Hoang, H., M. Jonsson, A. Larsson, R. Olsson, and C. Bergenhem, "Deadline first scheduling in switched real-time Ethernet - deadline partitioning issues and software implementation experiments," Proc. 1st International Workshop on Real-Time LANs in the Internet Age (RTLIA'02) in conjunction with the 14th Euromicro Conference on Real-Time Systems, Vienna, Austria, June 18, 2002, pp. 68-71.

Deadline First Scheduling in Switched Real-Time Ethernet – Deadline Partitioning Issues and Software Implementation Experiments

Hoai Hoang, Magnus Jonsson, Anders Larsson, Rikard Olsson, and Carl Bergenhem

School of Information Science, Computer and Electrical Engineering, Halmstad University, Halmstad, Sweden, Box 823, S-301 18, Sweden. {Hoai.Hoang, Magnus.Jonsson, Carl.Bergenhem}@ide.hh.se, http://www.hh.se/ide

Abstract

This paper presents work on a switched Ethernet network extended to allow for earliest deadline first (EDF) scheduling. We show by example that asymmetric deadline partitioning between the links of a real-time channel can increase the utilization substantially, still not violating the real-time guarantees. We also report measurements on a software implementation of the switch on an ordinary PC.

1 Introduction

An important trend in the networking community is to involve more switches in the networks (e.g., LAN, Local Area Networks) and a pure switched-based network becomes more and more common. At the same time, the industrial communication community has a strong will to adapt LAN technology (e.g. Ethernet) for use in industrial systems. The involvement of switches does not only increase the performance; the possibility to offer real-time services is also improved. Now when the cost of LAN switches has reached the level where pure switched-based networks have become affordable, the collision possibility in IEEE 802.3 (Ethernet) networks can be eliminated and methods to support real-time services can be implemented in the switches without changing the underlying widespread protocol standard.

Several protocols to support real-time communication over shared-medium Ethernet have been proposed [1] [2] [3]. However, these protocols are either changing the Ethernet standard or do not add guaranteed real-time services. Real-time communication over switched Ethernet has also been proposed (called EtheReal) [4]. The goal of the EtheReal project was to build a scaleable real-time Ethernet switch, which support bit rate reservation and guarantee over a switch without any hardware modification of the end-nodes. Ethereal is throughput oriented which means that there is no or limited support for hard real-time communication, it has no explicit support for periodic traffic so it is not suitable for industrial applications. A review of research on real-time guarantees in packet-switched networks is found in [5].

This paper presents work on a previously proposed switched Ethernet network with support for both bit rate and timing guarantees for periodic traffic [6]. Only a thin layer is needed between the Ethernet protocols and the TCP/IP suite in the end-stations. The switch is responsible for admission control, while both end-stations and the switch have EDF (Earliest Deadline First) scheduling [7]. Internet communication is supported at the same time as nodes connected to the switch can be guaranteed to meet their real-time demands when they communicate with each other. This is highly appreciated by the industry since it makes remote maintenance possible, e.g., software upgrades or error diagnostics.

The rest of the paper is organized as follows. The network architecture and traffic handling are presented in Section 2. In Section 3, asymmetric deadline partitioning is described and exemplified. Experiments with a software implementation of the switch are then presented in Section 4. The paper is concluded in Section 5.

2 Network architecture and traffic handling

We consider an example of a network with a full-duplex switched Ethernet and end-nodes. Both the switch and the nodes have software (RT layer) added to support guarantees for real-time traffic. All nodes are connected to the switch and nodes can communicate with each other over logical real-time channels (RT channels), each being a virtual connection between two nodes in the system.

