Reducing Power Consumption of Openflow Switch on the NetFPGA Platform

Tran Hoang Vu, Vu Quang Trong, Nguyen Huu Thanh, Pham Ngoc Nam
School of Electronics and Telecommunications, Hanoi University of Science and Technology, Hanoi, Vietnam.
Email: vu.tranhoang@hust.edu.vn

***Abstract* - Reducing power consumption of switches contributes to scale down the overall energy needed by data centers. In this paper, we propose some new energy saving modes for the OpenFlow switch based on NetFPGA platform, which are LOW-POWER mode and SLEEP mode. In normal scenario, switch operates in HIGH-POWER mode; however, when the traffic going through the switch becomes lower than a certain threshold level, it will be activated in LOW-POWER mode; moreover, if there is no traffic, it will operate in SLEEP mode to save most energy. Experimental results show excellent evident in energy saving according to various ranges of traffic load. Our switch can save about 46.6% of power consumption when running at LOW-POWER mode and up to about 60% if running at SLEEP mode by comparing to HIGH-POWER mode.**

Keywords – OpenFlow Switch, NetFPGA, Power control, Data Center Network, Green networking.

I. INTRODUCTION

Power consumption in the worldwide ICT infrastructure is one of top concerns about saving energy trend. In data centers, reduction of electrical usage is placed as the top of to-do list that helps organizations to cut down operating costs. Core servers, storage devices, network equipment, cooling systems cost in high use of electricity in which IT equipment accounts for about 50%, cooling devices takes of 25% [1]. These costs have risen steady and seem to have no limitation due to the expansion of nested devices.

A number of researches have pointed out that data centers can save energy by optimizing some components in servers and cooling systems [2] [3]. However, the energy efficiency of network devices has not received more attention as expected although the percentage of those devices is estimated about 50% in the whole data center [1]. Some mechanisms have been proposed to reduce power consumption of network devices. Those methods focus on two main categories: sleep mode [4] [5], and rate adaption [6] [7]. The former puts inactive devices into very low power mode - sleep state [5], and wakes them up if new packets arrive. However, this method is not suitable for burst traffic [7].

Reducing Energy Consumption in Data Center Network based on Traffic Engineering framework - also called as ECODANE[1] - has been worked out to focus on optimized network devices on which an intelligent network control system is applied. This novel managing system dynamically adjusts whole data center bases on correspond traffic passing through its switches, routers. Such a framework requires a flexible and configurable mechanism in both network device and control protocol such as Software Defined Network (SDN) and OpenFlow architecture. In [8] [9] shows some experimental results about disabling unused ports, links or even switches. Those testing systems could save from 25% to 40% of total power. This has opened new direction in researches nearly. In addition, an OpenFlow Switch Controller (OSC) [10] was designed to receive control messages from the OpenFlow Controller, and then turns on/off target ports. In testbed, NetFPGA OpenFlow switch [11], with OSC, can form a power-aware switch that is not only compliant with OpenFlow protocol [12] [13], but also have extended functions controlling unused ports. A Clock Controller [14] (CC) which can adjust the operating frequency of switch is added into the NetFPGA Switch to extend the ability of saving energy. This

[1] Part of the work in this paper is done within the scope of the ECODANE research project which is co-sponsored by the Ministry of Science and Technology (Vietnam) and the Federal Ministry of Education and Research (Germany)

feature actually is the premise to create some new operating modes of OpenFlow switches to put them into some different levels of power consumption while still keeping bandwidth sufficient to the required traffic flow. In this paper, we focus on some modifications on NetFPGA switches to equip them new power-aware functionalities as the following main parts:

- Based on variable rate on Ethernet links, and some different frequencies of NetFPGA core, three operating modes, including High-Power, Low-Power, and Sleep modes, are given to define power levels corresponding to available states of switch. Switches are now no longer running in the highest power level.
- To support new operating modes, OpenFlow protocol must be extended to carry control messages which enable/disable links, switches and modes of target switches.

The rest of the paper is organized as follows: Section II describes energy aware adaptation on NetFPGA Switch. Section III describes extend openflow standard. Section IV describes experimental results. After all, some conclusions are drawn in Section V.

