T-SE/13/8/14549

Modeling and Verification of Real-Time Protocols for Broadcast Networks

P. Jain
S. S. Lam

Reprinted from: IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, Vol. SE-13, No. 8, AUG. 1987.

Modeling and Verification of Real-Time Protocols for Broadcast Networks

PRADEEP JAIN AND SIMON S. LAM, FELLOW, IEEE

***Abstract*—A class of demand-assigned multiple-access (DAMA) protocols have been proposed for high-speed local area networks (LAN's) that offer integrated services for data, voice, video, and facsimile traffic. These protocols exploit the directionality of signal propagation and implement stringent real-time constraints to achieve collision-freedom. Correct implementation of DAMA protocols will require a very careful analysis of time-dependent interactions using a formal method. To date, most verification methods have been focused on asynchronous communication over point-to-point links.**

We present a model of broadcast bus networks and a methodology for verifying DAMA protocols. The novel features of our model include the ability to specify broadcast channels as well as the specification and analysis of real-time behavior. The broadcast characteristics of the communications medium are captured by some simple axioms. Real time is modeled as a discrete quantity using clocks and time variables. Real-time properties are specified by safety assertions, which are proved by showing that the programs comprising the protocol satisfy certain invariant assertions. These assertions are on history variables and can be verified by treating the protocol processes as sequential programs.

***Index Terms*—Broadcast channel, local area networks, multiple-access protocols, real-time constraints, verification.**

I. Introduction

DUE to the inherent nondeterminacy and distributed nature of communication protocols, debugging them is not easy. In particular, many of the errors are time-dependent and nonreproducible. Therefore, it is necessary that communication protocols be formally modeled and analyzed.

In this paper we present a methodology for modeling and analyzing broadcast bus networks. In a broadcast bus network, if two or more messages overlap in time at the same bus location, a collision results and the messages involved are garbled. Therefore, a multiple access protocol is needed to coordinate access to the bus. Demand-assigned multiple-access (DAMA) protocols belong to the class of protocols designed for this purpose [4], [5], [19]. They generally strive to provide real-time service and to avoid access conflicts by exploiting the directionality of signal propagation. DAMA protocols have been proposed for a new generation of very high speed local area networks (LAN's). The properties of interest for these protocols are: freedom from collision, fair service to all stations, and bounded packet delay.

Various analytical methods for protocol verification have been proposed in the literature [2], [3], [7], [8], [10], [12], [21], but none of these has been applied to DAMA protocols. Because of their real-time behavior and broadcast nature, DAMA protocols are more difficult to analyze than most protocols that have been formally verified. Several DAMA protocols have been proposed but, to date, their analysis has been ad hoc. There is clearly a need for a formal framework for analyzing these protocols.

In the remainder of this section, we briefly trace the evolution of DAMA protocols. Next we review existing protocol verification methods and examine their applicability to DAMA protocols. We then make a case for a new model for analyzing DAMA protocols.

A. Protocols for High-Speed LAN's

Of the various topologies for LAN's, the ring and broadcast bus configurations are the most popular.

The broadcast bus topology is a multipoint configuration. Taps serve as passive connection points between the bus and the stations. All stations can monitor the bus and each message transmitted can be received by all. The sharing of the medium gives rise to the problem of coordinating access to the bus.

When a contention scheme is used for multiple access, it is possible for collisions to occur. Such access conflicts are resolved probabilistically. A well-known example of this class is Ethernet's CSMA/CD protocol [11]. Ethernet-type LAN's were found to be simple to implement and reliable because the station-to-bus taps are passive. Contention protocols perform optimally under light load conditions. However, due to collisions, a fraction of the bandwidth is wasted and a bounded packet delay cannot be guaranteed. Moreover, the throughput performance degrades significantly as the following parameter increases in value

$$\alpha = \tau W / B$$

where τ is the end-to-end propagation delay, W is the channel bit-rate, and B is the number of bits per packet. This results in unacceptably low throughput for very high speed networks (50 Mbps or more) or networks with very long cables.

Manuscript received September 25, 1986. This work was supported by the National Science Foundation under Grant ECS 83-04734.

The authors are with the Department of Computer Sciences, University of Texas at Austin, Austin, TX 78712.

IEEE Log Number 8714549.

0098-5589/87/0800-0924$01.00 © 1987 IEEE

Ring networks provide an alternative whose throughput performance (for some protocols) does not degrade significantly as transmission speed increases. It is also possible to provide bounded packet delays in ring networks. Further, since the signal is regenerated at each node, greater distances can be covered.

However, ring networks have some drawbacks. In a ring configuration, repeaters are used to connect point-to-point links which form a closed loop. The repeaters are active elements and are more vulnerable to failures than passive taps. A break in any link or the failure of any repeater brings the entire network down. Ring access methods make use of tokens or time slots. Reliable operation of a ring network relies on the integrity of the transmission of a unique token, or on a unique marker and slot status. This makes the ring network even more vulnerable to errors. Also, it is more difficult to add new stations to a ring than to a bus.

More recently, a class of DAMA protocols has been proposed for broadcast bus networks [4]-[6], [19], [20]. These protocols incorporate some of the advantages of both ring networks and Ethernet. Stations are connected to a bus via taps. Since taps are passive elements, as opposed to the repeaters in a ring which are active elements, these networks are less susceptible to node and link failures. The protocols rely on observable channel events (such as the beginning or the end of a message, the bus becoming idle, etc.) for a station to determine when to transmit. Effectively, they employ a token-passing mechanism that is implicit and efficient, thereby circumventing the problems associated with explicit tokens in ring networks mentioned above.

Compared to CSMA/CD, DAMA schemes provide collision-free access to a shared bus. A station can transmit only when it holds the transmission right. The transmission right is allocated to each station in a fixed order. These networks are able to provide better channel throughput than CSMA/CD as well as bounded packet delay. In addition, both throughput and delay performance can be made much less sensitive to the parameter α, and to N, the number of stations connected to the bus. Such schemes are thus particularly attractive for the next generation of LAN's operating at speeds of 50-200 Mbps and providing integrated services (data, voice, video, facsimile, etc.). However, correct implementation of these DAMA protocols will require a very careful analysis of time-dependent interactions using a formal method.

B. Previous Work on Modeling and Verification of Protocols

From the point of view of analysis, DAMA protocols have two characteristics that distinguish them from protocols studied in the verification literature. First, the communication channel is a broadcast bus, i.e., a message transmitted by one station is received by all others. This is fundamentally different from point-to-point links where there is one sender and one receiver. Second, the DAMA protocols are real-time protocols which require certain time-constraints to be met for their correct functioning [18]. These time constraints arise from the use of timeouts and imposition of bounds on delays, which are required to ensure collision-free access to the medium and fair service to all stations.

Many models have been proposed to analyze communication protocols over point-to-point links. The most common of these are the finite state machine (FSM) model [2], [7], the Petri net model [14] and the programming language model [3], [8], [12], [13]. Reachability analysis is used to analyze an FSM or Petri net system. The major drawback of this approach is that the reachability graph can be very large, and sometimes infinite. This difficulty is compounded if time is included in the model. The programming language model is the most general and can be used for any number of stations, but its success depends upon a human verifier's ability to identify the proper invariants needed to verify the desired protocol property.

Time modeling is a critical factor in the analysis of DAMA protocols. Models of Petri nets enhanced with time variables have been proposed [1], [15], [16], [22]. In these models a time interval is associated with each transition. These time intervals specify the temporal conditions for the firing of transitions. A timed reachability analysis is then performed. But the construction and analysis of a timed reachability graph are even more intractable than ordinary reachability graphs. More recently, a model for time-dependent distributed systems has been proposed [18]. In this model, time is modeled explicitly by means of clocks, time variables, and time events. This model has been used to verify safety, liveness, and real-time properties of data-link and transport protocols [17]. Real-time properties are verified as safety properties.

The broadcast nature of the communication medium adds another dimension to the problem of analyzing DAMA protocols. The number of stations becomes an important parameter for such a system. Analytical methods based on reachability graph generation cannot deal with a variable number of stations. Thus, using such an approach, it cannot be said that a system verified for a certain number of stations will also work correctly for a different number of stations. Furthermore, in a broadcast bus, modeling timing delays and constraints is even more crucial than in point-to-point links. The actual propagation of the signal along the channel has to be modeled. Also, there could be multiple messages in the channel, in different sections of the bus. The relative positions of these signals are also important. All these factors make the problem of modeling a broadcast bus unique. To date, almost all verification techniques have focused on asynchronous communication over point-to-point channels.

