International Conference on Computer, Networks and Communication Engineering (ICCNCE 2013)

A Novel Descriptor-based Output Scheduling Technique forGeneral Multi-Core Network Processor

YongTing Hu, YanLongZhang,Tao Li, ZhiGang Sun
School of Computer, National University of Defense Technology, Hunan 410073, Changsha, China
yongtinghu@163.com，zhangyanlong126@163.com，sunzhigang@263.com，taoli.nudt@gmail.com

Abstract - Output scheduling is an important QoS technique for the network processor. The packet-based output scheduling technique is widely used in general multi-core network processor design due to its simplicity. However, large on-chip packet buffer required by the technique increases the chip area and production cost. Aiming at the problem, a novel output scheduling technique based on packet descriptor is proposed in this paper. The descriptor-based output scheduling technique can reduce storage space requirements without decreasing of system performance. Moreover, it can reduce the output latency of the packet traversing the network processor. Theoretical analysis shows that the proposed descriptor-based output scheduling technique has lower storage requirement compared with the packet-based output scheduling technique. The experiments based on FPGA prove the feasibility of the technique.

Index terms: network processor, output scheduling, packet descriptor.

1. Introduction

With the fast development of Internet application, network equipments need not only high processing performance, but also high flexibility. Network processor, which is a special-designed CPU for network device, emerges for that purpose. Now network processors have been widely used for the implementation of the network equipment, such as the router and gateway. Quality of service (QoS)[1] is an important function of a network processor which offers guaranteed and predicted data service to the network. Output scheduling is an important technique to grantee QoS. Output scheduling could allocate hardware source based on different users' requirement, diff-serve users when the network is congested. Many output scheduling algorithms have been used in the network processor, and a network processor could support several output scheduling algorithms. Today research on output scheduling is relatively ripe, the earliest algorithm is "first come, first serve (FCFS)", later to distinguish different priority users, "strict priority queue(PQ)" was proposed. But PQ is not fair to different users, low priority queue may not get service for a long time, so DRR[2] and WFQ[3] algorithms were proposed. To optimize the WFQ, W^2FQ[4] was proposed. For scheduling multiple resource fairly, DRFQ[5] was proposed.

Traditional network processors buffer output packets in a large off-chip memory. After its descriptor is scheduled, the packet will be transmitted from the off-chip memory to network interface. As the storage structure is different from general multi-core network processor, this technique cannot be utilized in general multi-core network processors. General multi-core network processor has become the development trend because its simplicity and flexibility.

The packet-based output scheduling technique is used in most general multi-core network processors, this technique is easy to implement and efficient. After processed, the packet is transmitted to output scheduler. After scheduled, the packet is transmitted to network interface. When there is burst traffic in the network, the scheduler needs to buffer massive packets. As the on-chip storage is limited, increasing the buffer space will gain the chip area and cost. The storage resource intensive is the biggest drawback of the packet-based output scheduling technique.

This paper presents a descriptor-based output scheduling technique for general multi-core network processor, significantly reducing the storage resource consumption of output scheduler. The technique could support all the algorithms above. Moreover, it enhances the system performance in scheduling. The second section describes the traditional network processor output scheduling technique; and the third section describes the basic idea of the descriptor-based output scheduling technique; the fourth section describes the theoretical analysis of the performance of the method; the fifth section presents experimental results and analysis; the sixth section concludes remarks.

2. Related work

The network processor NP4GS3[11] produced by IBM could support as much as 2048 flows. All the packets transmitted from processing unit are buffered in the off-chip memory called egress data store. Frames destined for the scheduler, target ports, or wrap ports have a Frame Control Block (FCB) assigned by the output scheduler. The FCB holds all the information needed to transmit the packet including starting buffer address, packet length, and packet alteration information. After the FCB is scheduled out, the packet will be removed out of the egress data store. As the difference on storage structure, this technique cannot be transferred to general multi-core network processors.

This project is supported by "Verification platform of reorganizable basic network"(2012CB315906)of the National Basic Research Program of China (973 Program)and "The innovation network architecture for the triple play" (2011AA01A10),"Research on 100Gpbs hardware acceleration engine model and mechanism" (61202483) of the National High Technology Research and Development Program of China (863 Program).