II. ENERGY-AWARE ADAPTATION ON NETFPGA SWITCH

A. The method to reduce the frequency of switch

In [15], we already know that switch power consumption can be calculated as the below equation:

$$P(f) = PC(f) + KPE(f) + NIEp(f) + RIEr(f) + R0Et(f) \quad (1)$$

where **f** is the NetFPGA clock frequency;

According to the formula (1), we can easily see how the power interrelates with the operating frequency. To save the power, we proposed that the operating frequency should be reduced by two, four, or eighth times or more. In [14] we built a clock divider module shown Fig.1. This module gets clock pulse from cpci_clk, so we only have to scale this clock source down to 32 times to achieve factor 1/64 of the core_clk. In fact, both clocks are kept as clock sources because it is required to keep a 125 MHz output clock in case of no reduction are required or in case it is necessary for doing some calibrations to avoid clock skews when changing clock between various levels. Because of queues between system interfaces – input queue and output queue, we can keep whole system to run properly even frequency is set at lower levels. We could scale down the frequency to 3.90625 MHz and could not go further because of system failure.

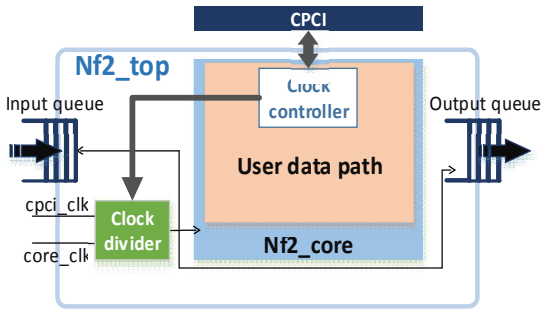

Figure 1. Structure of the frequency divider for NetFPGA Switch

B. Directly control the Ethernet Chip

We can set the bandwidth of the input/output packet stream of four Ethernet ports by changing the frequency of TEMAC module to control the transmit clock to the respective PHY [16]. However, in this paper, we introduce a different method to make switch run at several bandwidth level by directly impact on an internal register in Ethernet chip. Management Data Input/Output (MDIO), also known as Serial Management Interface (SMI) or Media Independent Interface Management (MIIM), is a serial bus defined for the Ethernet family of IEEE 802.3 standards [17] for the Media Independent Interface, or MII. The MII connects Media Access Control (MAC) devices with Ethernet physical layer (PHY) circuits. On the NetFPGA-1G board, we have one NIC of Broadcom BCM5464SR which supports MDIO with an MII register.

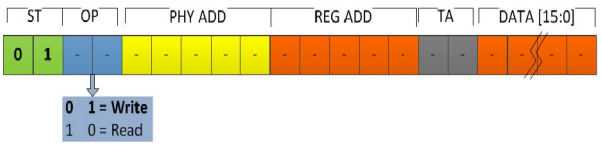

Figure 2. Structure of a MDIO message

In the BCM5464SR NIC, the manufacture included four MII registers to control four Ethernet port individually. Each MII register has its own the

register address, and has the same of physical address. With OpenFlow driver, we can send a message directly to them.

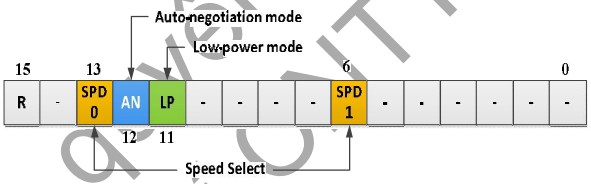

Figure 3. Special functional bits on MII register

To control the bandwidth on each port, first we must disable the auto-negotiation mode by setting the 12th bit to '0'. Then we use the combination of the 6th bit and 13th bit to choose three different link capacities, "00" means that port runs at 10Mbps, "01" is 100Mbps, "10" is 1Gbps and "11" is not used. To put a port to idle mode, we toggle the 11th bit to '0'. Addresses of four Ethernet ports start at 0x440000 and increase by 0x000080 for each port.