C. Analysis of DAMA Protocols

Many DAMA protocols have been proposed in the literature. But there is no general scheme to specify them. Although some authors have attempted to use finite state machines for specification, their representations are often too informal and too coarse [6], [19]. That is, the states and transitions are described in general terms, and important details are hidden. These specifications are ambiguous and incomplete. The analysis of these protocols, if any, has been ad hoc. Typically, the argument resorts to a case analysis, and the reasoning is operational in nature. In most cases the proof is based on time–space diagrams, which are a graphical representation of the communication among the stations. Even the more "formal" of these proofs essentially base their argument on a verbal description of the time–space diagrams. Each proof is carried out for a particular protocol and relies heavily on its operational details. Such proofs provide little help in analyzing other protocols.

In this paper we present a specification model and an analysis procedure for verifying the collision-freedom property of DAMA broadcast bus protocols. The proposed method is based on a programming language model. Our model incorporates the real-time behavior of these protocols, as well as the broadcast nature of the communication medium. The method is general and can be applied to various protocols based on the same or similar network configurations. A formal proof method is proposed. The proof method makes use of program verification techniques that are familiar and well-understood.

In Section II, we describe our model for broadcast bus protocols, and illustrate it by specifying a protocol example. In Section III, we present our proof system, and use it to show that the protocol specified in Section II is collision-free. In Section IV, we point out certain special features of our model.

II. Modeling Broadcast Bus Networks

In this section, we describe our approach to modeling and analyzing broadcast bus networks. We present our model for specifying a broadcast bus network in Section II-B. In Section II-C, we illustrate our model by using it to represent a DAMA protocol (due to Eswaran, Hamacher, and Shedler [4], henceforth referred to as the EHS protocol).

Before presenting the model, a brief description of the configurations and access methods for DAMA protocols is given.

A. Network Configurations and Access Mechanisms

Three configurations for networks based on a broadcast bus are popular:

1) Bidirectional Bus System (BBS): The signal transmitted by a station propagates in both directions and is received by all other stations (Fig. 1). The access mechanism used is called the scheduling delay access method [5], which attempts to stagger the starting times of transmissions following the end of a successful transmission. This is achieved by letting each station schedule its transmission after a delay. This delay depends upon the station's position with respect to the station that transmitted most recently.

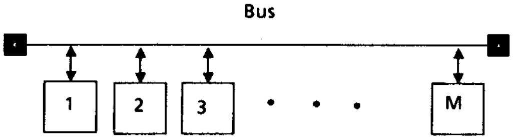

Fig. 1. Topology of a bidirectional bus system.

2) Unidirectional Bus System (UBS): The transmitted signal travels in only one direction. This directionality imposes an ordering on the stations. One way of achieving broadcast communication is to use two unidirectional buses, one in each direction (Fig. 2). Another way is to fold the cable (Figs. 3 and 4). There are two logical channels, one inbound (for receiving) and the other outbound (for sending). The access mechanism used is called the attempt and defer method [19], [20]. When a station is ready to transmit, it monitors the channel and waits until it is idle, at which time the station begins to transmit. However, if another transmission from upstream is detected, then this station immediately aborts its own transmission and defers to the one from upstream.

3) Bidirectional Bus with Control (BBC): A bidirectional bus is used along with a control wire which is used to control access to the bus (Fig. 5). The control wire imposes an ordering on the stations connected to the bus, which is logically the same as the ordering on a unidirectional bus. The access mechanism used is called the reservation access method [4], [6]. The control wire is used by a station to place a reservation for bus access. A consensus is reached among the ready stations on the basis of their positions on the wire.

B. The Model

In this section we describe how a broadcast bus network is formally specified in our model.

The network is modeled by a system of concurrent processes. Each station attached to the bus is represented by a *station process*. In the BBS and UBS configuration with two buses, each bus is modeled by a *bus process*. In the UBS configuration using a folded cable, the bus is represented by two bus processes: one for the inbound segment and the other for the outbound segment. In BBC systems, the bus is modeled by a bus process and the control wire is modeled by a *wire process*. (The term *channel process* will be used as a generic term for both a bus process and a wire process.)

Station Process: A station is specified by a set of local variables, a set of shared variables, and a sequential program implementing the access algorithm. (The local variables may include time variables needed to implement timing constraints.) Shared variables are shared with a user process or a channel process. The language used is Pascal, augmented with the following temporal primitives. (The axioms for these primitives will be stated in Section III-A.)

1) **set**

Let $v_1, v_2, \cdots, v_n$ be variables and $e_1, e_2, \cdots, e_n$ be expressions.

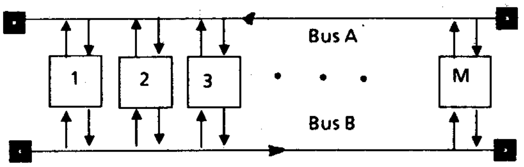

Fig. 2. Topology of a unidirectional bus system using two buses.

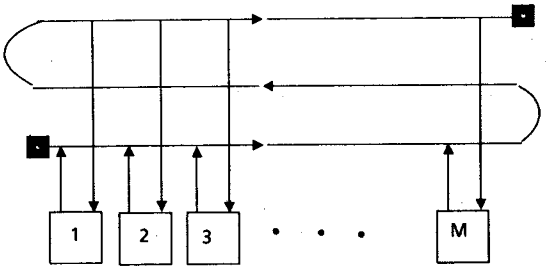

Fig. 3. Topology of a unidirectional bus system using a doubly folded cable.

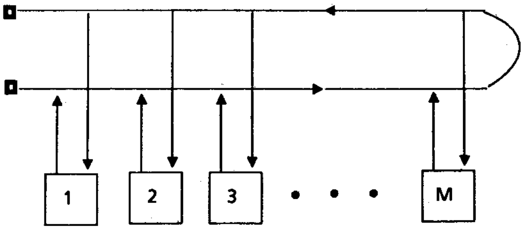

Fig. 4. Topology of a unidirectional bus system with a singly folded cable.

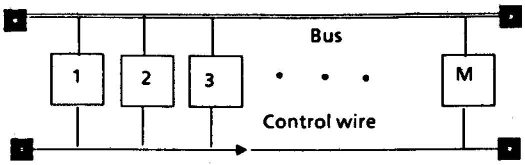

Fig. 5. Topology of a bidirectional bus with control.

The statement
set $v_1, v_2, \cdots, v_n := e_1, e_2, \cdots, e_n$
assigns e_1 to v_1, e_2 to v_2, $\cdots$, and e_n to v_n in one atomic operation.

2) **delay** T
The statement **delay** T causes the station process to halt and remain idle for time T.

3) **wait** C
The statement **wait** C, where C is a Boolean condition, causes the process to halt and remain idle until the specified condition C becomes true.

4) **wait** (C **or** T)
The statement **wait** (C **or** T), where C denotes a Boolean condition and T specifies a length of time, causes the process to halt and remain idle until the occurrence of the earlier of the events: condition C becomes true, or time T has elapsed since the beginning of the statement execution.

5) **wait** (C **for** T)
The statement **wait** (C **for** T), where C is a Boolean condition and T is a length of time, causes the process to halt and remain idle until the condition C has become true and remained true continuously for time T.
The parameter T can be omitted if it is 0, in which case this statement becomes equivalent to the **wait** statement above.

6) **transmit** T
The statement **transmit** T represents the transmission of a message for time T.

Each program is essentially a loop which a station process executes continually. The statements within the loop involve computations on local and shared variables, and sensing and setting of signals on the channels (to be described later).

Channel Process: The channel process specification consists of a description of the state of the channel, and the transition rules describing how the channel state changes with time. The bus and the control wire are divided into segments, the length of a segment being the distance the signal propagates on the bus (or wire) in one time unit. The segments are numbered 0 to N, with the numbers increasing along the control wire (or along the unidirectional bus). A station on the bus is referred to by its position on the bus, i.e., the station connected to the bus at position p will be called station p. (Therefore the station numbers need not be contiguous.)

The state of the channel consists of the state of each segment, i.e., the presence or absence of a signal in each segment. Thus the state of the channel at any given time is essentially described by specifying the portion(s) of the channel carrying a signal at that time. Transition rules are stated which describe how the status of each segment changes with time. The state transition rules capture the propagation of signals, transmitted by stations, along the channel.

This simple model of a channel is actually quite general: it can model simultaneous transmission by two or more stations, and can model the presence of several messages in the channel.

Interprocess Communication: In our model, interprocess communication is achieved by shared variables. A station or channel process has both local and shared variables. A user process has only shared variables (shared with a station process). Such a variable can be updated by the user process, thus causing a change in the state of the station process. Other than the specification of such state changes, user processes are not explicitly modeled.

