Real-Time Communication over Shared Media Local Area Networks

Rainer Steffen, Markus Zeller, Rudi Knorr
Fraunhofer Institute for Communication Systems (ESK)
Hansastr. 32, D-80686 Munich, Germany
Rainer.Steffen@esk.fraunhofer.de

Abstract

In this paper, we will present a method to enable hard real-time communication with guaranteed quality of service over shared media Ethernet. This method is based on an extension of the standard medium access control by a deterministic token-passing protocol that could also be applied to other network technologies such as HomePNA or Powerline. We show that the assignment of QoS- and real-time parameters (data rate & delay) can be done very flexibly and that the timing behavior can be analyzed mathematically and by simulations.

1. Introduction

Ethernet is the most frequently used wired local area network technology today. It is widely deployed in professional as well as in private environments. Ethernet networks can be built as switched networks, shared media networks (connected by hubs) or a combination of both. Shared media Ethernet has some advantages compared to switched networks, as it builds a logical bus topology that is flexible and easy to install. A bus topology is very appropriate for in-house networking because there is no need for concentrated instances (the switches) that merge and switch the data of the different devices. The field of applications covers all kinds of time critical multimedia distribution (audio, video) and hard real-time data transmission. It can be deployed for in-house networking as well as for distributed computer-controlled systems.

Unfortunately, Ethernet offers no mechanisms for negotiating and guaranteeing quality of service (QoS) and real-time conditions. Further, shared media Ethernet has no deterministic behavior because it utilizes the CSMA/CD (Carrier Sense Multiple Access with Collision Detection) medium access control mechanism. Thus, an apriori determination of throughput and timing characteristics of the different applications and the whole network is not possible [1].

This paper describes a method to enable shared media Local Area Networks to transmit data with a guaranteed QoS and hard real-time demands. The parameters could be set very flexibly to support simultaneous time critical services and applications.

The described QoS and real-time enhancements are not restricted to use within Ethernet, but could also be applied on other shared media network technologies such as HomePNA, Powerline Communication or even wireless networks.

The remainder of the paper is organized as follows: section 2.1 briefly describes related work that has been carried out to enhance the medium access control of wired shared media networks; section 2.2 presents a new token-passing protocol and section 2.3 describes the timing behavior of the new mechanism; section 3 presents some performance results; section 4 concludes the paper.

2. Real-Time Local Area Network

The described field of applications (multimedia, automation) necessitates a flexible negotiation of QoS-parameters and real-time demands. It must be possible to assign different parameters to the particular applications and services.

2.1 Related Work

Different proposals have been carried out to enhance the medium access control of Ethernet. Some systems improve the CSMA/CD performance by modifying the arbitration method [1]; others restrict the total utilization of the network or smooth the traffic to prevent long delays and too many collisions [2][3]. Another solution is to overlay CSMA/CD by a more adequate medium access control mechanism [4] [5].

Although all these systems show an improvement, they are not completely appropriate for QoS and real-time applications. This is due to missing deterministic, low flexibility, low total utilization, or the missing hard real-time support. Thus, a new mechanism, based on an overlaid token-passing protocol, will be introduced.

2.2 Token-Passing Protocol

To control the access to the shared medium, a token-passing protocol is deployed instead of the uncontrolled CSMA/CD mechanism of Ethernet. The protocol can be implemented in software, so that standard commercial off the shelf (COTS) hardware can be used. All devices connected to the shared media network comprise a quality of service sublayer that is located between the logical link control and the medium access control (figure 1). The QoS sublayer overlays the standard CSMA/CD mechanism of Ethernet and controls the access to the network by a token-passing protocol.

Within the whole shared Ethernet segment, there is exactly one token rotating. The holder of the token has the right to access the medium for a specified amount of time. After that time, it has to release the token to the next device. By the frequency of the token occupancy and the length of time the device is allowed to hold the token, it is possible to guarantee the negotiated QoS and real-time conditions. Since there is always just one token on the segment, the medium access is absolutely deterministic and collisions cannot occur. As a consequence there is no degradation of the total throughput at high traffic loads and the delay of the frames can be calculated and bounded. The length restriction, which is caused by the maximum signal propagation delay to detect collisions, is suspended. It is possible to connect an unlimited number of hubs in succession.

