A Switched Ethernet Protocol for Hard Real-Time Embedded System Applications

Alimujiang Yiming and Toshio Eisaka

Kitami Institute of Technology, Kitami, Hokkaido, Japan

alm_ym@mer.cs.kitami-it.ac.jp

Abstract: This paper presents a protocol to support hard real-time traffic of end-to-end communication over non real-time LAN technology. The network is set up with nodes and switches, and real-time communication is handled by software (protocol) added between the Ethernet protocols and the TCP/IP suite. The proposed protocol establishes a virtual circuit based on admission control and manages hard real-time traffic to bypass the TCP/IP stack. This makes considerably reduce the dwell time in the nodes, and increase the achievable data frame rate. After the bypassing, traffic schedule is performed according to dynamic-priority EDF algorithm.

The work does not need any modifications in the Ethernet hardware and coexists with TCP/IP suites, and then the LAN with the protocol can be connected to any existing Ethernet networks. It can be adopted in industrial hard real-time applications such as embedded systems, distributed control systems, parallel signal processing and robotics.

We have performed some experiments to evaluate the protocol. Compared to some conventional hard real-time network protocols, the proposed one has better real-time performances and meets the requirements of reliability for hard real-time systems.

Keywords: communication protocol; Ethernet; hard real-time; switch; traffic scheduling.

1. Introduction

With the increasingly demand for real-time industrial systems, the ability of computer networks to handle deadline guaranteed “hard” real-time communication is becoming more and more important [1-4]. High bandwidth and strict deadline guarantee are the critical necessary conditions for hard real-time applications. For the real-time systems that need high-bandwidth communication, Ethernet becomes the communication link most designers think of first today. Unlike traditional, bus-based CSMA/CD Ethernet, the switched Ethernet is a star-based topology which can provide collision domain to each of the ports of a switch. Therefore, end-node cooperation is needed only for bandwidth control, not any more to avoid collisions. In addition, current Ethernet standards promise transfer speeds of 100Mbit/s (Fast Ethernet) and more (Gbit Ethernet). This will considerably improve the Ethernet bandwidth for real-time communication applications.

Several researches have been done to treat hard real-time communication. MIL-STD-1533 [5] is an early development of the industrial protocol. It is reliable standard interface for token ring LAN. However, bandwidth has become a limiting factor for MIL-STD-1553. Another protocol that enables hard real-time communication on Ethernet is called RTCC (Real-Time Communication Control) [6]. RTCC is centralized approach that has the disadvantage that failure in the controller will lead to the entire network useless, unless some sort of recovery protocol is implemented. Other research using switched Ethernet for hard real-time communication is studied in [7]. The main idea of the work is traffic shaping to every end-node on the switched Ethernet network. This enables the network shared with non real-time nodes; however, the traffic shaping capabilities also bring about time delay. An Ethernet access protocol called EDF-CSMA [8] has been proposed, which can also deal with soft and non real-time communication. This method provides a solution to handle real-time messages with strict deadlines, making it suitable to a wider range of real-time applications. However, hardware modification is necessary to implement this protocol. Tutorial of hard real-time communication in packet switched networks is found in [9].

Recently, new schemes have been proposed based on call admission control (CAC) depending on quality of service (QoS) and choice of the packet service discipline. The common concept of the schemes is establishment of a *Real-time Channel*: a simplex, virtual connection between source nodes and destination nodes, with a priori guarantees for communication performance in switching networks [10-13].

To support the QoS demands of applications, both the ATM and the IP community have defined service classes that provided per-flow guarantees to applications [14-15]. In order to provide guaranteed services, resources need to be reserved for every accepted connection. ATM does this at the data-link layer. For every call it reserves a virtual channel over all links on the route/switch from the source to the destination. At the network layer, resource reservation can be done using Diffserv [16] or Multi Protocol Label Switching (MPLS) [17]. Applications at this level are classified as best-effort, rate sensitive or delay sensitive, therefore, applications at this level has no guarantee for strict deadline-sorted hard real-time communication.

RSVP [18] can be used to do reservation at the IP layer. In RSVP, the admission control and resource allocation policies are based on different levels of QoS specifications --- deterministic or guaranteed, statistical and best-effort. The main feature of RSVP is the maintenance of *soft-state* in the intermediate switches, which must be periodically refreshed.

This approach allows RSVP to adapt to network load and outages. RSVP is designed to deliver resource reservation requests to related switches. Therefore, RSVP is more appropriately a *state establishment protocol*. RSVP makes resource reservations for both unicast and many-to-many multicast applications, therefore, it is more appropriate for multimedia communication in a wide area network such as videoconferencing. Because RSVP is based on IP, there is no guarantee of deadline in application service lifetime, and have large runtime overhead, therefore, there are few applications in industrial control systems.

Moreover, the IP datagram header contains an 8-bit field called ToS (Type of Service). The field is defined with two parts, a precedence value and the ToS bits. The precedence value occupies the leftmost 3 bits and was meant to provide a form of priority queuing. The ToS bits specified how the network should make trade-offs between throughput, delay, reliability, and cost. However, there also is no guarantee for strict deadline-sorted hard real-time communication.

