Implementetations of Ethernet-Like Protocols Utilizing Ethernet Technology For Real Time Simulation Networking

Michael Georgiopoulos
Department of Electrical Engineering
University of Central Florida, Orlando, FL 32816, USA

Jack Thompson
Institute for Simulation and Training
12424 Research Parkway, Suite 300
Orlando, FL 32826

Nicos Christou
Department of Electrical Engineering
University of Central Florida, Orlando, FL 32816, USA

Yousuf C. H. Ma
Department of Computer Engineering
University of Central Florida, Orlando, FL 32816, USA

Abstract

Recent advances in computer and communication technologies have made possible the interconnection of large number of real-time simulators via local area networks. One of the most popular network access protocols is Ethernet. In this paper we discuss some of the limitations of the Ethernet protocol when it is used to interconnect real-time simulation devices. We also introduce two modifications of Ethernet called Ethernet-1 and Ethernet-2 that remedy some of the shortcomings of the Ethernet protocol. It is worth noting that Ethernet-1 and Ethernet-2 are implementable in hardware.

1 Introduction

Recent breakthroughs in several computer/communications core technologies have made possible the interconnection of large number of real time simulators (special purpose hardware) via local area networks. Two main applications/advantages of such networks are:

1) To provide a cost effective tool for the training of personnel in applications involving interactions among mobile vehicles. Examples of such applications include training exercises for police forces, fire/ambulance services, and military combat fighting.

2) To provide an effective "test before you build" development tool to be used for evaluating proposed modifications in existing systems, as well as an aid in designing/developing new systems. Tactics and coordination strategies might also be simulated and evaluated before they are adopted in real life.

The networking of real–time simulators departs from the traditional use of a computer network, whose function would normally be to provide sharing of computing resources among multiple users (nodes) on the network. When used to interconnect real–time simulators, the network is used almost exclusively for communication of process state information between the simulators engaged in a training exercise.

There are many inherent limitations to using a network in this application. For example as the number of simulators on the network and the workload per simulator increases, there may be a deterioration in throughput and a degradation of other network performance parameters. If data packet delays through the network become too large, for example, the effectiveness of a real–time training simulation may be overly compromised due to the time–critical response requirements in the simulation true–to–life, action–requiring training scenarios. Depending upon the network communication protocol being used, there may also be an increase in the frequency of retransmitted and lost packets.

The Ethernet [1] is one of the most popular protocols used to interconnect devices in the local area network level. The Ethernet protocol though exhibits serious limitations when used to interconnect real–time simulation devices. In this paper, a number of modifications of the Ethernet protocol are introduced in an attempt to remedy some of its problems. It is worth noting, that none of the Ethernet modifications guarantee better performance than Ethernet for traffic loads above Ethernet's throughput. On the other hand, for traffic loads below Ethernet's throughput they exhibit a better delay performance than Ethernet. All the modifications of the Ethernet protocol examined in this paper are implementable in hardware.

2 Motivation

As we mentioned in the introduction the networking of real–time simulation training systems at the local area network level imposes certain performance constraints that are not currently satisfied by the Ethernet protocol.

First, prioritization of packets or nodes in the network is absolutely necessary. For example, nodes that simulate highly moving vehicles, such as aircraft, need to access the network more frequently than nodes simulating highly moving vehicles. Besides that, a node may generate packets that require faster delivery time than other packets. In particular, a packet that contains information of a vecicle being hit should be successfully delivered faster than a packet containing information about the state of a slowly moving vehicle. Furthermore, in a local area network that supports voice and data communication among the simulators different priority has to be assigned to voice and data packets. Specifically, in some cases delay of data packets of the order of 100 ms are acceptable, while in most voice applications voice packets cannot experience delays larger than 15 ms. In the above scenario, voice packets should be given higher priority.

Secondly, in a real–time, interactive simulation scenario the network is used almost exclusively for communication of state information between the simulators engaged in the training exercise. Upon a state change, a simulator (node) on the network sends the information concerning its state to other nodes on the local area network (LAN). Each new state results in the generation of a new data packet at the application layer (i.e. at the node level). The packet is then submitted to the data link layer in order to start the process of its transmission. In this context, it is advantageous if the arrival of a new packet at the application layer, carrying the most current state of the node, initiates the abortion of the transmission process of the old packet. The discarding of the outdated packet helps speed up the transmission of the most current state of the node and does not waste network resources (i.e, channel bandwidth) to transmit unnecessary information. It is obvious that replacement of old state update packets will occur more frequently in a heavily loaded network. In real time simulation networks that support extrapolation of state information (such as SIMNET) the above scheme imposes extrapolation of state information only when it is absolutely necessary (i.e., only when the network is heavily loaded and some state update packets are lost due to the packet abortion process). The abortion of old packets is also necessary in voice applications where voice packets cannot be allowed to experience long delays because of the requirement not to empty receiver buffers so that "stream data" can be played out at the receiver (telephone earpiece).

