# Guaranteed real-time services over standard switched Ethernet

Xing Fan and Magnus Jonsson



Fan X, Jonsson M. Guaranteed real-time services over standard switched Ethernet. In: The IEEE Conference on Local Computer Networks, 30th Anniversary: proceedings, Sydney, Australia, November 15-17, 2005. Los Alamitos, Calif.: IEEE Computer Society; 2005. p. 490-492. Conference on local computer networks, 30.

DOI: http://dx.doi.org/10.1109/LCN.2005.73
Copyright: IEEE

Post-Print available at: Halmstad University DiVA
http://urn.kb.se/resolve?urn=urn:nbn:se:hh:diva-410

# Guaranteed Real-Time Services over Standard Switched Ethernet

Xing Fan and Magnus Jonsson

CERES, Centre for Research on Embedded Systems

School of Information Science, Computer and Electrical Engineering, Halmstad University, Halmstad, Sweden,

Box 823, S-301 18, Sweden. {Xing.Fan, Magnus.Jonsson}@ide.hh.se, http://www.hh.se/ide

## Abstract

*In this paper, we contribute with an approach to support guaranteed real-time services over standard switched Ethernet without additional hardware or software modification of the switch and the underlying standard. In our proposal, the traffic differentiation mechanism introduced by the IEEE 802.1D/Q standard and the standard hardware-implemented First Come First Served (FCFS) priority queuing are used in the switch and the source nodes. We have derived a feasibility analysis algorithm that can be used to ensure that the real-time requirements can be met. The algorithm also gives, as a sub-result, the needed buffer space in the end nodes and the switch. Moreover, our feasibility analysis supports variable-sized frames and switches with different bit-rates ports. The simulation analysis verifies our feasibility analysis.*

## 1. Introduction

A great number of protocols and schemes to improve the real-time characteristics of switched Ethernet have been proposed in recent years. However, the industrial real-time Ethernet solutions [1-4], such as Ethernet Powerlink, PROFINET, Ethernet/IP and EtherCAT, usually override the standard or reach their limitations when a certain degree of real-time capability is required. Meanwhile, the techniques proposed by the academic community [5-11] have been applied, with varying degrees of success and different drawbacks, as briefly discussed below. The EtheReal project [5] is throughput-oriented and only supports best effort traffic. The stochastic approaches [6] only yield average real-time performance, thus not providing guaranteed real-time services. Scheduling theory [12] has been applied on switched Ethernet for guaranteed real-time traffic, but with added cost of hardware/software modification and high computing demands introduced by the deadline sorting [7] [8]. NC (Network Calculus) [13] [14], an analytical technique to estimate the FCFS queuing delay, has been applied on fast Ethernet [9-11]. However, these NC solutions also require modification of the standard, such as implementing traffic shapers in the source nodes [9], or prioritization of logical real-time channels [10] [11]. In addition, the NC-based delay analysis is pessimistic [10] [11]. However, our approach has very low complexity by only using standard switched Ethernet with hardware-implemented First Come First Served (FCFS) priority queuing. Moreover, our feasibility analysis for FCFS queuing by using scheduling analysis has not previously been reported, to the best of our knowledge.

## 2. Network architecture and traffic handling

We consider a network with a star topology, where every node is connected to other nodes via the switch. The IEEE 802.1p [15] [16] queuing feature, which enables Layer 2 switches to set priorities for traffic and perform dynamic multicast filtering, is used to support our goal of a switch to distinguish between different traffic classes. By adding the 802.1p header to the frames, traffic is simply classified as hard real-time traffic, soft real-time traffic or non-real-time traffic and put into different priority queues. The hard real-time frames have the highest priority, while soft real-time frames have a lower priority than the hard real-time frames. Outgoing non-real-time traffic is treated as lowest priority. In our configuration, all the priority queues in the end nodes and in the switch are FCFS queues.

