A Scheduling Algorithm of Periodic Messages for Hard Real-time Communications on a Switched Ethernet

Hee Chan Lee,† and Myung Kyun Kim†,

University of Ulsan, Nam Gu, Ulsan 680749, Korea

Summary

Switched Ethernet has been considered as a promising network technology for real-time communications required by process control, factory automation and other real-time applications. The switched Ethernet has many features for real-time communications such as providing a large amount of bandwidth, micro-segmentation of traffic, and full-duplex links, but to provide the hard real-time communications on the switched Ethernet, the traffic on the switch needs to be regulated not to overrun the capacity of the transmission and reception links of the switch. This paper proposes a message transmission model for hard real-time communications of periodic messages on the switched Ethernet and also proposes an algorithm to schedule the messages to be transmitted within their deadlines. The proposed scheduling algorithm is a distributed one and is performed by the source and the destination nodes without the modification of the operational features of the standard Ethernet switch. When a new periodic message needs to be transmitted, it is first checked whether it can be scheduled on both the transmission and the reception links without affecting the already-scheduled messages, and a feasible schedule is made for the new message if it is schedulable. The experiment result shows that the proposed scheduling algorithm guarantees the timely delivery of periodic messages and provides a flexible message transmission scheme to hard real-time applications by allowing them to be able to add new periodic messages while transmitting the messages that are already scheduled.

Key words:

Real-time communications, switched Ethernet, message scheduling, industrial communications, real-time systems.

1. Introduction

Real-time distributed control systems are becoming more widely used in industrial environment [1] They are used to support a broad range of applications such as process control, factory automation, automotive, robotics, and so on. Ethernet has been a de facto network standard in office environment, but it does not guarantee a real-time delivery of messages due to collisions on the network that occur when more than one node transmit messages at the same time. On the other hand, switched Ethernet has been considered as a promising network technology for real-time communications required by the real-time applications. The switched Ethernet has many features for real-time communications such as providing a large amount of bandwidth, micro-segmentation of traffic on the network, and full-duplex links [2] However, to provide hard real-time communications on the switched Ethernet, some real-time features must be added to both the end nodes and the switches in order to regulate the amount of traffic on the network not to overrun the output queue of the switch [3, 4, 5]. Traffic shaping techniques have been proposed in [4, 5] to provide a real-time communication by limiting the amount of traffic on the network, but their methods only show that the maximum delay on the network is bounded without considering the explicit deadlines of messages. Hoang et al. [3] have proposed hard real-time communication methods based on EDF (Earliest Deadline First) scheduling over the switched Ethernet. Their approaches, however, assume that both end nodes and the switch can schedule messages according to the EDF policy, which requires the addition of RT(Real-Time) layer to support the EDF scheduling above the MAC layer both on the nodes and on the switch. Pedreiras et al. [6] have proposed an elastic message transmission model called FTT-Ethernet to support dynamic real-time message requirements on Ethernet. Their method uses a synchronized message transmission based on a master-slave model, where the master checks the feasibility of a new message in response to the request of the transmission of the message from the slaves, makes a feasible schedule for the message, and sends the schedule in TM (Trigger Message) message to the slaves. Their centralized scheme lowers the advantage of the distributed message transmission model of switched Ethernet (and also Ethernet) and the master becomes a single point of failure of the network.

This paper proposes a message scheduling algorithm of periodic messages for hard real-time communications over the switched Ethernet. The proposed scheduling algorithm is a distributed one and operates between the source and the destination nodes without requiring the modification of the operation of the standard switch. The switched Ethernet in this paper uses a synchronized message transmission model that is similar to the one proposed by Pedreiras et al. [6], but our model uses a distributed scheme to check the feasibility condition of a new

Manuscript received May 25, 2006.
Manuscript revised May 30 , 2006.

message and to make a feasible schedule for the message. When a new periodic message needs to be transmitted, it is first checked whether it is feasible both on the transmission link and on the reception link without affecting the periodic messages that are already scheduled, and a feasible schedule is made for the message if it is schedulable. For the feasibility check and the construction of the feasible schedule, admission control messages are exchanged between the source and the destination nodes. Our message scheduling algorithm guarantees that all of the scheduled messages are delivered within their deadlines, and provides a short amount of worst case delay jitter for the periodic messages, which is important to the real-time industrial applications. The experiment result shows that the proposed scheduling algorithm guarantees the timely delivery of periodic messages and provides a flexible message transmission scheme to hard real-time applications by allowing them to be able to add new periodic messages while transmitting the messages that are already scheduled.