A variable shared between a pair of processes (a station process and a channel process, or, a station process and a user process) can be read by both processes. It can be written by one process or both processes. A shared variable that can be written by a single process only is said to be an *exclusive-write* variable of that process. A shared variable that can be written by both processes is said to be a *mutual-write* variable.

Modeling of Time: The access methods rely on the observance of certain time constraints imposed on the system. Real time is modeled by means of a global clock,[1] time variables, and time events. Using the time variables, real-time constraints, such as timeouts and bounded delays, can be specified. The clock and time variables are discrete, i.e., they can have only integer values. A time event corresponds to a clock tick and marks the passage of one time unit. The real-time constraints and the time events impose a temporal ordering on the events of the collection of processes.

We assume that the execution of a **set** statement takes one time unit. Associating a delay of one time unit with each set statement ensures that there will be a unique system state for any time instant. If a set statement takes zero time, we would have two values for a variable at the same clock reading.

C. Example: Specification of the EHS Protocol

We now illustrate our model by specifying the EHS protocol.[2] The protocol is based on a BBC (bidirectional bus with control) system. Details of the BBC configuration are shown in Fig. 6.

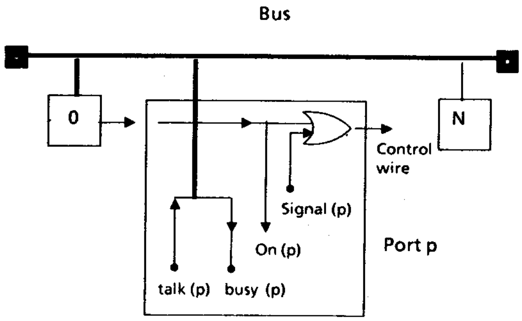

Fig. 6. Details of BBC configuration.

Each station port has a tap into the bus to transmit and receive packets and to receive a carrier sense signal. When the station transmits a message on the bus, it sets **talk**(p) to true. If **busy**(p) is true, station p detects a carrier on the bus. A control wire connects all the ports as shown in the figure. If **on**(p) is true, station p detects a signal on the control wire from the left. When the station wishes to transmit, it makes its intention known to the stations down the wire by sending a signal on the wire, i.e., it sets **signal**(p) to true.

Let $\boldsymbol{D}$ denote the end-to-end propagation delay on the bus and $\boldsymbol{R}(p)$ denote the propagation delay on the control wire from port p to the port at the right end of the wire. Let $\text{bdelay}_{pp'}$ denote the propagation delay along the bus from station p to station p'.

To transmit a packet, station p follows this protocol:

- Wait until the bus is observed (by port p) to be continuously idle throughout a time interval of length $2\boldsymbol{D}$.
- Set **signal**(p) to true.
- Wait for a time interval $\boldsymbol{R}(p) + \boldsymbol{D}$.
- Wait until the bus and the control wire are observed (by port p) to be both idle (i.e., **busy**(p) = false and **on**(p) = false); then begin transmission of the packet, simultaneously setting **signal**(p) to false.

The initial wait of $2\boldsymbol{D}$ before reserving the channel is to ensure that once p has sent a message, every other ready station gets a chance to transmit before p sends another message. Setting **signal**(p) to true informs every station downstream of p's intention to transmit. The waiting period $\boldsymbol{R}(p) + \boldsymbol{D}$ is to avoid a collision between p's transmission and a transmission by a downstream station. A collision between p's message and one coming from upstream is avoided by the requirement that p start transmission only after observing the bus and the wire to be idle at the same time.

Model for the EHS Protocol: We now formally specify the EHS protocol described informally above.

Station process p shares variables **talk**(p) and **busy**(p) with the bus process, and variables **signal**(p) and **on**(p) with the wire process.

Variable	Sharing processes	Written by	Read by
talk(p)	station p, bus	station p	station p, bus
busy(p)	station p, bus	station p, bus	station p, bus
signal(p)	station p, wire	station p	station p, wire
on(p)	station p, wire	station p, wire	station p, wire

When station p starts transmitting a message on the bus, it sets the variable **talk**(p) to true; **talk**(p) is set to false at the end of the transmission. When station p sends a signal on the control wire, the variable **signal**(p) is set to true; **signal**(p) is set to false when the station turns off its signal on the control wire. **talk**(p) and **signal**(p) affect the state of the channels, and can be considered as outputs of the station process.

The status of the bus is represented by a Boolean array called **busy**, and that of the control wire by the Boolean array **on**. There is one element in the arrays for each segment of the bus/wire. **busy**(p) represents the status of the bus at segment p, and **on**(p) represents the status of the control wire at segment p.

busy, on : array [0..N] of Boolean

busy(p) :true, if a carrier is present on the bus at position p
false, otherwise

on(p) :true, if a signal is present on the control wire at position p
false, otherwise

The transitions rules for the channel processes will be stated in Section III.

During its operation, station p continuously monitors the bus and the wire, and takes action based on the status of the bus and the wire as perceived by it, i.e., the values

[1] The assumption of a global clock can be relaxed using an approach similar to that of Shankar and Lam [18].

[2] We have chosen this protocol as an example primarily because its original specification in [4] is much clearer than the specifications of other DAMA protocols available in the literature. From a reliability point of view, the BBC configuration is less desirable than the other configurations.

of **busy**(p) and **on**(p). Therefore, **busy**(p) and **on**(p) can be considered as inputs to the station process.

The program for station p is given below. The variable *packet_to_send* is shared between the station process and a separate user process; it is set to true by the user process whenever it needs to transmit a packet. It is set to false by the station process after the packet has been transmitted. *transmission-time* denotes the time taken for a packet transmission. The initial value of each variable is *false*.

```
begin
   repeat
      wait ( packet__to__send );
      wait ( ( not busy) for 2*D );
      set signal := true;
      delay ( D + R(p) )
      wait (not on and not busy);
      set signal,talk := false,true;
      transmit (transmission-time);
      set talk := false;
      packet__to__send := false;
   forever
end.
```

III. Verification

In this section we present our proof system, and as an example, we prove that the EHS protocol specified in Section II-C is collision-free.

A. Proof System

We now give the axioms for the station and channel processes and describe our proof system. In the following, the variable "τ" refers to the value of the global time.

Axioms for the Station Process: The primitives that can be used in the description of a station process include those in a high-level procedural language: assignment, alternation, and repetition. The axioms for these constructs are the same as those described by Hoare in [9]. Apart from these procedural primitives, a process specification also makes use of the temporal primitives described earlier. These primitives specify the condition under which a process halts or resumes processing.

The shared variables of a station process can be partitioned into two subsets, depending upon whether they are exclusive-write variables of the station process or not. Local variables of a process are, by definition, exclusive-write variables of that process. The axioms for the temporal primitives, except for the **set** primitive, essentially state that during the execution of each such primitive, the exclusive-write variables of the station process do not change.

The predicate stated before a primitive is its precondition, and τ in the predicate refers to the global clock at that point in the program, i.e., before the execution of the statement. We will denote this value of the global clock as τ_{begin}. The predicate stated after a primitive is its postcondition, and τ in the predicate refers to the value of the global clock at that point, i.e., after the execution of the statement. This value of the global clock will be referred to as τ_{end}.

Let V be the set of exclusive-write variables of the station process, W be the set of mutual-write variables, and X be the set of shared variables (of the station process) which are exclusive-write variables of other processes. We shall also use the following notation:

$\{v_1, v_2, \cdots, v_m\}$	subset of V
$\{w_1, w_2, \cdots, w_n\}$	subset of W
$e_1, \cdots, e_{m+n}$	expressions
$e(\tau)$	value of expression e at time τ
P	predicate over variables in V
C	predicate over variables in $W \cup X$.

1) **set**

The **set** command takes one time unit (it is performed within one tick of the global clock).

$\{ (\tau = \tau_{\text{begin}}) \text{ and } P^{v_1, v_2, \cdots, v_m}_{e_1(\tau), e_2(\tau), \cdots, e_m(\tau)} \}$
set $v_1, \cdots, v_m, w_1, \cdots, w_n := e_1, \cdots, e_{m+n}$
$\{ (\tau = \tau_{\text{end}} = \tau_{\text{begin}} + 1) \text{ and } P(\tau) \}$

2) **delay** T

The **delay** T command takes time T to execute.

$\{ (\tau = \tau_{\text{begin}}) \text{ and } P(\tau) \}$
delay T
$\{ (\tau = \tau_{\text{end}} = \tau_{\text{begin}} + T) \text{ and}$
$(\forall t, \tau_{\text{begin}} \leq t \leq \tau_{\text{end}}: P(t)) \}$

3) **wait** C

The time taken for the **wait** C command is indeterminate, as it depends upon when C becomes true.

