Performance Analysis of the *BusNet* Protocol for Backplane Bus-Based Interprocessor Communication

Minyoung Sung, Naehyuck Chang, Jinsung Cho and Heonshik Shin
School of Computer Science & Engineering
Seoul National University
Seoul 151-742, Korea

Abstract

Nowadays, backplane bus-based multiprocessor systems often utilize the standard network protocol such as TCP/IP for communication between processors on the backplane bus. In such systems, it is common for the backplane bus to emulate the standard MAC protocols such as CSMA/CD. This paper aims to analyze the performance of the MAC emulation-based backplane network by constructing queueing models based on detailed bus operations. For this purpose, we choose BusNet as a target protocol. BusNet is an ANSI standard network protocol and its specification contains basic bus operations commonly used in most backplane buses. We investigate the throughput-delay characteristics of BusNet. We also compare the packet delay in BusNet with the IEEE 802.3 CSMA/CD network which BusNet is expected to be compatible with. The simulation result shows how an optimal block transfer scale can be determined in respect of the performance trade-off between BusNet and other real-time traffics.

Keywords: Backplane bus network protocol, Packet transfer delay, BusNet

1 Introduction

Processors in a shared-bus multiprocessor system typically communicate with each other by accessing directly the distributed and/or unified shared memory through the backplane bus protocol. Recently, with rapid advances in computing and communication technologies, the application software is commonly involved in complicated interprocessor communication. As a result, the modern trend is to use standard network protocols such as TCP/IP on the backplane bus. This gives advantages such as portability, robustness and rapid prototyping at the expense of slight performance degradation. These *backplane network protocols* commonly emulate the standard MAC (medium access control) protocols such as CSMA/CD in order to interface the existing upper protocol stack, i.e., IP. This approach gives an additional advantage in costs because application software can utilize standard network protocols without equipping every processor board with a network interface card. Despite the growth of usage and the benefit of backplane network protocols, however, there has been little work regarding the performance of the backplane bus as a communication medium. The performance of a shared-bus multiprocessor system depends not only on the processor speed, but also on the performance of the underlying backplane bus [1, 2]. In particular, the performance of the bus may vary significantly according to the bus arbitration policy, packet size, bus transfer parameters, etc. Therefore, a robust communication model of the backplane bus is essential to evaluate the design and performance of the multiprocessor systems. In this paper, we develop and validate an analytic model to study the MAC and link layer characteristics of the backplane bus.

Numerous works have been carried out concerning the performance of the MAC protocols, but have mostly focused on conventional local area networks. In particular, CSMA/CD has been extensively analyzed for the past two decades [3, 4]. Compared with the local-area networks, however, the backplane bus exhibits quite a different behavior in delay performance due to the physical channel characteristics. Physical features of the backplane buses have been investigated in several aspects. In [5], a simulation study is performed on a static priority arbiter, a rotating daisy chain and an ideal FCFS arbiter. Assuming fixed bus access time, it measures the mean and variation of the waiting time for the data transfer request. In [6], the bus bandwidth allocated to each processor is analyzed using a timed Petri-net model. Recently, several works have been presented which investigate the commercial backplane buses in detail. Examples of such works include MCA bus [7] and CAN bus [8]. However, most of these researches do not give a suitable model for the packet-based communication over the backplane bus. They lack in consideration for the packet-oriented data transfer or concentrating on estimating the bus bandwidth allocated to each processor.

In this paper we aim to analyze the packet transfer delay in the backplane bus network. We devise queueing systems where a packet request is served by a group of basic bus transfers called *transactions*. Unlike traditional packet service models where the entire packet transmission is non-preemptive by nature, packet transmission on the backplane bus can be preempted by other ready nodes since the bus is shared on the transaction basis. The accuracy of the analytic model is validated by a simulation study using a bus simulator which is constructed to reflect bus details. Using the analytic model, we discuss the throughput-delay performance of the backplane network in comparison with that of the conventional CSMA/CD network. We also present experimental results obtained for various values of block transfer scale. For analysis, we choose *BusNet* [9] as a target protocol. BusNet is an ANSI standard protocol for standardized communication over *VMEbus* [10].