C. Define New Operating Mode of NetFPGA Switch

Based on vary link-rates of Ethernet and some different frequency of OpenFlow Switch, we define three new main operating modes. They are *HIGH POWER* mode, *LOW POWER* mode, and *SLEEP* mode. We divide the different modes depended on the operating frequency of switch. In each mode, the bandwidth of one port can be independently controlled and restricted to the corresponding level as in Tab.I. The working mode of OpenFlow Switch is described as follows:

- HIGH POWER mode: Switch runs at the maximum frequency (f = 125 MHz). In this mode, Switch has the highest processing performance, but also consumes the most energy. This mode is activated when at least one Ethernet port is running at 1Gbps. Ethernet ports that are at lower traffic also can be change to lower bandwidth mode (100Mbps/10Mbps) or put to idle mode to reduce total energy.
- LOW POWER mode: When traffic on all Ethernet ports drop below 100Mbps, we decided to cut down the Switch's frequency to 62.5 MHz, and set the Ethernet port operating at maximum 100Mbps or lower. Of course, in this mode, switch power consumption is lower than HIGH POWER mode.
- SLEEP mode: This mode is activated only when there is not any traffic going through any Ethernet port. Switch does not need to process the data flow; we propose frequency at 3.90625 MHz of switch and completely turn off all ports. In this mode, switch is still maintaining connections with Controller and can wake-up intermediately to normal operation. This mode saves the most energy.

Table 1: three new operating modes of netFPGA switch

Mode	Frequency	Bandwidth per port
High power	125 MHz	Idle/10Mbps/100Mbps/1Gbps
Low power	62.5 MHz	Idle/10Mbps/100Mbps
Sleep	3.90625 MHz	Idle

The new three working modes of NetFPGA switch can solve two drawbacks [16] when reducing power consumption. First, in contemporary switch, SLEEP modes may be too drastic because this affects the state of link (as perceived by the corresponding receiver) or may cause software support issues in operating systems. However, in OpenFlow system, we have a NOX controller where an algorithm is running to ensure the capacity on links. Only in the suitable situation can switch be put in power save mode or SLEEP mode. In addition, even in SLEEP mode, the switch is interacts with controller, control signals are always exchanged between them and switch only has to change its frequency which can be scale up in few cycles of clock.

III. EXTEND OPENFLOW STANDARD

OpenFlow messages are sent between Controller and OpenFlow switches for managing, controlling them through OpenFlow channel. Each OpenFlow message begins with the OpenFlow header [18]:

```
struct ofp_header {
    uint_8  version;
    uint_8  type;
    uint_16 length;
    uint_32 xid;
};
```

The Switch receives instructions from the OpenFlow controller to control the working modes. These instructions include:

- OFPT_SWITCH_MOD message:

Type of message: Controller to Switch

Length: 32 Bytes

Functions: Change operating mode of Switch

Structure:

```
struct ofp_switch_mod {
    struct   ofp_header header;
    uint8 t  switch mode;
    uint8_t  pad[3];
};
```

The *switch_mode* field stores the information to configure the switch as shown in Tab.II. Structure of this message and switch mode field is described as below:

Table 2: ofpt_switch_mode message

OpenFlow Header	Switch Mode	Pad
8 Bytes	1 Bytes	3 Bytes

A value '1' in the flag bit will instruct the Switch to change its state. The MODE field indicates the working mode of the switch as HIGH POWER (M_1M_0=00), LOW POWER (M_1M_0=01) and SLEEP (M1M0=10). M1M0=11 is not used and set for reserved mode.

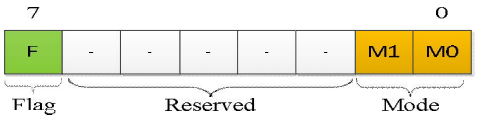

Figure 4. Switch mode field structure

- OFPT_PORT_MOD message:

Type of message: Controller to Switch

Length: 32 Bytes

Functions: Configure bandwidth of port on Switch.

Structure:

```
struct ofp_port_mod {
    struct   ofp_header header;
    uint16_t port_no;
    uint8_t  hw_addr[OFP_ETH_ALEN];
    uint32 t config;
    uint32_t mask;
    uint8 t  link state;
    uint32_t advertise;
    uint8_t  pad[3];
};
```

The *link_state* field stores the information to configure the port as shown in Fig.5. A value '1' in the flag bit will instruct the port to change its state. While {P_1, P_0} indicates port number, {B_1, B_0} is bandwidth of that port: "11" means port is running at 1Gbps, "10" means port is running at 100Mbps, "01" means port is running at 10Mbps, and "00" means port is on idle state.

