COMPUTER AIDED DESIGN OF ETHERNET LOCAL AREA NETWORKS

by

Gilbert Pui Sun TANG

A thesis
presented to the University of Manitoba
in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE (ELECTRICAL ENGINEERING)
in

The Department of Electrical Engineering
The Faculty of Engineering
THE UNIVERSITY OF MANITOBA

Winnipeg, Manitoba
August 7, 1987

ISBN 0-315-37387-3

COMPUTER AIDED DESIGN OF ETHERNET LOCAL AREA NETWORKS

BY

GILBERT PUI SUN TANG

A thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

ABSTRACT

A computer simulation of Ethernet was developed to assist network designers in the planning of Local Area Networks. With the use of the simulation, the efficiency of a specified network can be evaluated before the actual construction and installation of expensive physical components.

The computer program implemented the IEEE 802.3 CSMA/CD access protocol to simulate network events. A variety of devices including bridges, repeaters, and terminal servers are available in the design process. The computer program also allows the designer to specify the characteristics of each device such as the unique bandwidth limitation, the packet size, the number of packets per message, and any requirement for acknowledgment.

The simulation was tested with actual experimental data and proved to be reasonably accurate. A special user interface was developed which enabled non-programmer to easily use the simulation. Further testing in a large Ethernet environment remains to be done. The simulation was used to test a number of different designs for the Ethernet Local Area Network at the Health Sciences Centre. The results showed that the present network has excessive capacity to easily support all forseeable future configuration.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude for those people with whom I have had the pleasure of working on a daily basis in the Department of Information Services, Winnipeg Health Sciences Centre. All of these people have assisted me in some way, either directly or indirectly and my deep appreciation is extended to them. The people to whom I am particularly indebted are:

Dr. Rick Fenton, my supervisor, without whose intellectual guidance, patience, and tolerance, I would not have been able to learn and grow as an individual. Thank you for your confidence in me as a person and for your invaluable recommendations in my personal struggle of life.

Ms. Maria Larsen, for the concerns that were expressed on my behalf.

Mr. Andy Drobot, for his invaluable suggestions in this reseach.

Sincere thanks are also due to: Ms. Shannon Handfield, Mr. Robert Poirier, Mr. Dan Little, and Ms. Betty-Ann, whose presence in the department made for a more enjoyable workplace.

Dr. C.M. Laucht and Dr. M. Pawlak, my external advisors, for their suggestions in this reseach.

Miss Vilplene Cheng, for her supports, understanding and love.

My mother, whose patience, encouragement, and love has helped me attain my goals.

Also, the financial assistance from the Health Sciences Centre was highly appreciated.

I hereby declare that I am the sole author of this thesis.

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Gilbert Pui Sun, Tang

I further authorize the Univeristy of Manitoba to reproduce this thesis, in whole or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Gilbert Pui Sun, Tang

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

The Health Sciences Centre located in Winnipeg, Canada, is the largest hospital in the Province of Manitoba. The Centre is divided into 20 distinct buildings with a total area over 260,000 square meters. The Information Services Department of the Health Science Centre began to develop an Integrated Hospital Information System in 1983, with the objective of providing an efficient and cost-effective means to exchange patient, clinical and administrative information.

The Integrated Hospital Information System is a computerized information management system which consists of hardware and software components. A major hardware component is an advanced local area network (LAN). The LAN is an Ethernet which provides a data communication network linking all the computers and peripherals together within the Health Sciences Centre. The inter-connection between the devices provides a fast and efficient pathway for data processing operations.

1.2 DEFINITION

Vendors of Ethernet devices indicate that it is capable of operating efficiently at the specified data rate of 10 Megabits per second (Mbps). However, a number of reports in the literature have indicated that the bandwidth can become saturated if there is moderate activity (Shoch et al, 1982). A major issue of concern for the communication engineers at the Health Sciences Centre is to be able to estimate the performance of various network expansions preceeding the installation and purchase of expensive devices. This is especially true at this time since the network is only 25 percent complete.

1.3 OBJECTIVE

The objective of this research was to develop a computer aided design tool to estimate the performance of an arbitrarily specified Ethernet network. In this way, a cost-effective and efficient network can be designed and analyzed before the installation and purchase of network devices.

1.4 SCOPE

The entire thesis is divided into eight chapters. Chapter 2 describes the basic concept of Ethernet. Chapter 3 reviews the recent literature on Ethernet. Chapter 4 describes the computer aided design tool developed in this research. Chapter 5 is the validation of the software, with a specific example of its application given in Chapter 6. A general discussion is provided in Chapter 7. Chapter 8 is the available references.

CHAPTER 2

ETHERNET (CSMA/CD PROTOCOL)

Carrier Sense Multiple Access with Collision Detection (CSMA/CD) is a media access method in data communication. This method is used to control the use of a shared transmission medium. Due to commercial interest, CSMA/CD protocol was commercialized to become Ethernet. This chapter is divided into three sections which describe Ethernet or the CSMA/CD system. First, the background of Ethernet is mentioned. Then, the definition of an Ethernet is presented. Finally, the CSMA/CD media access method is described. A concise reference to Ethernet specifications is available in Appendix A; more detail may be found in IEEE CSMA/CD (1985).

2.1 BACKGROUND

The Xerox Corporation developed the first CSMA/CD system at the Palo Alto Research Centre in 1975. It was called Experimental Ethernet (Boggs and Metcalfe, 1976). Due to the cooperative efforts of Xerox, Intel, and Digital Equipment Corporation, the Ethernet Specification was developed in 1980. The most current version of the Ethernet Specification is version 2.0 (Digital et al, 1982). Appendix B summarizes the differences between the original Experimental Ethernet and the current Ethernet Specification.

The Institute of Electrical and Electronics Engineers (IEEE) 802 committee was formed in 1980 to establish a common set of standards for media access technologies (IEEE CSMA/CD, 1985). In 1983, the IEEE

Standards Board approved the CSMA/CD system as IEEE Standard 802.3. In this thesis, the term CSMA/CD is the general term for this access method, while the term Ethernet refers to the commercialized CSMA/CD system based upon the slight modification of IEEE Standard 802.3.

IEEE 802 standards for Local Area Network are closely related to the International Organization for Standardization (ISO) Open System Interconnection (OSI) Reference Model. The OSI Reference Model is a logical structure for data communication network operations. Layering is the basic structure used in the OSI Reference Model. Each layer consists of a set of rules or protocols for data communication. The OSI Reference Model is divided into seven layers: 1) Application, 2) Presentation, 3) Session, 4) Transport, 5) Network, 6) Data Link, and 7) Physical. Figure 2.1 shows the layers for the OSI Reference Model.

Tanenbaum (1981) described the detailed functions for each layer in the OSI Reference Model. The upper layers (layer 1 to layer 5) are used for addressing, data transportation, error detection, error recovery, and user applications. The upper layers are outside of the scope in this research. The media access techniques are covered by the lower two layers, Data Link and Physical.

IEEE Standard 802 access methods specify both the interface and protocol for the Physical and Data Link Layers which defined by the OSI Reference Model. Figure 2.2 shows the relationship between the

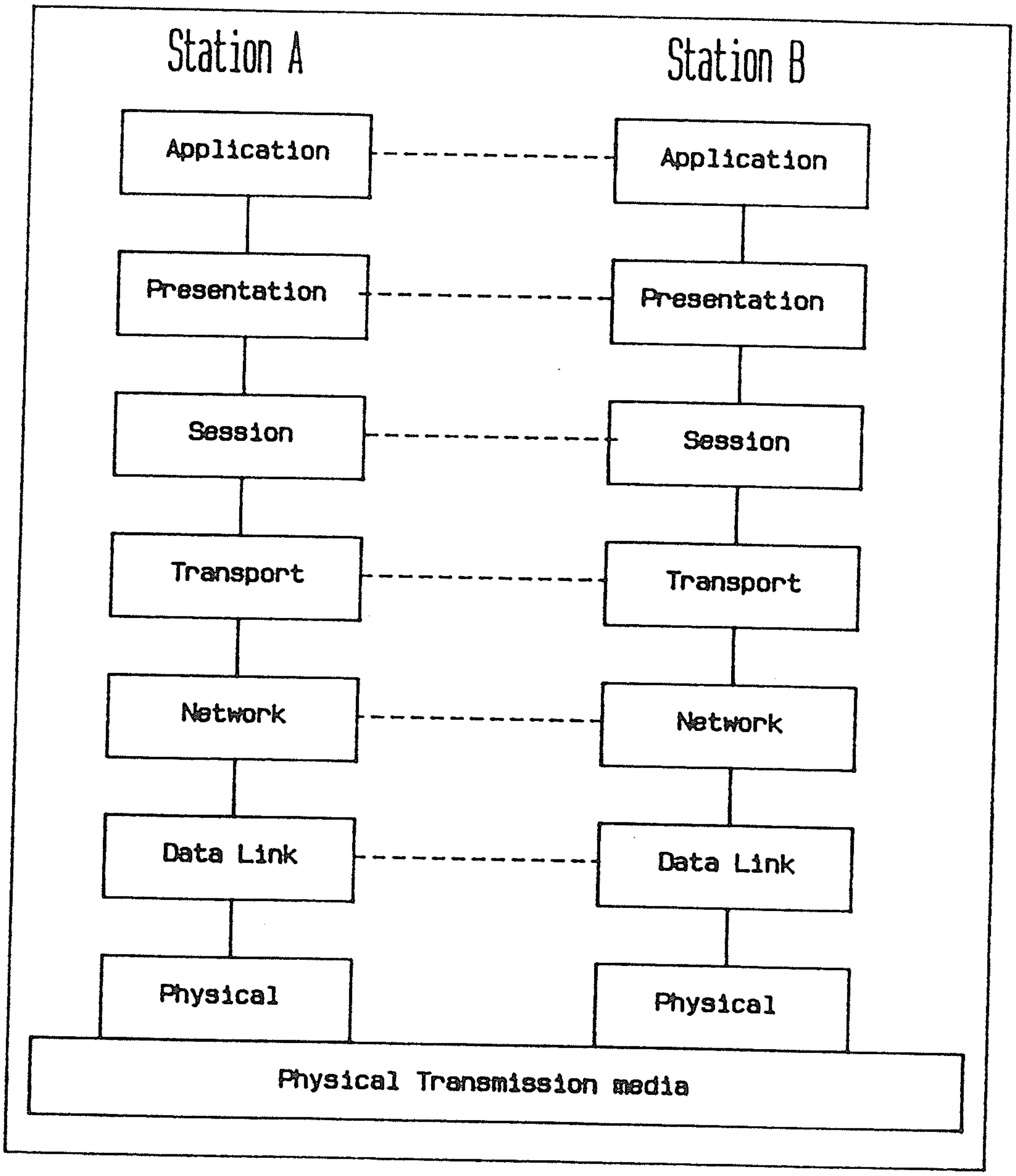


Figure 2.1 Open Systems Interconnection (OSI) Reference Model.

OSI Reference Model and IEEE Standard 802 access methods. Only 802.3 is considered in this report.

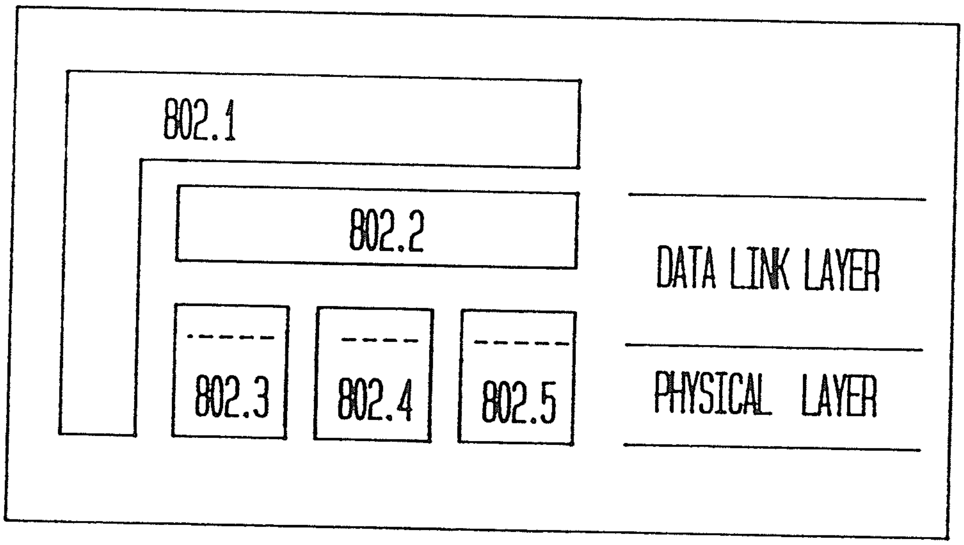


Figure 2.2 The relationship between OSI Reference model and IEEE Standard 802.
(from IEEE CSMA/CD, 1985)

2.2 DEFINITION OF AN ETHERNET NETWORK

An Ethernet network is a baseband LAN which interconnects two or more stations by physical transmission media. The media access method for Ethernet network is the CSMA/CD technique as described in the next section. Figure 2.3 illustrates a simple configuration of an Ethernet network. Ethernet operates at a transmission rate of 10 Megabits per second (Mbps) and uses a bus topology in which a set of stations are physically connected to a common length of coaxial cable through an interface unit. A station is a computer device which is capable of processing, transmitting and receiving network information. Each station on the Ethernet network is called a node. Each node is assigned an unique address to specify its location in the network. Xerox Corporation manages the allocation and distribution of unique addresses to Ethernet vendors throughout the world. These addresses consists of 6 hex digits providing a total of 2^{48} unique addresses.

2.2.1 Devices Within An Ethernet Network

An Ethernet network may consist of repeaters, nodes, transceivers, terminators, and transmission media. Figure 2.4 illustrates an example of an Ethernet network. This particular network is composed of five segments of transmission medium connected by four repeaters. Each node is physically attached to the medium

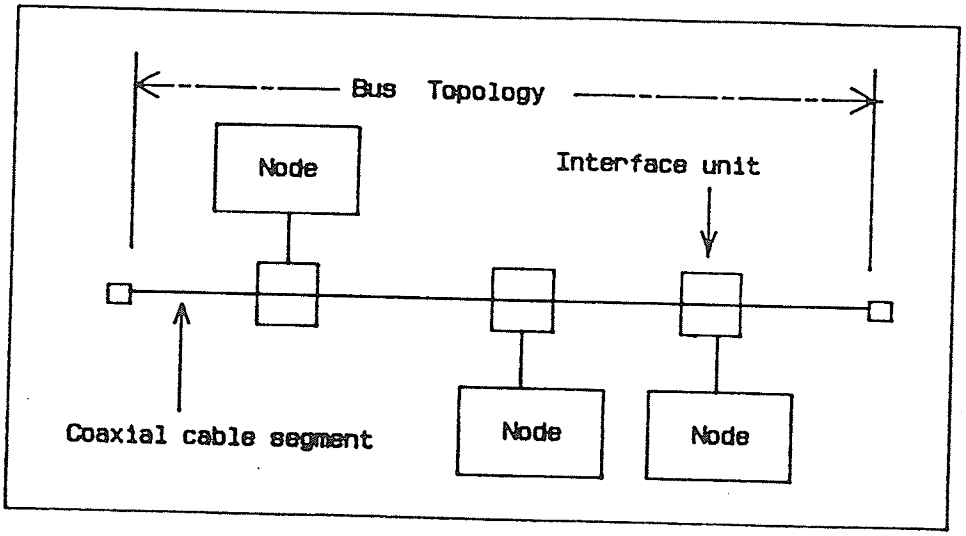


Figure 2.3 Simple Ethernet Configuration.

with interface units. Some devices such as terminals, may require servers in order to connected to the medium.

The commercially available transmission media for Ethernet are standard Ethernet coaxial cable, fiber optic cable, thinwire coaxial cable and transceiver cable. A unit of length for the cable is called a segment. The Ethernet coaxial segment is terminated at both ends with terminators. The transceiver cable provides a connection between a transceiver and a node. The transceiver is an interface device which is tapped physically on the coaxial cable to provide a functional interface between the node and the coaxial cable. Two segments of the coaxial cable can be joined by a Ethernet repeater.

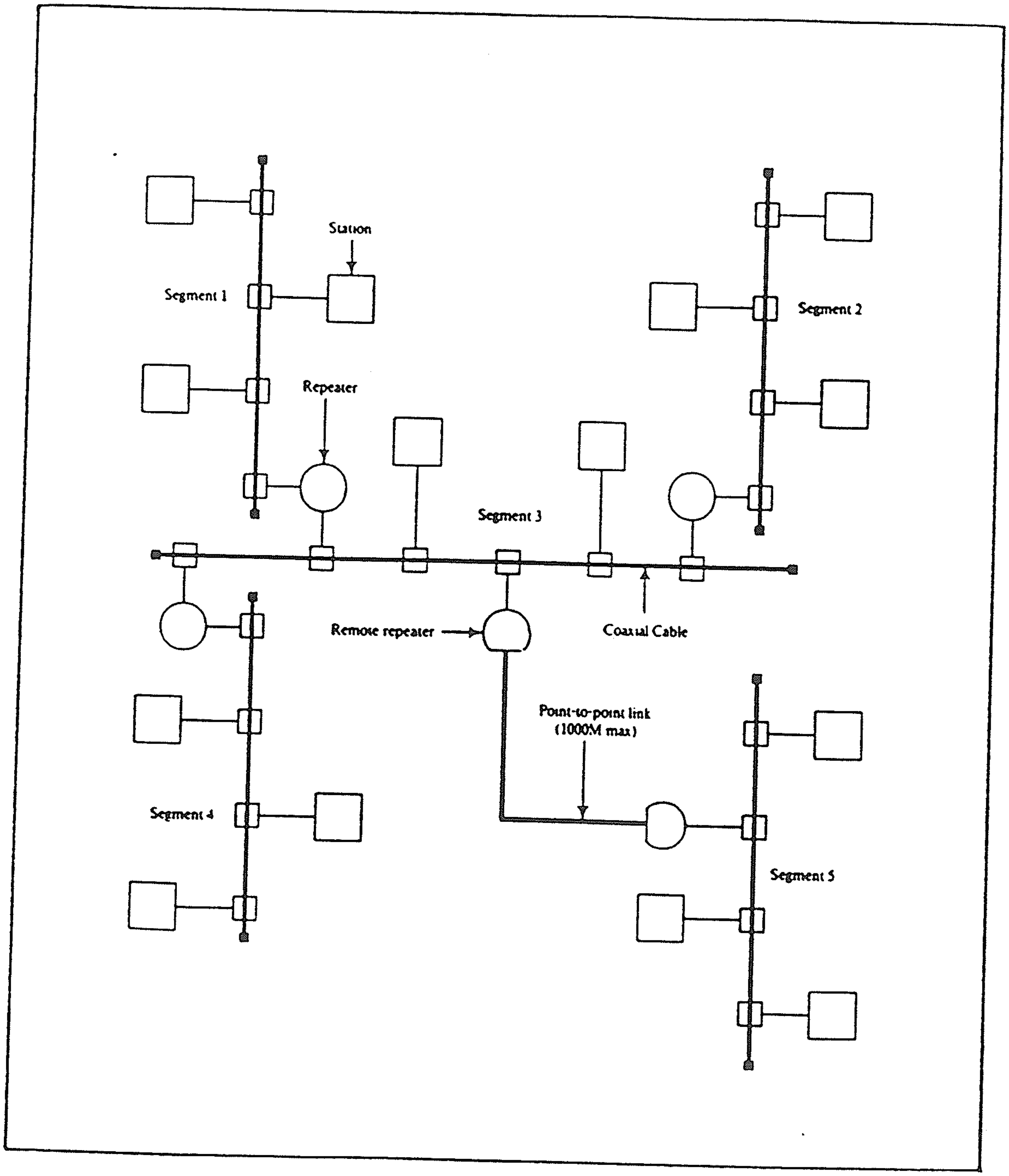


Figure 2.4 An example of an Ethernet Network.
(from Franta and Chlamtac, 1981)

Three types of repeaters are commercially available : the local repeater, the remote repeater, and the multiport repeater. The local repeater connects two Ethernet coaxial segments. The remote repeater consists of two local repeaters connected by a fiber optic link segment. The multiport repeater provides eight ports for connection of eight thinwire segments and one port for connection of the transceiver cable. Some of the commercial repeaters are described in Digital (1986). Figure 2.5 illustrates the relationship of the repeater to the OSI Reference Model. The repeater does not perform any data link functions, it only allows the bits to be transferred between the Physical Layers of the two nodes. A repeater can connect media with identical physical layers only.

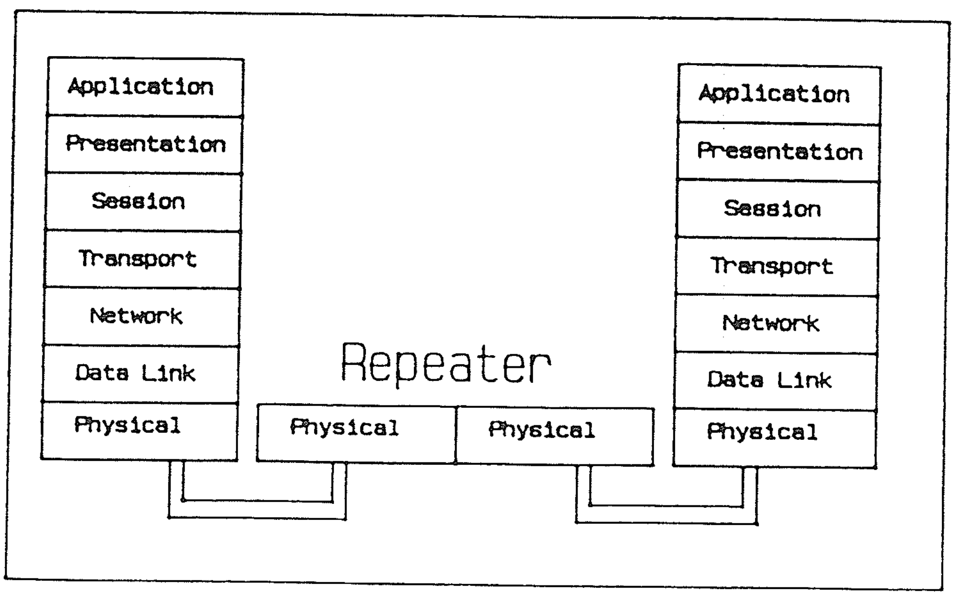


Figure 2.5 The relationship of the repeater to the OSI Reference model.

2.2.2 Limitations Of An Ethernet Network

Digital signals travel at approximately 0.77 times the speed of light on coaxial cable. Since signals degrade with distance on the transmission medium, vendors suggest that there are identifiable distance limitations for the Ethernet network (Digital, 1986). For example, the maximum lengths of an Ethernet coaxial segment, a thinwire coaxial segment, and a fiber optic link segment are 500 meters, 185 meters and 1000 meters respectively. The maximum length of a transceiver cable is 50 meters.

According to Digital (1986), Ethernet coaxial segments may contain a maximum of 100 transceivers with a minimum separation of 2.5 meters between transceivers; a thinwire segment may contain a maximum of 30 transceivers with a minimum separation of 0.5 meter. A standard Ethernet network has a distance limitation of 2800 meters between the furthest two nodes in the network. Figure 2.6 shows the distance limitation of an Ethernet network. This network consists of three Ethernet coaxial segments, one fiber optic link segment, two transceiver cables connected to the transmitting and receiving nodes, and four transceiver cables connected to two repeaters.

The distance limitation of Ethernet networks can be modified by using extended LAN devices to connect two or more Ethernet networks together.

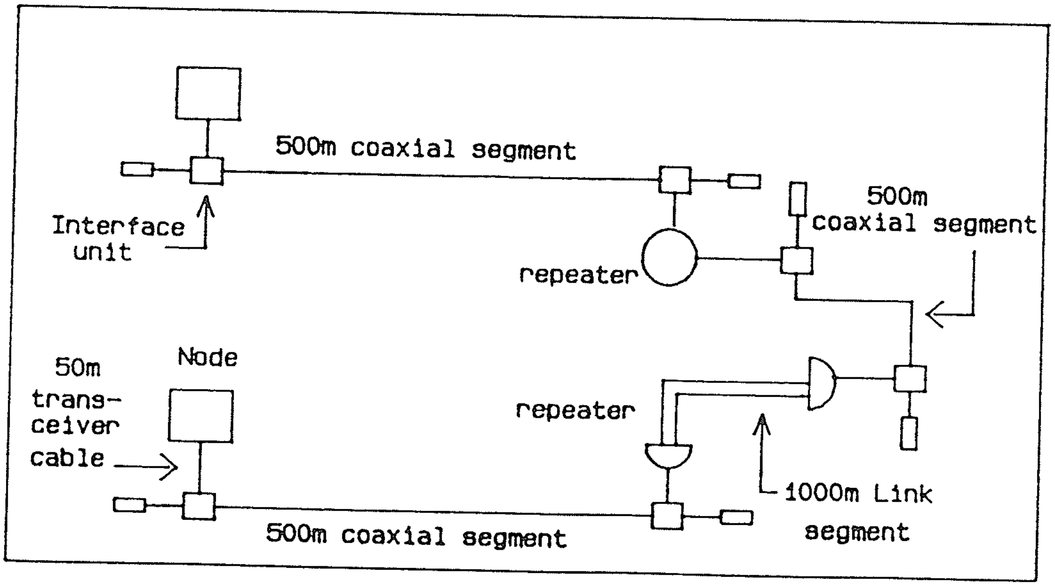


Figure 2.6 The distance limitation of an Ethernet network.

2.2.3 <u>Extended LAN Devices</u>

Two or more networks can be connected together with extended network devices. There are three types of devices to extend a local area network : bridge, router, and gateway. A bridge operates at the lower two layers (data link and physical) of the OSI Reference model. Figure 2.7 shows the relationship of the bridge to the OSI layers. Instead of bits transferred in the physical layer, a bridge allows information transferred between the data link layer of the two nodes. Therefore, a bridge can connect two local area networks with same or different type of physical layers. In addition, the bridge is able to recognize the address of the transmission and determine whether or not it should be buffered and forwarded from one network to the other. Digital (1986) recommends that an extended LAN consist of no more than seven bridges in series and span a distance not exceeding 22,400 meters.

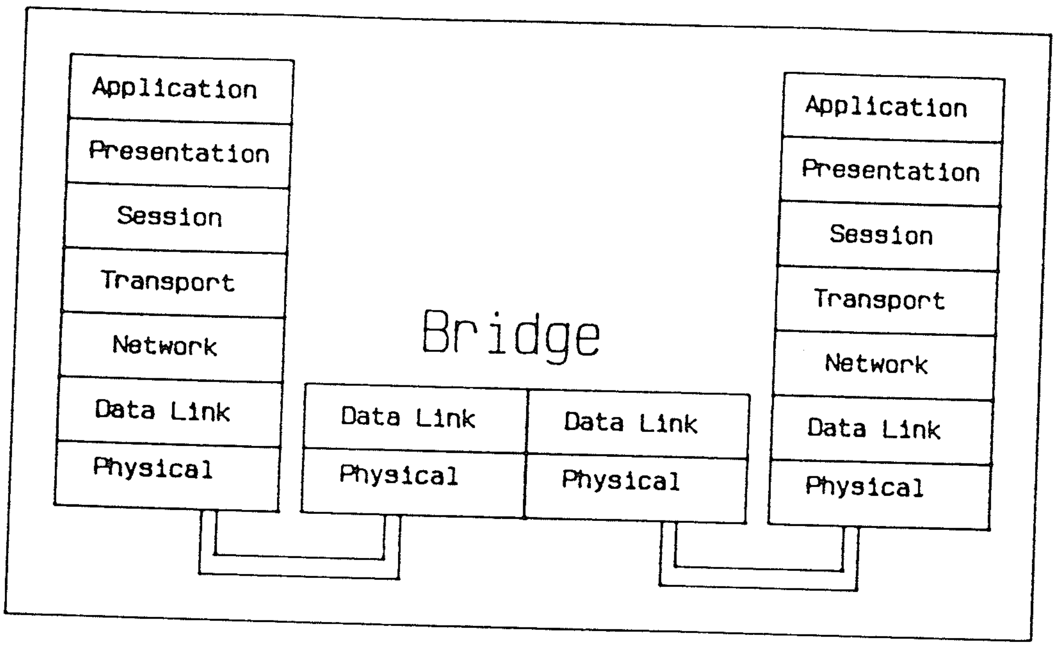


Figure 2.7 The relationship of the bridge to the OSI model.

A router operates within the lower three layers (Network, Data Link, Physical) of the OSI Reference model. The function of the router is to determine if a message is to be forwarded to the destination node in a remote network. Figure 2.8 shows the relationship of the router to the OSI Reference model.

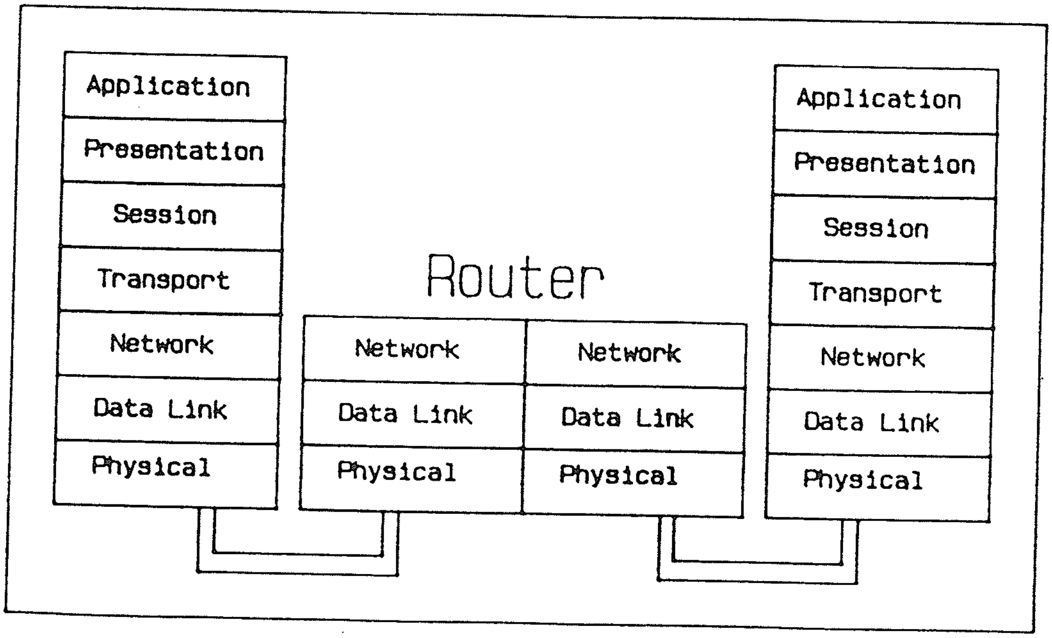


Figure 2.8 The relationship of the router to the OSI Reference model.

A gateway may operate in all layers of the OSI Reference model. It allows two networks with different but specified protocols to communicate with each other. Figure 2.9 shows the relationship of the gateway to the OSI Reference model. Detailed discussion of gateways and routers is external to scope of this thesis; for a detailed description refer to McNamara (1985), Digital (1982), and Digital (1986).

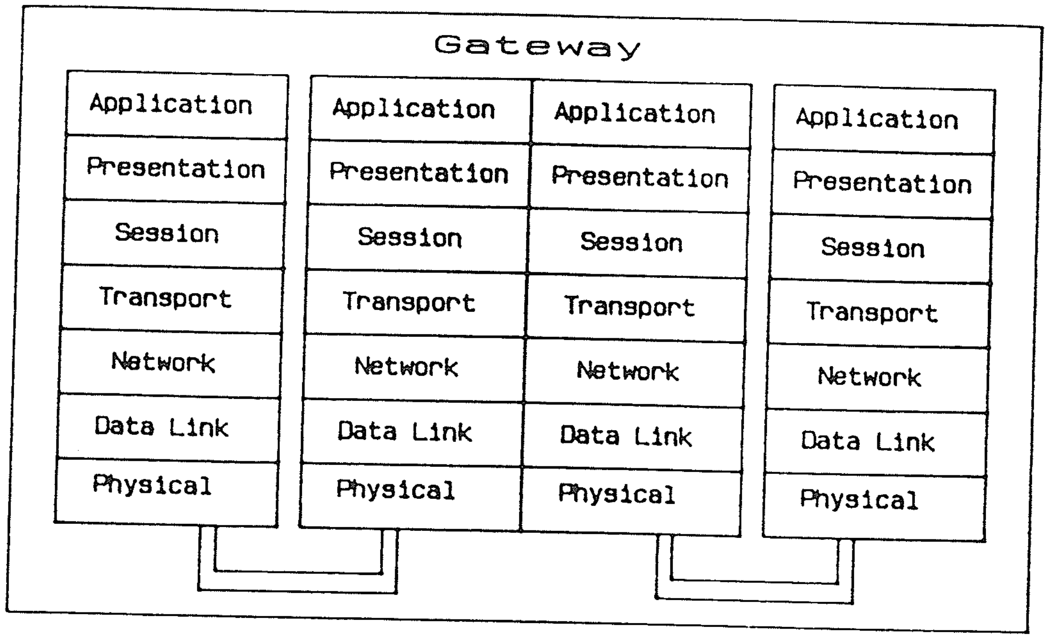


Figure 2.9 The relationship of the gateway to the OSI Reference model.

2.3 CSMA/CD MEDIA ACCESS METHOD

The three basic features of CSMA/CD are Carrier Sense, Multiple Access, and Collision Detection. Carrier sense (listen-before-talking) enables a node to detect activity or traffic on the transmission medium. If there is traffic , the node will defer, or wait. Multiple access allows all nodes to share a common bus transmission medium. Figure 2.10a shows Carrier Sense Multiple Access. Figure 2.10b shows the deferral due to Carrier Sense. Collision is defined as two or more simultaneously transmitted messages overlapping with each other on the transmission medium, producing interference. Collision detection (listen-while-talking) provides the ability to detect interference in the transmission. Figure 2.11a shows Collision Detection. A collision checking interval is the two way propagation delay required for the initial bits of a node's transmission to reach and return from the farthest node on the medium. Once a transmission is begun and successfully passes the collision window, no collision will occur because the carrier sense of other nodes will recognize the transmission in the medium. This is referred to as capturing the channel. If a collision occurs during the collision window, the transmitting nodes abandon their transmissions and remain silent for a random period of time (backoff) before attempting to retransmit. Figure 2.11b shows the backoff due to collision. In addition, each colliding node emits a jam signal (a short series of bits) to ensure that all other nodes have detected the collision (IEEE CSMA/CD, 1985 and Digital, 1982).

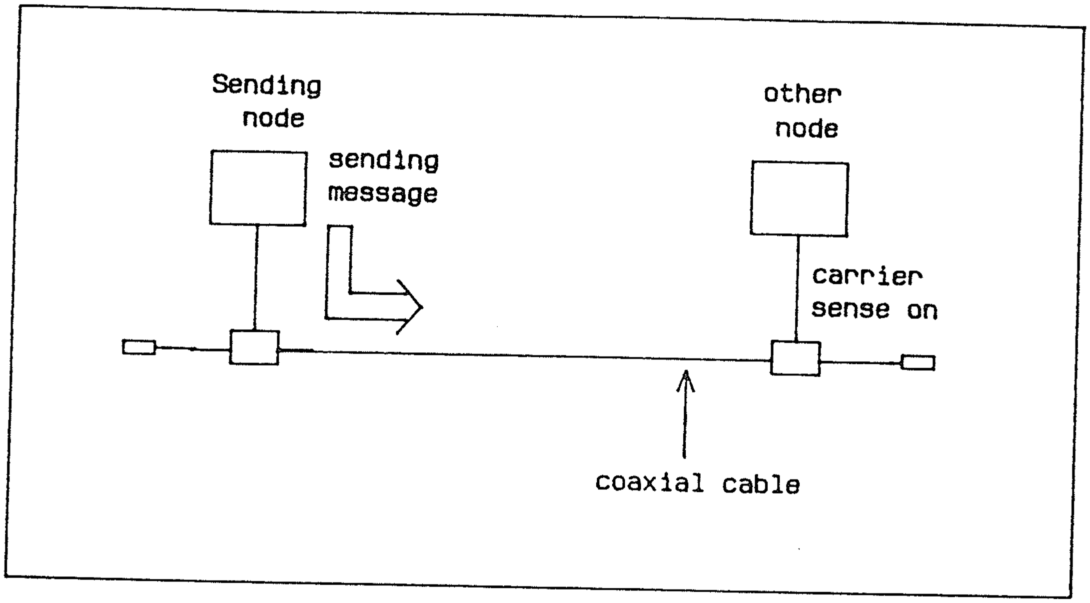


Figure 2.10a Carrier Sense Multiple Access.

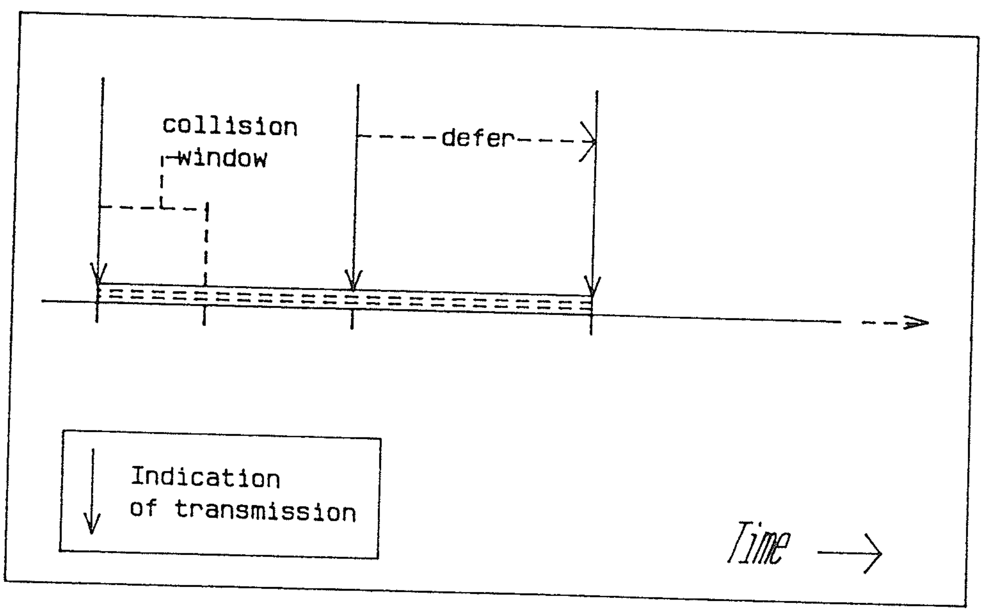


Figure 2.10b Carrier Sense and deferral.

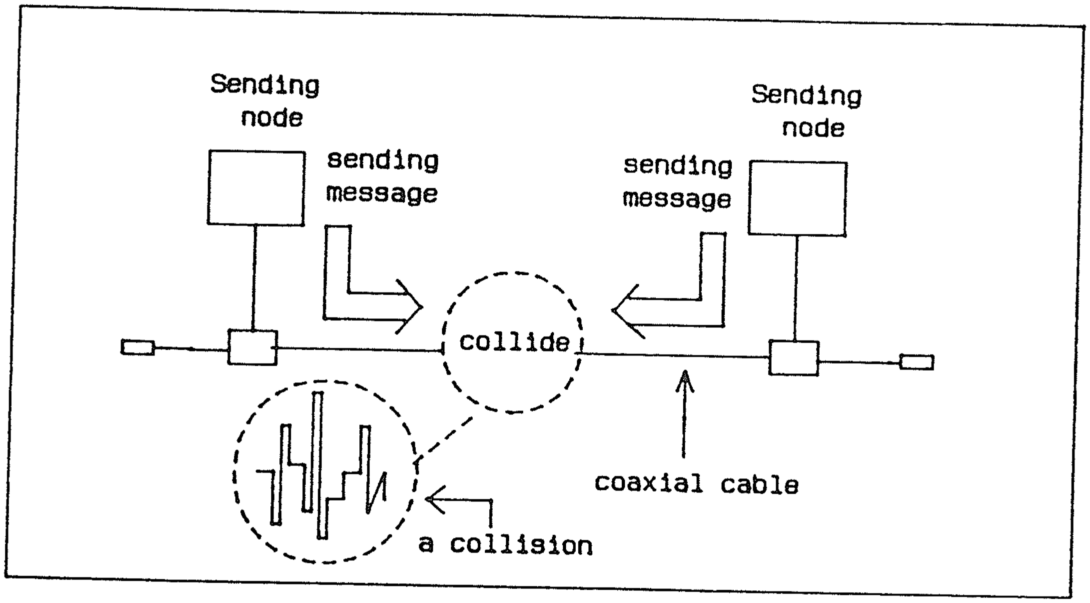


Figure 2.11a Collision Detection.