The rest of this paper is organized as follows. In Section 2, we introduce the concept of the backplane bus network protocol and briefly describe the arbitration schemes in VMEbus. In Section 3, we analyze the packet transfer delay in BusNet. Section 4 presents numerical results of the analytic model. This paper ends in Section 5.

2 The Backplane Bus Network Protocol

2.1 BusNet: A standard bus network protocol

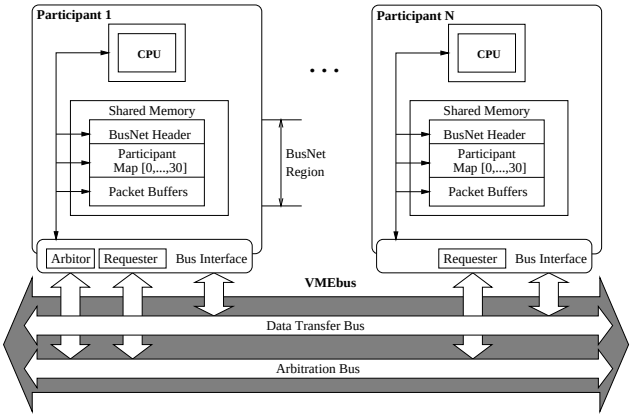


Figure 1: BusNet backplane network.

BusNet is an ANSI standard protocol originally developed by Force Computers Inc. BusNet emulates the standard MAC protocol, specifically the Ethernet, between the VMEbus and the ISO/OSI network layer. BusNet specification does not rely on special features such as read-modify-write. Fig. 1 shows a system using BusNet. Each processor or *participant* is identified by a logical address ranging from 0 to 30. It shares a segment of its local memory, called *BusNet region*, for access from other participants. Each BusNet region consists of a BusNet header, participant maps and packet buffers. The participant map contains a protocol descriptor (PD) to facilitate the exchange of packet between two peers. The PD is composed of `receive_status`, `transmit_status`, `buffer_offset`, `buffer_size` and `sequence_number`. The `receive_status` is used by the receiver to control access to the packet buffer. When the `receive_status` is in the `RDY` state, the transmitter may write a packet to the buffer in the receiver. The `transmit_status` indicates to the receiver that the packet buffer has been filled with a new packet. The `buffer_offset` and `buffer_size` contain the address and size of the packet buffer, respectively. The `sequence_number` is included for error checking. Table 1 illustrates the packet transmission steps. The `map_i` describes the participant map located in participant `i`.

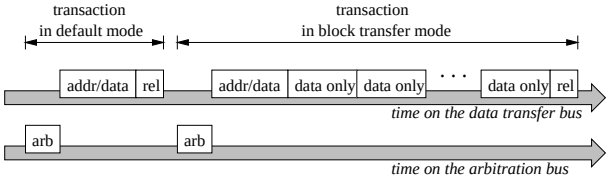
Table 1: Steps for a packet transmission in BusNet.

Step	Description
1	Transmitter (T) checks to see if Receiver (R) is ready. `map_T.PD[R].receive_status == RDY`
2	T determines the buffer offset and size parameters. (`map_T.PD[R].buffer_offset` and `map_T.PD[R].buffer_size`).
3	T sets `receive_status` to `IDLE`. `map_T.PD[R].receive_statue = IDLE`
4	T determines the VME address of R's packet buffer.
5	T transfers a packet to R's packet buffer.
6	T updates the `sequence_number` in R's PD. `map_R.PD[T].sequence_number += 1`
7	T sets `transmit_status` to `RDY`. `map_R.PD[T].transmit_status = RDY`
8	T generates a mailbox interrupt to R (optional).
9	R checks the number in `map_R.PD[T].sequence_number`.
10	R sets `transmit_status` to `IDLE`. `map_R.PD[T].transmit_status = IDLE`
11	R stores the address of newly allocated buffer in `map_T.PD[R].buffer_offset`.
12	R sets `receive_status` to `RDY`. `map_T.PD[R].receive_status = RDY`
13	R generates a mailbox interrupt to T (optional).

