PLATFORM AND METHODOLOGY FOR TEACHING DESIGN OF HARDWARE MODULES IN INTERNET ROUTERS AND FIREWALLS

John W Lockwood [*]

Department of Computer Science
Applied Reseach Laboratory
Washington University
Saint Louis, MO 63130
lockwood@arl.wustl.edu

ABSTRACT

An instructional platform has been developed that allows rapid prototype of network packet processing functions in hardware. This platform, called the Field Programmable Port Extender (FPX), enables engineering students to rapidly prototype and implement components for use in an Internet router or firewall.

Customized circuits allow networking equipment to increase the throughout and enhance functionality of packet processing operations. On the FPX, custom circuits are implemented as hardware modules. An infrastructure circuit on the FPX interconnect multiple modules and provides a common interface to shared resources. All logic on the FPX is implemented with Field Programmable Gate Arrays (FPGAs).

A teaching methodology has been developed which minimizes the learning curve for hardware engineering students that want to develop network modules but are relatively unfamiliar with Asynchronous Transfer Mode (ATM) and/or the Internet Protocol (IP) suite. Library functions have been developed to implement low-level details of the networking protocols. The use of these libraries and the infrastructure logic on the FPX allows the students to focus their effort on the design of their own module and to leverage the work of others.

1. INTRODUCTION

Internet routers and firewalls increasingly use customized circuits to perform performance-critical packet processing functions in hardware. Routers that use per-flow packet queuing hardware can, for example, provide Quality of Service for real-time traffic flows. Routers that have dedicated hardware to perform route lookups can more quickly determine the next-hop route of a packet that would otherwise be possible in software [1].

As more features are implemented in hardware, more microelectronic system engineering students will pursue careers developing new circuits to enhance the Internet. An instructional platform has been developed at Washington University which enables students to easily design such packet processing functions in hardware and to experiment with their systems in a network.

2. THE FPX PLATFORM

The Field Programmable Port Extender (FPX) fits between standard interfaces of a gigabit switch fabric and a line-card, as shown in Figure 1. This configuration allows the FPX to access and modify both the header and payload of all packets that enter and depart a network switch.
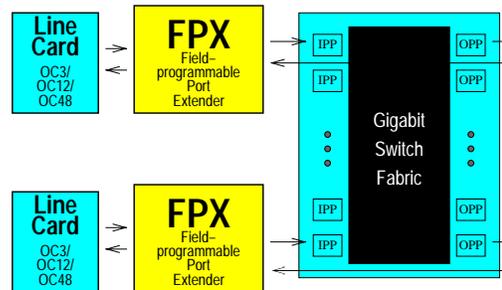
Figure 1. FPX Module at Network Edge

Individual functions on the FPX are implemented as network modules. A standard interface has been developed that defines the interface between the infrastructure and an FPX module. A graphical view of the FPX module entity is shown in Figure 2.

On the data interface, packets arrive and depart in frames that have been segmented into a sequence of Asynchronous Transfer Mode cells. A Start of Cell (SOC) indicates when cell arrives. A Transmit Cell Available (TCA) allows a module to backpressure traffic when there is congestion.

Modules have access to both external SRAM and SDRAM memories. The SRAM is typically used for applications that need to implement table lookup operations (like the Fast IP lookup algorithm), while the SDRAM interface is typically used for applications like packet queuing that transfer bursts of data and can tolerate a higher memory latency.
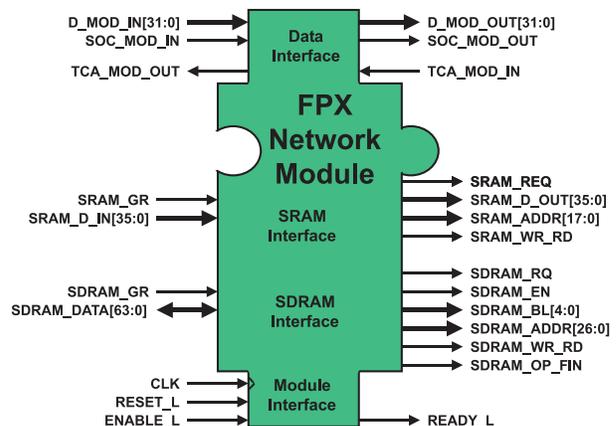
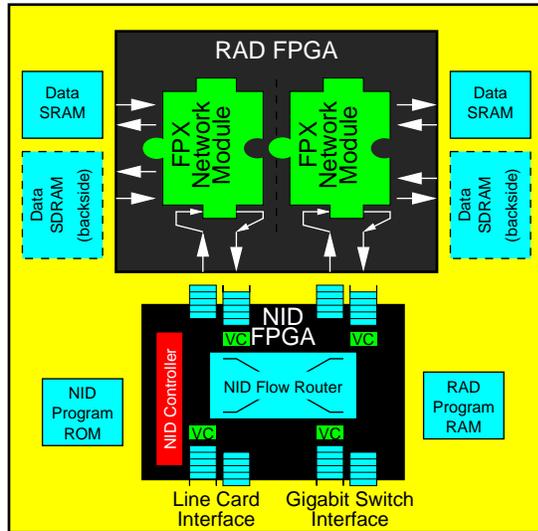

