RESEARCH ON TESTBED SYSTEM AND NEW METHOD TO SAVE ENERGY FOR OPENFLOW SWITCH

NGHIÊN CỨU HỆ THỐNG THỰC NGHIỆM VÀ GIẢI PHÁP MỚI ĐỂ TIẾT KIỆM NĂNG LƯỢNG CHO CHUYỂN MẠCH OPENFLOW

Tran Hoang Vu[1], Pham Ngoc Nam[2]
[1]The University of Danang, College of Technology; Email: tranhoangvu_university@yahoo.com.vn
[2]Hanoi University of Science and Technology; Email: nam.phamngoc@hust.edu.vn

Abstract - Improving energy efficiency of the switch is becoming an increasingly important research topic, motivated by the need to reduce energy costs for Data center. In order to orient the way of cutting power consumption on NetFPGA switch [11], we have made an experiment to determine how much energy consumed on each part. In this paper, we first outline our experimental setup and methodology, and then describe detailed figures of components. That helps us to have better decision of which part should be cut down the power consumption, e.g. Ethernet chip and FPGA chip,etc. Based on these results, we propose new method for energy efficient OpenFlow switch on NetFPGA platform. Experimental results demonstrate an excellent energy saving according to different working modes, and the designed system is feasible for the switch to save energy.

Key words - Openflow network; NetFPGA; data Center; Openflow switch; low power.

Tóm tắt - Nâng cao hiệu quả năng lượng của các thiết bị chuyển mạch đang trở thành một đề tài nghiên cứu ngày càng quan trọng, được thúc đẩy bởi sự cần thiết giảm chi phí năng lượng cho Trung tâm dữ liệu. Để định hướng cách cắt giảm tiêu thụ năng lượng trên chuyển mạch NetFPGA [11], chúng tôi đã thực hiện một thí nghiệm để xác định bao nhiêu năng lượng tiêu thụ trên mỗi phần. Trong bài báo này, trước tiên chúng tôi phác thảo thiết lập mô hình thực nghiệm và phương pháp, và sau đó mô tả số liệu chi tiết của các thành phần. Điều đó giúp chúng tôi để có quyết định tốt hơn phần nào nên được cắt giảm điện năng tiêu thụ, ví dụ như chip Ethernet và chip FPGA.v.v… Dựa trên kết quả này, chúng tôi đưa ra giải pháp mới nhằm tiết kiệm năng lượng cho chuyển mạch OpenFLow trên nền tảng NetFPGA. Kết quả thực nghiệm chứng minh sự tiết kiệm năng lượng tuyệt vời theo các chế độ làm việc khác nhau, và hệ thống thiết kế có tính khả thi cao cho chuyển mạch nhằm tiết kiệm năng lượng.

Từ khóa - mạng Openflow; NetFPGA; trung tâm dữ liệu; bộ chuyển mạch Openflow; giảm công suất.

1. Introduction

Energy consumption in infrastructure of information technology is a pressing concern. Data centers currently consume about 31GW and energy of global data center increase by 19% in 2012, according to the results of a survey on industry conducted by global Datacenter-Dynamics [1]. Many system components in the data center contribute to the overall power consumption, including servers, storage, networking equipment, power supplies, cooling, etc, in which the network devices accounted for to 20-30% of energy consumption [2]. Energy costs for data centers accounted for 44% of total operating costs [3]. At the same time, along with huge energy consumption, the data center also discharged large amounts of CO2. Therefore, the issue of network energy efficiency is receiving considerable attention [4], [5], [6], and some novel hardware devices enabling different power states are promising [7]. However, the results in [4] [5] [6] are not optimal energy consumption of the switches, when input traffic reduced or no passing traffic. In [8] 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 [9] for power aware networking research. In [10], we have proposed a power aware OpenFlow switch extension which enables energy saving in data centers. Based on the results in [8] [10], we developed a measurement system used PCIEXT KIT-64UB. With the measurement results obtained, we propose a solution to save power consumption of switches. The main contributions of our work are the following:

- We built the energy measurement system for switch by software and hardware.
- We extend OpenFlow protocol includes new message allows controlling OpenFlow Switch to change the link rate of switch in different modes of bandwidth.
- We designed a software controller to receive the commands from NOX and control change of bandwidth on each port of the switch to save energy in case of low flow or no passing traffic.

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 describes the design control software. 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.

