Quality of Service on Gigabit Ethernet for Event Builder

Y.Yasu[1], Y.Nagasaka[2],Yoji.Hasegawa[3], A.Manabe[1],
M.Nomachi[4], H.Fujii[1], Y.Watase[1]
[1] High Energy Accelerator Research Organization (KEK), Japan
[2] Nagasaki Institute of Applied Science (NIAS), Japan
[3], Shinshu University, Japan
[4] Osaka University, Japan

Abstract

Congestion avoidance techniques for event builders using switching network technologies is a major issue. Traffic management of the data flow is an essential point to avoid congestion. Global traffic control is one of the mechanisms to manage the traffic. It uses global information of the switching network. While, traffic shaping technique is another mechanism not to use them.

Quality of Service (QoS) is one of the solutions to avoid this congestion, because it provides the bandwidth allocation in the network. A Gigabit Ethernet technology, which is a major in the high performance network, however, has no functionality of the QoS while ATM has the functionality of the QoS on a hardware level. Recently, IP-based QoS is intensively studied in computer science. On Gigabit Ethernet with the QoS, it will be expected to avoid the congestion of the event builder traffic.

We used Alteon network interface card, which can handle large Maximum Transmission Unit (MTU) called jumbo frame. Jumbo frame improves the transfer speed. We also used Linux-based PCs for measuring the performance. We chose several parameters such as device driver parameters, MTU, chipset for PC, bus width of PCI (64/32-bit) and so on, related to the performance.

We found high performance chipset for PC and 64-bit PCI bus improved the transfer speed in TCP/IP up to 990Mbit/s with jumbo frame. We had the results that the overhead of the IP-based QoS is very small and the bandwidth allocation worked.

I. INTRODUCTION

In an event builder, event fragments from all sources are concentrated coherently into one destination via a switching network. However, commercial-available switching networks are designed for random traffic like Tele-communication data. They may not be able to handle coherent traffic like event builder. Congestion avoidance is indispensable for switch type event builder.

A way to avoid the congestion is to establish a global traffic control. A circuit switch was applied to an event builder at Fermilab [1] and KEK [2]. The data traffic is controlled by a global traffic signal. Another way is to design data flow by allocating the bandwidth for each node in a switching network. The RD31 project at CERN established a way to shape traffic on ATM network[3,4]. An other way to solve the problem is over-provisioning of the switching network, but we do not mention it.

Gigabit Ethernet is one of the technologies, which enables a high-speed transfer for the event builder. It is prefer-able to use industrial standard equipment for the event builder if possible, in order to reduce the cost of development and ease maintenance. Gigabit Ethernet is fully compatible with existing Ethernet installation, which is not only an international standard but also de facto standard. TCP/IP as candidate software for the data path also has the same advantages.

Thus, we investigated the feasibility of the QoS on Gigabit Ethernet, which is a way to design data flow by allocating bandwidth.

A. Requirements and Gigabit Ethernet with QoS

Latency of packet routing as low as possible is better for event builder, but the latency is not critical. Thus, high level protocol such as TCP/IP with QoS may be used. On the other hand, Gigabit Ethernet technology with jumbo frame was investigated [5,13] because the throughput is critical. It showed that jumbo frame doubled the transfer speed. It is expected that use of the jumbo frame works well for the event builder.

Recent Gigabit Ethernet companies guarantee "wire speed" at each port while the possibility of the congestion at burst data flow still remains within a switch. It is not clear to guarantee the speed on a coherent traffic. It is expected that Gigabit Ethernet with QoS shapes the traffic in a way to avoid congestion.

The next section describes congestion control and bandwidth allocation in Linux. 3 section describes the performance measurement of Gigabit Ethernet with QoS on PC/Linux.

II. CONGESTION CONTROL AND BANDWIDTH ALLOCATION

In TCP/IP, delayed ACK is a traffic control method to reduce unnecessary packets, by acknowledging multiple packets with a single ACK. The Nagle algorithm packs short messages to avoid congestion. Traffic control can be done by controlling the TCP sliding window size. However, it is not clear that those algorithms are enough to avoid the congestion on event builder.

