International Journal of Computing and Network Technology

Energy Saving for OpenFlow Switch on the NetFPGA platform Using Multi-Frequency

Tran Hoang Vu*, **Tran Thanh**, **Vu Quang Trong**, **Nguyen Huu Thanh**, **Pham Ngoc Nam**

School of Electronics and Telecommunications, Hanoi University of Science and Techonogy, Viet Nam

E-mail: vu.tranhoang@hust.edu.vn

Received: 15 Aug. 2013, Revised: 20 Oct. 2013, Accepted: 22 Oct. 2013, Published: 1 Jan. 2014

Abstract: Improving energy efficiency in switch is becoming an increasingly important research topic. In this paper, we present the lesson gained by experimenting with a power management mechanism aimed at reducing the power consumption. One solution for this problem is to control intelligently the power consumption of switches used in data centers. This paper proposes an extension to OpenFlow switches to support different power saving modes. The extension includes defining new messages in Openflow standard and designing a Clock Controller (CC) on the hardware of switch based on NetFPGA platform to implement a frequency driver of the switch under various ranges of traffic loads. Experimental results demonstrate an excellent energy saving according to different working modes. Our results show that all the functions of the Clock Controller of the designed system are valid, and the designed system is feasible for the switch to save energy.

Keywords: Openflow Switch, NetFPGA, Power control, Data Center Network, Green networking.

I. INTRODUCTION

One consequence of the growth in Internet and network equipment is the increase of power consumption. It is calculated that Internet power consumption seizes approximately 1% − 4% of total electricity consumption in countries with broadband Internet [1]. Among all kinds of network devices (excluding PCs), switches and routers represent most of power consumption of the Internet [2] [1]. Although current link utilization averages 30% and peaks 45% [3], switches and routers always use up their capacity (24 hours a day, 7 days a week) [4]. This results enormous energy waste and brings the opportunity for substantial energy conservation.

Therefore, the issue of network energy efficiency is receiving considerable attention [5], [6], [7], [8], and some novel hardware devices enabling different power states are promising [9]. In [10] and [11] we proposed a framework called [1] ECODANE (Reducing Energy Consumption in Data Center Networks based on Traffic Engineering) which focuses on optimizing power consumption of network components by designing an intelligent network control system that dynamically adapts the set of active network components corresponding to the total traffic going through the data center. The experimental results in [10] and [11] have shown that by disabling unused links (i.e. ports) and switches, an energy saving of 25% to 40% can be achieved. In [12] we design an OpenFlow Switch Controller (OSC) which receives control messages from the OpenFlow controller and controls switches and links. The design of OSC can be used as a block in OpenFlow compliant switches. Our prototype OSC can be used together with a NetFPGA based OpenFlow switch [13] for power aware networking research. Such a framework, however, requires the use of a more flexible and configurable network architecture such as Software-Defined Networking (SDN), in which the OpenFlow is one of the technologies widely used. There have been attempts to extend the current commercial switches to support OpenFlow protocols [14] [15] or to build a complete OpenFlow switch for research purposes [13]. However, these switches do not have power aware functionalities. In this paper, we propose an extension to OpenFlow switches to support different power saving modes. These modes are likely energy

[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).

saving for switch when input traffic changes. The main contributions of our work are the following:

- We extend the OpenFlow protocols to include new message which enables the OpenFlow controller to control switches to work at different frequencies.
- We design a Clock Controller(CC) which receives control messages from the OpenFlow controller and reduce the frequency adapted on traffic throughput the switch to save energy.

The rest of the paper is organized as follows. Section 2 presents the related work. The design of a Clock Controller is described in Section 3. Section 4 describes new messages which we propose to add to OpenFlow standard to support power management functionalities. Section 5 describes Experimental results. Conclusions are drawn in section 6.

II. Related Work

This Section presents an overview of related technology, standard and framework used in next Sections.

A. Openflow

OpenFlow protocol is an open and standardized protocol for the network controller communicating with the switch [16]. In a classical router or switch, the fast packet forwarding (data path) and the high level routing decisions (control path) occur on the same device. An OpenFlow Switch separates these two functions. The data path portion still resides on the switch, while high-level routing decisions are moved to a separate controller (Fig. 1). The OpenFlow Switch and Controller communicate via the OpenFlow protocol, which defines messages, such as packet-received, send-packet-out, modify-forwarding-table, and get-stats.