Figure 2. Network Module Entity

Figure 3. FPX Configuration

The FPX implements all logic using two FPGA devices: the Network Interface Device (NID) and the Reprogrammable Application Device (RAD). The interconnection of the RAD and NID to the network and memory components is shown in Figure 3.

The RAD contains the modules that implement the student's customized packet processing functionality. Each module on the RAD connects to one SRAM and to one SDRAM. In total, both modules have full control over four independent banks of memory.

The NID performs per-flow switching on traffic that enters and leaves each of the four interfaces of the FPX. Each port includes a Virtual Circuit (VC) lookup table, which is read to determine the next hop for each cell. The *NID flow router* selectively forwards traffic streams between the line card interface, the gigabit switch interface, and the module interfaces.

Modules on the FPX can be reprogrammed by sending the compiled bitfiles for the RAD modules over the network. The *NID controller* listens on a well-known network address and caches reconfiguration bitfiles in the *RAD Program RAM* [2].

A photograph of the FPX module is shown in Figure 4. The RAD is implemented with a Xilinx Virtex XCV1000E FPGA device and operates at 100 MHz. The NID is implemented with a smaller XCV600E FPGA. The NID operates on multiple clock domains, which are determined by the speed of the fiber optic line card and the speed of the switching fabric. The Networking interfaces on the FPX were optimized to operate at rates as high as SONET OC48 (2.4 Gigabits/second).

3. TEACHING METHODOLOGY

A teaching methodology has been developed which minimizes the learning curve for hardware engineering students that want to develop networking hardware but are relatively unfamiliar with ATM or Internet protocols. Students are given VHDL code that defines the entity of an FPX network module. They are also given access to libraries that perform common functions that include packet framing, control cell processing, and checksum computation.

Students learn to build networking components though

Figure 4. Photo of the FPX

experience. They first write and compile a hardware design to fit within the entity of a module. Next, they verify the operation of the circuit in a testbench, where the inputs and outputs are in the format of networking packets. Next, they combine their design with the infrastructure circuits of the FPX and synthesize their module into the FPGA device which implements the RAD.

3.1. Course

The design and prototype of FPX network modules is the focus of a new course at Washington University called "Acceleration of Algorithms in Reconfigurable Hardware". At the start of the semester, students compile and simulate a networking module called "Hello World". Though this, they learn how an networking module performs content searching and payload modification. Next, students implement machine problems for per-flow data queuing and data compression. At the end of the semester, teams implement a new networking module of their own design.

3.2. Workshop

In January of 2001, an NSF-sponsored workshop was held at Washington University for visitors from other universities interested in using the FPX for their own teaching or research. The implementation of a sample FPX module, including the VHDL source code and I/O pin mappings of an FPX networking module, were made given to the participants of the workshop and made available for download from the web [3]. These examples compile, simulate, and synthesize with standard simulation and synthesis tools, including ModelSim, Synplicity, and Xilinx Foundation tools [3].

REFERENCES

[1] J. W. Lockwood, J. S. Turner, and D. E. Taylor, "Field programmable port extender (FPX) for distributed routing and queuing," in *FPGA'2000*, (Monterey, CA), pp. 137–144, Feb. 2000.

[2] J. W. Lockwood, N. Naufel, J. S. Turner, and D. E. Taylor, "Reprogrammable Network Packet Processing on the Field Programmable Port Extender (FPX)," in *FPGA'2001*, (Monterey, CA), Feb. 2001.

[3] J. W. Lockwood, D. E. Taylor, T. Sproull, S. Dharmapurikar, F. Braun. "Field Programmable Port Extender Homepage" `http://www.arl.wustl.edu/projects/fpx/`