2. Empirical Model Measuring the Energy Consumption of Switches

Our experimental setup contains four parts: the NetFPGA-1G switch, 4 PC-hosts and an NOX controller connecting to switch, an extension board and our own power calculator and a high-fidelity oscilloscope for power measurement, each described in turn next:

- **Switch:** We use a NetFPGA revision 3 board having four 1Gbps Ethernet ports. The gateware on our

NetFPGA is the modified reference switch (based on the reference openflow_switch.bit, v1.0.4) [12]. We have added and removed code to make the switch run with or without some blocks. The host computer is running CentOS version 6.3, and NetFPGA driver version 3.0.1 taken from the NetFPGA website.

- **Power Measurement:** We notice that if we measure the power consumption of NetFPGA by including whole host PC, we will get a wide range of values, and of course, that numbers are untrusted. To make accurately isolate the power consumed by the NetFPGA board alone, we mount the NetFPGA card on an UltraView PCI Smart Extender PCIEXT-64UB card [13] which has break-out test-points for measuring current draw on the 3.3V- and 5V-voltage supply pins. These test-points are connected to our own power calculator board that display total power of NetFPGA with the precision of 1mW. We also use a TDS2040C Digital Oscilloscope (Tektronix) to verify the displaying result.

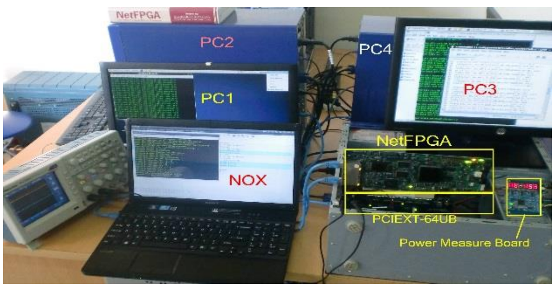

Figure 1. *The experimental setup with NetFPGA switch, an extender board, NOX controller and host PCs*

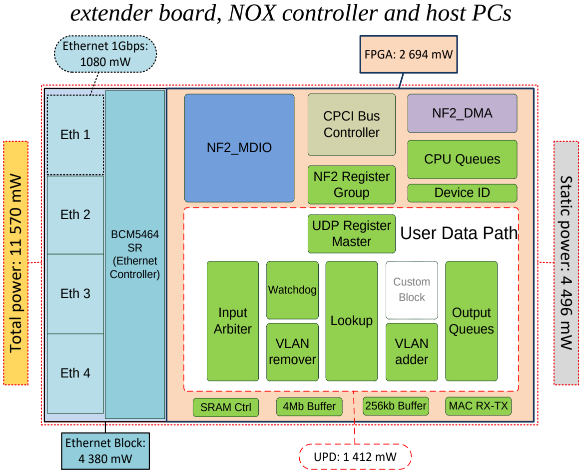

Figure 2. *Power profile of NetFPGA-1G Board*

The measurements consist of static power and dynamic power of NetFPGA board. First, the NetFPGA is not configured, and there is not any Ethernet cable connected to switch. In this case, the result on power measure boars (**4496 mW**) is the static power or minimum power consumed. If we can drop down the power of switch, we would not reduce under this level. With each additional 1Gbps Ethernet link cable plugged into switch, the power increases **1080 mW**, and with all ports filled, the total power consumption is Ethernet block is **4380 mW**. Then, configuring the FPGA to run as OpenFlow switch, and keep four 1Gbps ports at about 90% utilization of link capacity, we finally get the maximum power consumed. The highest number is **11570 mW**. To dig deeply into structure of routing core, we disable some functional blocks to get details about the power of FPGA chip. We isolate the User Data Path (UDP) module and rebuild the OpenFlow switch, the decreased number is power of UDP and its figures is **1412 mW**. Other blocks draw about **1282 mW** of power consumed.

Table 1. *Detailed power profile of NetFPGA-1G Board*

<table>
<tr><td colspan="4">Total:
11570 mW</td></tr>
<tr><td rowspan="3">Static: **4496 mW**</td><td colspan="3">Dynamic:
7074 mW</td></tr>
<tr><td colspan="2">FPGA:
2694 mW</td><td rowspan="2">Ethernet:
4380 mW
(1080 mW/Port)[a]</td></tr>
<tr><td>User-Datapath:
1412 mW</td><td>Other blocks:
1282 mW</td></tr>
</table>

Based on the results shown Figure 2, That helps us to have better decision of which part should be cut down the power consumption, e.g. Ethernet chip and FPGA chip.

