PERFORMANCE OF SINGLE ACCESS CLASSES ON THE IEEE 802.4 TOKEN BUS

M. Alex Colvin, M.S.
Alfred C. Weaver, Ph.D.
University of Virginia

Computer Science Report No. TR-86-09
April 23, 1986

This paper has been submitted for publication in the *IEEE Transactions on Communications*.

PERFORMANCE OF SINGLE ACCESS CLASSES ON THE IEEE 802.4 TOKEN BUS

M. Alex Colvin

Alfred C. Weaver

Department of Computer Science

University of Virginia

ABSTRACT

The IEEE 802.4 token bus defines both *synchronous* and *asynchronous* message access classes. Their performance is compared for networks implementing a single message class. Throughput bounds are derived from the access class timer. It is shown that for some configurations the asynchronous class yields lower *observable* message delays. Mean observable delay is minimized by allowing each Medium Access Controller unrestricted service.

1. Medium Access Control for the IEEE 802.4 Token Bus

The IEEE 802.4 Token Bus Standard[1] defines a protocol for shared use of a communications bus. Medium Access Controllers (*MACs*) control access to the bus by transmitting a token cyclically, forming a logical ring.

1.1. Access Classes

The Standard defines eight message service classes to be used to enforce message service priorities. A *MAC* maps these onto as many as four message access classes. The access class is an attribute of the message, not of the *MAC*. The *MAC* has a separate message queue for each supported access class. When the token is received, service begins for the highest priority class and proceeds from there to the lowest priority class. The token is passed when all classes have been served.

Service is not preemptive. Once a *MAC* has begun service for a class, arriving higher priority messages must wait for the token to pass around the logical ring. The access classes take effect by restricting the time available for message transmissions when a station receives the token. A station loads a "token hold timer" when beginning service for a class. The timer is checked before each message transmission. If it has expired, service for the class is terminated. Because the timer is not checked during message transmission, it is possible to overrun this limit by one message time.

The highest priority class is called *synchronous*. The token hold timer for synchronous service is loaded with the constant *hi_pri_token_hold_time*.

There are three *asynchronous* classes. In descending priority they are *urgent_asynchronous*, *normal_asynchronous*, and *time_available*. There is a constant *target_rotation_time* for each asynchronous class. For asynchronous service the token hold timer is loaded with the residue from another timer, the *token_rotation_timer* for that class. The *token_rotation_timer* is then reset to the class *target_rotation_time*. The *token_rotation_timer* times one complete token rotation before its residue is copied into the token hold timer.

Synchronous servers are guaranteed service each time the token is received, but there is a fixed limit on the amount of such service. If the traffic is light at some stations, heavily loaded stations may not increase their service. Asynchronous servers have no guarantee of service when the token is received, However, the amount of service can adapt to the network load. Time not used by lightly loaded stations becomes available to those more heavily loaded.

The synchronous class timer is applied to each *MAC* individually. The asynchronous class timer is applied to the network as a whole, as a single large distributed queue. This makes the asynchronous timer less sensitive to variations in load at an individual *MAC*.

1.2. Delay

One of the principal metrics of network performance is delay. The delay encountered by a message is the time elapsed from message creation at the source until its reception at the destination. *Queueing delay* is the delay as the message waits to reach the head of the access class queue. *Access delay* is the wait for token arrival. *Transmission delay* is the time spent transmitting the message, including any overhead and propagation delay.

Use of token hold timers does not place a bound on total delay. On the contrary, mean delay is increased, as some messages may be required to wait for several token rotations before transmission.

2. Bounds

The following discussion applies to an error-free network of N identical MACs, each with a mean utilization of u, for a total utilization of $U = Nu$. The token transmission time is X_{token}, including address, framing, and propagation delays. Constant length messages arrive at each MAC from identical Poisson processes. Each requires time X for transmission, including address, framing, and propagation delays.

All messages in the network belong to a single message class. Each MAC, therefore, serves a single queue.

