is sufficient to note that most processors are more than 50% memory structures. Furthermore, this fraction is increasing. The second problem of place and route among the FPGAs is due to pin constraints on the FPGA chips, which results in under utilization of the FPGAs — typically less than 10% of the gates. There

have been two solutions to this problem. The first, a novel scheme employed by Virtual Machine Works is to time multiplex the pins. The FPGA is programmed with a small finite state machine to control the multiplexing so that it appears to have a much larger number of “virtual” pins. The drawback is that systems constructed this way typically run slower due to the time multiplexing. The second, and more obvious solution, is to provide custom interconnects chip between the FPGAs. This arrangement is used by Quickturn in their second generation Enterprise system. Boards are constructed from FPGAs alternating with an interconnect chips (see diagram below). This significantly eases the inter-FPGA place and route problem, which in turn translates into shorter times to create an emulator for a design and better use of the available gates in the FPGAs.



The system we are presently using is an Enterprise emulation system. It has a usable capacity of 150K gates (6 boards). A fully configured system can support 270K gates. It is based on a Xilinx FPGA that contains about 1.5 k gates. Our system also contains 8M bytes of reconfigurable memory. To assist debugging, a logic analyzer with 1024 signal channels is built into the system. The emulator is attached to a workstation that has several large disks for storing netlists and that provides a network connection so the emulator can be used remotely. In the immediate future we will augment the existing system with fully configured Enterprise system, donated to us by the Ford Motor Co. The Enterprise is the work horse (see later) of our emulation activities. This will remain so for the time being; however, the technology has continued to develop. One of the driving forces has been the rapid increase in densities and speed of FPGAs, which are experiencing the exponential growth in performance characteristic of integrated circuit-based technology. Third generation systems have up to 3 million usable gates and run 2-3 times faster.

2.3. Emulators in the Design Process

The normal design cycle first develops a high-level (behavioral) description of a target system in an HDL (hardware description language — in our case Verilog). The Verilog description is usually translated to a structural form using some form of synthesis tool. This structural representation is then translated into circuit layout geometries. Emulation can take place once a structural design has been synthesized. The steps are as follows:

1. Use the fast high level behavioral Verilog code to generate benchmarks for validation
2. Synthesize structural Verilog code from the behavioral code (Synopsis is a common tool for this step)
3. Convert to netlist (such as EDIF format) for Quickturn input
4. If the leaf cells in the netlist are supported by the Quickturn library then goto step 6 else develop a library for the leaf cells using Quickturn library parts
5. Test library if necessary
6. Assign clock signal to the design
7. Select probe signals
8. Configure the netlist design (bottleneck, takes 10s of hours)
9. Import benchmarks (generated from step 1)
10. Emulate at speed (1 MHz measured)
11. Compare the emulation outcome with the correct outcome generated in step 1.

It is not necessary to simulate entire systems, in particular, co-simulation is supported in which it is possible to place a subsystem on the emulator and have the rest of the system simulate on the associated workstation. This is especially useful if peripherals such as disks and keyboards are part of the target system.

3. Experiences with Emulation

3.1. Two recent projects

Over the past year and a half our emulator has been used on a number of projects and classes. A notable recent experimental project has been to validate the major components in a 64 bit IEEE compliant FPU that we designed as part of a study of digital GaAs technologies. The design includes an iterative multiplier design, that was modeled after an early IBM design that retired 20 bits per iteration. This structure used 60K gates. It took about 2 days to translate

the netlist, compile it onto the emulator, and run billions of test cycles. The emulation clock speed was about 900KHz.

A second recent experiment was the emulation of the core of an Advanced Risc Machine's ARM 2. This processor, used in many low-power designs such as the Apple Newton, has a 32-bit data path, a 27 register file which includes the PC, and operates as a two stage pipeline. The emulation was the first part in a study to measure trade-offs in different low power technologies, specifically a conventional CMOS process — HP 0.5 um 3L — and a low power silicon-on-insulator process — Loral 0.8 um 3L. Both implementations contained about 60k devices. The emulator required about 20k equivalent gates (not including memory structures like the register file) and ran a suite of test programs at over 1 Mhz. This is a slow down of about 40:1. Again, mapping the netlist onto the emulator was a bottleneck, taking tens of hours. This has remained a source of inefficiency for most of our other emulation experiments too.

