Research on TCAM-based OpenFlow Switch Platform

Fei Long, Zhigang Sun, Ziwen Zhang, Hui Chen, Longgen Liao
School of Computer
National University of Defense Technology
Changsha, Hunan 410073, China

***Abstract*—OpenFlow is a new type of switch model in rapid development. This model supports the control of external policies to network process. Multi-table and multi-item group matching is the challenge in Openflow platform. Three-layer implementation model of Openflow switch is proposed in this paper. Openflow multi-table switching technique based on TCAM is studied, Iincluding the model of the multi-table process in Openflow and TCAM performance model. TCAM-based Openflow forwarding performance is analyzed under the above model.**

Keywords-component; OpenFlow; multi-table forward; TCAM;

I. INTRODUCTION

As we know Internet is experinceing revolution raised by the OpenFlow technology, with the innovative network concept to solve the new problems of the current network. It is recognized by many prestigious meetings and magazines as one of the future technology. Its core idea is very simple which transforms Switch/router control packet process into an independent process by OpenFlow Switch and Controller respectively. Flow processing is the core of the OpenFlow switch. It maintains a flow list in the network equipment to and only forward according to flow table. Its generation, maintenance and distribution is completely realized by the outside controller. These flow tables are not IP tuples as usual. In fact, the definitions of OpenFlow 1.1 include 15 key words, such as the port, VLAN, L2 / L3 /L4, etc. Every field could be wildcard, and network users can decide the granularity of flows. For example, users can route according to the destination IP. Then only destination IP field is effective and other fields are all wildcard. OpenFlow is the first practical method for software define network (SDN), which can solve all kinds of problems rapidly caused by SDN, and users can define flow by themselves and choose the path without caring the underlying hardware. OpenFlow takes back flows from all kinds of infrastructure (switches and routers) and then return them to network owners, individual users or applications. Such a feature allows users to plan their path strategy, such as find available bandwidth, fewer delayed or congestion and less hop path.

IBM and HP produced the Openflow switches. However, these switches are based on traditional switch chip to implement. Although it supports Openflow protocol, it's hard to support the 15 fields matching in data forwarding plane. How to support flow matching based on multi-item groups is always considered to be a challenge in the realization of OpenFlow switch. This paper presents a three-layer realization model of Openflow switches, TCAM-based multi-table forwarding. The Openflow multi-table description and the performance analysis of TCAM lookup function is proposed. And Netlogic's Ayama20000 TCAM is used as an example to anayze the performance of Openflow switch.

II. THE MODEL BASED ON THE DESIGN AND IMPLEMENTATION OF OPENFLOW

A. The design of three-layer model

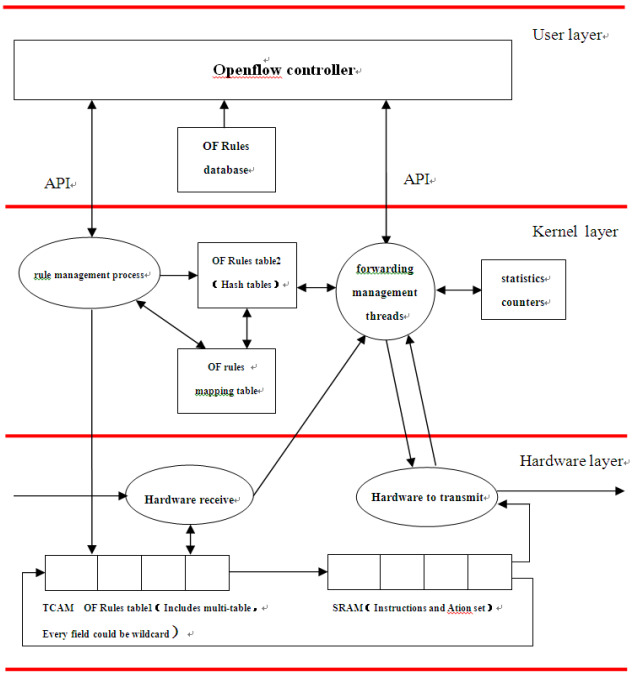
After the release of version 1.1 specification, Openflow has increased multi-table, the port group, scalable matching and other new features compared to before, but based on the new specification , less switch models can be realized. According to the characteristics of the new specification, we have designed a three-tier structure based on hardware layer, core layer, user layer to realize. Hardware layer uses Ternary Content Addressable Memory and related SRAM to realize to lookup ACL rapidly and to routing table. Realize accelerating logic functions by FPGA chip in hardware receiving and transmitting, meanwhile, implement packet analysis in hardware receiver modules by which to lookup hardware or software according to the rules of OpenFlow.

