

TOOLS FOR IN-CIRCUIT TESTING OF ON-LINE CONTENT PROCESSING HARDWARE

Jeff Mitchell

John Lockwood

Department of Computer Science and Engineering
Washington University in St. Louis
One Brookings Drive, Campus Box 1045, St. Louis, MO 63130
{jeff, lockwood}@ar1.wustl.edu

ABSTRACT

Tools have been developed that enable in-circuit testing of content processing hardware. The tools automate test and verification of new circuits using data from a predefined test-bench or with live traffic sent over the Internet. The tools integrate with existing, open-source, web-based groupware software. Content is processed with Field Programmable Gate Arrays (FPGAs) on an open hardware platform.

1. INTRODUCTION

One of the most challenging aspects of teaching a micro-electronics systems course involves testing of real hardware with live data. While it is instructive for students to experiment with hardware interactively, this activity can be frustrating if not conducted properly. If the students must learn inner workings of a platform in order to make use of it, they can become overwhelmed with detail. Ideally, a hardware platform should be readily accessible and easy to use. High-level tools that control and configure the hardware platform can significantly increase a student's productivity.

2. SIMULATION AND TESTING

For some types of circuits, it is possible to omit hardware testing altogether and rely on simulation to verify the correctness of a circuit design. Functional simulation verifies that the operation of a circuit meets a design specification. It does not, however, ensure that the circuit can be synthesized or that the circuit would meet necessary timing requirements. Back-annotated simulation ensures that a circuit can perform correctly after synthesis and place-and-route of the logic elements. For content processing applications, however, back-end simulation is slow and can require several hours or days to process large volumes of data.

Testing of circuits in actual hardware ensures that the circuit designs are functional, synthesizable, and operational with large data sets. Failing to test circuits in hardware can give a student a false sense of accomplishment. A circuit may simulate correctly for small data sets, but fail when deployed.

In this paper, we present a new suite of tools we developed to automate the testing of microelectronic systems over the Internet. Our system is implemented atop an open hardware platform and makes use of open-source software. We show how we use the tools to share a reconfigurable hardware platform among a group of students in a class. The tools arbitrate use of the physical hardware and provide a mechanism for students to manage their own design files. The tools have a generic interfaces that can be reused for different types of microelectronic courses.

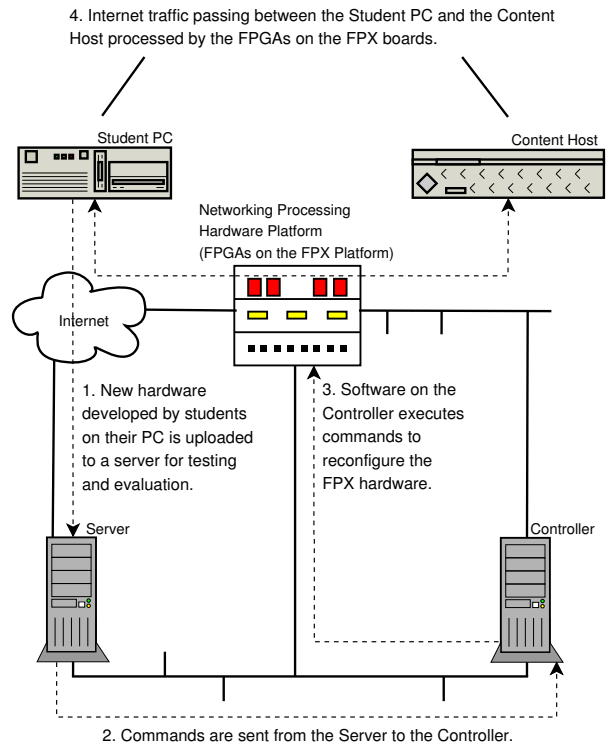


Figure 1. System Configuration

3. SYSTEM DESIGN

As shown in Figure 1, hardware circuits developed on a *Student PC* are configured into a *network processing hardware platform*. Network processing circuits are tested with data that is sent to and from a *content host*. Our tools make use of a push-pull mechanism between two computers: a *Server* and a *Controller*. The Server provides a user interface and runs a web server, a SQL database server, PHP scripts, and course management software. The Controller interfaces with the physical hardware, and executes commands to control and configure the underlying hardware.

4. HARDWARE CONTROL

A control paradigm was developed to communicate between the server and the controller. These scripts are called from high-level languages that include Perl and PHP. For functions that are interactive, daemons are used to interact between the hardware and software. Software daemons interactively transmit and receive commands over the network.