2.1. Token Rotation

The token rotation time is the time from reception of the token, at some arbitrarily selected server, to its subsequent reception at the same server, after passing around the logical ring.

At any time, either a message or a token is being transmitted. Let U be the network utilization, the fraction of the bandwidth consumed by messages; then $1-U$ is the fraction consumed by tokens. Let $\overline{C}$ be the mean token rotation time. The mean time transmitting messages during one token rotation is $U\overline{C}$. The time passing the token during a token rotation is NX_{token}. Thus,

$$\overline{C} = U\overline{C} + NX_{\text{token}} = \frac{NX_{\text{token}}}{1-U} \qquad (1)$$

This mean rotation time requires that the network be stable ($U < 1$) but does not depend on the token hold timers.

2.2. Access Classes

Even in a network with a single message class, the access class mechanism may be used to bound the token rotation time and, with it, the bus access delay.

For a network with synchronous traffic, the *hi_pri_token_hold_time* (T_{sync}) bounds the token rotation time C by

$$0 \leqslant C < C_{\max} = N(T_{\text{sync}}+X_{\text{token}}) \tag{2}$$

This restricts the maximum network utilization U.

$$0 \leqslant U < U_{\max} = \frac{T_{\text{sync}}}{T_{\text{sync}}+X_{\text{token}}} \tag{3}$$

For an asynchronous network the *target_rotation_time* (T_{async}) limits the duration of one token rotation $\overline{C}$ followed by one service period $u\overline{C}$. The utilization and cycle time are bounded by $\overline{C}+u\overline{C} < T_{\text{async}}$. This limits the network utilization U to be

$$0 \leqslant U < U_{\max} = N\left|\frac{T_{\text{async}}}{\overline{C}} - 1\right| = \frac{T_{\text{async}}-NX_{\text{token}}}{T_{\text{async}}+X_{\text{token}}} \tag{4}$$

and

$$T_{\text{async}} = X_{\text{token}}\frac{N+U_{\max}}{1-U_{\max}} \tag{5}$$

2.3. Delay

A recent paper by Fuhrmann and Cooper[2] expresses the mean delay as the sum of half the cycle time, the mean queueing delay, and the transmission delay. This applies when the network is large ($N\rightarrow\infty$) and the token holding time is unlimited. The sum of the three terms is

$$\overline{D} = \frac{\overline{C}}{2} + \frac{XU}{2\,(1-U)} + X = \frac{NX_{\text{token}}+XU}{2(1-U)} + X \tag{6}$$

The mean token rotation time is unaffected by the token hold timers. Mean delay is increased by reducing the timers. When finite token hold times are used, messages may have to wait several token rotations to gain access to the bus, increasing the delay.

3. Simulations

The figures show the results of computer simulations. In the simulated configuration messages arrive at 64 *MAC*s from identical Poisson processes. Each message transmission requires a constant $X = 306$ bit times. This time includes data, address, framing, and propagation delays. Token passing requires $X_{\text{token}} =$ 146 bit times ($.477\,X$). This corresponds to a 96 bit token with 50 bit times of overhead.

3.1. Mean Delay

Figure 1 shows the mean delay ($\overline{D}$) for messages. This includes access delay, queuing delay, and transmission time. Delay is normalized to message transmission time (X), and plotted on a logarithmic scale.

The simulated load was varied from 0 to $U_{\max}$. In these simulations there was no limit on token hold times, hence $U_{\max} = 1$.

The curve shows the delay described in (6) above. Circles indicate mean delay measured from simulations. The data points show good agreement with the behavior predicted by equation (6).

3.2. Access Classes

To examine the difference between synchronous and asynchronous access classes, simulations were done with both types of traffic using three different values of $U_{\max}$. For the synchronous-only networks the *hi_pri_token_hold—time*, $T_{\text{sync}} = 1$, 2, and 4 packet times (X).

The asynchronous *target_rotation_time* (T_{async}) is not directly comparable to the synchronous timer T_{sync}. For comparison, the asynchronous networks were simulated with T_{async} chosen to yield the same $U_{\max}$ as the synchronous networks, according to equation (5) above.