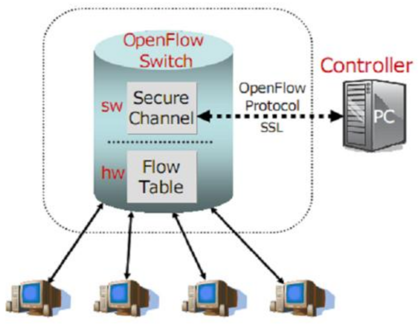

Figure 1. OpenFlow Switch.

B. NetFPGA System

An Openflow Switch currently has two main parts which are Software on a Linux OS, and Hardware on a NetFPGA-1G Board.

- The first part contains interface and driver for Openflow Switch. We can obtain this part from head website [16]. The software connects directly to Controller over a sercure channel with its own Openflow protocol. All routing infomation are exchanged on this line. There is no data-flow carried or mixed with control-flow.

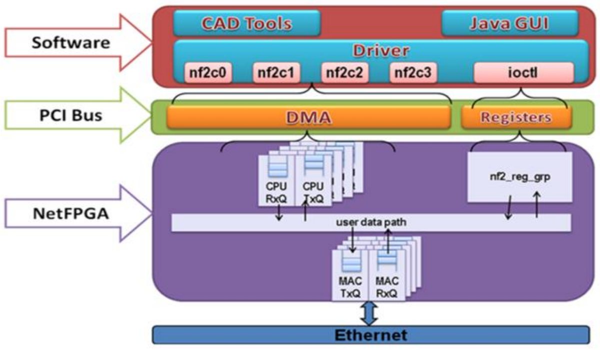

Figure 2. Diagram of the NetFPGA system.

- The second part is a NetFPGA-1G board containing a FPGA using Xilinx Virtex-II Pro 50 and a Gigabit Ethernet using Broadcom BCM5464 PHY [17]. This part gets routing infomation from software, update routing table and forward all packets from inputs to output.

C. Power Consumption Profiling of NetFPGA

In fig.3 describes the functional blocks of the Openflow switch on NetFPGA platform.

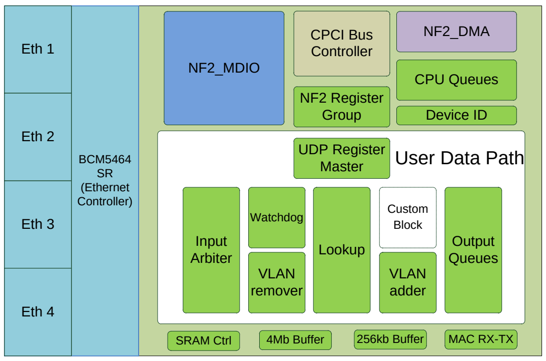

Figure 3. Block diagram of Openflow Switch on NetFPGA

We use PCIEXT-64UB board [18] to measure the power consumption of each block. The measurements we have used:

- Measurement of the total energy consumption of the switch, when not configured FPGA and not have any Ethernet port is connected. The result is that the static power consumption Switch. Pstatic ≈ 4797 mW
- Measurement of the total energy consumption of the switches, when all four ports operate with a bandwidth of around 1Gbps. This is the largest energy consumption of the switch when operating normally. $P_{max} \approx 11600$ mW
- Disconnect each switch's Ethernet port and the measured power consumption per port. Peth ≈ 1013 mW / port

We intervene in FPGA hardware code to disable / enable of block and a measurement energy for each block.

TABLE 1: RESULTS MEASUREMENT EACH BLOCKS POWER OF OPENFLOW SWITCH

Components	**P (mW)**
Static Power	4 797
4 Ethernet ports	4 050
NF2Core	2 733
Total	11 580

The results of our measurement are quite close to some of the results published in the paper [19]. Based on these results, we propose some solutions to reduce the energy consumption of the NF2CORE block. The solution is presented in section **III** and **IV**.

III. HARDWARE DESIGN OF CLOCK CONTROLLER

In this Section, we describe the design of CC (Fig.5) in the Openflow switch's hardware based on NetFPGA platform. First, we present a method to scale down frequency. Then we set out the model and design of CC. The content presented in this section, we give a solution for energy research for the new-generation switches.