Like the above-mentioned QoS architectures and protocols, in our work, we also established a way (virtual link) between source nodes and destination nodes by applying admission control based upon the requested QoS. However, in our protocol, we focus on real-time industrial systems such as factory automation, distributed control systems and embedded systems. In these fields, the following feature must be considered: 1. Hard real time communication should be guaranteed in service lifetime, 2. Modifications in the Ethernet hardware are not needed, 3. Still support existing best effort communication over standard Ethernet.

As a result, we provide a simple and practical way of design and implementation of an

Ethernet protocol to satisfy above all requirements. A key strategy to realize hard real-time communication is “bypassing” of TCP/IP suites. This makes considerably reduce the dwell time in the nodes, and increase the achievable data frame rate by evasion of the non-deterministic behavior inherent in the TCP and IP stacks. This is the main point of our work.

One of the goals in our work is to use cheap existing hardware and software whenever possible. Customized hardware would drive up the price of devices considerably and take up a lot of time to design and implement.

In our star-like network architecture, every end-node is connected with a private virtual link to a switch, so that there is a private traffic link for each direction. Then collisions can no longer occur on any network cable.

The real-time capabilities can now be embedded inside the switch. Present-day switches employ a technique called store-and-forward to transfer packets from one port to another, using per-port buffers for packets waiting to be sent on that port. But congestion may occur when one node is suddenly receiving a lot of packets from the other nodes. Current switches do not provide any guarantees as to which packets will be sent first. We solved this by providing a switch with bandwidth reservation capabilities inside the switch, and we used Earliest Deadline First (EDF) scheduling [19] to make decisions as to which packets are forwarded first. This provides guarantees for both bit rates and strict delivery deadlines.

Only a thin layer (RT layer) is added between the Ethernet protocols and the TCP/IP suite in each end nodes and switch to support guarantees for hard real-time traffic. The added RT layer in the nodes enables applications to reserve real-time channels on the network subject to a feasibility analysis. In this way we guarantee bandwidth for real-time traffic. Real-time channel (RT channel) is established between source nodes and destination nodes according to the RT layer, not defines a way of encapsulating other layer 2 and layer 3 protocols that the other protocols does.

In the proposed protocol, there are no modifications of the Ethernet hardware on the network interface cards, and coexists with TCP/IP suites. Which means that non-real-time nodes (nodes without RT layer added) can coexist in the network, and support for co-existing standard TCP/IP and UDP/IP Internet traffic without disturbing the real-time traffic. Therefore, the LAN with the proposed protocol can be connected to the existing Internet networks. It can be adopted in hard real-time applications such as embedded systems, distributed industrial systems, parallel signal processing and robotics.

An Ethernet LAN using the switch was constructed and several experiments have been performed to evaluate the proposed work. Comparing with the conventional hard real-time communication protocols, the proposed Ethernet protocol has better real-time performances, and meets the requirements of reliability for hard real-time systems.

2. Hard Real-time communication support

In this section, design and implementation of a switched Ethernet protocol for hard real-time communication are discussed. In subsection 2.1, outline of the network architecture is introduced. In subsection 2.2, we describe the RT channel establishment by applying the admission control. Subsection 2.3 illustrates the management of hard real-time traffic and best-effort traffic. Lastly, in subsection 2.4, we elaborate feasibility analysis for RT channel establishment and focus on scheduling of real-time frames.

2.1. *Network architecture*

We applied a full-duplex switched Ethernet which is connected to existing Internet.

Switches enable flexible network configuration of multiple and simultaneous links between various ports. A switched Ethernet enables some key benefits over traditional Ethernet, such as full duplex and flow control. Furthermore, it directs network traffic in an efficient way, establishing a kind of direct descent of communication between two ports of each frame structure.

In our work, a key strategy to realize hard real-time communication is *bypassing* of TCP/IP suite. In order to manage this bypass, both the switch and the nodes have software --- real-time layer (RT layer) added between the Ethernet protocols and the TCP/IP suite in the OSI reference model. All nodes are connected to the switch and nodes can communicate mutually on the logical real-time channels (RT channels), it is a virtual connection between two nodes of the system respectively. A node can either be real-time node or non real-time node depending on which level QoS is required. Non-real-time node can coexist in the network without disturbing the real-time traffic. MAC function, frame buffering and the concentrated transmission arbitration is included in the switch. Therefore, switch has the overall responsibility both for set-up of RT channels and for online control of packets passing through the switch.

The RT layer do-nothing to non real-time frames and makes them go through the ordinary circuit with TCP/IP suites. Namely, the RT layer is compatible with existing TCP/IP and handles both real-time and non real-time traffic depending on QoS.

2.2 *RT channel establishment*

Before the real-time traffic is transmitted, the RT channel should be established. The establishment of RT channel is including request and recognition communication after the source nodes, destination nodes and switch have agreement with channel establishment. As new real-time requests are made, they go through an admission control module that determines if there is sufficient network bandwidth available to satisfy the request of the node. Admission control is the problem of deciding which requests to accept and which to reject based upon the supported QoS, with the goal of maximizing the total profit accrued by the accepted requests. In other words, admission control is the problem of finding a feasible solution with maximum profit.

As for the establishment of an RT channel, each connection request is specified by two distinct nodes: the sending node and the receiving node in the network that want to communicate. Each transmission request has a certain bandwidth requirement and some time specification given by its starting time and its duration. If the network establishes a transmission request, it first decides on a path from the sending node to the receiving node of that transmission, through which the transmission is being routed. Then it allocates the requested amount of bandwidth on all links along that path during the time period in which the transmission is active.