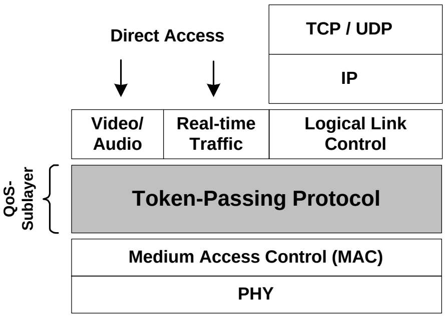

Figure 1: Extended Ethernet protocol stack

The protocol works completely decentrally so that no central instance (e.g. master) is needed. Once at least two stations are connected together, they identify one another and initialize the token-passing procedure. New devices can be applied and connected stations can be signed off. Further, the protocol monitors the status of the network in a decentralized manner to recognize errors (e.g. token loss due to transmission errors) and eliminate these errors. The monitoring is done by timers; in unreliable networks the token transmission has to be confirmed by an acknowledge frame. This allows a fast retransmission of lost tokens and detection of network misconduct.

2.3 Timing Behavior of the System

In addition to the described protocol, it is necessary to specify the timing behavior of the token-passing procedure. The frequency of the token arrivals and the token holding times must be computed. To guarantee the QoS and the hard real-time demands, the traffic is classified into *synchronous* and *asynchronous* streams similar to [6] and [7]. It is assumed that a synchronous stream generates data frames in isochronous time intervals $T_{p_i}^k$ with a maximum frame length of $C_{p_i}^k$ (in seconds). The generated frames will not be segmented into smaller units. To meet the hard real-time demands, these frames must be sent out before their deadline $D_{p_i}^k$ expires. The deadline must be chosen so that $D_{p_i}^k \geq T_{p_i}^k$. The most restrictive real-time requirement is $D_{p_i}^k = T_{p_i}^k$, and a frame must be sent out before a new frame is generated. Thus, a synchronous stream can be defined as $S_{p_i}^k = f(C_{p_i}^k, T_{p_i}^k, D_{p_i}^k)$. Each stream can have unique and independent parameters. A network device *k* may have several synchronous streams *i* with $\left(i = 1,2,...n_p^k\right)$ whereas n_p^k defines the number of synchronous streams of a station *k*.

A synchronous stream will be applicable for all kinds of QoS and real-time dependent applications such as multimedia transmission or industrial process control. When a new stream is negotiated, a synchronous bandwidth H_i^k is assigned to the stream that describes the length of time a device is allowed to hold the token. This time must be larger or equal to the maximum frame length $H_i^k \geq C_{p_i}^k$, so that at least one frame could be sent each time a token visits the stream. Additionally, an expected token rotation time *TTRT* is defined (time between two consecutive token arrivals).

An asynchronous stream $S_a^k = f(C_a^k, r_a^k)$ is defined as a function of the maximum asynchronous frame length C_a^k and the desired average data rate r_a^k. An asynchronous stream is not able to carry time-critical data, but allows the transmission of best effort traffic. A network device just holds one asynchronous stream.

Each application is dedicated either to a synchronous or to the asynchronous stream of the device. These streams are mapped to corresponding queues, so that each stream has its specific queue. The parameters of the streams are negotiated between the application or the middleware and the token-passing protocol.

Each time a token arrives, the stream first sends the synchronous frames as long as the current holding time is smaller or equal to the synchronous bandwidth H_i^k. After that, asynchronous frames will be sent as long as there is a remaining time $T_R > 0$. When the remaining time is expired or if there are no more asynchronous frames, the token will be passed to the next stream. Because the frames will not be segmented, it is possible to exceed the designated times. The arriving token can be an early token if the real token rotation time $T_{RT_i}^k$ is less than the target token rotation time *TTRT* ($T_{RT_i}^k < TTRT$), a late token if $T_{RT_i}^k > TTRT$, or an accurate token if $T_{RT_i}^k = TTRT$.

To compute the feasibility of the specified parameters, a protocol constraint can be defined similar to [6]: the data rates of all messages of the synchronous streams, including the overhead caused by the token-passing protocol and the processing delay of the network device Δ_{pr} must not exceed the total data rate of the network:

$$\sum_{k=1}^{n}\sum_{i=1}^{n_p^k}\frac{\left(C_{p_i}^k+\left(\frac{len(tok)+len(ack)}{r}+\Delta_{pr}\right)\right)}{T_{p_i}^k}\le 1 \qquad (1)$$

r describes the gross data rate of the network, $len(tok)$ and $len(ack)$ describe the length of the token and the acknowledgement frame.

To deal with the hard real-time requirements, the target token rotation time must be smaller or equal to the deadline of the stream. Thus, the following deadline constraint can be defined:

$$TTRT \le \min\left\{D_{p_i}^k\right\} \quad \forall_{i,k} \qquad (2)$$

and

$$A_i^k(l+1)-A_i^k(l)\le D_{p_i}^k \quad \forall_{i,k,l} \qquad (3)$$

with $A_i^k(l)$ describing the time at which a token visits the stream the *l*th time.