The rest of the paper is organized as follows. In section 2, the message transmission model on the switched Ethernet is discussed. In section 3, feasibility conditions for new periodic messages and a message scheduling algorithm for the periodic messages on the switched Ethernet are described. In section 4, an experiment for the proposed scheduling algorithm is described, and finally, conclusions and future works are discussed in the last section.

2. Message Transmission Model on Switched Ethernet

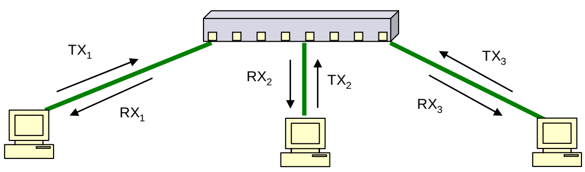


Fig. 1 Switched Ethernet example.

Each node on a switched Ethernet is connected to the switch by a full-duplex link which consists of a transmission link (TL) and a reception link (RL) as shown in Fig. 1 Both transmission and reception links operate independently and the switch uses a store-and-forward switching mode for forwarding frames from the input ports to the output ports. In the store-and-forward switching, the switch determines the output port and forwards the frame to the output port after receiving the entire frame

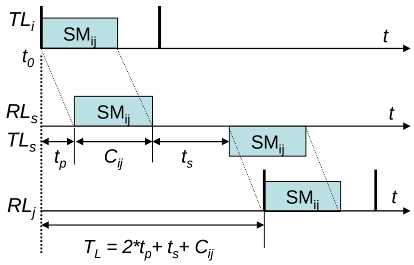


Fig. 2. Message delay in a switch.

When a switch operates in the store-and-forward switching mode, if a message SM_{ij} from node N_i to N_j is transmitted at time t_0 from N_i, the first bit of the message arrives at N_j after T_L = 2*t_p + t_s + C_{ij} amount of delay from t_0 if there is no collision at the output port of the switch at RL_j and the output queue of the switch at RL_j is empty. This is shown in Fig. 2, where t_p is a propagation delay on a link, t_s is a switching delay in the switch, and C_{ij} is a transmission time of message SM_{ij} In the figure, RL_s and TL_s denotes the incoming and outgoing link of the switch, respectively. The switching delay t_s depends on the switch vendor, but usually it is about 10 μs in 100 Mbps switches, and t_p depends on the length of a link, and it is less than 1 μs in a 100m link. C_{ij} depends on the size of the message and the network speed, and in the case of a minimum size frame on a 100 Mbps switch, C_{ij} is about 6.5 μs including IFS (inter-frame space). Thus, in a 100Mbps switch, the message delay T_L is about 20 μs

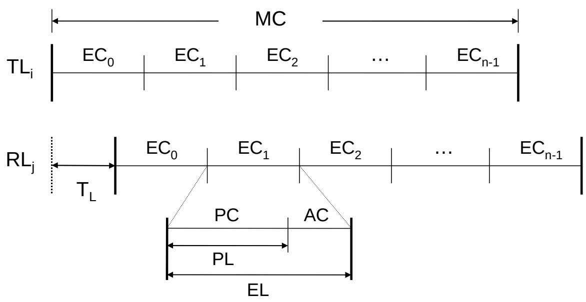


Fig. 3. Message transmission model in a switched Ethernet.

We assume that all of the nodes are synchronized and both the transmission link and the reception link are composed of a sequence of macro cycles (MCs), each of which is further divided into smaller basic message transmission units called elementary cycles (ECs) as shown in Fig. 3. Each EC is further divided into two time periods: periodic cycle (PC) and aperiodic cycle (AC). PC is used for transmitting periodic messages and AC is for transmitting aperiodic messages.

All of the periodic messages have their strict message transmission deadlines and the required network bandwidth should be allocated to the messages before transmission both on the transmission link and on the reception link in order to be delivered to the intended destination node within their deadlines. AC is used to transmit aperiodic messages such as control messages and maintenance messages. Examples of control messages are *Periodic_msg_req* and *Periodic_msg_rep* messages which are exchanged between the source and destination nodes for the admission control of a new periodic message. PL (Length of PC) and AL (Length of AC) is determined according to the ratio of the amount of message traffic of periodic messages and that of aperiodic messages. In this paper, we only consider the scheduling of periodic messages.

A periodic message SM_{ij} from node N_i to N_j has a real-time transmission requirement { P_{ij} , D_{ij} , C_{ij} }, where P_{ij} , D_{ij} , and C_{ij} denote the period, the deadline, and the length of the message, respectively. We assume that D_{ij} = P_{ij} = k * EL (Length of EC) for some integer k We also assume that the real-time transmission requirements of all of the periodic messages are known a priori, and the number of ECs in a MC is n = LCM(P_{ij} / EL) for all i and j (LCM means least common multiple).