Researches and developments of QoS are recently done for Internet application, in environments LAN and WAN. Internet Engineering Task Force (IETF) is working on terminology and architecture of QoS. It defined Resource ReSerVation Protocol (RSVP) [7] for multimedia application such as Voice

over IP and Video on demand. The essential point of traffic management is packet scheduling, known as queueing method. PC-based scheduling mechanism is studied in computer science [8,9]. Packet queueing disciplines are studied in techniques related to ATM in computer science. They are Class-Based Queueing (CBQ), Token Bucket Filter (TBF) and so on.

A. Terminology of QoS

We introduce elementary technologies of QoS first. There are admission control, packet classification, packet scheduling and traffic shaping. The admission control is a way to control reserving resources in a session such as RSVP. A setup protocol makes signaling on the path. Figure 1 shows Class-Based Queueing discipline. The packet classification is to classify incoming packets into groups called class.

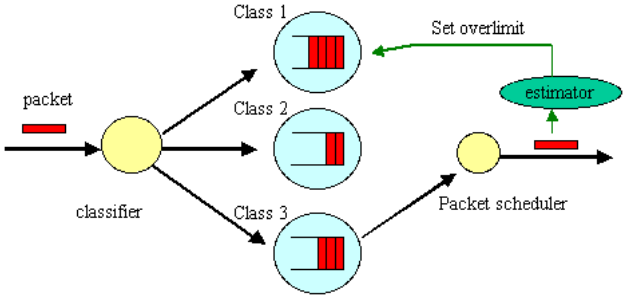

Figure 1: Class-Based Queueing

The packet scheduler is to arrange the scheduling for outgoing packets. There are many ways according to the queueing method and the buffer management. Figure 2 shows Token Bucket Filter as an example of a packet-scheduling algorithm. The outgoing packet will be sent according to the size of the token buffer and the rate. The traffic shaper is a technique to make the burst data flat.

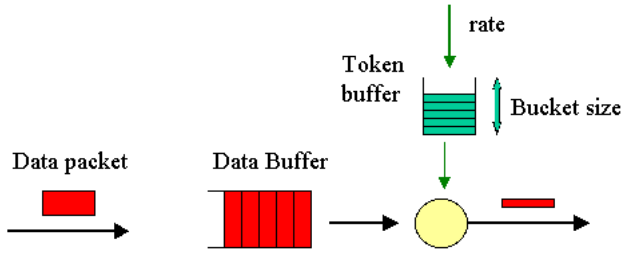

Figure 2. Token Bucket Filter

B. Traffic Management in Linux

For Linux, kernel 2.2.x and later support many kinds of QoS [10]. The developer toolkit also supplies some commands for manipulating the parameters in the kernel [11]. The typical command is called tc. Figure 3 shows a traffic control in TCP/IP for Linux. The traffic control is done only at the output queueing.

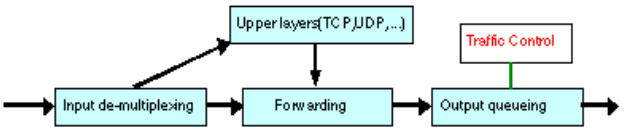

Figure 3. Traffic control in TCP/IP

The Linux kernel supports CBQ queueing discipline and TBF queueing discipline. Figure 4 is an example of queueing discipline for Linux. It defines a queueing discipline, CBQ as root with bandwidth 1000Mbit/s. When a packet comes into the queueing discipline, the packet will be sent to a classifier called u32, which classifies the packet. The packet will come into a class called CBQ, which also assigns 1000Mbit/s as the bandwidth. After the packet passes through the class, a queueing discipline called TBF will limit the transfer rate. The transfer rate will be limited to 100Mbit/s according to the rate of the token even if packet arrives over the rate.