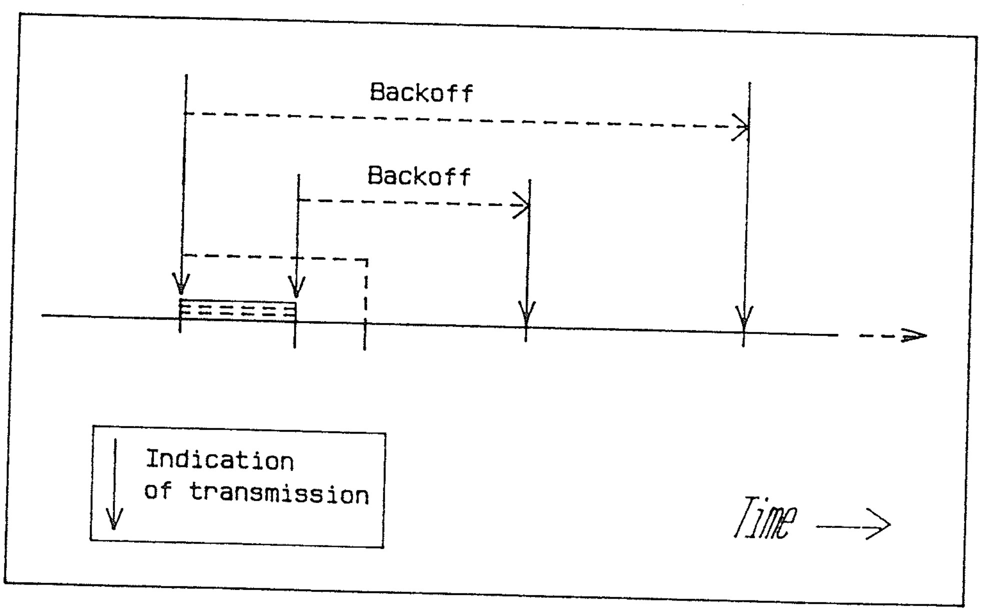


Figure 2.11b Collision and Backoff

2.3.1 Architecture And Functions

The relationship of IEEE 802.3 CSMA/CD to the ISO Reference Model is shown in Figure 2.12. The Data Link Layer of the OSI Reference Model is divided into two sublayers: Logical Link Control (LLC) and Media Access Control (MAC). The Physical layer of the OSI Reference Model corresponds to the Physical Signaling layer (PLS), the Attachment Unit Interface (AUI), the Medium Dependent Interface (MDI), and the Physical Medium Attachment (PMA) in the IEEE 802.3 LAN standard. CSMA/CD is supported by the Media Access Control sublayer (the bottom sublayer of the Data Link Layers) and the Physical Layer.

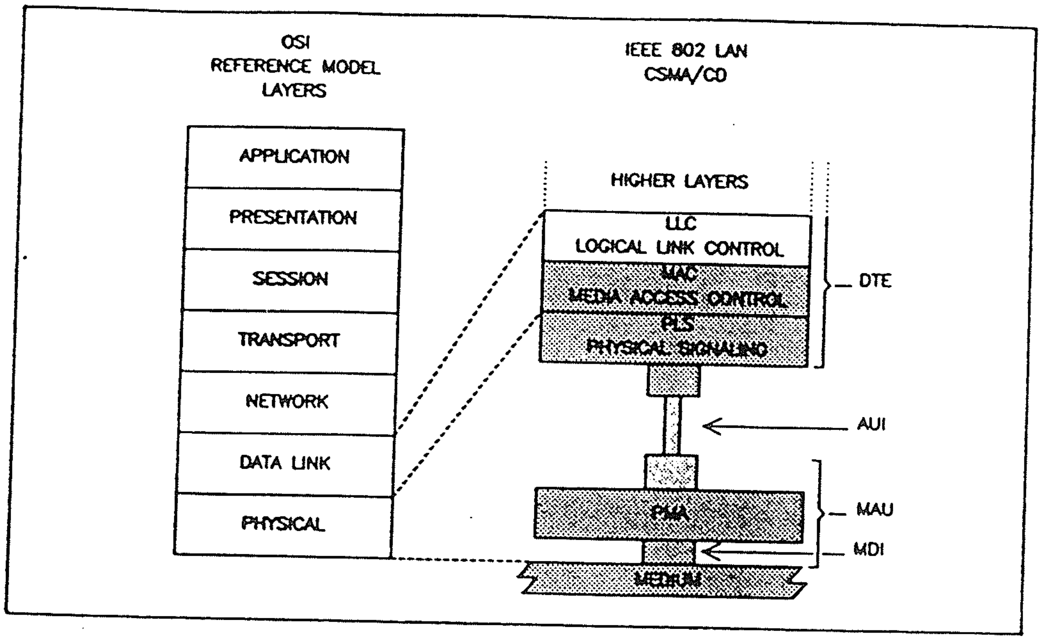


Figure 2.12 The relationship of IEEE 802.3 CSMA/CD to the ISO Reference model.

(from IEEE CSMA/CD, 1985)

Within the MAC sublayer, two interfaces are defined: the Physical Media Interface and the LLC Sublayer Interface. Figure 2.13 shows the MAC sublayer. The Physical Media Interface provides a path for bit streams between the PLS and MAC Sublayers. The LLC Sublayer Interface provides a path to transmit and to receive frames between the LLC Sublayer and MAC Sublayer.

The two main functions performed in the MAC sublayer are:

i) Data encapsulation (transmit and receive)

a) Framing - frame boundary delimitation, and frame synchronization.

b) Addressing - handling of source and destination addresses.

c) Error detection - detection of physical medium transmission errors.

ii) Media access management (collision avoidance)

a) Medium allocation - collision avoidance.

b) Contention resolution - collision handling.

2.3.2 MAC Packet

A frame is defined as a series of transmitted bits grouped together. It may also be referred to as a packet although the term packet sometimes refers to the network layer of the OSI reference

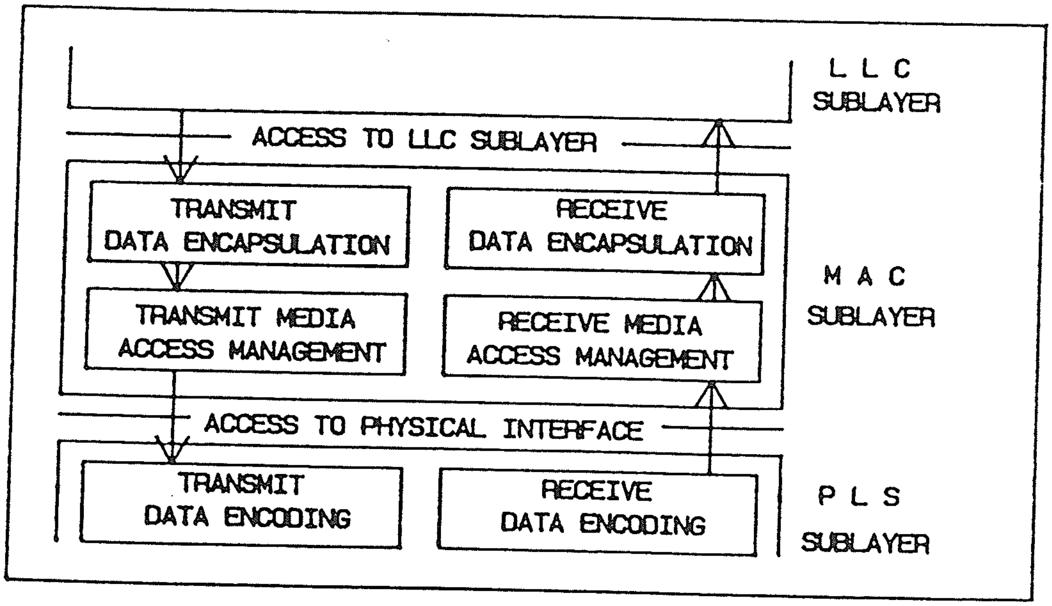


Figure 2.13 Media Access Control Functions

model (Tanenbaum, 1981). Each subgroup of the packet is called a field. The MAC sublayer packet consists of eight fields: preamble, start frame delimiter, destination address, source address, length, LLC Data, pad, and frame check sequence. Each field contains a different number of bits and serves different functions in the packet. During transmission, the fields within a packet are transmitted from the top field (preamble) to the bottom field (frame check sequence). The bits within a field are transmitted from left to right. Figure 2.14 shows the MAC packet format. The preamble field (64 bits) is used for packet synchronization. The start frame delimiter field (8 bits 10101011) is used to indicate the beginning of the packet. The address fields (48 bits each) indicate the address of the receiving node and the transmitting node. The length field (16 bits) indicates the number of octets (8 bits) in the data field. The data field is the LLC data. The pad field is a series of n octets added to the data field when the data field is less than the minimum number of bits required. The Frame Check Sequence field (32 bits) contains a cyclic redundancy check (CRC) code. The maximum packet size is 1518 bytes; the minimum packet size is 64 bytes. Further detail describing each field can be found in IEEE CSMA/CD (1985).

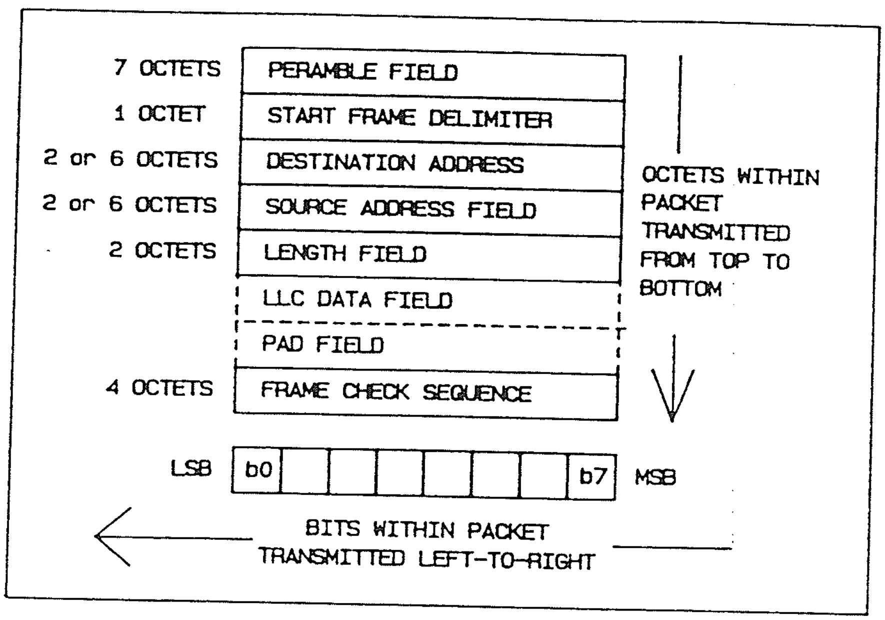


Figure 2.14 MAC packet format.

A message may consist of several packets. Messages with only one packet are called single-packet messages. Messages with more than one packet are referred as multi-packet messages. The minimum interpacket spacing between two packets in a message is called the interframe gap. The interframe gap is intended to provide interframe recovery time for other CSMA/CD sublayers and for the physical medium (IEEE CSMA/CD, 1985). IEEE CSMA/CD (1985) specifies the interframe gap to be 9.6 usec. Figure 2.15 shows a message consisting of three packets.

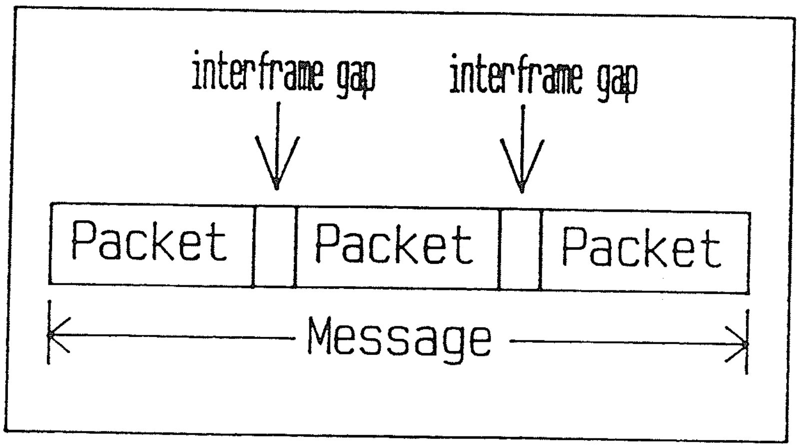


Figure 2.15 A message consists of three packets.

2.3.3 CSMA/CD Network Operation

The operations of CSMA/CD are described in this section. The transmitting node operations, receiving node operations, and collision recovery are discussed in detail.

2.3.3.1 Transmitting Node Operations

When the LLC sublayer requests transmission of a packet, the Transmit Data Encapsulation component constructs a MAC packet from the data supplied by the LLC sublayer. This component checks the validation of the data from the LLC sublayer, then the packet is passed to the Transmit Media Access Management component.

Once the Transmit Media Access Management component accepts the packet from the Transmit Data Encapsulation component, it will attempt to transmit the packet to the PLS sublayer. The Management component first monitors the carrier sense signal provided by the PLS sublayer. If the carrier sense indicates that another transmission has been allocated the transmission medium, the transmission of the packet will be deferred. When the medium is clear, the packet begins to transmit after an interframe gap (9.6 usec). The Management component sends a serial stream of bits to the PLS interface for transmission. The PLS generates the electrical signals on the physical medium to represent the bits of the packet. If the transmission is completed, the MAC sublayer will inform the LLC sublayer. The LLC sublayer will then wait for the next request for transmission from the upper layers.

2.3.3.2 Receiving Node Operations

At each receiving node, the reception procedures are summarized as following. The arriving packet is first detected by the PLS sublayer. The PLS responds to the packet by synchronizing with the preamble field of the packet and the PLS turns on the carrier sense signal. The received packet from the medium is decoded into binary data. The preamble field, and the start frame delimiter field are removed by the PLS and the remaining fields of the packet are passed to the Receive Media Access Management component. The Management component collects bits from the PLS while the carrier sense signal is on. When the carrier sense signal is off, the collected bits are passed to the Receive Data Decapsulation component. The Decapsulation component checks the destination address to decide whether this packet should be received. Also, the component checks for the validation of MAC packet. If the packet is valid and the destination address is recognized, the packet is passed to the LLC sublayer, otherwise, the packet is discarded.

2.3.3.3 Collision And Recovery

If two or more nodes detect that the medium is clear within the collision window, the nodes will begin to transmit and a collision will occur. The PLS notices the interference (collision) on the channel, turns on the collision detect signal, and performs recovery operations. The Transmit Media Access Management component transmits a jam signal (4 bytes of arbitrary data) to ensure that all colliding nodes notice the collision. All colliding nodes wait for a brief time interval (backoff) and attempt to transmit again with the normal transmission procedures. The backoff procedure of Ethernet is called Truncated Binary Exponential Backoff Algorithm.

2.3.3.4 Truncated Binary Exponential Backoff Algorithm

Once packets in the physical medium experience a collision, the sending nodes will backoff and wait a period of time before attempting retransmission. The period of time for waiting is called retransmission delay. The scheduling of retransmission is a critical feature in the performance of Ethernet and is controlled by a truncated binary exponential backoff algorithm. The algorithm uniformly distributes the probability of retransmission within the following interval:

$$0 <= r < 2**k$$

where

r = a random integer,

k = minimum(n,10),

n = the number of retransmission attempts.

During each retransmission, a random integer r is generated between the lower bound 0 and the upper bound 2**k. The packet will wait for a period of time equal to the integer product of r and the slot time, before retransmission. The slot time is slightly larger than the round-trip propagation delay of two nodes which are the maximum allowed distance apart. In IEEE 802.3, slot time is specified to be 512 bit times. The first transmission attempt proceeds with no delay because n=k=0 and r=0, and hence the interval is [0,1). The second and further retransmissions will experience a certain delay according to the integer number n and the integer number r. If the number of retransmission attempts is greater than 10, k will remain as 10 and the transmission interval will be [0,1024). The maximum number of retransmission attempts for each packet is 16. If all sixteen retransmission attempts fail due to collisions, this event is reported as a transmission error and the transmission interval is reset to zero. The protocols of the upper layers of the OSI Reference Model must recognize this error and decide whether to transmit or neglect the packet. Stallings (1984b) has indicated that the disadvantage of the truncated binary exponential backoff algorithm is the last-in, first-out phenomena, nodes which have experienced a small number of collisions will have greater chance of transmitting before nodes that have experienced a relatively large number of collisions.

2.3.3.5 Persistence

Three different persistent CSMA/CD protocols have been described in the literature; non-persistent CSMA/CD, 1-persistent CSMA/CD, and p-persistent CSMA/CD. Ethernet uses 1-persistent CSMA/CD protocol which is described in Figure 2.16. Stallings (1984b) described the different persistence of CSMA/CD as follows.

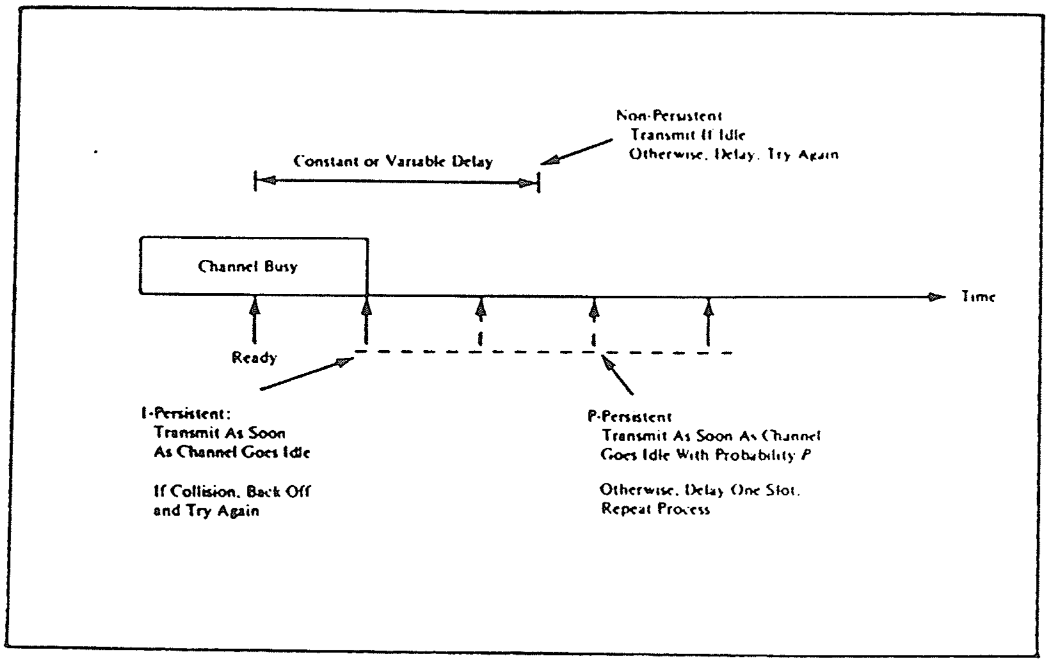


Figure 2.16 Different Persistence of CSMA/CD protocols.
(from Stallings, 1984b).

For a 1-persistent CSMA/CD protocol, a node which has a packet ready for transmission will monitor the physical medium to select one of the following actions.

i) If there is no transmission on the medium, the node transmits the packet immediately.

ii) Otherwise, the node waits until the medium goes idle and then transmits the packet. In addition, IEEE 802.3 and Ethernet specifies that the packet should wait for 9.6 usec before transmission.

iii) While transmitting, the node continues to perform the collision detection procedure. If a collision happens, it will perform the backoff algorithm and repeat step (i).

For the non-persistent protocol, a transmitting node will examine the medium and take the following actions:

i) If the medium is idle, the node transmits the packet.

ii) Otherwise, the node waits a period of time drawn from a probability distribution (the retransmission delay) and repeats step (i) until it successfully acquires the medium.

iii) While transmitting, the node continues to perform the collision detection procedure. If a collision occurs, it will perform the backoff algorithm and repeat step (i).

For p-persistent protocol, a transmitting node will examine the medium and take the following actions:

i) If the medium is idle, the node transmits the packet with probability p, and delays one time unit with probability (1-p). One time unit is equal to the maximum end-to-end propagation delay.

ii) If the medium is busy, the node will continue to listen until the channel is idle and repeat step (i).

iii) While transmitting, the node continues to perform the collision detection procedure. If a collision occurs, it will perform the backoff algorithm and repeat step (i)."

Since the initial design of Ethernet was assumed an excessive bandwidth (Shoch et al, 1982), 1-persistent protocol was believed to be the most efficient protocol. This is because in this protocol, a packet is scheduled for transmission without delay, as soon as the channel is idle.

CHAPTER 3

LITERATURE REVIEW

3.1 INTRODUCTION

In the CSMA/CD access method, packets are sent, one bit at a time, onto the physical channel. The messages may experience collisions which necessitate recovery procedures and cause delay. Therefore, the term performance refers to the accuracy and efficiency with which packets are transmitted from the sending nodes to the receiving nodes. Physical channel utilization, message delay, and error rates are examples of performance parameters. These parameters are directly affected by the number of users, the size of the packets, and the bandwidth of the computer resources. There are two ways to evaluate the performance of CSMA/CD: real measurement from actual networks, and theoretical study (analysis and simulation). The aim of this chapter is to summarize these methods and present a comprehensive view of results from the available literature. A discussion of the merits and inadequacies of the published references are discussed at the end of this chapter.

3.2 METHODS

Measurements from physical networks and theoretical study of conceptual networks are the two major methods which have been used to estimate the performance of CSMA/CD networks. Theoretical studies make use of a model, defined here as a set of conditions and attributes to represent a physical network.

In the review of the literature, there was no reference found which evaluated the IEEE 802.3 performance. Therefore, this review only identifies isolated techniques and results which are applicable and relevant to segments of the current study. The following subsections describe physical network measurements and theoretical studies of conceptual networks.

3.2.1 Physical Network Measurements

Real measurements of an operational network provide an accurate indication of performance which can be used to validate theoretical studies. Actual measurements of network performance are made by attaching special devices, or monitors, to the network, or by using monitoring software to record the network activity. This method is generally more accurate than theory, but it is very difficult to perform because of the fast and complex nature of network traffic.

Two articles which reported actual measurements of CSMA/CD networks are Metcalfe and Boggs (1976), and Shoch and Hupp (1980). Neither references describe their monitoring procedure in detail. Metcalfe and Boggs (1976) published the first paper to describe the Experimential Ethernet. A single Ethernet cable, one kilometer long and operating at 3 Mbps, was used to support 256 users. Shoch, and Hupp (1980) used Experimental Ethernet cable operating at 3 Mbps. This latter network spanned approximately 550 meters and connected over 120 nodes. Network activity was generated by file transfer between difference resources. Monitoring software was used to record the network activity by passive collection, which allowed a monitoring node to sense and record the network traffic in the cable without generating any traffic itself. Both papers used Experimental Ethernet which has not been used since. No detailed experiment measurement of an Ethernet network conforming to Ethernet Specification Version 2.0 was found in this review.

3.2.2 Theoretical Studies

A theoretical study involves the prediction of network activity, to derive certain performance results. Theoretical studies can be further divided into simulation or mathematical techniques.

In this review, the term simulation is applied to a method which makes use of a computer program, while mathematical techniques refer to the use of any direct symbolic formulation to derive network performance results. Both methods were poorly described in the literature. Also, the lack of any consistency in the methods and the results made it impossible to generalize individual techniques.

Simulation uses a computer program to construct the network model. Different values of the variables can be used to represent different configurations and characteristics of the networks. The length of the packets, the length of the cable, and the number of nodes are the examples of the variables. During program execution, the step by step iterations of the computer algorithms predict and represent the events in the network. The events may include network traffic generation, the backoff scheme, acknowledgment, and so on. Different simulations reported in the literature have emphasized different parameters for input to the program. When the execution of the program finished, the numerical results are recorded into a file as estimates of the network performance. Simulations are often based upon the Monte Carlo method (Bratley et al, 1983). They have been written in a variety of languages including C and Fortran and more recently in specially designed software packages which are based on Fortran (Dahmen et al, 1984) and C (Konstantas, 1983). Also, the model may be time or event driven, but the event driven approach has been more common because of its superior efficiency (Konstantas, 1983).

Queuing theory is the most frequently described mathematical technique found in this review. Since there is no standard procedure for the queuing theories, only some elements of queuing theory are presented. The model for queuing theory analysis can be characterized by the number of nodes, the amount of buffer space in the node, the queuing conditions, and the timing conditions. A finite number of nodes is often assumed. The amount of buffer space for the messages to queue in each node can be either infinite or finite. Within the queuing conditions, an infinite number of messages capable of being transmitted in the shared channel is usually assumed. These messages first arrive in the transmitting nodes, then are queued (made to wait) for a turn to transmit. The timing conditions control the arrival and departure of the transmitted messages. Once the queuing model is defined, the solution can be derived by direct mathematical analysis. For further references on queuing theory, refer to Apostolopoulos and Protonotarios (1986), Tasaka (1986), and Tobagi and Hunt (1979).

3.2.3 Terminology

Inconsistency in the use of terms in the literature makes it difficult to compare results. The following terms are defined to facilitate comparsion:

D : the transmission delay between the time a packet is ready for transmission from a node, and the start of the successful transmission. Figure 3.1 shows the definition of transmission delay.

S : the throughput of the network; the total rate of data (packets) being successfully transmitted between nodes.

U : the utilization of the network; the fraction of total channel capacity being used. Since the definition of utilization is relevant and confused in different papers, utilization in this review is clarified as channel capacity being used by uncollided packets only.

OG : the offered load to the network; the total rate of data (packets) presented to the network for transmission.

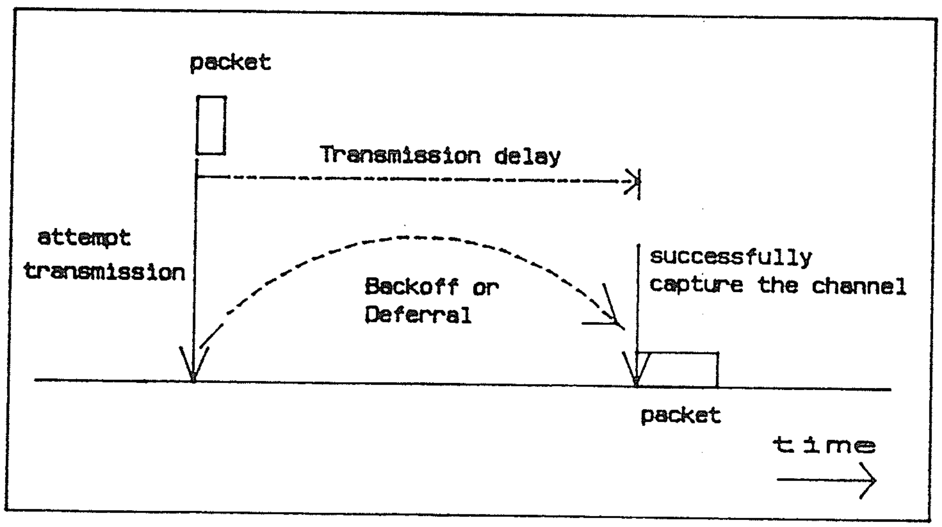


Figure 3.1 The definition of transmission delay.

For example, if the total data offered to the network is 1.2 Mbps, and the total data successfully transferred between nodes is 1 Mbps, then the offered load OG is 1.2 Mps, the throughput S is 1.0 Mbps. In this example the utilization U is 0.1, since the total capacity (bandwidth) of Ethernet channel is 10 Mbps. The parameter S is often normalized and interpreted as utilization U in some papers.

Alpha is an important parameter which affects the performance of the network (Stallings, 1984b). Alpha is defined as the ratio of the length of the transmission medium (expressed in the equivalent in bits between two nodes), to the size of the packet in bits. Alpha can be further expressed as the fraction of one way propagation time to the equivalent time length of the packet. Typical values of Alpha range from 0.01 to over 1.0 for local area networks (Stallings, 1984b).

$$\text{Alpha} = \frac{\text{one way propagation time (in bits)}}{\text{size of the packet (in bits)}} \qquad \ldots.(1)$$

3.3 **RESULTS**

This section summarizes the related performance results found in the literature.

3.3.1 General Statements

The major results from the literature survey are summarized in Table 3.1a to Table 3.1f. The Tables are divided into four columns which are Author, Method, Protocol, and Related Performance Results. Under Method, the different methods are distinguished. Under Protocol, any modifications to the CSMA/CD protocol which have been introduced are indicated. The modifications were mainly in the transmission attempt strategy. Only the major results are described in the Related Performance Results.

AUTHOR	METHOD	PROTOCOL	RELATED PERFORMANCE RESULTS
Arthurs and Stuck, 1984	Analytical	Modified CSMA/CD	1) Modified CSMA/CD offers a shorter mean retransmission delay than non-modified CSMA/CD.
Bux, 1981	Analytical	CSMA/CD	1) Inefficient operation if alpha is large. 2) transmission delay increases rapidly for loads at about 0.6.
Choudhury and Rappaport, 85	Simulation& analytical	Modified CSMA/CD	1) Shorter length packets reduce the channel utilization.
Colvin, 1983	Analytical	CSMA with collision avoidance	1) CSMA with collision avoidance resolves collisions more effectively than CSMA/CD.
Dahmen et al, 1984	Simulation	CSMA/CD	1) Greater packet sizes can have better performance with CSMA/CD. 2) Mean waiting time increases exponentially with the total offer loads. 3) Mean waiting time increases with packet length.
Haenle and Giessler, 1978	Simulation	No specified protocol	1) Expected graphs of throughput vs applied load. 2) Expected graphs of applied load vs delay.

TABLE 3.1a : Summary of the literature results

AUTHOR	METHOD	PROTOCOL	RELATED PERFORMANCE RESULTS
Heyman, 1986	Analytical	CSMA/CD	1) If packet size increases, throughput and average response time will improve. 2) Analytical equations.
Jenq, 1986	Analytical	CSMA/CD	1) Analytical equations for channel utilization. 2) Small value of alpha can give a better channel utilization.
Kanakia and Tobagi, 1986	Measurement	Data Link LLC protocol	1) the potential bottleneck occurs at the data link layer. 2) suggest further development on high speed network interface unit.
Marathe and Hawe, 1982	Analytical	CSMA/CD	1) Transmission delay is often smaller than the delays in the higher levels. 2) Offered load is directly proportional to the number of users. 3) Mean waiting time is proportional to number of users. 4) In heavy load (2000 users), a packet only experiences one collision on average per successful transmission.

TABLE 3.1b : Summary of the literature results

AUTHOR	METHOD	PROTOCOL	RELATED PERFORMANCE RESULTS
Metcalfe and Boggs, 1976	Measurement	CSMA/CD	1) For large packet size (above 4000 bits), the utilization of the experimental Ethernet stays well above 95%.
Moura et al, 1979	Simulation	modified CSMA/CD	1) The suggested backoff algorithm performed better than truncated binary backoff algorithm in term of delay throughput characteristic.
O'Reilly and Hammond, 1982	analytical simulation	CSMA/CD	1) If the number of stations increases, the throughput will increase up to a saturation point and remain constant.
Pendse and Soueid, 1985	analytical	modified CSMA/CD	1) The effects of channel capture based on priority.
Schacham and Hunt, 1982	analytical	1 persistent modified CSMA/CD	1) The mean retransmission delay strongly affects the network performance. If the mean is too small, more collisions occur. 2) Packet length and cable length affect the the network performance.
Sherman et al 1978	analytical	Non persistent CSMA/CD	NIL

TABLE 3.1c : Summary of the literature results

AUTHOR	METHOD	PROTOCOL	RELATED PERFORMANCE RESULTS
Shoch et al, 1982	analytical	CSMA/CD	1) Under heavy load, only a short period of time on the channel is lost due to collisions. 2) Channel efficiency depends on the size of the packets. Larger packet sizes have better network performance. 3) When the packet size approaches the collision interval, network performance degrades to CSMA without collision detection. 4) Ethernet has been designed to have excessive bandwidth. Ethernet network should be run with a sustained load less than 50% of channel capacity.
Shoch and Hupp, 1980	measurement	CSMA/CD	1) Ethernet is robust. The performance degrades slowly and recovers well from momentary overloads. 2) If Alpha decreases, the network performance increases. 3) On average, an individual packet does not experience many collisions. 4) The transmission error rates are very low and very few packets are lost.

TABLE 3.1d : Summary of the literature results

AUTHOR	METHOD	PROTOCOL	RELATED PERFORMANCE RESULTS
cont: Shoch and Hupp, 1980			5) Under normal loads, 99.18% of the total packets successfully transmitted without latency. 0.79% of the total packets were delayed due to deference. Only 0.03% of the packets involved in collisions. 6) Ethernet utilization increases with packet size. 7) Under extreme overload, Ethernet channel is still stable. The channel utilization remained above 97%.
Stallings, 1984	General study	CSMA/CD	1) General analytical equations for the performance of CSMA/CD.
Takagi and Kleinrock, 1985	Analytical	Modified CSMA/CD	1) Mean delay is exponential with total throughput. 2) Proposed slotted CSMA/CD.
Tasaka, 1986	Analytical	Modified CSMA/CD	1) The performance of bufferd users on slotted nonpersistent CSMA/CD protocol.

TABLE 3.1e : Summary of the literature results

AUTHOR	METHOD	PROTOCOL	RELATED PERFORMANCE RESULTS
Tobagi and Hunt, 1979	Analytical	non persistent CSMA/CD	1) The throughput delay characteristics of CSMA/CD are better than CSMA. 2) If the collision detect time is fixed, long packets have better performance in channel utilization than short packets.

TABLE 3.1f : Summary of the literature results

From Tables 3.1a to 3.1f, the significant results are summarized as following:

i) If Alpha is small, the high utilization U of the channel can be achieved. In order to make Alpha smaller, a greater packet size (above 4000 bits) is used. This is because packets are exposed to collisions only during the collision interval of their transmissions. Once a packet has been on the physical medium for that length of time, no collision should occur. This result was reported by Bux (1981), Choudhury and Rappaport (1985), Dahmen et al (1984), Heyman (1986), Shoch et al (1982), and Marathe and Hawe (1982).

ii) Certain modified CSMA/CD protocols have demonstrated superior performance to the Standard IEEE 802.3 CSMA/CD protocol. This result was reported by Arthurs and Stuck (1984), Colvin (1983), and Pendse and Soueid (1985).

iii) Message delays and collisions are infrequent phenomenon in the network traffic. On average an individual packet experiences few collisions per successful transmission (Marathe and Hawe, 1982 ; Shoch and Hupp 1980).

iv) During momentarily heavy overload, Ethernet performance degrades slowly and recovers well. Channel utilization remained above 97% in extremely heavy offered loads (Shoch and Hupp, 1980). However, the channel utilization in this paper included collisions, deferrals, interframe gap and packet data, and therefore over-estimated the network performance.

3.3.2 Performance Equations

It is difficult to compare performance equations among different references due to the fact that each paper used a different model. Also, equations derived from queuing theory are complex in nature, and not applicable in this research. Therefore, only the basic equations describing the simple case of unmodified CSMA/CD are presented in this review.

The equations of acquisition probability A, the transmission delay D, and the throughput S are derived by direct mathematical techniques which can be found in Stallings (1984b) and Metcalfe and Boggs (1976). See also section 3.2.3 for definitions.

Acquisition probability A is the probability that only one node attempts a transmission and successfully acquires the channel. In this case, real time is considered to be divided into slot time. Each slot time is one end-to-end round trip propagation time between the two furthest nodes. Let N be the number of nodes continuously sending out packets. Assume also that a station attempts to transmit in a current slot time with probability 1/N, or delays with probability 1-(1/N). Therefore,

$$A = (N) * (1/N) * ((1 - (1/N)) ** (N-1))$$
$$= (1 - (1/N)) ** (N-1) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3)$$

The transmission delay D can be represented in terms of slot time. The probability of a packet waiting 1 slot time before acquiring the channel is A*(1-A); the probability for waiting i slots is A*((1-A)**i). Therefore, the expected mean value of transmission delay, E[D] is the geometric distribution of above two probabilities.

$$E[D] = (1-A) / A \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4)$$

The throughput S can be derived as the ratio of packet transmission time to the packet transmission time and delay time.

$$S = \frac{\text{packet transmission time}}{\text{packet transmission time + transmission delay}} \quad \ldots(5)$$

In order to simplify Equation (5), Equation (1) is used. Also, the one way propagation delay is assumed to be half of the slot time, and the one way propagation delay is always calculated between the furthest two nodes. Equation (1) can then be expressed as:

$$\text{Alpha (a)} = \frac{\text{(1/2) slot time}}{\text{packet transmission time (bits)}} \qquad \ldots\ldots\ldots\ldots(6)$$

Substituting (6) into (5),

$$S = \frac{1/2a}{1/2a + ((1-A)/A)}$$

$$= \frac{1}{1 + 2a * ((1-A)/A)} \qquad \ldots\ldots\ldots\ldots\ldots(7)$$

If the number of stations (N) becomes infinite, the acquisition probability A becomes $\lim_{N \to \infty} (1-(1/N)) ** (N-1) = 1/e$ Therefore,

$$\lim_{N \to \infty} S = \frac{1}{1 + 3.44a} \qquad \ldots\ldots\ldots\ldots\ldots(8)$$

Equations (7) and (8) are commonly referred to the throughput or utilization of the CSMA/CD protocols.

3.3.3 Graphic Output

Graphic output has been used for both theoretical studies and real measurements. It is difficult to compare the graphs from different papers because each paper uses its own model and parameters. Therefore, this section only shows the general characteristics of the graphs from the available papers.

Figure 3.2 shows the channel utilization U as a function of the total offered load OG. As the total offered load increases, the channel utilization will increase to a maximum level. In the ideal case, the maximum utilization is the maximum bandwidth (capacity) of the physical channel. Also, the maximum utilization is depended on the packet size and Alpha in a heavily offered load. Shoch and Hupp (1980) derived this result from their measurements of an experimental Ethernet. Similar findings have been reported for the theoretical studies (Heyman, 1986 ; Jeng, 1986). It should be noted that the simplicity of this result is because there was no loss of bandwidth due to collisions, deferrals, and interframe gap.

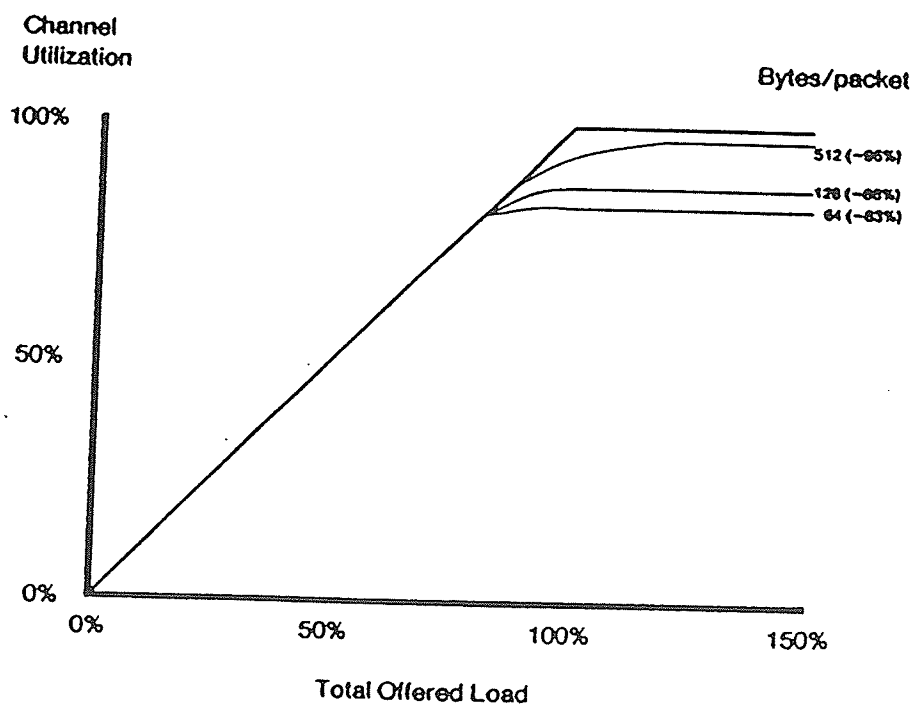


Figure 3.2 The channel utilization as a function of the total offered load. (from Shoch and Hupp, 1980).

Figure 3.3 shows the channel utilization U as a function of the number of stations. The assumption is made that the number of stations were continuously transmitting messages. As the number of stations increased, the channel utilization decreased because more collisions took place and more packets waited for transmission. For large size packets, the total utilization was only decreased by a small percentage. For small size packets, the total utilization was decreased by a large percentage. This result can be found in Shoch and Hupp (1980) and Metcalfe and Boggs (1976).