Figure 1 describes the establishment of an RT channel. The RT channel should be established between a source node and a destination node according to the RT layer that is added between the Ethernet protocols and the TCP/IP suite. Each node is connectable to multiple sending and receiving RT channels. When a node wants to send hard real-time frames, it directly accesses the RT layer. The RT layer then sends "RT channel establishment request" to the RT traffic management in the switch. The switch then evaluates the feasibility of traffic schedule of a path from the sending node to the receiving node of that transmission, by applying the admission control. Namely, the switch evaluates the feasibility of traffic schedule between a requesting node and a switch plus between a switch and a destination node. If the schedule is feasible, the switch responses with the network schedule parameters (see Fig.2) to the sending node. Otherwise, the switch sends out a set of recommended control parameters to the sending node. These control parameters are suggested based on the status of switch queue and the active queue control law.

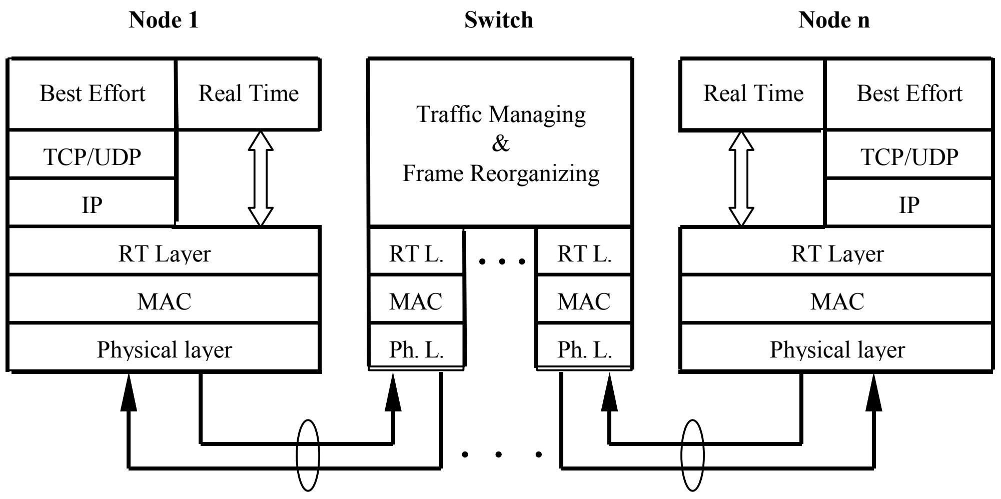

Fig. 1. Real-Time channel establishment.

Source	Type:	Response:	Connect	RT channel
MAC addr.	Response packet	0: Reject	request ID	ID
6 bytes	1 byte	1: Accept	1 byte	2 bytes

Fig. 2. Response parameter.

2.3 *Traffic management*

After RT channel is established, only real-time data traffic from the end-node bypasses the TCP/IP stack and thus considerably reduces the dwell time in the nodes, and increases the achievable data frame rate by evasion of the non-deterministic behavior inherent in the

TCP and IP stacks. Here dwell time of a node refers to one of the substantial influence factors for the real-time performance. An RT channel crosses two physical links: one from the source node to the switch, and the other from the switch to the destination node (uploads and downloads, respectively). The RT channel then provides real-time guarantees for both the upload and the download.

Besides hard real-time traffic, our Ethernet network protocol allows for best-effort traffic which does not affect the transmission of hard real-time packets. Namely, best-effort traffic (non real-time or soft real-time traffic) come from best-effort protocols (HTTP, SMTP, FTP, etc.) uses the services of the TCP/IP protocol suites and put in an FCFS-sorted (First Come First Serve) queue in the RT layer. In order to achieve this, best-effort traffic is allowed when no hard real-time packets want to transmit. If a best-effort sender needs higher bandwidth to send a large amount of data (for example, a long packet), it tries to make an additional reservation and transmit its data immediately after reservation. If the time is over and the long packet did not finished yet, it tries to make a reservation again.

When a hard real-time packet becomes ready to transmit again, the RT channel management immediately interrupts the best-effort traffic and goes to the corresponding node, so that the hard real-time traffic may start. According to the RT layer, the last node visited for best-effort traffic should be remembered, so the next round of best-effort traffic packet can start off at that node.

Because there are two different output queues for each port on the switch, “frame recognizing” is necessary. On that account, the switch has two MAC addresses: one is for control traffic (e.g., RT channel request frames); and the other is for real-time traffic over RT channels. And then, the switch will be able to recognize the different kinds of frames:

control frames, real-time data frames and non real-time data frames that come from TCP/IP stacks. The end-nodes recognize control frames by reading MAC source address that is set to the switch address. Non real-time frame carries the final destination MAC address in the Ethernet header when it leaves from the source node.

Another important role of the RT layer is to handle “RT data frame processing”. The RT layer makes the switch recalculate the Ethernet cyclic redundancy check (CRC) of an incoming real-time frame before putting it to the correct deadline-sorted output queue (see Fig. 3). This will also be helpful to increase the reliability of the real-time data frames. The checksums of non real-time frames do not need to be recalculated.