T_{sync}	T_{async}	$U_{\max}$
$1X$	$95.5X$	0.677
$2X$	$160.5X$	0.807
$4X$	$290.5X$	0.893

Figures 2 and 3 show the results of simulations of synchronous-only and asynchronous-only traffic for the three $U_{\max}$. Figures 2a, 2b, and 2c show the mean and Figures 3a, 3b, and 3c show the standard deviation of message delay.

Each configuration is simulated with loads ranging from 0 to the corresponding $U_{\max}$. The data points representing simulations are marked with a symbol. Vertical lines show the different $U_{\max}$.

The least delay (as well as the greatest maximum utilization) is shown by the network with unlimited token hold times in Figure 1.

The mean delay is everywhere greater for the synchronous-only networks than for the asynchronous-only networks. Further, the synchronous-only networks show greater variation in delay. The asynchronous network delay is unaffected by the token hold timers until the load approaches $U_{\max}$.

The mean token rotation time $\overline{C}$ has the same curve for all configurations, namely that described in equation (1) above.

4. Discussion

The standard states that if only one access class is to be implemented it must be synchronous. For some applications the lower mean delay and smaller delay variation of a single asynchronous class may be preferable.

The token hold timers are particularly well suited to a polling network, where a station generates messages only when a token is received. This, however, violates the separation of Logical Link and Medium Access layers described in the Standard.

Restricting token holding times may improve bounds on access delay. But when the message arrivals are independent of the network servers (as they would be when the *MAC*s are self-contained units), the *observable* measure is the total delay. The minimum mean delay is achieved by avoiding the priority class mechanism altogether.

REFERENCES

1. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard 802.4 — Token-Passing Bus Access Method and Physical Layer Specification*, IEEE (1985).

2. S. W. Fuhrmann and R. B. Cooper, "Application of Decomposition Principle in M/G/1 Vacation Model to Two Continuum Cyclic Queueing Models (Especially Token-Ring LANs)," *AT&T Technical Journal*, **64**(5), pp. 1091-1099 (May-June 1985).