3. Real-time Message Scheduling on the Switched Ethernet

Each node on the switched Ethernet maintains a message schedule which contains an order of messages to be transmitted on each EC of a MC. When a node wants to send a new periodic message during message transmission, an admission control process is carried out in the next AC between the source and the destination nodes of the message. If the new message becomes feasible, the message is added to the message schedule and begins to be transmitted from the next MC. The admission control step of a new periodic message SM_{ij} from N_i to N_j which has a real-time transmission requirement { P_{ij} , D_{ij} , C_{ij} } is as follows.

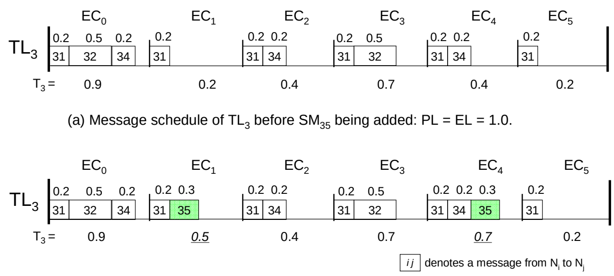
(i) Node N_i checks the feasibility of the message on transmission link TL_i and sends *Periodic_msg_req* message to node N_j in AC of the current EC if it is feasible on TL_i

(ii) When receiving the *Periodic_msg_req* message from N_i , node N_j checks the feasibility of the message transmission on reception link RL_j and returns the result to the node N_i in *Periodic_msg_rep* message.

(iii) When N_i receives the *Periodic_msg_rep* message, it checks the message and adds the message SM_{ij} in the message schedule.

3.1 Feasibility check on the transmission link

Scheduling of a message SM_{ij} with a real-time transmission requirement { P_{ij} , D_{ij} , C_{ij} } is to find a set of ECs in a MC, where adjacent ECs are P_{ij} distance away and all of the ECs in the set have available bandwidth greater than or equal to C_{ij} both on transmission link TL_i and on reception link RL_j . For the feasibility check on its transmission link, each node maintains the current amount of traffic in each EC of the link.



(a) Message schedule of TL_3 before SM_{35} being added: PL = EL = 1.0.

(b) Message schedule of TL_3 after SM_{35} being added : PL = EL = 1.0.

Fig. 4 Feasibility check of SM_{35} on TL_3 : real-time requirement of SM_{35} is {3, 3, 0.3}.

Fig. 4 shows an example of feasibility check of SM_{35} on transmission link TL_3 that has a message transmission requirement {3, 3, 0.3}. In the example, a MC consists of 6 ECs, which means that the period of each message can be 1, 2, 3, or 6. For the ease of the description, we assume that AL = 0 and PL = EL = 1.0. In the figure, T_3 = { $T_{3,i}$: $0 \le i \le 5$ }, where $T_{3,i}$ denotes the amount of current traffic in EC_i of TL_3 . Fig. 4-(a) shows the current state of the message schedule of TL_3 that is maintained in N_3

before message SM_{35} is added After that, when node N_3 wants to send a new message SM_{35} that has a real-time requirement {3, 3, 0.3} to node N_5, it first divides all of the ECs in a MC into the following three sets of ECs:

S_0 = { EC_0, EC_3 },

S_1 = { EC_1, EC_4 },

S_2 = { EC_2, EC_5 },

where the ECs in each of the sets are 3**EL* (P_{35}) distance away. After that, N_3 checks the feasibility of message SM_{35} for each of the three sets S_0, S_1, and S_2. In order that the message SM_{35} should be feasible on a set of ECs, all of the ECs in the set must have available bandwidth more than or equal to 0.3(C_{35}) In the example of Fig. 3-(a), S_0 is not feasible because the current amount of traffic in EC_0 of TL_3, $T_{3,0}$, is 0.9, and so the remaining bandwidth on the EC is (1.0 - $T_{3,0}$) = 0.1 < 0.3. On the other hand, S_1 and S_2 are feasible because all of the ECs in the sets have the remaining bandwidth more than or equal to 0.3. Fig. 3-(b) shows the message schedule after the new message SM_{35} has been added in the ECs of S_1