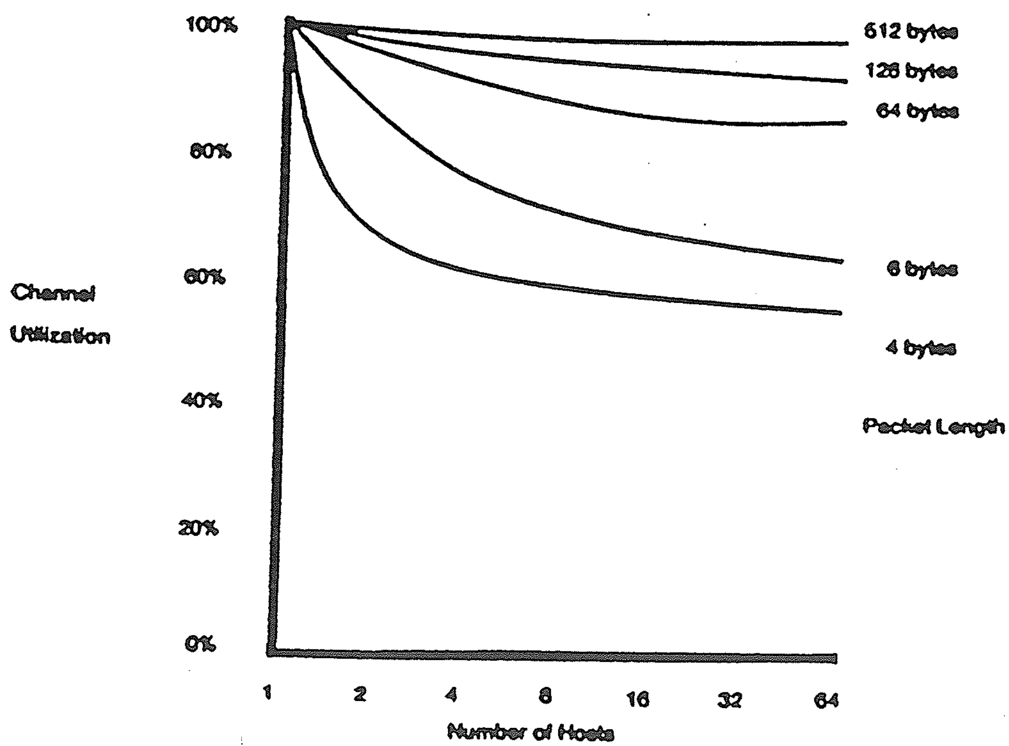


Figure 3.3 The channel utilization as a function of the number of hosts. (from Shoch and Hupp, 1980).

Figure 3.4 shows the transmission delay D as a function of the throughput S. If the throughput increases continuously, more collisions occur, and individual packet experience more delays and backoffs to achieve successful transmission. From Figure 3.4, in which transmission delay D is defined as the ratio of mean transfer time to the mean packet transmission time, D is exponentially proportional to the throughput S. Packets experienced a large delay when the throughput was greater than 0.5. Takagi and Kleinrock (1985), Bux (1981), and Dahmen (1984) reported this result from their theoretical studies.

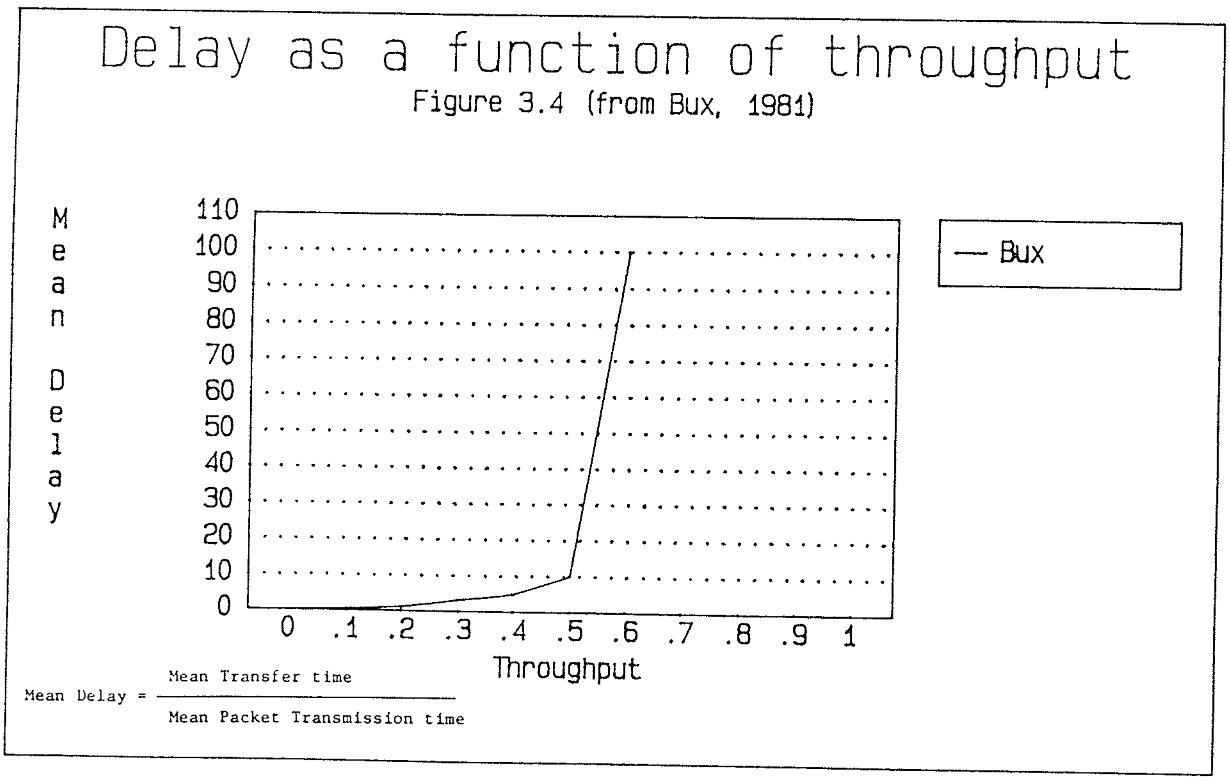
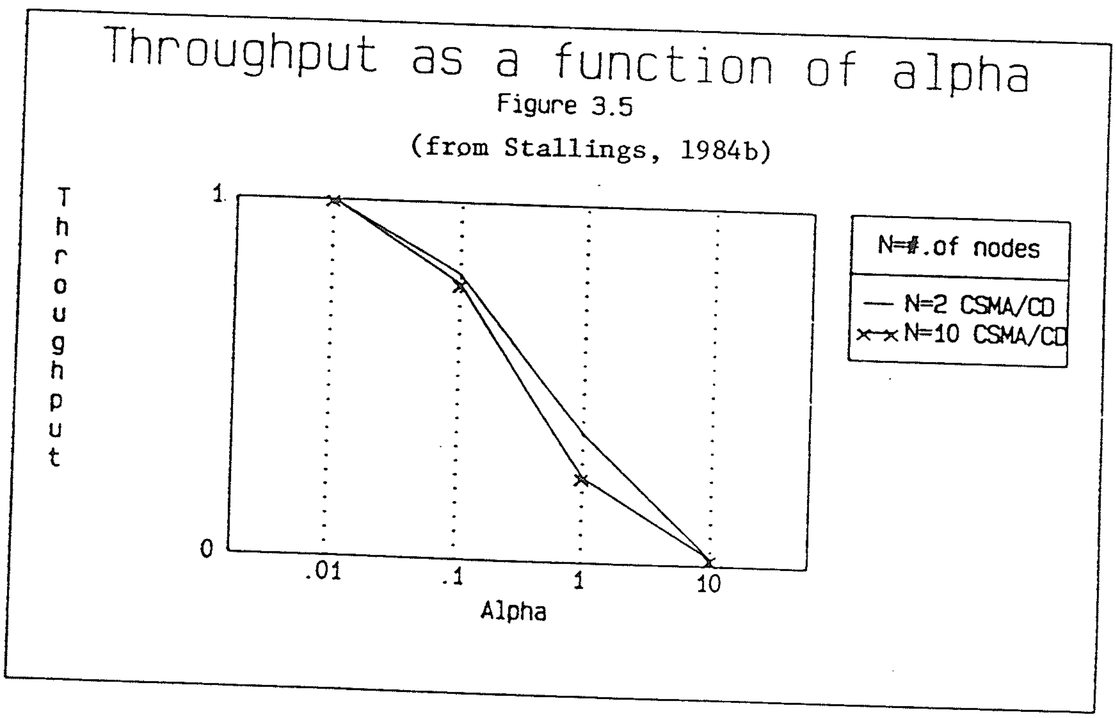


Figure 3.5 shows the throughput as a function of Alpha. Throughput is defined as the total rate of data being transmitted between nodes (Stallings, 1984b). The throughput was degraded when Alpha was increased. Thus, a decrease in the packet size can decrease throughput. This result was reported by Stallings (1984a) and Stallings (1984b). It is important to notice that this result applies to single-packet messages, and not necessarily to the throughput of multiple-packet messages. Also, Stallings (1984a) and Stallings (1984b) did not provide a formal proof of this result.



3.4 DISCUSSION

The accuracy and general implications of the literature review are discussed in this section.

3.4.1 Accuracy

Performance results of Ethernet as reported in the literature were derived from theory or physical measurements. Since both methods used certain assumptions and limitations to formulate a model, the performance results are strictly valid only for that particular model. Different models may have different results.

Some major limiting assumptions expressed by the literature review are:

i) The transmission medium is perfect.

All theoretical studies assumed that the transmission medium is noiseless, the channel interferences are caused by transmission collisions only. However, in the physical network, noise can be caused by any transmission fault. CRC errors may be generated in this case. The high reliability of current technology should reduce this difference to insignificant proportions.

ii) Constant packet length or variable packet length.

Since real Ethernet network activity consists of variable size packets, it is more realistic to use a variable packet length in network performance estimation. Tobagi and Hunt (1979), and Heyman (1986) used variable packet length while Bux (1981) and Jeng (1986) used constant packet length for their studies. The use of constant packet size should yield accurate average results, but may produce under-estimates of the performance when higher than average length packets are used.

iii) The Ethernet Specification is 10 Mbps for the bit rate. However, most of the results were derived from Experimental Ethernet which used 3 Mbps (Moura, 1979; Shoch and Hupp, 1980; O'Reilly and Hammond, 1982). Also, the structure of the MAC packet and backoff algorithm between the Ethernet Specification and the Experimental Ethernet are quite different. For example, the preamble size and the maximum backoff interval. Therefore, the performance results from the Experimental Ethernet cannot represent the performance of the Ethernet Specification.

iv) The overhead for Ethernet is 26 bytes. Some papers used smaller overhead, such as Shoch and Hupp (1980), and Bux (1981). If small overhead (less than 26 bytes) is used, the network performance is over-estimated.

v) Single-packet messages.

Higher levels of software generally use multi-packet messages. The Ethernet protocol specifies that each packet be separated by at least one interpacket gap 9.6 usec. The majority of articles used single-packet messages, including Tasaka (1986) in his theoretical study. If single-packet messages are used, the characteristics of the channel utilization of CSMA/CD is incompletely represented. This is because each individual packet in the message may be separately delayed due to deferrals or collisions; if only single-packet messages are considered, the chance for the delay to occur is smaller than the multi-packet messages.

vi) Ethernet uses 1-persistent CSMA/CD protocol while some papers used different persistent protocols such as Tobagi and Hunt (1979), and Tasaka (1986). Different persistent CSMA/CD protocols have different performance results. Only the results from 1-persistent protocols were of interest in this report.

vii) Network Size.

Small networks, generally one long single coaxial segment, were used by all articles. Neither repeaters nor bridges were used. Repeaters and bridges are typically required in any study of practical network designs.

viii) Constant propagation delay (time).

Most papers assumed that the propagation delay was small compared to the packet transmission time, and that it was identical for all source and destination nodes (Moura et al, 1979; Marathe and Hawe, 1982). In fact, propagation delay is highly variable.

ix) Real time was divided into slots, but not bit time.

Each slot time was equal to the two way propagation delay. The transmission can occur at the beginning of the slot only. This assumption under-estimated the channel traffic because transmissions can be attemptted at any bit time in the real situations.

x) No acknowledgment was required to complete a message transaction.

This assumption was used by all references and caused an over-estimation of network performance because acknowledgments are small packets which use the channel bandwidth. This is especially true for full duplex terminal traffic, where total message delay must include the receipt of an acknowledgment on echo.

xi) Different backoff algorithms.

Ethernet uses truncated binary exponential backoff algorithm. However, some papers used different backoff algorithms (Schacham and Hunt, 1982; Choudhury and Rappaport, 1985). The effect of these different algorithms on the results is unknown but can be expected to be significant by virtue of their presence.

In order to make the model tractable, assumptions are necessary. However, they should be carefully considered because assumptions may easily make the model unrealistic.

Many papers used modified CSMA/CD protocols instead of using the Standard IEEE 802.3 CSMA/CD protocol in this review. The modified CSMA/CD protocols claimed that their performances were better than Standard IEEE 802.3 CSMA/CD. However, the modified CSMA/CD were only analytical predictions. Since modified CSMA/CD had not been implemented in real environment, no measured evidence was provided.

In general, the published results found during this review were too restricted and not applicable to the current study.

3.4.2 <u>Observations</u>

There were very few papers which dealt specifically with the IEEE 802.3 CSMA/CD protocol. Among those papers, the majority focused on theoretical studies rather than physical measurements. This is likely because physical measurements are more expensive to perform than are the generation of theoretical results. Also, measurement results are more difficult to evaluate than theoretical results because of the complex activity involved in the physical network. Two actual measurements of experimental Ethernet were found in this review. No measurements based on the Ethernet Specification were found in the available references.

Since the terminology was inconsistent in the papers, many authors calculated the channel utilization which included the collisions and deferrals. This is an extreme over-estimation of the bandwidth utilization.

The performance of modified CSMA/CD protocols were claimed to be better than the Standard IEEE 802.3, however, standardization is more important in practice. In addition, Ethernet is commonly accepted by most of the vendors. Therefore, the performance of Ethernet is more of interest than modified CSMA/CD protocols.

In the theoretical studies, queuing theory and simulations are used commonly. Simulations can represent a more complicated but flexible network than queuing theory, because the computer program can be creative and flexible (Mayne, 1986). However, developing a precise program to represent the network behaviour correctly and effectively will require a substantial amount of work.

In summary, no single paper can be used as definitive reference for this study. Published performance results were dependent upon the particular conditions of the models. The methods were too limited and restricted. Nevertheless, the general nature of an Ethernet has been documented and may serve to explain the results described in subsequent chapters.

CHAPTER 4

METHODS

4.1 INTRODUCTION

Simulation is the general method which was used in this research. Other techniques such as queuing theory and direct mathematical analysis, as introduced in the previous chapter, were found to be too limited and restricted. The objective of this research was to accomodate complicated and realistic networks, such that mathematical techniques were not appropriate (Bratley et al, 1983; Sauer and MacNair, 1983).

In this research, the conceptual model was IEEE 802.3 CSMA/CD. Simulation was used to generate the solution. In addition the procedures account for multilple networks and certain features characteristic of higher levels of software, such as multi-packet messages and device bandwidth which are useful in performance analysis. In this chapter, the technique and computer program of the simulation are described in detail.

4.2 TECHNIQUE

Simulation made use of an event driven procedure at the bit level. Time was assumed to be finite and consisted of a sequence of time slices. The time duration of each time slice was in the range of 1 to 20 seconds. Each time slice was divided into 100 nanosecond bit times. Events involve any active use of the media, and may occur at the start of any bit time. Figure 4.1 shows the definition of time.

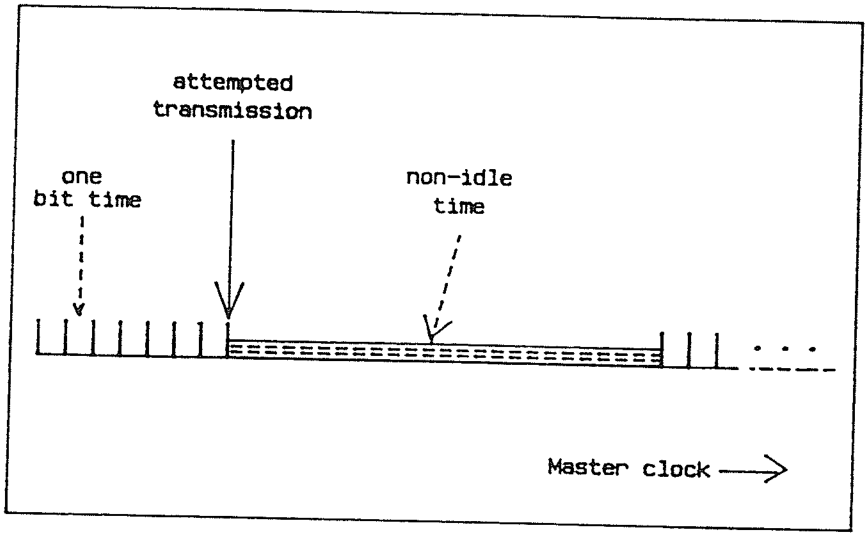


Figure 4.1 Graphic representation of time.

The events encountered in the network simulation are : collisions, backoff, delay, attempted transmission, deferral, and successful transmission. In the simulation, attempted transmissions are first randomly or specifically distributed in the time slice. Then, all the network events are resolved according to the position on the time slice in the manner described in subsequent paragraphs.

The following notations are used in order to present the events in the time slice:

Ti = the ith attempted transmitted packet of a message.

Mj = the jth attempted transmitted message on the time scale.

Nk = the kth network.

P = the time period used by transmitted packet.

B = the position in the time scale where the transmission begins.

C = the position in the time scale where the collision period passed.

E = the position in the time scale where the transmission ends.

G = the interframe gap which is 9.6 usec.

Thus Nk(Mj(Ti)) refers to the ith attempted transmitted packet of the jth message in the kth network.

In the algorithm developed in this research, each network has its own time scale or time clock. Network 1 has time clock1 while network 2 has time clock2, and so on. Different time clocks are superimposed to form a master clock. Figure 4.2a shows an example of two time clocks, with their superposition on to master clock in Figure 4.2b. The notation Ni is referred to an event in the ith network.

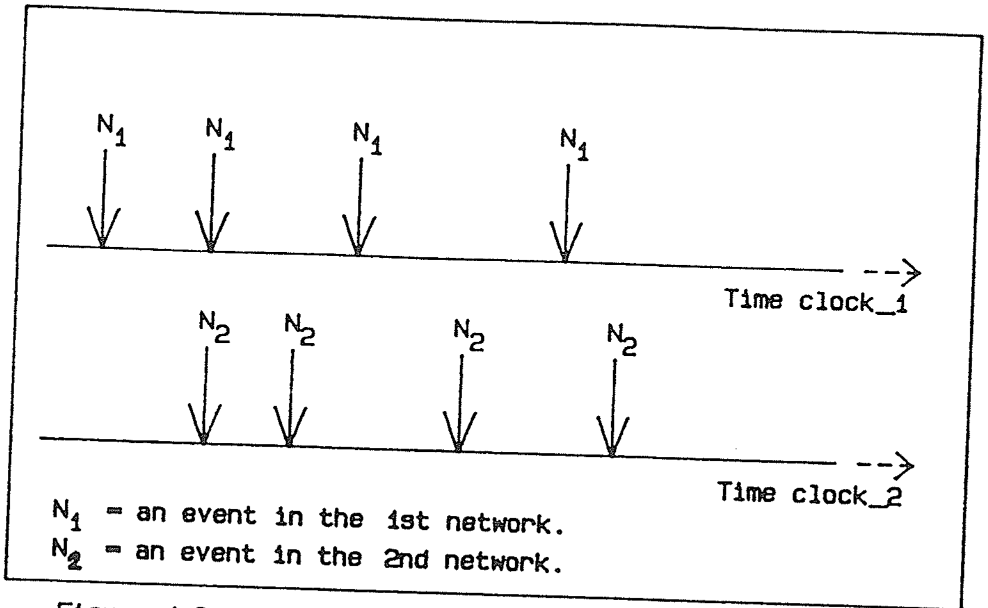


Figure 4.2a Time clocks of the multi-network.

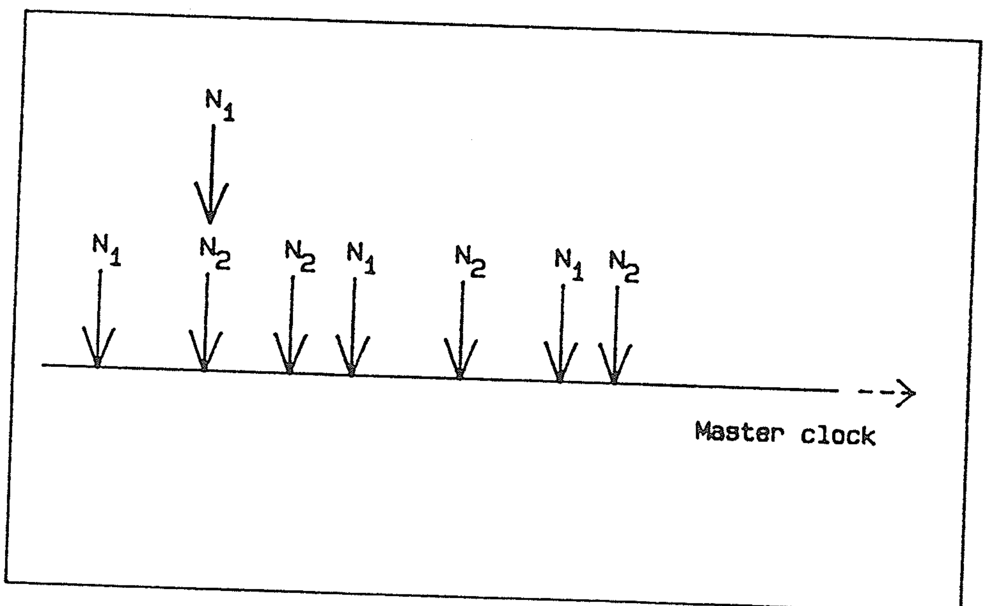


Figure 4.2b Master clock of the multi-network.

For simplicity the algorithm is explained with the aid of a single network and a message consisting of a single packet. Several cases are considered to demonstrate the possible range of events.

Case 1: Successful Transmission.

Successful transmission of a packet is shown in Figure 4.3. Successful transmission is fulfilled if a node attempting to transmit a single packet at a bit time (position B), with no other packets positioned within the collision window B-C. The collision window is calculated for each attempted transmission within the collision checking interval, which is equal to the maximum two way propagation time, and is set in IEEE 802.3 to be 512 bit times. Beyond C, carrier sense will avoid collision. If any attempted transmissions are positioned between C and E, they will defer.

Deferred transmission occurs when a node attempts to transmit a packet while the channel is being used by another transmitted packet. Figure 4.4 shows the definition of deferred transmission. A packet T1 successfully passes the collision interval B1-C. Another packet T2, attempts to transmit at the position B2. However, as the bandwidth of the channel was being used by the previously transmitted packet T1, T2 must defer to a further position T2'.

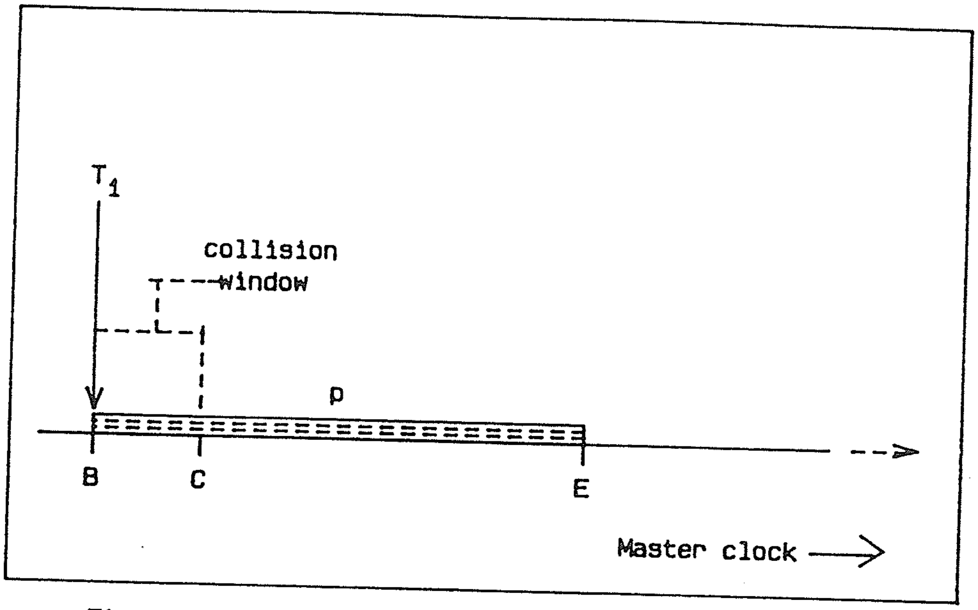


Figure 4.3 Successful transmission of a packet.

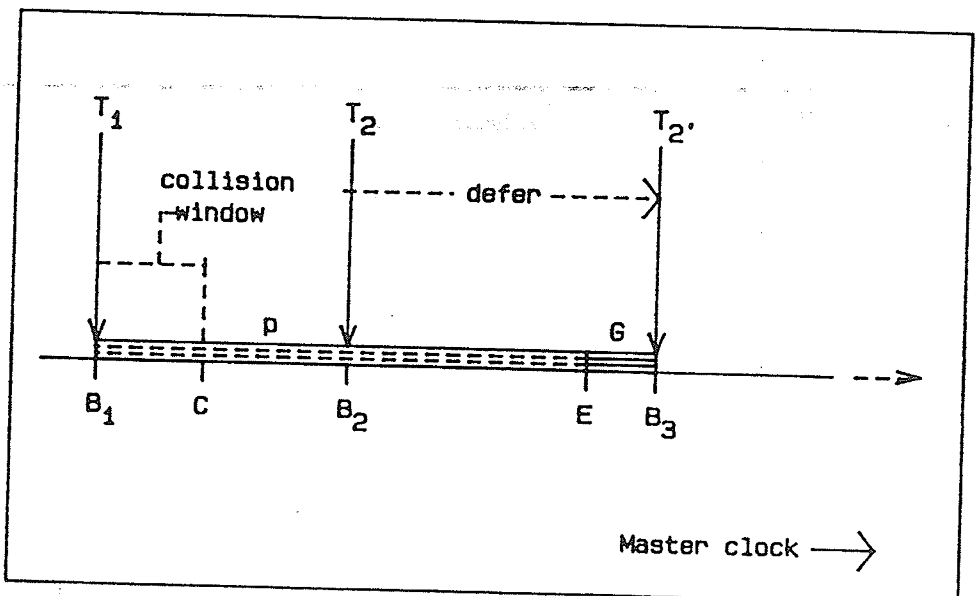


Figure 4.4 Deferred transmission.

A collision is caused by several transmissions interfering with each other. Figure 4.5a and figure 4.5b shows the two cases of collision.

Case 2: Collision (case a).

In figure 4.5a, two transmissions (T1 and T2) which are on the same network attempt to transmit at the same bit time (position B). T1 and T2 collide with each other and produce an interference. In the simulation, T1 and T2 are moved into a further position in the time slice by the truncated binary backoff procedure specified by IEEE 802.3 (IEEE CSMA/CD, 1985).

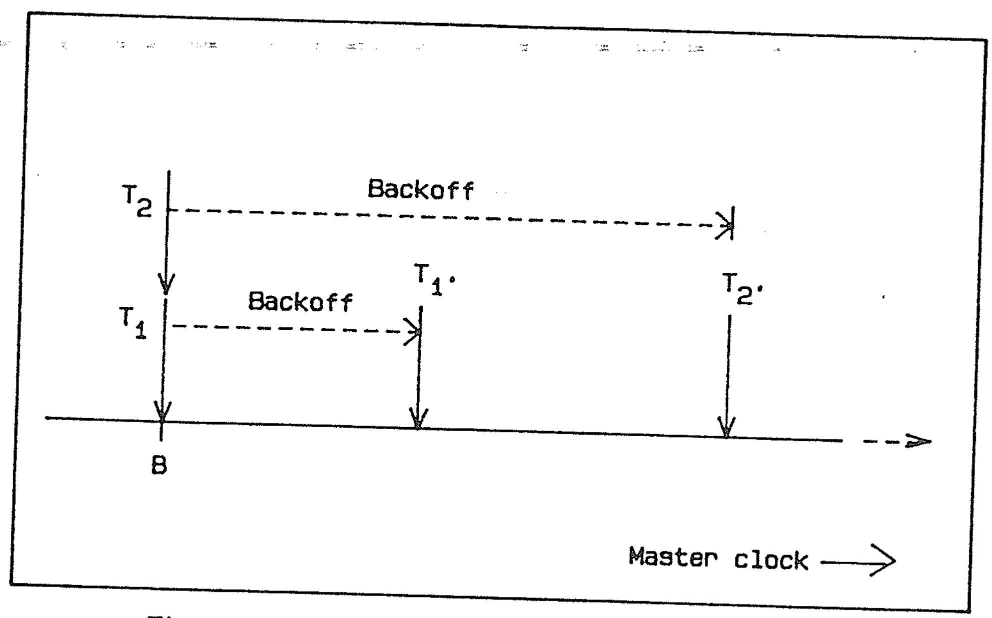


Figure 4.5a Collision (case a)

Case 3: Collision (case b).

In figure 4.5b, the packet T1 is begun at the position B1. In this event, T1 did not pass its collision interval B1-C because another packet T2 is started and a collision occurs by the interference of T1 and T2. Both transmitted packets T1 and T2 are aborted and backed off to positions B1' and B2', respectively, in the time scale. If packet T2' is a successful transmission later on in the time scale, the delay is the time interval between B2 and B2'. Similarly, if packet T1' is a successful transmission at position B1', the delay is the time interval between B1 and B1'.

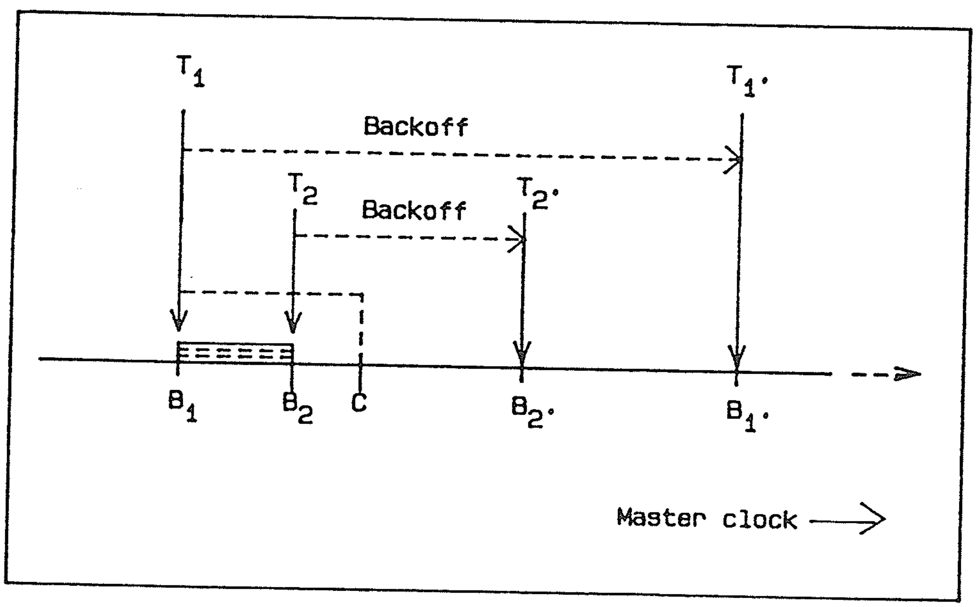


Figure 4.5b Collision (case b)

This algorithm is a simplification of actual events in that it ignores the finite propagation delay between the initiation of a transmission and the detection of a collision. Over a 500 meter segment, the worst case delay would be 4.3 microseconds.

To this point, network events for a single network and a message consisting of a single packet have been illustrated. However, a message may consist of multiple packets. The network events for a message which consists of two packets are described as follows.

Case 4: Successful Transmission - Multiple Packet.

Successful transmission of a single two-packet message is shown in Figure 4.6. A node attempts to transmit a message M1 which consists of two packets, T1 and T2. Packets T1 and T2 are queued in the node, and transmitted one by one according to their position in the time slice. Transmission of the first packet of the message, M1(T1), is attempted at position B1. The second packet of the message, M1(T2), is attempted at position B2. Both M1(T1) and M1(T2) successfully pass their respective collision intervals, B1-C1 and B2-C2 as the network is idle at these times.

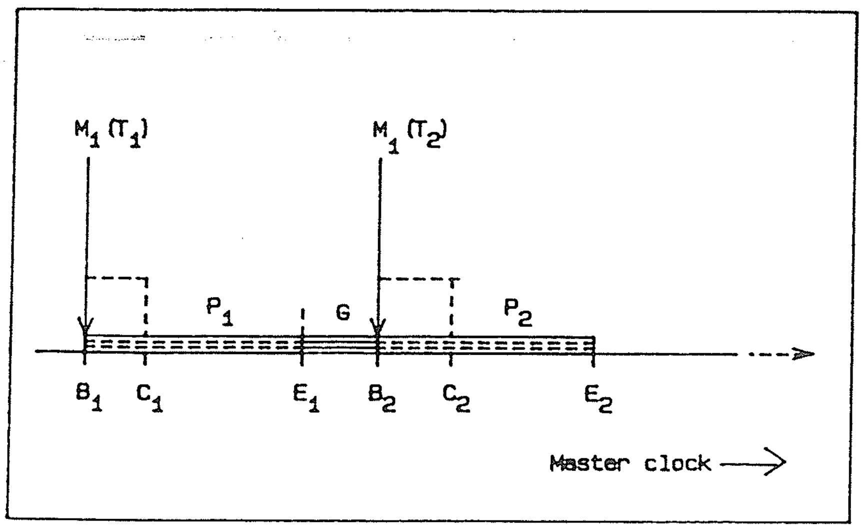


Figure 4.6 Successful transmission of a message.

Case 5: Reply Packets.

After the receiving node successfully receives all the transmitted packets of a message, an acknowledgment (replied packet) will be send back from the receiving node to the transmitting node if indicated by the transmitting node. The acknowledgment is a single packet, with a size equal to the length of one packet of the receiving node. For example, if T1 is assumed to be the last successfully transmitted packet of the message M and T1 contained a request for acknowledgment, the packet Tk would be generated at the end of the transmitted packet T1.

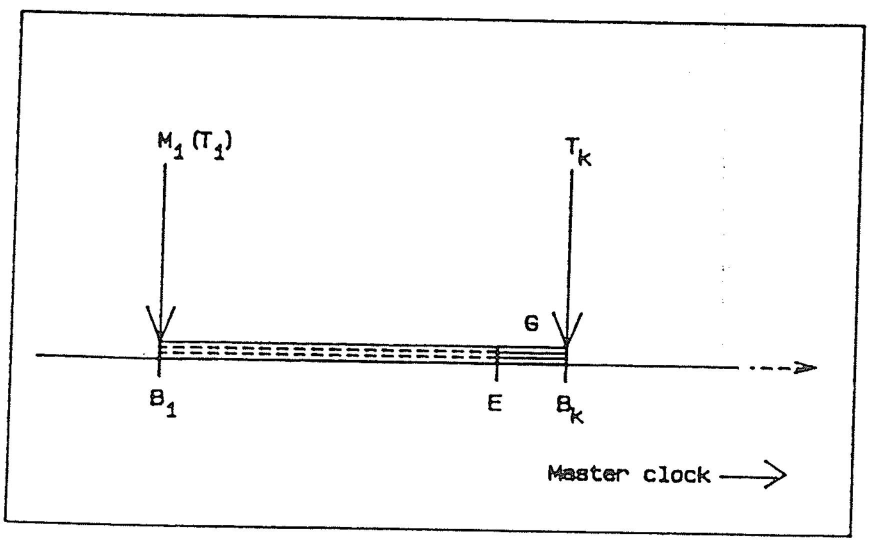


Figure 4.7 Acknowledgment.

Case 6: Deferred Transmission.

Figure 4.8 demonstrates a deferred transmission. Packet M1(T1) is assumed to be successfully transmitted. A second message, M2, consists of two packets, T1 and T2. During the bit by bit transmission of the M1(T1), a packet M2(T1) is scheduled at position B21, within interval C1-E1. Because of M1(T1), M2(T1) must defer to the position B21', which is the end of the packet M1(T1). M2(T2), which belong to the message M2, has to be shifted from B22 to position B22'. It should noted that the time period between B21 to B22 is made the same as B21' to B22'.

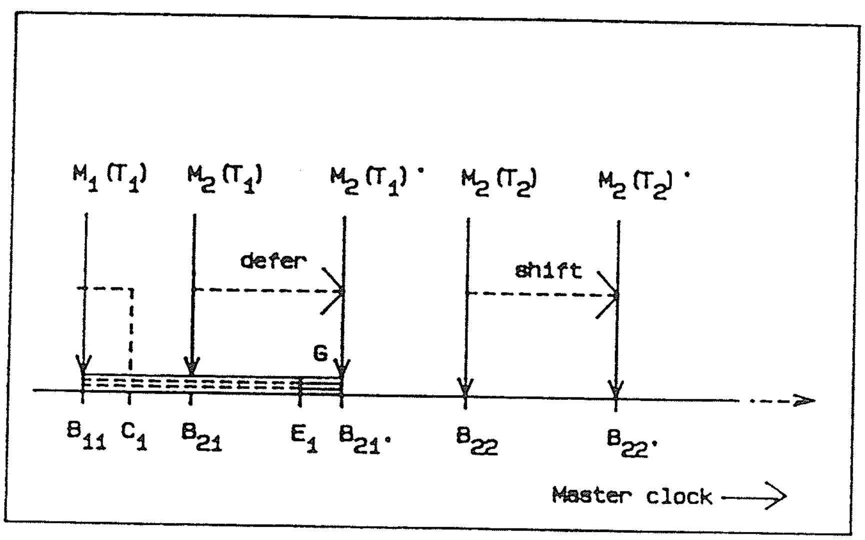


Figure 4.8 Deferred transmission.

Case 7: Collision - Multi-Packet.

Figure 4.9 shows a similar situation in Figure 4.5 but with a collision in the multi-packet message. The packet M1(T1) is attempted at position B11. Another packet M2(T1) is attempted at position B21, which is within the collision interval B11-C1. M1(T1) and M2(T1) collide with each other. The backoff procedures randomly move M1(T1) and M2(T1) into new positions, B11' and B21' respectively. Since M2(T1) is moved, M2(T2) has to be shifted to a new transmitted position from B22 to B22'. It should be noted that the time interval between B21 to B22 is the same as B21' to B22'.

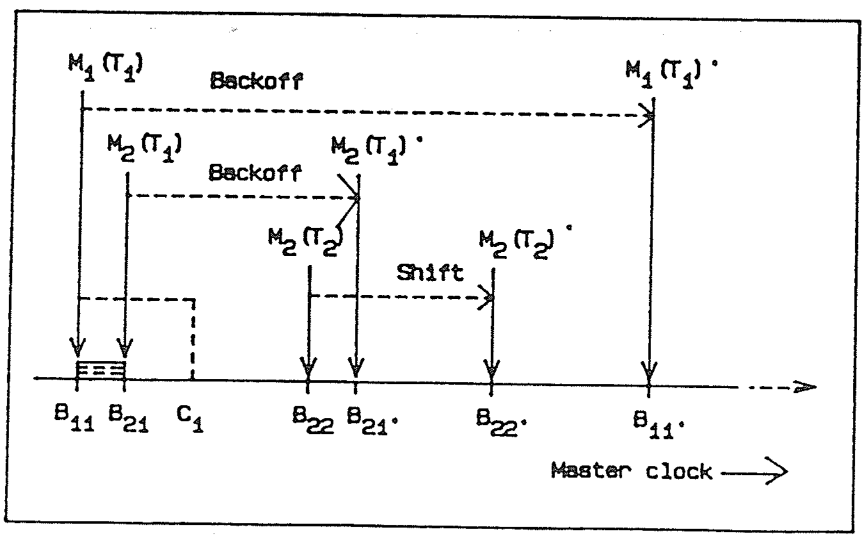


Figure 4.9 Collision for multi-packet message.

To this point, the network events of a single network for both single-packet and multi-packet messages has been described. Multiple network are also of interest. A multi-network is defined by two or more single networks connected together by bridges. Figure 4.10 shows the simplest case of a multi-network. Two single networks, 1 and 2, are connected by a bridge. Since a bridge is a non-broadcast link device, each network has it own set of independent events. If a packet is transmitted on network 1, and both the transmitting and the receiving nodes are on the network 1, this event will not have any effect on the network 2. Since the bridge recognizes the destination address of the transmitted packet, it will not forward and broadcast unless required. However, network events of one network will affect the activity of the other networks if the transmitting and receiving nodes are on different networks.

Cases 1 to 7 for the single network apply to the multi-network, respecting the broadcast nature of the bridge. However, there are two cases which are unique to a multi-network environment; successful transmissions on separate networks, and transmission through a bridge.

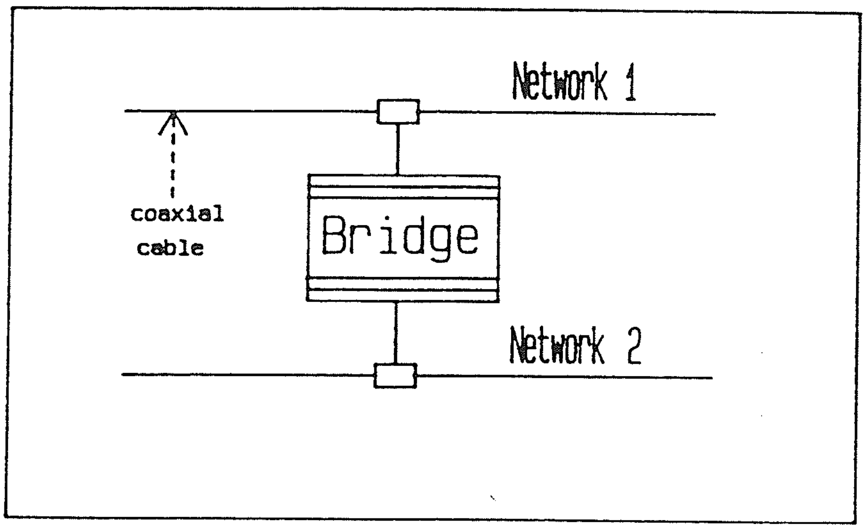


Figure 4.10 The configuration of multi-network.