A. Method to scale down Frequency

As the paper [20], Switch power consumption is calculated as below formula:

$$P(f) = P_C(f) + KP_E(f) + N_I E_p(f) + R_I E_r(f) + R_0 E_t(f) \quad (1)$$

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 to: f/2, f/4, f/8, f/16, f/32, f/64

As we can see, the cpci_clk has its own value which equals to a half of the core_clk. Therefore, a way of dividing switch operating frequency by 2 can be done much easily by using cpci_clk instead of core_clk. To implement this, we use a multiplexer (available in OpenFlow source code):

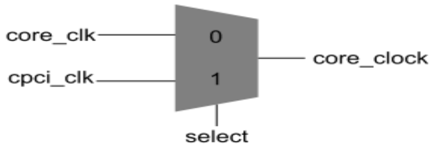

Figure 4. Dividing frequency by 2.

This method, however, only is applied in case of frequency is divided by 2. In other cases, we can not divide frequencies deeper because the clock related to divided-by-*n* frequency is unavailable (*n* is a power of 2 but larger than 2). To achieve lower frequency levels, we have to build a new frequency dividing module. This module gets clock pulse which is ***cpci_clk*** (f/2), so we only have to divide frequencies by up to 32, satisfy 32*2 = 64. After dividing, resulted frequency is pushed to all modules which are using the ***core_clk_int***.

By dint of queues between system interfaces, we can keep whole system to run properly even frequency is set at lower levels.

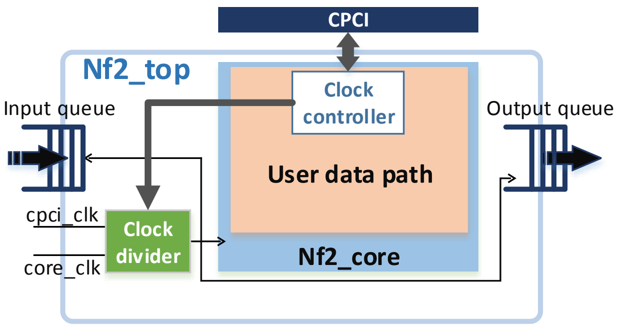

Figure 5. Switch's dividing frequency diagram.

B. Building Clock Controller on Hardware.

This session describes Clock Controller Module's Design which receives a controlling signal from processing center and execute this signal. To carry out, we need to know the received signal's in/out direction which is transported from center via NOX, and the processing approach inside Switch 's processing center.

The information coming to User Data Path clock has two types which have the separate approaching ways:

- Control signals coming from NOX: They are updated in Register Master, after that, at each clock pulse, the register value is shifted into a successor register according to pipeline diagram. Control signals then come to minor blocks where they can be read/written depending on block functions and be cycled back to Register Master, diverted to NOX. Based on that, NOX can analyze the system responders.
- Data packets coming from Ethernet gates: They are transported through a specific link in series of queues belonging to multi blocks in User Data Path. The processed data packets will be diverted to other Ethernet gates.

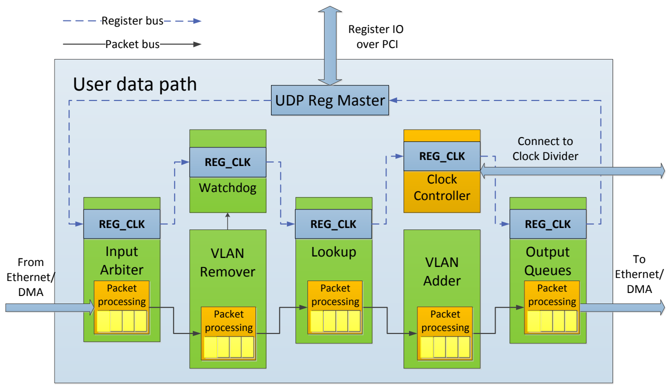

Figure 6. Clock Controller location in User Data Path

As described in Fig.6, these controlling signals come form NOX to Switch are transport through Register Bus via different packets. Therefore, inserting a self-defined controlling block is completely possible to catch up these data packets and process controlling demands properly.