Now general multi-core network processor is the develop trend. Many companies produce general multi-core network processors such as RMI XLR[12] and Cavium OCTEON[10]. For intelligence protection, these companies maintain the confidentiality of the technical details. As far as we know, in most general multi-core network processors, after processed

by the output processor, the packet is sent to the output buffer queue according to the different priorities of packets. The output scheduler is responsible for scheduling output packets. At the same time, the output scheduler will also receive flow control information from the network interface. The flow control makes the scheduling module avoid the scheduling module excessive rate of sending packets to the network interface and the network interface buffer overflow, effectively reducing the packet loss rate.

The most commonly used output scheduling algorithms are strict priority algorithm (PQ), Weighted Fair Queuing (WFQ) algorithm and the deficit round robin algorithm (DRR). The PQ algorithm stringent sequentially output in accordance with the priority of the queue. WFQ algorithm is a queue scheduling algorithm based on time. By maintaining virtual system clock[7], according to the proportion of different flow share processing resources to calculate when a stream should be sent out. DRR algorithm is a queue scheduling algorithm based on round robin. In this algorithm each flow is assigned a transmission limit. The transmission will decrease when the packet is sent out. If the remaining limit could not support for sending the next packet, the scheduler will poll the next stream.

3. Basic idea

Many mainstream general multi-core network processors use packet-based output scheduling technique. Output scheduling shown in Figure 1 can be divided into the following steps:

1) Being processed, the packet descriptor is sent to the completed unit.
2) The complete unit sends DMA read request according to the address information carried in the packet descriptor structure.
3) Packets returned from main memory to the complete unit.
4) The Complete module sends packets to the output scheduler.
5) The scheduler sends packets to the network.

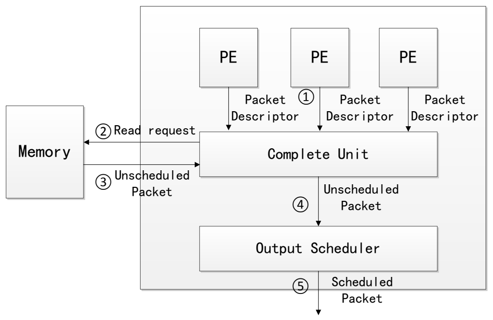

Fig 1 Packet-based Output scheduling

For large storage consumed in the scheduling module we proposed an output scheduling method based on descriptor. The output scheduling as shown in figure 2 can be divided into the following steps:

1) Being processed, the packet descriptor is sent to the complete unit.
2) The output scheduler receives the descriptor from complete unit, schedules the descriptor and constructs packet read request.
3) Packets returned from main memory are sent to the complete unit.
4) The Complete module sends packets to the output scheduler for scheduling processing.
5) After scheduled, the packets transmit from complete unit to the network.

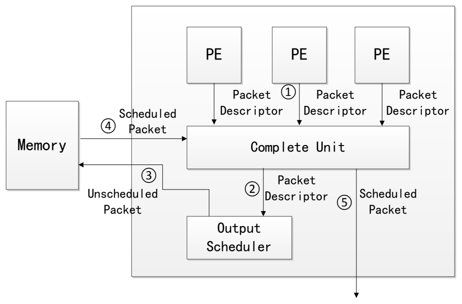

Fig 2descriptor-based output scheduling

4. Theoretical analysis

The descriptor-based output scheduling technique can be a significant reduction in hardware storage space under the premise that it will not degrade the system performance. Analysis on both storage space and processing performance will be presented in this section to prove the advantage of the descriptor-based technique.

A. Storage space

Storage space used by the output scheduler is shown in Figure 3. The storage space can be divided into the input queue and output queue. Multiple flows carried by a plurality of input queues are scheduled and transmitted to the output queue.