Core layer sets five modules, including rule management process, OpenFlow rule table, OpenFlow rules mapping table, forwarding thread management, statistical counters. Rule management process primarily completes controller interaction by API and the user layer. The OpenFlow controller needs to convection operating table to the hardware layer and the layer OpenFlow rule table update command to send changes to achieve such changes on the table operating. In core layer, OpenFlow rule table uses Hash algorithm to achieve fast fifteen group without wildcard to lookup table; OpenFlow rules mapping table is used to back up the data stream table, as well as to achieve some extensions. Such as timeout updating,etc. Management thread is mainly used to forward the data stream passing in three layer passes; Statistics counter is to achieve OpenFlow specifications for each flow, flow table and port, etc. Set the counter on this layer is due to the consideration of the large amount of storage space and resources consumption, etc. The main achievement is the user-layer OpenFlow controller, users can change switch configuration by controller. These three-layer models can not

 978-1-4673-0199-2/12/$31.00 ©2012 IEEE

only achieve fifteen fields to complete the current matching, but also can support the current use of the normal rules of network data packets. It can be a very good level of interaction achieved through the core layer configuration wildcard OpenFlow fields look-up table, MetaData definition of other functions.

We can also configure the hardware layer over TCAM with the Aging (Aging) combined with the rules of the core layer mapping table to achieve a better standard of OpenFlow hard timeout and idle timeout.



B. *Three-layer model of the workflow*

The Three-layer model of the workflow in particular:

1. After receiving a packet from the hardware,then looking up,analysising table to realize matching (look-up table to achieve a wildcard, and multi-table lookup), if the rule matches from the hardware forwarding the issue, and sent to the forward management to update the counter.

2. Through the layer management of the core layer thread to OpenFlow rule table forwarding rules in Table 2 (without a wildcard hash to look-up table). If the rules match, issue from the hardware forwarding process, and sent to the forwarding management to update the counter.

3. If all failed, OpenFlow rule table realizes the requests management with Openflow controller of user layer by forwarding management thread.

4. The controller writes rules mopping table to rules management of OF kernel via API, then writes into TCAM to realize mopping of TCAM and directly to OF table 2. mapping table completes request and completion of controller by rules management.

5. On user layer, set up a ruler database of Openflow to backup and modify configuration.

III. TCAM PERFORMANCE ANALYSIS AND MODEL

A. *Abbreviations and Acronyms TCAM and OpenFlow description*

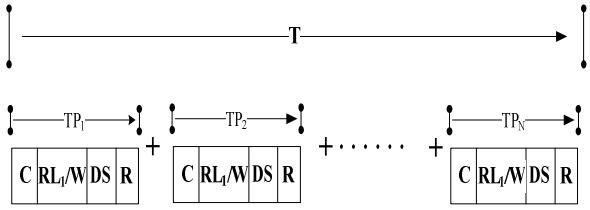
The following table defines the parameters associated with TCAM character description.

Symbolic name	units	description
V	-	TCAM rate per second
T	Ns	clock cycle
TP	Ns	Processing time for a single table
C	Ns	TCAM share of beats in command
W	Bit	TCAM write the bit width of each shot number
R	Ns	SRAM share return the number of beats
B	Bit	TCAM width of the base to process packets
PE	-	The number of packet processors

The following table describes the use of multi-table of OpneFlow: Suppose OpenFlow switches has <RN1,RN2~RNn> tables, each table has three subsets of attributes <RLi,RNi,RMi>, RLi means the rules of length, RNi means the number, RMi means the rules whether contains the mask match, 1 means including, and on contrary for 0.