After checking the feasibility on its transmission link, the source node transmits *Periodic_msg_req* message to the destination node in AC of the current EC. For the feasibility check on the reception link, the source node carries in the *Periodic_msg_req* message the real-time requirement of the new message, the current amount of traffic in each EC of the transmission link, and the information about which sets of ECs are feasible on the transmission link. When the destination node receives the message, it checks the feasibility on the reception link and returns the result back to the source node in *Periodic_msg_rep* message. When receiving the *Periodic_msg_rep* message, the source node schedules the message to be added in the message schedule. **Algorithm 1** is an algorithm to check the feasibility of a new message SM_{ij} on transmission link TL_i

32 Feasibility check on the reception link

As we already mentioned in Section 2 (Fig. 2), if message SM_{ij} is transmitted in an EC by source node N_i, the first bit of the message arrives at its destination node N_j after T_L = 2* t_p + t_s + C_{ij} amount of delay from t_0 (transmission starting time at node N_i) when there is no message collision at the output port of the switch and the output buffer is empty. T_L corresponds to the time after (2* t_p + t_s) amount of delay from the transmission finishing time at node N_i

In order that a new message should be feasible in an EC of the reception link, the new message must be able to be transmitted completely within the EC without affecting the other messages that have already been scheduled in the EC before the message. For the feasibility check on its reception link, each node maintains the *expected transmission finishing time* in all of the ECs of the reception link which is defined as follows.

// **Algorithm 1**: feasibility check of SM_{ij} on TL_i

// Real-time requirement of SM_{ij} : { P_{ij}, D_{ij}, C_{ij} }

// Macro cycle length : *M*

1. *p* = P_{ij} / *EL*;
2. **for** (*k* = 0; *k* ≤ (*p* - 1); *k*++) {
3. $T_{max} = \max_{0 \le m \le M/p-1} \{ T_{i,k+m*p} \}$;
4. **if** (T_{max} + C_{ij} ≤ *PL*)
5. S = S ∪ {*k*}; // feasible on TL_i
6. }
7. **if** (*S* == NULL) // not feasible on TL_i
8. Rejects the transmission request of message SM_{ij} ;
9. **else** // Feasible on TL_i
10. Send *Periodic_msg_req* [SM_{ij}, { P_{ij}, D_{ij}, C_{ij} }, T_i, S] to N_j ;
11. **return**;

Definition 1 (*Expected transmission finishing time*) : The *expected transmission finishing time* in EC_i of reception link RL_j, $R_{j,i}$, is the expected time at which all of the messages arrived in EC_i at the output queue of RL_j can be transmitted completely.

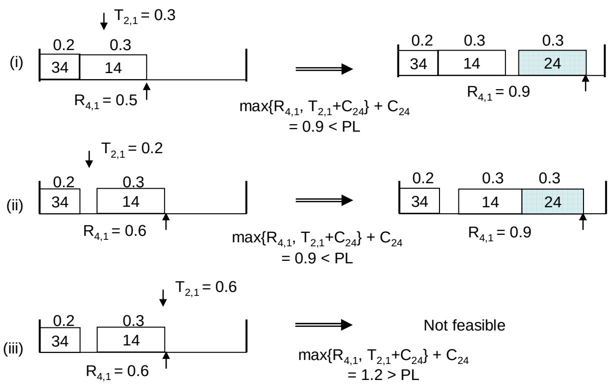
When node N_j receives *Periodic_msg_req*[SM_{ij}, { P_{ij}, D_{ij}, C_{ij} }, T_i, S] message from N_i, it checks the feasibility of SM_{ij} in the ECs of the reception link RL_j in

S using R_j, T_i and C_{ij}. In order for SM_{ij} to be feasible in EC_k of RL_j, it must be satisfied that

$$\mathbf{max}\ \{R_{j,k},\ T_{i,k}+C_{ij}\}+C_{ij}\le PL. \qquad (1)$$

If the message SM_{ij} is feasible on both TL_i and RL_j, then the message is scheduled to be transmitted in EC_k and the expected transmission finishing time $R_{j,k}$ is updated as follows:

$$R_{j,k}=\mathbf{max}\ \{R_{j,k},\ T_{i,k}+C_{ij}\}+C_{ij} \qquad (2)$$



(a) Before SM_{24} (C_{24}=0.3) is added. (b) After SM_{24} (C_{24}=0.3) is added.

Fig. 5 An example of feasibility check of SM_{24} in EC_1 of RL_4: feasible cases.