3. Solution to Save Energy Consumption of OpenFlow Switch

Through the distribution of energy consumption in switch (Figure 2), we propose the solution to saving energy consumption by changing the the link-rate or turning off the port of switch.

3.1. Communication between NetFPGA with Ethernet.

As can be seen on Figure 3, a complete structure of NetFPGA switch has two lines that connect Ethernet module to FPGA chip: 1) Exported clocks and input/output queues, 2) Direct control signals via slave registers using in MDIO protocol [14]. Two main blocks connected to Ethernet modules are described as following:

- User Data Path (UPD) block is a component of NF2_CORE including an OpenFlow router with MACs (Media Access Control) and four pairs of Tx-Rx (Transmit- Receive) directly connected to the physical port
- Management Data Input/Output (MDIO): This module monitors state of four Ethernet ports and set up their configurations based on control messages via MII Registers [15] that contain all setting used in runtime of ports. Each Ethernet port has its own separated MII Register; therefor it can be controlled individually with others. In this paper, we use MII Register to controller turn on/off ports, and set them to run at various bandwidth levels.

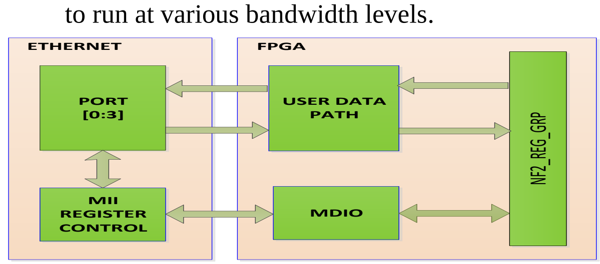

Figure 3. *Diagram of the NetFPGA system*

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.

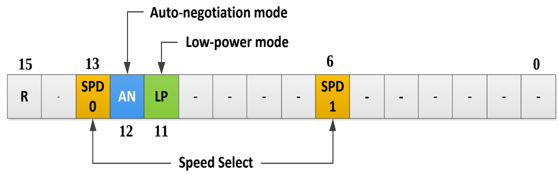

Figure 4. *Special functional bits on MII register*

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.

3.2. Designing a New Software Controller for Openflow Switch

The Ethernet ports consume one-third energy of the total power (Figure 2) if it is set at 1Gbps of bandwidth; however, they can be reconfigured to operate in lower power mode such as 100Mbps, 10Mbps and Idle. Therefore, as a given solution, we have built control software shown on Figure 5 to receive control commands from NOX or POX via PCI bus to reduce energy as following methods:

- Control on/off state of each Ethernet port to save energy on switch using a specific routing algorithm that will determine which link should be powered down when no traffic sent on that link. In other word, two ports locating at headings of link are set to idle mode.
- Change the link speed to one of three states: 10Mbps, 100Mbps, 1Gbps. NetFPGA switch is activated in the default mode at 1Gbps of bandwidth on each port. In case of lower packet load on port, for instance, 90Mbps, we can reduce the capacity of link to 100Mbps. Because the modulations used in Gigabit mode and 100Mbps mode are different to the other, the power consumption for those modes also have a gap between them. Experimental results show that NetFPGA Ethernet port consumes about 52mW when operating at 10Mbps, 112mW at 100Mbps and approximately 1000mW at 1Gbps.

From what mentioned, we have built the software in the control mode based on the change of status of MII Control Register. MII_CONTROL_REGISTERs are defined in IEEE 802.3 standard, implemented on BCM5464 IC, and also declared in the file reg_defines_open-flow_switch.h with structure shown on Figure 4. Each port has a separated register to update control signal individually. However, there is no way to communicate with those registers due to limitation of MDIO protocol implemented on hardware, so software controller which can send control messages over status register is a better method to approach desire goal.

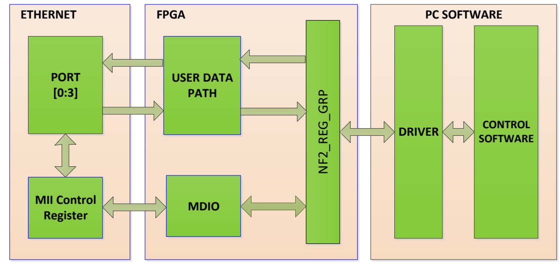

Figure 5. *Soft-controller for Openflow switch*

When receiving the request messages from the NOX or POX controller, switch checks current values in the MDIO_PHY_CONTROL_REG registers to get the status of the device, then writes the new values into the control registers for each port. Diagrams on Figure 6 and Figure 7 show how to change port state to control each port: turn on/off or set the link speed to 10Mbps, 100Mbps, and 1Gbps.