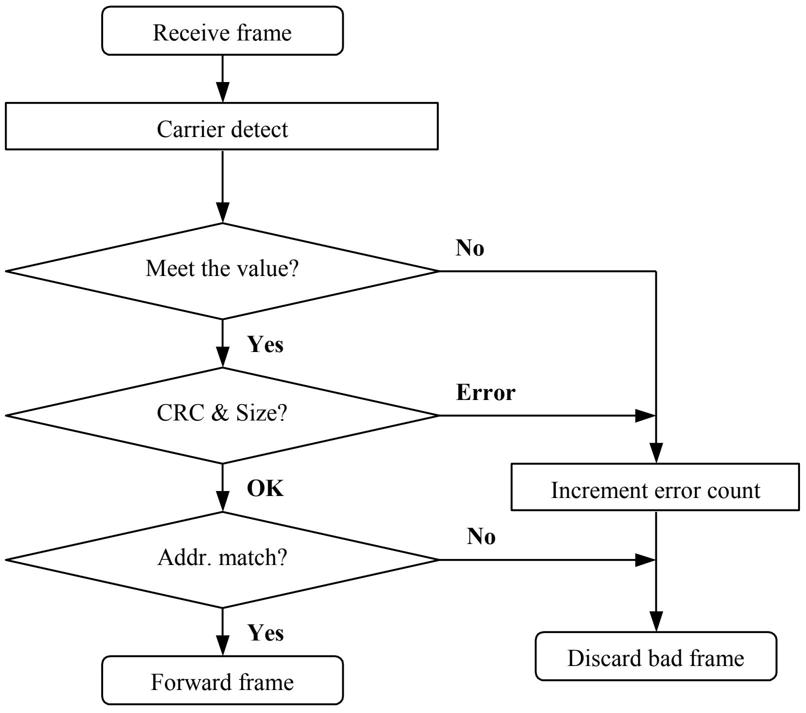

Fig. 3. Real-time data frame processing.

2.4 *Scheduling of real-time frames*

In this subsection, we elaborate feasibility analysis for RT channel establishment and scheduling for traffic management.

We assume that Node 1 (source node) wants to send real-time traffic to Node 2 (destination node). The real-time guarantee is supported by RT channel, and the scheduling of real-time frames in the switch and end-node is made according to EDF theory. First, we check the feasibility of the real-time traffic according to calculate the total utilization of all the request frames. RT channel of the *i*-th task is characterized by $\{T_{pd,i}, C_i, T_{d,i}\}$; where $T_{pd,i}$ is period of the data, C_i is time required to complete the execution of the task per the period, $T_{d,i}$ is the relative deadline used for the end-to-end EDF scheduling.

The task period $T_{pd,i}$ can be described as:

$$T_{pd,i} = T_{n1,i} + T_{n2,i} + T_t \tag{1}$$

where $T_{n1,i}$ and $T_{n2,i}$ are the deadlines of each real-time frame for upload and download, respectively; and T_t is the total delay introduced by the switch. In a fully switched Ethernet there is only one equipment (end-node) per each switch port. The total delay introduced by a switch including:

. The switching latency (including traffic classification and switch fabric set-up time),

. The frame forwarding latency which depends on the forwarding mode and eventually on the frame length if the "store & forward" mode is running,

. The buffering delay when the frame is queued.

The switching latency is a fixed value which depends on the switch performance and often provided by the switch vendor; the frame forwarding delay can be obtained knowing in which mode the switch is running; buffering delay exists in a switch whatever the switch is with full wire-speed or not. In fact message buffering occurs whenever the output port cannot forward all input messages on time. However, for the input traffic, scheduling analysis can give the worst-case buffering delay, providing thus the hard real-time guarantee.

Because the system is full duplex, for each link we consider two independent tasks; one is about the download parts through the link, and the other is the upload parts. The task of the upload link and download link is then executed with the set of instruction queue in the order decided by the switch.

According to EDF theory, the total utilization of all the request frames is then calculated as:

$$U = \sum_i \frac{C_i}{T_{pd,i}} \tag{2}$$

Suppose $T_{pd,i} \le T_{d,i}$ for simplicity, it is well known that EDF scheduling is feasible if and only if $U \le 1$.

If the test for task *i* succeed and real-time channel is established, real-time data frame bypasses the TCP/IP stacks and put in a deadline-sorted queue scheduled by RT layer in the switch and end-nodes according to the EDF theory.

The processes of hard real-time traffic and best-effort traffic transmission is shown in Fig. 4, and is summarized as follows:

1. When the switch received a packet from an end-node, it recognizes which application the packet is come from (real-time or best-effort).

2. If the packet come from the real-time application, then the RT layer interrupt transmitting best-effort traffic immediately so that the hard real-time traffic may start. And

the information for last transmitted queue of the best-effort traffic should be stored so that the next round of best-effort traffic can start off at that queue.

3-A. The switch will be able to recognize the different kinds of frames: control frames (e.g., RT channel request frames) and real-time data frames. If the received packet is the control frame, then the RT layer must go through an admission control module that determines if there is sufficient network resource (bandwidth and guaranteed time limit) available to satisfy the request of the node.

4-A. If a request is admitted, the switch answers to the requesting nodes with a set of network schedule parameters, and make RT channel virtual connection. And then allocates the requested amount of bandwidth and/or buffer space on this connection link.

4'-A. Otherwise, the switch sends out a set of recommended control parameters to the sending node.