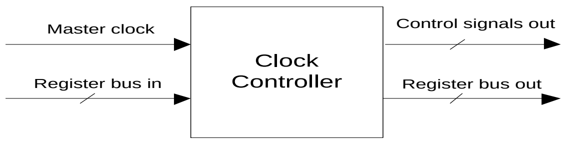

Figure 7. Describing Clock Controller signals

As mentioned before, we have developed a controlling module named Clock Controller which reads the messages from NOX and controls output clock according to the received information. The Clock Controller consists of the following inputs and outputs:

- Master Clock: main clock signal from NF2_CORE which provides clock pulse to all modules.
- Register Bus In: receives control signals from the previous module.
- Register Bus Out: forwards control signals to the next module.
- Control Signals Out: puts signals to control the clock frequency divider.

Inside the Clock controller module, we create a register that saves control demands in each clock cycle. According to value stored this register, Clock Controller module can give an appropriated divided frequencies. We will introduce signal construction send to Clock Controller module in the next session.

IV. EXTEND OPENFLOW STANDARD

In this Section, we present about particular parts of extending the Openflow protocol messages to control operating frequency of Switch with a definition of new operating mode of switch and their parameters.

OpenFlow messages sent between Controller and OpenFlow switches for managing, controlling them through OpenFlow channel. Each Openflow message begins with the OpenFlow header [16]:

```
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 mode of itself. 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];
};
```

TABLE II: OFPT_SWITCH_MOD MESSAGE

Opflow header	Datapath ID	Switch State	Option	Pad
8bytes	8 bytes	1bytes	4bytes	3bytes

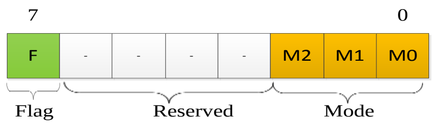

Figure 8. Switch state field structure

A value '1' in the flag bit will instruct the Switch to change its state. The MODE field indicates the working frequency of switch. Table III describes some available frequency levels according to three bits {M2, M1, M0} of *switch_state* in OFPT_SWITCH_MOD message. Switch runs normally at 125MHz with $M_2M_1M_0$ = "000". When increasing value of that field, controller can tell switch to run at lower frequencies. We could scale down the frequency to 3.90625 MHz and could not go further because of system failure.

TABLE III: AVAILABLE FREQUENCY OF SWITCH

Mode	M2	M1	M0	Frequency of Switch MHz	Factor
0	0	0	0	125	1
1	0	0	1	62.5	1/2
2	0	1	0	31.25	1/4
3	0	1	1	15.625	1/8
4	1	0	0	7.8125	1/16
5	1	0	1	3.90625	1/32
6	1	1	0	Not available(N/A)	N/A
7	1	1	1	N/A	N/A

Algorithm flowchart in Fig.9 below illustrates the process of receiving and processing Openflow Switch control messages. In fact, when we implement this module NF2_Core, openflow switch can not response to nox controller if it run at frequency lower than 3.90625MHz.

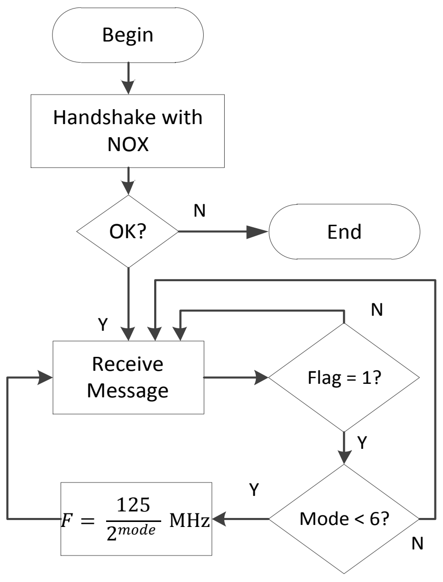

Figure 9. Communicating flow chart between the NOX and Switch

V. EXPERIMENTAL RESULTS

In order to test our design, we have built a hardware test-bed including a NOX Controller, and an OpenFlow Switch based on NetFPGA-1G Board (Fig. 10).

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 [16] is used.

We also use PC1 and PC2 are connected port C0 and C1 on Switch to generate packets on links. PC3 and PC4 are connected to port C2 and C3 and use timestamp field in header of TCP packets to determine bandwidth of switch.