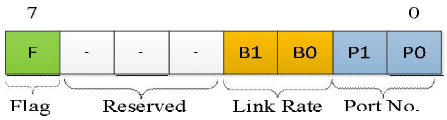

Figure 5. Link state field

The algorithm shown on flowchart in Fig.6. below illustrates the process of receiving and processing OpenFlow Switch control messages. After successfully handshaking with controller, switch runs in listening mode – capture all control messages and determine running mode. As soon as receiving new operating mode, switch *changes* the limit bandwidth on each port and set the corresponding frequency of FPGA chip. It is notable that before turning switch to *SLEEP* mode, switch must be sure that there is no packet left in system queues; therefore, a queue monitor is used to check if queues are empty or not.

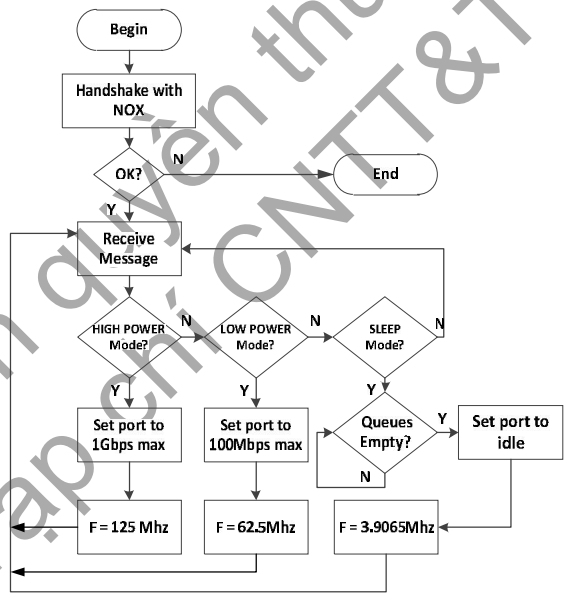

Figure 6. Communicating flow chart between the NOX and Switch

IV. EXPERIMENTAL RESULTS

Our test-bed has a NOX controller and an Openflow switch with modified controller that enables to send and receive new OFPT_PORT_MOD message. The NOX controller version 1.0.0 is implemented on a host PC running Ubuntu version 10.10. OpenFlow switch version 1.0.0.4 based on NetFPGA version 3.0.1 which is developed by Stanford [18] is used.

We also use a client and a server to generate traffic load on ports of Openflow switch in oder to put switch under hardworking mode and consume highest power. In fact, PC1 sends a stream at approximately 1Gbps and PC2 will monitor state of Openflow switch. The test-bed's model can be seen on Fig.7 and Fig.8. Oscilloscope and Power Measure Board are used to read ADC value at test-point 3.3V and 5.0V via PCIEXT-64UB [19], and then display, and calculate power consumed.

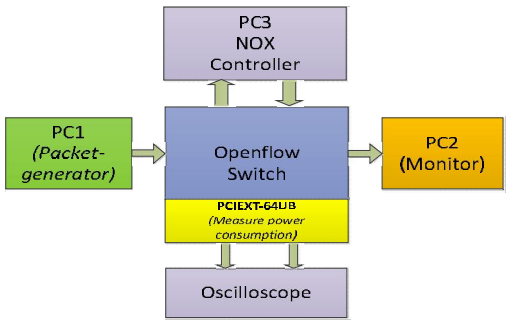

Figure 7. Testbed for power consumption measurement

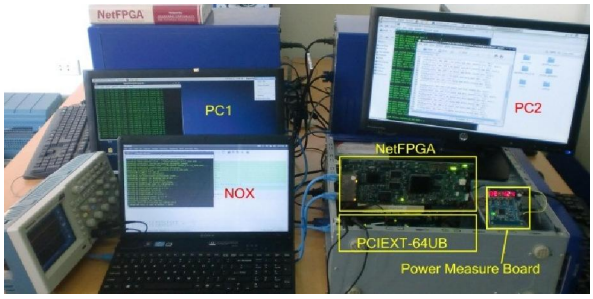

Figure 8. Test-bed system for measurement

A. Power of Switch at Multi-frequency

With each level of clock frequency, we have measured the power consumption of the FPGA chip, and got results shown on the chart in Fig.9.

We also measured the power consumption of the whole NetFPGA card running at the different frequencies. The results are shown on chart in Fig.10.