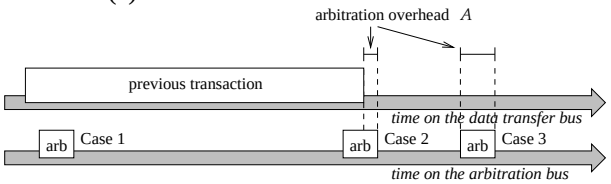
2.2 Arbitration and data transfer in VMEbus

VMEbus *interface module* in Fig. 1 controls access to the *data transfer bus* (hereafter bus) by the *arbiter* and the *requester* [10]. The *arbitration bus* consists of *four bus request lines* and *four bus grant lines*. While the request lines with the same priority are hooked up together in the form of a wired-OR, the grant lines are daisy-chained across

the slots. Several arbitration schemes can be produced by the combination of ⟨arbiter-type, requester-type⟩. VMEbus standard describes three types of arbiters: *prioritized*, *single-level* and *round-robin*. The prioritized arbiter issues a bus grant corresponding to the highest priority among the bus requests detected. The single-level arbiter responds only to the requests on the highest-priority line. The round-robin arbiter responds to the four request lines in a round-robin order. In addition to the arbitration by the arbiter, the bus grant daisy-chain performs a secondary level of arbitration. Requesters can be configured in one of two types: *demand* and *fair*. In the demand mode, requesters sharing a common request line are prioritized according to their slot positions. The requester closer to the arbiter has the higher priority. For a fair access, the fair requester is employed. The fair requester, once granted the bus, does not issue another request if there are other requests pending on its own priority level.



(a) The default and block transfer modes



(b) Three possible placements of arbitration

Figure 2: The bus transaction.

For generality, our analysis considers two arbitration schemes, i.e., PRI and FAIR. In the PRI mode, each participant has a unique priority. The bus ownership is always granted to the highest-priority participant among those requesting the bus. This mode includes the ⟨prioritized, demand⟩ and ⟨single-level, demand⟩ pairs. In the FAIR mode, a fair share of the bus is guaranteed. The ⟨single-level, fair⟩ pair corresponds to this mode. The ⟨round-robin, fair⟩ pair can be viewed as a FAIR mode when all the request lines have the same number of participants. The rest of the combinations are not considered in this paper.

We consider two transaction modes: the default (DFT) and block (BLT) modes. As shown in Fig. 2 (a), every data cycle in the DFT mode requires a unique arbitration. The BLT mode differs from the DFT mode in that once a master obtains the bus, it requires only one address cycle followed by a number of data cycles.

3 Packet Transfer Delay in BusNet

We define *packet transfer delay* as the time duration from packet arrival to transfer completion. We assume that each participant is associated with a queue of infinite capacity. And at each participant Φ_i, packet arrives in a Poisson distribution with rate λ_i, hence resulting in an aggregate arrival rate λ.

3.1 Transmission time on the VMEbus

Transmission time is defined as the time taken on the bus for a packet transfer when there is no contention for the bus. For a stable system, it requires that

$$U = \lambda\nu < 1. \tag{1}$$

Appendix A illustrates the notations used in this paper. Assuming transmission-efficient BLT mode, a p-byte packet is transmitted by a group of n BLT transactions. While each transaction is carried out in a non-preemptive way, packet transmission itself can be interrupted by other participants. The mean transmission time ν is given by

$$\nu = n\sigma, \tag{2}$$

where the mean transaction time σ is given by

$$\sigma = E[\mathcal{A}] + \tau_a + (m-1)\tau_d + \tau_r. \tag{3}$$

Typical values for the transaction parameters are given in Appendix A. They are based on the Tundra SCV64 VMEbus interface [11]. Note that random variable $\mathcal{A}$ denotes the additional arbitration overhead as shown in Fig. 2 (b). We classify the possible arbitration placement depending on the instant at which the transaction request arrives. Case 1 is when the arbitration completes before the end of the previous transaction, hence resulting in zero overhead. In Case 2, the arbitration is partially overlapped with the previous transaction. The average overhead is $\tau_b/2$. Finally, the overhead amounts to τ_b in Case 3. Table 2 summarizes the probabilities associated with $\mathcal{A}$. So, we have

$$E[\mathcal{A}] = \tau_b\left(1 - U\left(1 - \frac{\tau_b}{2\sigma}\right)\right). \tag{4}$$

We derive $E[\mathcal{A}]$ from Eqs. (1) – (4) :

$$E[\mathcal{A}] = \frac{\tau_b}{1+\lambda n\tau_b}\left(1 - \lambda n\left(\tau_a + m\tau_d + \tau_r - \frac{\tau_b}{2}\right)\right). \tag{5}$$

3.2 Packet transfer delay

Let the mean packet service time S_i denote the mean time between the instants that a packet request reaches the

Table 2: The probabilities for the arbitration placement.