Case 8 : Successful transmissions on separate networks.

Successful transmission for a message in a multi-network is shown in figure 4.11. N1(Mj(Ti)) refer to the ith packet of the jth message in network 1; N2(Ml(Tk) refer to the kth packet of the lth message in network 2. N1(Mj(Ti)) is attempted at position B1. N2(Ml(Tk) is attempted at position B2. It would appear that N2(Ml(Tk) is transmitted within the collision interval of N1(Mj(Ti)). In fact, no collision occurs because N2(Ml(Tk)) is on network 2 (using time clock2) while N1(Mj(Ti) is on network 1 (using time clock1).

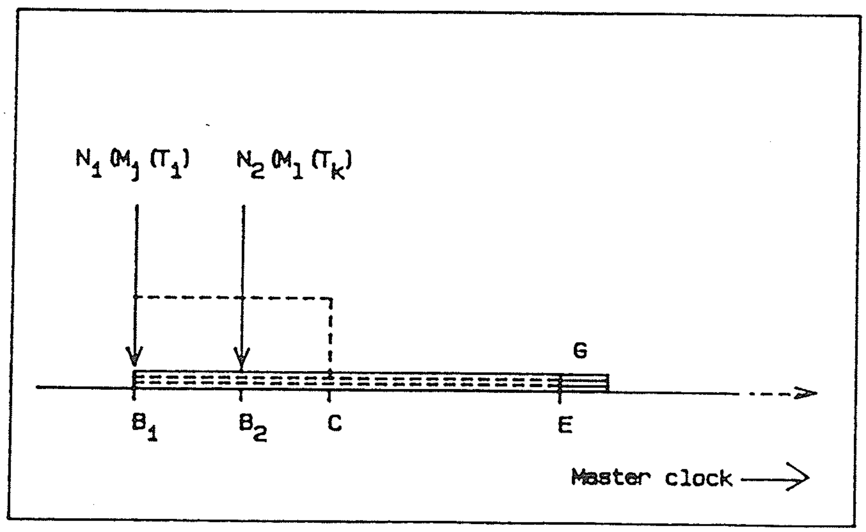


Figure 4.11 Successful transmission in a multi-network.

Case 9: Transmission through a Bridge.

A packet may transmit from one network to another network. In this case, the bridge serves as a path for the packet transfer. Figure 4.12a, 4.12b, and 4.12c show a packet transmitted from one network to another network. Packet N1(Mj(Ti)) is forwarded to network 2. N2(M1(Tk)) and N2(Mn(Tm)) are the network events in network 2. N1(Mj(Ti)) successfully passes its collision interval on its own network 1. The bridge buffers the packet N1(Mj(Ti)) and attempts to forward it to network 2 after a specified delay. If N1(Mj(Ti)) arrives on network 2 successfully, it will become N2(Mp(To)) the transmitted packet in network 2. However, if N1(Mj(Ti)) fails to forward to network 2 due to collision, the bridge will buffer it and attempt to forward it again. The bridge in effect becomes a second generator of the packet.

Figure 4.12a, 4.12, and 4.12c can be represented in a master clock instead of two time clocks. The master clock for a packet which transmitted from network 1 to network 2 is shown in figure 4.13a and 4.13b.

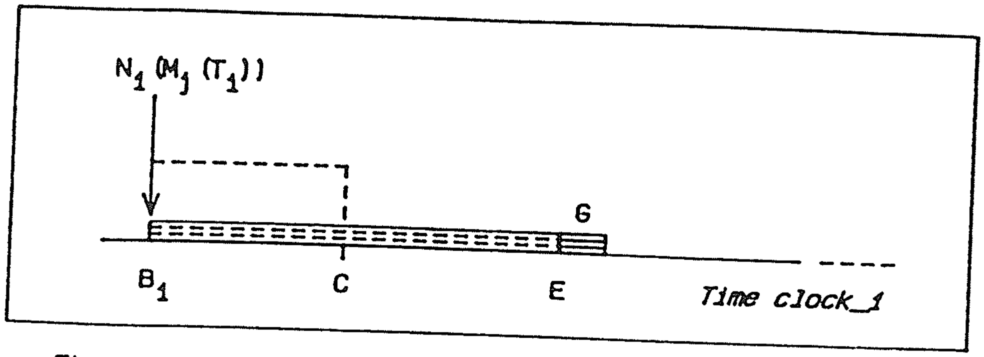


Figure 4.12a Transmission on Network 1 attempted to forward to Network 2.

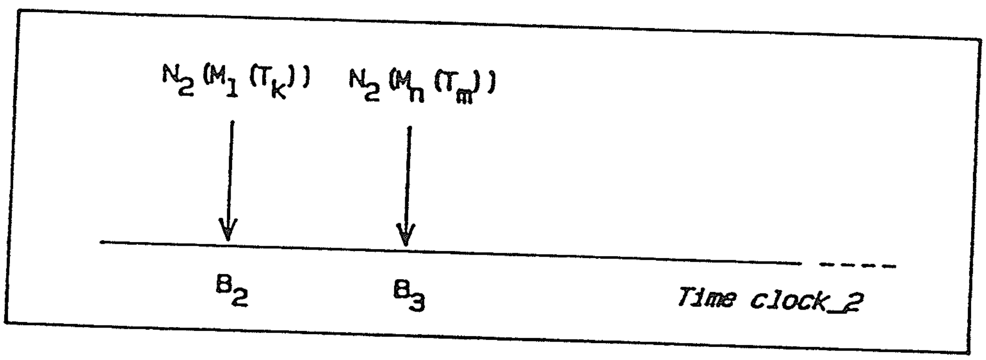


Figure 4.12b Transmissions originally in the network 2.

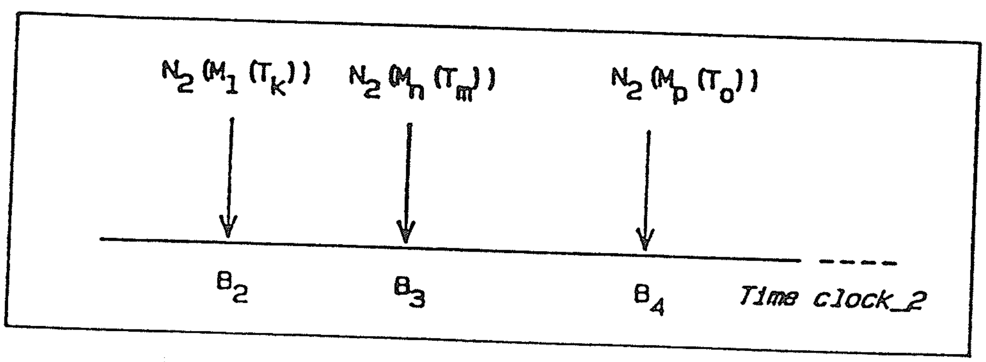


Figure 4.12c Transmission from Network 1 received on on Network 2.

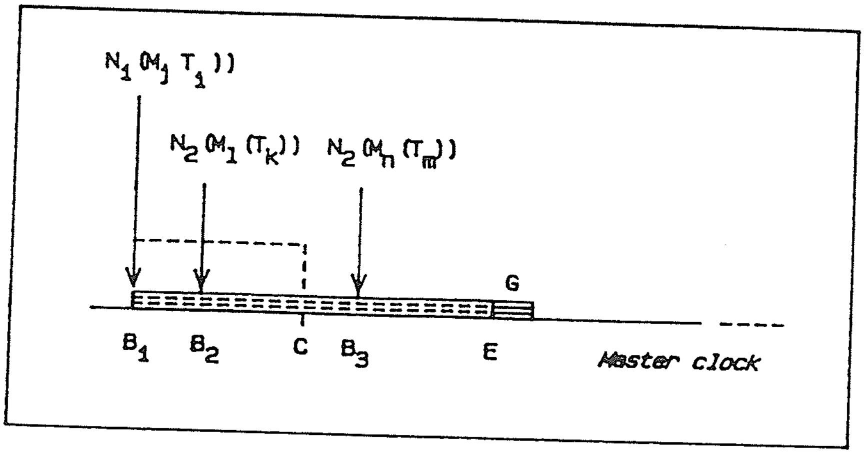


Figure 4.13a Packet from network 1 to network 2 through the bridge.

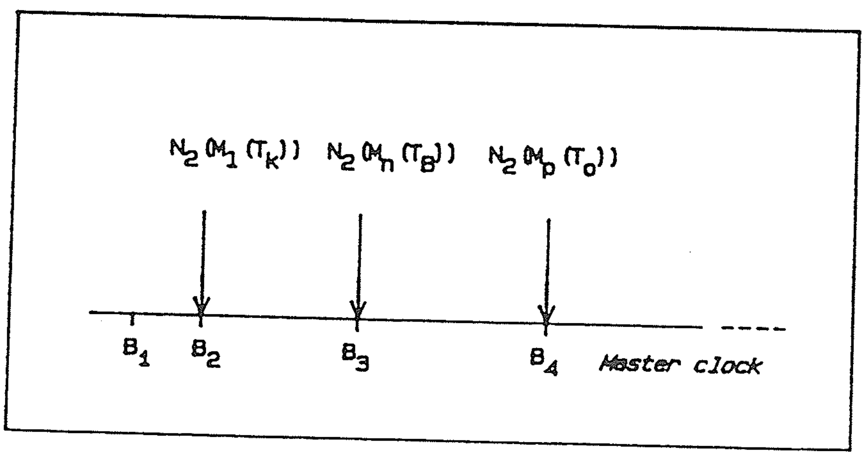


Figure 4.13b Packet originally from network 1 on network 2.

In order to specify the route for the messages to pass through a large multi-network, a multi-network path table is manually entered into the program. During program execution, the inter-network route for messages can be identified by utilizing the multi-network path table. An typical example of the network transmitted path table is given in Figure 4.19a at the end of this chapter.

In summary, the network events for multi-packet messages in multiple networks are accounted for in the simulation. During simulation, all the network events are randomly distributed in the master clock. Next, all the network events are resolved to the position on the master clock according to IEEE 802.3 and device bandwidth limitations; the network events may defer and delay to a further position in the clock if necessary. In order to check for the collisions, node to node distances are used to compute the collision windows. All the statistical network information such as backoff, deferrals, delays, attempted transmissions, successful transmissions, collisions are captured. Once the last event in the master clock is resolved, the simulation is completed.

4.3 COMPUTER PROGRAM

A computer program was developed based on the technique described in section 4.2. Different programming languages were compared in order to choose a suitable one. The flowcharts and modules given in the following subsections describe the program in detail. A user guide to the simulation is given in Appendix C.

4.3.1 Selection Of Programming Language And Facilities

The computer facility in Health Sciences Centre is limited to two super-mini-computers, VAX 8650 and VAX 11/785, and one micro-computer, Micro-VAX I. These computers are supported by a Virtual Memory Storage (VMS) operating system.

The programming techniques potentially available were concurrent programming, common interprocess event flag programming, and structured programming. Concurrent Pascal programming was introduced in IEEE CSMA/CD (1985) to explain the implementation of IEEE 802.3 CSMA/CD access method. However, IEEE CSMA/CD (1985) emphasized that the program was not intended to be executed by a computer. As well, concurrent Pascal was not available at the Health Sciences Centre. Common interprocess event flag programming on the VAX/VMS operating system was also considered. Interprocess event flags are status posting bits that can be set or cleared indicating the occurrence of an event (Deitel, 1984). If simultaneous execution of several processes is required, interprocess flags can be used to establish communication and to synchronize their activity (Digital, 1983). Thus, each process can represent the activity of a single network. Network events can be passed from one process (network) to other process (network) by synchronization of the event flags. In this research, synchronization of flags was also rejected due to the fact that timing is difficult to synchronize for complicated network events. The most reasonable choice was structured programming in which the whole program was divided into several subroutines and a mainline subroutine was used to control the execution of the other subroutines.

The advantages and disadvantages of various computer languages were compared in order to find the most suitable one for the requirements of the model. No special-purpose simulation language was available in Health Sciences Centre, such as GPSS, Simscript, and Simula described by Bratley et al (1983). The computer languages available were VAX Basic and VAX Macro assembly. VAX Macro was rejected because it is a non-structured language, not easy to verify, and very difficult to control a complex program. Therefore, VAX Basic was selected.

VAX-11 Basic is a structured and high-level programming language. It facilitated programming and controlling of the algorithm in section 4.2 and supported all operating system utilities. It also supports subroutine and procedural structures such as record definition.

4.3.2 Flowcharts

Before writing the detailed instructions for the computer program, flowcharts were developed. The flowcharts for the network simulation program are shown in Figure 4.14a to Figure 4.14f.

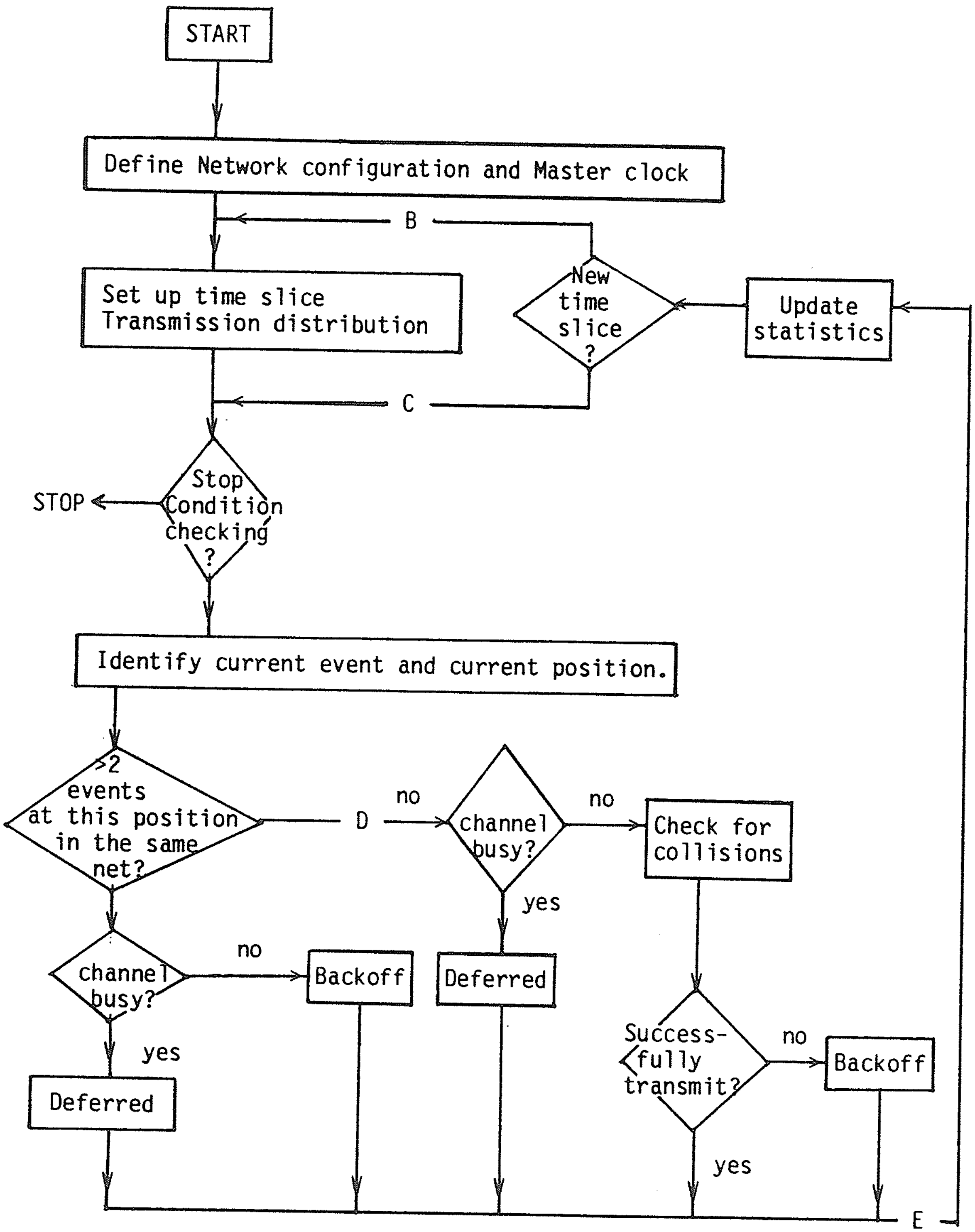


Figure 4.14a The flowchart of the computer program.

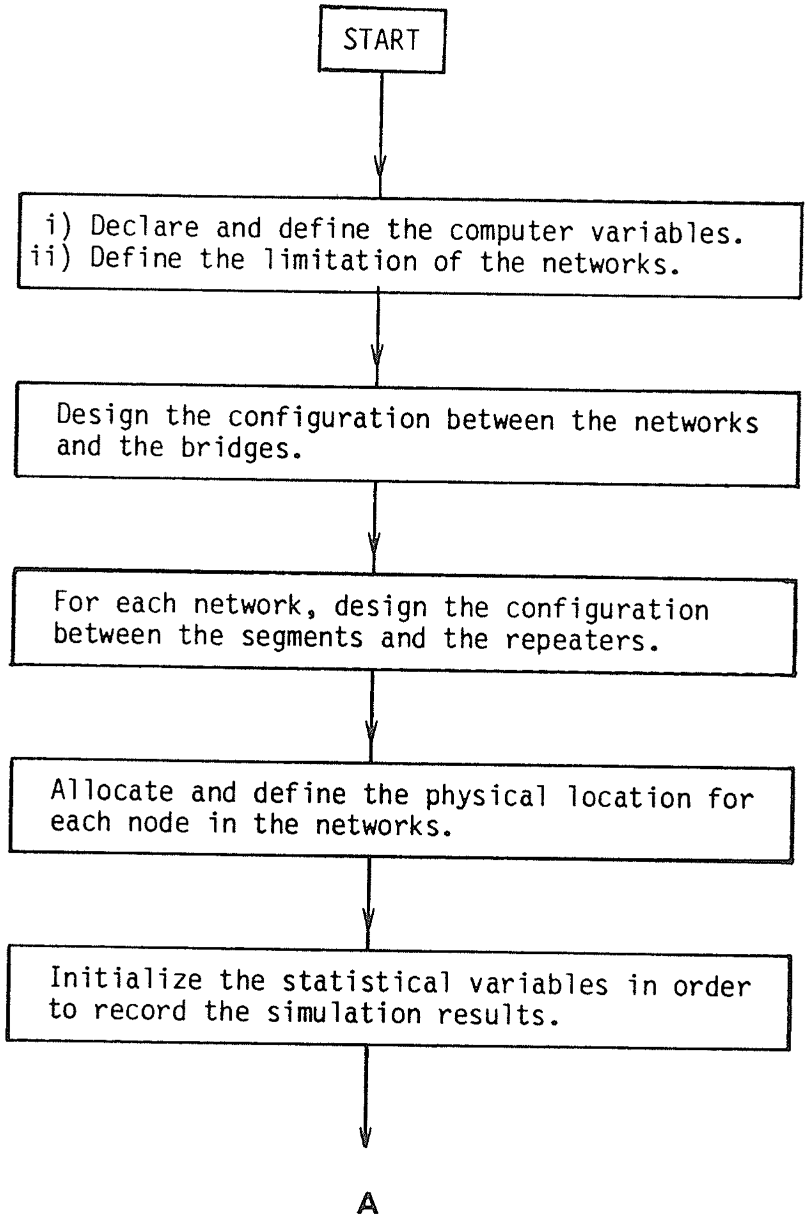


Figure 4.14b The flowchart of the computer program.

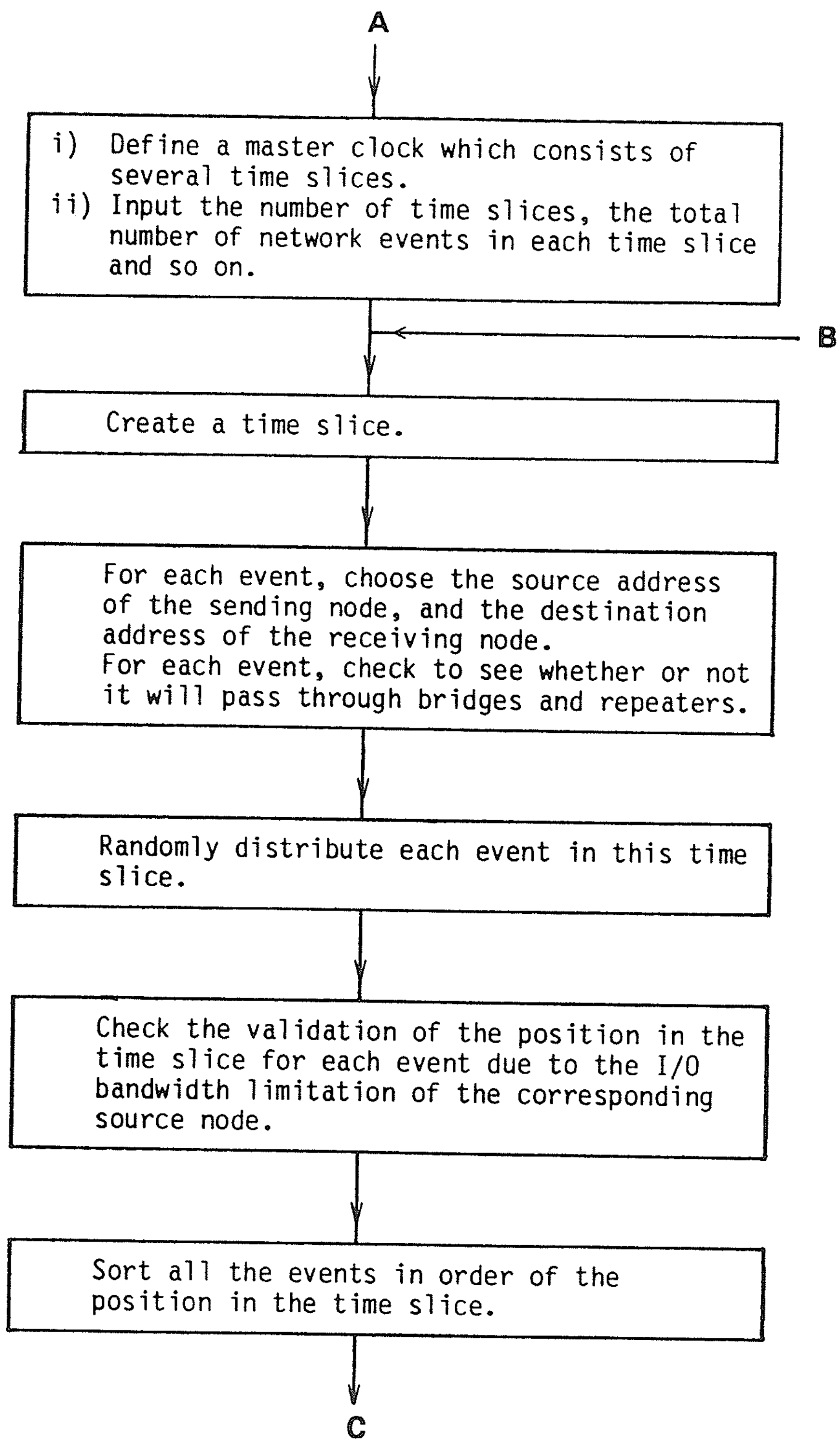


Figure 4.14c The flowchart of the computer program.

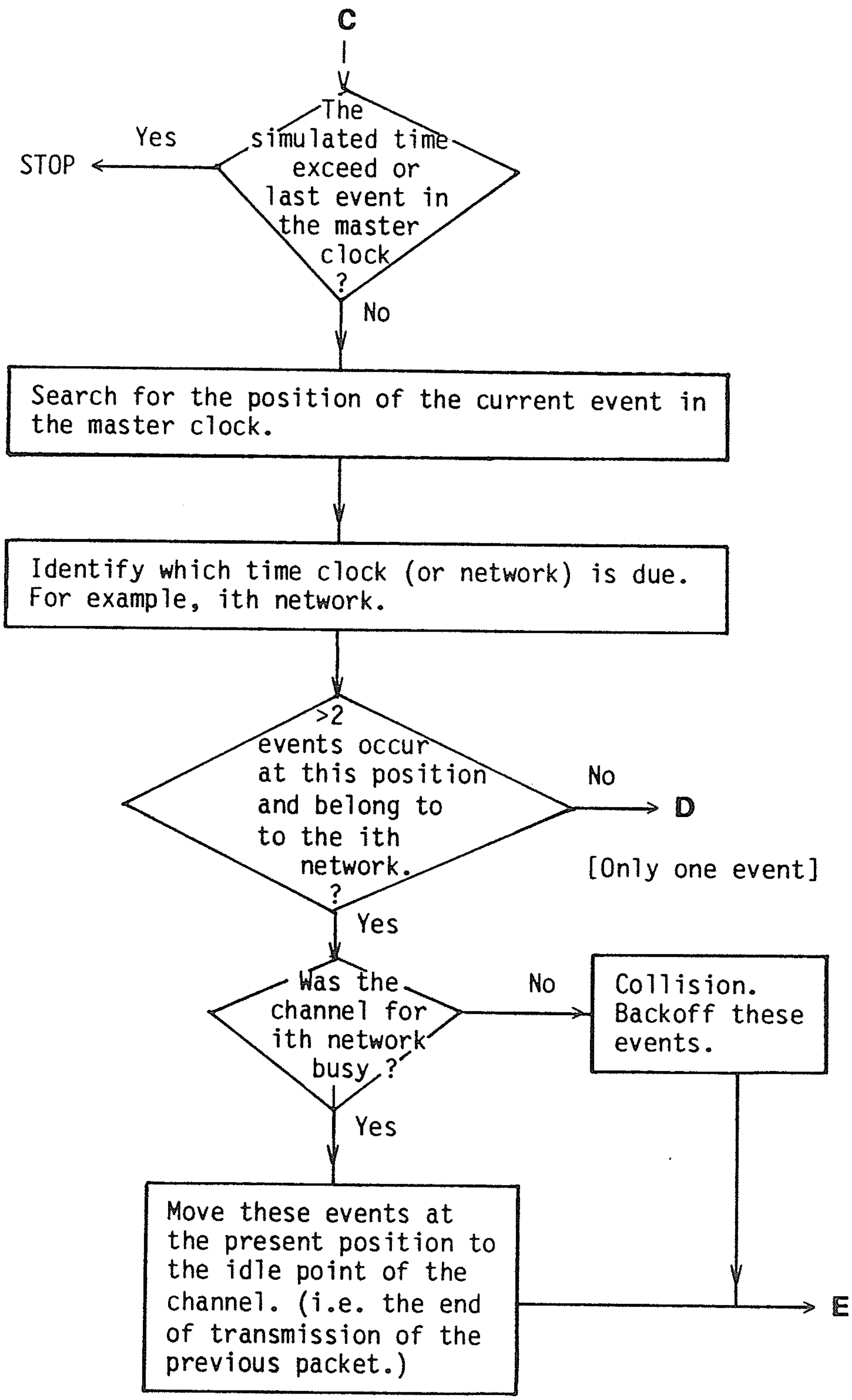


Figure 4.14d The flowchart of the computer program.

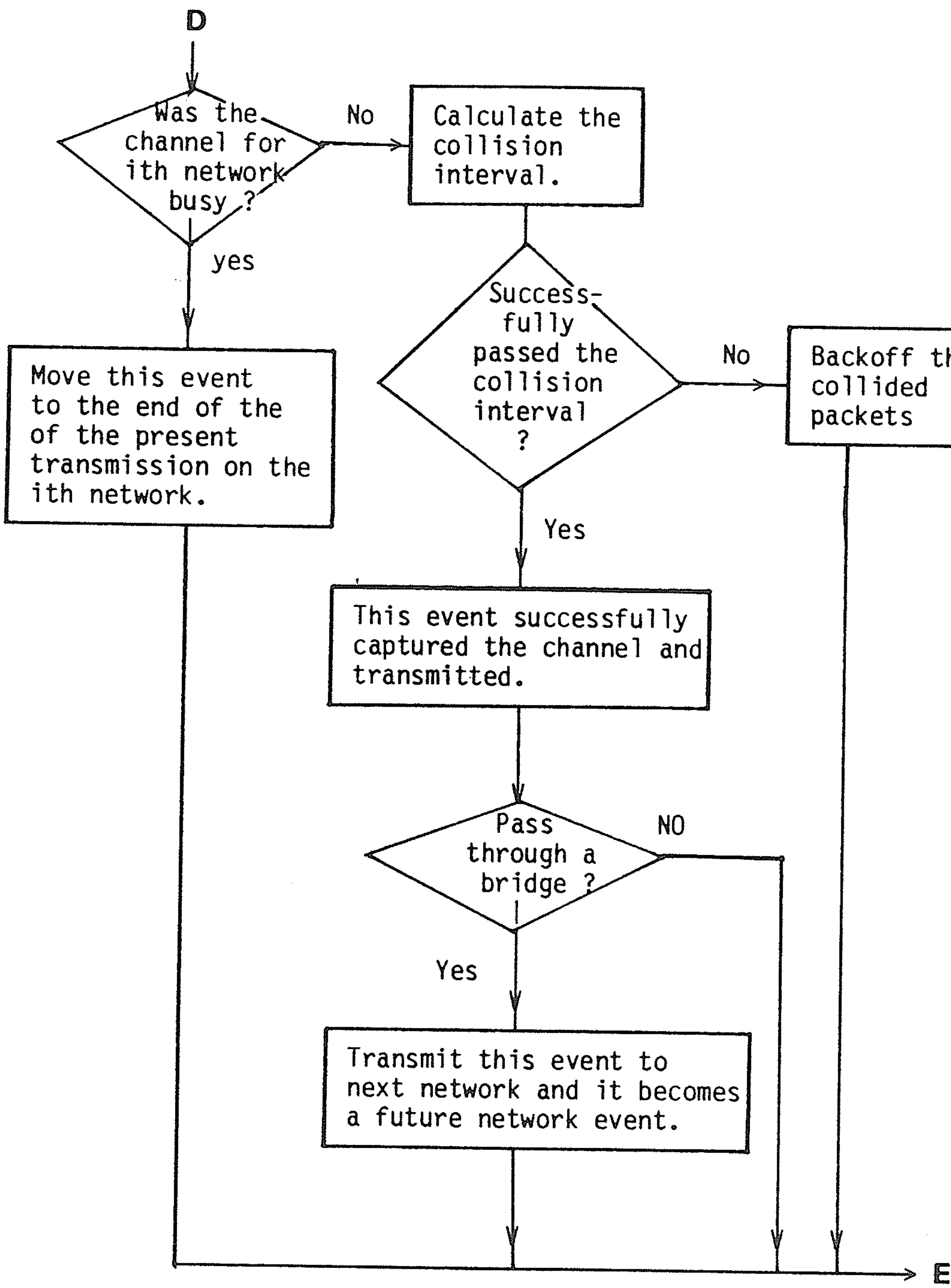


Figure 4.14e The flowchart of the computer program.

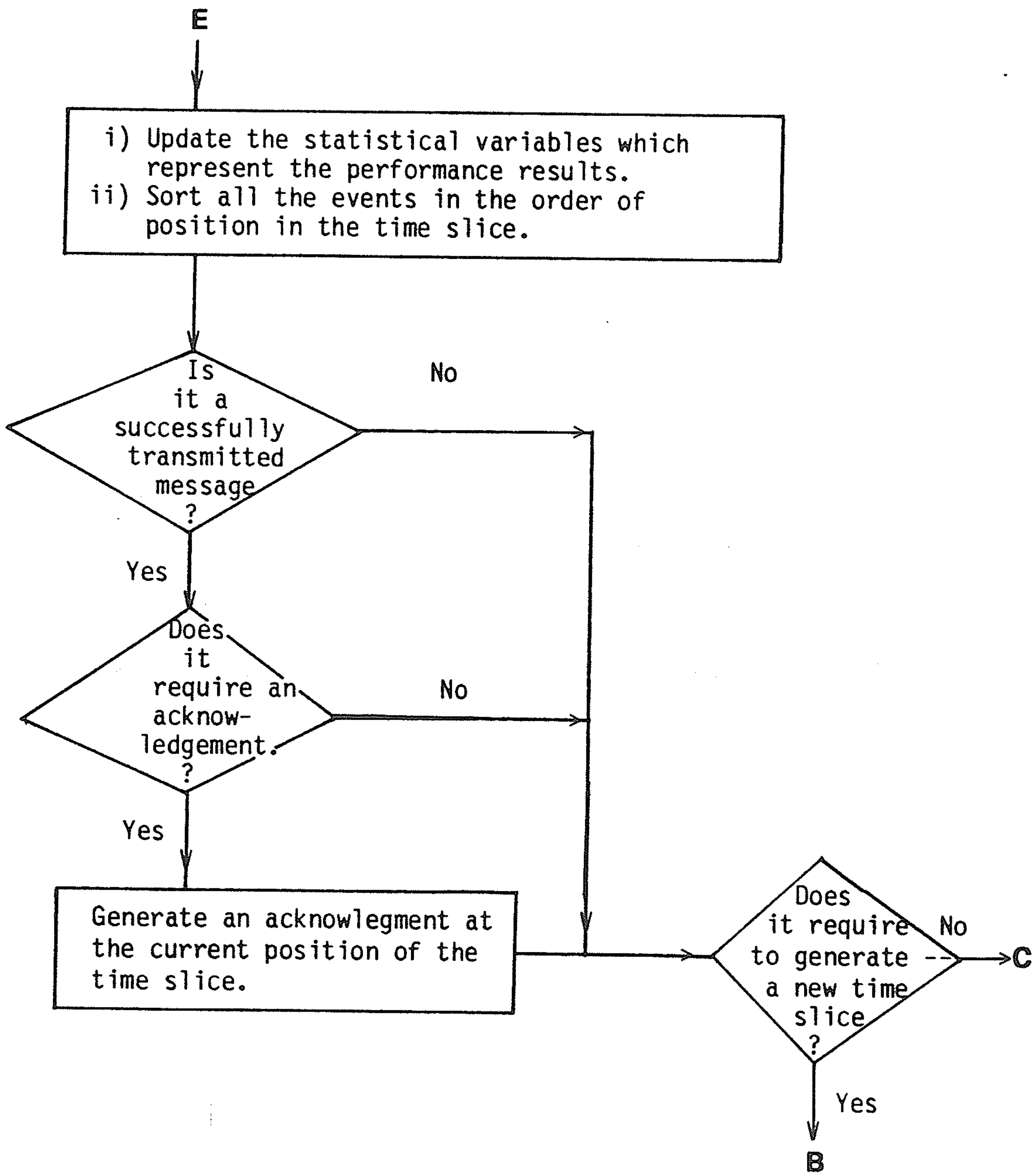


Figure 4.14f The flowchart of the computer program.

The simulation begins with a definition of the variables in an input data file. This is read by the program in order to construct the network configuration. The input data specifies the connections between the networks and the bridges, the connections between the repeaters and the segments within each network, and the physical location of each node.

Figure 4.14c is the flow diagram of the timing procedure. A master clock which consists of several time slices is defined. The number of bit times is assigned in each time slice. Network events occur at the beginning of any bit time. Then, the length of the time slice and the maximum number of network events in the time slice are assigned. For each network event, the source address (the address of the sending node) and the destination address (the address of the receiving node) are assigned from a uniformly random distribution. Special flags are set by the program for packets which must be transmitted through bridges or repeaters. The network events are then distributed randomly in the time slice. Once all the events are set up for this time slice, each network event is solved in the order of its position in this time slice.

Figure 4.14d, 4.14e, and 4.14f show the steps to resolve each event in the time slice. The time limit is first checked to determine whether the program ends or not. The current event in the time slice is searched and its source address (source network, source segment, source node) is identified. Then, the number of events which belong to the same source network are found, exactly at the bit time position of the time slice. If there is only one event in this case, the following steps will continue at step D in figure 4.14e. If there are more than 2 events, the channel is checked to see whether it is busy or idle. If the channel is busy, these events will defer to the end of transmission of the previous packet plus the interframe gap, and the program will continue to E in figure 4.14f. If the channel is busy, a collision is assumed to occur, and the backoff procedure is invoked, and execution is continued at E in figure 4.14f.

At D in figure 4.14e, only one network event has occurred at this position of the time slice. The channel is checked to determine if it is idle or busy. If the channel is busy, this event will be deferred to a farther idle point in the time slice, and execution will continue at E in figure 4.14f. Otherwise, the collision checking interval is examined and the collision window for each event in the checking interval is calculated. If any events occur inside the collision window, the colliding packets will backoff and continue at E in figure 4.14f. If all such events pass the respective collision windows, the channel is captured by this event. It is also determined whether this event passes the bridge or not. If so, it is placed in next network and continue at E in figure 4.14f.

At E in figure 4.14f, the statistics of the network performance are updated. The remaining events are resorted in the order of the position in the time slice. If the current message is successful, an acknowledgment message is sent back to the source address if the transmitted message so requested. If the position of any event exceeds the present time slice, execution is looped back to B in figure 4.14c. Otherwise, execution is looped back to C in figure 4.14d.

4.3.3 Subroutines

The computer program was divided into 48 subroutines. A mainline program is used to control the execution of the whole program by calling the subroutines. Only certain important subroutines are presented in this section.

A special routine is used to decide the number of transmissions in the time slice. The number of transmission in the time slice is determined by an input parameter called "number of sets of transmissions". One set of transmissions is equal to the number of nodes in the network. For example, if the network consists of 100 nodes, one set of transmissions provides 100 messages in the time slice and these transmissions are randomly assigned source and destination addresses.

The input and output capacity routine (I/O capacity) is executed after all the events are randomly assigned in the current time slice. The purpose of this routine is to ensure that the events randomly distributed in the time slice are valid with respect to the bandwidth limitation for each node. Therefore, messages which belong to the same source address should have adequate delays between messages. Figure 4.15 shows an example of bandwidth limitation for a node. The bandwidth limitation for Ethernet is 10 Mbps; the bandwidth limitation for a particular node (node A) is for example 100,000 bits per second. Two packets M1(T) and M2(T), which have the same source address, are initially attempted at positions B1 and B2 respectively.

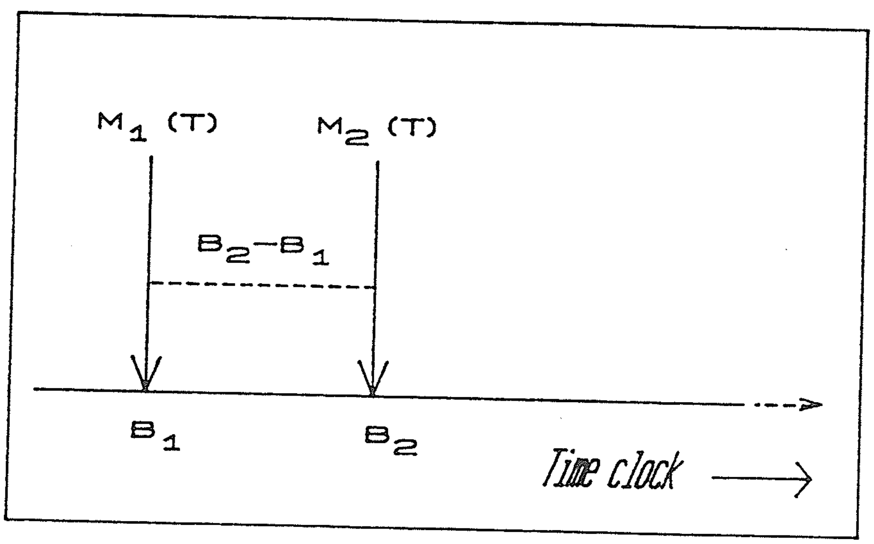


Figure 4.15 An example of bandwidth limitation.

If node A can transmit with a maximum of 100,000 bits in one second (10,000,000 bit times), node A should allow a maximum of X bit times to be transmitted in B2-B1 bit times, where X is given by

$$X = \frac{\text{Bandwidth of node A} * (B2-B1)}{10{,}000{,}000}$$

$$= \frac{100{,}000 * (B2-B1)}{10{,}000{,}000}$$

If the size of the message M1 is smaller than X bits, M2 is valid at B2. Otherwise, M2 is illegally distributed, and is deleted from the time slice. Initially, the illegal messages were repositioned but this required approximately ten times more execution time.

The propagation time (delay) subroutine is used to check whether an attempted transmission has passed the collision interval or not. Figure 4.16 shows the definition of checking interval and collision window on the master clock. Collision interval, as defined here, is the one way propagation time from the first sending node to next sending node. The checking interval covers the longest interval required to detect any collisions or deferrals, and is equal to 512 bit times. If there is any attempted transmission inside the checking interval, the collision interval between the current transmission and the attempted transmission is then found. If the attempted transmission is outside the collision interval, there is no collision. Otherwise, the current transmission and the later attempted transmission collid and the backoff algorithm is invoked.