$\{ (\tau = \tau_{\text{begin}}) \text{ and } P(\tau) \}$
wait C
$\{ (\tau = \tau_{\text{end}} \geq \tau_{\text{begin}}) \text{ and}$
$(\forall t, \tau_{\text{begin}} \leq t \leq t_{\text{end}}: P(t)) \text{ and } C(\tau) \}$

4) **wait** (C **or** T)

$\{ (\tau = \tau_{\text{begin}}) \text{ and } P(\tau) \}$
wait (C **or** T)
$\{ (\tau = \tau_{\text{end}}) \text{ and } (\tau_{\text{begin}} \leq \tau \leq \tau_{\text{begin}} + T)$
$\text{and } (\forall t, \tau_{\text{begin}} \leq t \leq \tau_{\text{end}}: P(t)) \text{ and}$
$(C(\tau) \text{ or } (\tau_{\text{end}} = \tau_{\text{begin}} + T)) \}$

5) **wait** (C **for** T)

$\{ (\tau = \tau_{\text{begin}}) \text{ and } P(\tau) \}$
wait (C **for** T)
$\{ (\tau = \tau_{\text{end}} \geq \tau_{\text{begin}} + T) \text{ and}$
$(\forall t, \tau_{\text{begin}} \leq t \leq \tau_{\text{end}}: P(t)) \text{ and}$
$(\forall t, 0 \leq t \leq T: C(\tau - t)) \}$

6) **transmit** T

The **transmit** T statement is similar to the **delay** statement as the station variables are not changed.

$\{ (\tau = \tau_{\text{begin}}) \text{ and } P(\tau) \}$
transmit T
$\{ (\tau = \tau_{\text{end}} = \tau_{\text{begin}} + T) \text{ and}$
$(\forall t, \tau_{\text{begin}} \leq t \leq \tau_{\text{end}}: P(t)) \}$

Using the axioms stated above, we can verify if certain assertions hold at various points in the program. Typically, we are interested in proving some invariant assertion for the station process, which will then be used to verify the correctness properties of the protocol.

History Variables: The proof system will involve assertions on the past values of the shared variables. Therefore it is convenient to define the following history variables for the BBC system. (History variables for the other two configurations can be defined analogously.) In the following, p denotes a channel segment number and τ denotes the global time. (p ranges over integers $0..N$, and τ ranges over integers.)

signal(p, τ): value of **signal**(p) at time τ

true, if there is a station at position p and
it is sending a signal on the wire at
time τ
false, otherwise

on(p, τ): represents the presence of a signal on the wire at position p at time τ

true, if there is a signal on the wire at position p
at time τ
false, otherwise

talk(p, τ): value of **talk**(p) at time τ

true, if there is a station at position p and
it is transmitting on the bus at time τ
false, otherwise

busy(p, τ): represents the presence of a carrier on the bus at position p at time τ

true, if there is a carrier on the bus at position
p at time τ
false, otherwise

Axioms for the Channel Processes: The following axioms define the state transition rules for the channel processes.

1) For the control wire:

a) $\forall p\colon 0 < p \le N, \forall \tau > 0$:
$\text{on}(p, \tau) = \text{on}(p-1, \tau-1)$ or $\text{signal}(p-1, \tau-1)$

b) $\forall \tau : \text{on}(0, \tau) = \text{false}$
(the station at the head-end never senses the wire to be on)

2) For the bidirectional bus: We define two new predicates for the bus process: $\textbf{busy}_{\text{LR}}$ and $\textbf{busy}_{\text{RL}}$

$\text{busy}_{\text{LR}}(p, \tau)$ = true, if there is a carrier on the bus at location p at time τ going from *left to right*
false, otherwise

$\text{busy}_{\text{RL}}(p, \tau)$ = true, if there is a carrier on the bus at location p at time τ going from *right to left*
false, otherwise

a) $\forall p\colon 0 < p \le N, \forall \tau > 0$:
$\text{busy}_{\text{LR}}(p, \tau) = \text{busy}_{\text{LR}}(p-1, \tau-1)$ or $\text{talk}(p, \tau)$

b) $\forall \tau : \text{busy}_{\text{LP}}(0, \tau) = \text{talk}(0, \tau)$

c) $\forall p\colon 0 \le p < N, \forall \tau > 0$:
$\text{busy}_{\text{RL}}(p, \tau) = \text{busy}_{\text{RL}}(p+1, \tau-1)$ or $\text{talk}(p, \tau)$

d) $\forall \tau : \text{busy}_{\text{RL}}(N, \tau) = \text{talk}(N, \tau)$

e) $\forall p, \forall \tau$:
$\text{busy}(p, \tau) = \text{busy}_{\text{LR}}(p, \tau)$ or $\text{busy}_{\text{RL}}(p, \tau)$

Axioms for the channel processes for the unidirectional bus systems are similar.

Basic Theorems for Channel Processes: Using the axioms stated above, the following theorems can be proved. These theorems are essentially timing relations between events on the channel and their effects at remote points. Along with the invariant assertions proved for the station process, these theorems are used to verify protocol correctness properties.

Theorem 1: The following relations hold for the wire process:

a) $\forall p, 0 < p \le N, \forall \tau$: [$\text{on}(p, \tau) \Rightarrow$
$\exists p' < p$: $\text{signal}(p', \tau - \text{wdelay}_{pp'})$]

b) $\forall p, \forall \tau$: [$\text{signal}(p, \tau) \Rightarrow$
$\forall p' > p$: $\text{on}(p', \tau + \text{wdelay}_{pp'})$]

where $\text{wdelay}_{pp'}$ is the propagation delay between p and p' along the wire.

The theorem states that for every time instant that station p detects a signal on the wire, there exists a station to its left, p', which must have its signal set to true at $\text{wdelay}_{pp'}$ before this particular time instant. Similarly, for every time instant that station p has its signal set to true, each station to its right, p', will sense the wire to be on after a delay which is equal to $\text{wdelay}_{pp'}$. The proof of the theorem is given in the Appendix.

Theorem 2: The following relations hold for the bus process:

a) $\forall p, \forall \tau$: [$\text{busy}(p, \tau) \Rightarrow$
$\exists p'$: $\text{talk}(p', \tau - \text{bdelay}_{pp'})$]

b) $\forall p, \forall \tau$: [$\text{talk}(p, \tau) \Rightarrow$
$\forall p'$: $\text{busy}(p', \tau + \text{bdelay}_{pp'})$]

where $\text{bdelay}_{pp'}$ is the propagation delay between stations p and p' along the bus.

Theorem 2 is similar to Theorem 1, and gives the timing relations for the bus process. The proof of Theorem 2 is given in the Appendix.

The stations interact with each other only in directly via the communication bus and control wire. The effect of the activity of one station (starting or stopping transmission, turning wire signal on or off) is seen by each of the other stations as a change in the status of the bus or wire. This effect is felt at the other station after a time delay which is equal to the propagation delay between the two stations. Our model captures these timing relations in a simple and concise manner.

Proving Properties: The correctness property of interest is stated as a predicate over the variables of the protocol system. To verify a given property, we associate invariant assertions with the program for the station process. These invariant assertions can be verified using the axioms for the station process stated earlier. The timing relations are captured by Theorems 1 and 2. The correctness proof then reduces to showing that the invariant assertions, along with the two theorems, imply the correctness property.

B. Example: Proof of the EHS Protocol

We make the following assumptions about the configuration of the EHS protocol:

1) The stations are separated by an integral number of segments of both the bus and the control wire. The minimum distance between two stations is one segment.

2) The time unit is larger than the time to detect a signal or carrier on the control wire or bus.

To prove that the EHS protocol is collision-free, we use another predicate **clear** (p, τ), which is defined as:

$$\textbf{clear}(p,\tau) \equiv \text{not } \textbf{on}(p,\tau) \text{ and not } \textbf{busy}(p,\tau) \text{ and } \forall\tau'\text{: } 0 \le \tau' \le D + R(p)\text{: } \textbf{signal}(p,\tau - \tau') \tag{A1}$$

clear (p, τ) describes the condition which must hold for a station to transmit. The condition states that a station can start transmitting only when both the bus and the wire are idle, and the station has been signaling on the wire for the last $D + R(p)$ time units.

The following assertion holds invariantly for the station program of the EHS protocol:

$$\textbf{talk}(p,\tau) \Rightarrow \exists\tau' < \tau\text{: } \left[\textbf{clear}(p,\tau') \text{ and } \forall\tau''\text{: } \tau' < \tau'' \le \tau\text{: } \textbf{talk}(p,\tau'')\right] \tag{A2}$$

The above assertion states that if station p is talking at time τ, then the condition for starting transmission must have been true (**clear** was true) at some previous time τ' and the station has been talking since then.