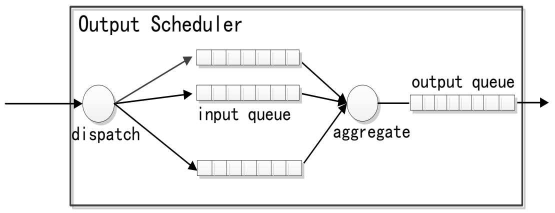

Fig3The stucture of output scheduler

According to Guido Appenzeller et al. [8], in order to make the congestion bottleneck bandwidth utilization in the network maximum, size of the node buffer(assume B) in the network need to be no less than $\mathrm{RTT} * \mathrm{L}/\sqrt{n}$. As long as the buffer is no less than this value, utilization of the bottleneck link will be close to 100% when congestion occurs, wherein RTT is the average round trip delay of all streams, L is the link bandwidth, n is the amount of flows contained in the stream number. According to this theory, as the average round-trip delay is typically 250ms, we assume that the link bandwidth 10Gbps link, there are 100 flows in total, scheduling buffer storage requires to achieve 250Mb.

Let us consider descriptor-based output scheduling technique. The general Ethernet packet size ranges from 64 bytes to 1500 bytes, we take 800 bytes as average. A descriptor only requires 8 bytes to store packet summary information required for a variety of I / O operations and queue scheduling, so that the original 250MB of storage space could decrease to 2.5MB by using the descriptor method.

Assume that the number of output queue is n, the number of packets stored in each queue is k, the output scheduler storage is B, the averaged packet length is 800 bytes. Using packet-based output scheduling technique, $B = 800kn$; using the descriptor-based output scheduling technique, the length of the descriptor is8b, then$B = 8kn$. Contrast packet-based and descriptor-based output scheduling technique, storage consumption is as shown in figure 4:

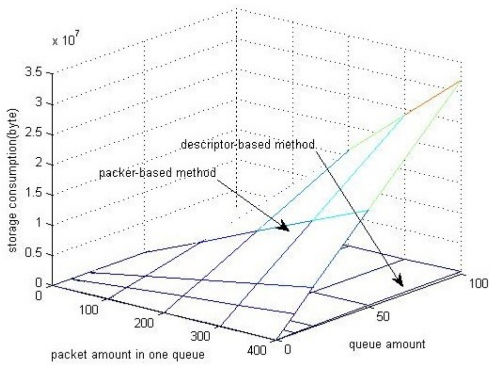

Fig 4 storage consumption of packet-based and descriptor-based metdods

B. Processing performance

From the third section in this paper, we know that packet transmission can be divided into three procedures in pipeline: transmission in the system bus; dispatching into the input queue; aggregating into the output queue, some parameter used in this section is defined as table1:

TABLE 1parameter in this section

Parameters	*remark*
$T_{desp\_in\_queue}$	Descriptor-based method's time to enqueue
$T_{pkt\_out\_queue}$	Packet-based method's time to dequeue
$T_{desp\_out\_queue}$	Descriptor-based method's time to dequeue
$T_{pkt\_out\_queue}$	Packet-based method's time to dequeue
Len_{tlp}	TLP head length
Len_{rd}	Read request packet length
Len_{pkt}	Packet length
Len_{desp}	Descriptor length

In general packet I / O technique, the transmission of a packet needs to send the read request twice (once for reading the packet descriptor, and the other for the packet). Adding the packet descriptor and the packet transmission operation, the data that needs to transfer a packet is actually:

$$4TLP + Len_{pkt} + Len_{desp} + 2Len_{pkt\_desp\_rd}$$

So the system's I/O rate ($S_{\mathrm{i/o}}$) is:

$$S_{\mathrm{i/o}} = \frac{B}{4TLP + Len_{pkt} + Len_{desp} + 2Len_{pkt\_desp\_rd}}$$

The rate of packet-based output scheduling method is:

$$\min\{S_{\mathrm{i/o}}, 1/T_{pkt\_in\_queue}, 1/\mathrm{T_{pkt\_out\_queue}}\}$$

The rate of descriptor-based output scheduling method is:

$$\min\{S_{\mathrm{i/o}}, 1/T_{desp\_in\_queue}, 1/T_{desp\_out\_queue}\}$$

As descriptor length is much shorter than packet length, so:

$$T_{desp\_in\_queue} < T_{pkt\_in\_queue}$$

$$T_{desp\_out\_queue} < T_{pkt\_out\_queue}$$

The performance of descriptor-based output scheduling technique is no less than that of packet-based scheduling method. Meanwhile descriptor-based scheduling output technique could buffer more packets, so the packet loss rate is lower, and the performance will be improved.