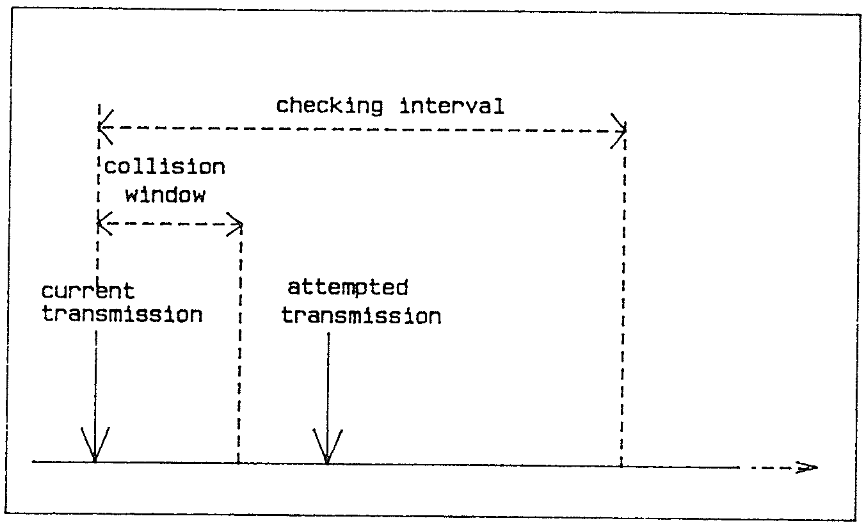


Figure 4.16 The definition of collision interval and checking interval.

In the configuration of networks in figure 4.17, node A attempts to transmit a packet to node B. During the checking interval, node C attempts to transmit a packet. The program first identifies the physical location of node A (DA) and then node C (DC). Thus, the physical distance between node A and node C can be calculated. This distance can be converted into bit time by using the propagation velocity of coaxial cable which is 0.77C (C=300,000 km/s) (IEEE CSMA/CD, 1985). The converted distance plus the delays in the repeater and bridge can be referred to as the collision interval of node A and node C. If the position of the attempted transmission of node C is outside the collision interval, there was no collision. Otherwise, the transmitted packet from node A will collid with the transmitted packet from node C.

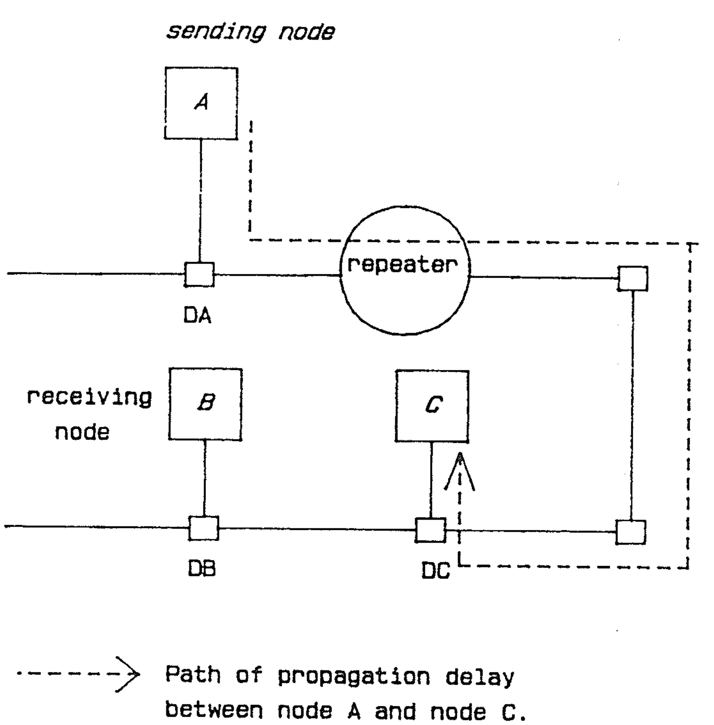


Figure 4.17 The path of propagation delay.

The stop condition of the program is tested by a subroutine called "time limit check". If there is no transmission in the current time slice, and the current time slice is the last time slice in the master clock, the program normally ends. If there is no transmission in the current time slice and the current time slice is not the last time slice in the master clock, a new time slice, and a new set of transmissions will be generated. The program will continue to resolve transmissions in a time slice until a normal end.

4.3.4 List Of Assumptions

1) Bandwidth limitation

   If there is any invalid transmission based upon the checking of I/O subroutine, the invalid transmission is deleted from the time slice. In reality, there should not be any invalid transmission if the I/O devices are functionally corrected.

2) Physical Delay in the devices

   The physical delays in the devices were not included in the simulation program. The typical example of one way delay for two transceivers and one repeater is 0.3 usec.

3) Backoff Position

   The position which backoff begins is always at the current position of the attempted transmission. This ignores the time period due to the propagation delay. The worst case of this time period for two nodes at the farther end of a 500m segment is 4.3 usec. The backoff retransmission interval is a multiple of 51.2 usec which is far exceeds the worst case value (4.3 usec) of this time period.

4) The jam signal

The jam signal which described in IEEE 802.3 was not included in this research. The jam signal is 32 bit times (3200 nsec).

4.3.5 Output Format And Evaluation Procedures

Once the program ends the results are recorded in four files. The first file, HSC.OUT2, records the configuration of the networks and the simulation results. The second file, HSC.OUT3, reports the step by step simulation procedure which can be used for error checking. The third file, HSC.OUT4, is the list of all the initial transmissions in the master clock. File HSC.OUT4 can be used when required to verify that the distribution of transmissions is both rare and random. The fourth file, HSC.OUT5, is used to report any programming errors which occur during the execution. Since files HSC.OUT3, HSC.OUT4, and HSC.OUT5 are used for program validation, only the file HSC.OUT2 is of interest in this subsection.

The example of the output file HSC.OUT2 is given in figure 4.19a to figure 4.19h. This output file is based on the configurations of the example multi-network in figure 4.18. Two networks, five nodes and one bridge are included in figure 4.18. L1 to L7 are the physical locations for each node.

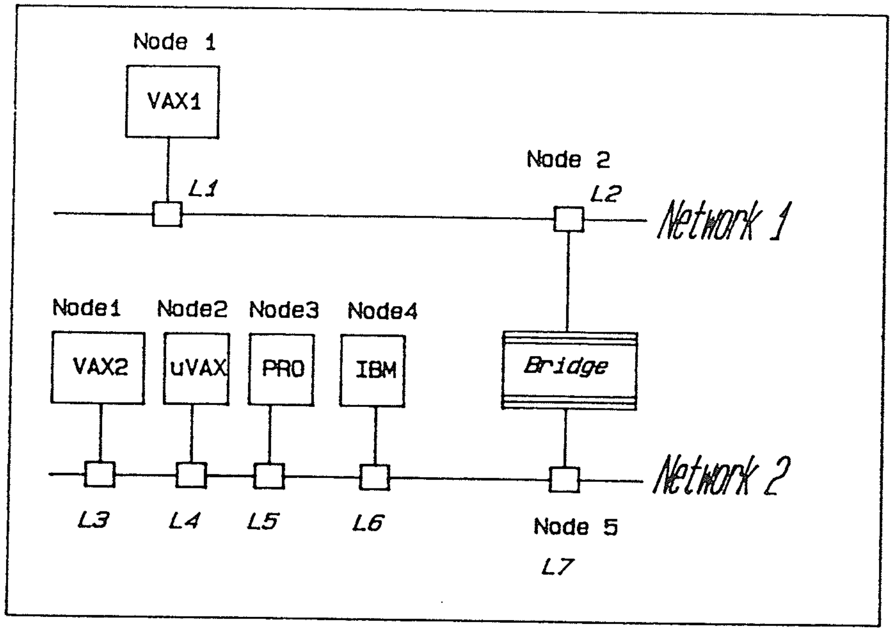


Figure 4.18 The network configuration used to illustrate the output file.

The output file HSC.OUT2 is divided into twelve sections. Section (i) defines the size of the overhead for each packet. The overhead can be specified by the user. In this case, the overhead was specified according to IEEE 802.3. Section (ii) shows the number of networks and bridges in this simulation. Section (iii) shows the paths for the transmission of a packets from one network to another network. Section (iv) defines the physical locations and connections of the bridges. The bridge is connected between network 1 (with node number 2, and physical location 100.0 meters) and network 2 (with node number 5, and physical location 100.0 meters). Section (v) defines the repeaters and segments connection within each networks. There was no repeater and only one coaxial segment in both network 1 and network

2. Section (vi) defines the transmission segment path for each network. Section (vii) shows the physical location and device type for each node in networks. Section (viii) defines the transmission probability for a packet to transmit from one network to another network. Also, the transmission probability for a packet to transmit within the same segment (the segment of the packet's source address) which was assigned. Section (ix) defines all the characteristics for each node such as the device name, the bandwidth, the number of packets for each message, the length of each packet, and acknowledgment requirement (reply flag). Basically, section (i) through section (ix) document the configuration of the networks.

The simulation results are summarized in sections (x), (xi), and (xii). Section (x) defines the size of the time slice, the number of time slices, the number of transmissions in each time slice, and the total simulation time.

Section (xi) shows the statistical results for each node. In the example, VAX2 (the node number 1, segment 1, and network 1) had 1658 successfully transmit messages, the total time for the message delay was 133280 bit times, the total time for transmitting messages which includes the message delays is 18,702,880 bit times. Therefore, the total time used by the data of the messages was 18,702,880-133,280 = 18,569,600 bit times. Each message may consist of several packets. Single-packet messages were used in this case. Therefore, the number of successfully transmitted packets was equal to the number of successfully transmitted messages. Neither single collision nor multiple collisions occurred in this case. Single collision is defined as one and only one collision occuring during the entire packet transmission. Multiple collision is defined as two or more collisions occurring during the entire packet transmission. An initial deferral occurs when a packet cannot acquire the channel at the initially assigned position due to a busy channel. There were 151 attempted packets, representing 151/3699 = 4.08% of the total attempted packets had to be deferred in this case. If a packet has 16 consecutive and unsuccessful attempts, an error will be reported. There was no errors in this case.

Section (xii) shows the summary of the simulation results. The number of transmitted messages is the total number of messages randomly distributed in the master clock initially. The total number of single collisions and multiple collisions occurring during transmission of packets are also recorded. The total non-idle time is the total time used by delays and data in the messages. Non-idle time is defined primarily for the purposes of calculating message delay. Also, the total time used by data only in the messages is reported. Finally, the time on the time clock of each network when all the transmissions were completed is recorded. The channel utilization for each network is the ratio of the total time used by data and the ending time in the master clock.

The system flowchart is given in figure 4.20 which shows the data flow among the various high-level modules.

```
THE LOCAL AREA NETWORK SIMULATION PROGRAM

The configuration of the local area network

The Packet (frame) consists of :
          Preamble field      =  56
          SFD field           =  8                        [i]
          Destination address =  48
          Source address      =  48
          Length field        =  16
          CRC field           =  32

The sum of transmit time for the above fields are 208 bit times

-------------------------------------------------------------

THE NETWORK CONNECTION:                                   [ii]

The number of networks in the Local Area Network = 2
The number of bridges  in the Local Area Network = 1
INPUT VALUE ACCEPTED, PROGRAM PROCEED.

-------------------------------------------------------------

THE NETWORK TRANSMITTED PATH TABLE:                       [iii]
 1  1  0  0  0  0  0  0
 1  2  0  0  0  0  0  0
 2  1  0  0  0  0  0  0
 2  2  0  0  0  0  0  0
-------------------------------------------------------------
                                                          [iv]
THE CONNECTION AND LOCATION OF BRIDGES:

Bridge    From     From     From    To       To        To
number     net  location  nodenum  net  location  nodenum
     1       1    100.00        2    2    100.00        5
```

Figure 4.19a Output format of the computer program.

```
THE REPEATER CONNECTIONS:
                                                        [v]
DATA ACCEPTED FOR
Network number          = 1
Number of local      repeater(s)   = 0
Number of remove     repeater(s)   = 0
Number of multiport  repeater(s)   = 0
Number of coaxial    segment(s)    = 1
Number of linked     segment(s)    = 0
Number of coaxial    thinwire(s)   = 0
TOTAL segments (coax. & thinwire)= 1

--------------------------------------------------------------

DATA ACCEPTED FOR
Network number          = 2
Number of local      repeater(s)   = 0
Number of remove     repeater(s)   = 0
Number of multiport  repeater(s)   = 0
Number of coaxial    segment(s)    = 1
Number of linked     segment(s)    = 0
Number of coaxial    thinwire(s)   = 0
TOTAL segments (coax. & thinwire)= 1

--------------------------------------------------------------


FOR NETWORK NUMBER  1

FOR NETWORK NUMBER  2
--------------------------------------------------------------

FOR NETWORK NUMBER  1                                  [vi]

FROM SEGMENT   PASS SEGMENT   TO SEGMENT   GO THROUGH REPEATERS
  NUMBER          NUMBER         NUMBER       FIRST      SECOND

    1               1              0          -----      -----

SEGMENT PATH TABLE FOR NETWORK  1 WAS ALLOCATED.
THERE (IS)ARE  1 SEGMENT(S) IN THE NETWORK  1.
```

Figure 4.19b Output format of the computer program.

```
FOR NETWORK NUMBER  2

FROM SEGMENT   PASS SEGMENT   TO SEGMENT   GO THROUGH REPEATERS
  NUMBER          NUMBER         NUMBER      FIRST       SECOND

     1               1              0        -----       -----

SEGMENT PATH TABLE FOR NETWORK  2 WAS ALLOCATED.
THERE (IS)ARE  1 SEGMENT(S) IN THE NETWORK  2.

--------------------------------------------------------------

NETWORK:  1  COAX :  1
NUMBER OF MAUs    :  2                                  [vii]

The physical location for each node:

20  100
The device type for each node:

VAX1 B
--------------------------------------------------------------
--------------------------------------------------------------



NETWORK:  2  COAX :  1
NUMBER OF MAUs    :  5

The physical location for each node:

20  30  40  50  100
The device type for each node:

VAX2 UVAX PRO  IBM  B
```

Figure 4.19c Output format of the computer program.

```
THE PROBABILITY OF NET TRANSMISSION:
 1  1  0  0                                               [viii]
 1  2  0  0
 2  1  0  0
 2  2  0  0
THE PROBABILITY OF TX WITHIN THE SAME SEGMENT IS  1
MAXNUM-INDICATORS = 2050
-------------------------------------------------------------
The device type            = VAX1
The Band width limit       = .2E+007
The number of packets      = 1                            [ix]
The assigned packet length = 160
The reply flag             = F

The device type            = VAX2
The Band width limit       = .2E+007
The number of packets      = 1
The assigned packet length = 10992
The reply flag             = T

The device type            = UVAX
The Band width limit       = 500000
The number of packets      = 1
The assigned packet length = 696
The reply flag             = F

The device type            = PRO
The Band width limit       = 450000
The number of packets      = 1
The assigned packet length = 160
The reply flag             = F

The device type            = IBM
The Band width limit       = 450000
The number of packets      = 1
The assigned packet length = 10992
The reply flag             = F
```

Figure 4.19d Output format of the computer program.

```
The device type              = B
The Band width limit         = 0
The number of packets        = 0
The assigned packet length = 0
The reply flag               = F

--------------------------------------------------------------
The sets of indicators              = 291.6
The reserve factor for set of tx = 0                        [x]
Initial number of transmissions  = 2041
Total number of transmissions    = 2041
Initial set up did not exceed the time slide boundary.
TIME SLIDE =   100000000
Initial set up did not exceed the time slide boundary.
TIME SLIDE =   100000000

Current number of indicators      =  0
The timeslice counter             =  1
Each timeslice ( in bit time)     =  100000000
The maximum number of timeslices=  1
The total simulation time        =  100000000

THE NETWORK NUMBER  1                                      [xi]
THE SEGMENT NUMBER  1

NETWORK= 1  SEGMENT= 1  NODE= 1
Number of successful transmit messages      =     1658
Total time for the messages delay           =     0
Total time used for messages and delay      =     610144
Total time used for messages transmissions =     610144
Number of successful transmit packets       =     1658
Number of multiple collisions (packets)     =     0
Number of collisions (packets)              =     0
Number of initial deferred transmissions    =     0
Number of errors                            =     0
```

Figure 4.19e Output format of the computer program.

```
NETWORK= 1  SEGMENT= 1  NODE= 2
This is a bridge node.
Number of successful transmit messages      =    0
Total time for the messages delay           =    0
Total time used for messages and delay      =    0
Total time used for messages transmissions  =    0
Number of successful transmit packets       =    0
Number of multiple collisions (packets)     =    0
Number of collisions (packets)              =    0
Number of initial deferred transmissions    =    0
Number of errors                            =    0

--------------------------------------------------------------

THE NETWORK NUMBER  2
THE SEGMENT NUMBER  1

NETWORK= 2  SEGMENT= 1  NODE= 1
Number of successful transmit messages      =    1658
Total time for the messages delay           =    133280
Total time used for messages and delay      =    18702880
Total time used for messages transmissions  =    .185696E+008
Number of successful transmit packets       =    1658
Number of multiple collisions (packets)     =    0
Number of collisions (packets)              =    0
Number of initial deferred transmissions    =    49
Number of errors                            =    0

--------------------------------------------------------------

NETWORK= 2  SEGMENT= 1  NODE= 2
Number of successful transmit messages      =    0
Total time for the messages delay           =    0
Total time used for messages and delay      =    0
Total time used for messages transmissions  =    0
Number of successful transmit packets       =    0
Number of multiple collisions (packets)     =    0
Number of collisions (packets)              =    0
Number of initial deferred transmissions    =    0
Number of errors                            =    0
```

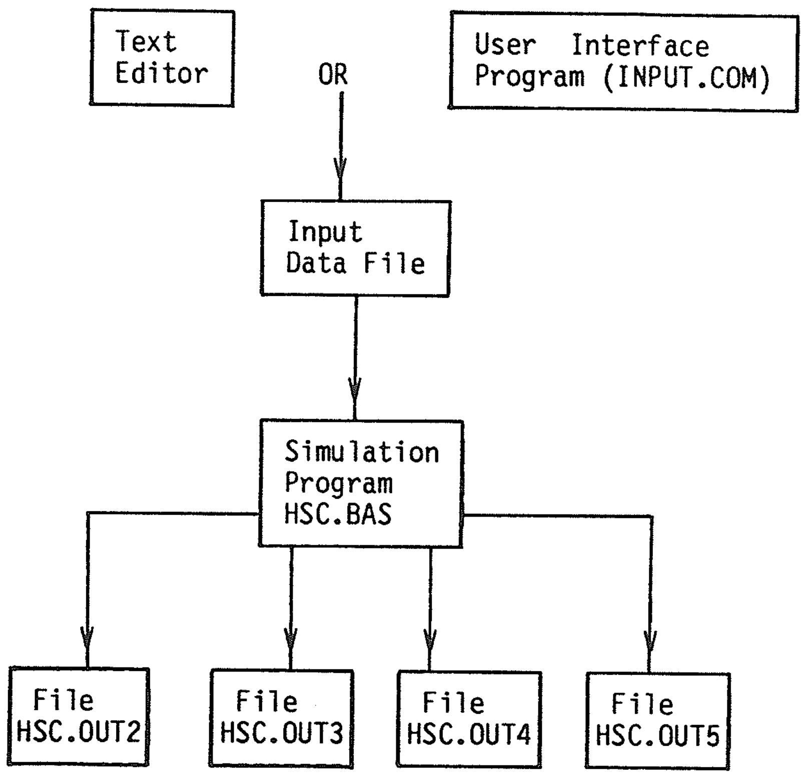
Figure 4.19f Output format of the computer program.

```
NETWORK= 2  SEGMENT= 1  NODE= 3
Number of successful transmit messages      =     0
Total time for the messages delay           =     0
Total time used for messages and delay      =     0
Total time used for messages transmissions  =     0
Number of successful transmit packets       =     0
Number of multiple collisions (packets)     =     0
Number of collisions (packets)              =     0
Number of initial deferred transmissions    =     0
Number of errors                            =     0

-------------------------------------------------------------

NETWORK= 2  SEGMENT= 1  NODE= 4
Number of successful transmit messages      =     383
Total time for the messages delay           =     550026
Total time used for messages and delay      =     4839626
Total time used for messages transmissions  =     .42896E+007
Number of successful transmit packets       =     383
Number of multiple collisions (packets)     =     0
Number of collisions (packets)              =     0
Number of initial deferred transmissions    =     102
Number of errors                            =     0

-------------------------------------------------------------

NETWORK= 2  SEGMENT= 1  NODE= 5
This is a bridge node.
Number of successful transmit messages      =     0
Total time for the messages delay           =     0
Total time used for messages and delay      =     0
Total time used for messages transmissions  =     0
Number of successful transmit packets       =     0
Number of multiple collisions (packets)     =     0
Number of collisions (packets)              =     0
Number of initial deferred transmissions    =     0
Number of errors                            =     0
```

Figure 4.19g Output format of the computer program.

```
***** S U M M A R Y ****
                                                        [xii]

STATISTICS OF NODES FOR NETWORK 1 :

The number of transmitted messages =  1658
The number of initially deferred   =  0
The number of single   collisions  =  0
The number of multiple collisions  =  0
The number of errors               =  0
The total non-idle time in channel =  610144
The total time used for messages   =  610144
The ending time in the master clock=  99907320
The channel utilization            =  0.0061

-------------------------------------------------------------

STATISTICS OF NODES FOR NETWORK 2 :

The number of transmitted messages =  2041
The number of initially deferred   =  151
The number of single   collisions  =  0
The number of multiple collisions  =  0
The number of errors               =  0
The total non-idle time in channel =  23542506
The total time used for messages   =  22859200
The ending time in the master clock=  100010585
The channel utilization            =  0.2286


NORMAL END OF THE PROGRAM.


CPU     TIME=% 65363.3  sec
Connect TIME=1505.000 sec
Current TIME= 10:18 PM
Date        = 12-Jun-87
```

Figure 4.19h Output format of the computer program.



HSC.OUT2 - the configuration of the networks.
- the simulation results.

HSC.OUT3 - the step by step simulation procedure.
- used for error checking.

HSC.OUT4 - the list of the initial transmissions.

HSC.OUT5 - the programming errors trap.

Figure 4.20 The system flowchart

CHAPTER 5

VALIDATION

It is desirable to have as accurate results as possible for the network simulation. Therefore, validation is required to check for the accuracy and precision of the program and the model. The accuracy and precision must be characterized before the simulation can be used with confidence to study different models. The goal of this chapter is to describe the accuracy and precision of the simulation.

5.1 PROGRAMMING VALIDATION

The simulation program was divided into 48 subroutines. Each subroutine was checked to verify that it produced reasonable output for all possible inputs. The program was also checked by altering one input parameter at a time while keeping all others fixed. It was found that all outputs were as expected, based upon a wide variety of different inputs such as packet length, time slice, acknowledgment packet, bandwidth limitation, and physical location of nodes.

During the testing of the computer program, two phenomena were found which provided additional validation for the program. The first phenomenon was the Poisson distribution of the transmitted messages in a lightly loaded network. The second phenomenon was the input and output bandwidth limitation.

It is known that when an event is both rare and random, the number of events are Poisson in distribution (Bratley et al, 1983). In the simulation program, if the transmissions are rare and randomly generated in the time slice, the transmitted messages should Poisson. This was confirmed as described in Appendix D.

The bandwidth limitation (I/O subroutine) was tested by using evenly spaced transmissions in a single network with single-packet messages. Figure 5.1 shows the transmissions in the time slice for the I/O checking. The time interval between packets was made only just large enough for a successful transmission. If the entire program is functionally correct, it should not have recorded any collisions. During the simulation, all messages, M1 to Mi, were transmitted successfully. No collisions or errors were found. When the interval was decreased, collisions occurred, supporting the validity of the collision algorithm.

5.2 COMPARISON WITH LITERATURE

In literature review, Chapter 3, no single paper was found which described exactly the IEEE 802.3 CSMA/CD. There were several limiting assumptions in each article which made it difficult for comparison with the simulation method. Therefore, no detailed reference could be used for the validation of the simulation method.

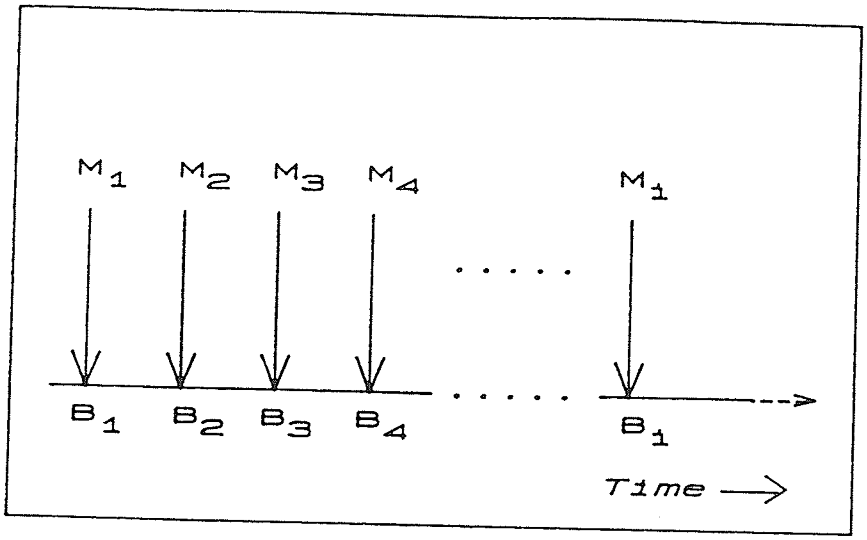


Figure 5.1 Transmissions in the time slice for the I/O checking.

5.3 ACTUAL EXPERIMENT

An experiment was designed to verify the accuracy and precision of the simulation. Figure 5.2 shows the flow diagram of this experiment. The three subsequent sections describe the methods, the results and the discussion of this experiment.

The method section describes the procedures of this experiment. The experimental network was first defined and the network traffic was generated. The statistics of the experimental network were collected by direct measurement. Then, the same amount of network traffic was input to the simulation and the corresponding network statistics were recorded. The results section compares the statistics between the simulation and the measurements of the experimental network. A discussion is provided at the end to summarize the accuracy of the simulation.

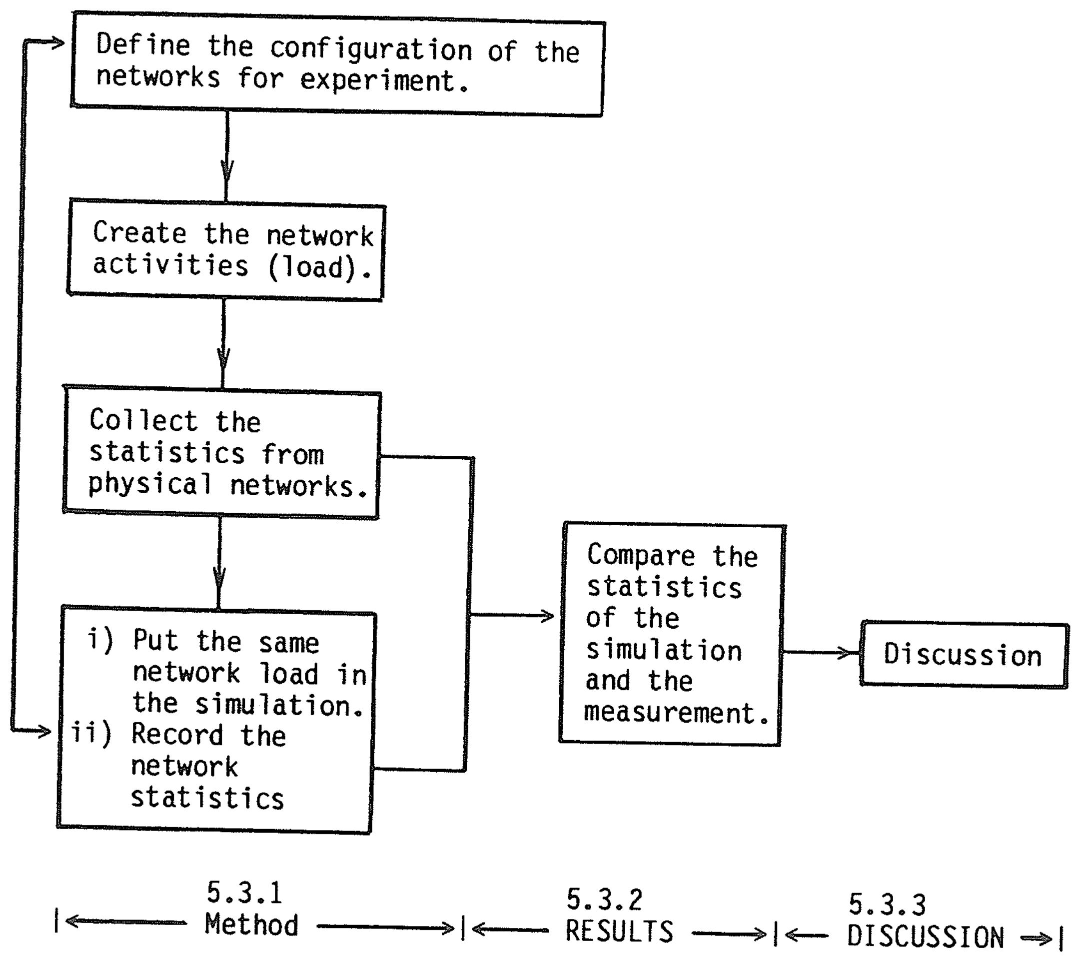


Figure 5.2 The flow diagram of the actual experiment

5.3.1 Method

The experimental method is described by first defining the experimental network. Next, the network load is described followed by an illustration of the statistical procedures. Finally, simulation of the network is described.

5.3.1.1 The Experimental Network

The construction of a physical network was limited to the available computer devices in the Health Sciences Centre. By trial and error, it was found that the network configuration depicted in Figure 5.3 generated the maximum traffic (offered load). The configuration consisted of two networks, network 1 and network 2. Six different devices were attached to the networks. The physical location of each device is given in Figure 5.3. These devices were:

VAX1 = super-mini-computer (VAX 785)

VAX2 = super-mini-computer (VAX 8650)

UVAX = mini-computer (Micro-VAX I)

IBM = micro-computer (IBM AT)

Bridge01 = bridge (LAN Bridge 100)

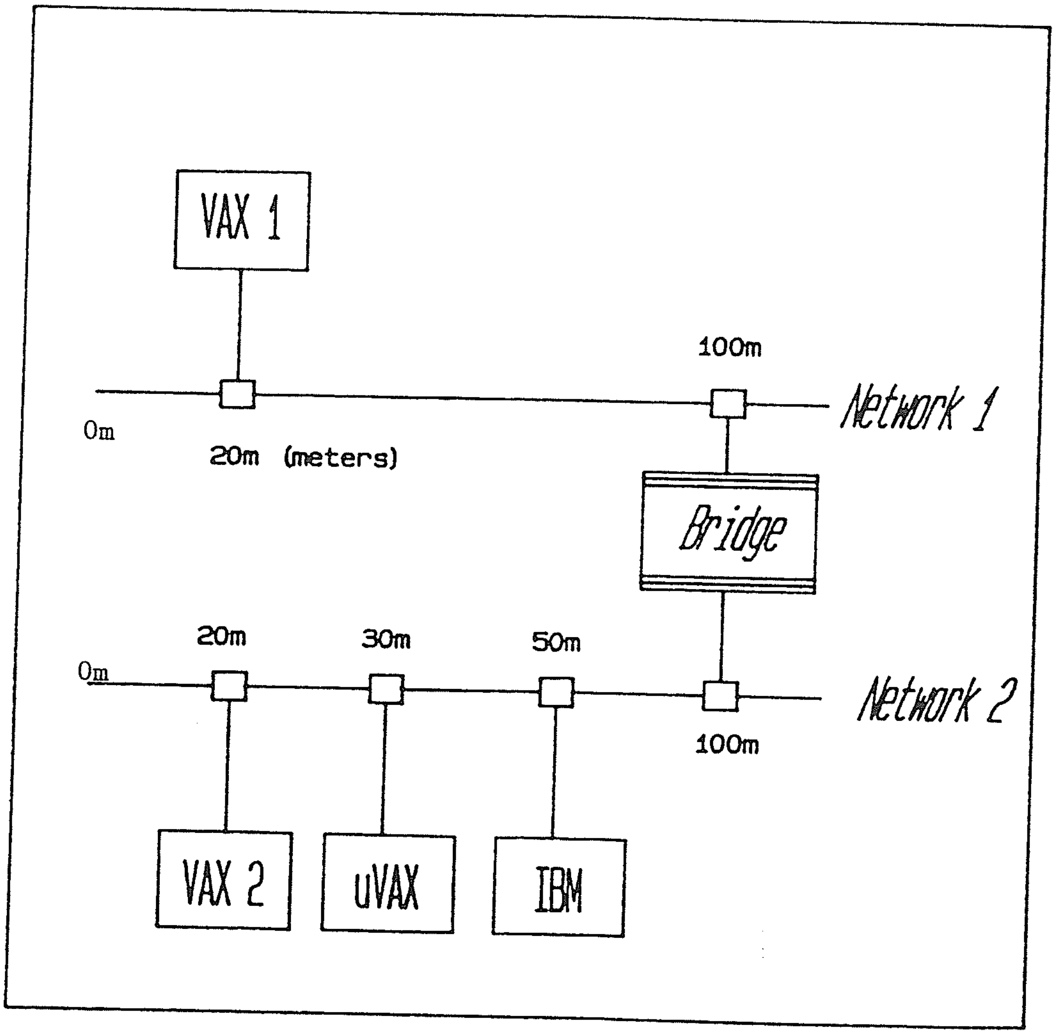


Figure 5.3 Configuration of the experimental network.

5.3.1.2 The Network Load

Once the configuration of the physical network was defined, the network traffic was generated by a utility called DECnet Test Sender/ DECnet Test Receiver (DTS/DTR). DECnet is a family of software and hardware communications products which enable the operating systems of Digital Equipment Corporation (DEC) to participate in a network environment. DTS and DTR are the DECnet transmitter and receiver test programs. There are four basic network tests provided by DTS and DTR. Only the "Data Test" utility was employed in this study. Data Test causes a node to continuously transmit messages with a specified message size to another node in the network. For example, the following data test command was executed in node VAX2:

```
DATA/PRINT/NODE=VAX1/SIZE=1800/SECONDS=30
```

Node VAX2 continuously transmitted messages to node VAX1 for a period of 30 seconds. The message size was specified as 1800 bytes per message. Messages may consist of multi-packets in the physical networks, depending upon their size. By trial and error, it was found that the network was predefined to allow a maximum size of packet (1498 bytes) to be transmitted on the channel. A message with 1800 bytes broke down into two packets. The first packet was 1498 bytes and the second packet was 302 bytes. A typical example of the DTS/DTR output format is shown in Table 5.1.

```
DTS Version 2.00 initiated on  8-JUN-1987 17:07:07.41
      DATA/NODENAME=VAX1/PRINT/SIZE=1800/SEC=30
%DTS-S-NORMAL, normal successful completion

Test parameters:
   Test duration (sec)  30
   Target nodename      "VAX1"
   Line speed (baud)    1000000
   Message size (bytes) 1800

Summary statistics:
   Total messages XMIT  2281    RECV   0
   Total bytes XMIT     4105800
   Messages per second  76.0
   Bytes per second     136860
   Line thruput (baud)  1094880
   % Line utilization   109.4

DTS terminated on  8-JUN-1987 17:07:38.90
```

Table 5.1 The output format of DTS/DTR Data Test.

Different sizes of messages were tried in the range of 100 bytes to 4096 bytes. It was found that message with 1400 bytes allowed the node to offer a maximum amount of traffic to the channel. Therefore, in the network traffic generation, single-packet messages with 1400 bytes were used. Appendix E shows the DTS results of different messages size.

The traffic load of the experiment was set up by the following two DTS command:

i) DATA/NODENAME=VAX1/PRINT/SIZE=1400/SEC=180 (executed in VAX2)

ii) DATA/NODENAME=UVAX/PRINT/SIZE=1400/SEC=180 (executed in IBM)

DTS Data Test (i) requested the node VAX2 to continuously send messages to the node VAX1; DTS Data Test (ii) requested the node IBM to send messages to UVAX. Both data tests executed simultaneously and sent out messages with a size of 1400 bytes. Figure 5.4 shows the path of the data sent from the nodes.

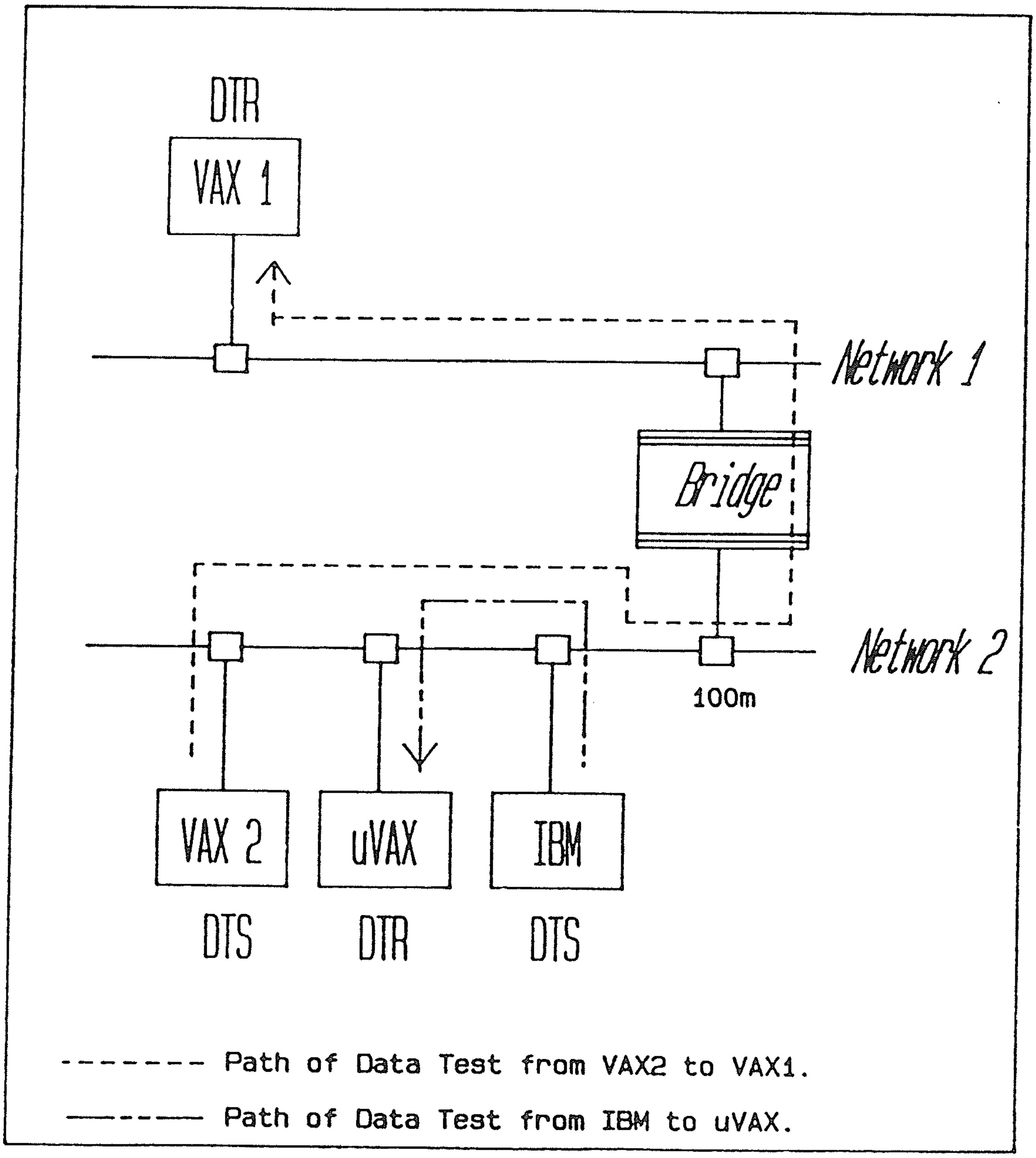


Figure 5.4 Path of the Data Tests.

5.3.1.3 Statistics From Physical Network

Once DTS generated the traffic load, the statistics from the physical network were collected. The utilities available in Health Sciences Centre for collecting networks statistics are:

i) Network Control Program (NCP),
ii) Remote Bridge Management Software (RBMS),
iii) 911 Monitor.

Network Control Program (NCP) is a DECnet-VAX utility that accepts terminal commands to configure, control, monitor and test a DECnet network (Digital NCP, 1986). The NCP command, SHOW LINE COUNTER, was used to show the statistics of the line (physical channel) in the network. A typical example of the NCP output format is shown in Table 5.2. The definition of NCP counters can be found in Digital NCP (1986). The "bytes received counter" shows the total number of bytes received over the line while the "bytes sent counter" shows the total number of bytes sent over the line. The "data blocks received/sent counter" shows the number of packets received/sent over the line. The "single collisions counter" indicates the total number of times that a frame was successfully transmitted on the second attempt. The "multiple collisions" counter indicates the total number of times that a frame was successfully transmitted only on the third or later attempt. The initially deferred counter indicates the total number of times that a frame transmission was deferred on its first transmission attempt. The "send failures" counter indicates the total number of times a transmit attempt failed. This type of failure includes the excessive collisions, carrier check failed, short circuit, open circuit, frame too long, and remote failure to defer. The "receive failures" counter indicates the total number of blocks received with some data error such as block check error, framing error, and frame too long.

```
Known Line Counters as of  8-JUN-1987 18:51:46

Line = UNA-0

         36628  Seconds since last zeroed
        702213  Data blocks received
          6894  Multicast blocks received
             0  Receive failure
      36946271  Bytes received
        419930  Multicast bytes received
           217  Data overrun
             0  Local buffer errors
        970601  Data blocks sent
          6292  Multicast blocks sent
            19  Blocks sent, multiple collision
            26  Blocks sent, single collision
           404  Blocks sent, initially deferred
      74175149  Bytes sent
        569229  Multicast bytes sent
             0  Send failure
            16  Collision detect check failure
           655  Unrecognized frame destination
            18  System buffer unavailable
             0  User buffer unavailable
```

Table 5.2 The output format of NCP Line counter.