In our network configuration, both the switch and the end-nodes use the Earliest Deadline First (EDF) algorithm for traffic control. The switch is responsible for admission control, MAC functions, frame buffering and traffic scheduling. The switch periodically sends synchronization frames to the end-nodes, at an interval, T_{cycle}, of ten maximum sized frames, T_{frame}, i.e.,

$$T_{cycle} = 10T_{frame}. \quad (1)$$

In this way, every node has a uniform comprehension about global time, with the resolution of T_{frame}. In this paper, we assume Fast Ethernet (100 Mbit/s) with a maximum frame size of 12 144 bits which, with some extra time for timing

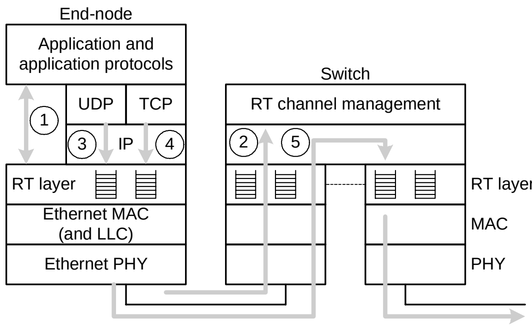

Figure 1: Layers and output queues.

uncertainties and for simplicity, gives $T_{frame} = 125$ μs, which just happens to match the time resolution of many telecommunication systems.

The function of and interaction with the RT layer etc shown in Figure 1 is explained below. When an application wants to setup an RT channel, it interacts directly with the RT layer (1). The RT layer then sends a question to the RT channel management software in the switch (2). Outgoing real-time traffic from the end-node uses UDP and is put in a deadline-sorted queue in the RT layer (3). Outgoing non-real-time traffic from the end-node typically uses TCP and is put in an FCFS-sorted (First Come First Serve) queue in the RT layer (4). In the same way, there are two different output queues for each port on the switch too (5).

An RT channel with index i is characterized by:

$$\{T_{period,i}, C_i, T_{deadline,i}\} \tag{2}$$

where $T_{period,i}$ is the period of data, C_i is the amount of data per period, and $T_{deadline,i}$ is the relative deadline used for the end-to-end EDF scheduling. Both $T_{period,i}$, C_i, and $T_{deadline,i}$ are expressed as the number of maximal sized frames, i.e., the number of T_{frame}.

When a node wants to establish an RT channel, it sends a request frame (ReqF) with source and destination node MAC and IP addresses and $\{T_{period,i}, C_i, T_{deadline,i}\}$ to the switch. A connection ID to distinguish between several possible connection requests is also added. When receiving a ReqF, the switch will calculate the feasibility of the traffic schedule between the requesting node and the switch and between the switch and the destination node. The ReqF is then forwarded to the destination node, after adding a network unique ID in the RT channel ID field. The destination node responds with a response frame (ResF) to the switch telling whether the establishment is accepted or not. The switch will then, after taking notation of the response, forward the ResF to the source node.

The RT layer in an end-node prepares outgoing real-time IP datagrams by changing the IP header before letting the Ethernet layers sending it (see Figure 2). The IP source address and the 16 most significant bits of the IP destination address, 48 bits together, are set to the absolute deadline of the frame. A 48 bit absolute deadline with a resolution of $T_{frame} = 125$ μs, gives a "life time" longer than

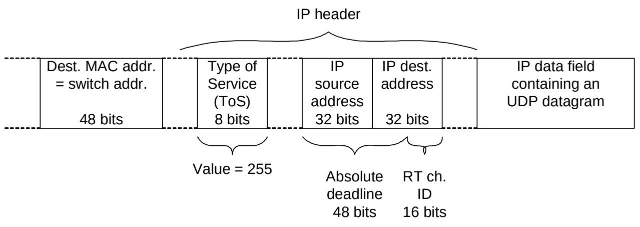

Figure 2: Data frame sent over an RT channel.

one thousand years. The 16 least significant bits of the IP destination are set to the RT channel ID for the RT channel to which the frame belongs. The MAC destination address is set to a special address that all nodes use for real-time traffic, while the Type of Service (ToS) field is always set to value 255.

The switch exchanges the source and destination IP addresses and the MAC destination address of an incoming real-time frame with the correct ones (as stored in the switch when the RT channel was established) for delivery to the final destination.