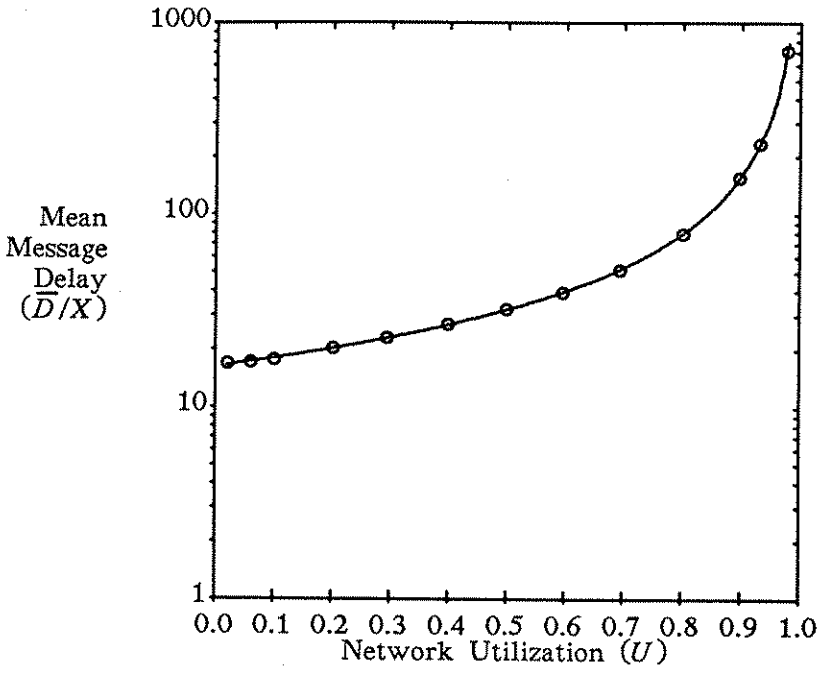

Figure 1. Mean Delay vs. Utilization

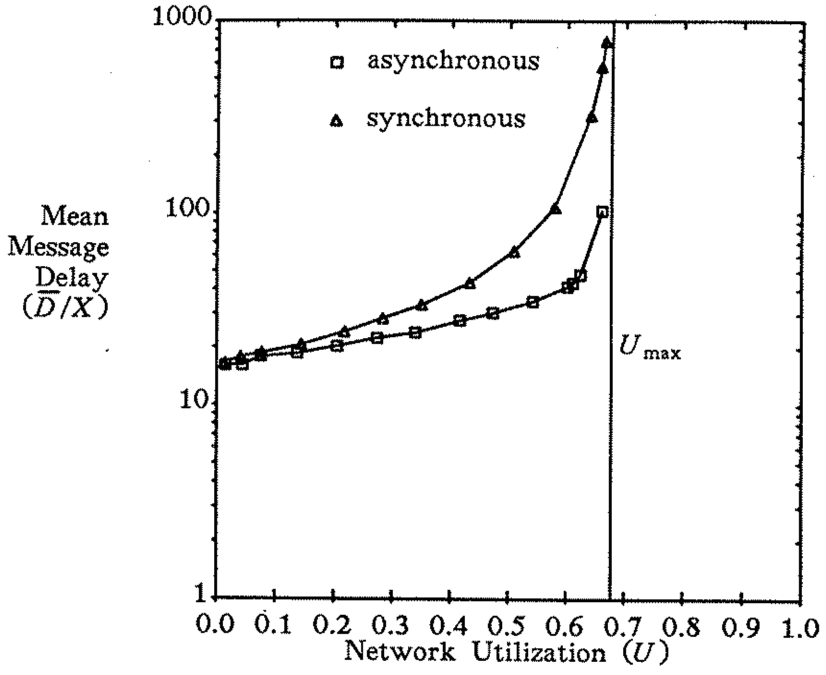

Figure 2a. Mean Delay vs. Utilization with U_{max} = .677

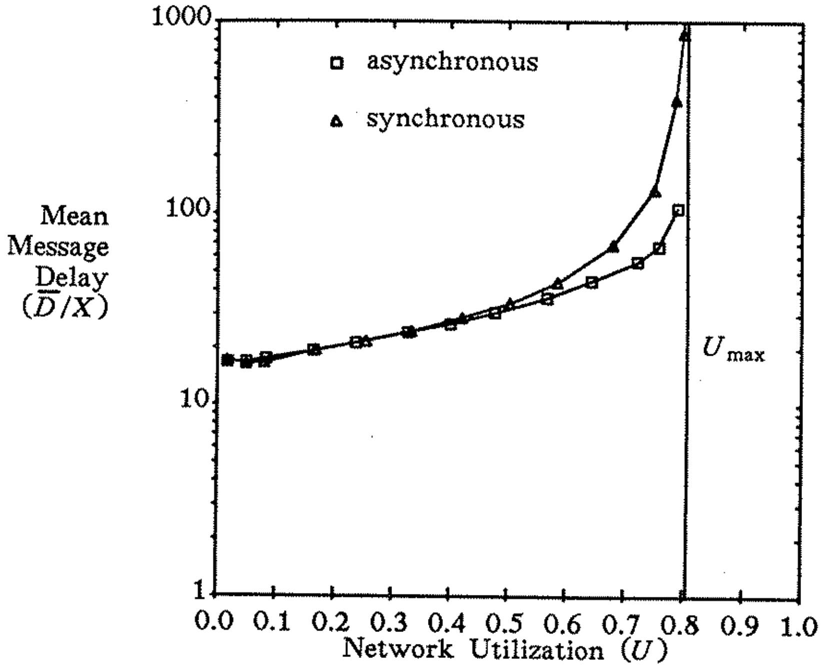