Remote Bridge Management Software (RBMS) was used to monitor and control the bridges in the network. The monitoring feature of the LAN Bridge 100 was used to gain additional performance measurements. The commands used for collecting the statistics in the bridge are:

a) $ RUN SYS$SYSTEM:NMS_RBMS$BCP

b) USE DATABASE BRIDGE bridge-name KNOWN LINE

c) SHOW COUNTER

The DCL command (a) is used to invoke the RBMS. The RBMS command (b) causes all subsequent commands to use the defined lines of the bridge 'bridge-name'. The last command (c) shows the statistics of the defined lines in the bridge. A typical example of the RBMS output format is shown in Table 5.3.

Three statistical counters are defined as following (Digital RBMS, 1986):

Filtered frame : The number of frames received but not forwarded by the bridge.

Frames : Transmitted - all frames transmitted by the bridge.

Received - all frames received by the bridge, with the exception of bad frames and received frames lost.

By trial and error, it was found that the term "frame" in the RBMS counter is the term "data block" in the NCP line counter.

```
RBMS>
Line counters for Line 1  as of     8-JUN-1987 18:20:39
Bridge   BRIDGE01,   Address   08-00-2B-03-FC-51

Bridge seconds:                                   7815243
Invalid protocol messages:                              0
Filtered frames:                                 12658069
Bad frames:                                            36
Lifetime exceeded frames:                               0
No forwarding entry frames:                             0
Forwarding transition count:                            1
Collision Presence Test errors:                         0

                         Transmitted                    Received

Frames:                     57119909                    64374234
Bridge frames:               6184066                           8
Frames lost:                       0                           0


Line counters for Line 2  as of     8-JUN-1987 18:20:41
Bridge   BRIDGE01,   Address   08-00-2B-03-FC-51

Bridge seconds:                                   7815246
Invalid protocol messages:                              4
Filtered frames:                                 42195015
Bad frames:                                            83
Lifetime exceeded frames:                               0
No forwarding entry frames:                             0
Forwarding transition count:                            1
Collision Presence Test errors:                         0

                         Transmitted                    Received

Frames:                     70970755                   108026963
Bridge frames:               7827454                     1644629
Frames lost:                       0                           0
```

Table 5.3 The output format of RBMS Line counter.

The 911 Monitor is an Ethernet monitor which consists of hardware and software components. The hardware is a bridge (LAN Bridge 100) which is attached to the network, and the monitoring software is installed inside the bridge and a VAX host. The monitor collects the statistics from the network. It then generates an Ethernet packet, which contains the network statistics for display of the results at the host VAX computer. The 911 Monitor used in the experiments was a prototype device; it is now commercially available as the "LAN Traffic Monitor" by DEC.

Figure 5.5 shows the installation of the 911 monitor in the network. The configuration consisted of two networks, the 911 Monitor only collected the statistics from the network 2.

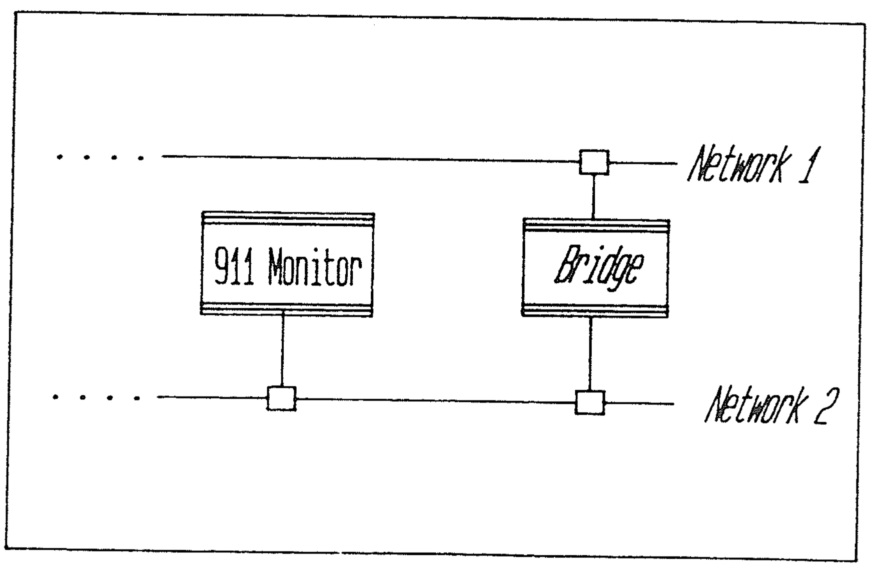


Figure 5.5 The 911 Monitor in the network.

The display of interest for validation in this experiment is "Utilization Display". A typical example of the utilization display is showed in Table 5.4. The long term utilization of the network which was recorded was based on a period of 13 days in this case.

```
911 MONITOR  V 1.46      * Stations  65 *            AVG     CURR    PEAK
03-JUN-1987 16:33:23     * Typ Fld    7 *          (00:30)  (00:30)
LAN : hsc                * MC Addr    7 * Util (%)  0.2      0.2     0.3
Uptime : 00  00:17:59    * CRC Err    9 * Pkt/Sec    25       25      56
                         * 802 Pkts  16 * MC (%)    6.2      6.2    11.7
*UTILIZATION STATS DISPLAY********************************************
 Statistics are based on a 911 Up Time of : 13 days 01:17:39

911 Overflow information, percent of counter capacity used :  0.49 %

Long Term Utilization :   0.11 %
Long Term Packet Rate :     19 pkt/sec

Long Term MC vs SD statistics (bytes and packets):

                          Byte Count                Percent
      SD Bytes            1480256128                91.31
      MC Bytes             140853520                 8.69

                           Pkt Count                Percent
      SD Packets            19233569                91.69
      MC Packets             1743504                 8.31
```

Table 5.4. The Utilization Display of the 911 Monitor.

SD = single destination

MC = multicast.

The measurement of the experimental network were performed as follows. The counters of RBMS and NCP were zeroed initially. Then, DTS was used to generate the network load. After the DTS ended, the statistics from the counters of the NCP and RBMS were recorded. For the SHOW LINE COUNTER of the NCP, the bytes sent/received counters showed the actual number of bytes in each node, and the data blocks sent/received showed the number of messages in each node. Then, the filtered frame counter and frames counter in RBMS were used to verify that the statistics collected from NCP were correct.

The next step was to calculate the total channel utilization for each network based on the above statistics. It should noted that the channel utilization in this case was the data actually used by the physical channel.

$$\text{Channel Utilization} = \frac{\text{Sum of the bytes sent in the network (Mbits)}}{\text{Total time for the DTS specified (sec)} * 10 \text{ Mbps}}$$

Finally, the utilization display of the 911 monitor was used to verify that the calculated channel utilization was correct.

5.3.1.4 Simulation

At this point, the real statistics of the networks were known. These statistics included the number of messages sent from each node and the number of bytes sent from each node. The simulation was then set up based on these statistics. Normally, transmitted messages in the simulation program are randomly distributed in the time slice. However, the transmitted messages in this experiment had to be assigned in order to match the load generated by DTS. The simulation program was modified slightly such that the transmitted messages of VAX2 and IBM in the program were evenly distributed in the time slice. It was found that when a transmitted message was sent from VAX2 to VAX1, VAX1 replied back with an acknowledgment packet to the VAX2. The simulation statistics were then compared to the measured statistics.

5.3.2 Results

The goal of this section is to compare the the measured results and the simulation results in the experiment. The measured results are based upon six trials of DTS Data Test. The measured statistics of the six trials are given in Appendix F. Table 5.5 is a summary of the measured statistics and simulated results.

	Measured Statistics				Simulation Statistics				
Node =>	VAX2	VAX1	IBM	uVAX	VAX2	VAX1	IBM	B (NET2)	B (NET1)
Second	1	1	1	1	1	1	1	1	1
Message size(bytes)	1400.0	47.2	1400.0	---	1400	47	1400	---	---
Overhead (bytes)	---	---	---	---	26	26	26	---	---
Messages send	139.1 [9.8]	82.6 [5.9]	35.2 [1.0]	---	139	89	35	---	---
Bytes send	194600.0	3898.7	49280.0	---	194600	4183	49000	---	---
Messages sent, multiple collisions	0.030 [0.016]	0.008 [0.006]	0.006 [0.006]	---	0	0	0	0	0
Messages sent, single collision	0.090 [0.025]	0.011 [0.008]	0.002 [0.002]	---	0	0	0	0	0
Messages sent, initially deferred	5.879 [0.468]	0.107 [0.023]	0.000 [0.000]	---	5	0	6	4	0
Send failure	0.000 [0.000]	0.000 [0.000]	0.069 [0.015]	0.00	0	0	0	0	0
Receive failure	0.000 [0.000]	0.000 [0.000]	0.051 [0.016]	0.00	0	0	0	0	0

TABLE 5.5 Summary of the measured results and simulation results.

B (NET1) = Bridge node on the network 1. B (NET2) = Bridge node on the network 2.

[ ] = Standard deviation. --- = Not available

5.3.3 Discussion

The simulation results were compared to the measured results. It was found that the some of the measured statistics closely matched those of the simulation for both VAX2 and VAX1. There were, however, some initial deferrals on the bridge of the simulated network 2 which were not seen in the real measurements. These initial deferrals were most likely caused by acknowledgments which originally came from VAX1 to the bridge, and destined for the network 2. Since the bridge does not maintain these statistics, there is no current means to compare the deferrals in the bridge from the simulation to the measurement.

There were also some initial deferrals on the IBM during simulation not seen in the measurement. The reason for these deferrals might be the send failure of the IBM device. The real send failure counter indicated the total number of times a transmission failed. This counter can be affected by excessive collisions, carrier check failed, and remote failure to defer (Digital NCP, 1986). The fact that only the IBM had send failures on the real network implies that DECnet facility or the transceiver controller might be at fault. Since the simulation predicted 6 deferrals compared to only 0.069 actual send failures, it would appear that the carrier sense and error detection circuitry might be a probable source of the difference. As well, there was no send failure or receive failure occurred in the uVAX which implies that the IBM error detection circuitry was not functional.

In summary, the comparable simulation results differ from the measured statistics by less than 18% (5 compared with 5.88 deferrals) in the same offered traffic load. However, futher data and analysis is required to complete this comparison.

5.4 COMMENTS

Due to the bandwidth limitation of the computer resources in Health Sciences Centre, a traffic load above 23% could not be generated. There is no current means to compare the simulation results of higher traffic loads to the measurement of the physical network. Further validation of the simulation program is required with a physical network which can generate high traffic loads. Only with high network utilization and collision rates can the accuracy of the simulation be fully tested.

CHAPTER 6

APPLICATION OF THE METHOD

6.1 INTRODUCTION

The Winnipeg Health Sciences Centre is in the process of expanding the existing Ethernet local area network. Several design options are being considered for the expansion, as well as modifications to the existing configuration in order to maximize the performance of the networks. The simulation developed in this research was used as a computer aided design tool to predict network performance and obtain a satisfactory configuration.

6.2 PRESENT HEALTH SCIENCES CENTRE NETWORK

Three subsections are given to describe the current network configuration, the simulation results, and the actual measurement of the present networks.

6.2.1 Network Configuration

Figure 6.1 is a schematic network diagram showing the physical location of segments, repeaters, and bridges. The configuration consists of two networks, Network 1 and Network 2, connected by a bridge. A second redundent bridge is located between segment 3 of each network. As this bridge is normally in a monitor mode, it was not included in the model.

Network 1 consists of one local repeater, one remote repeater, and two multiport repeaters. Table 6.1 shows all the physical location of segments corresponding to the repeater type in network 1. There are twelve segments in network 1, which includes two Ethernet coaxial segments, one link segment and nine Ethernet thinwire segments. Table 6.2 shows the physical location and the address of each device in network 1. The addresses in the Table 6.2 are not Ethernet addresses, but the input parameters for the simulation program. A total of fourteen terminal servers (DECserver), one Delni and one VAX 11/785 are connected to network 1. DECservers can connect up to eight input/output devices to the computer systems on the Ethernet networks (Digital TS, 1986). The Delni is a concentrator that allows up to eight Ethernet compatible devices (not terminals) to be grouped together (Digital, 1986).

Network 2 consists of four local repeaters, and one remote repeater. Table 6.3 shows all the physical location of segments corresponding to the repeater type in network 2. There are seven segments in network 2 which includes six Ethernet coaxial segments, and one link segment. Table 6.4 shows the physical location and the address of each device in network 2. Network 2 consists of three Delnis, twenty-six terminal servers (DECservers), three micro-computers and one VAX 8650 (VAX2).

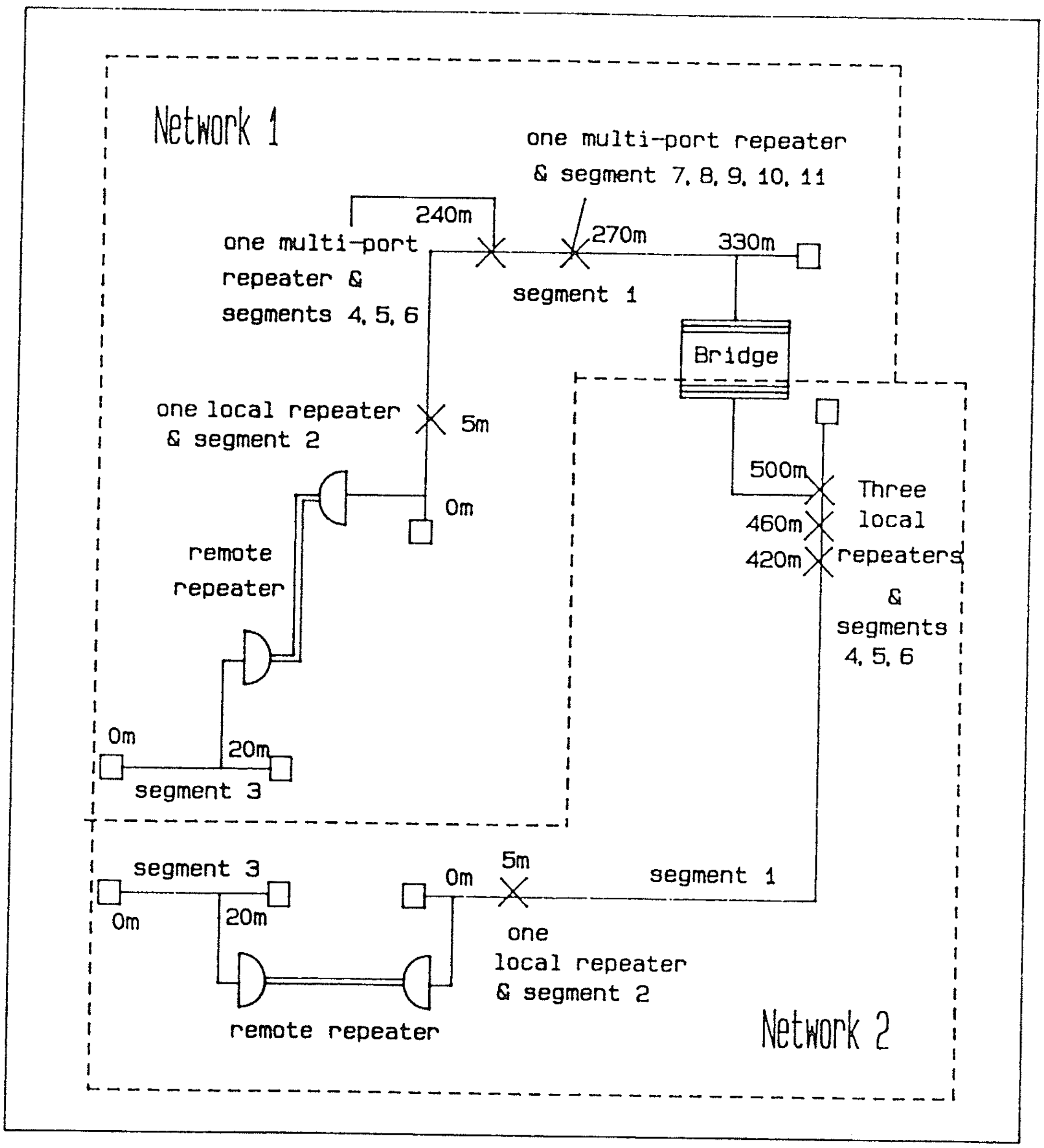


Figure 6.1 Health Science Ethernet Network.

Repeater Type	From segment number	Physical location (meters)	To segment number	Physical location (meters)
local	1	5.0	2	0.0
remote	1	0.0	3	20.0
Multi	1 1 1	240.0 240.0 240.0	4 5 6	0.0 0.0 0.0
Multi	1 1 1 1 1	270.0 270.0 270.0 270.0 270.0	7 8 9 10 11	0.0 0.0 0.0 0.0 0.0

Table 6.1 The physical location of repeaters in Network 1.

Note: Link segment does not have segment number.

Node address	Device type	Physical distance (meters)
1.1.1	B	330.0
1.2.1 1.2.2 1.2.3	TS TS TS	85.0 87.5 110.0
1.3.1 1.3.2	VAX1 DEL	10.0 15.0
1.4.1 1.4.2 1.4.3	TS TS TS	50.0 70.0 72.5
1.5.1 1.5.2	TS TS	100.0 100.5
1.6.1	TS	110.0
1.7.1	TS	50.0
1.8.1	TS	80.0
1.9.1	TS	50.0
1.10.1	TS	120.0
1.11.1	TS	50.0

Table 6.2 The physical location of nodes in Network 1.
Note:
TS = DECserver
DEL = Delni

Repeater Type	From segment number	Physical location (meters)	To segment number	Physical location (meters)
local	1	5.0	2	0.0
remote	1	0.0	3	20.0
local	1	420.0	4	150.0
local	1	460.0	5	150.0
local	1	500.0	6	150.0

Table 6.3 The physical location of repeaters in Network 2.

Node address	Device type	Physical distance (meters)
2.1.1	B	500.0
2.2.1	TS	70.0
2.2.2	TS	72.5
2.2.3	TS	89.5
2.2.4	TS	92.0
2.3.1	VAX2	2.5
2.3.2	UVAX	5.0
2.3.3	PRO	10.0
2.3.4	IBM	12.5
2.3.5	TS	20.0
2.4.1	DEL	100.0
2.5.1	TS	20.0
2.5.2	TS	150.0
2.5.3	TS	152.5
2.5.4	TS	182.5
2.5.5	TS	250.0
2.5.6	TS	280.0
2.6.1	TS	25.0
2.6.2	TS	35.0
2.6.3	TS	45.0
2.6.4	TS	55.0
2.6.5	TS	85.0
2.6.6	DEL	125.0
2.6.7	TS	140.0
2.6.8	TS	180.0
2.6.9	TS	220.0
2.6.10	TS	250.0
2.6.11	TS	252.5
2.6.12	TS	270.0
2.6.13	TS	275.0
2.6.14	TS	280.0
2.6.15	TS	300.0
2.6.16	TS	320.0

Table 6.4 The physical location of nodes in Network 2.

Note:
TS = DECserver, DEL = Delni

6.2.2 Simulation Setup

Before the simulation, the input parameters for the computer program must be found. It was known that most of the traffic is created by the terminal servers communicating with the main computers, VAX2 and VAX1. By observing the counters in the terminal servers and averaging the statistics in the counters, the average packet size sent out by VAX1, VAX2, and the Terminal servers was found during the period from 10:00 to 16:00. The detailed statistics of terminal servers are given in Appendix G. Therefore, the input parameters to the simulation were as summarized in Table 6.5. Table 6.5 shows the number of nodes, the packet size, the I/O bandwidth limitation and acknowledgment requirement (Reply packet) for each device. The time slice was chosen to be one second (10,000,000 bit-times). Also, a single packet per message was used to represent the full duplex communication used by the terminal servers.

Device	VAX2	VAX1	UVAX	IBM	PRO	DEL	TS	B
Type =>	VAX 8650	VAX 785	Micro VAX	IBM PC AT	PRO 380	DELNI	DEC Server	See Note
No. of Nodes	1	1	1	1	1	3	40	2
Packet Size => (bits)	592	608	502	502	502	502	502	---
Overhead (bits)	208	208	208	208	208	208	208	---
Total Packet Size => (bits)	800	816	710	710	710	710	710	---
I/O Band width	2.0 Mps	2.0 Mps	0.5 Mps	0.45 Mps	0.45 Mps	0.896 Mps	0.112 Mps	---
Reply Packet	Yes	Yes	No	No	No	No	No	No
Time slice = 10,000,000 bit times = 1.0 sec Single packet per message.								

Table 6.5 Input data for the simulation program.

Note : Bridge (B) is considerd to be two special nodes; one on each network. The bandwidth limitation equals to 10 Mps.

6.2.3 Simulation Results

The simulation results were summarized in Table 6.6a and Table 6.6b. Since there are two networks, the results were grouped according to the network number. Seven trials of different load offerings were used. Tables 6.6a and 6.6b were divided into 14 columns. The columns are:

Column A : the number for the trials.

Column B : shows the number of sets of transmissions. Each set of transmissions equals to the total number of nodes in the networks.

Column C : the number of messages distributed in the time slice.

Column D : the total time used by the data and delays of the messages.

Column E : the total time used by the data of the messages only.

Column F : the current position in the time slice when the program ended.

Column G : the average message size.

Column H : the average delay per message.

Column I : the channel utilization.

Column J : the number of initially deferred packets.

Column K : the number of single collisions.

Column L : the number of multiple collisions.

Column M : the number of errors.

Column N : the CPU time used for the simulation run.

All time is in units of bit time, except CPU time which is in seconds.

The graphs of each column in Table 6.6a and Table 6.6b as a function of number of messages, were plotted in Figures 6.2 to 6.7. The average message size was 773.0 bits for Network 1 and 762.7 bits for Network 2.

Simulation Results: **Network 1**

A	B	C	D	E	F	G	H
Trial	SET	Num. of mess.	Non idle time	Time for tx. messages	End time	Average Message size	Delay per mess.
i	6.4	209	284299	162382	10001467	776.9	583.3
ii	10.0	327	449162	251992	9972470	770.6	603.0
iii	20.0	633	923043	487696	9999347	770.5	687.8
iv	30.0	863	1320797	666578	10005051	772.4	758.0
v	40.0	1067	1563544	824456	9974137	772.7	692.7
vi	50.0	1244	1862860	962316	10001037	773.6	713.9
vii	60.0	1434	2221130	1107074	9990220	772.0	776.9
viii	90.0	1807	2779990	1399570	10004271	774.5	763.9
ix	120.0	2107	3696392	1631332	9999483	774.2	980.1

Simulation Results: **Network 2**

A	B	C	D	E	F	G	H
Trial	SET	Num. of mess.	Non idle time	Time for tx. messages	End time	Average message size	Delay per mess.
i	6.4	307	375854	234170	9994375	762.8	461.5
ii	10.0	457	566620	348950	9971664	763.6	476.3
iii	20.0	824	1107799	628330	9995204	762.5	581.9
iv	30.0	1147	1553855	875030	10004245	762.9	591.8
v	40.0	1345	1921983	1027040	9999005	763.6	665.4
vi	50.0	1590	2221155	1213860	10001949	763.4	633.5
vii	60.0	1748	2416934	1333150	9991121	762.7	620.0
viii	90.0	2245	3441806	1709420	10005001	761.4	771.7
ix	120.0	2567	4262535	1955500	9995374	761.7	898.7

Table 6.6a Simulation Results for Health Sciences Centre Network

Simulation: Network 1

A	B	C	I	J	K	L	M	N
Times	SET	Num. of mess.	C.U. %	DEF	S.C.	M.C.	ERR	CPU Time (sec)
i	6.4	209	1.62	4	0	3	0	209.2
ii	10.0	327	2.53	5	0	4	0	581.6
iii	20.0	633	4.88	38	2	21	0	3426.2
iv	30.0	863	6.66	58	3	45	0	9033.4
v	40.0	1067	8.27	81	2	47	0	16465.3
vi	50.0	1244	9.62	110	4	73	0	26191.6
vii	60.0	1434	11.08	136	10	79	0	36883.2
viii	90.0	1807	13.99	228	14	116	0	76661.4
ix	120.0	2107	16.31	343	24	207	0	133887.0

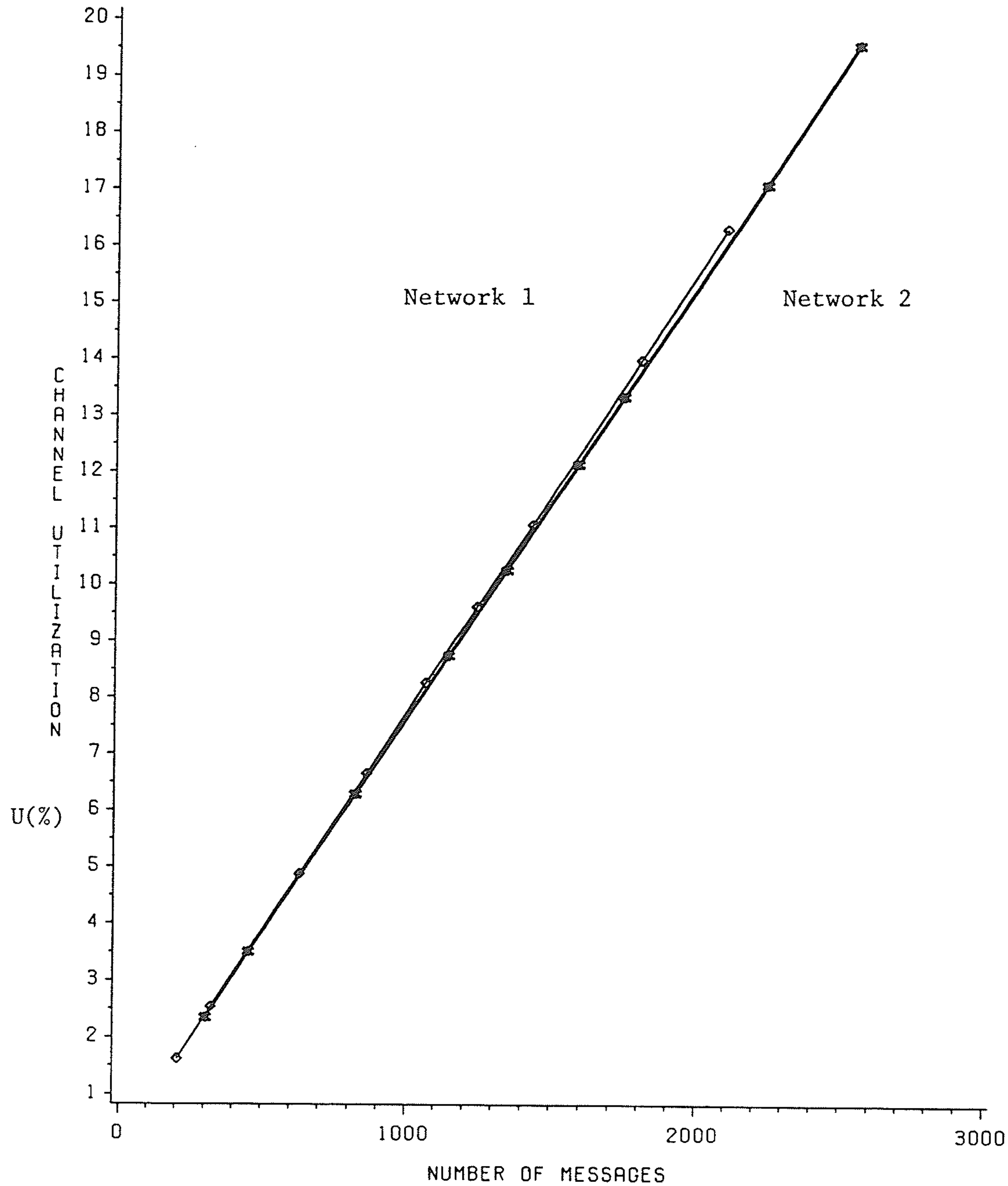
Simulation: Network 2

A	B	C	I	J	K	L	M	N
Times	SET	Num. of mess.	C.U. %	DEF	S.C.	M.C.	ERR	CPU Time (sec)
i	6.4	307	2.34	4	0	2	0	209.2
ii	10.0	457	3.50	12	0	8	0	581.6
iii	20.0	824	6.29	27	4	30	0	3426.2
iv	30.0	1147	8.75	58	8	44	0	9033.4
v	40.0	1345	10.27	82	6	61	0	16465.3
vi	50.0	1590	12.14	116	14	94	0	26191.6
vii	60.0	1748	13.34	128	12	79	0	36883.2
viii	90.0	2245	17.09	233	17	158	0	76661.4
ix	120.0	2567	19.56	343	29	252	0	133887.0