An Oscilloscope and a Power Measure Board are used to read ADC value at test-point 3.3V and 5.0V via PCIEXT-64UB and then display, and calculate power consumed.

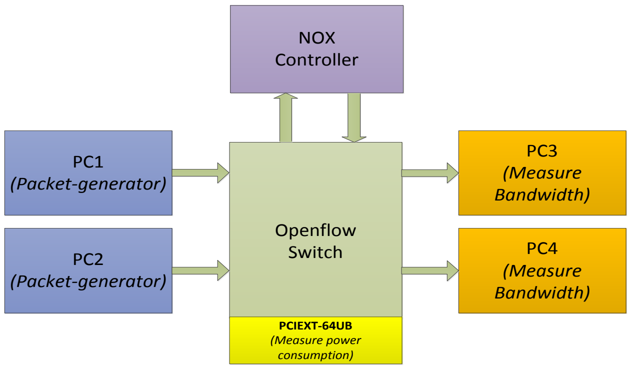

Figure 10. Testbed for power consumption measurement

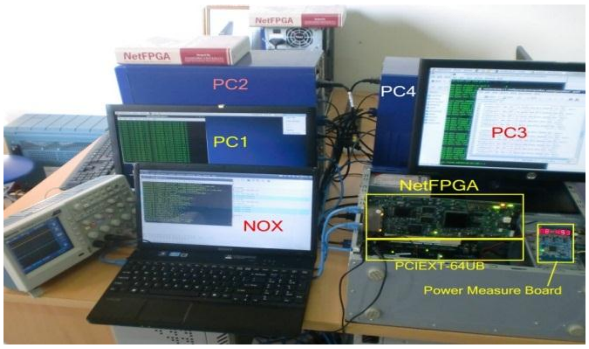

Figure 11. The experimental setup with NetFPGA switch, an extender board, NOX controller and host PCs

Experimental results show that NOX could control the switch with new message, and switch could run at lower frequencies.

Firstly, we measure the power consumption of routing module on Switch which part we changed input frequency – Nf2core with defined modes above. Results of this measurement are described on Fig.12. In normal mode 0, Nf2core needs about 2.694W, while it only needs 1.347W when running at mode 1. The higher mode we set, the more energy saved.

After that, we run switch at bandwidth of 1Gbps per port and only change frequency. In this test, we not only get power consumption of switch but we also gain an important parameter of switch, that is through put could be processed in and out of switch. The results is showed in Table IV and Table V. Interestingly, only at the frequency lower than 31.25MHz could the switch not be able to forward any packages, that means we can put switch to a state in which switch just sleep.

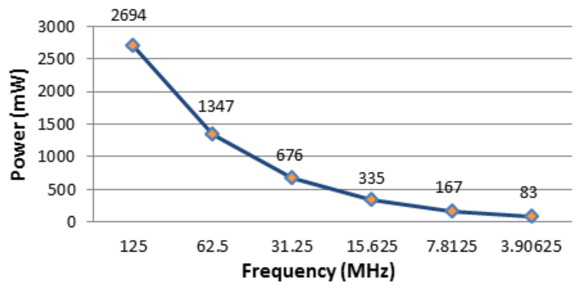

Figure 12. Power consumption of FPGA chip depends on frequency.

TABLE IV: SUMMARY OF NETFPGA POWER MODE

Mode	Frequency of Switch (MHz)	Factor	Power of Swich P(mW)	Power saved (mW)
0	125	1	11576	0
1	62.5	1/2	10228	1348
2	31.25	1/4	9872	1701
3	15.625	1/8	9554	2022
4	7.8125	1/16	9271	2305
5	3.90625	1/32	8965	2611

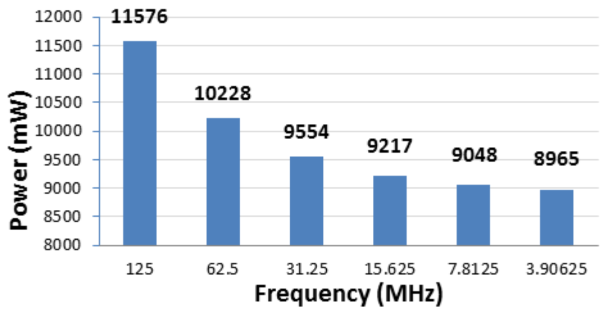

Figure 13. Power consumption of whole NetFPGA board at multiple frequencies

TABLE V: MULTI FREQ AND THROUGHPUT

Mode	Frequency (MHz)	Throughput max (Mbps)
0	125	910
1	62.5	125
2	31.25	8.4
3	15.625	0
4	7.8125	0
5	3.90625	0