Fig. 5 shows some examples of feasibility check on the reception link and updating the expected transmission finishing time. In the figure, $T_{2,1}$ denotes the transmission starting time of message SM_{24} on TL_2, and $R_{4,1}$ denotes the current expected transmission finishing time of EC_1 on RL_4. In case (i),

$$\mathbf{max}\{R_{4,1},\ T_{2,1}+C_{24}\}+C_{24}=0.9\le PL,$$

so, SM_{24} is feasible. We can see that SM_{24} is also feasible in case (ii). However, in the case of (iii),

$$\mathbf{max}\{R_{4,1},\ T_{2,1}+C_{24}\}+C_{24}=1.2>PL,$$

so, the message SM_{24} is not feasible on EC_1 of RL_4. If the message is feasible, the destination node updates the expected transmission finishing time $R_{4,1}$ according to the equation of (2). Fig. 5-(b) show the update result of $R_{4,1}$ after message SM_{24} is added in EC_1

Algorithm 2 is an algorithm to check feasibility on the reception link when a destination node receives *Periodic_msg_req* message from a source node.

// **Algorithm 2**: feasibility check of SM_{ij} on RL_j

// N_j receives *Periodic_msg_req* [SM_{ij},{ P_{ij} , D_{ij} , C_{ij} }, T_i , S] from N_i

// Macro cycle length : M

// next(S): next element of S

1. $p = P_{ij}\ / EL$;

2. **while** ((k = next(S)) $\le$ NULL) {

3. $\quad R_{\max}=\max\limits_{0\le m\le M/p-1}\{\mathbf{max}\ \{R_{j,k+m*p},\ T_{j,k+m*p}+C_{ij}\}\}$;

4. $\quad$ **if** ($R_{\max}+C_{ij}\le PL$) { // feasible

5. $\quad\quad$ Send *Periodic_msg_rep* [SM_{ij}, k] to N_i ;

6. $\quad\quad$ **for** (m = 0; $m \le$ (M/p - 1); m++) {

7. $\quad\quad\quad R_{j,k+m*p}=\mathbf{max}\ \{R_{j,k+m*p},\ T_{j,k+m*p}+C_{ij}\}+C_{ij}$;

8. $\quad\quad$ }

9. $\quad\quad$ **return**;

10. $\quad$ }

11. }

12. Send *Periodic_msg_rep* [SM_{ij} , NULL] to N_i ;

13. **return**;

---

33 Update the message table

When node N_i receives *Periodic_msg_rep*[SM_{ij} , k] message from node N_j, it sees the feasibility check result on the reception link. If k is NULL, SM_{ij} is not feasible on the reception link RL_j, so it rejects the request of the message transmission. If k is not NULL, the message SM_{ij} is feasible in the ECs in { EC_k , EC_{k+p} , EC_{k+2*p} , … , EC_{k+n-p} }, where n is the number of ECs in a MC and p = P_{ij} /EL. Thus, N_i updates the current amount of traffic of ECs in the set and updates the message table. As mentioned in subsection 3.1, the newly added message is scheduled to be sent after all of the messages that have been already scheduled before the message are transmitted. **Algorithm 3** is an algorithm to update the message table when a source node receives *Periodic_msg_rep* from a destination node. In the algorithm, *Update_msg_table* is a function to update the message table such that the message SM_{ij} is scheduled to be transmitted in ECs of $\{EC_k, EC_{k+p}, EC_{k+2*p}, \cdots, EC_{k+n-p}\}$. At the start of the next MC, the message table becomes the current message table, and all of the newly added messages are transmitted from the MC.

// **Algorithm 3**: message table update of TL_i

// N_i receives *Periodic_msg_rep* [SM_{ij}, *k*] from N_j

// Macro cycle length : *M*

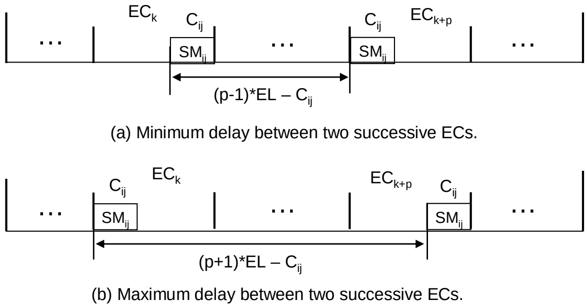
// Update_msg_table (SM_{ij}, *k+m*p*): update message table EC_{k+m*p} of on TL_i

if ($k \neq$ NULL) { // feasible
    $p = P_{ij} / EL$;
    for ($m = 0$; $m \leq (M/p - 1)$; m++) {
        $T_{i,k+m*p} = T_{i,k+m*p} + C_{ij}$; // update T_i
        Update_msg_table (SM_{ij}, $k+m*p$);
    }
} **else** // not feasible
    Rejects the transmission request of message SM_{ij};
return;

Theorem 1: The worst case delay jitter of a periodic message that is scheduled by the proposed scheduling algorithm is 2*EL.