Table 6.6b Simulation Results for Health Sciences Centre Network

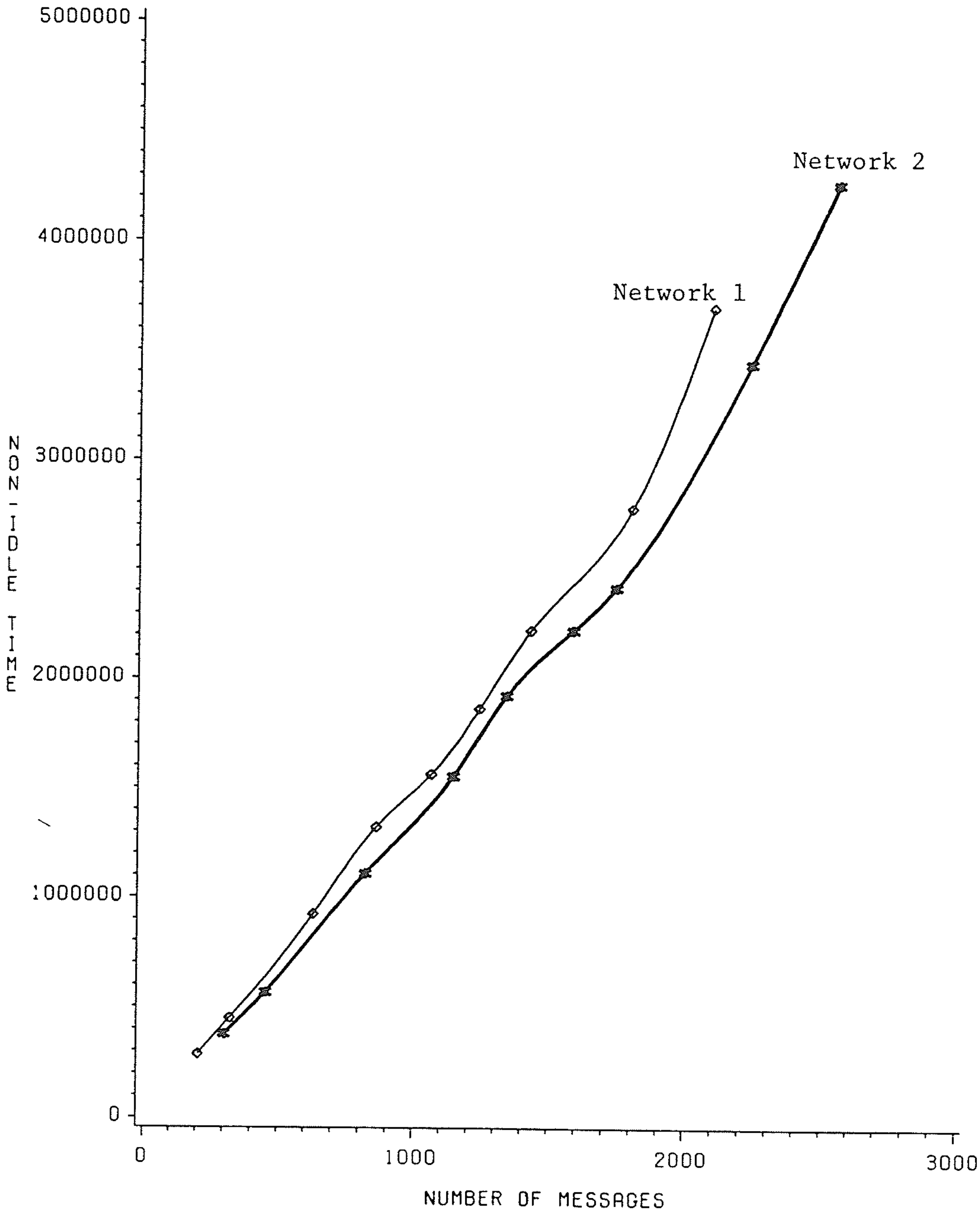
CHANNEL UTILIZATION VS NUMBER OF MESSAGES

FIGURE 6.2



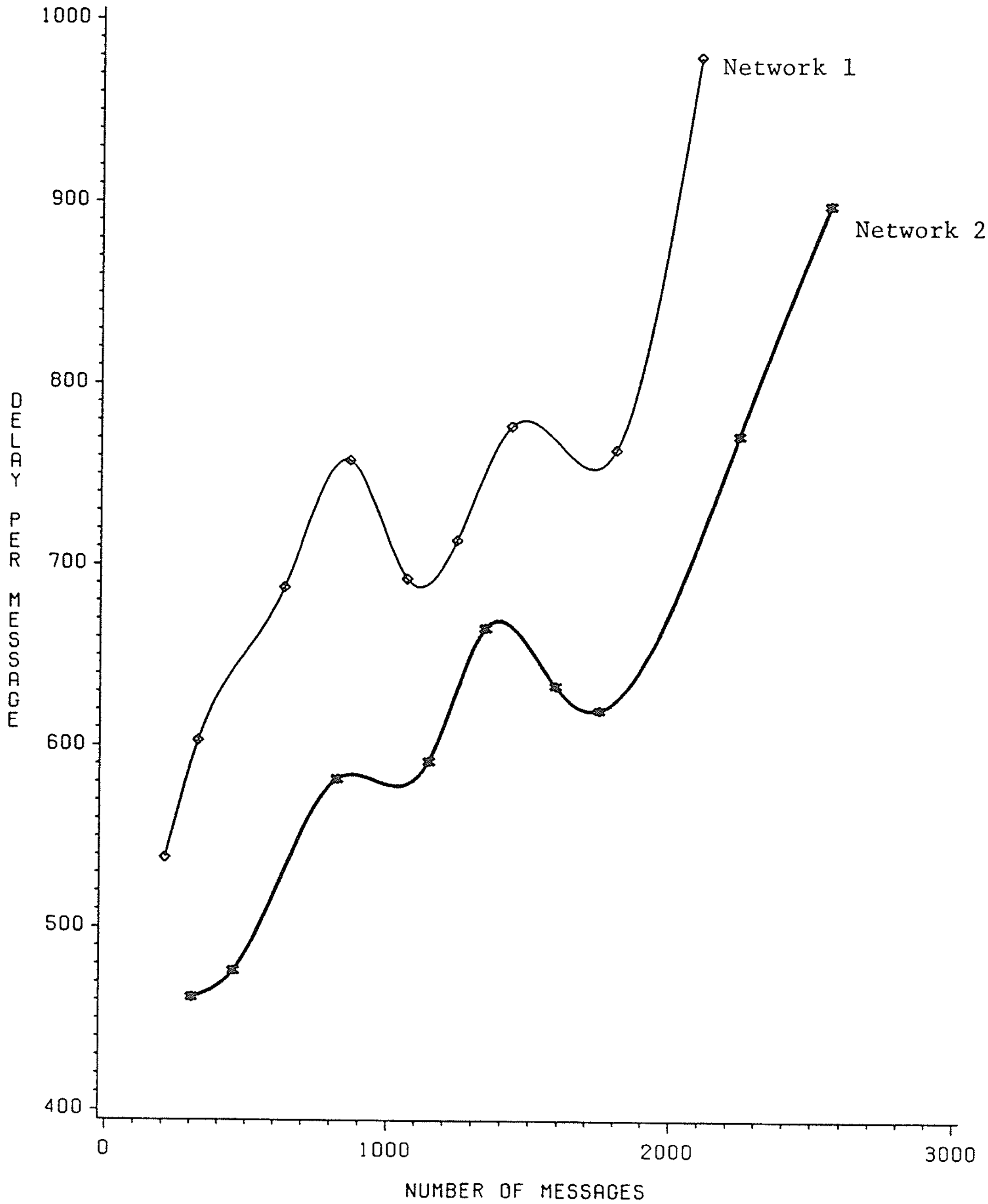
NON–IDLE TIME VS NUMBER OF MESSAGES

FIGURE 6.3



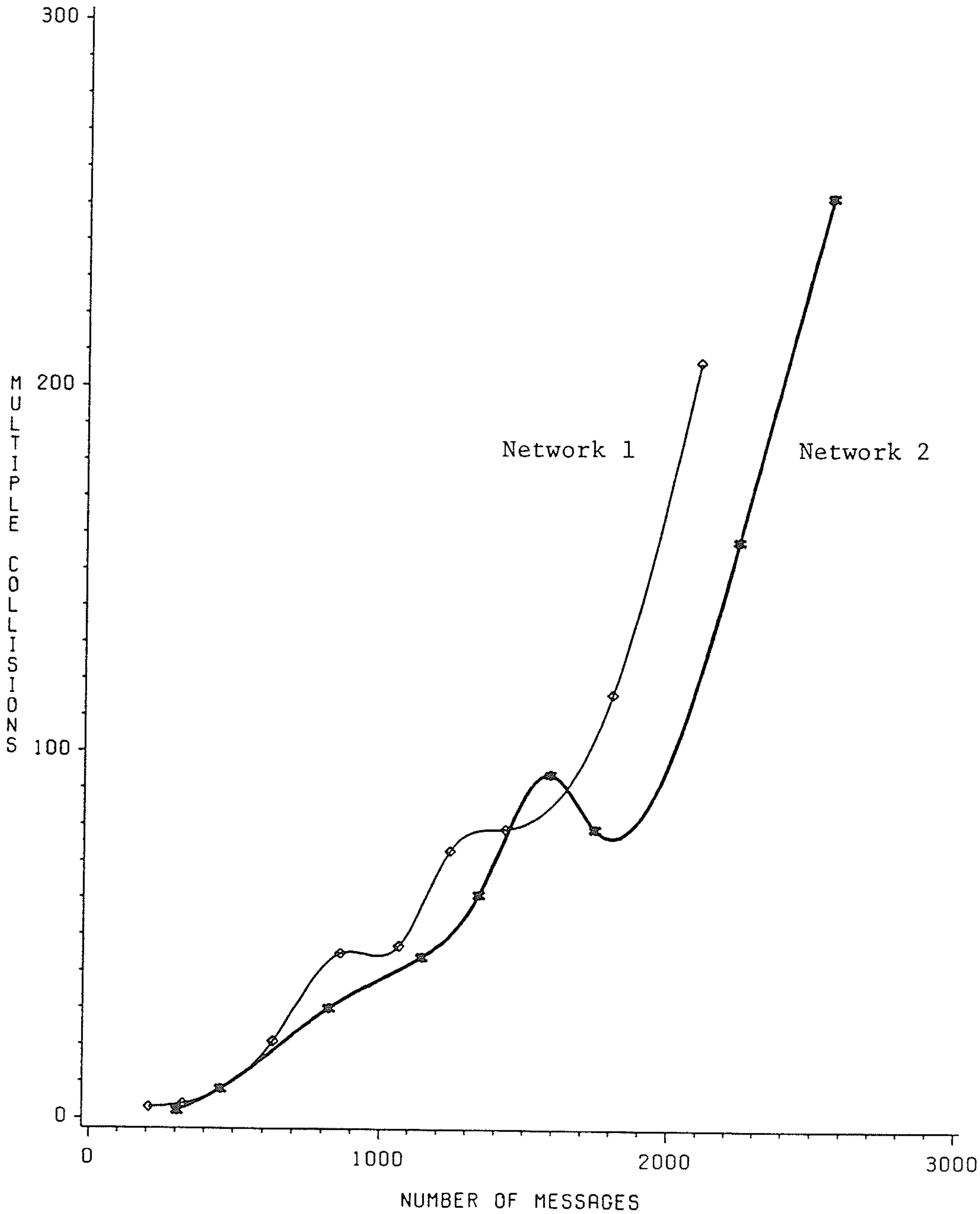
DELAY PER MESSAGE VS NUMBER OF MESSAGES

FIGURE 6.4



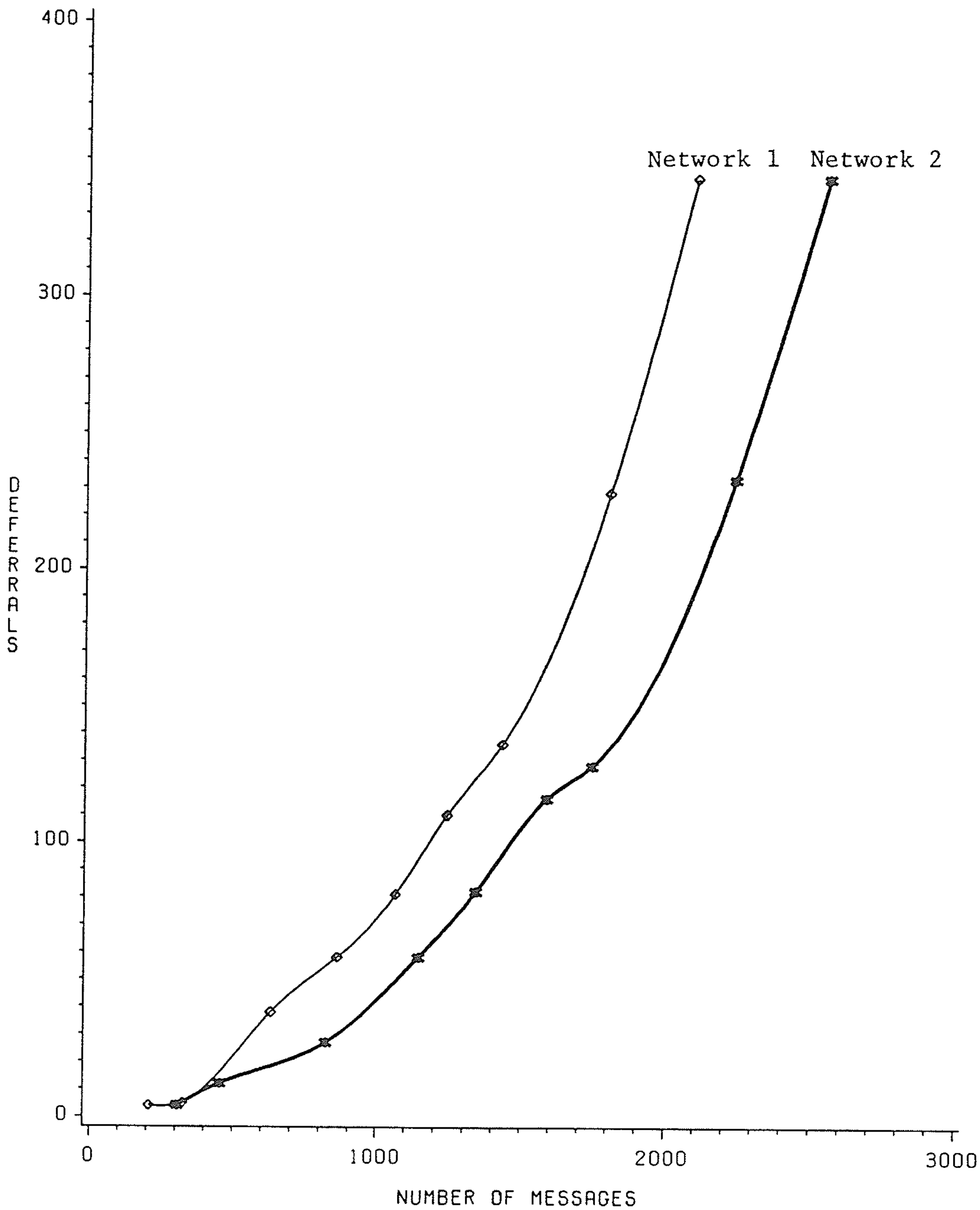
MULTIPLE COLLISIONS VS NUMBER OF MESSAGES

FIGURE 6.5



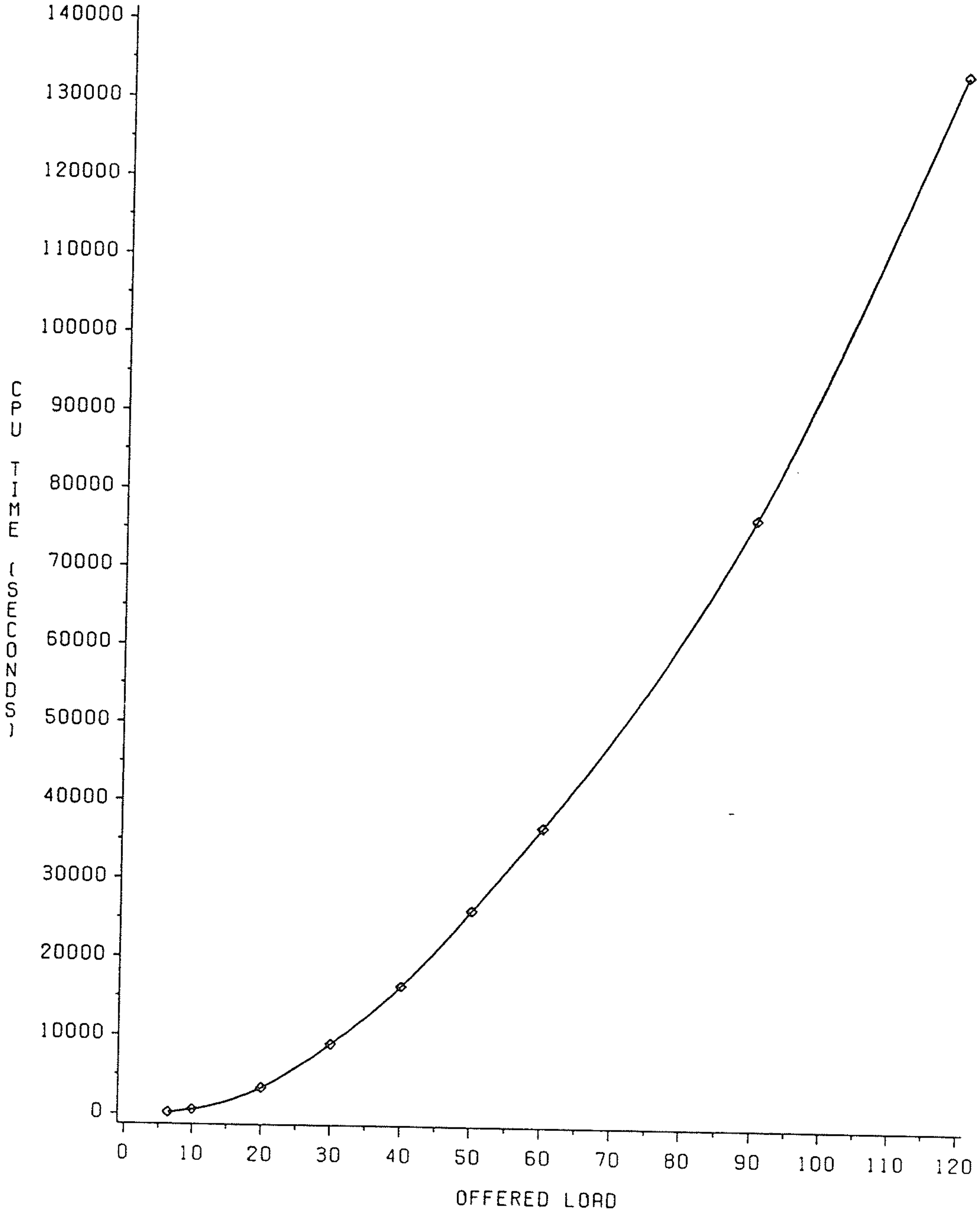
DEFERRALS VS NUMBER OF MESSAGES

FIGURE 6.6



CPU TIME VS OFFERED LOAD

FIGURE 6.7



The average offered load for the present network was 6.4 sets of transmissions. This value was calculated based on the counters of the terminal servers. Table 6.7 summarizes the simulation results for 6.4 sets of offered load in the average of four simulation runs. The average and standard deviation for the delay per message in network 1 were 620.1 and 47.2 bit times respectively, the average and standard deviation for the delay per message in network 2 were 454.8 and 10.7 bit times respectively. The finite standard deviation explains why the Figure 6.4 is not smooth.

NETWORK 1:	(i)	(ii)	(iii)	(iv)	Average	SD
No. of tx. TX. messages. >	209	223	229	239	225.0	12.5
No.of initially deferred. >	4	5	5	6	5.0	0.8
No. of single collisions >	0	0	1	1	0.5	0.6
No. of multiple collisions >	3	7	5	2	4.3	2.2
No. of errors >	0	0	0	0	0.0	0.0
Non idle time >	284299	311829	332916	325187	313557.8	21362.3
Time used by TX. messages >	162382	172216	176158	185060	173954.0	9400.9
Simulated time>	10001467	9993611	10000548	9977730	9993339.0	10981.0
Ave. Mess. size	776.9	772.2	769.2	774.3	733.2	3.3
Delay/message >	583.3	626.1	684.5	586.3	620.1	47.2
Channel Utilization >	1.62	1.72	1.76	1.85	1.74	0.10

NETWORK 2:	(i)	(ii)	(iii)	(iv)	Average	SD
No. of tx. TX. messages. >	307	269	293	278	286.8	16.7
No.of initially deferred. >	4	3	3	4	3.5	0.6
No. of single collisions >	0	0	0	0	0.0	0.0
No. of multiple collisions >	2	2	2	0	1.5	1.0
No. of errors >	0	0	0	0	0.0	0.0
Non idle time >	375854	326358	360932	335852	349749.0	22706.8
Time used by TX. messages >	234170	206200	224320	212230	219230.0	12488.6
Simulated time>	9994375	9992805	10001354	9962347	9987720.3	17318.1
Ave. Mess. size	762.8	766.5	765.6	763.4	746.6	1.8
Delay/message >	461.5	446.7	466.3	444.7	454.8	10.7
Channel Utilization >	2.34	2.06	2.24	2.13	2.19	0.12

Table 6.7 Simulation results for 6.4 sets of offered load.

The present Health Sciences Centre Network is at a very low level of traffic. The channel utilized by the data of the messages are 1.74% with standard deviation 0.10% for network 1 and 2.19% with standard deviation 0.12% for network 2. The average delay per messages is 620.1 bit times (62.01 usec) for network 1 and 454.8 bit times (45.48 usec) for network 2.

6.3 FUTURE HEALTH SCIENCE CENTRE NETWORK PERFORMANCE

6.3.1 Network Configurations

Health Science Centre is in the process of expanding the present network. More nodes such as terminal servers and computer resources will be added on the present network. The computer program can be used to assist the network designer to simulate the actual behaviour of the future Health Sciences Network.

For the future network, the number of terminal servers will be increased first. The input parameters and the rate of offered load per device is expected to be constant. Only the number of terminal servers will be altered. Two individual simulation runs which used 140 and 240 terminal servers were performed. The simulation results are given in the next section.

The number of computers will also be increased in the future network. This means that more Digital VAX computers will be purchased in order to fulfill the demand of additional users. In the simulation, the bandwidth limitation of the two VAXs were increased to 6 Mbits/sec from 2 Mbits/sec. Thus, one 6 Mbits/sec VAX is equivalent to three of the current VAX computers. The simulation results are also given in the next section.

6.3.2 Simulation Results

The simulation results for increasing the number of terminal servers to be 140 and 240 are summarized in Table 6.8 and Figure 6.8.

For 40 terminal servers, the channel utilization is predicted to be under 2.5% of the Ethernet bandwidth for each network. About 97% of the offered transmitted messages did not experience initial deferrals, single collisions, or multiple collisions in each network.

For 140 terminal servers, the channel utilization is predicted to be under 6% of the Ethernet bandwidth for each network. About 94% of the offered transmitted messages did not experience initial deferrals, single collisions, or multiple collisions in both networks.

For 240 terminal servers, the channel utilization is predicted to be under 9% of the Ethernet bandwidth for each network. About 91% of the offered transmitted messages did not experience initial deferrals, single collisions, or multiple collisions in both networks.

Based upon this results, the future network should easily support a large number of terminal servers.

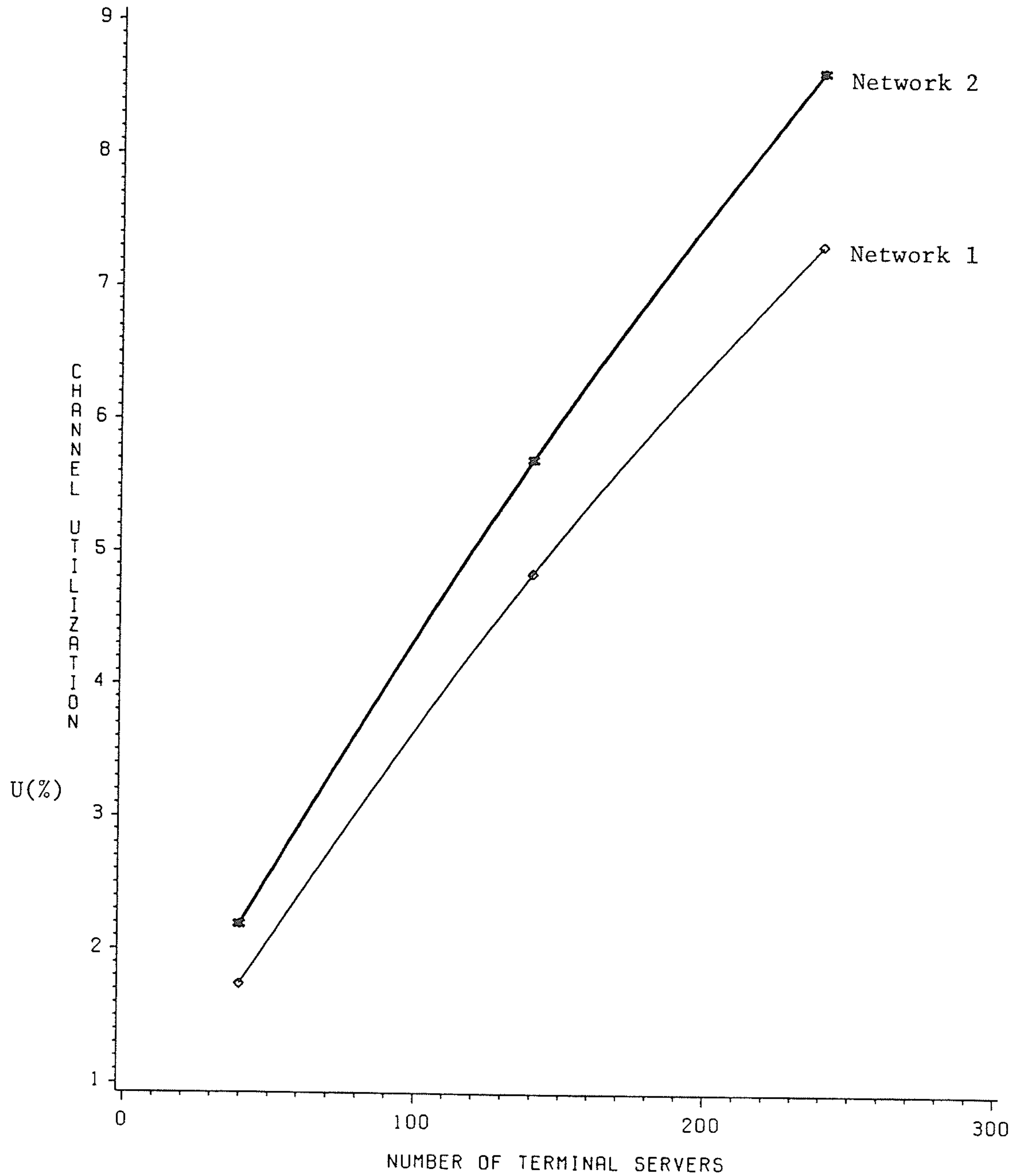
NETWORK 1:	Present	Future	
No. of terminal servers >	40	140	240
No. of tx. messages No. of initially deferred No. of single collisions No. of multiple collisions No. of errors	225.0 5.0 0.5 4.3 0.0	628 22 2 16 0	947 54 2 28 0
Non idle time Time used by tx.mess. Simulated time.	313557.8 173954.0 9993339.0	863980 483298 9992193	1304300 731306 9988367
Average message size Delay Per message Channel Utilization	733.2 620.1 1.74	769.6 606.2 4.84	772.2 605.0 7.32

NETWORK 2:	Present	Future	
No. of terminal servers >	40	140	240
No. of tx. messages No. of initially deferred No. of single collisions No. of multiple collisions No. of errors	286.8 3.5 0.0 1.5 0.0	746 20 2 13 0	1130 55 4 40 0
Non idle time Time used by tx.mess. Simulated time.	349749.0 219230.0 9987720.3	951682 570070 10000137	1531098 861610 10000385
Average message size Delay Per message Channel Utilization	764.6 454.8 2.19	764.2 511.5 5.70	762.5 592.5 8.62

Table 6.8 Simulation results for different number of Terminal Servers.

UTILIZATION VS NUMBER OF TERMINAL SERVERS

FIGURE 6.8



The simulation results of two VAXs (with 6 Mbps bandwidth limitation) and 40 terminal servers are given in Table 6.9. The number of sets of transmission was abitrarily chosen to be 50 which should produce moderate network activity. The channel utilization is 12.1% for network 1 and 15.9% for network 2. The average delay per messages is 1003.2 bit times (100.32 usec) for network 1 and 810.0 bit times (81.0 usec) for network 2. The average packet size is 772.0 bit times for network 1 and 763.9 bit times for network 2. Based upon this results, the future network should be able to support more VAX computers.

```
STATISTICS OF NODES FOR NETWORK 1 :

The number of transmitted messages =  1567
The number of initially deferred   =  267
The number of single   collisions  =  15
The number of multiple collisions  =  156
The number of errors               =  0
The total non idle time in channel =  2781634
The total time used for messages   =  1209666
The ending time in the master clock=  10001037
The channel utilization            =  0.1210


STATISTICS OF NODES FOR NETWORK 2 :

The number of transmitted messages =  2084
The number of initially deferred   =  295
The number of single   collisions  =  23
The number of multiple collisions  =  197
The number of errors               =  0
The total non idle time in channel =  3280393
The total time used for messages   =  1591960
The ending time in the master clock=  10001949
The channel utilization            =  0.1592

CPU     TIME=          61793.700 sec
```

Table 6.9 Simulation results for two VAXs with the bandwidth limitation of 6 Mbps.

CHAPTER 7

DISCUSSION

The simulation developed in this research provides a computer aided design mechanism in which arbitrarily specified Ethernet networks can be tested for performance and efficiency. The results of the simulation agree in general with those from the review of the literature in Chapter 3; the delay per message, the number of collisions and the number of deferrals exponentially increase with the number of messages. In addition, the simulation can be used in the design of a large network configuration with detailed and exact implementation of IEEE 802.3 CSMA/CD. The simulation generally proved to be valid, although more extensive analysis and testing is still required.

The simulation was applied to light offered loads as shown in Chapter 5. For the available data, the program produced results which differ from actual measured performance by less than 20%. Further validation of the program and the model is required for heavily loaded networks. This would likely require a very large physical network such might include 10 to 20 VAX computers.

The simulation has been used to plan the expansion of the Health Sciences Centre network and has provided detailed performance predictions which have guided future design. The results indicate that the present Health Sciences Centre network has more than sufficient bandwidth to support all forseeable requirements. Additional testing of the simulation is required before it can be used with confidence on networks with utilizations greater than 30%.

CPU time used to execute the simulation is exponential with the offered load, as seen in figure 6.7. Offered loads above 30% require more than 20 hours of CPU time for a VAX 8650; the simulation program is therefore impractical for heavy loads. Most of the CPU execution time was spent sorting the transmissions in the order of position on the time slice. Future enhancements should be made to increase the program efficiency by using different sorting algorithms such as the partition exchange sort (Bate et al, 1984), or by using MACRO subroutines for the intensively used component of the program.

In its present form, the simulation program randomly assigns the addresses of the sending nodes and the receiving nodes. However, it would be useful to be able to assign and control particular nodes to be the sending nodes. This could be implemented through a minor change to the program.

The checking interval is 512 bit times in the program. The figure of 512 bit times was assumed to be the worst case which is based upon the farthest two nodes on the network limitation (2800 meters). If the distance between the farthest two nodes in a simulation is much smaller than 2800 meters, a smaller checking interval can be used to increase the program efficiency.

The ability to graphically display the transmissions in the time slice is another suggestion for further consideration. VAX Basic Version 3.0 can provide a good graphic output to the terminal. VAX Basic version 3.0 is not currently available in the Health Sciences Centre and will be installed in next year.

In conclusion, the concept of computer aided design of Ethernet networks has proved to be of practical value. The current implementation appears to be sufficiently accurate for practical purposes, but additional testing is needed. In its current state, the simulation has proved to be useful, and the results obtained for the Health Sciences Centre justify further efforts towards a more efficient program design.

CHAPTER 8

REFERENCES

Alisouskas, F. (1985). Digital and Data Communications. Englewood Cliffs, N.J. : Prentice Hall.

Apostolopoulos, T. and Protonotarios, E. (1986). Queueing Analysis of Buffered CSMA/CD Protocols. IEEE Transactions on Communications 34(9), 898-905.

Arthurs, E. and Stuck, B. (1982). A Theoretical Performance Analysis of Polling and Carrier Sense Collision Detection Communication Systems. Bell Lab., Murray Hill, New Jersey.

Arthurs, E. and Stuck, B. (1984). A Modified Access Policy for Ethernet Version 1.0 Data Link Layer. IEEE Transactions on Communication, 32(8), 977-979.

Balaban, P. and Shanmugam, K. (1980). A Modified Monte-Carlo Simulation Technique for the Evaluation of Error Rate in Digital Communication Systems. IEEE Transactions on Communication, 28(11), 1916-1924.

Bate, J., Doyle, M., and Ferch, H. (1984). Computer Programming and Data Structures using Pascal. Manitoba : The Charles Babbage Research Centre.

Beauchamp, K. (1983). Information Technology and the Computer Network. New York : Springer-Verlag.

Bratley, P., Fox, B. and Schrage, L. (1983). A Guide to Simulation. New York : Springer-Verlag.

Bray, G. and Pokrass, D. (1985). Understanding ADA. New York : Wiley.

Bux, W. and Rudin, H. (1984). Performance of Computer Communication System. New York : North-Holland.

Bux, W. (1981). Local Area Subnetworks: A Performance Comparison. IEEE Transactions on Communication, 29(10), 1465-1473.

Carson, J. and Forman, E. (1981). Analysis of Local Area Network Performance. Proceedings IEEE Computer Networking Symposium, New York: IEEE Computer Society Press, 92-98.

Chanson, S. and Kumar, A. (1983). Using Markov Chains to Model Communication Protocols. Proceedings IEEE Computer Networking Symposium 1983, New York : IEEE, 9-16.

Cheong, V. and Hirschheim, R. (1983). Local Area Networks. New York : Wiley.

Chorafas, D. (1984). Designing and Implementing Local Area Networks. Toronto : McGraw-Hill Book Co.

Chorafas, D. (1985a). Bit-Oriented Protocols. Communications Solutions CS93-104-101. Datapro Research Co., Delran, New Jersey 08075.

Chorafas, D. (1985b). The Handbook of Data Communication and Computer Network. Princeton, N.J. : Petrocelli Books.

Chou, W. (1983). Computer Communication. Englewood Cliffs, N.J. : Prentice-Hall.

Choudhury, G. and Rappaport, S. (1985). Priority Access Schemes Using CSMA/CD. IEEE Transactions on Communications, 33(7), 620-626.

Colvin, A. (1983). CSMA with Collision Avoidance: A New Technique for Cost Reduction. Proceedings of Localnet 1983 (Europe), Northwood, UK : Online Publications, 223-238.

Dahmen, N., Killat, U. and Stecher, R. (1984). Performance Analysis of Token Bus and CSMA/CD Protocols Derived from FORCASD Simulation Runs. H. Rudin and W. Bux, editors, Performance of Computer Communication Systems, Amsterdam:North-Holland, 79-94, 1984.

Datapro (1986). An Overview of Local Area Networks.
Communications Solutions CS20-450-101.
Datapro Research Co., Delran, New Jersey 08075.

Davies, D. and Watts, D. (1979). Computer Networks and their Protocols
New York : Wiley.

Deitel, M. (1984). An Introduction to Operating Systems.
Don Mills, Ontario : Addision Wesley.

Digital (1981). VAX Architecture.
Boylston, Massachusetts : Digital Equipment Inc.

Digital (1982). Introduction to Local Area Networks.
Digital Equipment Co.

Digital (1983a). VAX Hardware Handbook.
Colorado Springs, Colorado : Digital Equipment Corporation.

Digital (1983b). VAX Software Handbook.
Merrimack , New Hampshire : Digital Equipment Corporation.

Digital (1986). Networks and Communications Buyer's Guide.
West Concord, MA : Digital Equipment Corporation.

Digital, Intel, and Xerox (1982). The Ethernet - Data Link Layer and Physical Layer Specifications.
Maynard, MA : Digital Equipment Corporation.

Franta, W. and Chlamtac, I. (1981). Local Networks.
Toronto : LexingtonBooks.

Friend, G., Fike, J., Baker, H., and Bellamy J. (1984).
Understanding Data Communications. Dallas, Texas : Texas Instruments Inc.

Gehani, N. (1985). ADA: Concurrent Programming.
Englewood Cliffs, N.J. : Prentice-Hall.

Gee, K. (1983). Introduction of Local Area Computer Networks.
New York : Wiley.

Goos, G. (1985). Local Area Network: an Advanced Course.
New York : Springer-Verlag.

Haenle, J. and Giessler, A. (1978). Simulation of Data Transport Systems of Packet-Switched Networks. S. Schoemaker, editor, Computer Networks and Simulation, Amsterdam: North-Holland, 104-117, 1978.

Harper, W. and Pollard, R. (1982). Data Communications Desk Book. Englewood Cliffs, N.J. : Prentice-Hall.

Hayes, J. (1984). Modeling and Analysis of Computer Communications Networks. New York : Plenum Press.

Heyman, D. (1986). The Effects of Random Message Sizes on the Performance of the CSMA/CD Protocol. IEEE Transactions on Communications 34(6), 547-553.

IEEE LLC (1984). IEEE Standards for Local Area Networks: Logical Link Control. New York : IEEE Press.

IEEE CSMA/CD (1985). IEEE Standards for Local Area Networks: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications. New York : IEEE Press.

Ilyas, M. and Mouftah, H. (1986). Message Delay Distributions in Store-and-forward Computer Networks. IEEE International Network Planning Symposium, New York : IEEE, 212-216.

Jenq, Y. (1986). Theoretical Analysis of SLOTTED ALOHA, CSMA, and CSMA/CD Protocols. Techtronix Applied Research Lab., Beaverton; Oregon.

Kanakia, H. and Tobagi, F. (1986). Performance Measurements of a Data Link Protocol. Defence Advanced Research Projects Agency under Contract MDA-903-84-K-0249.

Konstantas, D. (1983). Virtual Time CSMA: A study. Systems Research Group, University of Toronto. Toronto: Computer

Kuo, F., editor, (1981). Protocols and Techniques for Data Communicatio Networks. Englewood Cliffs, N.J. : Prentice Hall.

Kurose, K. and Shen, C. (1986). GENESIS: A Graphical Environment for the Modeling and Performance Analysis of Protocols in Multiple Access Network. IEEE Network Planning Symposium, New York:IEEE, 222-227,1986.

Marathe, M. and Hawe, B. (1982). Predicted Capacity of Ethernet in a University Environment. Electron. and Commun. 30(3), 92-93.

Mayne, A. (1986). Linked Local Area Networks. Toronto : John Wiley & Sons, Inc.

McNamara, J. (1985). Local Area Networks: An Introduction to the Technology. Burlington, Massachusetts : Digital Press.

Meditch, J. and Lea, C. (1983). Stability and Optimization of the CSMA and CSMA/CD Channels. IEEE Transactions on Communications, 31(6), 763-774.

Meijer, A. and Peeters, P. (1982). Computer Network Architectures. Toronto : Pitman Publishing Ltd.

Metcalfe, R. and Boggs, D. (1976). Ethernet : Distributed Packet Switching For Local Computer Networks. CACM, 19(7), 395-404.

Mier, E. (1984). The Evolution of a Standard Ethernet. BYTE Magazine, Dec., 9(13), 131-142.

Minnich, N. and Cotton, C. (1983). An Evaluation of Two Unibus Ethernet Controllers. IEEE 8th Conference on local computer networks. Oct., 1983 Minneapolis, Minnesota. Silver Spring, MD : IEEE Computer Society Press.

Molloy, M. (1984). Collision Resolution on the CSMA/CD Bus. IEEE 9th Conference on Local Computer Networks. Silver Spring, MD : IEEE Computer Society Press, 44-47.

Moura, J., Field, J and Wong, J. (1979). Evaluation of Collision Control Algorithms in Ethernets. Sixth Data Communications Symposium 1979, New York : IEEE, 82-86.

O'Reilly, P. and Hammond, J. (1982). An Efficient Algorithm for Generating the Busy/Idle Periods of a Channel Using CSMA and Loaded by an Arbitrary Number of Stations.
Computer Network Proceedings: Compcon 82, Fall.
Silver Spring, MD : IEEE Computer Society Press, 427-436.

Owen, (1982). PCM and Digital Transmission Systems.
New York : McGraw Hill.

Pendse, S. and Soueid, E. (1985). CSMA/CD With Channel Capture.
10th Conference on Local Computer Networks.
Minneapolis, Minnesota. Oct., 1985.
Silver Spring, MD : IEEE Computer Society Press, 54-60.

Peters, J. and Sallam, H. (1985). Business Computing (a Structured Approach to Basic on the PDP-11 & VAX-11.)

Ravasio, P., Hopkins, G. and Naffah. (1982). Local Computer Networks. (IFIP). New York : North Holland Pub. Co.

Sauer, C. and MacNair, E. (1983). Simulation of Computer Communication Systems. Englewood Cliffs, New Jersey : Prentice-Hall, Inc.

Schacham, N. and Hunt, B. (1982). Performance Evaluation of th CSMA/CD (1-persistent) Channel-Access Protocol in Common Channe Local Networks. P.C. Ravasio, G.Hopkins and N.Naffah (editors IFIP TC 6 International In-depth Symposium on Local Compute Networks, New York:North Holland Pub. Co., 401-413.

Schoemaker, S. (1981). Computer Networks & Simulation. Amsterdam, New York : North Holland Pub. Co.

Schoemaker, S. (1982). Computer Networks & Simulation II. Amsterdam, New York : North Holland Pub. Co.

Sherman, R., Gable, M. and McClure, G. (1978). Concepts, Strategies for Local Area Data Network Architectures. G.R. Davis, Editor. The Local Network Handbook, McGraw Hill, 2-10, 1982.

Shoch, J. and Hupp, J. (1980). Measured Performance of an Ethernet Local Network. Comm. ACM. 23(12), 711-721.

Shoch, J., Dalal, Y. and Redell, D. (1982). Evolution of the Ethernet Local Computer Network. R. Shotwell, editor, New York:North-Holland, The Ethernet Sourcebook, 13-30, 1985.

Stallings, W. (1984a). Local Network Performance.
IEEE Communicaton Magazine. 22(2)

Stallings, W. (1984b). Local Networks.
New York : Macmillan Publishing Co.

Syslo, M., Deo, N. and Kowalik, J. (1983). Discrete Optimization Algorithms with Pascal Programs.
Englewood Cliffs, N.J. : Prentice Hall.

Takagi, H. and Kleinrock, L. (1985). Mean Packet Queueing in a Buffered Two-User CSMA/CD System.
IEEE Transactions on Communication, 33(10), 1136-1139.

Tanenbaum, A. (1981). Computer Network.
Englewood Cliffs, New Jersey : Prentice Hall.

Tanenbaum, A. (1983). An Overview of Local Networking Techniques.
Communications Solutions CS20-455-101.
Datapro Research Co., Delran, New Jersey 08075.

Tasaka, S. (1986). Dynamic Behavior of a CSMA/CD System With a Finite Population of Buffered Users.
IEEE Transactions on Communications 34(6),576-586.

Tobagi, F. and Hunt, V. (1979). Performance Analysis of Carrier Sense
Multiple Access With Collision Detection.
Proceedings of the LACN Symposium, May 1979.
New York: IEEE Computer Society Press, 1423-1447.

Tropper, C. (1981). Local Computer Network Technologies.
New York : Academic Press Inc.

Vijay, A. (1982). Design and Analysis of Computer Communication Network
New York : McGraw Hill.

Wong, E. (1983). Introduction to Random Processes.
New York : Springer-Verlag.

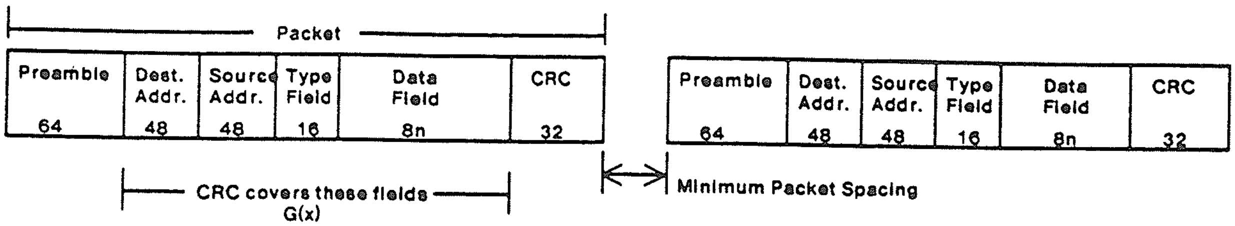
Young, Stephen J. (1982). Real Time Languages Design And Development.
Chichester, West Sussex : E. Horwood.

APPENDIX A

ETHERNET SPECIFICATION

Concise Ethernet Specification

Packet Format



Stations must be able to transmit and receive packets on the common coaxial cable with the indicated packet format and spacing. Each packet should be viewed as a sequence of 8-bit bytes; the least significant bit of each byte (starting with the preamble) is transmitted first.

Maximum Packet Size: 1526 bytes (8 byte preamble + 14 byte header + 1500 data bytes + 4 byte CRC)

Minimum Packet Size: 72 bytes (8 byte preamble + 14 byte header + 46 data bytes + 4 byte CRC)

Preamble: This 64-bit synchronization pattern contains alternating 1's and 0's, ending with two consecutive 1's. The preamble is: 10101010 10101010 10101010 10101010 10101010 10101010 10101010 10101011.

Destination Address: This 48-bit field specifies the station(s) to which the packet is being transmitted. Each station examines this field to determine whether it should accept the packet. The first bit transmitted indicates the type of address. If it is a 0, the field contains the unique address of the one destination station. If it is a 1, the field specifies a logical group of recipients; a special case is the broadcast (all stations) address, which is all 1's.

Source Address: This 48-bit field contains the unique address of the station that is transmitting the packet.

Type Field: This 16-bit field is used to identify the higher-level protocol type associated with the packet. It determines how the data field is interpreted

Data Field: This field contains an integral number of bytes ranging from 46 to 1500. (The minimum ensures that valid packets will be distinguishable from collision fragments.)

Packet Check Sequence: This 32-bit field contains a redundancy check (CRC) code, defined by the generating polynomial:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^{8} + x^{7} + x^{5} + x^{4} + x^{2} + x + 1$$

The CRC covers the address (destination/source), type, and data fields. The first transmitted bit of the destination field is the high-order term of the message polynomial to be divided by G(x) producing remainder R(x). The high-order term of R(x) is the first transmitted bit of the Packet Check Sequence field. The algorithm uses a linear feedback register which is initially preset to all 1's. After the last data bit is transmitted, the contents of this register (the remainder) are inverted and transmitted as the CRC field. After receiving a good packet, the receiver's shift register contains 11000111 00000100 11011101 01111011 (x^{31}, ... ,x^{0}).

Minimum Packet Spacing: This spacing is 9.6 usec, the minimum time that must elapse after one transmission before another transmission may begin.

Round-trip Delay: The maximum end-to-end, round-trip delay for a bit is 51.2 usec.

Collision Filtering: Any received bit sequence smaller than the minimum valid packet (with minimum data field) is discarded as a collision fragment.

Control Procedure

The control procedure defines how and when a host station may transmit packets into the common cable. The key purpose is fair resolution of occasional contention among transmitting stations.

Defer: A station must not transmit into the coaxial cable when carrier is present or within the minimum packet spacing time after carrier has ended.

Transmit: A station may transmit if it is not deferring. It may continue to transmit until either the end of the packet is reached or a collision is detected.

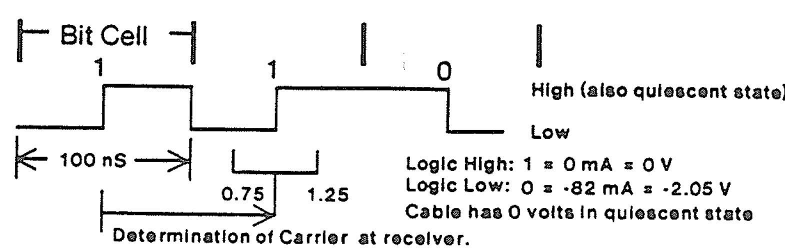
Abort: If a collision is detected, transmission of the packet must terminate, and a *jam* (4-6 bytes of arbitrary data) is transmitted to ensure that all other participants in the collision also recognize its occurrence.

Retransmit: After a station has detected a collision and aborted, it must wait for a random *retransmission delay*, defer as usual, and then attempt to retransmit the packet. The random time interval is computed using the backoff algorithm (below). After 16 transmission attempts, a higher level (e.g. software) decision is made to determine whether to continue or abandon the effort.

Backoff: Retransmission delays are computed using the *Truncated Binary Exponential Backoff* algorithm, with the aim of fairly resolving contention among up to 1024 stations. The delay (the number of time units) before the n^{th} attempt is a uniformly distributed random number from $[0 \text{ to } 2^{n}-1]$ for $0 < n \leq 10$ ($n = 0$ is the original attempt). For attempts 11-15, the interval is *truncated* and remains at [0 to 1023]. The unit of time for the retransmission delay is 512 bit times (51.2 usec).

Channel Encoding

Manchester encoding is used on the coaxial cable. It has a 50% duty cycle, and insures a transition in the middle of every bit cell ("data transition"). The first half of the bit cell contains the complement of the bit value, and the second half contains the true value of the bit.

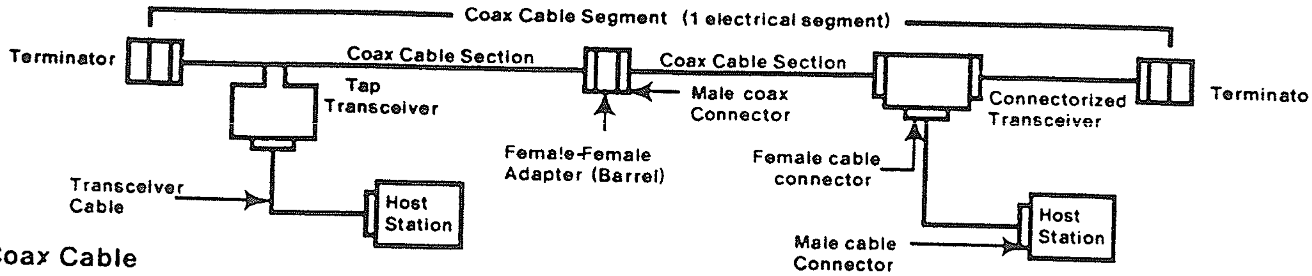


Data Rate

Data rate is 10 M bits/sec = 100 nsec bit cell ± 0.01%.

Carrier

The presence of data transitions indicates that carrier is present. If a transition is not seen between 0.75 and 1.25 bit times since the center of the last bit cell, then carrier has been lost, indicating the end of a packet. For purposes of deferring, carrier means any activity on the cable, independent of being properly formed. Specifically, it is any activity on either receive or collision detect signals in the last 160 nsec.



Coax Cable

Impedance: 50 ohms ± 2 ohms (Mil Std. C17-E). This impedance variation includes batch-to-batch variations. Periodic variations in impedance of up to ± 3 ohms are permitted along a single piece of cable.

Cable Loss: The maximum loss from one end of a cable segment to the other end is 8.5 db at 10 MHz (equivalent to ~500 meters of low loss cable).

Shielding: The physical channel hardware must operate in an ambient field of 2 volts per meter from 10 KHz to 30 MHz and 5 V/meter from 30 MHz to 1 GHz. The shield has a transfer impedance of less than 1 milliohm per meter over the frequency range of 0.1 MHz to 20 MHz (exact value is a function of frequency).

Ground Connections: The coax cable shield shall not be connected to any building or AC ground along its length. If for safety reasons a ground connection of the shield is necessary, it must be in only one place.

Physical Dimensions: This specifies the dimensions of a cable which can be used with the *standard tap*. Other cables may also be used, if they are not to be used with a tap-type transceiver (such as use with connectorized transceivers, or as a section between sections to which standard taps are connected).

Center Conductor:	0.0855" diameter solid tinned copper
Core Material:	Foam polyethylene or foam teflon FEP
Core O.D.:	0.242" minimum
Shield:	0.326" maximum shield O.D. (>90% coverage for outer braid shield)
Jacket:	PVC or teflon FEP
Jacket O.D.:	0.405"

Coax Connectors and Terminators

Coax cables must be terminated with male N-series connectors, and cable sections will be joined with female-female adapters. Connector shells shall be insulated such that the coax shield is protected from contact to building grounds. A sleeve or boot is acceptable. Cable segments should be terminated with a a female N-series connector (can be made up of a barrel connector and a male terminator) having an impedance of 50 ohms ± 1%, and able to dissipate 1 watt. The outside surface of the terminator should also be insulated.

Transceiver

CONNECTION RULES

Up to 100 transceivers may be placed on a cable segment no closer together than 2.5 meters. Following this placement rule reduces to a very low (but not zero) probability the chance that objectionable standing waves will result.

COAX CABLE INTERFACE

Input Impedance: The resistive component of the impedance must be greater then 50 Kohms. The total capacitance must be less than 4 picofarads.