The annotated program for the EHS protocol is shown on the next page. It was easily be checked that the above assertion is indeed invariant.

The condition for collision-freedom is:

$$\forall p, \forall p', \forall\tau, \forall\tau'\text{: } \left[\textbf{talk}(p,\tau) \text{ and } \textbf{talk}(p',\tau') \Rightarrow (p = p') \text{ or } \left(|\tau - \tau'| > \text{bdelay}_{pp'}\right)\right] \tag{A3}$$

The proof that (A3) is a necessary and sufficient condition for collision-freedom is given in the Appendix. Using Theorems 1 and 2, and the invariant assertion stated above (A2), it can be shown that the EHS protocol satisfies the condition for collision-freedom (A3). The proof is given in the Appendix.

```
begin
  repeat
    wait ( packet_to_send );
    wait ( (not busy) for 2*D );
      { $\forall\tau', 0 \le \tau' \le 2*D$: not busy (p, $\tau - \tau'$) }
    set signal := true;
    delay ( D + R(p) );
      { $\forall\tau', 0 \le \tau' \le D + R(p)$: signal (p, $\tau - \tau'$) }
    wait (not busy and not on);
      { not busy (p, $\tau$) and not on (p, $\tau$) and
          $\forall\tau', 0 \le \tau' \le D + R(p)$: signal (p, $\tau - \tau'$) }
                                          ( $\equiv$ clear (p, $\tau$) )
    set signal,talk := false, true;
      { $\exists\tau' < \tau$ : [ clear (p, $\tau'$)
               and $\forall\tau'', \tau' < \tau'' \le \tau$ : talk (p, $\tau''$)] }
    transmit (transmission-delay);
      { $\exists\tau' < \tau$ : [ clear (p, $\tau'$)
               and $\forall\tau'', \tau' < \tau'' \le \tau$ : talk (p, $\tau''$)] }
    set talk := false;
    packet_to_send := false;
  forever
end.
```

IV. Some Features of Our Model

In our method, proving a correctness property comprises three steps: deriving the timing relations (Theorems 1 and 2 of Section III-A), proving invariants for the station program, and showing that the correctness property follows from the timing relations and program invariants.

1) The theorems for the channel processes stated in Section III essentially state the timing relations between events at a station and their effects at remote points in a simple and concise manner. It is possible to state the timing relations in this manner because of the way the channel is being modeled. Treating time as a discrete quantity enables us to treat the channel as made up of an integral number of segments, which permits inductive reasoning. We are thus able to identify the portions of the channel that are busy at any time, and track the propagation of the signal along the channel.

The channel processes, and hence the timing relations, remain unaltered for different protocols based on the same configuration. Thus, this method has more general applicability than other attempts at analyzing DAMA protocols.

2) The invariant of interest for the station program is the assertion that holds during transmission (in our example, assertion (A2) in Section III-B). This invariant captures the essence of the multiple-access algorithm followed by each station. This assertion involves only one station—the interaction among the stations has already been captured by the timing relations stated earlier.

This invariant can be proved by examining just one process—that of a station. Thus, the problem reduces to a sequential program verification. Checking for noninterference is not required, even though shared variables are being used. This is because the channel processes do not interfere with a station process, and the assertions used in the verification of a station process refer only to its own state and the *past* state of channel processes. The station process does interfere with the channel processes; the effect of that interference is incorporated in the channel axioms.

3) The proof is independent of the number of the stations and their positions in the network.

The condition for collision-freedom is very general, and can be used for all such protocols.

The specification of the station process and the channel axioms capture all the relevant physical details. The proof does not have to resort to arguments involving any more physical details.

An advantage of analyzing protocols in a formal manner such as this is that it can help highlight some implementation constraints, which would be difficult to identify using an informal approach. For example, the proof of the EHS protocol clearly highlights the requirement that the bus should be faster than the control wire ($\text{wdelay}_{pp'} \geq \text{bdelay}_{pp'}$) for the protocol to provide collision-free service. Also, it can be seen that a station need not signal for $D + R(p)$ time before starting transmission! For collision-freedom, it is sufficient if the station waits for $R(p) + \text{bdelay}_{Np}$ (roundtrip delay from station p to the end station along the wire and from the end station to station p along the bus).

V. Conclusions

DAMA protocols are particularly suitable for the next generation of LAN's, which will operate at very high speeds and will offer integrated services for data, voice, video, and facsimile traffic. These protocols exploit the directionality of signal propagation and implement stringent timing constraints to achieve collision-freedom. Correct implementation of these protocols will require a very careful analysis of time-dependent interactions using a formal method. To date, most verification methods have been focused on asynchronous communication over point-to-point links.

We have proposed a model for verifying real-time protocols for high-speed LAN's. The novel features of our methodology are the ability to model broadcast channels as well as the specification and analysis of the real-time behavior of these protocols. The method has potential applicability to a wide class of broadcast bus networks. A formal proof technique is used, instead of resorting to case analysis and informal arguments. The proof method is simple, and the verification problem essentially reduces to a sequential program verification. We have demonstrated the use of our model by specifying an example protocol in our model and proving it to be collision-free.

Appendix
Proofs of Theorems and the EHS Protocol

The axiom numbers that appear in the following proofs refer to the axioms for channel processes, as stated in Section III-A.

Theorem 1: The following relations hold for the wire process:

a) $\forall p, \forall \tau$: [on $(p, \tau) \Rightarrow \exists p' < p$: signal $(p', \tau - \text{wdelay}_{pp'})$]
b) $\forall p, \forall \tau$: [signal $(p, \tau) \Rightarrow \forall p' > p$: on $(p', \tau + \text{wdelay}_{pp'})$]

where $\text{wdelay}_{pp'}$ is the propagation delay between stations p and p' along the wire.

Proof of Theorem 1-a): By induction on p.

Base case:

$p = 0$

on $(0,\tau) =$ false (axiom 1b)

Induction Hypothesis (I H):

Assume that it holds for p

on $(p,\tau) \Rightarrow \exists p' < p$: signal $(p',\tau - (p - p'))$ (I H)

on $(p+1,\tau) =$ on $(p,\tau - 1)$ or signal $(p,\tau - 1)$ (axiom 1a)

$\Rightarrow \exists p' < p$: signal $(p',\tau - 1 - (p - p'))$ or signal $(p,\tau - 1)$ (I H)

$\equiv \exists p' < p$: signal $(p',\tau - (p+1 - p'))$ or signal $(p,\tau - (p+1 - p))$

$\equiv \exists p' < p+1$: signal $(p',\tau - (p+1 - p'))$

Hence, $\forall p, \forall \tau$: [on $(p, \tau) \Rightarrow \exists p' < p$: signal $(p', \tau - \text{wdelay}_{pp'})$] □

Proof of Theorem 1-b): By induction on p'.

Base Case:

$p' = p + 1$

signal $(p,\tau) \Rightarrow$ on $(p+ 1,\tau+ 1)$ (axiom 1a)

Induction Hypothesis:

Assume that it holds for $p' > p$

signal $(p,\tau) \Rightarrow$ on $(p',\tau + p' - p)$ (I H)

$\Rightarrow$ on $(p'+ 1, \tau + p' - p + 1)$ (axiom 1a)

$\equiv$ on $(p'+ 1, \tau+ (p'+ 1) - p')$

Hence, signal $(p, \tau) \Rightarrow \forall p' > p$: on $(p', \tau+ p' - p)$ □

Theorem 2: The following relations hold for the bus process:

a) $\forall p, \forall \tau$: [busy $(p,\tau) \Rightarrow \exists p'$: talk $(p',\tau - \text{bdelay}_{pp'})$]
b) $\forall p, \forall \tau$: [talk $(p,\tau) \Rightarrow \forall p'$: busy $(p',\tau + \text{bdelay}_{pp'})$]

where $\text{bdelay}_{pp'}$ is the propagation delay between stations p and p' along the bus.

Before proving the theorem, we prove four lemmas.

Lemma 1: $\text{busy}_{\text{LR}}(p,\tau) \Rightarrow \exists p' \leq p : \text{talk}\ (p',\tau - \text{bdelay}_{pp'})$.

Proof of Lemma 1: By induction on p.