Symbolic name	units	description
RNi		The number of table
RNn	—	TCAM table in the n
RLi	Bit	The length of the table
D	ns	System delay time
DS	ns	Single table system delay
E	—	Resource utilization
N	—	The total number of table

The time relationship between the characters:



B. *TCAM Performance Analysis*

As for the lookup process of hardware, by deriving the performance of TCAM can get the forwarding rates and the total number of resource occupying of different tables. The TCAM with maps to TCAM, the length are RL ¬ 1, RL2 ~ RLn for each. The clock period is T, Then we can get the rate of per second by formula.

$$V = \frac{1/T}{\sum_{i=1}^{n}(C + \left\lceil \frac{RL_i}{W} \right\rceil + R)}$$

We can multiply the table's length RLi with its number RNn to get the total numbers of the occupying resource rate of inner TCAM.

$$E = \sum_{i=1}^{n} RL_i * RN_n ; (E < N)$$

C. TCAM-based packet processor set

Each TCAM has a different type of bus bandwidth, if we want to make full use of TCAM, its bus can't be idle. We need to calculate the delaying time of TCAM, then to design a reasonable PE according to delay. In order to solve the problem, we designed a general TCAM model. First, get a general system delay. Second, we can get the entire processing time that message through TCAM.

$$DS = 2^{\log_2 \frac{RL_i}{B}}$$

Secondly, we can get the message through the entire TCAM processing time required:

$$TP = \sum_{i=1}^{n} (C + \left\lceil \frac{RL_i}{W} \right\rceil + DS + R)$$

We set a period : a, then we get the following formula:

$$PE = \frac{Ta}{\sum_{i=1}^{n} (Ca + \frac{RL_i}{W} a + DSa + Ra)/n}$$

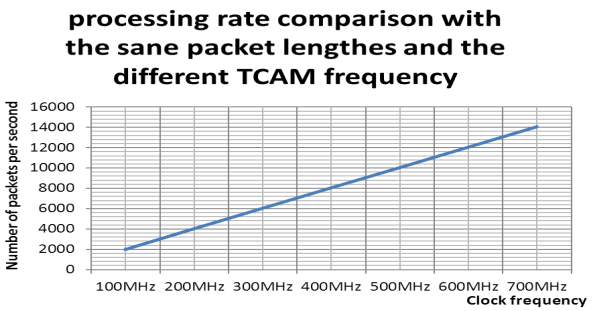
Considering the use of PE, we need to select the operating mode is linear or concurrent. If you choose a single node in the linear mode, it's easy to cause a bottleneck in the whole process when a single node makes a mistake. If use concurrently, the number of the total bandwidth should less than chips'.If QDR-II of TCAM of Ayama20000 supports 200MHz, the frequency can not exceed 200MHz.

D. The parameter calculation of Ayama20000

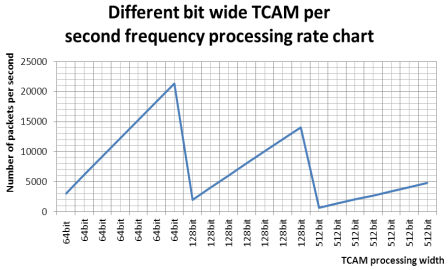
There are four tables in Openflow switch: table T1 = <128,8 K, 1>; table T2 = <64,8 K, 1>; table T3 = <512,4 K, 1>; Table T4 = <512,2 K , 1>, TCAM in Ayama20000 series writes width W = 32, C = 1, R = 1, DS = {2,4,8,8}, put data in formula, the table as following:

Table No:	V: TCAM rate per second	T: the entire clock cycle
T1	4019.94	65536
T2	6103.52	49152
T3	2712.67	106496
T4	5425.35	53248

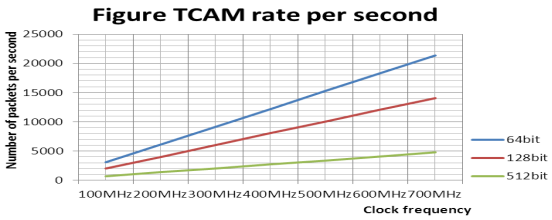
We use 128bit packet processing width with different clock cycles to obtain the following graph:



with 64bit, 128bit, 512bit packet processing width and the same clock cycle, as following graph:



with 64bit, 128bit, 512bit different packet width of three tables and the clock cycles of 100MHz ~ 700MHz to calculate,



should appear outside of the quotation marks. A parenthetical phrase or statement at the end of a sentence is punctuated