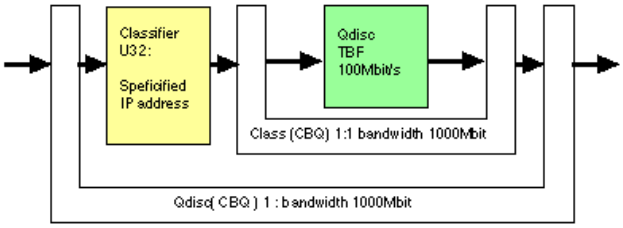

Figure 4. Queueing Disciplines for Linux

III. PERFORMANCE EVALUATION

A. Measurement

On the measurement of transfer speed, we evaluated the transfer speed on three frequencies of CPU, three types of chipset, and on 64/32-bit PCI. We also measured copy speed of memory on the PCs.

On the measurement of QoS, we evaluated 1x1, 1x3 and 3x1 systems with QoS. We used TBF in CBQ for allocating transfer rate.

Before the measurements, we tuned network driver and TCP buffer size. We evaluated TCP buffer size. The value of 528288 was best. We had evaluated Nagle algorithm [13] and then adopted Nagle algorithm for the measurement.

We used the netperf utility [14] as the measurement tool to evaluate performance at point-to-point link.

B. Setup

The configuration of the PCs is shown in Table 1. We used three types of PCs. A pair of PC733 and a pair of PC800 were used for the evaluation of transfer speed. Four PCs of PC500 were used with a Gigabit Ethernet switch for the evaluation of QoS and transfer speed. The switch we used was Alteon AceSW180 [6], which has 9 ports of Gigabit Ethernet with 8Gbit/s backplane. Linux driver for AceNIC was used [12].

Table 1. Configuration of PC500, PC733 and PC800

System	PC500	PC733	PC800
CPU type	PentiumIII	PentiumIII	PentiumIII
CPU Freq.	500MHz	733MHz	800MHz
Cache	512KB	256KB	256KB
Chipset	440GX	SuperWorks ServerSet III LE	440BX
Bus speed	100MHz	133MHz	100MHz
Memory type	SDRAM	SDRAM	SDRAM
Memory size	256MB	128MB	256MB

PCI(Bus width)	32-bit	64/32-bit	32-bit
OS version	2.2.14-12	2.2.14-12	2.2.14-5
gcc version	egcs-2.91.66	egcs-2.91.66	egcs-2.91.66
NIC	AceNIC(1MB)	AceNIC(1MB)	AceNIC(1MB)
Driver version	v0.47	v0.47	v0.47

C. Copy speed of memory

We measured the copy speed of memory on the PCs by using a Linux system call, memcpy. The speed depends on CPU speed and performance of chipset. High performance memory copy is crucial to obtain high bandwidth using TCP/IP.

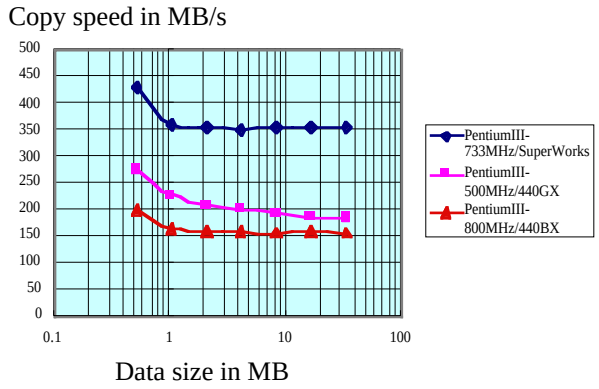

Figure 5. Performance of memory copy

PC733 was the fastest because it has a high performance chipset.

D. Tuning Network Driver

The parameters in the driver code were tuned to get the best performance. We have modified and tested the parameters with regard to coalescing. The coalescing affects the times of interrupts to CPU from the NIC. We varied the following two parameters tx_coal on the transmitting side and rx_coal on the receiving side. When the value of the parameter is small, the interrupt time increases. We tuned those parameters for normal frame and jumbo frame in Table 2.

Table 2. tx_coal and rx_coal for normal and jumbo frames

	tx_coal	rx_coal
Normal (best/default)	996/400	1000/120
Jumbo(best/default)	200/20	300/30

We varied the values of tx_coal and rx_coal parameters. In normal frame, which means MTU equals 1500, the pair of values for tx_coal/rx_coal, 40/12, 400/120, 996/1000 and 2000/2000 were chosen for the evaluation.