Figure 2b. Mean Delay vs. Utilization with $U_{\max}$ = .807

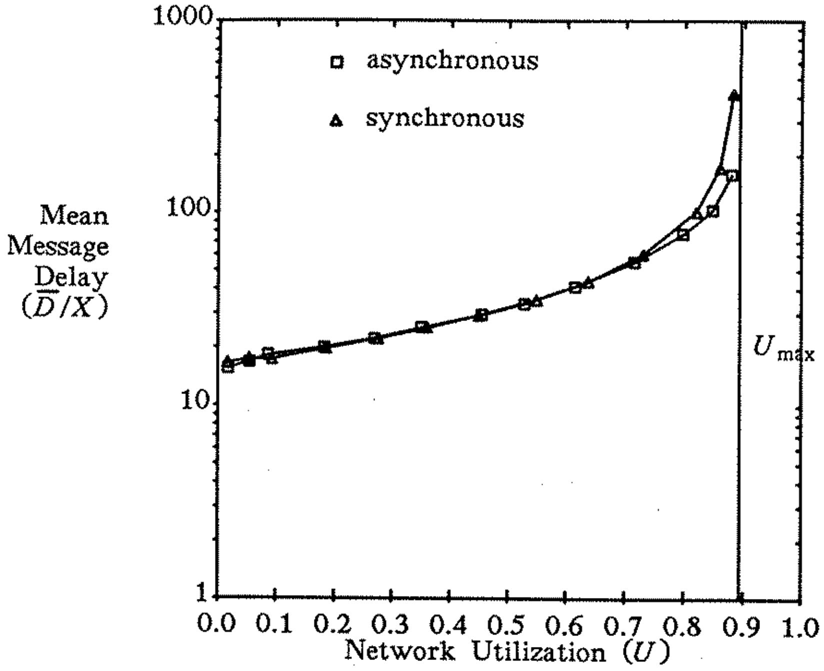

Figure 2c. Mean Delay vs. Utilization with $U_{\max}$ = .892

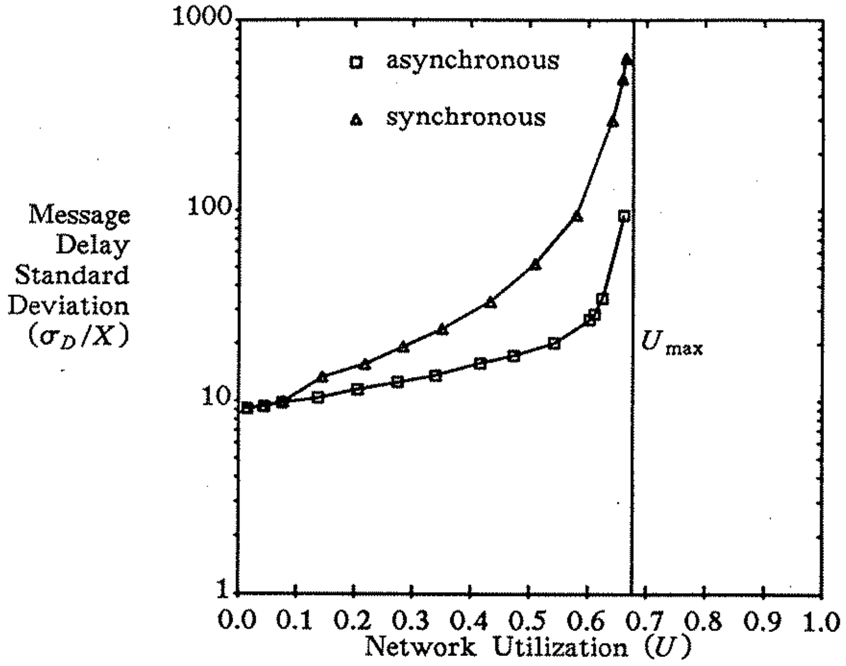

Figure 3a. Delay Standard Deviation vs. Utilization with $U_{\max}$ = .677