3 Asymmetric Deadline Partitioning

Below, we will show by example that the possible amount of guaranteed real-time traffic can be increased by partitioning the deadline asymmetrically between the different links crossed by an RT channel. We compare the asymmetric partitioning with a symmetric partitioning for a master-slave situation, i.e., a typical case of real-time traffic pattern in industrial systems.

We assume that master node M_1 is responsible for five slave nodes S_1 to S_5. The master node has one RT channel to each slave node via the switch. The real-time guarantee from M_1 to S_i is uphold by RT channel i, $1 \leq i \leq 5$. The latency due to synchronization and in-transmission frames [6] is neglected. For the deadline scheduling we assume:

$$T_{deadline,,i} = T_{period,i} = T_{D1,i} + T_{D2,i} \tag{3}$$

for each RT channel i, where $T_{D1,i}$ is the relative deadline for real-time traffic from the master node to the switch and $T_{D2,i,}$ is the relative deadline from the switch to the destination node. In the same way, let us assume that N_1 and N_2 represent the total number of RT channels on the links a specific RT channel crosses, i.e. the load of the links to and from the switch. For simplicity, all channels are assumed having the same characteristics and being unidirectional with the master node as the source node.

With an asymmetric deadline partitioning, $T_{deadline}$ is partitioned so the local deadline for a link of the end-to-end path is weighted according to the load of the link divided by the sum of the loads across the whole path. For our example, this gives:

$$T_{D1,i} = \frac{N_1}{N_1 + N_2} T_{period,i} = \frac{5}{6} T_{period,i} \tag{4}$$

$$T_{D2,i} = \frac{N_2}{N_1 + N_2} T_{period,i} = \frac{1}{6} T_{period,i} \qquad (5)$$

This is a simple partitioning method to show the opportunities with deadline partitioning. Our future work includes looking at partitioning method that can handle more complex traffic patterns and dynamic channel setup as the network is designed for.

According to the basic EDF theory [7], the utilization of real-time traffic is defined as

$$U = \sum \frac{C_i}{T_{period,i}}. \qquad (6)$$

One is assured that all deadlines are met if the utilization of real-time traffic does not exceed a certain level, $U_{\max}$:

$$U = \sum \frac{C_i}{T_{period,i}} \le U_{\max} \qquad (7)$$

This guarantee holds for deadline scheduling of traffic when the deadline for a specific link is equal to the period multiplied by a constant $k \le 1$, for all RT channels traversing the link in the same direction. When scheduling a channel with 100 % theoretical utilization, $U_{\max} = k$. For deadline scheduling of traffic with arbitrary deadlines, see [8] or subsequent work (e.g. [9]). We define $U_{\max1}$ as the maximum utilization for the link from the master node to the switch and $U_{\max2}$ as the maximum utilization from the switch to the slave node. In the theoretical case $U_{\max}$ is 100 %, but when using the network proposed in this paper the worst-case maximum utilization for the link from the switch and to the slave node is reduced from 100 % to 90 % due to having 10 % bandwidth for the network control. In symmetric deadline partitioning [6], we have $U_{\max2} = 45$ % and $U_{\max1} = 50$ %. When using asymmetric deadline partitioning with the weights from Equations 4 and 5 ($k_1 = 5/6$ and $k_2 = 1/6$, respectively), we get the following maximum utilizations instead:

$$U_{\max 1} = \frac{5}{6} \cdot 100\% = 83\% \qquad (8)$$

$$U_{\max 2} = \frac{1}{6} \cdot 90\% = 15\% \qquad (9)$$

We denote $C_{\max1}$ and $C_{\max2}$ as the maximum capacity (transmission time per period) per channel for the first and the second link traversed by an RT channel, respectively. When assuming the same period, T_{period}, and deadline, $T_{deadline}$, for all RT channels, we have

$$\frac{5C_{\max 1}}{T_{period}} = U_{\max 1} \Rightarrow C_{\max 1} = \frac{U_{\max 1} T_{period}}{5} \qquad (10)$$