5-B. After RT channel is established, hard real-time data is delivered through the circuit and bypassing the TCP/IP stacks by reading MAC addresses in response parameter.

6-B. The RT layer makes the switch recalculate the Ethernet cyclic redundancy check (CRC) of an incoming hard real-time frame before putting it to the correct deadline-sorted output queue. This will also be useful to increase the reliability of the hard real-time data frames.

7-B. Real-time data passed the above check is then put in a deadline-sorted queue scheduled by RT layer in the switch and end-nodes according to the EDF theory, and then,

8-B. Forward the deadline-sorted data to the destination node. The allocated resources are released when the connection is completed.

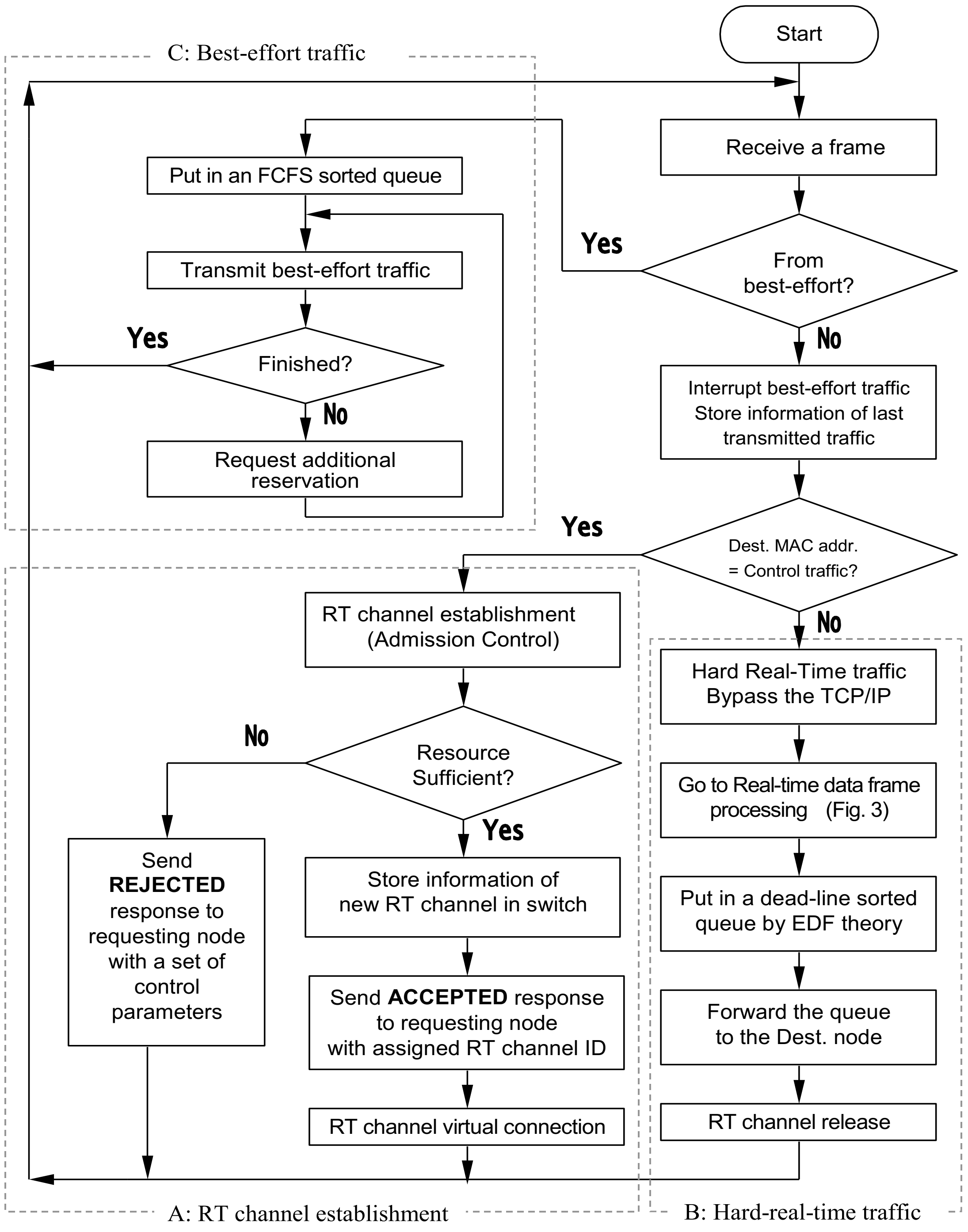
Start
C: Best-effort traffic
Receive a frame
Put in an FCFS sorted queue
Yes
From best-effort?
Transmit best-effort traffic
Yes
Finished?
No
No
Interrupt best-effort traffic
Store information of last transmitted traffic
Request additional reservation
Yes
Dest. MAC addr. = Control traffic?
No
RT channel establishment (Admission Control)
Hard Real-Time traffic
Bypass the TCP/IP
No
Resource Sufficient?
Go to Real-time data frame processing (Fig. 3)
Yes
Send REJECTED response to requesting node with a set of control parameters
Store information of new RT channel in switch
Put in a dead-line sorted queue by EDF theory
Send ACCEPTED response to requesting node with assigned RT channel ID
Forward the queue to the Dest. node
RT channel virtual connection
RT channel release
A: RT channel establishment
B: Hard-real-time traffic

Fig.4. Processes of traffic transmission

3-C. On the other hand, by carrying the final destination MAC address in the Ethernet header when leaving from the source node, non or soft real-time data is delivered through the circuit including the TCP/IP stacks in an FCFS-sorted queue, and transmit the traffics at the idle time of the schedule.