## 3. Real-time analysis

The time constraints are guaranteed by dynamic addition of real-time channels, each being a virtual unidirectional connection between two nodes in the system and characterized (with index $i$) by:

$$\{ Source_i, Dest_i, T_{period,i}, Cap_i, T_{deadline,i} \}, \quad (1)$$

where $Source_i$ indicates the source node, $Dest_i$ indicates the destination node, $T_{period,i}$ is the period of data generation, $Cap_i$ is the amount of pure data per period and $T_{deadline,i}$ is the relative deadline used for the admission control. $Cap_i$ is expressed as the number of data bytes of the whole message, while $T_{period,i}$ and $T_{deadline,i}$ are expressed in µs. Then the total amount of traffic (data and header) per period from real-time channel $i$, $C_i$ is:

$$C_i = \begin{cases} \left( \left\lfloor \frac{Cap_i}{T_{\max d}} \right\rfloor T_{ef} + ((Cap_i \bmod T_{\max d}) + T_h) \right), \\ \text{if } (Cap_i \bmod T_{\max d}) \geq T_{\min d}; \\ \left( \left\lfloor \frac{Cap_i}{T_{\max d}} \right\rfloor T_{ef} + 72 \right), \\ \text{if } 0 < (Cap_i \bmod T_{\max d}) < T_{\min d}; \\ \left\lfloor \frac{Cap_i}{T_{\max d}} \right\rfloor T_{ef}, \quad \text{if } (Cap_i \bmod T_{\max d}) = 0; \end{cases}, \quad (2)$$

where $T_{ef}$ is the length of a full-sized Ethernet frame, $T_h$ is the length of the header in an Ethernet frame, $T_{mind}$ is the minimum length of the data field in the Ethernet frame

Proceedings of the IEEE Conference on Local Computer Networks 30th Anniversary (LCN'05)
0-7695-2421-4/05 $20.00 © 2005 IEEE

without pad field and $T_{maxd}$ is the length of the data field in a full-sized Ethernet frame.

The feasibility test done by the admission controller includes two steps, checking the utilization constraint and the delay constraint. The utilization constraint is checked first. The utilization for a physical link connecting the switch and the end node $k$, $U_k$, must be less than or equal to 100%:

$$U_k = \sum_i \frac{C_i}{T_{period,i} Lr_k} \le 100\ \%\ , \qquad (3)$$

where $Lr_k$ is the bit rate of the considered link. The delay constraint is that the sum of the worst-case delay from the source node to the switch, $T_{d1,i}$, and the worst-case delay from the switch to the destination, $T_{d2,i}$, must be less than or equal to $T_{deadline,i}$:

$$T_{d1,i} + T_{d2,i} \le T_{deadline,i}\ . \qquad (4)$$

Instead of calculating $T_{d1,i}$ and $T_{d2,i}$ for each real-time channel, the delay constraint test can be improved by simply calculating the delay for each source node and each output port in the switch, since the following equation holds:

$$\begin{aligned} T_{d1,i} &= T_{d1,j} = Dnode_k \quad if\ Source_i = Source_j = k \\ T_{d2,i} &= T_{d2,j} = Dport_k \quad if\ Dest_i = Dest_j = k \end{aligned}\ , \qquad (5)$$

where $Dnode_k$ is the worst case delay from source node $k$, and $Dport_k$ is the worst case delay for the packets through output port $k$ in the switch.

The worst case situation at the source node is when all the related real-time channels start their periods at the same time and the maximum allowed capacity ($C_i$) of each real-time channel is used. Thus, $Dnode_k$ is calculated as the summation of all real-time channels' capacities:

$$Dnode_k = \sum_{Source_i = k} C_i / Lr_k \qquad (6)$$

The calculation of $Dport_k$ is shown in Algorithm 1, which is a kind of utility function that checks the queuing delay (queuing population) at different time instances. We use *Nchs* to denote the number of logical real-time channels in the system (including the existing channels and the new channel which is to be checked). *Nports* is used to denote the number of output ports in the switch. To reduce the time and memory complexity and avoid pessimistic analysis, several points have been addressed in Algorithm 1. First the test interval in Algorithm 1 is bounded to one hyperperiod, which is the period of time from the instant that all channels periods start at the same time until they start at the same time again. Second, instead of making a byte-by-byte check, checking the queuing delay is event-driven. That is, the queuing delay is checked only at such time points, at which a new period of any related logical real-time channel starts or at which the output queue from any source node is empty. Third the traffic smoothing transmission characteristic, i.e., the physical bit-rate prevents each source node of congesting the switch input port with frames from several real-time channels at the same time, is considered in the algorithm.