Base case:

$p = 0$

$\text{busy}_{\text{LR}}(0,\tau) \Rightarrow \text{talk}\ (0,\tau)$ (axiom 2b)

Induction hypothesis:

Assume that it holds for p

$\text{busy}_{\text{LR}}(p,\tau) \Rightarrow \exists p' \leq p : \text{talk}\ (p',\tau - (p - p'))$ (I H)

$\text{busy}_{\text{LR}}\ (p+1,\tau) = \text{talk}\ (p+1,\tau)$ or $\text{busy}_{\text{LR}}(p,\tau - 1)$ (axiom 2a)

$\Rightarrow \text{talk}\ (p+1,\tau)$ or $\exists p' \leq p : \text{talk}\ (p',\tau - 1 - (p - p'))$ (I H)

$\equiv \text{talk}\ (p+1,\tau - (p+1 - (p+1)))$ or $\exists p' \leq p : \text{talk}\ (p',\tau - (p+1 - p'))$

$\equiv \exists p' \leq p+1 : \text{talk}\ (p,\tau - (p - p'))$

Hence, $\forall p$, $\forall \tau$: $\text{busy}_{\text{LR}}\ (p,\tau) \Rightarrow \exists p' : \text{talk}\ (p',\tau - \text{bdelay}_{pp'})$ □

Lemma 2: $\text{busy}_{\text{RL}}(p,\tau) \Rightarrow \exists p' \geq p : \text{talk}\ (p',\tau - \text{bdelay}_{pp'})$.

We define $x = N - p$, $x' = N - p'$

$$\text{busy}_{\text{RL}}(N - x,\tau) \Rightarrow \exists x' \leq x : \text{talk}\ (N - x',\ \tau - (x - x')).$$

Proof of Lemma 2: By induction on x.

Base case:

$x = 0$

$\text{busy}_{\text{RL}}(N,\tau) \Rightarrow \text{talk}\ (N,\tau)$ (axiom 2d)

Induction Hypothesis:

Assume that it holds for x

$\text{busy}_{\text{RL}}(N - x,\tau) \Rightarrow \exists x' \leq x : \text{talk}\ (N - x',\ \tau - (x - x'))$ (I H)

$\text{busy}_{\text{RL}}(N - (x + 1),\tau) \Rightarrow \text{busy}_{\text{RL}}(N - x,\ \tau - 1)$ or $\text{talk}\ (N - (x + 1),\ \tau)$ (axiom 2c)

$\Rightarrow \exists x' < x : \text{talk}\ (N - x',\ \tau - 1 - (x - x'))$ or $\text{talk}\ (N - (x+1),\ \tau)$ (I H)

$\equiv \exists x' < x : \text{talk}\ (N - x',\ \tau - 1 - (x - x'))$ or $\text{talk}\ (N - (x+1),\ \tau - 1 - (x - (x+1)))$

Hence, $\text{busy}_{\text{RL}}(p,\tau) \Rightarrow \exists p' \geq p{:}\text{talk}\ (p',\ \tau - \text{bdelay}_{pp'})$ □

Lemma 3: $\text{talk}\ (p,\ \tau) \Rightarrow \forall p' \geq p{:}\ \text{busy}_{\text{LR}}(p',\ \tau{+}\text{bdelay}_{pp'})$.

Proof of Lemma 3: By induction on p'.

Base case:

$p' = p$

$\text{talk}\ (p,\ \tau) \Rightarrow \text{busy}_{\text{LR}}(p,\ \tau)$ (axiom 2a)

Induction hypothesis:

Assume that it holds for $p' > p$

$\text{talk}\ (p,\ \tau) \Rightarrow \text{busy}_{\text{LR}}(p',\ \tau + p' - p)$ (I H)

$\Rightarrow \text{busy}_{\text{LR}}(p' + 1,\ \tau + (p' + 1) - p)$ (axiom 2a)

Therefore, $\text{talk}\ (p,\tau) \Rightarrow \forall p' \geq p{:}\ \text{busy}_{\text{LR}}(p',\ \tau + p' - p)$ □

Lemma 4: $\text{talk}\ (p,\tau) \Rightarrow \forall p' \leq p{:}\ \text{busy}_{\text{RL}}(p',\ \tau{+}\ \text{bdelay}_{pp'})$.

Proof of Lemma 4: By induction on $p - p'$ $(=x)$.

Base case:

$x = 0\ (\equiv p = p')$

$\text{talk}\ (p,\ \tau) \Rightarrow \text{busy}_{\text{RL}}(p,\tau)$ (axiom 2c)

Induction hypothesis:

Assume that it holds for $p - p' = x$

$\text{talk}\ (p,\ \tau) \Rightarrow \text{busy}_{\text{RL}}(p',\tau + p - p')$

$\equiv \text{busy}_{\text{RL}}(p - x,\ \tau + x)$

$\Rightarrow \text{busy}_{\text{RL}}(p - x - 1,\ \tau + x + 1)$ (axiom 2c)

$\equiv \text{busy}_{\text{RL}}(p - (x + 1),\ \tau + (x + 1))$

Therefore $\text{talk}\ (p,\ \tau) \Rightarrow \forall p' \leq p{:}\text{busy}_{\text{RL}}(p',\tau + p - p')$

Proof of Theorem 2-a): The theorem follows from Lemmas 1 and 2, and axiom 2e.

Proof of Theorem 2-b): The theorem follows from Lemmas 3 and 4, and axiom 2e.

Proof of Condition for Collision-Freedom: There is a collision at location p on the bus at time t if one of the following two conditions is satisfied:

1) Two signals coming from two different stations meet at a location p between the stations.
2) A signal from a different station arrives at station p, at the same time as station p is transmitting.

Formally, a collision is defined as:

$$\begin{aligned}\text{collision } (p, t) \equiv &\ \big(\text{busy}_{\text{LR}}(p-1, t-1) \text{ and busy}_{\text{RL}}(p+1, t-1)\big)\\ &\text{or } \big(\text{talk } (p, t) \text{ and } \big(\text{busy}_{\text{LR}}(p-1, t-1) \text{ or busy}_{\text{RL}}(p+1, t-1)\big)\big)\end{aligned} \tag{i}$$

The condition for collision-freedom is:

$$\forall p', \forall p'', \forall t', \forall t'': \text{talk } (p', t') \text{ and talk } (p'', t'') \Rightarrow (p' = p'') \text{ or } \big(|t' - t''| > |p' - p''|\big) \tag{ii}$$

We now prove that (ii) is a necessary and sufficient condition for collision-freedom, i.e.,

$$\begin{aligned}&\big[\forall p', \forall p'', \forall t', \forall t'': \text{talk } (p', t') \text{ and talk } (p'', t'') \Rightarrow (p' = p'') \text{ or } \big(|t' - t''| > |p' - p''|\big)\big]\\ &\equiv \forall p, \forall t: \text{not collision } (p, t)\end{aligned}$$

Proof:

Part 1: Prove that if a collision occurs, then (ii) is false.

Case 1: Collision of the type:

$$\begin{aligned}&\exists p, \exists t{:}\text{busy}_{\text{LR}}(p-1, t-1) \text{ and busy}_{\text{RL}}(p+1, t-1)\\ &\Rightarrow \exists p' < p{:}\text{talk } (p', t-1-(p-1-p')) \text{ and } \exists p'' > p{:} \text{ talk } (p'', t-1-(p''-(p+1))) \qquad \text{(Lemmas 1 \& 2)}\\ &\equiv \exists p' < p{:}\text{talk } (p', t-(p-p')) \text{ and } \exists p'' > p{:} \text{ talk } (p'', t-(p''-p))\end{aligned}$$

Consider t' and t'' such that

$$\begin{aligned}&t' = t - p + p'\\ \text{and } &t'' = t + p - p''\end{aligned}$$

Then, $t' - t'' = p'' + p' - 2p$

$$\Rightarrow |t' - t''| \le |p'' - p'| \text{ (since } p' \le p \le p'')$$

Therefore, $\exists p', \exists p'', \exists t', \exists t''$:

$$\text{talk } (p',t') \text{ and talk } (p'',t'') \text{ and } (p' \ne p'') \text{ and } |t' - t''| \le |p' - p''|$$

Case 2: Collision of the type:

$$\begin{aligned}&\text{talk } (p',t') \text{ and busy}_{\text{LR}}(p'-1, t'-1)\\ &\Rightarrow \text{talk } (p', t') \text{ and } \exists p'' < p'{:} \text{ talk } (p'', t'-1-(p'-1-p'')) \qquad \text{(Lemma 1)}\end{aligned}$$

Consider $t'' = t' - 1 - (p' - 1 - p'')$

$$\begin{aligned}&= p'' - p' + t'\\ &\Rightarrow t'' - t' = p'' - p'\\ &\Rightarrow |t'' - t'| \le |p'' - p'|\end{aligned}$$

Therefore, $\exists p', \exists p'', \exists t', \exists t''$:

$$\text{talk } (p',t') \text{ and talk } (p'',t'') \text{ and } (p' \ne p'') \text{ and } |t' - t''| \le |p' - p''|$$

Case 3: Collision of the type:

$$\begin{aligned}&\text{talk } (p', t') \text{ and busy}_{\text{RL}}(p'+1, t'-1)\\ &\Rightarrow \text{talk } (p', t') \text{ and } \exists p'' > p'{:}\text{talk } (p'', t'-1-(p''-(p'+1))) \qquad \text{(Lemma 2)}\end{aligned}$$

Consider $t'' = t' - 1 - (p'' - (p'+1))$
Then, $t'' - t' = p' - p''$

$$\Rightarrow |t' - t''| \le |p' - p''|$$

Therefore, $\exists p', \exists p'', \exists t', \exists t''$:

$$\text{talk } (p',t') \text{ and talk } (p'', t'') \text{ and } (p' \ne p'') \text{ and } |t' - t''| \le |p' - p''|$$

□

Part 2: Prove that if the condition (ii) is false, a collision will occur.