To guarantee hard real-time demands, it is necessary to consider the worst case behavior of the system. The worst case can occur when all the devices send for the maximum time $H_i^k + C_{p_i}^k$ so that the worst case cycle time can be formulated as

$$T_{Cycle_i}^k = TTRT + C_a^k + \sum_{k=1}^{n}\sum_{i=1}^{n_p^k} H_i^k + C_{p_i}^k \qquad (4)$$

To comply with the deadline constraint (2) and (3), the necessary condition can be formulated as

$$D_{p_i}^k \ge A_i^k(l+1)-A_i^k(l)=T_{Cycle_i}^k \quad \forall_{i,k,l} \qquad (5)$$

With (4) and (5) it is possible to formulate an equation to calculate the target token rotation time:

$$t \le TTRT \le D_{p_i}^k - C_a^k - \sum_{k=1}^{n}\sum_{i=1}^{n_p^k} H_i^k + C_{p_i}^k \qquad (6)$$

If $TTRT$ satisfies (6), the formulated hard real-time conditions will always be met!

3. Performance Results

With the mechanisms presented in section 2.3 it is possible to describe the worst case behavior of the protocol analytically to guarantee the real-time constraints. To analyze the exact (not just worst case) behavior of the system, exemplary simulations have been carried out. In the following simulations, a total of 7 devices are connected together, sending 7 synchronous and 5 asynchronous streams with very different requirements to data rate and delay. The data sources generate a continuous data rate (CBR) with isochronous frame arrivals and are switched on and off in predefined time intervals to generate different scenarios. Figure 2 shows the total utilization (without overhead) of the token-passing mechanism. The total utilization of this token-passing Ethernet network is about 80%, and the synchronous streams have a share of approximately 54 Mbit/s of the total utilization. The rest of the traffic is generated by asynchronous (best effort) traffic. A breakdown of the utilization cannot be viewed at any time, as would be expected with standard Ethernet.

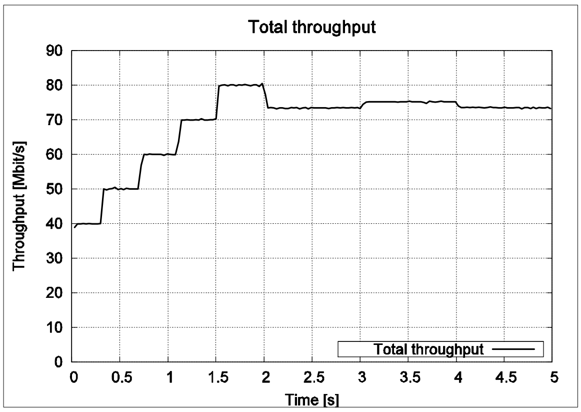

Figure 2: Total throughput of the network (token-passing)

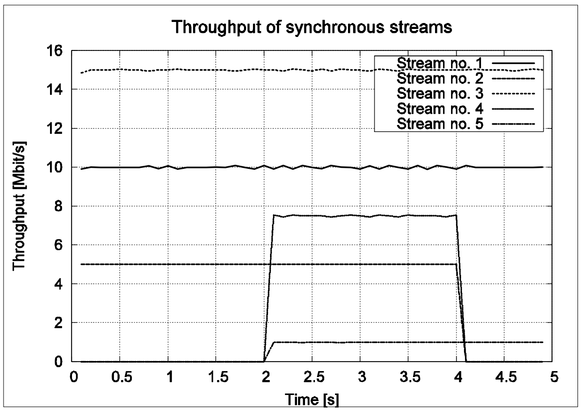

Figure 3: Throughput of synchronous streams (token-passing)

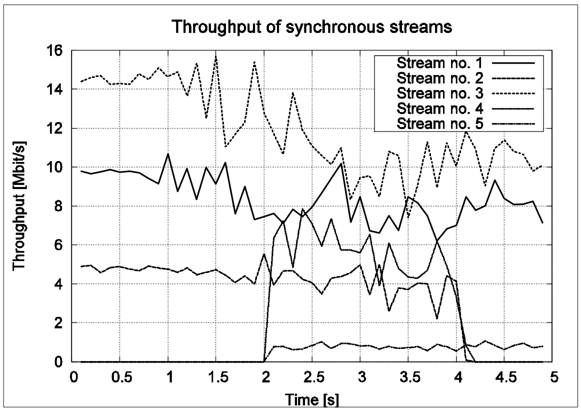

Figure 4: Throughput of synchronous streams (CSMA/CD)

Figure 3 shows the behavior of five of the seven synchronous streams. The desired data rates can be transmitted every time without any disturbance, whereas Figure 4 shows the characteristic of the synchronous streams within standard CSMA/CD Ethernet. Because of the high network load many collisions will occur, so that the throughput will degrade and the Quality of Service and real-time agreements cannot be met. The degradation of standard Ethernet can be viewed clearly at a load of approximately 50% - 55%. A prediction of the exact behavior of standard Ethernet is not possible because of the uncontrolled medium access.