for the master link and

$$\frac{C_{\max 2}}{T_{period}} = U_{\max 2} \Rightarrow C_{\max 2} = U_{\max 2} T_{period} \qquad (11)$$

for a slave link. For example, let us assume that

$$T_{period} = T_{deadline} = 24 \text{ ms} \qquad (12)$$

According to Equations 4, 5, 8, and 9, we then have:

$$\begin{aligned} T_{D1} &= 20 \text{ ms} \\ T_{D2} &= 4 \text{ ms} \\ C_{\max 1} &= \frac{U_{\max 1} T_{period}}{5} = 4 \text{ ms} \\ C_{\max 2} &= U_{\max 2} T_{period} = 3.6 \text{ ms} \end{aligned} \qquad (13)$$

The second link is the bottleneck because $C_{\max2} < C_{\max1}$. With $C = C_{\max2} = 3.6$ ms for all RT channels we get a utilization of $U = C / T_{period} = 0.15$ on each slave link and $U = 5C / T_{period} = 0.75$ on the master link.

In the asymmetric case, we have 75 % utilization on the master link compared with 50 % in the symmetric case. We still guarantee worst-case delay for real-time traffic. The analysis given above holds for the opposite direction (from each slave and to the master) and for other, not overlapping, master slave groups in the network too.

4 Software Implementation

We have implemented the switch using a PC with a 200 MHz Pentium processor, some network interface cards and LINUX 2.4.2. The two most important parts in the switch regarding the real-time communication are the RT-layer and the RT channel management (see Figure 1). The RT channel management is responsible for approving RT channels requested by nodes in the system. Information about the approved RT channels is made available to the RT layer, which handles the actual scheduling and forwarding of data frames. If the load on the switch becomes too high it simply discards non-RT frames.

The tests were performed in a network with the switch and three nodes, two sending and one receiving. All nodes were equipped with 100 Mbit/s full-duplex Ethernet cards. The PC acting as switch operates at 200 MHz. In the first test, the frame size, N_{byte}, is set to 1 000 bytes, including headers. In the receiving node, a timer, $t_{measure}$, starts at the arrival of the first RT frame and is stopped when the last frame is detected. During this time all received frames are registered either as RT frames or non-RT frames. The number of registered RT frames and non-RT frames are denoted as N_{RT} and N_{NRT}, respectively. The corresponding data rates for received data, R_{RT} and R_{NRT}, are calculated as:

$$R_{RT} = \frac{8 N_{Byte} N_{RT}}{t_{measure}} \qquad (14)$$

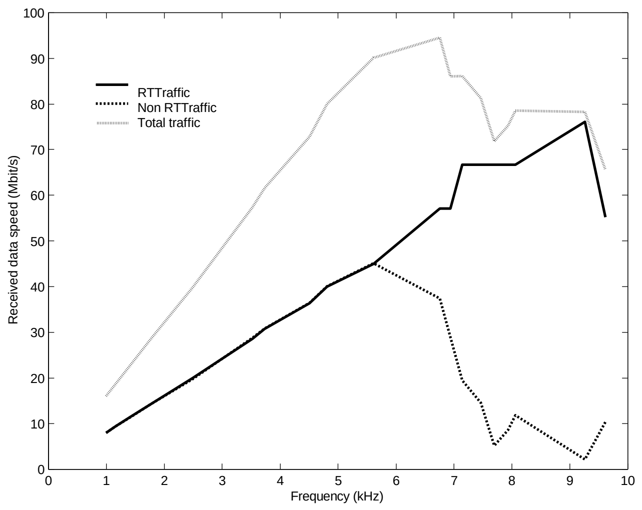

Figure 3: Test results.

$$R_{NRT} = \frac{8N_{Byte}N_{NRT}}{t_{measure}} \qquad (15)$$

The measured data rates in the receiving node are plotted against the frequency of the injected periodic traffic (see Figure 3). In this test, both RT traffic and non-RT traffic were injected with the frequency indicated in the figure. The result shows that the switch prioritizes the RT frames. It also shows that the breakpoint, where the switch begins to discard non-RT frames, is when the total traffic load reaches 96 Mbit/s, i.e., when the total traffic-load approaches the maximum capacity of the switch, the switch starts to discard non-RT traffic. As the traffic load increases, the RT-Switch is finally forced to discard RT traffic as well. This happens when the period of the traffic from the sending application is about 0.1 ms. This situation can only arise when the system runs without any channel management.