4-C. If a best-effort sender needs to send a large amount of data (for example, a long packet), it tries to make an additional cycling time reservation and transmit its data immediately after reservation. If the time is over and the long packet did not finished yet, it tries to make a reservation again.

3. Performance evaluation

In order to evaluate our work, we made a LAN with a full-duplex switched Ethernet and end-nodes, by using desktop computer with AMD-K7(tm) Processor 700MHz and several embedded Ethernet development boards which is produced by YDK Technologies Inc. that provides a hardware platform based on Altera® ACEX^{TM} devices (see Fig. 5). The boards include hardware and software components that provide network connectivity for the embedded systems. The components include:

1. A network-interface card (NIC) that plugged directly into the development board.
2. A SOPC Builder library component.
3. A C language library that provides an Ethernet Network-protocol stack.

We use a 5-ports Ethernet switch with full-duplex links at 100Mbps. The size of data packet is from 64 bytes to 1538 bytes. There are no modifications in the Ethernet hardware on the NIC (Network Interface Card). Ethernet development board is linked with PC via the Ethernet switch. In order to establish interaction and communication with the Ethernet development board, we downloaded the software (RT layer) to Flash Memory on the board.

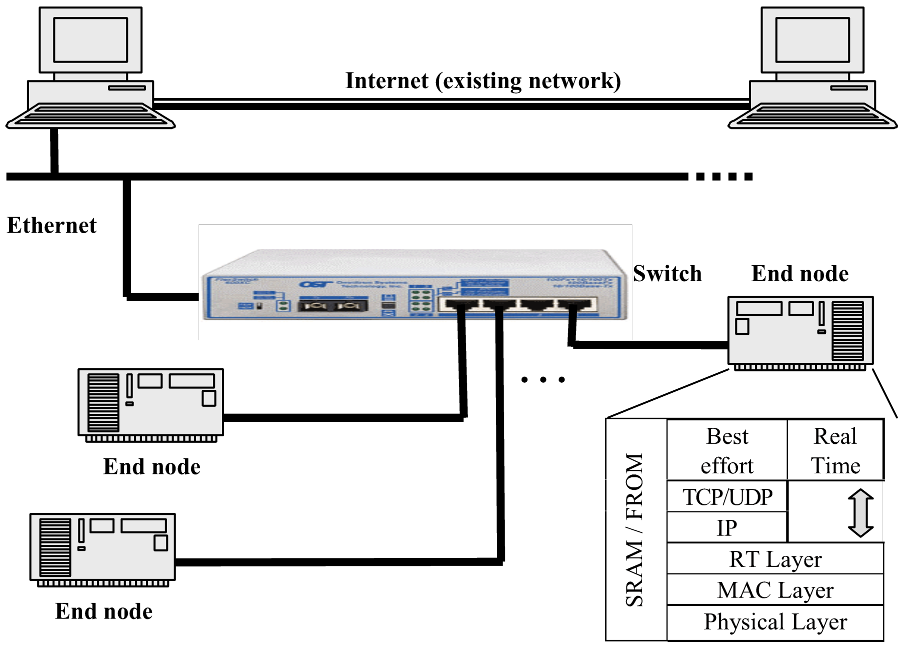

Fig. 5. LAN with full-duplex switched Ethernet.

When download the software to the flash memory, the system module pins are logically connected to pins on the ACEX device. The external physical pins on the ACEX device are in turn connected to other hardware components on the board, allowing the Nios embedded processor to interface with SRAM, FROM, LEDs, LCDs, buttons and switch.

Below we discuss about the transmission latency of the real-time frame in worst-case situation. When all RT-channel starts simultaneously, or all the messages that use all the capability allowances of the RT channel, RT channel equipped with the longest deadline will be scheduled at last so that it may have the worst-case latency. Here for all RT channels, the maximum latency is characterized by:

$$T_{m\_lat} = \max_{i}\{T_{n1,i} + T_{n2,i} + T_t\} \tag{3}$$

where $T_{n1,i}$ is the latency from source nodes to switch, $T_{n2,i}$ is the latency from switch to destination nodes, and T_t is the total latency of the switch.

Besides utilization and worst-case latency, another important performance is a runtime overhead: R_i defined as:

$$R_i = \frac{T_{pd,i} - L_i \times 8 / B}{T_{pd,i}} \tag{4}$$

where L_i is the length of data in a request frame, $L_i \times 8$ is the number of bits in the frame; $T_{pd,\ i}$ represents the period duration from the startup to the end of the frame, and B represents the Ethernet bandwidth.