1. Initialization.
   **Input** (*Nchs, Nports, k, Lr*[*1..Nports*]);
   *t=0; tstep=0; Q=0;* $Dport_k$*=0;Plink=zeros*[*1..Nports*];
2. Find out $Dport_k$, the worst case delay in the switch for packets through output port *k*.
   2.1 Find out how many bytes on the way for each incoming physical link queued in each source node.
   **for** *j = 1..Nchs*
   **if** ($Dest_j$ ==*k* ) && *mod*(*t*, $T_{period,j}$)==0
   **then** *Plink* [$Source_j$] = *Plink* [$Source_j$] + $Cf_j$ ;
   **end if**
   **end for**
   2.2 Find out next checking point, either the time when the queue at an end node is empty or the time when a new period of a logical real-time channel starts.

$$tstep = \min\left( \bigcup_{i=1}^{Nports} \frac{Plink[i]}{Lr_i} \cup \bigcup_{i=1}^{Nchs} \left( T_{period,i} - \mathrm{mod}(t, T_{period,i}) \right) - \{0\} \right);$$

   2.3 Send bytes from each incoming link to the output buffer in the switch.
   **for** *j=1..Nports*
   **if** (*Plink* [*j*] >*0*)
   **then** *Plink* [*j*]= *Plink* [*j*]- $Lr_j$ *tstep;*
   *Q = Q +* $Lr_j$ *tstep;*
   **end if**
   **end for**
   2.4 Remove bytes from the output buffer if the buffer is not empty.
   **if** (*t* >*0*) && (*Q* > 0)
   **then** **if** (*Q* >= *tstep* )
   **then** *Q = Q -* $Lr_k$ *tstep ;*
   **else** *Q = 0 ;*
   **end if**
   **end if**
   2.5 Keep track of the worst-case queuing delay.
   **if** (*Q* > $Dport_k$ )
   **then** $Dport_k$ = *Q*;
   **end if**
   2.6 Increase *t* and reset *tstep.*
   *t=t+tstep; tstep=0;*
3. If it is not the end of the *hyperperiod*, check the queuing delay.
   **If** (*t<hyperperiod*)
   **then** repeat step 2;
   **end if**
4. **Return** ( $Dport_k/Lr_k$))

**Algorithm 1. $Dport_k$ calculation (FCFS queuing).**

If the above utilization constraint and the delay constraint are met, the new logical real-time channel can be accepted. Moreover, the buffer size for the hard real-time traffic in source node $k$, $BN_k$, and the buffer size for the hard real-time traffic to output port $k$ of the switch, $BS_k$, expressed in the number of bytes, are derived as:

$$\begin{aligned} BN_k &= Dnode_k Lr_k \\ BS_k &= Dport_k Lr_k \end{aligned}\ . \qquad (7)$$

When a logical real-time channel $i$ has been established, the network guarantees delivery of each real-time frame with a bounded delay:

$$T_{db,i} = T_{deadline,i} + T_{node} + T_{switch} + T_{trans} + 2T_{pro}, \quad (8)$$

where $T_{pro}$ is the maximum propagation delay over a link between an end node and the switch, $T_{trans}$ is the time needed to transmit the considered frame including the inter-frame gap over the Ethernet medium. $T_{switch}$ and $T_{node}$ are the worst-case process latencies for an Ethernet frame at the head of the hard real-time queue to leave the source node and to leave the switch, respectively, since we cannot interrupt the transmission of frames that have been stored in the NIC (Network Interface Card), even though they might have earlier deadlines than other frames. By having the delay bound, $T_{db,i}$, specified by the application, we can get the relative deadline for the admission control by:

$$T_{deadline,i} = T_{db,i} - T_{node} - T_{switch} - T_{trans} - 2T_{pro}. \quad (9)$$

## 4. Simulation analysis

To verify our result, we have simulated a network with a single full-duplex fast Ethernet switch and eight end nodes. In each simulation, logical real-time channels, randomly generated with uniformly distributed source and destination nodes, are added one by one and checked whether or not they can been accepted. The traffic intensity is increased by increasing the number of logical channels. Such simulations are run 100 times to get the average utilization at different traffic loads for different network characteristics. Figure 1 shows how the values of the deadlines influence the utilization. The period is set to 5 ms for all the logical real-time channels, the capacities are randomly generated from 1492 bytes to 8000 bytes, and the deadlines are randomly selected from a certain set. We observe different results when the deadline set is changed among the values $A=\{x \mid 1 \text{ ms} \le x \le 5\text{ms}\}$, $B=\{3\text{ms}\}$, $C=\{5\text{ms}\}$ and $D=\{10\text{ms}\}$. Note that the lowest utilization is obtained when all the deadlines are chosen from set A which includes very short deadlines. The reason is that shorter deadlines make it more difficult to meet the delay constraint. Figure 1 also reveals that our system can reach rather high utilization (close to the theoretical utilization upper bound), i.e., utilization reaches 93% when deadlines are twice the periods (the theoretical utilization bound is 100% in this case ).

