International Journal of Computer Networks & Communications (IJCNC) Vol.3, No.5, Sep 2011

PERFORMANCE COMPARISON BETWEEN THE CLICK MODULAR Router and the NetFPGA Router

Leonardo Linguaglossa[1], Alfio Lombardo[1], Diego Reforgiato[1], Giovanni Schembra[1]

[1]Department of Computer Science and Telecommunication Engineering,
V. le A. Doria, 6, 95125,
University of Catania
Catania, Italy

{leonardo,alfio.lombardo,diegoref,schembra}@diit.unict.it

ABSTRACT

[1]It is possible to forward minimum-sized packets at rates of hundreds of Mbps using commodity hardware and Linux. We had a preference for the Click Modular Router platform due its flexibility and the fact that it claimed to have equal or higher performance than native forwarding if used with its polling drivers. Moreover, the NetFPGA is an open networking platform accelerator that enables researchers and instructors to build working prototypes of high-speed, hardware-accelerated networking systems. NetFPGA reference designs comprised in the system include an IPv4 router, an Ethernet switch, a four-port NIC, and SCONE (Software Component of NetFPGA). Researchers have used the platform to build advanced network flow processing systems. We have followed the RFC1242 - Benchmarking Terminology for Network Interconnection Devices - and the RFC2544 - Benchmarking Methodology for Network Interconnection Devices - in order to define the specific set of tests to use to describe the performance characteristics of the two routers. We have also shown a test comparison between the NetFPGA and the Click router about a file transfer using the FTP and the HTTP protocol.Overall, the NetFPGA router performance outperforms the Click router performance.

KEYWORDS

NetFPGA, Click Modular Router, RFC1242, RFC2544

1. INTRODUCTION

The innovation in computer networks used in our everyday life has become notably crucial. Usually, network devices as IP routers, bridges, hub and switches are compacted and closed platforms, which are not possible to change or enhance. Their functionalities are limited and restricted by vendors who are often hostile to allow researchers and programmers to modify and extend their products. This implies a substantial decrease in the rate of innovation and improvement.

[1]This work is partially supported by the Italian MIUR PRIN 2007 project "SORPASSO". Moreover, the work leading to this invention has benefited from a fellowship of the Seventh Framework Programme of the European Community [7aPQ/2007-2013] regarding the Grant Agreement n. PIRG03-GA-2008-231021.

DOI : 10.5121/ijcnc.2011.3503

Moreover, routers are increasingly expected to do more than route packets. Boundary routers, which lie on the borders between organizations, must often prioritize traffic, translate network addresses, tunnel and filter packets, and act as firewalls, among other things. Furthermore, fundamental properties like packet dropping policies are still under active research [10], [19], [1], and initiatives like Differentiated Services [8] bring the need for flexibility close to the core of the Internet. Unfortunately, most routers have closed, static, and inflexible designs. Network administrators may be able to turn router functions on or off, but they cannot easily specify or even identify the interactionsof different functions. Furthermore, it is difficultfor network administrators and third party software vendors to extend a router with new functions. Extensions require access to software interfaces in the forwarding path, but these often do not exist, do not exist at the right point, or are not published.Click, a flexible, modular software architecture for creating routers has been presented in [15]. Click was developed at the MIT (Massachusetts Institute of Technology) with the aim of increasing the flexibility and the extensibility of network routers.Click routers are built from fine-grained components; this supports fine-grained extensions throughout the forwarding path. The components are packet-processing modules called elements. The basic element interface is narrow, consisting mostly of functions for initialization and packet handoff, but elements can extend it to support other functions (such as reporting queue lengths). To build a router configuration, the user chooses a collection of elements and connects them into a directed graph. To this purpose, a configuration script has to be written in the Click language, and to create the right connection between elements, two different packet transfer mechanisms are supported by Click, the *push* and the *pull* one. In a *push* connection the packet transfer is initiated by the source element, which passes it to the downstream element. The *pull* connection, on the other hand, works in the dual way: in this case it is the downstream element that initiates the packet transfer, asking the upstream element to send it a packet. Furthermore, in order to improve the performance provided by software-based routers, high-performance processing infrastructures, which use multiprocessor systems, can be used. The graph's edges, which are called connections, represent possible paths for packet handoff. To extend a configuration, the user can write new elements or compose existing elements in new ways, much as UNIX allows one to build complex applications directly or by composing simpler ones using pipes.Thanks to its flexibility, Click Modular Router is extensively used today.

The main advantages coming from the use of software routers, which are also capturing the interest of the commercial telecommunications area in the last years, can be summarized as follows:

1) a greater flexibility, that is, the ability to adapt the network to the continuous Internet evolution;

2) a longer life-time for the equipments;

3) an easier and faster deployment of new services, which allows network providers and equipment vendors to quickly react to the user demand.

On the other hand, a critical drawback of the software processing can be a lower performance as compared to traditional hardware solutions, developed, customized and optimized for faster processing of specific protocols. Of course, an important challenge in software packet processing is to achieve performance comparable to the traditional hardware solutions.
Luckily, hardware routers today do exist and one of them is represented by the NetFPGA project.