IV. Design and Implementation of TCAM-based

A. The TCAM-based establishment of multi-table

Ayama 20000 contains a core with 9M or 18M. The core supports 256K72-Bit18M, 128K72-Bit 9M. Each TCAM has 72bit data fields and 72bit mask field. Every core allows a variety of table width and size. To complete OpenFlow with 18M of TCAM, they can be 512K, 256K, 128K, 64K and 32K with the width of 32bits, 72bits, 144bits, 288bits and 576bits width. Compare with OpenFlow, its fields contain 356bit.hower, in the flow table, including 15 fields, two-timeout bit, group, port and queue, which means position and handle position with 576bit. The Ayama 20000 have two QDR™SRAMs, we can store instruction set in OpenFlow's management. After the rule corrects, then point to packet operating rule in SRAM. If it designed by OpenFlow switch which contains the flow of OpenFlow, furthermore, it can operate the packet that widely used. So we can set the various types of tables according to the width of 72bits, 576bits, 144bits, 288bits of 64K, 8K, 32K, 16K respectively.Design and Implementation of TCAM-based.

B. The TCAM Aging-based Timeout achievement

TCAM Aging helps to track the currently used table. Update the flow tables which over the flow table. Ayama 20000 supports aging with 256K at most, these tables may exists in one, two, or four lookup tables, which have an aging memory module to store aging information. The module is partition SRAM of 4K *64bit. Every bit in it corresponds to a lookup table to store aging information. Each loopup operation correspond to a bit and the aging bit mark as visited. The packet processor must read periodically aging memory entries

to determine which tables are not be visited ternatively recently. Such an information leads these forwarding entries to fail. The aging table devids into two modles:first, the singlebuffer mode which procceses the same time with rading aging mechanism. The model has two settings. One reads information for a period of time, and then stop to age. Then go on reading the whole table. Second,the double buffering mode. Table A is devided into A, B parts. To complete reading and aging alternately at the same time by A,B table reading each other. But the modle must be limited in 32K.

OpenFlow are Idle timeout and Hard timeout, their relative setting as following: if the idle timeout configured, the idle timeout setting will age without receiving dataflow. If Hard timeout sets, it ages, no matter received the data flow or not. If both set, the flow will age without data flow via the time of Idle timeout or aging after Hard timeout, no matter which one gets first. If neither sets, the table will be effective permanently without overtiming, but you can remove list item by the data packets of OFPFC-DELETE.

V. DESIGN CONCLUSION

The core of OpenFlow completes organization by the flexible management of the table to realize various data packets rapidly forward. Hope to create a better OpenFlow switch with an idea combination between the model and TCAM.

REFERENCES

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J.Turner.OpenFlow: Enabling innovation in campus networks. SIGCOMM Comput.Commun. Rev.,38(2):69–74,2008.

[2] OpenFlow white pape, http://www.OpenFlow.org/

[3] OpenFlow Consortium. OpenFlow Switch specification v1.1, http://www.OpenFlowswitch.org/

[4] OpenFlow Consortium. OpenFlow Switch specification v1.0, http://www.OpenFlowswitch.org/

[5] OpenFlow Consortium. OpenFlow Switch Specification v0.9, http://www.OpenFlowswitch.org/

[6] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker. Ethane: taking control of the enterprise. SIGCOMM Comput.Commun. Rev., 37(4):1–12, 2007. [2]

[7] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado,N. McKeown, and S. Shenker. NOX: towards an operating system for networks. SIGCOMM Comput.Commun. Rev., 38(3):105–110, 2008.

[8] Ayama20000_Device_Manual_rev_3.1

[9] TCAM_nlnse20000_rev3_3

[10] new army OpenFlow Internet innovation forum [J]. China Education Network