Utilization, Data frame transmission latency and the frame runtime overhead can be obtained by implementing the proposed protocol to the LAN. Figure 6 illustrates the utilization on real-time data frame. From the figure we can learn that the trend of utilization is increasing while the traffic increases, until arriving at the peak value that is more than 90%. Sudden decreases happen on the curve sometimes, which is caused by some short frames having pad field whose utilization are lower than longer frames. The utilization curve is always smooth, because we assume the sufficient resources (bandwidth and time specification) have been obtained in our work, when the RT channel is established. Under this circumstance, there should be no overload exists in the real-time channel. The result shows that the deadlines have been met for all data because utilization of all data frames is less than 100% using EDF scheduling. Dynamic priority scheduling with the EDF algorithm has a distinct advantage over fixed priority scheduling: the schedulable bound for EDF is 100% for all task sets [20]. This means that we can fully utilize the computing power of the CPU. Embedded systems are in general fully loaded, as they attempt to get as close to 100% utilization as possible while still maintaining the necessary predictability.

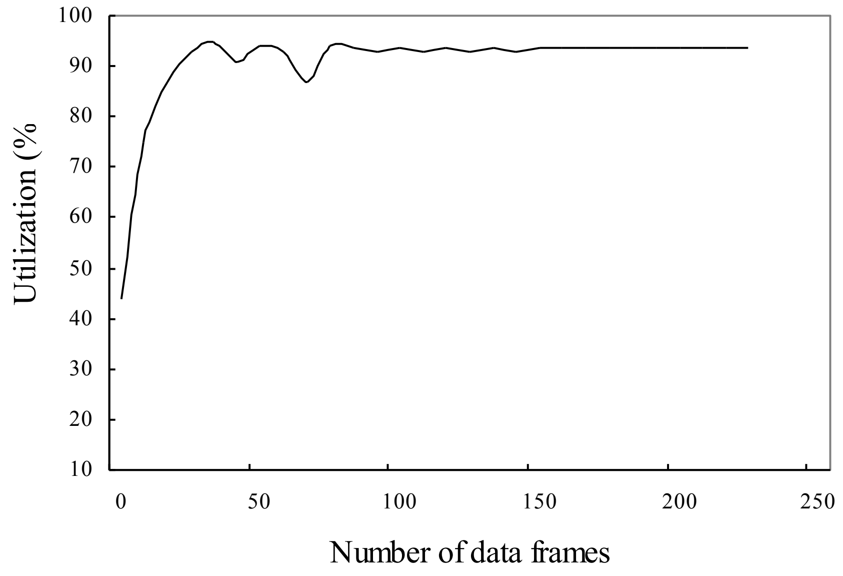

Fig. 6. Utilization on real-time data frame.

To the best of our knowledge, except a few implementations of hard real-time communication protocols on Ethernet, most of the protocols are generally soft real-time, which means that there are few protocols provides guarantees for both bit rate and strict delivery deadlines, so that it is difficult to compare them with the proposed hard real-time protocol. Therefore, we made performance comparison of the proposed protocol only with the hard real-time communication protocols: MIL-STD-1553B protocol and RTCC protocol. Figure 7 shows the comparison of the data frame transmission latency of these three kinds of hard real-time communication protocols. Even for the Ethernet frames that have the data field maximized (1538 bytes in IEEE 802.3 standard), the latency of the proposed protocol is about 620 microseconds. This latency is quite short in a LAN with a full-duplex switched Ethernet at 100 Mbps, and should be meet the demands of hard real-time communication for industrial distributed control systems.

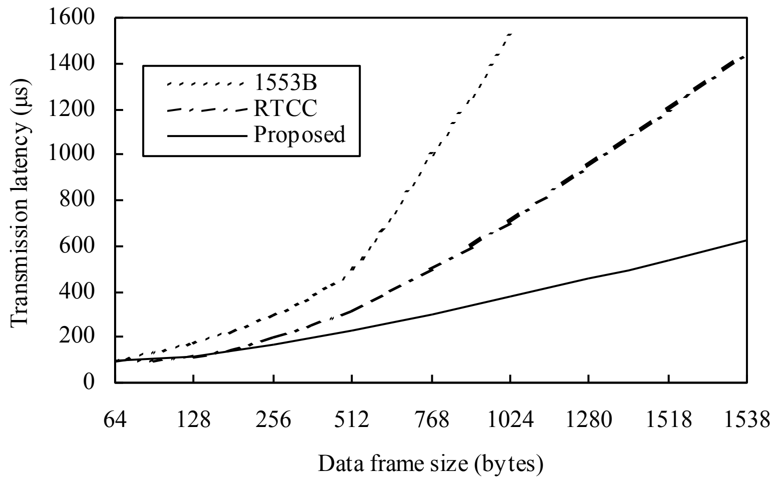

Fig. 7. Transmission latency of data frame.

Runtime overhead of data frames are demonstrated in Fig. 8. The figure shows that the runtime overhead of the proposed protocol is higher than the other hard real-time supported protocols at the small-sized data frame. However, as the data frame size become larger (from about 900 bytes), the proposed protocol has more better runtime overhead than the other protocols.

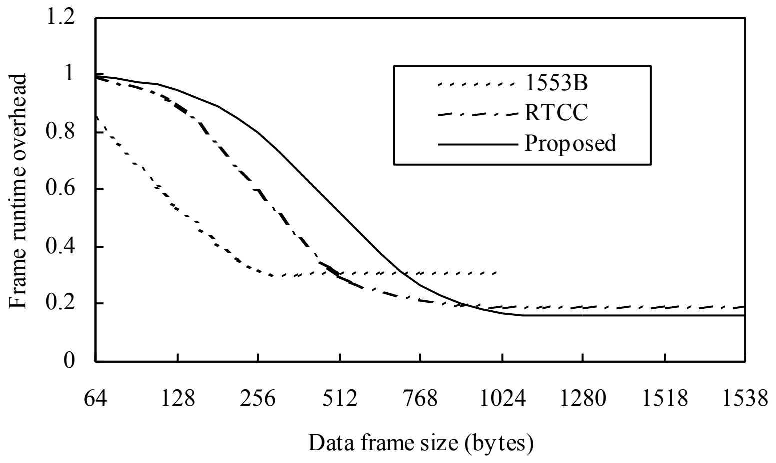

Fig. 8. Runtime overhead of data frame.