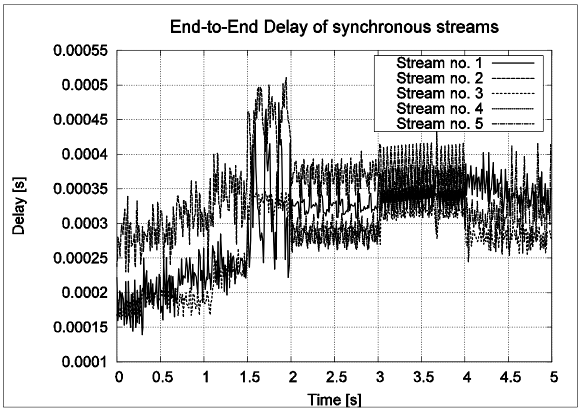

Figure 5: End-to-end delay of synchronous streams (token-passing)

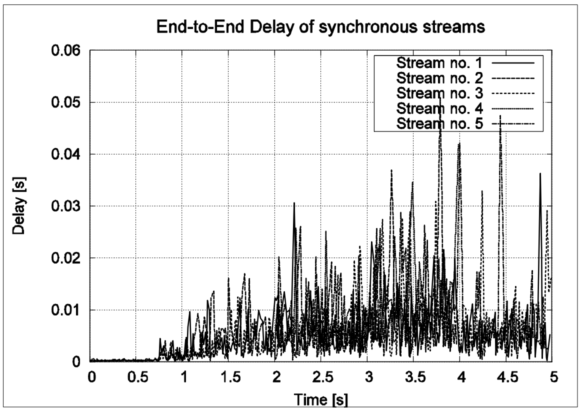

Figure 6: End-to-end delay of synchronous streams (CSMA/CD)

Figure 5 shows the end-to-end delay of the synchronous streams of the token-passing medium access control mechanism. The delay of the synchronous streams is kept quite low at about 0.35ms with low jitter. Figure 6 shows the delay of standard Ethernet that has a strong varying delay of up to 51ms. The delay increases drastically when the total load is above 50% - 55%. Standard Ethernet is not able to guarantee hard real-time constraints. Even in light load situations only a statistical real-time behavior can be expected.

The performance of the network mainly depends on the gross data rate of the medium, the amount of streams, and the deadline requirements of these streams. The lower the deadline of the streams the shorter the token rotation times. The deterministic of the new medium access is bought by an overhead caused by the token-passing procedure. The minimum delay that can be guaranteed depends on the number of synchronous streams within the network and has a lower limit at about 100µs. The maximum throughput of the network can be more than 90%, depending on the timing requirements of the streams.

4. Conclusion

A new token-passing mechanism as a supplement to shared media networks has been described. This mechanism allows a flexible and dynamic reservation of Quality of Service and hard real-time demands for each application. The parameters data rate and delay can be set very flexibly to meet the requirements of the different applications. The medium access is absolutely deterministic so that no collisions can occur. The enhanced network performs far better than standard CSMA/CD Ethernet at medium and high traffic load. The timing behavior of the system can be described analytically and worst case considerations can be carried out.

The token-passing mechanism described here is a general purpose medium access control mechanism that is not restricted to use within Ethernet but could also be applied to other shared media networks

References

[1] M. L. Molle, „A New Binary Logarithmic Arbitration Method for Ethernet“, Technical Report SCRI-298, Computer Systems Research Institute, University of Toronto, April 1994

[2] S. Kweon, K. G. Shin, G. Workman, „Achieving Real-Time Communication over Ethernet with Adaptive Traffic Smoothing“, RTAS 2000, Washington DC, June 2000

[3] R. Caponetto, L. Lo Bello, O. Mirabella, “Fuzzy Traffic Smoothing: Another Step towards Statistical Real-Time Communication over Ethernet Networks”, RTLIA 2002, Vienna, Austria, 2002

[4] C. Venkatramani, T. Chiueh, „Design, Implementation and Evaluation of a Software-based Real-Time Ethernet Protocol“, ACM SIGCOMM 95, 1995

[5] W. Hwang, H. Tode, K. Murakami, „Design of the HomeMAC: QoS Based MAC Protocol for the Home Network“, IEICE Trans. Commun. Vol. E85-B, No. 5, May 2002

[6] G. Agrawal, B. Chen, W. Zhao, S. Davari, „Guaranteeing synchronous Message Deadlines with the Timed Token Medium Access Control Protocol“, International Conference on Distributed Computing Systems, June 1992

[7] E. Tovar, F. Vasques, „Guaranteeing Real-Time Message Deadline in PROFIBUS Networks“, 10th Euromicro Workshop on Real-Time Systems, June 1998