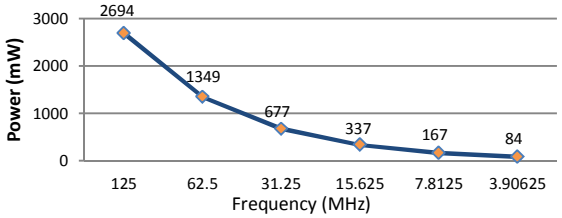

Figure 9. Power consumption of FPGA chip depends on frequency

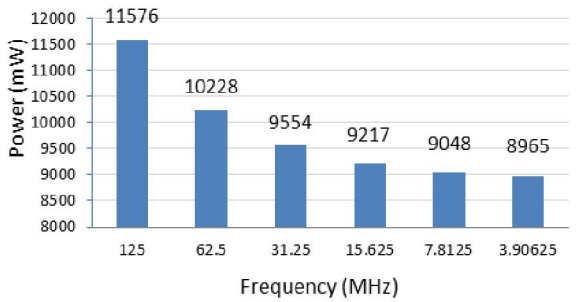

Figure 10. Power consumption of whole NetFPGA board at multiple frequencies

In conclusion, our switch can save about **97%** of dynamic power consumption of FPGA chip when running at lowest frequency mode. In total, if using this clock controller, switch can save about **22.6%** energy (**2611** mW of **11576** mW). Compared to another paper [20], they only control the frequency of several functional blocks in the FPGA chip, and the amount of saving power is a small number. Instead of modification of some sub-blocks, we change the frequency of whole FPGA chip, and then the energy saved is much larger than ever.

B. *Power of Switch at different bandwidth level*

We measured the power consumption of switch with different bandwidth: 1Gbps, 100Mbps, 10Mbps and the off state on four ports. Based on the results shown in Fig.11, the energy saves the most, about 4W, when we decline the bandwidth from 1Gbps down- to 100Mbps. When we continue to decrease bandwidth to 10Mbps, we do not save much energy.

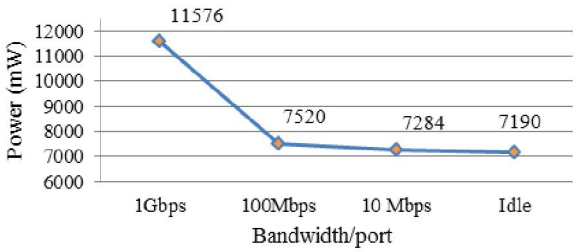

Figure 11. Power consumption depends on bandwidth

Based on those results, the energy saves the most, about 4W, when we decline the bandwidth from 1Gbps down- to 100Mbps. Moreover, we continue to decrease bandwidth to 10Mbps, we do not save much energy.

C. *Power of Switch at three new defined working modes*

We designed the software monitor that show state of switch when running at different modes. Firstly, we set NetFPGA switch to run in three modes as proposed. Those modes are monitored by software shown on Fig.12, Fig.13 and Fig.14. Traffic passing through each mode of switch is set at the maximum level.

