Stepwise Optimizations of UDP/IP on a Gigabit Network *

Hyun-Wook Jin[1], Chuck Yoo[1], and Sung-Kyun Park[2]

[1] Department of Computer Science and Engineering, Korea University
SEOUL, 136-701 KOREA
{hwjin,hxy}@os.korea.ac.kr
[2] SK Telecom, SEOUL, 110-110 KOREA

Abstract. This paper describes stepwise optimizations of UDP/IP on Myrinet. We eliminate internal overheads such as fragmentation, checksum computation, data copy, and multiple DMA initializations. In addition, we reduce the NIC overhead with a faster NIC. Stepwise optimizations clearly illustrate how much bandwidth is wasted by each overhead and what factors should be considered for designing a gigabit network protocol. As a result, we show that UDP/IP can achieve 926Mbps on 32bit 33MHz PCI platform.

1 Introduction

The physical network bandwidth is dramatically increasing in recent years. Accordingly, in order to fully utilize the bandwidth, many research groups have been trying to develop a user-level communication protocol [2, 8] or optimize a traditional protocol such as TCP/IP and UDP/IP [3, 9]. Existing works, however, are focusing on the removal of only one specific overhead, such as data copy overhead.

This paper aims to stepwisely eliminate several critical overheads of UDP/IP one by one on Myrinet [1]. This paper also demonstrates that the performance of UDP/IP is improved by reducing the overhead of Network Interface Card (NIC). As a result, we can show that the optimized UDP/IP fully utilizes the bandwidth of Myrinet. Our stepwise optimizations contribute to the understanding of how much bandwidth is encroached by each overhead and what factors cause bottlenecks on a gigabit network.

2 Optimizations of UDP/IP

The details of stepwise optimizations are described in the following subsections. Optimizations are performed using a pair of workstations equipped with an Intel Pentium III 450MHz processor and an M2F-PCI32C (LANai-4) Myrinet NIC.

* This research was supported by University Software Research Center Supporting Project from Korea Ministry Information and Communication.

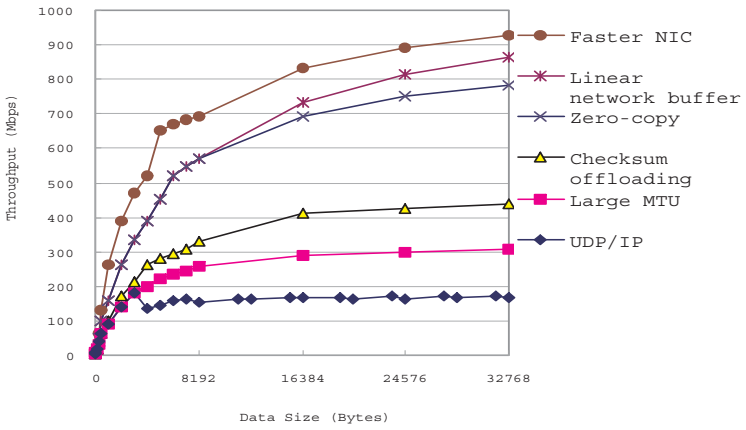

Fig. 1. Throughput gains by removing overheads one by one

Workstations are directly connected via a Myrinet cable. The kernel is Linux (kernel version 2.2), and we adopt Myrinet Software (version 3.22c) [5] for the device driver and the firmware of the NIC. In Subsection 2.4, we altered the Myrinet NIC and Myrinet Software into an M2L-PCI64B (LANai-9) Myrinet NIC and GM (version 1.4) [6], respectively. We measure the throughput using *ttcp*.

2.1 Large Maximum Transmission Unit (MTU)

A small MTU size leads to a fragmentation that results in a per-packet overhead. Actually, we observe that the (de)fragmentation overhead is roughly $31\mu s$ on the sender and $9\mu s$ on the receiver. In addition, the packet header per fragment wastes the network bandwidth. Moreover, the defragmentation routine of the Linux kernel version 2.2 performs the copy operation that moves all received fragments into a large buffer in order to merge fragments. This copy operation induces about $10\mu s$ overhead per 1KB. As a result, the throughput reaches its peak at the MTU size of 4KB while it remains slightly below the peak maximum with a larger packet size than MTU as the line labeled *UDP/IP* in Figure 1.

A notable characteristic of gigabit networks is to provide us with a large MTU size (e.g., jumbo frame of Gigabit Ethernet). In the case of Myrinet, the MTU size really has no limit. We enlarge the Myrinet MTU size large enough to support the user data up to 32KB. Figure 1 shows that the throughput with large MTU size increases continuously even for data sizes larger than 4KB.

2.2 Checksum Offloading and Zero-Copy

Per-byte overheads, such as checksum computation and data copy overheads, increase as the packet size increases. UDP performs the checksum computation

for the whole network data and also moves it between the user and kernel buffers via a copy operation. These per-byte overheads are principal factors in a network bottleneck because the packet size tends to become large in order to achieve a better network utilization.