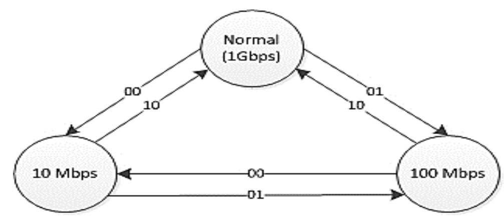

Figure 6. *Diagram of changing of speed link rates with 6th and 13th bits*

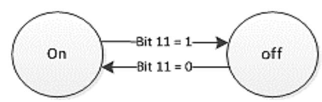

Figure 7. *Turn on/off port using the 11th bit*

The Register Addresses of four Ethernet ports start at 0x440000 and increase by 0x000080 for each (Table 2). However, we added new registers holding port code to locate which port being controlled quicker and easier. Four states of port also are coded as Message codes listed in Table 3. In fact, message code is target value of destination register. Hardware module, when receiving message code, operates bitwise functions to set MII register quickly.

Table 2. *Address of the register to control Ethernet's ports*

Port	Address	PortCode
Ethernet0	0x440000	0x0001
Ethernet1	0x440080	0x0010
Ethernet2	0x440100	0x0100
Ethernet3	0x440180	0x1000

Table 3. *The code for the link rate control*

Message	MsgCode
OFF	0x0900
S10	0x0100
S100	0x2100
S1000	0x0140

4. Extend Openflow Standard

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

OpenFlow messages are 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. A new instruction is:

- OFPT_PORT_MOD message:

  Type of message: Controller to Switch

  Length: 32 Bytes

  Functions: Configure state 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 Figure 8. 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.

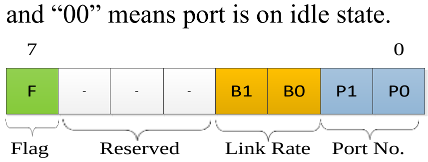

Figure 8. *Link state field*

Algorithm shown on flowchart in Figure 9 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 instructions sent to it. As soon as receiving new operating mode of ports, switch changes the limit bandwidth on each port and set the corresponding value given. It is notable that before turning port to idle mode, port must be sure that there is no packet left in its queues; therefor, a queue monitor is used to check queue empty or not.

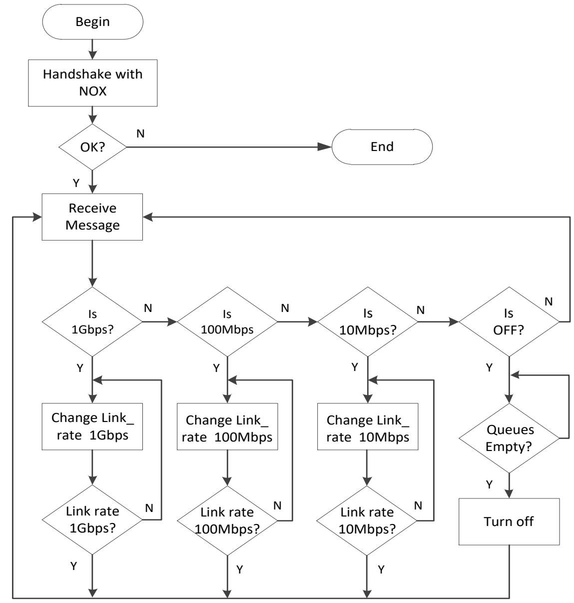

Figure 9. *Communication between NOX and Switch*

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. A complete test-bed can be seen on Figure 1.

We also used 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 capture all packets forwarded by Openflow switch. Test-bed’s model can be seen on Figure 10.

Firstly, we measure the power consumption of switch with different bandwidth: 1Gbps, 100Mbps, 10Mbps and the off state on 4 ports. Based on the results shown in Table IV, we can see that the energy savings 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.