Figure 6. Transfer speed between PC500s in normal frame

Figure 6 shows the transfer speed on PC500s in normal frame. At the values of 996/1000 and 2000/2000, the transfer speed was better in comparison with that at default values. We chose the value of 996/1000, which is recommended by the developer.

Figure 7 shows the transfer speed in jumbo frame, which means MTU equals 9000. At values of 200/300, the transfer speed was better in comparison with that at default values.

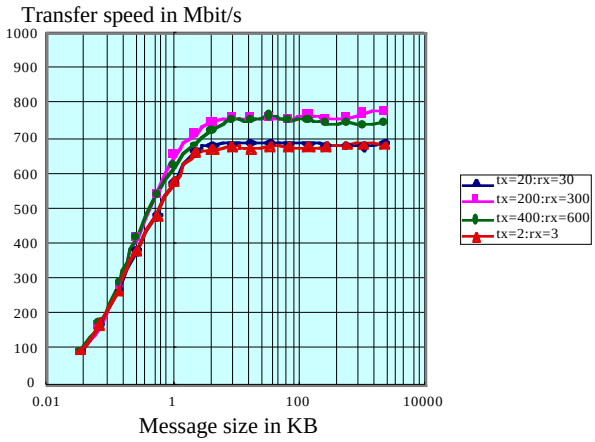

Figure 7. Transfer speed between PC500s in Jumbo frame

Thus, we chose 996/1000 in tx_coal/rx_coal for normal frame and 200/300 in tx_coal/rx_coal for jumbo frame. We confirmed that the feature was similar on PC733s and PC800s.

Figure 8 shows the transfer speed on PC500s, PC733s and PC800s in case of MTU1500. The speed did not depend on CPU speed.

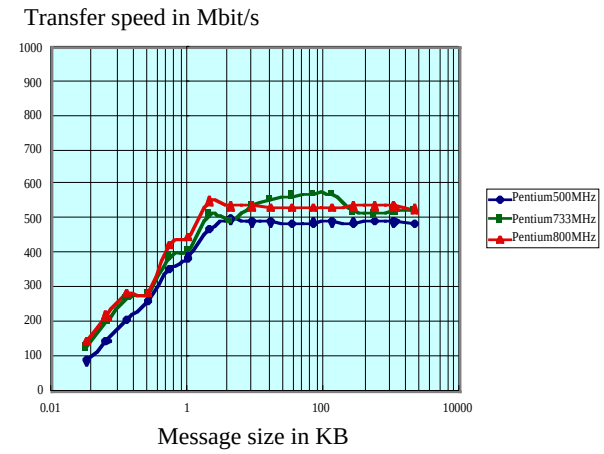

Figure 8. Transfer speed in normal frame on various PCs

However, the transfer speed with jumbo frame changed the feature. PC800 with highest performance CPU and worst chipset had poor performance while PC733 with highest performance chipset was best. Figure 9 shows the result.

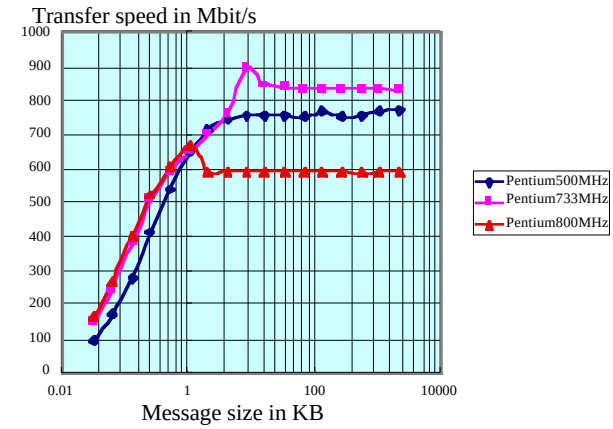

Figure 9. Transfer speed in jumbo frame on various PCs

E. 64-bit PCI vs. 32-bit PCI

We evaluated transfer speed on 64-bit PCI. Figure 10 shows the transfer speed on PC733s in cases of 64-bit PCI and 32-bit PCI. At MTU1500, the transfer speed with 32-bit PCI was similar to that with 64-bit PCI. However, the transfer speed with 64-bit PCI was better to that with 32-bit PCI at MTU9000. The speed reached to 990Mbit/s.