In this paper, we intend to examine two modifications of the Ethernet protocol. The first modification, Ethernet-1, allows us to prioritize nodes in the network. The second modification, Ethernet-2, aborts the transmission of an old packet at the data link layer if a new packet arrives at the application layer with more current information. Ethernet-2 can also be used in cases where we want to incorporate packet prioritization in the network. Then, the arrival of a high priority packet at the application layer aborts temporarily the transmission process of a low priority packet residing at the data link layer.

3 Network System Configuration Model

Various choices exist for the implementation of a LAN (e.g., transmission medium, topology, access protocols, etc.) to interconnect simulation devices. In this paper, we present the results of an evaluation study for three Ethernet-like protocols (i.e., Ethernet, Ethernet-1, Ethernet-2). The configuration used for all the above protocols is the bus topology. Figure 1 gives an example of an existing network configuration used to interconnect real-time simulation/training devices [2]. In this configuration, a number of nodes (simulators), typically eight, are connected through a multi-port transceiver to a single point on the coaxial cable (via a medium access unit). Each node follows the Ethernet or the Ethernet-1 or the Ethernet-2 protocol to access the bus. The performance evaluation of Ethernet, Ethernet-1 and Ethernet 2 is based on the network configuration of Figure 1 with 100 simulators (nodes).

4 Protocol Description

In the following we describe the Ethernet, Ethernet-1 and Ethernet-2 protocols. In **Ethernet**, when a node is ready to transmit a packet on to the network, it first monitors the network bus to determine whether any other transmissions from other nodes are in progress. If the node senses the network channel to be busy, it simply waits for the channel to become idle before attempting to transmit its packet. Once the channel is sensed idle, the node waits a prespecified amount of time (interframe gap) to assure the channel is clear and then begins transmitting its packet. During its own transmission, the node also monitors the channel in order to detect whether its packet is interfering (colliding) with packets from other nodes. If a collision is detected, each node involved in the collision transmits a bit sequence on to the network known as a jam signal, after which each node in the collision waits (backs-off) for a randomly generated amount of time before re-attempting its transmission. In particular, the backoff time is equal to $R \cdot \mathrm{SlotTime}$, where R is an integer uniformly distributed in the interval $[0, 2^{\min(10,n)})$ and Slot Time is a time constant usually taken to be 50 μs ; n is the number of unsuccessful node attempts to access the channel. If a packet experiences more than 16 collisions it is discarded from the network.

Ethernet-1 distinguishes the network nodes into two distinct classes. The low priority class and the high priority class. A high priority node implements the Ethernet protocol. A low priority class node implements the Ethernet protocol up to the time it collides for the first time. Then, it backs off by an amount of time equal to $R \cdot \mathrm{SlotTime}$, where now R is an integer uniformly distributed in the interval $[0, 2^{\min(10,n+K)})$ and K can take any value between 1 and 7. It easy to see that whenever a collision happens Ethernet-1 gives an advantage to the high priority nodes. Ethernet-1 is expected, compared to Ethernet, to improve the delay of the high priority nodes at the expense of increased delay for the low priority nodes. Note that Ethernet does not discriminate between high and low priority nodes.

Ethernet-2 follows the rules of the Ethernet protocol with one exception. When a new packet arrives at the application layer of any node in the network it aborts the transmission of any old, unsuccessfully transmitted packet which resides in the data link layer. Note that Ethernet discards packets only after 16 unsuccessful transmission attempts, but Ethernet-2 in addition to the aforementioned Ethernet's discarding process discards packets in cases where a new packet arriving at the application layer finds an old, unsuccessfully transmitted packet in the data link layer. Ethernet-2 is expected, compared to Ethernet, to improve the delay of the packets that are eventually successfully transmitted at the expense of increased packet loss rates.

5 Results

In Table 1, we show the average packet delay for low priority and high priority nodes in Ethernet-1 when $K = 1$ and for different percentages of high and low priority traffic. In the same Table, the average packet delay in Ethernet is listed for comparison purposes. These results justify our expectations (i.e., high priority nodes experience a decrease in the average packet delay). It is worth noting, by observing the results in Table 1, that even low priority nodes experience smaller delays in Ethernet-1 than in Ethernet for medium to high traffic loads. This is due to the fact that for medium to high traffic loads the standard back-off Ethernet algorithm is not capable to efficiently separate the collided nodes so as to avoid future collisions. For example, in Ethernet if two nodes collide for the first time there is a probability of 0.25 that they will collide immediately after their first collision. If more than two nodes collide for the first time (a more likely event as the traffic load increases) the probability that another collision will follow immediately after the first collision is greater than 0.25. On the contrary, Ethernet-1 by allowing the low priority nodes to follow a different back-off algorithm manages to reduce the probability of having two successive collisions. In general, Ethernet-1, in the case of a collision event manages to separate the collided nodes more efficiently than Ethernet. In Table 2, the average packet delay for low and high priority nodes in Ethernet-1, when $K = 2$ and for different percentages of high to low priority traffic is depicted. Comparing Tables 1 and 2 we see that by choosing a larger K value we can reduce the average packet delay of the high priority nodes at the expense of increased average packet delay for the low priority nodes.