The NetFPGA platform [6] allows everyone to prototype and develop multi-Gigabit networking applications. It is an open platform and user community developed to enable researchers to build high-speed, hardware-accelerated networking systems. The platform is used by instructors to show how to build line rate Ethernet switches and Internet Protocol (IP) routers. The open-source NetFPGA distribution consists of gateware, hardware and software. As far as the hardware is concerned, it consists of a PCI card that has an FPGA, memory (SRAM and DRAM), and four 1-GigE Ethernet ports. Source code and scripts are provided to build reference designs, enhance a design, or create new applications using supplied libraries. Hardware description source code (gateware) and software source code are freely available online.The NetFPGA platform not only consists of the NetFPGA board, but also the development environment and scripts that allow for rapid prototyping and development of hardware projects. The development environment is available from the NetFPGA [6]. Recently, a new NetFPGA board equipped with four 10-GigE Ethernet interfaces has been released.

Reference designs comprised in the system include an IPv4 router, an Ethernetswitch, a four-port NIC, and SCONE (Software Component of NetFPGA). Researchers have used the platform to build advanced network flow processing systems. A single NetFPGA board can route packets over any number of subnets, and multiple NetFPGA boards can be installed in the same PC. In addition, there are several user-contributed projects available such as the netflow probe, OpenFlow switch [21], the Packet Generator, the RCP router, the URL extraction [22] and the traffic monitor [23].

This paper shows the result of the specific set of tests (as described in the document RFC1242 [12] and RFC2544 [13]) measured to report the performance characteristics of the Click Modular Router compared to the NetFPGA router.

This paper is structured as follows: Section 2 introducesthe Click Modular Router architecture; Section 3 describesthe NetFPGA platform; Section 4 shows the set up we haveprepared in order to run the tests. The experimentationand the obtained results are shown in Section 5 whereasSection 6 ends the paper with the conclusions and potentialfuture works.

2. CLICK MODULAR ROUTER ARCHITECTURE

The Click Modular Router is a software architecture which lets people create their own personal router in a flexible and easy way. The Click Modular Router project was born to give users the possibility to change the behaviour of their device without influence on the device provided by vendors, because everyone can develop his router and so it can be able to perform the task they configured it for. Click routers [16] consist of components called elements plugged together into configurations. Elements process packets invarious ways, creating them, modifying them, classifying them into different paths, and so forth. Packets flow from element to element along the edges of a configuration graph. A Click configuration file is similar to the declaration of a block diagram, and for every element, assumed to be as a block, are specified a certain number of input ports and a certain number of output ports. Usually the number of the input and the output ports can be different. Example elements include"From- Device(eth1)", which reads and emits packets from network device eth1, and "Discard", which drops any packets it receives. Here is a simple router configuration file using those elements:

FromDevice(eth1) → Discard;

This file, like any Click configuration, uses a simple declarativelanguage. Compiler-like optimization and analysis passescan transform Click-language files to improve performance [14].Connections between elements can use either push or pullprocessing. In push processing, packets are actively pushedforward through the graph. In pull processing, packet requestsmove backwards through the graph. Pull processingmodels packet transfer as an “upcall”: downstream elementscall upwards to retrieve a packet. The combination of pushand pull can model complex control flow patterns, includingexplicit queues. A Click driver can run inside the Linuxkernel, or at user level on any Unix-like OS. Most elementsource files can be compiled for any driver and moreover, the open source philosophy allows people to modify the source existing file, written in C++, to create their own elements, or to create a new source code to be compiled.This is the simplest way to have new elements with personal functions. Click kernelconfigurations can run at or close to the limits of conventionalPC buses [9],[16],[14]. For example, an optimizedClick IP router can forward 740,000 minimum-size packetsa second over Gigabit Ethernet on a 1.6 GHz Athlon MPwith 64-bit/66 MHz PCI [14]. Users may be in contact with a running configuration of the Click Modular Router either beginning a telnet session, if it is executed at user-level, or with the system/proc file, if Click is run at kernel-level. These two methods to interface users with Click use a kind of access point provided by some elements of the Click configuration, also known as handlers. Each handler may support a read or write operation.

The Fig. 1shows how a simple Click configuration file is presented. The reader may notice that some elements may not have input port (this is considered to be a source), and others may not have output port (this one, instead, is considered to be a destination).

Handlers allow users to perform every operation; in such a way, the user, for instance,may see the number of packets in a queue or the rate at which a source is sending packets.

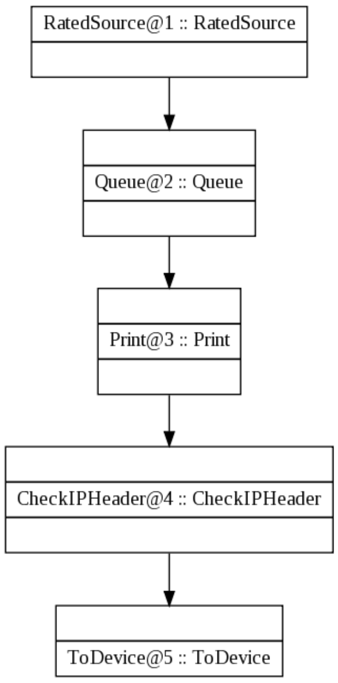

RatedSource@1 :: RatedSource
Queue@2 :: Queue
Print@3 :: Print
CheckIPHeader@4 :: CheckIPHeader
ToDevice@5 :: ToDevice

Figure 1. Elements of a Click Configuration developed using the Click-Viz,a program released together with the Click router which transforms the configuration file into a block diagram.