Case 1: talk (p', t') and talk (p'', t'') and $|p' - p''| > |t' - t''|$ and $(p' \neq p'')$
Without loss of generality, we assume : $p' < p''$
Consider p and t such that:

$$p = (p' + p'' + t'' - t')/2$$
$$t = (p'' - p' + t' + t'')/2$$

It can be checked that:

$$p' \leq p \leq p''; \quad t > t'; \quad t > t''$$

As long as each message is longer than one unit length, appropriate values for t' and t'' can be chosen which will result in integer values for the above expressions.

talk $(p', t') \Rightarrow \text{busy}_{\text{LR}}(p - 1, t' + p - 1 - p')$ (Lemma 3)
$\equiv \text{busy}_{\text{LR}}(p - 1, t' + p' - 1 + (p' + p'' + t'' - t')/2)$
$\equiv \text{busy}_{\text{LR}}(p - 1, (p'' - p' + t'' + t')/2 - 1)$

talk $(p'', t'') \Rightarrow \text{busy}_{\text{RL}}(p + 1, t'' + p'' - (p + 1))$ (Lemma 4)
$\equiv \text{busy}_{\text{RL}}(p + 1, t'' + p'' - (p' + p'' + t'' - t')/2 - 1)$
$\equiv \text{busy}_{\text{RL}}(p + 1, (p'' - p' + t'' + t')/2 - 1)$
With $t = (p'' - p' + t'' + t')/2$, the following condition for a collision is true:
$\text{busy}_{\text{LR}}(p - 1, t - 1)$ and $\text{busy}_{\text{RL}}(p+1, t - 1)$

Case 2: talk (p',t') and talk (p'',t'') and $|t' - t''| = |p'' - p'|$ and $(p' \neq p'')$
a) $t' - t'' = p'' - p'$
talk $(p'',t'') \Rightarrow \forall p \leq p''{:}\text{busy}_{\text{RL}}(p+1, t''+p'' - p - 1)$ (Lemma 4)
Therefore, talk (p',t') and talk (p'',t'')
$\Rightarrow$ talk (p',t') and $\forall p \leq p''{:}\text{busy}_{\text{RL}}(p+1, t'' + p'' - p - 1)$
With $p = p'$, the following condition for a collision is satisfied:
talk (p',t') and $\text{busy}_{\text{RL}}(p'+1, t' - 1)$
b) $t'' - t' = p'' - p'$
talk $(p',t') \Rightarrow \forall p \geq p'{:}\text{busy}_{\text{LR}}(p - 1, t'+p - p' - 1)$ (Lemma 3)
Therefore, talk (p',t') and talk (p'',t'')
$\Rightarrow \forall p \geq p'{:}\text{busy}_{\text{LR}}(p - 1, t'+ p - p' - 1)$ and talk (p'',t'')
With $p = p''$, the following condition for a collision is satisfied:
talk (p'',t'') and $\text{busy}_{\text{LR}}(p'' - 1, t'' - 1)$ □

Lemmas 5 and 6 will be used in the verification of the EHS protocol.

Lemma 5: The following relation holds for the EHS protocol:

not clear (p, τ) and not talk $(p,\tau) \Rightarrow$ not talk $(p,\tau + 1)$

Proof of Lemma 5: The following assertion has been proved to be invariant for the EHS protocol:

talk $(p',\tau') \Rightarrow \exists\tau'' < \tau'$:[clear (p',τ'') and $\forall t : \tau'' < t \leq \tau'$:talk (p',t)] (A2)
$\equiv$ (clear $(p', \tau' - 1)$ and talk (p',τ'))
or ($\exists\tau'' < \tau' - 1$: [clear (p',τ'') and $\forall t : \tau'' < t \leq \tau' - 1$:talk (p',t)])
$\Rightarrow$ clear $(p',\tau' - 1)$ or talk $(p';\tau' - 1)$

Therefore, not talk (p',τ') and not clear $(p',\tau') \Rightarrow$ not talk $(p', \tau' + 1)$

Lemma 6: The following relation holds for the EHS protocol:

[$\forall\tau, \tau_1 \leq \tau \leq \tau_2$: not clear (p,τ)] and not talk (p, τ_1)
$\Rightarrow \forall\tau', \tau_1 \leq \tau' \leq \tau_2$: not talk $(p,\tau' + 1)$

Proof of Lemma 6: By induction on τ'.
Base case:

$\tau' = \tau_1$
[$\forall\tau, \tau_1 \leq \tau \leq \tau_2$: not clear (p,τ)] and not talk (p,τ_1)
$\Rightarrow$ not talk $(p, \tau_1 + 1)$ (Lemma 5)

Induction Hypothesis:

Assume that it holds for $\tau' = \tau''$, $\tau_1 \le \tau'' < \tau_2$

$$[\ \forall\tau,\ \tau_1 \le \tau \le \tau_2\text{: not clear } (p,\tau)\ ] \text{ and not talk } (p,\tau_1) \Rightarrow \text{not talk } (p,\ \tau'' + 1)$$
$$\Rightarrow [\ \forall\tau,\ \tau_1 \le \tau \le \tau_2\text{: not clear } (p,\tau)\ ] \text{ and not talk } (p,\ \tau'' + 1) \Rightarrow \text{not talk } (p,\ \tau'' + 2) \qquad \text{(Lemma 5)}$$

Therefore,

$$[\ \forall\tau,\ \tau_1 \le \tau \le \tau_2\text{: not clear } (p,\tau)\ ] \text{ and not talk } (p,\tau_1)$$
$$\Rightarrow \forall\tau_1,\ \tau' \le \tau' \le \tau_2\text{: not talk } (p,\tau' + 1)$$

{ Also,

$$[\ \forall\tau,\ \tau_1 \le \tau \le \tau_2\text{: not clear } (p,\tau)\ ] \text{ and not talk } (p,\tau_1)$$
$$\Rightarrow \forall\tau',\ \tau_1 \le \tau' \le \tau_2\text{: not talk } (p,\tau') \text{ (since talk } (p,\tau_1) = \text{false) }\}$$

Proof of the EHS Protocol: We show that the EHS protocol is collision-free.

The condition for collision-freedom is:

$$\forall u',\ \forall u'',\ \forall v',\ \forall v''\text{: talk } (u',v') \text{ and talk } (u'',v'') \Rightarrow (u' = u'') \text{ or } (|v' - v''| > \text{bdelay}_{u'u''})$$

Proof: The following assertion is invariant for the program as proved earlier, and will be used together with Theorems 1 and 2 to show that the EHS protocol satisfies the colision-freedom property.

$$\text{talk } (p',\tau') \Rightarrow \exists\tau'' < \tau' \text{ :[clear } (p',\tau'') \text{ and } \forall\tau_1,\ \tau'' < \tau_1 \le \tau' \text{ :talk } (p',\tau_1)] \qquad \text{(A2)}$$

Without any loss of generality, assume $p' < p''$.