Proof:

According to our algorithm, a message SM_{ij} which has a real-time requirement $\{P_{ij}, D_{ij}, C_{ij}\}$ is scheduled to be transmitted in ECs of $\{EC_k, EC_{k+p}, EC_{k+2*p}, \cdots, EC_{k+n-p}\}$, where $n = MC/EL$, $p = P_{ij}/EL$, and $0 \leq k \leq p-1$. The maximum delay between two messages scheduled in two successive ECs in the set is $(p+1)*EL - C_{ij}$ as shown in Fig. 6-(b), and the minimum delay is $(p-1)*EL - C_{ij}$ as shown in Fig. 6-(a). So the worst case delay jitter is $((p+1)*EL - C_{ij} - (p-1)*EL - C_{ij}) = 2*EL$. Q.E.D.



(a) Minimum delay between two successive ECs.

(b) Maximum delay between two successive ECs.

Fig. 6 Delay for a scheduled message.

4. Experiment of the Proposed Scheduling Algorithm

We have implemented the proposed scheduling algorithm on systems using Linux 2.6.10 with high resolution POSIX timers [9] and experimented on a 100Mbps switched Ethernet using Cisco Catalyst WS-C1912C-EN switch. The switched Ethernet consists of a switch and 5 nodes, and each node is connected to the switch through a full-duplex link which operates at 100Mbps in both transmission and reception links.

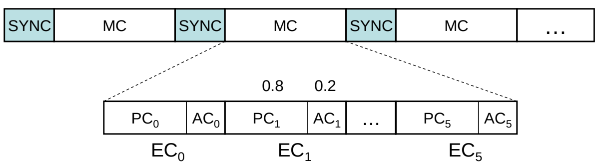


Fig. 7 Message transmission model in the experiment.

We assume all nodes in a switched Ethernet are synchronized. In the experiment, we used for the synchronization a master node which broadcasts a SYNC message at the beginning of each MC. Each node recognizes the reception time of a SYNC message as the start time of a MC as shown in Fig. 7. We have used the master node only to synchronize the nodes in the network, and it is not necessary if we use one of the other synchronization methods [7, 8] The SYNC message is 64 bytes long (minimum-size frame) and contains the information on the message transmission structure such as *M* (the number of ECs in a MC), *EL* (the size of EC), *PL* (the size of PC), and *AL* (the size of AC). In our experiment, $M = 6$, $EL = 1$ ms, $PL = 0.8$ ms, and $AL = 0.2$ ms, as shown in Fig. 7.

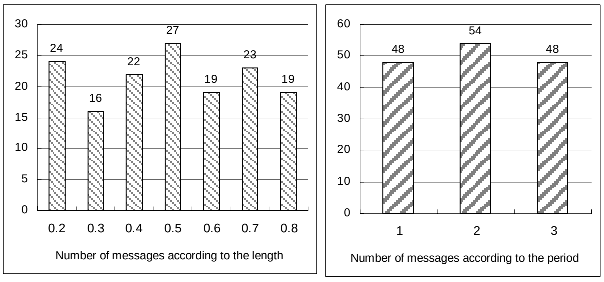


Fig. 8 Distribution of the message generated in the experiment

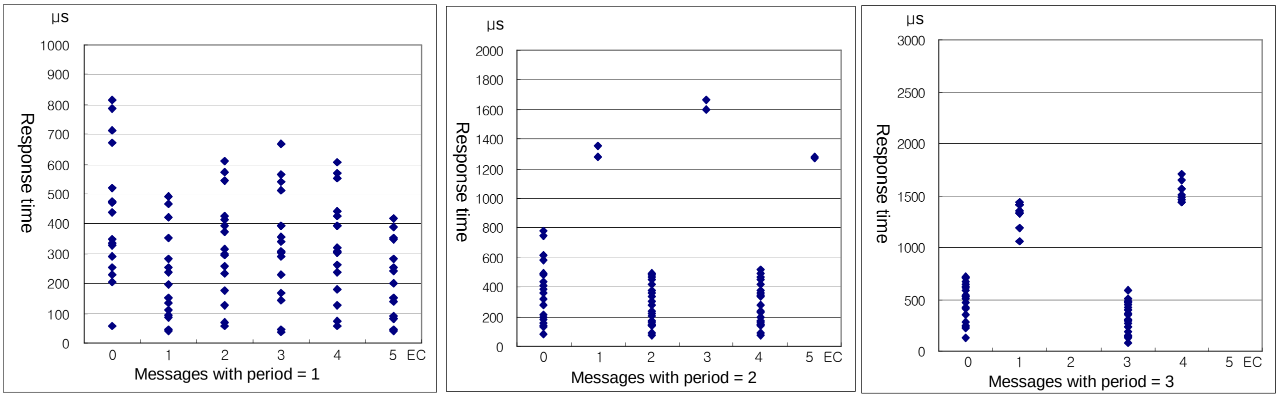
For the experiment, we have generated 30 messages randomly for each node, and the messages are stored in the message buffer in each node in their order of generation time. The lengths of the generated messages were chosen randomly in [20, 80] μs and the periods of the messages were chosen randomly in {1, 2, 3} *ms* which is a multiple of *EL*. Fig. 8 shows the distribution of the messages generated in this experiment.