Additional tests where made including the two extremes: (*i*) Only maximum sized frames (1518 bytes) are transmitted. This gives a maximum throughput for the switch, measured to 96 Mbit/s, which is near wire speed. (*ii*) Only minimum sized frames (64 bytes) are transmitted. This gives the lowest throughput, measured to 18 Mbit/s. This case also gives us a maximum switching capacity of 14 400 frames/s.

The tests that have been performed shows that it is possible to build an Ethernet switch with deadline scheduling by using only standard components. This gives an indication on the possibility of implementing real-time capabilities in an Ethernet network. One can get a good feeling from the measurements about the amount of real-time traffic and number of ports that can be supported if, for example, a processor should assist a switch chip by handling deadline scheduling in software.

5 Conclusions

In this paper, we have presented an Ethernet network with support for real-time traffic by deadline scheduling. We have showed by example that the performance can benefit a lot from asymmetric deadline scheduling. An increase in utilization from 50 % to 75 % on the outgoing link from the master node (the bottleneck) is observed, still not violating the real-time guarantees.

From a software-implemented switch, we have showed experimental results. The measurements showed that the switch bottlenecks are 96 Mbit/s (measured for maximum-sized frames) and 14 400 frames/s (measured for minimum-sized frames). From these measurements one can get a good feeling of the amount of real-time traffic and number of ports that can be supported by an Ethernet switch that is fully or partly implemented in software.

References

[1] C. Venkatramani and T. S. Chiueh, "Supporting real-time traffic on Ethernet," *Proc. 15th IEEE Real-Time Systems Symposium (RTSS'94)*, pp. 282-286, 1994.

[2] D. W. Pritty, J. R. Malone, D. N. Smeed, S. K. Banerjee, and N. L. Lawrie, "A real-time upgrade for Ethernet based factory networking", *Proc. IEEE IECON'95*, vol. 2 , 1995.

[3] S.-K. Kweon, K. G. Shin, and G. Workman, "Achieving real-time communication over Ethernet with adaptive traffic smoothing," *Proc. 6th IEEE Real-Time Technology and Applications Symposium (RTAS'2000)*, Washington, D.C., USA, 31 May - 2 June 2000, pp. 90-100.

[4] S. Varadarajan and T. Chiueh, "EtheReal: A host-transparent real-time Fast Ethernet switch," *Proc. ICNP*, Oct. 1998.

[5] H. Zhang, "Service disciplines for guaranteed performance service in packet switching network," *Proc. of the IEEE*, vol. 83, no. 10, Oct. 1995.

[6] H. Hoang, M. Jonsson, U. Hagström, and A. Kallerdahl, "Switched real-time Ethernet with earliest deadline first scheduling - protocols and traffic handling", *Proc. Workshop on Parallel and Distributed Real-Time Systems (WPDRTS'2002) in conjunction with International Parallel and Distributed Processing Symposium (IPDPS'02)*, Fort Lauderdale, FL, USA, April 15-16, 2002.

[7] C. L. Liu and J. W. Layland, "Scheduling algorithms for multipprogramming in hard real-time traffic environment", *Journal of the Association for Computing Machinery*, vol. 20, no. 1, Jan. 1973.

[8] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishement in wide-area networks," *IEEE Journal of Selected Areas in Communications*, vol. 8, no. 3, pp. 368-379, Apr. 1990.

[9] Q. Zheng and K. G. Shin, "On the ability of establishing real-time channels in point-to-point packet-switched networks," *IEEE Transactions on Communications*, vol. 42, no. 2/3/4, pp. 1096-1105, Feb./Mar./Apr. 1994.