3. NetFPGA Platform

The NetFPGA is an accelerated network hardware that augments the function of a standard computer. It consists of three parts: hardware, gateware, and software. The development board itself is a PCI card that can be installed in any PC with an available full-length slot. In more detail, the hardware of the board has the following core components:

- Xilinx Virtex-II Pro 50
- 4x1 Gbps Ethernet ports using a soft MAC core
- Two parallel banks of 18 MBit Zero-bus turnaround (ZBT) SRAM
- 64 MBytes DDR DRAM

The FPGA directly handles all data-path switching, routing, and processing of Ethernet and Internet packets, leaving software to handle only control-path functions [17]. Hosted on the board are a user-programmable FPGA (with two PowerPC processors), SRAM, DRAM, and four 1Gbps Ethernet ports. Software and gateware (Verilog HDL source code) are available for download under an open source license from the NetFPGA website [6]. This allows jump starting prototypes and quickly building on existing designs such as an IPV4 router or a NIC. The gateware is designed to be modular and easily extensible.

A new NetFPGA board has been announced: it has 4x10GigE SFP+ interfaces, a PCI Express interface to the host, (Gen2 x8 channels), and a Xilinx Virtex-5 TX240T FPGA. The board has SRAM and DRAM (27 Mbytes QDRII SRAM, 288 Mbytes RLDRAM-II) and a high bandwidth expansion connector for daughter cards. It is goal of the authors of this paper to get the new NetFPGA board and try new and existing projects on it.

Designs are implemented as modular stages connected together in a pipeline, allowing the addition of new stages with relatively small effort [18]. The pipeline is depicted in Figure 2.

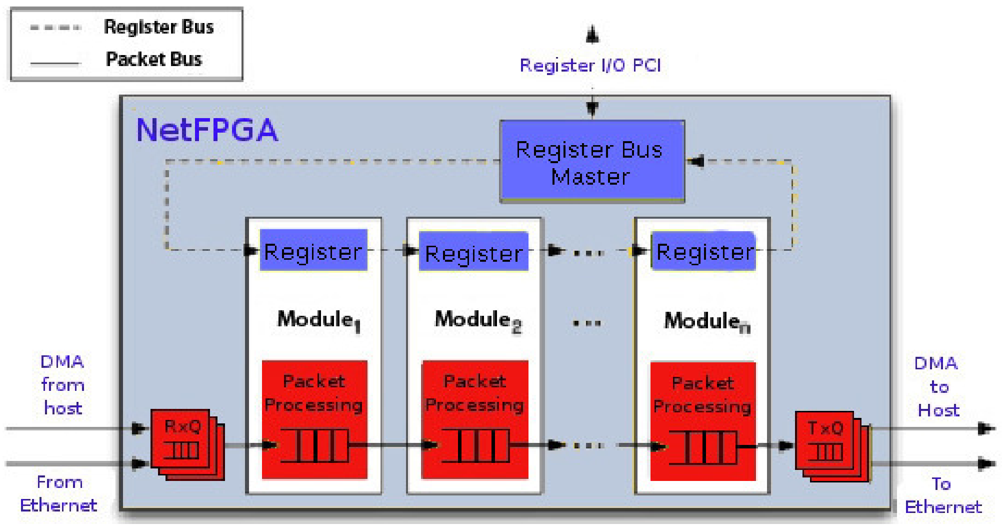


Figure 2. Modular NetFPGA pipeline structure. The high-bandwidth packet bus (in blue) issued for packet processing while the register bus (in red) is used to carry control and status information between software modules and the hardware.

The platform can be used to show how to build Ethernet switches, Internet Prototcol (IP) routers using hardware rather than software, to implement precise network measurement systems, and to design hardware-accelerated network processing systems. The platform can be used by researchers to prototype advanced services for next-generation networks.

Accent Technology [3] offers pre-assembled NetFPGA computersystems as approved by Stanford University. Thesepre-built and completely tested Linux-based computers areavailable in a compact desktop cube or standard 1U rackmountableserver configuration. In the researcher laboratories,the NetFPGA is usually installed inside a desktopPC so researchers can access the hardware [11, 20]. Severalprojects have already been developed in the NetFPGA (seethe NetFPGA project page [5]).

3.1 Gateware and Software

One of the most appealing features of the NetFPGA platform is the availability of the open-source Verilog gateware, and the related software. Its design is modular and allows users to implement new modules and connect them in new configurations. The host PC via the PCI bus performs the programming and administration of the development board. This allows users to remotely develop and deploy designs since physical access to the board is not required. There are three main components of the NetFPGA architecture:

1. Kernel module. It is used to communicate to the NetFPGA through a register interface implemented using the shared memory and through the PCI bus of a Linux based PC. The DMA is used by all the reference systems to receive and send network packets from the card. Software, running on the host PC, writes control data and reads statistic counters using several registers.

2. Common utilities used to communicate with the card. Among the utilities there are a bitfile download utility, and programs to read and write on to registers (ie. Regdump which dumps the contents of the registers, a java gui that allows the user to change entries in the routing table and ARP cache as well as the router's MAC and IP addresses, and a standalone command line interpreter (CLI) which allows the user to change routing table entries, ARP cache entries and other settings).