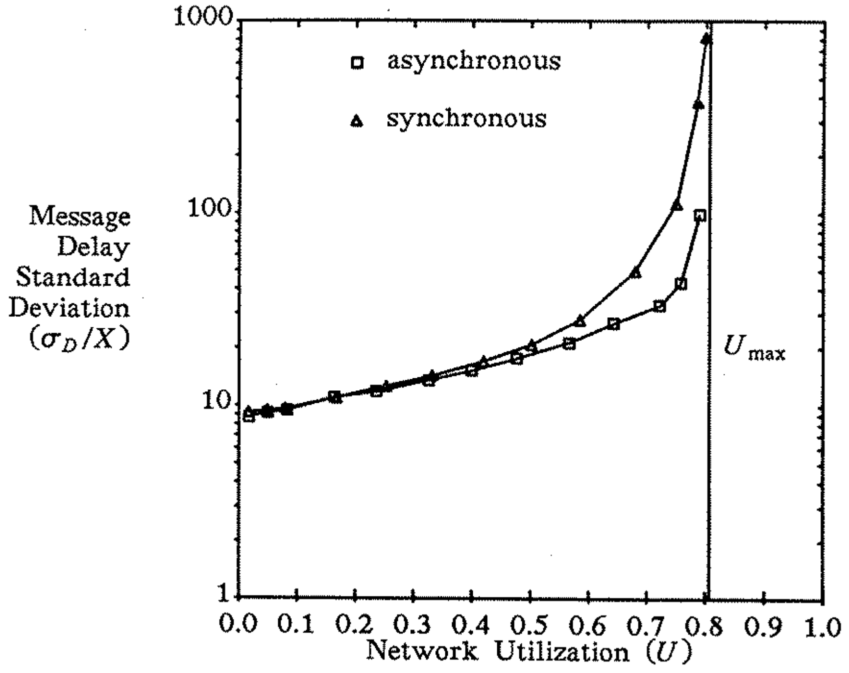

Figure 3b. Delay Standard Deviation vs. Utilization with $U_{\max}$ = .807

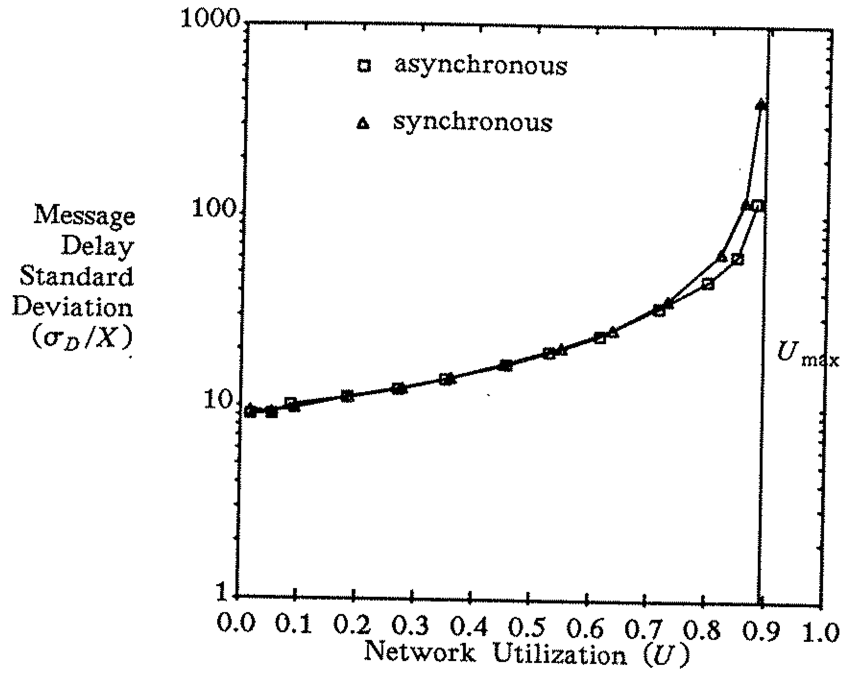

Figure 3c. Delay Standard Deviation vs. Utilization with U_{max} = .893