In conclusion, our switch can save about **95%** of power consumption of routing module when running at lowest frequency mode. In total, if using this clock controller, switch can save about **22.5%** energy (2611mW of 11576mW).

VI. CONCLUSION

In this paper, we have successfully built a Clock Controller in OpenFlow switch to scale down frequency of the switch in order to save energy in data centers. This solution somewhat reduces the power consumption in data centers. At the same time, we also create the new control messages, as well as in actual building is a perfect model for testing the deeper study of hardware. Moreover, the design of the Clock Controller block can be used as a reference design for power management block in commercial OpenFlow compliant switches.

Based on the results obtained in this paper, in the future we are going to propose the energy-saving modes for the OpenFlow switch are Low-power mode and Sleep mode.

REFERENCES

[1] J. Baliga, K. Hinton, and R. Tucker, "Energy consumption of the internet," in Proceedings of COIN-ACOFT 2007
[2] M. Gupta and S. Singh, "Greening of the internet," in Proc. of ACM SIGCOMM, 2003.
[3] http://arstechnica.com/old/content/2008/09/what-exaflood-net-backbone-shows-no-signs-ofosteoporosis.ars, 2008.
[4] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright, "Power awareness in network design and routing," in Proc. of IEEE INFOCOM, 2008.
[5] M. Gupta and S. Singh, "Greening of the internet," in Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications, SIGCOMM '03, (New York, NY, USA), pp. 19–26, ACM, 2003.
[6] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing network energy consumption via sleeping and rate-adaptation," in Proceedings of NSDI, USA, 2008.
[7] P. Barford, J. Chabarek, C. Estan, J. Sommers, D. Tsiang, and S. Wright, "Power awareness in network design and routing," in Proc. of IEEE INFOCOM 2008, Phoenix, USA, April 2008, 2008
[8] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "A power benchmarking framework for network devices," in Proceedings of the 8th International IFIP-TC 6 Networking Conference, NETWORKING '09, (Berlin, Heidelberg), pp. 795–808, Springer-Verlag, 2009.
[9] "Ciscoenergywise,http://www.cisco.com/,"
[10] 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
[11] 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.
[12] Tran Hoang Vu, Pham Ngoc 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
[13] "Netfpga gigabit router." [Online]. Available: www.netfpga.org
[14] [Online]. Available: http://www.hp.com/networking/
[15] [Online]. Available: http://www-03.ibm.com/systems/networking/switches/rack/g8264/
[16] http://www.openflow.org/documents/openflow-spec-v1.1.0.pdf
[17] https://github.com/NetFPGA/netfpga/wiki/Guide
[18] http://ultraviewcorp.com/displayproduct.php?part_id=4&sub_id=2
[19] V. Sivaraman, A. Vishwanath, Z. Zhao, and C. Russel, "Profiling per-packet and per-byte power consumption in the NetFPGA gigabit router," *IEEE INFOCOM Workshopon Green Communications and Networking (GCN)*, 2011.
[20] 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

Tran Hoang Vu is a Ph.D. student in Electrical Engineering of Hanoi University of Science and Technology (Vietnam), where he has been since 2010. He 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 2008, 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 embeddedsystems and low-power embedded system design.

Tran Thanh is a Ph.D. student in Electrical Engineering in ESRC laboratory of Hanoi University of Science and Technology (Vietnam), where he has been since 2010. He has a B. Eng. degree in Electronics and Telecommunications from Da Nang University of Technology and a M.Sc. degree from the University of Danang (Vietnam) in 2008, respectively. He works at The Vietnam Research Institute of Electronics, Informatics and Automation. His research is in reconfigurable computing, reconfigurable embedded systems and FPGA security.

Vu Quang Trong received B. Eng. Degree In electronics and Telecommunications from Hanoi University of Science and Technology (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.