Case	1	2	3
P[Case]	$U\frac{\sigma-\tau_b}{\sigma}$	$U\frac{\tau_b}{\sigma}$	$1-U$

front of the queue and that the transmission is completed. Let Q_i denote the queueing time. The packet transfer delay in the FAIR mode is given by $W_i = S_i + Q_i$. For analysis, we introduce a virtual token which is passed among participants in a round-robin order with zero token passing time. A ready participant has the control of the bus while it holds the token. The token holding time corresponds to σ. We model this system as N independent M/G/1 queues and consider one of such queues. Let P_{Φ_i} denote the probability that Φ_i is busy with the bus which is given by $P_{\Phi_i} = \lambda_i S_i$. With the probability P_{Φ_i} and the second moment of the service time S_i^2 given, W_i is derived by the Pollaczek-Khintchine formula [12].

Proposition 1 *In the FAIR mode, the mean packet transfer delay is given by*

$$W_i = S_i + Q_i = \frac{P_{\Phi_i}}{\lambda_i} + \frac{S_i^2\lambda_i}{2(1-P_{\Phi_i})}, \quad (6)$$

$$P_{\Phi_i} = \frac{U_i}{1+U_i}\Big/\Big(1-\sum_{j=1}^{N}\frac{U_j}{1+U_j}\Big),$$

$$S_i^2 = \nu^2\Big(1+2\sum_{j=1,j\neq i}^{N}P_{\Phi_j}+\sum_{x=0}^{N-1}x^2\sum_{\forall\Delta_x}\prod_{k\in\Delta_x}P_{\Phi_k}\prod_{l\in\bar{\Delta}_x}(1-P_{\Phi_l})\Big),$$

where $\Delta_x \subset \{1,\ldots,i-1,i+1,\ldots,N\}$, $|\Delta_x| = x$, *and* $\bar{\Delta}_x = \{1,\ldots,i-1,i+1,\ldots,N\} - \Delta_x$.

Proof. Proof is omitted due to space limitation. Refer to the full version of this paper [13]. □

In the PRI mode, Φ_i can start transaction only when all the higher-priority queues are empty. We assume that the lower participant number is associated with the higher priority. Let W_i' denote the packet transfer time excluding the time taken for the actual data transfer. Between any two consecutive transactions in a packet, other packet transmissions can interrupt with higher priorities. Delays incurred by such interference must also be incorporated into W_i'. The packet delay is given by $W_i = W_i' + \nu$. We derive W_i by extending Cobham's formula [12] for the M/G/1 priority queueing system.

Proposition 2 *In the PRI mode, the mean packet transfer delay is given by*

$$W_i = \frac{\frac{\sigma}{2}\sum_{j=1}^{N}U_j + \sum_{j=1}^{i-1}U_jY_j}{\Big(1-\sum_{j=1}^{i-1}U_j\Big)\Big(1-\sum_{j=1}^{i}U_j\Big)} + \frac{Y_i}{1-\sum_{j=1}^{i}U_j} + \nu, \quad (7)$$

where $Y_i = \frac{3}{2}(\nu-\sigma)\sum_{j=1}^{i}U_j - (\nu-\sigma)U_i$.

Proof. Proof is omitted due to space limitation. Refer to the full version of this paper [13]. □

4 Performance Evaluation

In this section, we evaluate the delay performance of BusNet. We introduce *bus throughput* as the fraction of bus time utilized by data phases:

$$\rho = \frac{\tau_a + m\tau_d}{\sigma}U = n(\tau_a + m\tau_d)\lambda.$$

Eqs. (1) – (3) and (5) produce the upper bound of the throughput, $\rho_{max}(I)$ as

$$\frac{\tau_a + m\tau_d}{(1-I)\tau_b + \frac{1}{2}\left(\tau_a + m\tau_d + \tau_r + \sqrt{(\tau_a + m\tau_d + \tau_r)^2 + 2I\tau_b^2}\right)},$$

where I is the indicator function which evaluates to 1 for overlapped arbitration and to 0 otherwise.