There are some works to improve the performance of checksum computation, which achieve a significantly improved rate [4, 7]. In spite of those improvements, the checksum computation still burns the host processor.

In order to offload the checksum computation from the host processor, we adopt a hardware checksum computation. Many gigabit network NICs include the function of the checksum computation, which is generally integrated with the DMA engine. Figure 1 shows that UDP/IP without the checksum computation achieves 44% higher throughput than the UDP/IP of the improved version as described in the previous subsection.

Moreover, to eliminate the copy operation, we implement the mechanism that moves a data directly between the user buffer and NIC without copying it to/from the kernel buffer. An interesting result to be noted is that the throughput of UDP/IP without both checksum computation and copy operation is much higher than that without only the checksum computation as shown in Figure 1. The reason why the removal of the copy operation (after the checksum computation) results in a higher improvement is due to the cache effect. Because the received data is not cached, any operation to touch the data in the first time takes longer. When the checksum computation is followed by the copy operation, the checksum computation is done with the uncached data. If the checksum computation is removed in UDP/IP, the copy operation has to be done with the uncached data. So removing both of the checksum computation and the copy operation eliminate the overhead of touching uncached data.

2.3 Linear Network Buffer

The performance of DMA between the host and NIC memories is affected by the linearity of the network buffer. If the network buffer is not contiguous, NIC needs a scatter or gather to move a chunk of data. A scatter or gather is a list of vectors, each of which indicates the location and length of one linear memory segment in the overall receiving or sending request. Therefore, several vectors in a scatter or gather induce multiple DMA initializations as much as the number of vectors. In the case of traditional protocols, the kernel allocates a physically linear memory area for the network buffer. On the other hand, the zero-copy UDP/IP of the previous subsection uses the memory area in user space as a network buffer, which is not physically linear. Therefore, the network buffer crossing the page boundary leads to multiple DMA initializations.

To reduce the DMA initialization overhead, we add a system call that allocates a physically linear memory area and also maps this area into the user space allowing the application to use it as a network buffer. Figure 1 shows the physically linear buffer improves the throughput up to 10% compared with the last improvement described in the previous subsection.

2.4 Faster NIC

Another important component of an end system is NIC because of its relatively high overhead when compared with the host overhead reduced by previous subsections.

In order to reduce the NIC overhead, we adopt another Myrinet NIC equipped with a faster processor and an effective firmware. We measure the throughput of the improved UDP/IP on GM with M2L-PCI64B (LANai-9) Myrinet NIC. The GM provides a highly optimized firmware for Myrinet NIC, and the LANai-9 RISC operates at 132MHz that is 4 times faster than LANai-4. Figure 1 shows the measurement results. The maximum throughput of the present optimized UDP/IP is 926Mbps that is a remarkable performance with a traditional protocol on a 32bit 33MHz PCI platform.

3 Conclusions

We eliminate fragmentation, checksum computation, data copy, and multiple DMA initializations from UDP/IP on Myrinet in a stepwise manner. An interesting result is that we can significantly improve the performance by removing both checksum computation and data copy. In addition, this paper shows how the low overhead NIC influences the throughput. The optimized UDP/IP achieves 926Mbps on a 32bit 33MHz PCI platform.

References

1. Boden, N. J., Cohen, D., Felderman, R. E., Kulawik, A. E., Seitz, C. L., Seizovic, J. N., and Su, W. -K.: Myrinet – A Gigabit-per-Second Local-Area Network. IEEE-Micro, Vol. 15, No. 1, pp. 29-36, February 1995.
2. Dunning, D., Regnier, G., McAlpine, G., Cameron, D., Shubert, B., Berry, A. M., Gronke, E., and Dodd, C.: The Virtual Interface Architecture. IEEE Micro, Vol. 8, pp. 66-76, March-April 1998.
3. Gallatin, A., Chase, J., and Yocum, K.: Trapeze/IP: TCP/IP at Near-Gigabit Speeds. In Proceedings of 1999 USENIX Technical Conference, June 1999.
4. Kay, J. and Pasquale, J.: Measurement, Analysis, and Improvement of UDP/IP Throughput for the DECstation 5000. In Proceedings of 1993 USENIX Winter Conference, pp. 249-258, 1993.
5. Myricom Inc.: Myrinet User's Guide. http://www.myri.com, 1996.
6. Myricom Inc.: The GM Message Passing System. http://www.myri.com, January 1998.
7. Partridge, C. and Pink, S.: A Faster UDP. IEEE/ACM Transactions on Networking, Vol. 1, No. 4, pp. 429-440, August 1993.
8. Prylli, L. and Tourancheau, B.: BIP: a new protocol designed for high performance networking on myrinet. In Proceedings of IPPS/SPDP98, 1998.
9. Yoo, C., Jin, H. -W., and Kwon, S. -C.: Asynchronous UDP. IEICE Transactions on Communications, Vol.E84-B, No.12, pp. 3243-3251, December 2001.