3. Reference pipeline. It is described in Section 3.2

3.2 Reference Pipeline

The reference pipeline consists of the user datapath, eight receive queues and eight transmit queues. Its architecture is depicted in Figure 3. Both the queues are classified into two types: MAC and CPU. The MAC queues are assigned to one of the four NetFPGA ports whereas one CPU queue is associated with each of the MAC queues. Developers can create their own modules.. They can add and connect all the available modules to the User Data Path. There are a few modules that are present in almost all the NetFPGA projects: they are the Input Arbiter and the Output Queues modules. In particular, the Input Arbiter supplies a wide 64-bit packet pipeline to the eight input queues in a round robin fashion. Details of the implementation are contained within the NetFPGA Verilog library (which contains the source code of all the modules). All the modules may be inserted into the pipeline through the register system. There is a register interface which allows software programs running on the host PC to send data and receive data from the hardware modules. Registers and counters are assigned names that are common to the hardware design and C or Perl software that runs on the host PC.

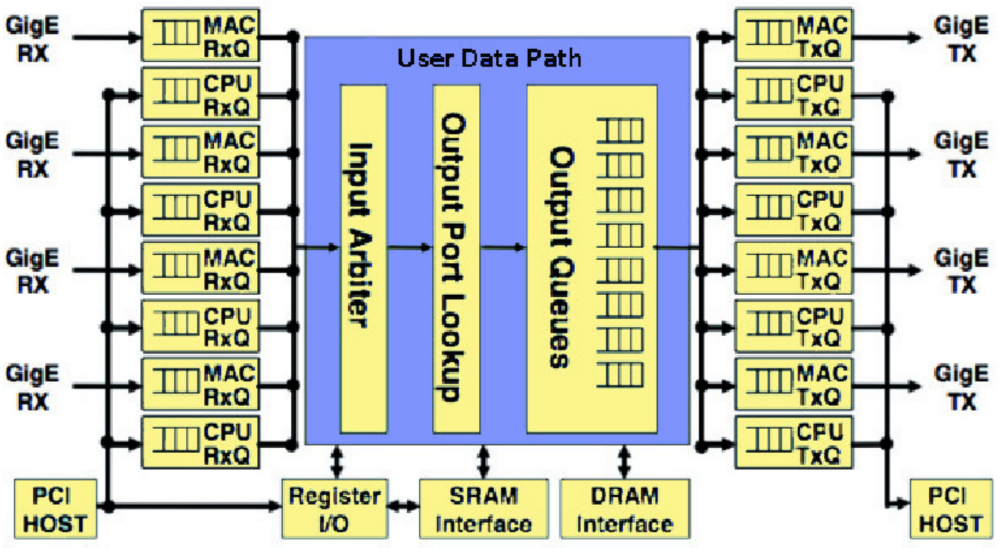


Figure 3. NetFPGA Reference Pipeline

4. COMPARISON ANALYSIS SETUP

This section shows the settings for the evaluation of Click [16]and NetFPGA [6] performance for IP routing. The PCs wehave used are two AMD X2 6000 with dual core processorrunning at 3.0 GHz, 2 GB DDR2 RAM, an Intel Pro/1000 PT Dual-port NIC, and an ASUS M3N78-VM motherboardwith an on-board gigabit NIC. The PCs are equipped witha NetFPGA [6] board. The operating system installed isthe 32-bit version of CentOS 5.2 using kernel 2.6.24.7. Apatch [2] has been applied to the kernel in order to have theClick Modular Router working on the same machine. Moreover, the whole NetFPGA package hasbeen installed: it includes the IPv4 router, a four-port NIC,an IPv4 Router with Output Queues Monitoring System,the PW-OSPF software that interacts with the IPv4 Router(SCONE), and the Router Kit which is a daemon that reflectsthe routing table and ARP cache from the Linux hostto the IPv4 router on NetFPGA. While Click’s main functionalityis to act as a router, it can also be used to generateand count packets. We set up one of the machine with avery simple fast TCP generator configuration and used theother PC to receive and count packets. To evaluate theClick’s performance we run the Click packet generator fromthe first PC and used the Click router listening on the eth0interface on the other PC; to evaluate the NetFPGA’s performancewe changed the behavior of the second PC thatused the NetFPGA router SCONE listening on the nf2c0interface.

5. EVALUATION

In this section we will report all the comparative tests we have performed following the guidelines of the documents RFC1242 [12] and RFC2544 [13].

5.1Constant Load

Along this test fixed length frames have been sent at a fixed interval time. Although it is rare, to say the least, to encounter a steady state load on a network device in the real world, measurement of steady state performance may be useful in evaluating competing devices. The frame size have been set to the following bytes size: {64, 128, 256, 512, 1024, 1280} and all the simulations were run without any anomalies. Actually, the constant load test was run for first in order to verify the SCONE Router and the Click Modular Router. The router configuration in the Packet Generator Click file was set to send a certain number of packets at a fixed rate for two minutes. First, packets have been sent to the Click router and then to the SCONE Router. Then, we manually changed the rate in the configuration file and the tests have been repeated for other values of the rate. The maximum rate at which packetswere sent was just below the threshold of the throughput as the task of this test was not tofind the maximum value of the rate, but rather to verify the behaviour of the two DUTs (Click Modular Router and the NetFPGA router)in a stable situation.