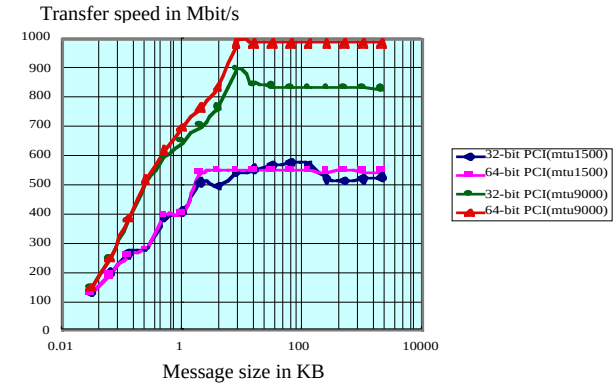

Figure 10. 64-bit PCI vs. 32-bit PCI for transfer speed

F. QoS performance on 1x1 system

The QoS was adopted at source node. The rate was assigned by TBF. With the QoS, the measured rate showed that the bandwidth allocation worked well. Furthermore, the CPU overhead of QoS was very small and nearly 1%.

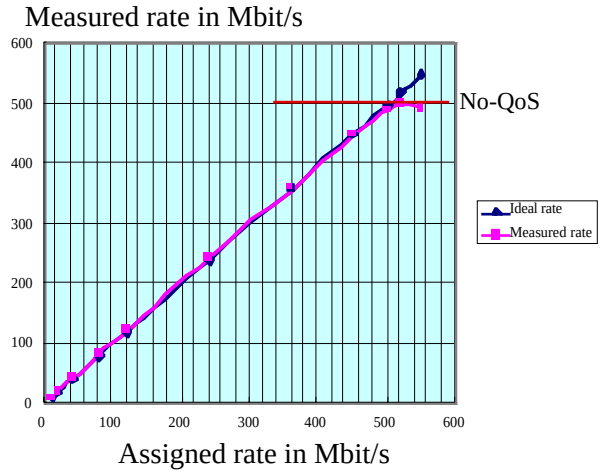

Figure 11. QoS performance on 1x1 system

G. QoS performance on 1x3 system

The rate was also assigned by TBF. Namely, TBF limits the transfer to each destination node at source node. With the QoS, the measured rate showed that the bandwidth allocation worked at small rate, but did not work well at full rate. Furthermore, the transfer speed without QoS at each destination node exceeds 200Mbit/s and the speed to each destination was nearly identified.

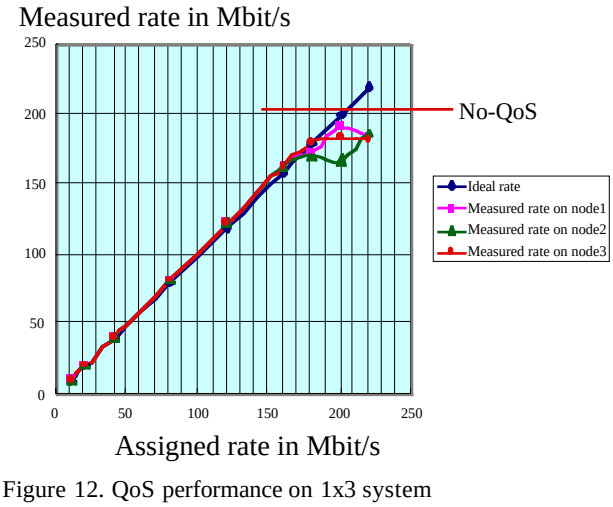

Figure 12. QoS performance on 1x3 system

H. QoS performance on 3x1 system

The rate was also assigned by TBF. Namely, TBF limits the transfer to each destination node at source nodes. With the QoS, the measured rate showed that the bandwidth allocation worked well.