In both experiments, only MIL-STD-1553B protocol used 1Mbps Ethernet because it is the nominal speed of the protocol.

From these figures it is noticed that the proposed protocol can get better real-time performance than the other hard real-time communication protocols.

4. Conclusions

In this paper, we have presented a protocol to support hard real-time communication over non real-time LAN technology, with a switched Ethernet based network concept. For the network configuration, based on establishment of the RT channel strategy, both switch and end-nodes have an RT layer added to support hard real-time traffic based on EDF algorithm. Hard real-time traffic from the end-node bypasses the TCP/IP stacks and thus considerably speed up real-time communication. The real-time traffic schedule is performed according to dynamic-priority EDF algorithm without modifications in the Ethernet hardware, therefore, it is flexible and efficient.

We have constructed a simple Ethernet LAN with the proposed protocol and evaluated the protocol. Through the comparison with some conventional hard real-time network protocols, we have shown that the proposed Ethernet protocol has better real-time performances, higher bandwidth and a larger data frame, and meets the requirements of reliability for hard real-time systems.

In the proposed work, there are no modifications in the Ethernet hardware on the NIC. This allows connecting the Ethernet LAN to existing Internet networks. Thus, it can be adopted in hard real-time applications such as embedded systems, distributed industrial systems, parallel signal processing and robotics.

References

1. G. Buttazzo: Hard Real-time Computing Systems, Predictable Scheduling Algorithms and Applications - Real-Time Systems Series 2nd Edition, Springer Verlag, (2004)
2. J. Stankovic and K. Ramamritham: Hard Real-Time Systems, IEEE Computer Society Press, (1988)
3. C. Krishna and K.G. Shin: Real-Time Systems, McGraw-Hill International edition, (1997)
4. M. Joseph: Real-Time Systems, Prentice Hall, (1996)
5. http://www.condoreng.com/support/downloads/tutorials/MIL-STD-1553Tutorial.PDF
6. Z. P. Wang, G. Z. Xiong, J. Luo, M. Z. Lai and W. Zhou: A hard real-time communication control protocol based on the Ethernet, Proceedings of 7th Australasian Conference on Parallel and Real-Time Systems (PART 2000), pp.161-170, (2000)
7. J. Loeser and H. Haertig: Low-latency hard real-time communication over switched Ethernet, Proceedings of 16th Euromicro Conference on Real-Time Systems (ECRTS'04), pp. 13-22, (2004)
8. S. Ouni and F. Kamoun: Hard and soft real time message scheduling on Ethernet networks, Proceedings of the 2nd IEEE International Conference on Systems, Man and Cybernetics of the twenty-first century, 6, (2002)
9. H. Zhang: Service disciplines for guaranteed performance service in packed switching network, Proc. of the IEEE, vol.83, no.10, (1995)
10. D. Ferrari and D. Verma: A scheme for real-time channel establishment in wade-area networks, IEEE Journal of Selected Areas in Communications, vol.8, no.3, pp.368-379, (1990)

11. L. Zhang: Designing a new architecture for packet switching communication networks, IEEE Communications Magazine, vol. 25, n. 9, pp. 5-12, (1987).
12. G. Agrawal, B. Chen, W. Zhao, and S. Davari: Guaranteeing Synchronous Message Deadlines with Time Token Medium Access Control Protocol, *IEEE Transactions on Computers*, Vol. 43, No. 3, pp 327-339, (1994)
13. Hoang, H., M. Jonsson, U. Hagström, and A. Kallerdahl: Switched real-time Ethernet with earliest deadline first scheduling - protocols and traffic handling, *Proc. of the International Parallel and Distributed Processing Symposium (IPDPS'02)*, (2002)
14. A. Leon-Garcia and I. Widjaja: Communication Networks - Fundamental Concepts and Key Architectures, McGraw-Hill Osborne, (2001)
15. M. Schwartz & T. E. Stern: Routing Protocols, in Computer Network Architectures and Protocols (Second Ed.), Ed. C.A. Sunshine, Plenum Press, New York / London (1989)
16. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss: An architecture for differentiated service, RFC 2475, (1998)
17. E. Rosen, A. Viswanathan and R. Callon: Multiprotocol label switching architecture, RFC 2005, (1997)
18. L. Zhang, S. Deering, D. Estrin, S. Shenker and D. Zappala: RSVP: a new resource reservation protocol, IEEE Network Magazine 30-9, pp.8-18, (1993)
19. C. L. Liu and J. W. Layland: Scheduling algorithms for multi-programming in hard real-time traffic environment, Journal of the Association for Computing Machinery, 20-1, pp.46-61, (1973)
20. Phillip A. Laptane, Real-Time System Design and Analysis, IEEE Press, third edition, (2004)