5. Experimental results and analysis

We use Altera FPGA (field-programmable gate array, theStratixIVEP4SGX180KF40C2) to achieve the proposed descriptor-based scheduling technique, by using the QuartusII hardware programming tools[9] to analyse the code, comprehensive analysis its occupied resources. The resource consumption is as shown in the table 2:

TABLE 2 resource consumption of descriptor-based output scheduling

Resource name	using total quantity	occupies proportion / %	Resources total
Combinational LUTs	1532	1.02%	140600
Logic registers	1093	0.77%	140600
Total block memory bits	82080	0.7%	11704320

In order to verify the feasibility of the system, we apply it to schedule the real traffic. The following figure shows the waveform in QuartusII, tdma_des_wr is the output fifo write signal. When the signal is set high, it is proved that the packets have been scheduled output.

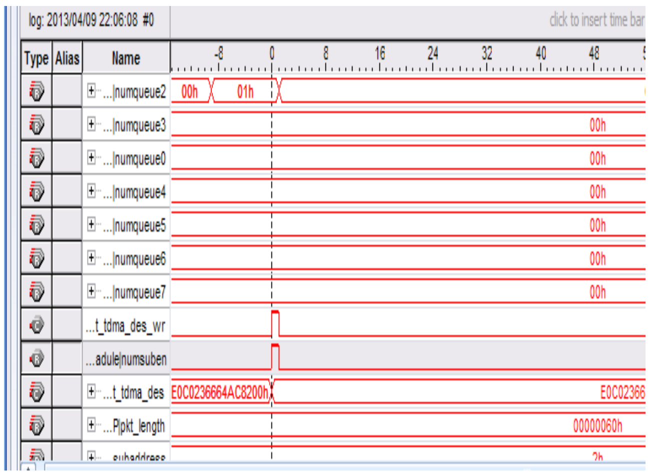

Fig 5waveformfromQuartusII

6. Conclusion

To avoid the main memory competition and decreasing the off-chip storage consumption, we proposed a descriptor-based output scheduling technique for general multi-core network processor. The technique could significantly decrease the storage consumption and will not affect the output scheduling performance. By comparing it with the packet-based output scheduling technique, we theoretically prove that it will reduce the storage consumption by 87.5% to 99.5% with the increase of packet length. When the enqueue and dequeue operation latency decreases, the output latency of the packet traversing the network processor will decrease.

Reference

[1] LiangZhaoZeng,Ngu A.H.H.,Dumas.M,KalagnanamJ,Chang H. QoS-aware middleware for Web services composition . IEEE Transactionson. Software Engineering Vol 30,pp311 – 327. May 2004

[2] M. Shreedhar, G. Varghese. Efficient fair queuing using deficit round robin. ACM Transactions on Networking, 4(3):375–385, 1996.

[3] A. Demers, S. Keshav, S. Shenker. Analysis and simulation of a fair queueing algorithm. In SIGCOMM, pages 1–12, 1989.

[4] J. Bennett , H. Zhang. WF2Q. Worst-case fair weighted fair queueing. In INFOCOM, 1996.

[5] Ali Ghodsi, VyasSekar, MateiZaharia, Ion Stoica. Multi-Resource Fair Queueing for Packet Processing. In SIGCOMM, pages 1–12, 2012.

[6] Kung, N.T., Morris, R. Credit-based flow control for ATM networks IEEE Transactions on network.Volp: 9 ,pp 40 – 48 Apr 1995.

[7] L. Zhang, Virtual clock. A new traffic control algorithm for packet switchingnetworks. SIGCOMM CCR, 20:19–29, August 1990.

[8] Guido Appenzeller,IsaacKeslassy Nick McKeown.sizing router buffer.In SIGCOMM 2004

[9] Cheng Wang,Ji-huaWu,The Design of Altera FPGA / CPLD ,2005(In Chinese).

[10] http://www.caviumnetworks.com/OCTEON_II_MIPS64.html

[11] IBM PowerNP™NP4GS3Datasheet[EB/OL]

http://www.rmiembassyus.org/