5.2Throughput

The throughput is the maximum rate at which none of the offered frames are dropped by the device. Figure 4 shows the throughput obtained for the Click router and the NetFPGA router. The x-axis shows the frame size for the measurements we have taken. The y-axis shows the amount of received frames in bytes per second. This test was performed in a “incremental” way: we have repeated the same steps of the previous test, but this time the rate in the Packet Generator configuration file was increased every time until a loss packet occurred. The rate when the first loss occurs has been considered the limit rate for the considered device.

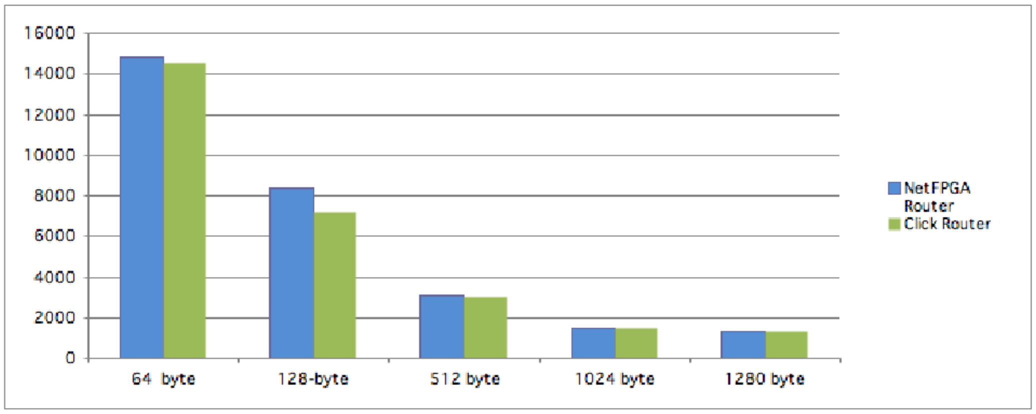


Figure 4. Throughput for Click router and NetFPGA router.

5.3Frame Loss Rate

This test measures the percentage of frames that should have been forwarded by the Click and the NetFPGA router under steady state (constant) load that were not forwarded due to lack of resources. This can be a useful indication of how a device would perform under pathological network conditions such as broadcast storms. We have used several frame size for this test and, for each frame size we repeated the test over 1000 runs. Fig. 5 shows the frame loss percentage for the Click router and the NetFPGA router averaged over all the runs.The x-axis shows the frame size for the measurements we have taken. The y-axis shows the frame loss percentage.

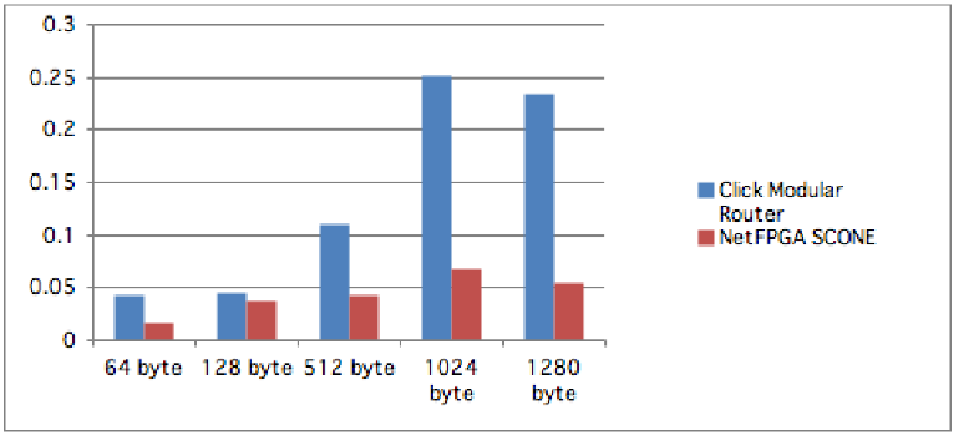


Figure 5. Frame loss rate for Click router and NetFPGA router for different values of the frame size

5.4Overloaded Behavior

Devices in an overloaded state will lose frames. The device might lose frames that contain routing or configuration information. An overloaded state is assumed when there is anyframe loss. We have set both the Click Modular Router andthe NetFPGA router to an overloaded state by sending aburst of data packets. Figure6 shows the different behaviour of the two devices. 85674 bytes are sent between the interval time [t_1, t_2]; the Click receives only 45765 bytes whereas the NetFPGA receives 82464 bytes. The reader notices thatthe NetFPGA router is able to receive more data after the time $t_2$because after filling the User Data Path, extra packets are first transmitted to the PCI bus for a certain time. As soon as the PCI bus cannot handle more data, these start to be lost. Moreover, the Click router stops receiving packets as long as the rate becomes higher than its maximum rate; as such, the received packets will be then a constant number.