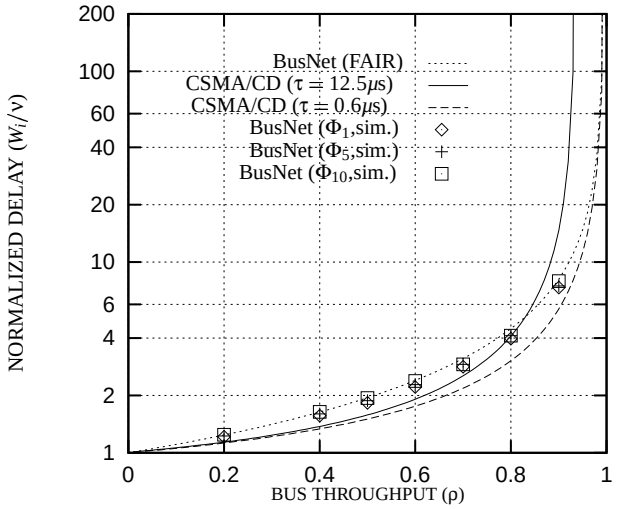
Fig. 3 shows the packet delay versus bus throughput for the BusNet and CSMA/CD networks when $m = 64$ and $N = 10$. The packet transfer delay is normalized to be W_i/ν. We assume identical arrival rates (i.e., $\lambda_1 = \ldots = \lambda_N$). The packet size is 1518 bytes. This, in BusNet, corresponds to $p = 1582$ bytes due to the additional BusNet header.

As an analytic model of CSMA/CD, we adopt Lam's result [3]. With constant packet length, the mean packet transfer delay is given by

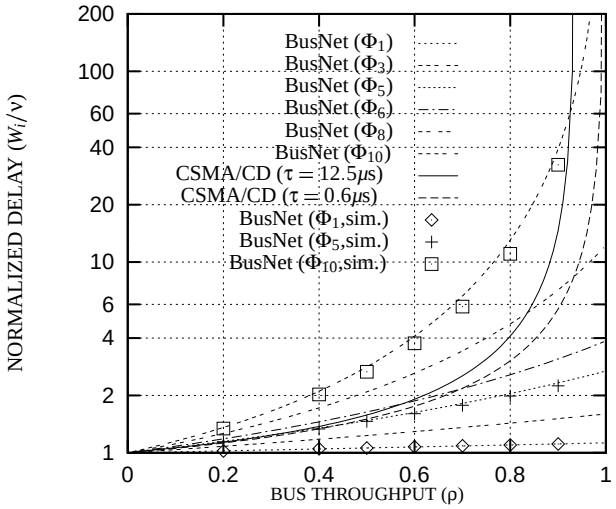
$$\forall i,\ W_i = \nu + \frac{\tau(4e+1)}{2} - \frac{(1-e^{-2\lambda\tau})(2/\lambda + 2\tau/e - 6\tau)}{2\left(e^{-\lambda\nu}e^{-(1+\lambda\tau)} + e^{-2\lambda\tau} - 1\right)} + \frac{\lambda\left(\nu^2 + (4e+2)\tau\nu + 5\tau^2 + 4e(2e-1)\tau^2\right)}{2(1-\lambda(\nu+\tau+2e\tau))},$$

where τ refers to the end-to-end propagation delay. For a transmission rate of 10 Mbit/s, ν is 1158.1 μs. The delay curves are plotted for τ values of 12.5 μs and 0.6 μs which correspond to cable lengths of 2.5 km and 50 m, respectively. Lam also gives the throughput bound of the CSMA/CD bus as $\rho < \frac{e}{2\tau/\nu + (1+\tau/\nu)e}$.

Fig. 3 (a) illustrates the delay curve for the FAIR mode. Both of BusNet and CSMA/CD show tolerable delay performance except at high bus load where they exhibit sharp increases in transfer delay as ρ approaches their throughput bounds. As can be seen from Fig. 3 (b), BusNet in the PRI mode exhibits a significant variance in throughput-delay performance according to the priority of the participant: the higher the priority, as expected, the better the performance. For participants with relatively high priorities,



(a) Throughput-delay in the FAIR mode



(b) Throughput-delay in the PRI mode

Figure 3: Throughput-delay characteristics.