Table 3 shows the average packet delay induced by Ethernet-2 where a new packet arriving at the application layer of a node aborts the transmission process of an old, unsuccessfully transmitted packet residing in the data link layer. As expected, in Ethernet-2, the average packet delay is decreased significantly at the expense of increased packet loss rates. For example, at traffic load of 6 Mb/s Ethernet-3 induces an average packet delay of 0.66 ms and a packet loss rate of 4.9 % compared to Ethernet's average packet delay of 5.9 ms and packet loss rate of 0.038 %. Similarly at traffic load of 7.5 Mb/s Ethernet-2 induces an average packet delay of 1.1 ms and packet loss rate of 18 % compared to Ethernet's average packet delay of 57.1 ms and packet loss rate of 1.3 %. A packet loss rate of 18 % is substantial, especially if a node simulates a high speed vehicle. One way to remedy Ethernet-2's high packet loss rates is to incorporate priorities. Then, high priority nodes corresponding to fast moving vehicles will experience reasonable packet loss rates.

6 Implementation Aspects of Ethernet-1, Ethernet-2

Ethernet-1 and Ethernet-2 can be implemented using the INTEL 82586 LAN communications controller [3].

The 82586 meets the IEEE 802.3 ([4]) standard: 10 megabits per second bit rate and 9.6 microseconds Interframe Spacing. The 82586's programmable network parameters allows it to serve as a controller for a wide range of carrier sense multiple access with collision detection (CSMA/CD) type LANs. Many parameters are configurable including all framing parameters, Slot Time and Interframe Spacing. Furthermore, we can implement various Back-off retransmission strategies with the 82586. Among them is the alternate back-off algorithm (Ethernet-1) and the linear priority back-off algorithm (see [3]). Finally, an arrival of a packet at the application layer can stop the transmission of a packet at the data link layer through some kind of abort mechanism supported by the 82586 (Ethernet-2).

7 Conclusions

In this paper we focused on the problem of networking large number of real-time vehicle simulators and we presented the results of a comparison study of three different protocol access methods (Ethernet, Ethernet-1 and Ethernet-2).

The motivation for our work stemed from the fact that within the class of contention protocols Ethernet does not guarantee optimum performance when used to interconnect real time simulation devices. We saw that Ethernet-1, compared to Ethernet, guarantees a better delay performance than Ethernet for the high priority nodes at the expense of increased delay for the low priority nodes. Furthermore, Ethernet-2, compared to Ethernet, achieved a better delay performance at the expense of increased packet loss rates. Ethernet-1 and Ethernet-2 were found to be hardware implementable by utilizing the INTEL 82586 LAN controller. Other modifications of the Ethernet protocol that are implementable with the INTEL 82586 are also worth examining.

Acknowledgements

This research was supported by the U.S. Defense Advanced Research Projects Agency (DARPA) under contract N62339-89-C-0043.

References

[1] R. Metcalfe and D. R. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks", *Communications of ACM*, Vol. 19, No.7, 1976, pp. 395-403.

[2] A. Pope, "The SIMNET Network and Protocols", *BBN Report*, No. 7102, BBN Communications Corporation, MA, July 1989.

[3] INTEL Microcommunications Handbook, Components, Boards, Systems, Software for Microcomputer Communications; Order Number 231658-002, 1987.

[4] ANSI/IEEE- International Standard 8802/3 "Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specification", *IEEE Computer Society Press*, 1985.

[5] M. Bassiouni, M. Georgiopoulos and J. Thompson, "Real Time Simulation Networking: Network Modeling and Protocol Alternatives", *Proceedings of the 11th Interservice/Industry Training Systems Conference (I/ITSC)*, November 1989, pp. 52-61.

About the Authors

M. Georgiopoulos received his Ph.d degree in Electrical Engineering from the University of Connecticut in 1986. He is currently an Assistant Professor in the Electrical Engineering Department at the University of Central Florida. His current research interests include communication networks, spread spectrum communications and neural networks.