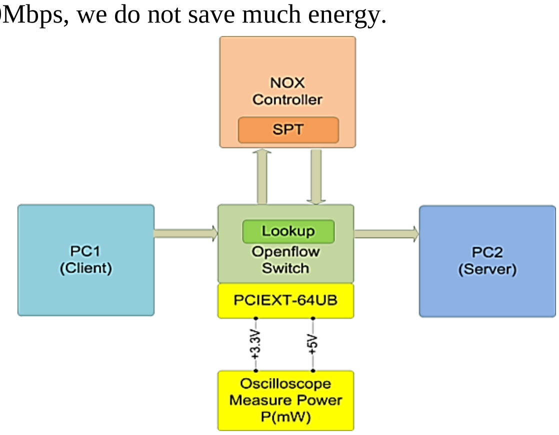

Figure 10. *Testbed for power consumption measurement*

Table 4. *Experimental results of Ethernet controller*

Mode	Bandwidth (Mbps) on 4 ports	P(mW)	P_{saved} (mW)
1	1Gbps	11576	0
2	100Mbps	7520	4056
3	10 Mbps	7284	4292
4	Off	7190	4386

Based on the results shown Figure 11, we can reduce about ***35.0%*** of power consumption when changing from 1Gbps down to 100Mbps of bandwidth. In addition, that number can be up to nearly ***37.9%*** if we turn off all ports of the switch. Therefore, we can change the status of each port according to their traffic in order to save energy.

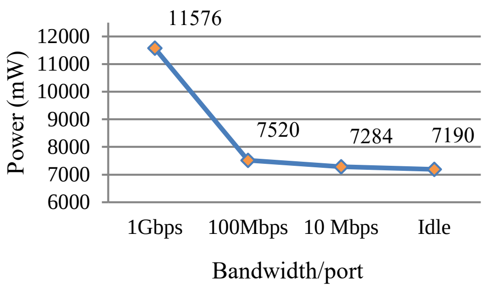

Figure 11. *Power consumption depends on bandwidth*

5. Conclusion

In this paper, we have given the power measurement of NetFPGA Switch. Thereby, we know exactly the power consumption of each block in OpenFlow switches. From there, we give the solution to save energy by designing the Software Controller on OpenFlow Switches to change the link speed on each port, which can save energy consumption of the switch in order to save energy in data centers

Based on this paper, in the future, we will propose the energy-saving status for OpenFlow Switch such as Low-power mode and Sleep mode.

REFERENCES

[1] http://www.datacenterdynamics.com/focus/archive/2011/09/global-data-center-energy-use-grow-19-2012

[2] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, N. McKeown,"Elastic tree: *Saving Energy in Data Center Networks*,USENIX NSDI, April, 2010

[3] U.S. Environmental Protection Agency's Data Center Report to Congress, [Online]. Available: http://tinyurl.com/2jz3ft

[4] 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.

[5] P. Barford, J. Chabarek, C. Estan, J. Sommers, D. Tsiang, and S. Wright, Power awareness in network design and routing, *Proc. of IEEE INFOCOM 2008*, Phoenix, USA, April 2008, 2008

[6] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, A power benchmarking framework for network devices, *Proceedings of the 8th International IFIP-TC 6 Networking Conference*, NETWORKING '09, (Berlin, Heidelberg), pp. 795–808, Springer-Verlag, 2009.

[7] "Ciscoenergywise,http://www.cisco.com/,"

[8] 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, *Proceeding of the 2012 International Conference on Advanced Technologies for Communications (ATC 2012)*, pp. 309-313. Hanoi, Vietnam

[9] Netfpga gigabit router. [Online]. Available: www.netfpga.org

[10] Tran Hoang Vu, Tran Thanh, Vu Quang Trong, Pham Ngoc Nam, Nguyen Huu Thanh "NetFPGA Based OpenFlow Switch Extension for Energy Saving in Data Centers" in REV Journal on Electronics and Communications, Vol. 3, No. 1–2, pp.77-84, January – June, 2013

[11] NetFPGA Gigabit Router, www.netfpga.org.

[12] OpenFlow Switching Reference System, git://gitosis.stanford.edu/ openflow.git.

[13] Ultraview PCI Smart Extenders, www.ultraviewcorp.com/display product.php?part id=4&sub id=2.

[14] 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

[15] http://wiki.netfpga.org/foswiki/NetFPGA/OneGig/Releases

[16] http://www.openflow.org/documents/openflow-spec-v1.1.0.pdf

(The Board of Editors received the paper on 07/03/2014, its review was completed on 31/03/2014)