Figure 2. Photograph of the Network Processing Platform: A Washington University Gigabit Switch, three Field-Programmable Port Extender (FPX) platforms, and three Gigabit Ethernet network interfaces

5. USER INTERFACE

Our user interface is built atop scripts developed for the eGroupWare project [1]. eGroupWare is an open-source, free software suite that provides a web interface to group management software. It maintains user accounts, provides file transfer capabilities, provides discussion forums, and manages a shared calendar using PHP scripts and a SQL backend database. We used eGroupWare access control and user management capabilities to control access to the underlying hardware platform.

The web-based user interface is generated by the PHP scripts. Interaction with the system is performed by HTML forms, buttons, select boxes, and text input fields. These elements inherit a consistent look and feel from eGroupWare's templates. The user interface is simple, intuitive to use, and can easily be changed as needed.

The Server was implemented with a dual-processor AMD Athlon server running SUSE LINUX 9.1. The Controller was implemented with a single-processor AMD Athlon desktop running Red Hat Linux 9.0. The networking processing platform was implemented with the hardware described below. From the web interface, students selected parameters and retrieved data captured for detailed analysis. Captured data was displayed both on a web page and a copied into a folder within their eGroupWare file manager.

6. NETWORK PROCESSING PLATFORM

A Washington University Gigabit switch with three Field-programmable Port Extender (FPX) platforms and three Gigabit Ethernet line cards were used as the hardware platform [2]. A photograph of the system is shown in Figure 2. One FPX performed Transmission Control Protocol/Internet Protocol (TCP/IP) processing of data [3], a second FPX implemented the student's content processing circuit, and a third FPX captured traffic. Three Gigabit Ethernet line cards were used: one sent and received traffic to the student PC located anywhere on the Internet, a second Gigabit Ethernet line card sent and received data to and from the content host, and a third Gigabit Ethernet line card monitored traffic statistics.

Students designed their circuits on their student PCs, uploaded their bitfiles to the Server, and used the web interface to select which bitfile to program into hardware. At a lower level, this caused commands to be sent from the Server to the Controller to run certain scripts. These scripts, in turn,

sent commands to the hardware to reset and initialize the switch, program the bitfiles into hardware, set up the data flows, and route Internet traffic flows through their hardware circuit.

After the three FPX boards were programmed, students sent traffic through their PC and accessed data on the content host. All traffic that passed between the student PC and the content host was multicast to the TCP Processor, which in turn sent data to the student's circuit. In this manner, students were able to run live Internet traffic through their circuits. Statistics about the operation of the circuit as well as the content of input/output packets were stored in memory using the third FPX platform.

7. COURSEWARE TESTING

This system was first successfully used in the *Reconfigurable System on Chip Design* course taught during the Fall 2004 semester at Washington University in St. Louis [4]. Students were assigned two machine problems and a final project that made use of the network processing platform. In their first machine problem, students used the TCP/IP tracking hardware to collect Internet traffic statistics. In the second machine problem, students designed and implemented a circuit that performed semantic processing of content from a web server. In particular, they implemented a circuit that counted the frequency of words that appeared in a set of poems stored on a content server. For their final projects, students implemented circuits that detected email spam, planned collision-free motion for a robot, processed SNORT rules, automatically identified the use of foreign languages, and provided content delivery functions.

The benefits of this design to the students were significant. Students were productive and able to focus their time on the design of their circuits. The tools allowed a single hardware platform to be efficiently shared by all 14 members in the class.

8. CONCLUSION

Our tools enable in-circuit testing of content processing hardware. From a web browser, hardware is reconfigured. Content is processed in student-generated FPGA circuits as it flows through the Internet in TCP/IP traffic flows. The system processes volumes of streaming data at a rate of 1 billion bits per second. The tools interface with the eGroupWare software and the FPX hardware platform, and can be reused to provide on-line testing for a variety of microelectronic system design courses.

REFERENCES

- [1] The eGroupWare project, <http://www.egroupware.org>
- [2] J. W. Lockwood, "Evolvable Internet hardware platforms," in *The Third NASA/DoD Workshop on Evolvable Hardware (EH'2001)*, pp. 271–279, July 2001.
- [3] D. Schuehler and J. W. Lockwood, "A modular system for FPGA-based TCP flow processing in high-speed networks," in *14th International Conference on Field Programmable Logic and Applications (FPL)*, (Antwerp, Belgium), pp. 301–310, Aug. 2004.
- [4] J. W. Lockwood, Washington University, Department of Computer Science and Engineering. "Reconfigurable System on Chip Design" (CSE 566), Fall 2004, <http://arl.wustl.edu/~lockwood/class/cse566-f04/>