J. Thompson received his B.S. degree in Electrical Engineering from the University of Central Florida in 1978. He is currently a Research Associate for the Insitute of Simulation and Training (IST) at the University of Central Florida. Mr. Thompson has technical responsibility for all IST activities involving computer networking.

N. Christou is a graduate student in the Electrical Engineeering Department of the University of Central Florida. He received his Masters degree in December 1989 from the University of Central Florida. He is currently working towards his Ph.d. His research interests lie in the area of computer/communication networks.

Y. C. H. Ma is a Masters student in the Computer Engineering Department at the University of Central Florida. His research interests include computer networks and computer vision.

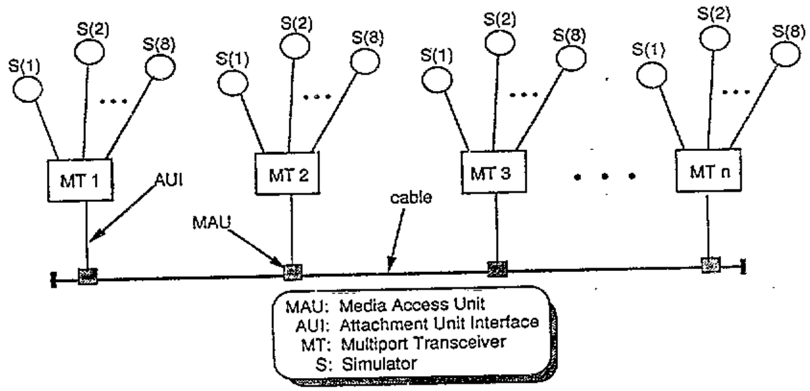

FIGURE 1. Bus Network Topology System Configuration

Table 1

Average Packet Delay in Ethernet-1 with $K = 1$, and Ethernet

TT (Mb/s)	HPT %	LPT %	HPD (ms)	LPD (ms)	ETH-D (ms)
1.5	10	90	0.120	0.125	0.124
1.5	20	80	0.120	0.125	0.124
1.5	40	60	0.120	0.125	0.124
3.0	10	90	0.151	0.177	0.172
3.0	20	80	0.155	0.176	0.172
3.0	40	60	0.158	0.179	0.172
4.5	10	90	0.348	0.445	0.439
4.5	20	80	0.343	0.430	0.439
4.5	40	60	0.346	0.456	0.439
6.0	10	90	4.019	4.847	5.934
6.0	20	80	3.479	4.488	5.934
6.0	40	60	3.637	4.510	5.934
7.5	10	90	35.97	38.82	57.238
7.5	20	80	34.28	37.47	57.238
7.5	40	60	37.82	41.20	57.238

Legend

TT:	Total Traffic in megabits per second
HPT %:	Percentage of High Priority Traffic in Ethernet-1
LPT %:	Percentage of Low Priority Traffic in Ethernet-1
HPD:	Average Packet Delay of High Priority Traffic in Ethernet-1
LPD:	Average Packet Delay of Low Priority Traffic in Ethernet-1
ETH-D:	Average Packet Delay in Ethernet

Table 2

Average Packet Delay in Ethernet-1 with $K = 2$, and Ethernet

TT (Mb/s)	HPT %	LPT %	HPD (ms)	LPD (ms)	ETH-D (ms)
1.5	10	90	0.118	0.128	0.124
1.5	20	80	0.119	0.128	0.124
1.5	40	60	0.119	0.128	0.124
3.0	10	90	0.145	0.193	0.172
3.0	20	80	0.145	0.193	0.172
3.0	40	60	0.153	0.198	0.172
4.5	10	90	0.214	0.497	0.439
4.5	20	80	0.254	0.533	0.439
4.5	40	60	0.278	0.513	0.439
6.0	10	90	1.954	6.830	5.934
6.0	20	80	1.687	6.448	5.934
6.0	40	60	2.134	6.454	5.934
7.5	10	90	14.35	69.37	57.238
7.5	20	80	17.13	88.50	57.238
7.5	40	60	19.36	93.48	57.238

Legend

TT:	Total Traffic in megabits per second
HPT %:	Percentage of High Priority Traffic in Ethernet-1
LPT %:	Percentage of Low Priority Traffic in Ethernet-1
HPD:	Average Packet Delay of High Priority Traffic in Ethernet-1
LPD:	Average Packet Delay of Low Priority Traffic in Ethernet-1
ETH-D:	Average Packet Delay in Ethernet

Table 3

Average Packet Delay in Ethernet-2 and Ethernet

Traffic Load (Mb/s)	Ethernet-2 Ave Delay (ms)	Ethernet Ave Delay (ms)
1.5	0.123	0.124
3.0	0.168	0.172
4.5	0.291	0.439
6.0	0.660	5.934
7.5	1.093	57.238