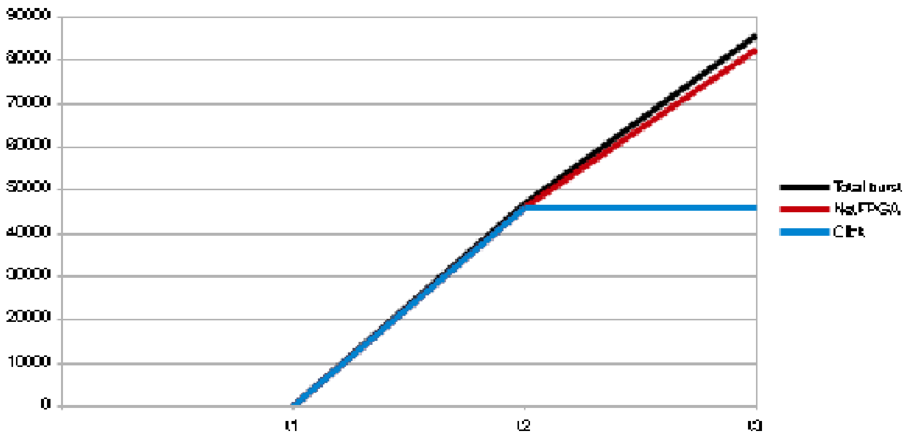


Figure 6. Overloaded behavior for the Click Modular Router and the NetFPGA router.

5.5Restart Behavior

This test measures the re-initialization of system causing data loss. During a period of time after a power up or reset, network devices do not accept and forward frames. The duration of this period of unavailability can be useful in evaluating devices. In addition, some network devices require some form of reset when specific setup variables are modified. If the reset period were long it might discourage network managers from modifying these variables on production networks. Table 1 shows the average time (in milliseconds) of the restart behavior over 1000 runs between the Click and NetFPGA routers. The average time is higher for Click as each time the Click reboots, it has to be loaded again into the operating system kernel module.

Table 1. Restart behavior average time for the Click router and NetFPGA router over 1000 runs (in milliseconds).

Click Router	NetFPGA Router
1990 ms	880 ms

5.6Single Frame Behaviour

A data "stream" consisting of a single frame can require the Click router or the NetFPGA router to do a lot of processing. They will often take much more time to process a singleframe presented in isolation than it would if the same framewere part of a steady stream. There is a worry that somedevices would even discard a single frame as part of the cache setup procedure under the assumption that the frame is only the first of many. Only one frame is sent to the two routers for the purpose of this test. We have used several frame sizes and, for each frame size we repeated the test over 1000 runs. Fig. 7 shows the received time for the Click router and the NetFPGA router averaged over all the runs.The x-axis shows the frame size for the measurements we have taken.

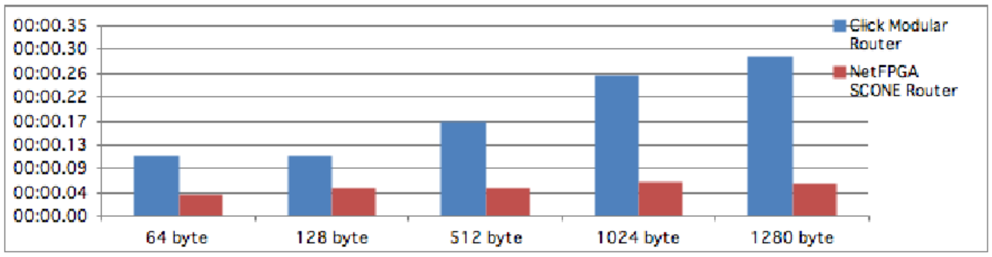


Figure 7. Single frame behavior for Click router and NetFPGA router for different values of the frame size.

5.7FTP/HTTP File Transfer

In these last two tests we want to show how fast a file transfer is using the NetFPGA router with respect to the Click router. The reader notices that these tests are not referenced into the RFC1242 [12] or RFC2544 [13] but we decided to include them to show the Click and

NetFPGA routers behaviour in real scenarios. The topology we have used is shown in Fig. 8 where PC2 sends data to PC1 using two different protocols.

Fig. 9 shows a file transfer using the FTP protocol whereas Fig. 10 shows the same file transfer using the HTTP protocol. The file size was 1.8 GB. The red line in Fig. 9 is shorter than the blue line as, using the NetFPGA router, the file transfer using the FTP protocol is faster. In particular, file transfer took 68 seconds using the NetFPGA router and 97 seconds using the Click router. The same considerations apply to Fig. 10 for the HTTP protocol where the file transfer took 79 seconds using the NetFPGA router and 113 seconds using the Click router.

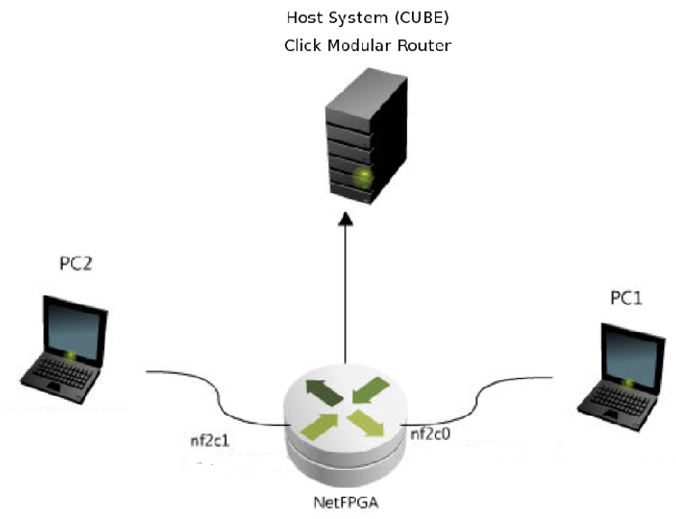


Figure 8. Topology used for the FTP/HTTP file transfer.