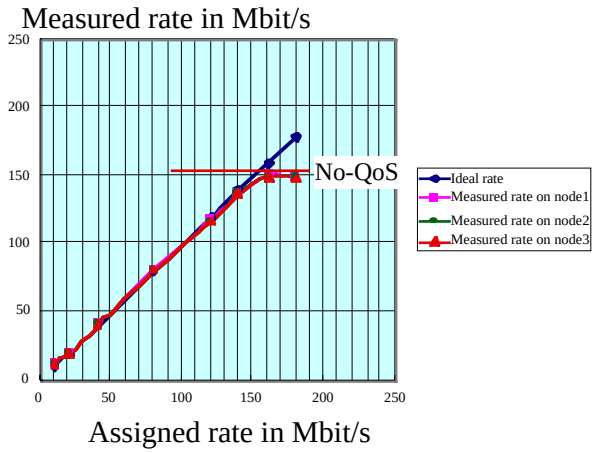

Figure 13. QoS performance on 3x1 system

IV. CONCLUSION

The traffic management of event data flow is necessary for the event builder. We investigated another solution of the traffic management using IP-based QoS on PC/Linux instead of ATM-based traffic management.

On the evaluation of basic performance with Gigabit Ethernet on PC/Linux, we tuned the network driver and then measured the transfer speed with various types of CPU and chipset, and 64-bit PCI. The result showed that the transfer speed did not depend on CPU speed in normal frame. It also showed that high performance chipset, 64-bit PCI and jumbo frame were effective and necessary to get best performance.

On the evaluation of the QoS, transfer rate assignment was possible by using tc command, but it did not make clear that the QoS on Gigabit Ethernet was effective for event builder.

V. ACKNOWLEDGMENTS

The authors wish to thank Prof. Takahiko Kondo at KEK for his support and encouragement, Dr. Hiroyuki Sato at KEK for his help, and people of Netone Systems Corporation and Alteon Web Systems in Japan for their help.

V. REFERENCES

[1] Ed. Basotti et al., A Proposed Scalable Parallel Open Architecture Data Acquisition System for Low to High Rate Experiments, Test Beam and All SSC Detectors, IEEE Trans. NS, Vol.37, No.3 (1990)

[2] Y.Nagasaka et al., Performance Analysis of a Switch-type Event Builder with Global Traffic Control System, IEEE Trans. NS, Vol.43, No.1(1996)

[3] D.Calvet et al., Evaluation of a Congestion Avoidance Scheme and Implementation on ATM Network based Event Builders, Proc. Second International Data Acquisition Workshop on Networked Data Acquisition Systems, Osaka, Japan, 13-15, November 1996, World Scientific Publishing 1997, pp.96-107.

[4] D.Calvet et al., Operation and Performance of an ATM based Demonstrator for the Sequential Option of the ATLAS Trigger, IEEE Trans. NS, Vol.45, No.4(1998)

[5] Y.Yasu at al., Evaluation of Gigabit Ethernet with Java/HORB, Contributed to the International Conference on Computing in High Energy Physics, CHEP98, Chicago, August 31 - September 4, 1998

[6] Home page of Alteon Web Systems, http://www.alteonwebsystems.com/

[7] R.Braden, L.Zhang, S.Berson, S.Herzog and S.Jamin, Resource ReSerVation Protocol (RSVP) – Version1 Functional Specification, RFC2205.

[8] S.Floyd and V.Jacobson, Link-sharing and Resource Management Models for Packet Networks, IEEE/ACM Transactions on Networking, Vol.3 No.4, August 1995

[9] K.Cho, A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers, Proceedings of USENIX 1998 Annual Technical Conference, New Orleans LA, June 1998

[10] Alexey Kuznetsov, Implementation of queuing disciplines in Linux kernel. They are available in Linux kernel 2.2.x. http://www.linuxhq.com/

[11] Alexey Kuznetsov, tc command is available. ftp://ftp.inr.ac.ru/ip-routing/

[12] Jes Sorensen, Linux device driver for AceNIC, http://home.cern.ch/~jes/gige/acenic.html

[13] Yoji Hasegawa, Yasushi Nagasaka, Yoshiji Yasu, DAQ/EF-1 Event Builder system on Linux/Gigabit Ethernet, ATL-DAQ2000-008, March 2000

[14] Netperf Home page,

http://www.netperf.org/netperf/NetperfPage.html