GUI Design Aid for Customizing Virtual Networks with FPGA Reconfiguration
To Chong, Deepak Unnikrishnan, Prof. Russel Tessier
University of Massachusetts Amherst

Abstract

The continuous growth of the internet demands significant experimentation at the network core consisting of routers and switches. Network virtualization allows researchers to conduct experiments in network core by constructing many logical networks using physical network infrastructure. Traditional implementations of network virtualization use software techniques that suffer from poor performance. Field Programmable Gate Arrays (FPGAs) offer a flexible, high performance alternative to software techniques by virtue of their adaptability and parallelism. A novel reconfigurable network virtualization platform using FPGAs is being developed at the Reconfigurable Computing Group, UMass Amherst. This platform implements multiple virtual routers on a Xilinx FPGA chip. Although this system offers high performance, usability remains a primary challenge for wide scale deployment.

Overview

• Traditionally, network virtualization is the process of combining hardware and software network resources and network functionality into a single, software-based administrative entity, a virtual network.

• Our network virtualization system is implemented on a NetFPGA board as shown in Fig. 1. This board consists of a Xilinx Virtex II FPGA and 4 one-gigabit Ethernet ports (Fig. 2). Users can program the FPGA from a host PC that runs Linux operating system. Users can embed up to 32 virtual routers depending on the size of the FPGA.

Progress and Results

• I developed an algorithm to automatically generate Verilog descriptions of virtual routers based on user preferences and implemented the algorithm into a GUI as shown in Fig. 3.

• The generated hardware descriptions can then be synthesized using Xilinx tools to create a programmable bit stream for the FPGA.

• I successfully tested a design with four virtual routers for the user data path (UDP) (the path that specifies the logic between input and output signal) module as shown on the schematic in Fig. 4 and waveform synthesis in Fig. 5.

Future Work

• Design custom Verilog modules that can change individual parameters such as Time To Live (TTL) value and IP header (MAC address/ IP address) inside the virtual router.

• Standardize the design with only input and output ports to facilitate the selection process for the users so that only minimum understanding of the virtual router is sufficient to operate the program.

• Implement Click router modular language into the current GUI design to generate plug-and-play fine grained router features

Research Objectives

• Design a tool that can automatically generate hardware description language for the FPGA-based virtualization system from a simple graphical user interface (GUI).

• Allow users to parameterize fine-grained features of individual virtual routers.

• Develop a dataflow style programming environment that allows users to create custom virtual routers on the fly.

Design a tool that can automatically generate hardware description language for the FPGA-based virtualization system from a simple graphical user interface (GUI).

Selected References


Special Thanks To Professor Tessier at University of Massachusetts Amherst for supporting this research and Deepak Unnikrishnan, Ph.D student at University of Massachusetts Amherst for assisting me in completing my REU research project.