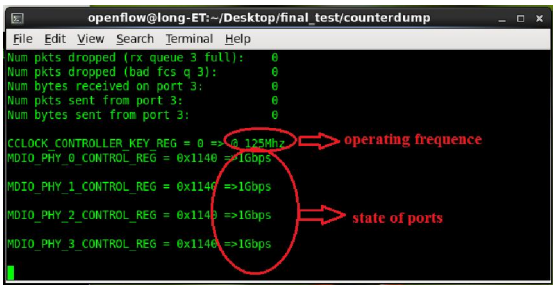

Figure 12. Monitor states of switch when running High power mode

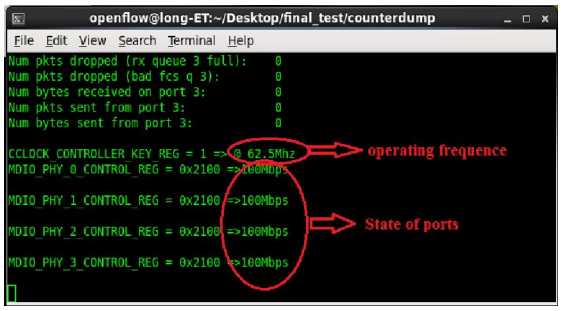

Figure 13. Monitor states of switch when running Low power mode

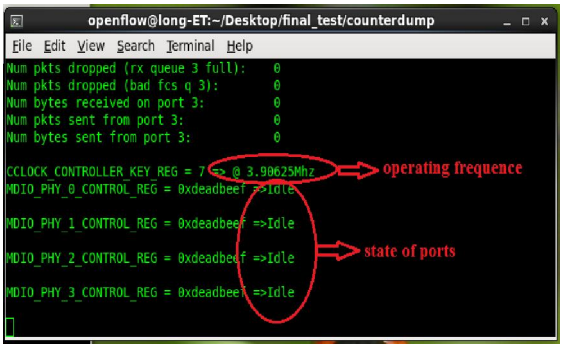

Figure 14. Monitor states of switch when running Sleep mode

Then we measured the power consumption of the whole NetFPGA card. The results are shown on chart in Fig.15. Those results are the greatest value of the power consumption corresponding with modes on the Openflow switch.

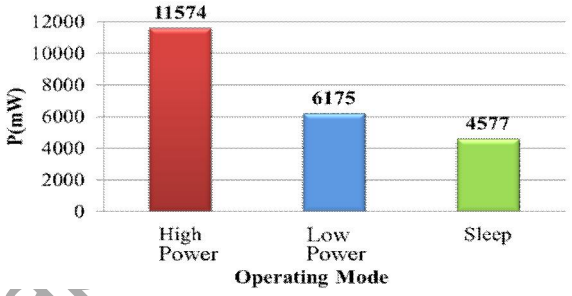

Figure 15. Power of NetFPGA Switch at new three operating modes

In conclusion, our switch can save about **46.6%** of power consumption when running at LOW POWER mode and up to about **60%** if running at SLEEP mode by comparing to HIGH POWER mode.

V. CONCLUSION

In this paper, we have defined and implemented at least two new modes of OpenFlow Switch to reduce power consumption that enables energy saving in data centers. We also have implemented some mechanisms to flexibly and effectively reduce power consumption of switch. Finally, we have defined new OpenFlow messages and protocols

We are focusing on building a standalone NetFPGA switch running an embedded OS without host PC, supporting OpenFlow protocol, and still holding our new advanced modifications.

REFERENCES

[1] Five Strategies for Cutting Data Center Energy Costs Through Enhanced Cooling Efficiency, White paper from the Experts in Business-critical Continuity, http://www.thegreengrid.org

[2] Emerson Network Power white paper, Energy Efficient Cooling Solutions for Data Centers , www.liebert.com

[3] http://www.cs.berkeley.edu/~istoica/classes/cs294/09/CERN_Whitepaper_r04.pdf

[4] M. Gupta, "A feasibility study for power management in lan switches," in Proceedings of the 12th IEEE International Conference on Network Protocols, 2004.

[5] M. Gupta and S. Singh, "Dynamic ethernet link shutdown for energy conservation on ethernet links," in IEEE International Conference on Communications, 2007.

[6] C. Gunaratne, K. Christensen, B. Nordman, and S. Suen, "Reducing the energy consumption of ethernet with adaptive link rate (alr)," IEEE Transactions on Computers, vol. 57, pp. 448–461, 2008.

[7] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing network energy consumption via sleeping and rate-adaptation," in Proc. of NSDI, 2008. Berkeley, CA, USA: USENIX Association, pp. 323–336.

[8] Truong Thu Huong, Pham Ngoc Nam, Nguyen Huu Thanh, Daniel Schlosser, Michael Jarschel, Rastin Pries, "ECODANE – Reducing Energy Consumption in Data Center Networks based on Traffic Engineering" (poster), in the Proceedings of 11th Würzburg Workshop on IP: Joint ITG and Euro-NF Workshop "Visions of Future Generation Networks" (EuroView2011), August 1st - 2nd 2011, Würzburg, Germany

[9] Nguyen Huu Thanh, Pham Ngoc Nam, Truong Thu Huong, Nguyen Tai Hung, Luong Kim Doanh and Rastin Pries, "Enabling Experiments for Energy-Efficient Data Center Networks on OpenFlow-based Platform", in the Proceedings of the 4th International Conference on Communications and Electronics 2012 (ICCE 2012), pp. 239-244, August 1st - 3rd 2012, Hue, Vietnam.