Splitting (A2) into two parts (one corresponding to the predicate **clear**, and the other to the universally quantified predicate **talk**):

1) $\text{talk } (p',\tau') \Rightarrow \text{clear } (p',\tau'')$
2) $\text{talk } (p',\tau') \Rightarrow \forall\tau_1,\ \tau'' < \tau_1 \le \tau' \text{ :talk } (p',\tau_1)$

(Note: Both 1 and 2 refer to the same τ' and τ'')

From 1: not on (p',τ'') and not busy (p',τ'') and $\forall\tau_2\text{: } 0 \le \tau_2 \le D + R\ (p') \text{ :signal } (p',\tau'' - \tau_2)$
(definition of clear)

$$\text{(i) talk } (p',\ \tau') \Rightarrow \text{not busy } (p',\ \tau'')$$
$$\Rightarrow \text{not talk } (p'',\ \tau'' - \text{bdelay}_{p'p''}) \qquad \text{(a)}$$
(Theorem 2)

$$\text{(ii) talk } (p',\ \tau') \Rightarrow \forall\tau_2,\ 0 \le \tau_2 \le D + R\ (p') \text{ :signal } (p',\tau'' - \tau_2)$$
$$\Rightarrow \forall\tau_2,\ 0 \le \tau_2 \le D + R\ (p') \text{ : on } (p'',\tau'' - \tau_2 + \text{wdelay}_{p'p''}) \qquad \text{(Theorem 1b)}$$
$$\equiv \forall\tau_3,\ \tau'' + \text{wdelay}_{p'p''} - (D + R\ (p')) \le \tau_3 \le \tau'' + \text{wdelay}_{p'p''} \text{ :on } (p'',\tau_3)$$
$$\Rightarrow \forall\tau_3,\ \tau'' + \text{wdelay}_{p'p''} - (D + R\ (p')) \le \tau_3 \le \tau'' + \text{wdelay}_{p'p''} \text{ : not clear } (p'',\tau_3)$$
(b)
(definition of clear)

From 2: $\forall\tau_1,\ \tau'' < \tau_1 \le \tau' \text{ :talk } (p',\ \tau_1)$

$$\Rightarrow \forall\tau_1,\ \tau'' < \tau_1 \le \tau' \text{ :busy } (p'',\ \tau_1 + \text{bdelay}_{p'p''}) \qquad \text{(Theorem 2a)}$$
$$\equiv \forall\tau_4,\ \tau'' + \text{bdelay}_{p'p''} < \tau_4 \le \tau' + \text{bdelay}_{p'p''} \text{ :busy } (p'',\tau_4)$$
$$\Rightarrow \forall\tau_4,\ \tau'' + \text{bdelay}_{p'p''} < \tau_4 \le \tau' + \text{bdelay}_{p'p''} \text{ :not clear } (p'',\tau_4) \qquad \text{(c)}$$
(definition of clear)

Combining (b) and (c):

$$\text{talk } (p',\tau') \Rightarrow \forall\tau_5 \text{ :[} \tau'' + \text{wdelay}_{p'p''} - (D + R\ (p')) \le \tau_5 \le \tau' + \text{bdelay}_{p'p''} \text{ :not clear } (p'',\ \tau_5)\ ]$$
(because $\text{wdelay}_{p'p''} > \text{bdelay}_{p'p''}$)
$$\Rightarrow \forall\tau_5,\ \tau'' - \text{bdelay}_{p'p''} \le \tau_5 \le \tau' + \text{bdelay}_{p'p''}\text{:not clear } (p'',\tau_5)$$
(since $D \ge \text{bdelay}_{p'p''}$ and $R\ (p') \ge \text{wdelay}_{p'p''}$)
$$\Rightarrow \forall\tau_5,\ \tau'' - \text{bdelay}_{p'p''} \le \tau_5 \le \tau' + \text{bdelay}_{p'p''} \text{ :not talk } (p'',\tau_5) \qquad \text{(Lemma 6 \& (a))}$$
$$\Rightarrow \forall\tau_5,\ \tau' - \text{bdelay}_{p'p''} \le \tau_5 \le \tau' + \text{bdelay}_{p'p''} \text{ :not talk } (p'',\tau_5) \qquad \text{(since } \tau' > \tau'')$$
$$\Rightarrow \forall\tau_5,\ |\ \tau' - \tau_5\ | \le \text{bdelay}_{p'p''} \text{ :not talk } (p'',\tau_5)$$

Therefore, talk $(p',\ \tau')$ and talk $(p'',\ \tau'') \Rightarrow (p' = p'')$ or $(|\tau' - \tau''| > \text{bdelay}_{p'p''})$ □

REFERENCES

[1] B. Berthomieu and M. Menasche, "An enumerative approach for analysing timed Petri nets," in *Proc. Inform. Processing '83*, IFIP, 1983.

[2] G. V. Bochmann, "Finite state decomposition of communication protocols," *Comput. Networks*, vol. 2, pp. 361–372, Oct. 1978.

[3] B. L. Divito, "Verification of communication protocols and abstract process models," Ph.D. dissertation, Dep. Comput. Sci., Univ. Texas at Austin, Aug. 1982.

[4] K. Eswaran, V. C. Hamacher, and G. S. Shedler, "Collission-free access control for communication bus networks," *IEEE Trans. Software Eng.*, vol. SE-7, pp. 574–582, Nov. 1981.

[5] M. Fine and F. Tobagi, "Demand assignment multiple access schemes in broadcast bus local area networks," *IEEE Trans. Comput.*, vol. C-33, pp. 1130–1159, Dec. 1984.

[6] L. Fratta, "An improved access protocol for data communication bus networks with control wire," in *Proc. ACM SIGCOMM Symp.*, Austin, TX, Mar. 1983.

[7] M. Gouda, "Closed covers: To verify progress of communicating finite state machines," *IEEE Trans. Software Eng.*, vol. SE-10, pp. 846–855, Nov. 1984.

[8] B. Hailpern and S. Owicki, "Verifying network protocols using temporal logic," presented at the NBS Trends and Applications Symp., May 1980.

[9] C. A. R. Hoare, "An axiomatic basis for computer programming," *Commun. ACM.*, vol. 12, pp. 571–580, 1969.

[10] S. S. Lam and A. U. Shankar, "Protocol verification via projections," *IEEE Trans. Software Eng.*, vol. SE-10, pp. 325–342, July 1984.

[11] R. D. Metcalfe and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks," *Commun. ACM*, vol. 19, July 1976.

[12] J. Misra and K. M. Chandy, "Proofs of networks of processes," *IEEE Trans. Software Eng.*, vol. SE-7, pp. 471–426, July 1981.

[13] S. Owicki and D. Gries, "An axiomatic proof technique for parallel programs," *Acta Inform.*, vol. 6, pp. 319–340, 1976.

[14] J. L. Peterson, *Petri Net Theory and the Modeling of Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1981.

[15] C. Ramchandani, "Analysis of asynchronous concurrent systems by timed Petri nets," Ph.D. dissertation, Massachusetts Inst. Technol., Project Mac Rep. MAC-TR-120, 1974.

[16] R. R. Razouk and C. V. Phelps, "Performance analysis using timed Petri nets," Univ. California, Irvine, Tech, Rep. TR-77.

[17] A. U. Shankar and S. S. Lam, "An HDLC protocol specification and its verification using image protocols," *ACM Trans. Comput. Syst.*, vol. 1, no. 4, pp. 321–368, Nov. 1983.

[18] ——, "Time-dependent distributed systems: Proving safety, liveness and real-time properties," Univ. Texas as Austin, Tech. Rep. TR-85-24; revised Oct. 1986, to appear in *Distributed Computing*.

[19] F. Tobagi, F. Borgonovo, and L. Fratta, "Expressnet: A high-performance integrated-services local area network," *IEEE J. Select. Areas Commun.*, vol. SAC-1, pp. 898–912, Nov. 1983.

[20] C. Tseng and B. Chen, "D-net: A new scheme for high data-rate optical local area networks," *IEEE J. Select. Areas Commun.*, vol. SAC-1, pp. 493–499, Nov. 1983.

[21] P. Zafiropoulo *et al.*, "Towards analysing and synthesising protocols," *IEEE Trans. Commun.*, vol. COM-28, pp. 655–660, Apr. 1980.

[22] W. M. Zuberek, "Timed Petri net and preliminary performance evaluation," presented at the 7th Annual Symp. Computer Architecture, 1980.

Pradeep Jain received the Bachelor's degree in mechanical engineering and the Master's degree in computer science from the Indian Institute of Technology, Kanpur, India, in 1979 and 1982, respectively.

He is currently a Ph.D. candidate in computer science at the University of Texas at Austin. He is doing research in the area of communication protocols.

Simon S. Lam (S'69–M'74–SM'80–F'85) received the B.S.E.E. degree (with Distinction) from Washington State University in 1969, and the M.S. and Ph.D. degrees in engineering from the University of California at Los Angeles in 1970 and 1974, respectively.

From 1974 to 1977, he was a research staff member at the IBM Thomas J. Watson Research Center, Yorktown Heights, NY. Since September 1977, he has been on the faculty of the University of Texas at Austin, where he is a Professor of Computer Sciences and holds the Second David Bruton Jr. Centennial Professorship. His research interests are in the areas of computer networks, communication protocols, performance modeling, and the specification and verification of distributed systems.

Dr. Lam received the 1975 Leonard G. Abraham Prize Paper Award from the IEEE Communications Society. He currently serves on the editorial boards of three journals, IEEE TRANSACTIONS ON COMMUNICATIONS, *Performance Evaluation*, and *Proceedings of the IEEE*.