Φ_1 to Φ_3, the transfer delays do not increase significantly even when ρ is near one. In particular, the increased delay of Φ_1 is negligible and can be bounded well under 13%. On the contrary, participants Φ_8 to Φ_{10} expose poor delay performance as ρ approaches one.

The figures also suggest that the simulation and analysis agree very well. The simulation results have been obtained from a bus simulator constructed in SMPL language [14]. The simulator was designed to reflect the bus details such as arbiter and requester types and transfer parameters. The simulation for the FAIR mode was conducted under the ⟨single-level, fair⟩ scheme, where fairness is supported solely by the requesters as described in Section 2.2. Although the simulation result shows an increasing tendency in the variance of delay among participants as ρ increases, the variance is negligible for the practical range of ρ.

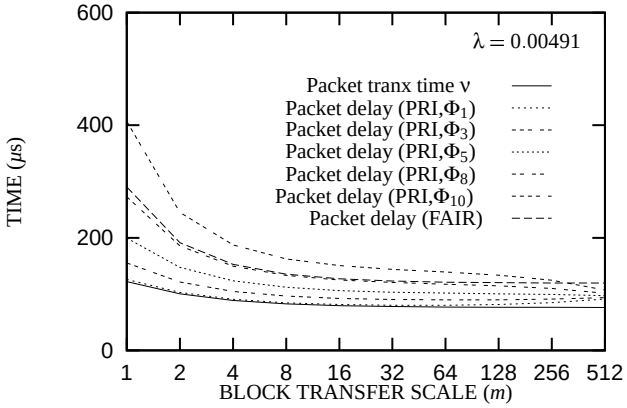


Figure 4: The effect of block transfer scale on packet delay.

Fig. 4 shows the effect of the block transfer on the packet delay for moderate bus load. We used $p = 2048$. Observe that as m becomes larger, both ν and W_i decrease rapidly until m reaches around 64. After m exceeds 64, little improvement is noticed. It is interesting to find that this value of m coincides with the maximum block transfer permitted in the Tundra SCV64. However, BusNet traffic in a large transfer scale is undesirable for other urgent data transfers such as real-time I/O. The performance of real-time communication is primarily measured by the 'schedulability', i.e., the guarantee that messages will be delivered within their deadlines [15]. A large scale of block transfer increases the bound on the time for real-time messages to access the bus, hence resulting in poor real-time performance. Therefore, an optimum value of m should be determined considering the performance trade-off between the BusNet and real-time traffics.

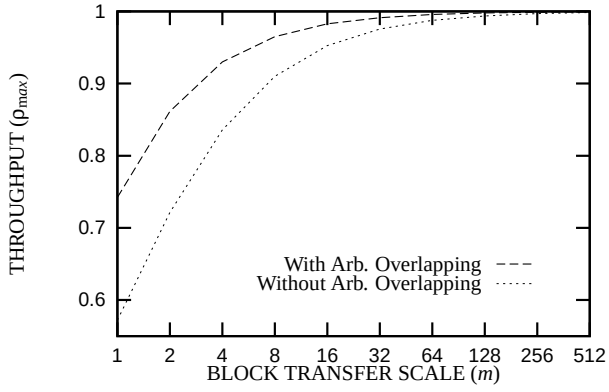


Figure 5: Attainable throughput bound.

Fig. 5 shows that a small m brings down ρ_{max} severely. In particular, the achievable bus throughput is at best 0.75 without block transfers. By utilizing the block transfer, we can obtain a ρ_{max} value close to one.

5 Conclusion

Backplane bus network protocols are widely used for interprocessor communication in a multiprocessor system. In this paper, we have studied the performance of the backplane network protocol as a communication medium. We analyzed the mean packet transfer delays under the PRI and FAIR arbitration schemes. For an accurate analysis, we developed a detailed bus transaction model which investigates the physical bus features including the block transfer and overlapped arbitration.