The proposed scheduling algorithm is designed such that new periodic messages can be added dynamically while the already scheduled messages are being transmitted. To show the dynamic admission control of the scheduling algorithm, we have divided the messages into two parts with the same size of messages, and have performed the admission control of the messages in two steps as follows.

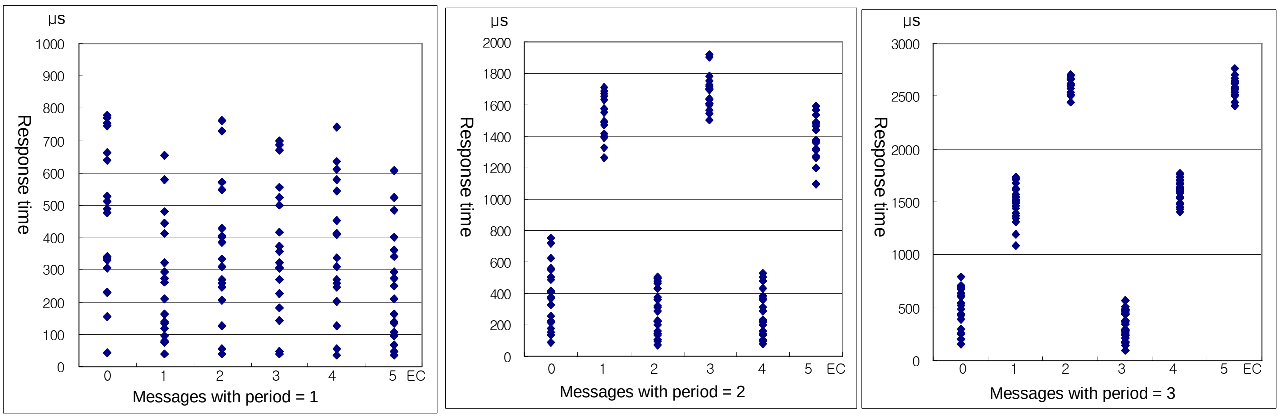
(i) Each node has performed the admission control process for the first part of 15 messages before stating the first MC, and

(ii) Each node has performed the admission control process for the remaining 15 messages one by one until the first admission failure occurs while transmitting the already scheduled messages in the next MC.

To show the timely delivery of the scheduled messages, we evaluated the response time for each message. The response time of a message instance is the time duration from the release time of the message instance to the transmission finishing time of the message. We assumed



(a) Response times of messages scheduled in the first step.



(b) Response times of messages after the remaining messages added in the second step.

Fig. 9 Response times of scheduled messages.

that a new message instance is activated at the beginning of each period of the message. Fig. 9 shows the response times for the messages on a switched Ethernet. Fig. 9-(a) shows the response times for the first 15 messages that were scheduled in the first step, and Fig. 9-(b) shows the result after the remaining 15 messages have added in the next MC. In the figure, we can see that all messages which were scheduled by the proposed scheduling algorithm are delivered within their deadlines. In the second step, each node continues the addition of new messages until the first admission failure happens. Among 150 messages (30 messages for each of 5 nodes), 99 messages were scheduled. The admission failure denotes that there is no remaining bandwidth enough to admit a new message.

To see the network utilization obtainable by the proposed message scheduling algorithm, we have evaluated the network utilization at the time when the first admission failure happens in the second step. The network utilization is defined a follows:

$$U = \frac{\sum_{r \in MS} C_r}{M * PL},$$

where *MS* denotes a set of messages that are transmitted in an MC, and C_r is the length of message *r* In our experiment, the network utilization was $U = 0.69$.