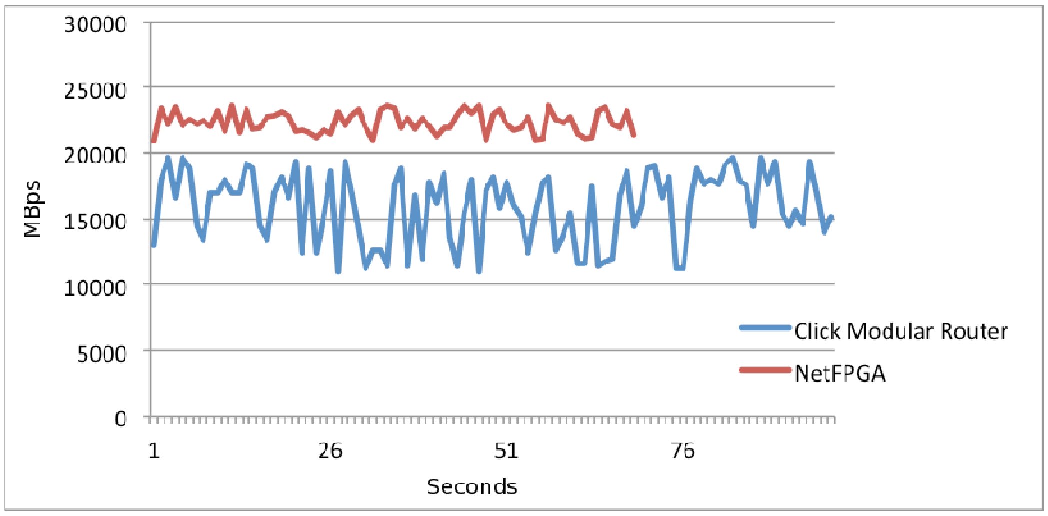


Figure 9. FTP file transfer.

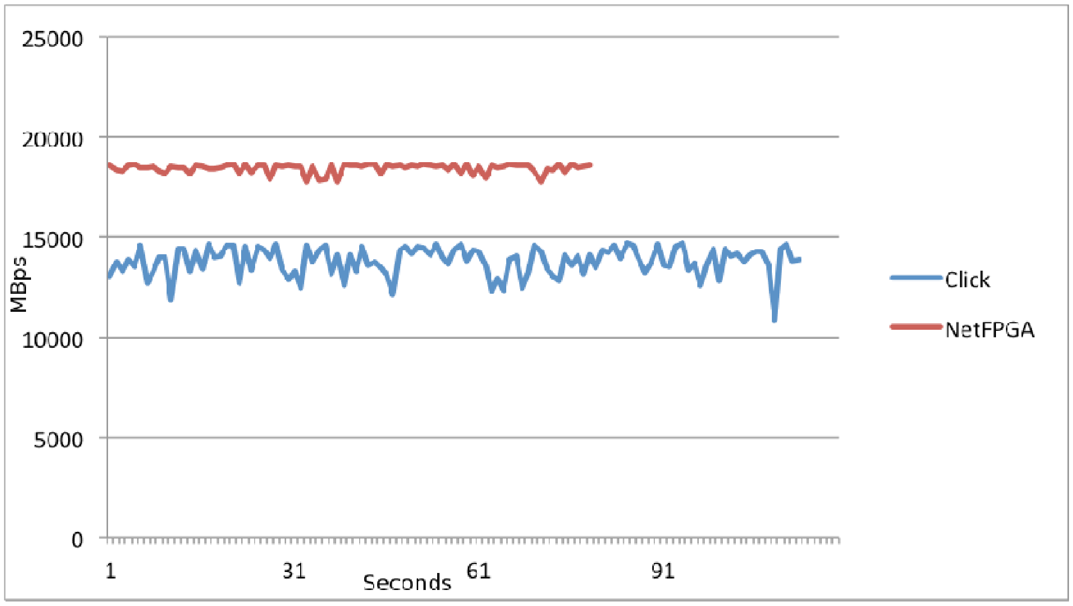


Figure 10. HTTP file transfer.

6. CONCLUSIONS

With the availability of high performance router software based on PC hardware and fast development of commodity PC industry, the PC-based software router attracted more and more scientific researchers and business users because of its flexibility and low cost. We have analysed two router systems:

1. The Click Modular Router is an open, extensible, andconfigurable software router framework. The Click IP router demonstrates that real routers can be built from small, modular elements, and its modularity is compatible with good forwarding performance for PC hardware.
2. NetFPGA is a sandbox for networking hardware – itallows students and researchers to experiment with new ways to process packets at line rate. One of themany systems built using NetFPGA is an IPv4 referencerouter [4], which runs the Pee-Wee OSPF [7] routing protocol, and does address lookup and packet forwarding at line rate.

In this paper we have analysed both the systems and, followingthe test configuration description in RFC1242 [12] and RFC2544 [13], we have shown that the NetFPGA router outperforms the Click router.

ACKNOWLEDGEMENTS

The authors would like to thank the reviewers for providing usefulcomments that have improved the quality of the paper.

7. REFERENCES

[1] Cisco corporation. 1999. distributed wred. Technical report. http://www.cisco.com/univercd/cc/td/doc/product/software/ios111/cc111/wred.htm, as of january 2000.

[2] http://read.cs.ucla.edu/click.