In the PRI mode, BusNet shows a significant variance of delay among participants. In particular, the participant with the highest priority is most favored so that the delay increase is kept well under 13% throughout the bus throughput. We also examined the effect of block transfer scale. The block transfer mechanism greatly improves the delay performance. With a large block transfer scale, we can obtain the bus throughput value near one. However, we observed that an optimal block transfer scale should be determined considering the undesirable effect of block transfer on real-time performance.

A Nomenclature

Symbol	Description
N	Number of participants.
Φ_i	Participant i ($1 \leq i \leq N$).
λ_i	Packet arrival rate at Φ_i.
λ	Aggregate packet arrival rate. $\lambda = \sum_{i=1}^{N} \lambda_i$.
U_i	Bus utilization by Φ_i. $U_i = \lambda_i \nu$.
U	Overall bus utilization. $U = \sum_{i=1}^{N} U_i$.
W_i	Mean packet transfer delay at Φ_i.
ν	Mean packet transmission time.
σ	Mean BLT transaction time.
n	Number of BLT transactions required to transmit a p-byte packet. $n = \lceil \frac{p}{wm} \rceil$.
p	Packet size in bytes.
w	Width of the bus in bytes (typically 4).
m	Block transfer scale, i.e., the number of data cycles in a block transfer (typically up to 64).
$\mathcal{A}$	Arbitration overhead. Random variable.
τ_b	Duration of the arbitration phase (typically 78 ns).
τ_a	Duration of the address phase (typically 159 ns).
τ_d	Duration of the data phase (typically 149 ns).
τ_r	Duration of the bus release phase (typically 41 ns).
ρ	Bus throughput.

References

[1] C. Chen and F. Lin, An Easy-to-Use Approach for Practical Bus-Based System Design, IEEE Trans. Computers, vol. 48, pp. 780–792, 1999.

[2] C. Lee and T. Parng, A Subsystem-Oriented Performance Analysis Methodology for Shared-Bus Multiprocessors, IEEE Trans. Parallel and Distributed Systems, vol. 7, pp. 755–767, 1996.

[3] S. S. Lam, A Carrier Sense Multiple Access Protocol for Local Networks, Computer Networks, vol. 4, pp. 21–32, 1980.

[4] Y. Matsumoto and Y. Kakahashi and T. Hasegawa, The Effects of Packet Size Distributions on Output and Delay Processes of CSMA/CD, IEEE Trans. Communications, vol. COM-38, pp. 199–214, 1990.

[5] W. L. Bain and S. R. Ahuja, Performance Analysis of High-Speed Digital Buses for Multiprocessing Systems, Proc. of the 8th Int'l. Symp. on Computer Architecture, pp. 107–133, 1981.

[6] M. K. Vernon and S. T. Leutenegger, Fairness Analysis of Distributed Arbitration Protocols, CSTR#744, Univ. of Wisconsin-Madison, 1988.

[7] K. A. Kettler and J. K. Strosnider, Scheduling Analysis of the Micro Channel Architecture for Multimedia Applications, Proc. of the IEEE Int'l Conf. on Multimedia and Computing Systems, pp. 403–414, 1994.

[8] K. Tindell, H. Hansson and A. Wellings, Analysing Real-Time Communications: Controller Area Network (CAN), Proc. of the IEEE Real-Time Systems Symposium, pp. 259–263, 1994.

[9] VITA 19.0-1997, Summary and introduction to the BusNet standard, 1997.

[10] VMEbus Specification: Draft Specification Revision D1.6, 1993.

[11] SCV64 User Manual, Tundra Co., 1998.

[12] P. G. Harrison and N. M. Patel, Performance Modeling of Communication Networks and Computer Architectures, Addison-Wesley, UK, 1993.

[13] M. Sung and N. Chang and J. Cho and H. Shin, Performance Analysis of the *BusNet* Protocol for Backplane Bus-Based Interprocessor Communication, http://cselab.snu.ac.kr/pub/pdcs00-f.pdf, 2000.

[14] M. H. MacDougall, Simulating Computer Systems, MIT Press, Cambridge, MA, 1987.

[15] K. Tindell and A. Burns and J. Wellings, Analysis of Hard Real-Time Communications, Real Time Systems 9 (1995) 147-171.