3.2. Current Projects

Emulation is being used in small graduate classes on testing. We are also developing an implementation of the popular DLX computer for use as a teaching aid in our undergraduate computer architecture class (the DLX machine is the case study detailed in the Hennessy and Patterson textbooks). Our goal is to create a thoroughly documented Verilog description of the DLX that can be synthesized in a number of ways including emulation.

Our studies of the intra-FPGA synthesis problem has lead to a new synthesis algorithms for FPGAs. It relies on integer programming techniques and is faster and more efficient than earlier synthesis methods. The first version of this work was reported in ICCAD [3].

In addition to the class projects, our emulator is currently being employed to validate a set of six chip designs that will form part of a project to study the use of advanced packaging technologies in the design of high-clock rate processors (ARPA). Our goal is to be PowerPC compatible with a clock speed in the range 800 Mhz to 1 GHz.

Perhaps the most interesting research to grow out of our experience with validation has been the initiation of a project to develop model based hardware design verification tech-

niques. This is a new DARPA funded project [4]. The objective of this project is to develop and evaluate a practical hardware design verification methodology and supporting CAD tools for high-performance microprocessors based on the synthesis of models for actual design errors, and the adaptation of test technology for physical faults to detect these design errors. Several projects currently under way at Michigan, including high-clock rate processors mentioned above, will serve as the experimental test-bed and demonstration systems for the proposed research. A comprehensive database will be constructed of the actual design errors encountered in these projects. The design errors will be identified through a combination of simulation and emulation. Systematic means of simulating (modeling) and detecting these errors will be developed and evaluated. A high-level (functional) methodology will be developed to construct simulation models and detection methods for design errors. Verification tests will be obtained via techniques borrowed from physical fault testing, which will exploit recent results in our research into high-level, model-based testing methods. If successful, this project will replace the use of emulation for validation. It will not, of course, replace emulation for concurrent hardware/software development, nor will it replace emulation for rapid prototyping.

4. Conclusion and Future Prospects For Emulation Technology

Our work in emulation had as one of its primary goals collaboration with a manufacturer of emulators in the development of a rapid prototyping capability. This together with validation and concurrent hardware/software development were also the early goals of the industry. To date, rapid prototyping has received the least development effort from the emulator manufacturers. There are a number of reasons among which is the difficulty of rapidly taking a behavioral description of a target system and mapping it onto an emulator. Recently, Zycad, a new entry into the emulation business (Zycad is best known as a manufacturer of hardware simulation accelerators) has begun investigate rapid prototyping. They have contacted us to exchange ideas.

Emulation will always have to compete with simulation on general purpose computers, which although slow, is flexible and does not require special equipment. In addition, techniques for improving simulation speeds are continually being developed. However, there are a number of developments that will continue to push the capability of FPGA-based emulation.

First is the growth noted earlier in FPGA densities. Second is the development of advanced packaging techniques like flip-chip mounting of bare die on PCBs and MCMs with area interconnect. The importance of these packaging developments are that they permit the construction of small high capacity emulators in which there is a high degree of interconnectivity between the FPGAs. This feature, as we have seen, is highly desirable if netlist mapping is to be done quickly. Finally, new research is examining the idea of “compiling directly to silicon” that also derives from FPGA technology. It can be expected to improve the mapping algorithms and attach a critical bottleneck in the use of emulators.

5. References

- 1 T. Mudge et al., *Rapid Prototyping and Evaluation of High Performance Computers*, NSF MIPS Grant.
- 2 *Business Week*, June 1st, 1992.
- 3 A. Chowdhary and J. Hayes, “Technology mapping for FPGA using integer programming,” *Proc. ICCAD-95*, San Jose, CA, Nov. 1995.
- 4 J. Hayes, T. Mudge, and R. Brown, *Hardware Design Verification for Microprocessors*, DARPA Grant.