[3] http://www.accenttechnologyinc.com/.

[4] Netfpga group, "netfpga reference router", http://netfpga.org/wordpress/netfpga-ipv4-referencerouter/.

[5] Netfpga project table. http://netfpga.org/foswiki/bin/view/netfpga onegig/projecttable.

[6] Netfpga team. netfpga website. http://netfpga.org.

[7] Stanford university cs344 class, pee-wee ospf protocol details, http://yuba.stanford.edu/cs344/pwospf/.

[8] S. Blake, D. Black, M. Carlson, E. Davies, Z.Wang, and W. Weiss. An architecture for differentiated service. 1998.

[9] B. Chen and R. Morris. Flexible control of parallelism in a multiprocessor pc router. In Proceedings of the General Track: 2002 USENIX Annual Technical Conference, pages 333–346, Berkeley, CA, USA, 2001. USENIX Association.

[10] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. IEEE/ACM Trans. Netw., 1(4):397–413, 1993.

[11] G. Gibb, J. W. Lockwood, J. Naous, P. Hartke, and N. McKeown. Netfpga: An open platform for teaching how to build gigabit-rate network switches and routers. In In IEEE Transactions on Education, August, 2008.

[12] http://www.ietf.org/rfc/rfc1242.txt.

[13] http://www.ietf.org/rfc/rfc2544.txt.

[14] E. Kohler, R. Morris, and B. Chen. Programming language optimizations for modular router configurations. SIGOPS Oper. Syst. Rev., 36(5):251–263, 2002.

[15] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The click modular router.

[16] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The click modular router. ACM Trans. Comput. Syst., 18(3):263–297, 2000.

[17] J.W. Lockwood, N. McKeown, G. Watson, G. Gibb, P. Hartke, J. Naous, R. Raghuraman, and J. Luo. Netfpga - an open platform for gigabit-rate network switching and routing. In International Conference on Microelectronic Systems Education, 2007.

[18] J. Naous, G. Gibb, S. Bolouki, and N. McKeown. Netfpga: reusable router architecture for experimental research. In In PRESTO 08: Proceedings of the ACM workshop on Programmable routers for extensive services of tomorrow, pages 17, New York, NY, USA, 2008. ACM.

[19] T. J. Ott and N. Aggarwal. Tcp over atm: Abr or ubr? SIGMETRICS Perform. Eval. Rev., 25(1):52–63, 1997.

[20] G. Watson, N. McKeown, and M. Casado. Netfpga – a took for network research and education. In In 2nd Workshop on Architecture Research using FPGA Platforms (WARFP), February, 2006.

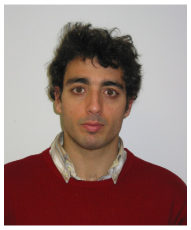
[21] Jad Naous, David Erickson, Adam Covington, Guido Appenzeller, and Nick MvKeown, Implementing an OpenFlow Switch on the NetFPGA Platform, ANCS'08, San Jose, CA, USA, November 6-7, 2008.

[22] M. Ciesla, V. Sivaraman, and A. Seneviratne, URL Extraction on the NetFPGA Reference Router, Developers Workshop 2009

[23] Alfio Lombardo, Diego Reforgiato, Giovanni Schembra, An accelerated and energy-efficient traffic monitor using the NetFPGA, Proceedings of the 19th ACM/SIGDA International symposium on Field programmable gate arrays.

Authors

Leonardo Linguaglossa received the Laurea degree in Computer Science Engineering at the University of Catania in 2010. His research area includes new network technologies such as NetFPGA and OpenFlow.



Diego Reforgiato Recupero has been a Post Doctoral Researcher at the Department of Computer Science and Telecommunications Engineering (DIIT) since 3/11/2008 working on peer-to-peer video. The support for this fellowship has been given by PROVIDEO, a Marie Curie International Grant (IRG) that Dr. Reforgiato won in summer 2008 together with the University of Catania. From 4/1/2005 to 5/1/2008 Dr. Reforgiato was a Post Doctoral Researcher at the Institute for Advanced Computer Studies, University of Maryland College Park, working with Prof. V.S. Subrahmanian. Dr. Reforgiato received the PhD in December 2004, from the University of Naples Federico II in Computer Sciences

Alfio Lombardo received his degree at the University of Catania, Italy, in 1983. Until 1987, he acted as consultant at CREI, the center of the Politecnico di Milano for research on computer networks. In 1988 he joined the University of Catania where he is full professor of Telematics. There he was the leader of the University of Catania team in V and VI FP European projects. Moreover, he was involved in Italian Ministry for University and Scientific Research (MIUR) as leader of the University of Catania team. He is responsible of the OpenLab at the University of Catania. His research interests include distributed multimedia applications, multimedia traffic modeling and analysis, Next generation Internet.

Giovanni Schembra received the degree in electronics engineering from the University of Catania, Italy, in 1991. Working in the Telecommunications area, he received the master degree from CEFRIEL (Milan - Italy), in 1992. In 1995 he received the Ph.D. degree in electronics, computer science and telecommunications engineering. Currently he is an Associate Professor in Telecommunications at the University of Catania. Within the Network of Exellence Newcom he coordinated the Work Package "Architectures and Cross-Layer Aspects". His research interests regard multimedia communications over the Internet, peer-to-peer networks, TCP protocol performance analysis, network coding.