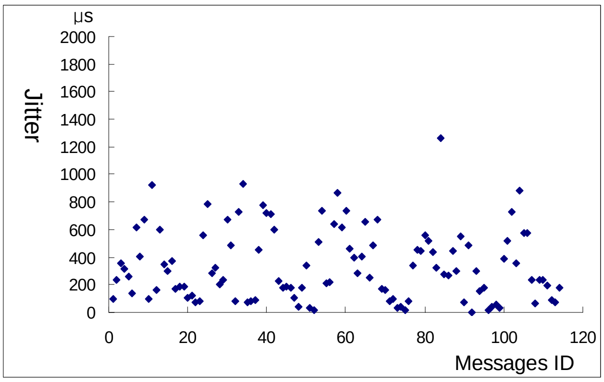


Fig. 10 Delay jitter for the messages scheduled in the experiment.

The proposed algorithm guarantees the timely delivery of the periodic messages within their deadline and also provides a short worst-case delay jitter for the scheduled messages. Fig. 10 shows the delay jitter for each of the messages scheduled in the experiment. The average delay jitter was 340 μs

5. Conclusions and Future Works

Real-time distributed control systems have been more widely used in industrial applications like process control, factory automation, vehicles and so on. In those applications, each task must be executed within a specified deadline, and also the communications between the tasks have to be completed within their deadlines to satisfy the real-time requirements of the tasks. Switched Ethernet which is the most widely used in the office has good operational features for real-time communications. The switched Ethernet, however, needs some mechanisms to regulate the traffic on the network in order to satisfy the hard real-time communication requirements of the industrial applications.

This paper proposed a synchronized transmission model for hard real-time communications of periodic messages over the switched Ethernet, and also proposed a dynamic message scheduling algorithm for the real-time periodic messages. The proposed scheduling algorithm uses a distributed scheme and is performed between the source and the destination nodes without requiring the modification of the operational features of the standard Ethernet switch. For a new message transmission request, our message scheduling algorithm checks first the feasibility of the message both on the transmission and reception links, and makes a feasible schedule for the message to guarantee the timely delivery of the message without affecting the messages scheduled already. Thus, our scheme is suitable to the applications which require hard real-time communications and whose communication requirements change dynamically.

Acknowledgments

The authors would like to thank University of Ulsan and Ministry of Commerce, Industry and Energy and Ulsan Metropolitan City which partly supported this research through the Network-based Automation Research Center (NARC) at University of Ulsan for the financial support to this research.

References

[1] Buttazzo Georgio C.: Hard real-time computing systems: Predictable scheduling algorithms and applications(2nd Ed.), (2005).
[2] Jasperneit J., and Neumann P.: Switched Ethernet for factory communication. In Proc. of ETFA - 8th IEEE Int'l Conf. on Emerging Technologies and Factory Automation, Antibes, France (2001).
[3] Hoang H., Jonsson M., Hagstrom U., and Kallerdahl A.: Switched real-time ethernet with earliest deadline first scheduling- protocols and traffic handling. In Proc. of Int'l

Workshop on Parallel and Distributed Real-Time Systems, Fort Lauderdale, FL, USA (2002).
[4] Loeser J. and Haertig H.: Low-latency hard real-time communication over switched Ethernet. In Proc. ECRTS (EUROMICRO Conf. Real-Time Systems) (2004).
[5] Kweon S. K., Shin K. G., and Workman G.: Achieving real-time communication over Ethernet with adaptive traffic shaping. In Proc. RTAS (IEEE Real-Time Technology and Applications Symposium) (2000).
[6] Pedreiras P., Almeida L., and Gai P.: The FTT-Ethernet protocol: Merging flexibility, timeliness and efficiency. In Proc. ECRTS (EUROMICRO Conf. Real-Time Systems) (2002), pp. 134--142.
[7] Kopez H. and Ochsenreiter W.: Clock synchronization in distributed real-time systems. IEEE Tr. on Computers C-06, 8 (1987), pp. 833--839.
[8] IEEE: 1588 IEEE standard for a precision clock synchronization protocol for networked measurement and control systems. Tech. Rep., IEEE Instrumentation and Measurement Society (2002).
[9] Linux with high resolution POSIX timers: http://sourceforge.net/projects/high-res-timers

Hee Chan Lee received the B.S. degree in Computer Engineering from University of Ulsan in 2004. He is now with School of Computer Engineering and Information Technology of University of Ulsan. He is interested in real-time communications, industrial communications, and home networks.

Myung Kyun Kim received the B.S. and M.S. degrees in Computer Engineering from Seoul National University in 1984. He received M.S. and PhD. Degrees from KAIST in 1986 and 1996, respectively. He is now with School of Computer Engineering and Information Technology of University of Ulsan. He is interested in real-time communications, home networks, and industrial communications