[10] Tran Hoang Vu, P.N.Nam, T.Thanh, L.T. Hung, L.A.Van, Ng. D. Linh, T.D. Thien, N.H.Thanh, "Power Aware OpenFlow Switch Extension for Energy Saving in Data Centers" In Proceeding of the 2012 International Conference on Advanced Technologies for Communications (ATC 2012), pp. 309-313. Hanoi, Vietnam.

[11] John W. Lockwood, Nick McKeown, Greg Watson, Glen Gibb, Paul Hartke, Jad Naous, Ramanan Raghuraman, and Jianying Luo "NetFPGA- An Open Platform for Gigabit-rateNetwork Switching and Routing" IEEE International Conference on Microelectronic Systems Education, June 3-4, 2007 Sa Diego, CA.

[12] "New HP 5400R zl2 Switch Series" http://h17007.www1.hp.com/us/en/networking/products/switches/index.aspx#.U-dORPl_tmU

[13] "IBM System Networking RackSwitch G8264" http://www03.ibm.com/systems/networking/switches/rack/g8264/index.html

[14] T.H.Vu, T. Thanh, V. Q. Trong, N.H. Thanh, P.N. Nam "Energy Saving for OpenFlow Switch on the NetFPGA platform Using Multi-Frequency" in International Journal of Computing and Network Technology, No.1, pp.9-15, 1 Jan 2014.

[15] Lombardo.A, Panarello. C, Reforgiato. D, Schembra. G, "Power control and management in the NetFPGA Gigabit Router" Conf. FutureNetw, Berlin, p.1-8, July,2012

[16] Y. Sinan Hanay, Wei Li, Russell Tessier, T.Wolf "Saving Energy and Improving TCP Throughput with Rate Adaptation in Ethernet" Conf. ICC, Ottawa, p. 1249 – 1254, June 2012

[17] "IEEE 802.3 Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access method and Physical Layer specification",

http://standards.ieee.org/getieee802/download/802.3-2008_section2.pdf

[18] "OpenFlow Switch Specification Version 1.1.0 Implemented (Wire Protocol 0x02)", February 28, 2011.

[19] "Product Specifications and User Manual of PCI-EXT64U/UB and PCIeEXT-16HOT" v1r08 – April 12, 2013, Ultraview Corporation.

[20] W. Meng, Y.Wang, C. Hu, K.He, J.Li, B.Liu *"Greening the Internet using Multi-Frequency Scaling Scheme"* Conf.AINA, Fukuoka, p. 928 – 935, March 2012.

AUTHORS' BIOGRAPHIES

Tran Hoang Vu received B. Eng. degree in Electronics and Telecommunications from Da Nang University of Technology and M.Sc. degree from the University of Danang (Vietnam) in 2004 and 2007, respectively. From 2004 until now he has been working at Danang College of Technology-The University of Danang, Vietnam. His research interests include Reducing power consumption of Data Center Networks, reconfigurable embedded systems and low-power embedded system design.

Vu Quang Trong received B. Eng. Degree In electronics and Telecommunications from Hanoi University of Science and Technoloy (Vietnam) in 2013. Since 2010 he has been working at the Embedded System and Reconfigurable Computing Lab, HUST. His research interests include Reducing power consumption of Data Center Networks, reconfigurable embedded systems and low-power embedded system design.

Nguyen Huu Thanh received B.S and M.Sc in Electrical Engineering from Hanoi University of Science and Technology, Vietnam in 1993 and 1995, respectively. In 2002, he received his PhD with summa cum laude in Computer Science from the University of Federal Armed Forces Munich (Germany). From 2002 to 2004 he has been with the Fraunhofer Institute for Open Communication Systems (FOKUS) in Berlin, Germany. From 2004 Nguyen Huu Thanh is associate professor in the School of Electronics and Telecommunications, HUST. His research interests include radio resource management in 4G systems, QoS/QoE, the Future Internet, energy-efficient networking and software-defined networking.

Pham Ngoc Nam received B. Eng. degree In electronics and Telecommunications from Hanoi University of Science and Technology (Vietnam) and M.Sc. degree in Artificial Intelligence from K.U. Leuven (Belgium) in 1997 and 1999, respectively. He was awarded a Ph.D. degree in Electrical Engineering from K.U.Leuven in 2004. From 2004 until now he has been working at Hanoi University of Science and Technology, Vietnam. His research interests include reconfigurable embedded systems and low-power embedded system design.