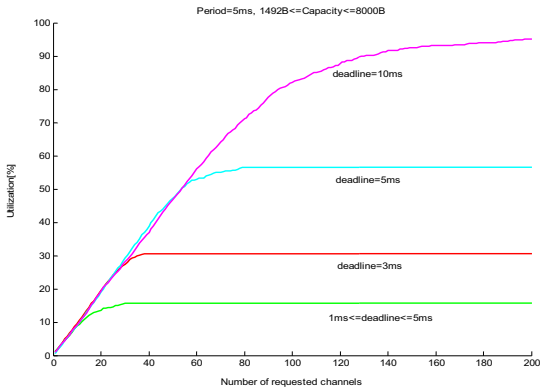


**Fig 1. Utilization comparison.**

## References


[1] Ethernet Powerlink. http://www.ethernet-powerlink.com.

[2] EtherCAT, Available at: http://www.ethercat.org.

[3] Profinet IRT. Available at: http://www.profibus.com.

[4] Ethernet/IP Specification, Release 1.0, vol. 1-2. ControlNet International and Open DeviceNet Association, June 2001.

[5] S. Varadarajan, "Experiences with EtheReal: a fault-tolerant real-time Ethernet switch," *Proc. of 8th IEEE International Conference on Emerging Technologies and Factory Automation*, vol. 1, 2001, pp. 183-194.

[6] B. Y. Choi, S. Song, N. Birch, and J. Huang, "Probabilistic approach to switched Ethernet for real-time control applications", *Proc. of 7th International Conference on Real-time Computing Systems and Applications (RTCSA'2000)*, Dec. 2000.

[7] H. Hoang, M. Jonsson, U. Hagstrom, and A. Kallerdahl, "Switched real-time Ethernet with earliest deadline first scheduling – protocols and traffic handling and simulation analysis," *Proc. of the International workshop on Parallel and Distributed Real-Time Systens,* Fort Lauderdale, FL, USA, 2002.

[8] X. Fan and M. Jonsson, "Efficient support for high traffic-volumes of short-message real-time communication using an active Ethernet switch," *Proc. of 10th International Conference on Real-Time and Embedded computing Systems (RTCSA'04)*, Gothenburg, Sweden, Aug. 25-27, 2004 pp. 517-533.

[9] J. Loeser and H. Haertig, "Low-latency hard real-time communication over switched Ethernet", *the 16th Euromicro International Conference on Real-time Systems,* 2004.

[10] J. Jasperneite, P. Neumann, M. Theis, and K. Watson, "Deterministic real-time communication with switched Ethernet", *Proc. of 4th International Workshop on Factory Communication Systems*, Västerås, Sweden, Aug. 28-30, 2002.

[11] A. Koubaa and Y. Song, "Evaluation and improvement of response time bounds for real-time applications under non-preemptive fixed priority scheduling", *International Journal on Production Research*, vol. 42, no. 14, June 2004, pp. 2899-2913.

[12] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in hard real-time traffic environment", *Journal of the Association for Computing Machinery*, vol. 20, no. 1, Jan. 1973, pp. 179-194.

[13] R. L. Cruz, "A calculus for network delay, I: network elements in isolation", *IEEE Transaction on Information Theory*, vol. 37, no. 1, Jan. 1991, pp. 114-131.

[14] R. L. Cruz, "A calculus for network delay, II: network analysis", *IEEE Transactions on Information Theory*, vol. 37, no. 1, Jan. 1991, pp. 132-141.

[15] IEEE 802.1D, *Information Technology – Telecommunications and Information Exchange between Systems-Local and Metropolitan Area Networks – Common Specification – Media Access Control (MAC) Bridges,* Std, Piscataway, NJ, USA, 1998.

[16] IEEE Std 802.1Q, *Virtual Bridged Local Area Networks,* Std, Piscataway, NJ, USA, 2003.