Nominal Transmit Level: The important parameter is average DC level with 50% duty cycle waveform input. It must be -1.025 V (41 mA) nominal with a range of -0.9 V to -1.2 V (36 to 48 mA). The peak-to-peak AC waveform must be centered on the average DC level and its value can range from 1.4 V P-P to twice the average DC level. The voltage must never go positive on the coax. The quiescent state of the coax is logic high (0 V). Voltage measurements are made on the coax near the transceiver with the shield as reference. Positive current is current flowing out of the center conductor of the coax.

Rise and Fall Time: 25 nSec ± 5 nSec with a maximum of 1 nSec difference between rise time and fall time in a given unit. The intent is that dV/dt should not significantly exceed that present in a 10 MHz sine wave of same peak-to-peak amplitude.

Signal Symmetry: Asymmetry on output should not exceed 2 nSec for a 50-50 square wave input to either transmit or receive section of transceiver.

TRANSCEIVER CABLE INTERFACE

Signal Pairs: Both transceiver and host station shall drive and present at the receiving end a 78 ohm balanced load. The differential signal voltage shall be 0.7 volts nominal peak with a common mode voltage between 0 and +5 volts using power return as reference. (This amounts to shifted ECL levels operating between Gnd and +5 volts. A 10116 with suitable pulldown resistor may be used). The quiescent state of a line corresponds to logic high, which occurs when the + line is more positive than the - line of a pair.

Collision Signal: The active state of this line is a 10 MHz waveform and its quiescent state is logic high. It is active if the transceiver is transmitting and another transmission is detected, or if two or more other stations are transmitting, independent of the state of the local transmit signal.

Power: +11.4 volts to +16 volts DC at controller. Maximum current available to transceiver is 0.5 ampere. Actual voltage at transceiver is determined by the interface cable resistance (max 4 ohms loop resistance) and current drain.

ISOLATION

The impedance between the coax connection and the transceiver cable connection must exceed 250 Kohms at 60 Hz and withstand 250 VRMS at 60 Hz.

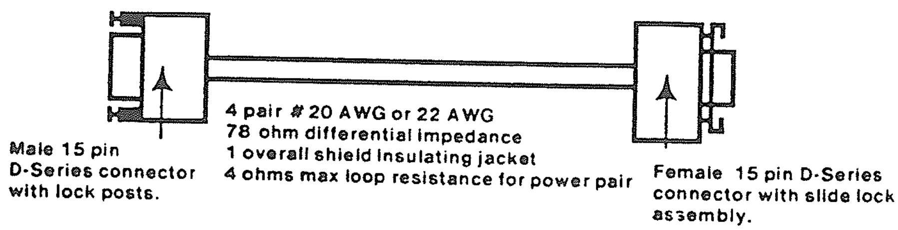
Transceiver Cable and Connectors

Maximum signal loss = 3 db @ 10 MHz. (equivalent to ~50 meters of either 20 or 22 AWG twisted pair).

Transceiver Cable Connector Pin Assignment

1.	Shield*		
2.	Collision +	9.	Collision -
3.	Transmit +	10.	Transmit -
4.	Reserved	11.	Reserved
5.	Receive +	12.	Receive -
6.	Power Return	13.	+ Power
7.	Reserved	14.	Reserved
8.	Reserved	15.	Reserved

*Shield must be terminated to connector shell.



APPENDIX B

COMPARSION BETWEEN EXPERIMENTIAL ETHERNET AND ETHERNET SPECIFICATION

Table B 1 summarizes the difference the between the Experimental Ethernet and Ethernet Specification, Version 2.0.

	Experimental Ethernet	Ethernet Specification
Data Rate	2.94 Mbps	10 Mbps
Maximum end to end length	1 km	2.5 km
Maximum segment length	1 km	500 m
Encoding	Manchester	Manchester
Coax. cable impedance	75 ohms	50 ohms
Coax. cable signal levels	0 to +3 V	0 to 2 V
Transceiver cable connectors	25 & 15 pin D series	15 pin D series
Length of preamble	1 bit	64 bits
Length of CRC	16 bits	32 bits
Length of address fields	8 bits	48 bits

Table B 1 Comparison of Experimental Ethernet and Ethernet Specification

APPENDIX C

USER GUIDE FOR SIMULATION

USER GUIDE FOR THE SIMULATION PROGRAM

The simulation can be executed by the following steps:

1) Sketch the configuration of the network.
2) Enter the appropriate network data into a input file.
3) Run the compiled version of the simulation program.

Step 1: Sketch network configuration

The configuration of the network must be first sketched. An example of a sketch configuration is illustrated in figure C-1. This configuration consists of two networks which are connected by a bridge. One segment and three nodes are located in network 2. Network 1 consists of two segments which connected by one local repeater, and a total of three nodes. The description of bridge, repeaters, and segments can be found in chapter 2. The repeaters are classified as local repeater (LOCAL), remote repeater (REMOTE) and multiport repeater (MULTI). The segments are classified as coaxial segment (COAX), link segment (LINK) and thinwire segment (THIN). The segment where the bridge connected is called spine segment. The spine segment is always defined as segment 1 in a network. One connection of both the repeater and bridge must be always connected to the spine segment in this simulation. Nodes can be classified as a device, such as a terminal server (TS), a bridge (B), a station (S), a computer device and so on. Different types of devices may have different characteristics including bandwidth limitation, acknowledgment requirement, packet length and number of packets per message. The source address for each node consists of network address, segment address and node address, and is represented by three numbers. For example, the source address for the node located on network 1, segment 2 and node number 1 is represented by (1,2,1). The bridge is considered to be a node in both networks. The connection of the bridge in network 1 is considered to be a specical node which called Node (1,1,2). The connection of the bridge in network 2 is called Node (2,1,3). Node(1,1,1), Node (2,1,1) and Node(2,1,2) are stations S. Node(1,2,1) is terminal server TS. Node(1,1,2) and Node (2,1,3) are B (bridge). The physical location for each node has to be assigned by the user.

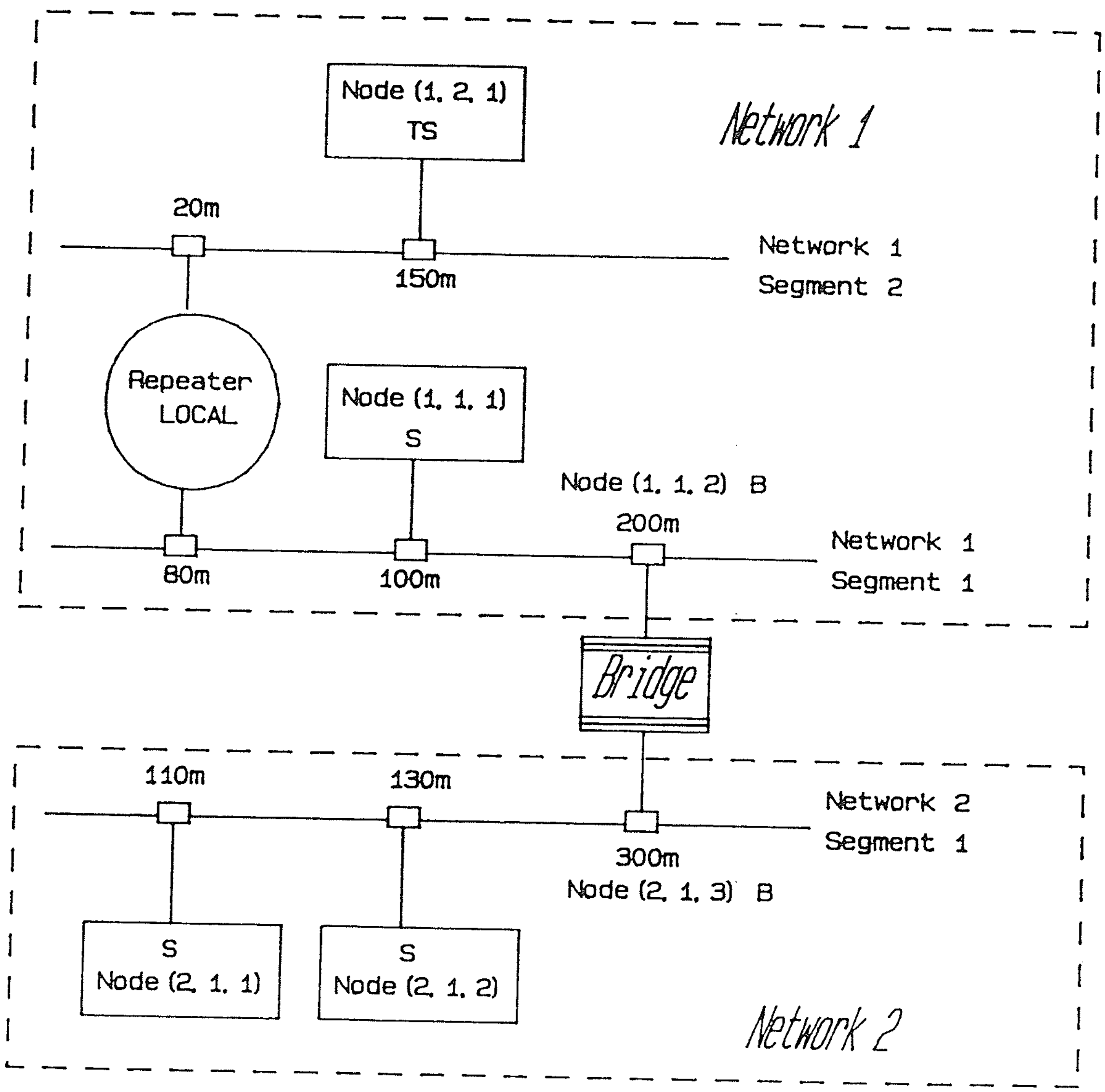


Figure C-1 An Example of Network Configuration.

Step 2: Enter the appropriate network data into an input file

Once the sketch of the network configuration is completed, the appropriate network data for the configuration is entered into an input file. A design aid program INPUT.COM was developed to assist the user in this regard. The listing of the INPUT.COM is given at the end of this appendix.

The INPUT.COM program is self explanatory and easy to use. Once the program executed, questions with detailed explanations are displayed on the terminal. When the execution of INPUT.COM is completed, the data is written into an input file called HSC.DAT. A typical example of the HSC.DAT based upon the previous configuration is shown in figure C-2.

The INPUT.COM is written in Digital Command Language (DCL) and can be executed in any VAX/VMS operation system by invoking the following command:

```
@disk:[directory]INPUT.COM
```

NOTE : If the user is familiar with the input data for the simulation program, the user can edit the input file HSC.DAT and manually input all the data instead of using the program INPUT.COM.

```
2,NET                        !  number of networks
1,BRIDGE                     !  number of bridge
1,1                          !-
1,2                          !  Table for the network path table.
2,1                          !
2,2                          !_
1,2,200,300,2,3              !‾ Bridge connection
1,0,0,2,0,0                  !- Number of repeaters and number of segments
0,0,0,1,0,0                  !_
1,1,-1,-1,-----              !‾ Segment connection
1,2,80,20,LOCAL              !  location of repeater
1,1,-1,-1,-----              !  Segment connection
1,NET                        !  Network 1
1,2,COAX                     !  Segment 1, number of nodes in segment, COAX
100 &                        !  Physical location of Node(1,1,1)
200                          !  Physical location of Node(1,1,2)
S &                          !  Device type of Node (1,1,1)
B                            !  Device type of Node (1,1,2)
2,1,COAX                     !  Segment 2, number of nodes in segment, COAX
150                          !  Physical location of Node(1,2,1)
TS                           !  Device type of Node (1,2,1)
2,NET                        !  Network 2
1,3,COAX                     !  Segment 1, number of nodes in segment, COAX
110 &                        !  Physical location of Node(2,1,1)
130 &                        !  Physical location of Node(2,1,2)
300                          !  Physical location of Node(2,1,3)
S &                          !  Device type of Node (2,1,1)
S &                          !  Device type of Node (2,1,2)
B                            !  Device type of Node (2,1,3)
NETPROB,0.0                  !- Network transmission probability
1,1,0.0,0.8                  !  Network 1 to network 1 in the range of 0.0,0.8
1,2,0.8,1.0                  !  Network 1 to network 2 in the range of 0.8,1.0
2,1,0.9,1.0                  !  Network 2 to network 1 in the range of 0.9,1.0
2,2,0.0,0.9                  !  Network 2 to network 2 in the range of 0.0,0.9
SEGMENTPROB,0.75             !  Probability to transmit within same segment
MAX_NUM_OF_INDICATORS,2000   ! Maximum number of transmissions.
TIMESLICE,10000000           ! The length of each time slice in bit-times
MAX_TIMESLICE,3              ! Maximum number of time slice
RESERVE_FACTOR_TIMESLICE,0.2! Reserved length for the time slice
NUM_DIFF_DEVICE,3            ! Number of difference devices
TS,400000,2,2000,T       !  Device name,bandwidth,packets per messages, and
S,50000,1,1000,F         !  acknowledgment flag.
B,0,0,0,F
RESERVE_FACTOR_SET_TX,0.0 ! Reserved factor of the offered load.
NUM_OF_SET_TX,5           ! Offered transmissions in terms of how many
                          ! set of nodes.
```

Figure C-2 An typical example of input data file.

Step 3: Executed Simulation Program

The BASIC source file of the simulation program is called HSC.BAS. Once the input file HSC.DAT is formatted, the simulation program can be executed by submitting its image file HSC.EXE into a batch queue.

The following steps are executed in order to obtain the image file HSC.EXE from the BASIC source file HSC.BAS.

i) Compile the source basic file HSC.BAS to obtain a object file HSC.OBJ using the following DCL command:
   BASIC/LIST/OBJECT/SYNTAX   HSC.BAS

ii) Link the object file HSC.OBJ to obtain the image file HSC.EXE.
   LINK HSC.OBJ

The image file HSC.EXE can be submitted to the batch queue by the following steps.

i) Create a submitting file. For example, BAT.COM

ii) The BAT.COM file consists of the following statement.
   RUN   disk:[directory]HSC.EXE

iii) Submit the BAT.COM into the batch queue VAX$BATCH.
   SUBMIT/NOLOG/QUEUE=VAX2$BATCH   BAT.COM

Once the BAT.COM is submitted in the batch queue, the image file HSC.EXE will be executed. The simulation results are recorded in the output files which are illustrated in section 4.3.4.

```
$ !------------------------------------------------------------------------
$ ! This is command procedure   I N P U T . C O M
$ !
$ ! Local Area Network Simulation Input Data.
$ !
$ ! Note: This command procedure is specified for use with
$ !       DEC VT-200 (compatable) terminal.
$ !
$ !               created by Gilbert Tang.
$ !
$ !------------------------------------------------------------------------
$ !
$   ON ERROR          THEN GOTO ENDING
$   ON SEVERE_ERROR THEN GOTO ENDING
$   ON WARNING        THEN GOTO ENDING
$   ON CONTROL_Y      THEN GOTO ENDING
$ !
$ !------------------------------------------------------------------------
$ ! S E T U P
$ !
$   WOUT    == "WRITE SYS$OUTPUT"
$   WFILE   == "WRITE OUTFILE"
$   BEL[0,7]     = %X07
$   ESC[0,8]     = %X1B
$   SET_NORMAL  = "''ESC'[1;0m"     ! normal
$   SET_FLASH   = "''ESC'[0;7m"     ! no blinking & background
$   SET_NOFLASH = "''ESC'[0;0m"     ! no blinking
$   TOP ="''ESC'#3"                 ! double size char. (top)
$   BOT ="''ESC'#4"                 ! double size char. (bottom)
$   BIG ="''ESC'#6"
$   GRA ="''ESC'(0"
$   US  ="''ESC'(B"
$   WOUT US
$   LINE1 = "ssssssssssssssssssss"
$   LINE2 = "oooooooooooooooooooo"
$   BRIKE = "vwvwvwvwvwvwvwvwvwvwvwvwvwvwvw"
$ !
$ !------------------------------------------------------------------------
```

```
$ !-----------------------------------------------------------------------
$ ! M A I N     P R O G R A M
$ !
$   GOSUB HEADER
$
$   GOSUB GRAPH
$
$   GOSUB INFILE
$
$   GOSUB NET_BRIDGE
$
$   GOSUB NET_TABLE
$
$   GOSUB BRIDGE_CONNECTION
$
$   GOSUB NUMBER_SEG_REPEATER
$
$   GOSUB REPEATER_LOCATION
$
$   GOSUB NODE_LOCATION
$
$   GOSUB NET_PROB
$
$   GOSUB SEG_PROB
$
$   GOSUB INPUT_TIMESLICE
$
$   GOSUB DIFFERENT_DEVICES
$
$   GOSUB TRANSMISSION_INFO
$
$   GOSUB ENDING
$
$ !-----------------------------------------------------------------------
```

```
$ !----------------------------------------------------------------
$ ! H E A D E R
$ !
$   HEADER:
$      WOUT "''ESC'[0;0H''ESC'[2J''ESC'[8;24r"
$      WOUT BEL,BEL
$      MESSAGE = "         LOCAL AREA NETWORK SIMULATION     "
$      WOUT TOP,SET_FLASH ,MESSAGE
$      WOUT BOT,SET_FLASH ,MESSAGE,SET_NOFLASH
$      WOUT BIG,GRA,LINE1,LINE1
$      WOUT BRIKE,"u",US,"<  By Gilbert Tang   >",GRA,"t",BRIKE
$      WOUT BIG,LINE2,LINE2,US
$      WAIT 00:00:01
$
$   RETURN
$ !----------------------------------------------------------------
$ !
$ ! I N F I L E :
$ !
$   INFILE:
$
$      SUB_NAME  == "I N F I L E "
$      GOSUB SUBHEADER
$
$      INQUIRE FILENAME "Enter the filename for the LAN data"
$      IF FILENAME .EQS. "" THEN FILENAME = "HSC.DAT"
$      DIRDEF = F$DIRECTORY()
$      OPEN/WRITE OUTFILE 'DIRDEF''FILENAME'
$      !WFILE DIRDEF,FILENAME
$
$   RETURN
$ !
$ !----------------------------------------------------------------
```

```
$ !-----------------------------------------------------------------------
$ ! G R A P H
$
$  GRAPH:
$
$    WOUT GRA,"   lqqqk                   lqqqk     lqqqk         "
$    WOUT GRA,"   x   x    NODE qqqq>     x   x     x   x         "
$    WOUT GRA,"   mqwqj                   mqwqj     mqwqj         "
$    WOUT GRA,"     x                       x         x           "
$    WOUT GRA,"aqqqqvqqqqqwqqqqqqqqqqwqqqqqqvqqqqqqqqqvqqqqqqqqqwqqa"
$    WOUT GRA,"           x          x          ^               x   "
$    WOUT GRA,"         lqvqk      lqvqk        x             lqvqk "
$    WOUT GRA,"         x   x      x   x        x             x   x "
$    WOUT GRA,"         mqqqj      mqqqj    COAXIAL SEGMENT   mqqqj "
$    WOUT US
$  RETURN
$
$ !-----------------------------------------------------------------------
$ !
$ ! N E T _ B R I D G E
$
$  NET_BRIDGE:
$
$     SUB_NAME  == "N E T _ B R I D G E "
$     GOSUB SUBHEADER
$
$     WOUT ""
$     INQUIRE NUMNETWORK "Enter the number of network(s) "
$     INQUIRE NUMBRIDGE  "Enter the number of bridge(s)  "
$
$     WFILE NUMNETWORK,",NET"
$     WFILE NUMBRIDGE ,",BRIDGE"
$
$  RETURN
$
$ !-----------------------------------------------------------------------
```

```
$ !------------------------------------------------------------------
$ !
$ ! N E T _ T A B L E
$
$   NET_TABLE:
$
$     SUB_NAME  == "N E T W O R K   T A B L E "
$     GOSUB SUBHEADER
$     WOUT ""
$     WOUT ""
$     WOUT "Example: CREATE THE NETWORK TABLE"
$     WOUT     "                                  ",US,-
               "Enter netpath or (E)xit :1,1   "
$     WOUT     " Network Configuration          ",US,-
               "Enter netpath or (E)xit :1,2   "
$     WOUT     "                                  ",US,-
               "Enter netpath or (E)xit :1,2,4"
$     WOUT GRA," lqqqk     lqqqk     lqqqk ",US,-
               "Enter netpath or (E)xit :1,3   "
$     WOUT GRA," x 1 tqqqqqu 2 tqqqqu 4 x ",US,-
               "Enter netpath or (E)xit :2,1   "
$     WOUT GRA," mqwqj     mqqqj     mqqqj ",US,-
               "Enter netpath or (E)xit :2,2   "
$     WOUT GRA,"   x                          ",US,-
               "Enter netpath or (E)xit :2,4   "
$     WOUT GRA," lqvqk                        ",US,-
               "Enter netpath or (E)xit :3,1   "
$     WOUT GRA," x 3 x                        ",US,-
               "Enter netpath or (E)xit :3,3   "
$     WOUT GRA," mqqqj                        ",US,-
               "Enter netpath or (E)xit :4,2,1"
$     WOUT     "                              ",US,-
               "Enter netpath or (E)xit :4,2   "
$     WOUT     "                              ",US,-
               "Enter netpath or (E)xit :4,4   "
$     WOUT     "                              ",US,-
               "Enter netpath or (E)xit :E     "
$     WOUT     ""
$
$     INPUTPATH:
$     INQUIRE NETPATH "Enter netpath or (E)xit"
$     IF NETPATH .NES. "" .AND. NETPATH .NES. "E" THEN WFILE NETPATH
$     IF NETPATH .NES. "E" THEN GOTO  INPUTPATH
$     WOUT "
$
$   RETURN
$
$ !------------------------------------------------------------------
```

```
$ !------------------------------------------------------------------
$ !
$ ! B R I D G E _ C O N N E C T I O N
$
$   BRIDGE_CONNECTION:
$
$       SUB_NAME  == "B R I D G E   C O N N E C T I O N"
$       GOSUB SUBHEADER
$
$       BRIDGE_LOOPIX = NUMBRIDGE
$       IF BRIDGE_LOOPIX .EQ. 0 THEN GOTO SKIPBRIDGE
$
$       WOUT "THE CONNECTION BETWEEN THE NETWORKS AND THE BRIDGES:"
$       WOUT ""
$       WOUT "Bridge is used to connect two networks (net A and net B)"
$       WOUT ""
$       WOUT GRA," lqqqqqqqqqqk     lqqqqqqqk    lqqqqqqqqqqk "
$       WOUT GRA," x            x     x         x    x            x "
$       WOUT GRA," x NETWORK A tqqqqqu BRIDGE tqqqqu NETWORK B x "
$       WOUT GRA," x            x     x         x    x            x "
$       WOUT GRA," mqqqqqqqqqqj     mqqqqqqqj    mqqqqqqqqqqj "
$       WOUT US
$
$       BRIDGE_CONNECT_LOOP:
$       IF BRIDGE_LOOPIX .EQ. 0 THEN GOTO SKIPBRIDGE
$
$       WOUT "For bridge ",NUMBRIDGE-BRIDGE_LOOPIX+1
$       INQUIRE NETA "Bridge Connects from Net A (Enter network number)"
$       INQUIRE NETB "                        to Net B (Enter network number)"
$       WOUT         ""
$       INQUIRE PHYA "Physical location of bridge in Net A"
$       INQUIRE PHYB "Physical location of bridge in Net B"
$       WOUT         ""
$       INQUIRE NODA "Node  number  of the bridge in Net A"
$       INQUIRE NODB "Node  number  of the bridge in Net B"
$       WOUT         ""
$
$       WFILE NETA,",",NETB,",",PHYA,",",PHYB,",",NODA,",",NODB
$
$       BRIDGE_LOOPIX = BRIDGE_LOOPIX - 1
$       GOTO BRIDGE_CONNECT_LOOP
$   SKIPBRIDGE:
$   RETURN
$
$ !------------------------------------------------------------------
```

```
$ !------------------------------------------------------------------
$ !
$ ! N U M B E R _ S E G _ R E P E A T E R
$
$   NUMBER_SEG_REPEATER:
$
$      SUB_NAME  == "S E G M E N T   A N D   R E P E A T E R"
$      GOSUB SUBHEADER
$
$      SEG_R_LOOPIX = NUMNETWORK
$      IF SEG_R_LOOPIX .EQ. 0 THEN GOTO SKIP_SEG_R
$
$      WOUT GRA," lqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqk"
$      WOUT GRA," x                                              x"
$      WOUT " x",US,"  For each network, input the following:    ",GRA,"x"
$      WOUT " x",US,"  1) the number of local repeaters,         ",GRA,"x"
$      WOUT " x",US,"  2) the number of remote repeaters,        ",GRA,"x"
$      WOUT " x",US,"  3) the number of multiport repeaters,     ",GRA,"x"
$      WOUT " x",US,"  4) the number of coaxial segments,        ",GRA,"x"
$      WOUT " x",US,"  5) the number of link segments,           ",GRA,"x"
$      WOUT " x",US,"  6) the number of thinwire segments.       ",GRA,"x"
$      WOUT GRA," x                                              x"
$      WOUT GRA," mqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqqj"
$      WOUT US
$
$      SEG_R_LOOP:
$      IF SEG_R_LOOPIX .EQ. 0 THEN GOTO SKIP_SEG_R
$
$         WOUT "For network :",NUMNETWORK-SEG_R_LOOPIX+1
$         INQUIRE LOCALR  "Enter number of local repeaters      "
$         INQUIRE REMOTE  "Enter number of remote repeaters     "
$         INQUIRE MULTIP  "Enter number of multiport repeaters "
$         INQUIRE COAX    "Enter number of coaxial segments     "
$         INQUIRE LINK    "Enter number of link segments        "
$         INQUIRE THIN    "Enter number of thinwire segments    "
$         WOUT ""
$
$         WFILE LOCALR,",",REMOTE,",",MULTIP,",",COAX,",",LINK,",",THIN
$
$         SEG_R_LOOPIX = SEG_R_LOOPIX - 1
$         GOTO SEG_R_LOOP
$   SKIP_SEG_R:
$   RETURN
$
$ !------------------------------------------------------------------
```

```
$ !-----------------------------------------------------------------------------
$ !
$ ! R E P E A T E R _ L O C A T I O N
$ !
$   REPEATER_LOCATION:
$
$     SUB_NAME  == "R E P E A T E R    L O C A T I O N"
$     GOSUB SUBHEADER
$
$     WOUT ""
$     WOUT "Enter the interconnection of one network"
$     WOUT "Example: "
$     WOUT GRA,"      2                    3        4 5 6  ",-
              " A=LOCAL REPEATER"
$     WOUT GRA," qqqqqqqqwqqq    qqqqwqqqqq       x x x  ",-
              " B=REMOTE REPEATER"
$     WOUT GRA,"          x           x           x x x  ",-
              " C=MULTIPORT REPEATER"
$     WOUT GRA,"          x           x           x x x  ",-
              " 1=SPINAL COAXIAL SEG."
$     WOUT GRA,"       lqqvqqk     lqqvqqk       lvqvqvk ",-
              " 2,3=COAXIAL SEGMENT
$     WOUT GRA,"       x  A  x     x  B  x       x  C  x ",-
              " 4,5,6=THINWIRE SEGMENT"
$     WOUT GRA,"       mqqwqqj     mqqwqqj       mqqwqqj "
$     WOUT GRA,"          x           x             x    "
$     WOUT GRA," qqqqqqqqqvqqqqqqqqqqqvqqqqqqqqqqqqqqvqqqq"
$     WOUT GRA,"                      1                  "
$     WOUT US
$     WOUT ""
$
$     R_LOOPIX = NUMNETWORK
$     NET_REPEAT_LOOP:
$     IF R_LOOPIX .EQ. 0 THEN GOTO END_REPEATER_LOOP
$     WFILE "1,1,-1,-1,-----"
$
```

```
$      MOREREPEATER:
$      WOUT "For network: ",NUMNETWORK-R_LOOPIX+1
$      INQUIRE/NOPUN  R_T -
                 "Is there any repeaters for this network ? "
$      IF .NOT. R_T THEN GOTO R_HERE
$      WOUT "The repeater connects the spinal segment (segment 1) to"
$      INQUIRE SEGNUM "segment number ?"
$      INQUIRE LOA -
          "Enter the location of repeater connects the spinal segment"
$      INQUIRE LOB -
          "Enter the location of repeater connects the segment ''SEGNUM'"
$      INQUIRE RTYPE "Enter the type of repeaters (LOCAL,REMOTE,MULTI)"
$      WFILE "1,",SEGNUM,",",LOA,",",LOB,",",RTYPE
$      INQUIRE MORER "Enter next repeater's information ? (y or n)"
$      IF MORER THEN GOTO MOREREPEATER
$      INQUIRE THIN1 "Which number is the first thinwire or none <CR>"
$      INQUIRE THIN2 "Which number is the last  thinwire or none <CR>"
$      IF THIN1 .NES. "" THEN WFILE THIN1,",",THIN2
$      WOUT ""
$
$      R_HERE:
$      R_LOOPIX = R_LOOPIX - 1
$      GOTO NET_REPEAT_LOOP
$
$      END_REPEATER_LOOP:
$      RETURN
$
$ !------------------------------------------------------------------
```

```
$ !--------------------------------------------------------------------
$ ! N O D E _ L O C A T I O N
$ !
$   NODE_LOCATION:
$
$      SUB_NAME  == "N O D E    L O C A T I O N"
$      GOSUB SUBHEADER
$
$      WOUT ""
$      WOUT "N O D E    L O C A T I O N"
$
$      NET_LOOPIX = NUMNETWORK
$
$      NET_NLOOP:
$      IF NET_LOOPIX .EQ. 0 THEN GOTO END_NET
$      PRESENT_NETNUM = NUMNETWORK-NET_LOOPIX+1
$      WFILE PRESENT_NETNUM,",","NET"
$
$      WOUT ""
$      WOUT "For network: ''PRESENT_NETNUM'"
$      INQUIRE/NOPUN TOTALSEG -
       "How many segments in network ''PRESENT_NETNUM' again ?  "
$
$      TSEG = TOTALSEG
$
$      SEG_NLOOP:
$         IF TSEG .EQ. 0 THEN GOTO END_SEG
$         PRESENT_SEG = TOTALSEG-TSEG+1
$         INQUIRE NUMOFNODE -
          "Enter number of nodes for segment ''PRESENT_SEG'   "
$         INQUIRE TYPESEG "Enter type of segment ''PRESENT_SEG' "-
                          "(COAX or THIN)"
$
$         WFILE PRESENT_SEG,",",NUMOFNODE,",",TYPESEG
```

```
$
$         LOCLOOP = NUMOFNODE
$         LOCLOOP_HERE:
$         IF LOCLOOP .EQ. 0 THEN GOTO END_LOC
$            INQUIRE NODELOC "Enter node location"
$            IF LOCLOOP .NE. 1 THEN WFILE NODELOC," &"
$            IF LOCLOOP .EQ. 1 THEN WFILE NODELOC
$            LOCLOOP = LOCLOOP-1
$         GOTO LOCLOOP_HERE
$         END_LOC:
$
$
$         TYPELOOP = NUMOFNODE
$         TYPELOOP_HERE:
$         IF TYPELOOP .EQ. 0 THEN GOTO END_TYPE
$            INQUIRE NODETYPE "Enter node type"
$            IF TYPELOOP .NE. 1 THEN WFILE NODETYPE," &"
$            IF TYPELOOP .EQ. 1 THEN WFILE NODETYPE
$            TYPELOOP = TYPELOOP-1
$         GOTO TYPELOOP_HERE
$         END_TYPE:
$
$         TSEG = TSEG -1
$      GOTO SEG_NLOOP
$      END_SEG:
$
$      NET_LOOPIX = NET_LOOPIX-1
$      GOTO NET_NLOOP
$   END_NET:
$
$   RETURN
$ !-----------------------------------------------------------------------
```

```
$ !----------------------------------------------------------------
$ ! N E T _ P R O B
$ !
$   NET_PROB:
$
$      SUB_NAME  == "N E T W O R K    P R O B A B I L I T Y"
$      GOSUB SUBHEADER
$      WOUT GRA,"      \TO          NETWORK          "
$      WOUT GRA,"   FROM\   1     2              N    ",US,"EXAMPLE: "
$      WOUT GRA,"         lqqqqqwqqqqqw wqqqqqk"
$      WOUT GRA,"         x 0.0 x 0.90x          x 0.XXx",US,-
            "From net number: 1"
$      WOUT GRA,"      1  x 0.9 x 0.91x          x 1.0 x",US,-
            "  to net number: 1"
$      WOUT GRA," N       tqqqqqnqqqqqu tqqqqqu",US,-
            "Enter the range for prob.: 0.0,0.9"
$      WOUT GRA," E       x 0.8 x 0.0 x          x 0.XXx"
$      WOUT GRA," T   2   x 0.82x 0.8 x          x 1.0 x"
$      WOUT GRA," W       tqqqqqnqqqqqu tqqqqqu"
$      WOUT GRA," O
$      WOUT GRA," R
$      WOUT GRA," K       tqqqqqnqqqqqu tqqqqqu"
$      WOUT GRA,"         x 0.71x 0.7 x          x 0.0 x"
$      WOUT GRA,"      N  x 0.75x 0.71x          x 0.7 x"
$      WOUT GRA,"         mqqqqqvqqqqqj mqqqqqj"
$      WOUT US
$
$      WFILE "NETPROB,0.0"
$
$      MORENET_PROB:
$      INQUIRE NETA "From net number"
$      INQUIRE NETB "  to net number"
$      INQUIRE RANGE "Enter the range for prob. (e.g 0.1,0.8)"
$      WFILE NETA,",",NETB,",",RANGE
$      INQUIRE MORE  "Input next entry for net probability table ?"-
                     "(y or n)"
$      WOUT ""
$      IF MORE THEN GOTO MORENET_PROB

$   RETURN
$ !----------------------------------------------------------------
```

```
$ !-----------------------------------------------------------------
$ !
$ ! S E G _ P R O B
$
$   SEG_PROB:
$
$      SUB_NAME  == "S E G M E N T   P R O B A B I L I T Y"
$      GOSUB SUBHEADER
$
$      WOUT  ""
$      WOUT  "S E G M E N T   P R O B A B I L I T Y"
$      WOUT  ""
$      WOUT  "Enter the probability for the transmission within "
$      INQUIRE SEGP "the same segment"
$
$      WFILE "SEGMENTPROB,",SEGP
$
$   RETURN
$
$ !-----------------------------------------------------------------
$ !
$ ! I N P U T _ T I M E S L I C E
$ !
$   INPUT_TIMESLICE:
$
$      SUB_NAME  == "T I M E S L I C E   D A T A
$      GOSUB SUBHEADER
$
$      WOUT  ""
$      WOUT  "T I M E S L I C E   D A T A"
$      WOUT  ""
$      INQUIRE MAX_INDICATORS -
       "Enter the maximum number of transmissions allows in the program"
$      WFILE "MAX_NUM_OF_INDICATORS,",MAX_INDICATORS
$
$      INQUIRE LEN_TIMESLICE  -
       "Enter the length of the time slice (in bit time)"
$      WFILE "TIMESLICE,",LEN_TIMESLICE
$
$      INQUIRE MAX_TIMESLICE  -
       "Enter the maximum number of time slices"
$      WFILE "MAX_TIMESLICE,",MAX_TIMESLICE
$
$      INQUIRE TIMESLICE_FACTOR  -
       "Enter the reserve factor for the time slice"
$      WFILE "RESERVE_FACTOR_TIMESLICE,",TIMESLICE_FACTOR
$
$   RETURN
$
$ !-----------------------------------------------------------------
```

```
$ !-----------------------------------------------------------------
$ !
$ ! D I F F E R E N T _ D E V I C E S
$
$   DIFFERENT_DEVICES:
$
$       SUB_NAME  == "D I F F E R E N T   D E V I C E S"
$       GOSUB SUBHEADER
$
$       WOUT  ""
$       WOUT  "D I F F E R E N T   D E V I C E S"
$       WOUT  ""
$       INQUIRE NUM_DIFF_DEV -
        "Enter the number of different devices"
$       WFILE "NUM_DIFF_DEVICE,",NUM_DIFF_DEV
$
$       NUMDEVS = NUM_DIFF_DEV
$
$       NUMDEVS_LOOP:
$       WOUT " "
$       IF NUMDEVS .EQ. 0 THEN GOTO END_NUMDEVS
$       WOUT "For the device number ",NUM_DIFF_DEV-NUMDEVS+1
$       INQUIRE TYPENAME -
             "Enter the device type"
$       INQUIRE BW -
             "Enter the I/O bandwidth for this device"
$       INQUIRE NUM_PACKETS -
             "Enter the number of packets per message for this device"
$       INQUIRE PACKET_LENGTH -
             "Enter the packet length (in bit time)"
$       INQUIRE REPLY "Enter whether a acknowlegment after a"-
                      " message is transmitted (T/F)"
$
$       WFILE TYPENAME,",",BW,",",NUM_PACKETS,",",PACKET_LENGTH,-
                      ",",REPLY
$
$       NUMDEVS = NUMDEVS - 1
$       GOTO NUMDEVS_LOOP
$
$   END_NUMDEVS:
$   RETURN
$
$ !-----------------------------------------------------------------
```

```
$ !-----------------------------------------------------------------
$ !
$ ! T R A N S M I S S I O N _ I N F O
$
$   TRANSMISSION_INFO:
$
$       SUB_NAME  == "T R A N S M I S S I O N   D A T A"
$       GOSUB SUBHEADER
$
$       WOUT ""
$       WOUT "T R A N S M I S S I O N   D A T A"
$       WOUT ""
$       WOUT "One set of transmission is equal to total number of nodes."
$       WOUT "Reserve factor is the fraction of transmission set"-
             " reserved."
$       WOUT ""
$
$       INQUIRE SET_TX  "Enter number of set of transmissions"
$       WOUT            "Enter the reserve factor for the number "
$       INQUIRE RES_SET "of set of transmissions (0.0 - 1.0)"
$
$       WFILE "RESERVE_FACTOR_SET_TX,",RES_SET
$       WFILE "NUM_OF_SET_TX,",SET_TX
$
$   RETURN
$
$ !-----------------------------------------------------------------
$ !
$ ! S U B H E A D E R
$ !
$   SUBHEADER:
$
$       TOTAL_LEN == 40
$       SUB_LENG  == F$LENGTH(SUB_NAME)
$
$       BLANK_LEN == SUB_LENG/2
$       BLANK_STR == F$FAO("!#AS",(TOTAL_LEN-SUB_LENG)/2," ")
$       SUB_OUT   == F$FAO("!#AS",TOTAL_LEN,BLANK_STR+SUB_NAME)
$
$       WOUT "''ESC'7"
$       WOUT "''ESC'[6;OH''ESC'[1;5m''ESC'#6",SUB_OUT
$       WOUT "''ESC'8"
$
$   RETURN
$
$ !-----------------------------------------------------------------
```

```
$ !-----------------------------------------------------------------------
$ !
$ ! E N D I NG
$ !
$   ENDING:
$      CLOSE OUTFILE
$
$      WOUT ""
$      INQUIRE/NOPUN PRINTOUT "Do you like to print the file ? "
$      IF PRINTOUT THEN LASER 'DIRDEF''FILENAME'
$
$      WOUT "''ESC'[0;0H''ESC'[2J''ESC'[0;24r"
$      MESSAGE = "                      GOOD LUCK                    "
$      WOUT TOP,SET_FLASH ,MESSAGE
$      WOUT BOT,SET_FLASH ,MESSAGE,SET_NOFLASH
$      WOUT ""
$
$   EXIT
$ !
$ !_______________________________________________________________________
```

APPENDIX D

POISSON DISTRIBUTION TEST

This appendix describes a statistical test of the simulation program. If the transmissions are made rare, and are randomly distributed in the time slice, the transmissions should be Poisson in their distribution [Bratley et al, 1983].

The input to the simulation program was designed to make the transmissions rare and random. In this way, the distribution of transmissions in the time slice would be expected to be Poisson in nature. Output file HSC.OUT4 was used to test this expectation. This file was based upon a time slice which was consisted of 10,000,000 bit times, and the number of transmissions in the time slice was set to 1000. The time slice was arbitrary divided into 1000 sampling cells, such that the mean (m) was equal to 1.0. Thus,

$$\text{Mean (m)} = \frac{\text{Number of transmissions}}{\text{Number of cells}} = \frac{1000}{1000} = 1.0$$

and each cell consists of 10,000 bit times.

Each cell was examined in order to find out the number of transmissions contained within the cell. The next step was to count the occurrence of the number of transmissions. Table D-1 compares the values of the Poisson function (m=1) with the simulation results. The simulation results were close to the values of the Poisson function (m=1).

Number of transmissions (x)	Number of occurrence	Normalized number of occurrence	Expected value f(x) m=1.0
0	388	0.388	0.3679
1	343	0.343	0.3679
2	173	0.173	0.1839
3	74	0.074	0.0613
4	21	0.021	0.0153
5	1	0.001	0.0031
6	0	0.000	0.0005
7	0	0.000	0.0001

Table D-1 Comparison of Poisson function (m=1) with Poisson result from simulation.

The Chi-Square test was used to compare the two frequencies. It was concluded that with a Chi-Square sum of 0.01 and 5 degrees of freedom, the simulation correctly generated a Poisson distribution with a confidence level of greater than 99.5 percent.

::::::::::::::::::
::: APPENDIX E :::
::::::::::::::::::

APPENDIX E

DTS RESULTS FOR DIFFERENT MESSAGE SIZES

The DTS Data Test utility was used to find the appropriate message size for the experiment in section 5.3. DTS sent packets from VAX2 (VAX 8650) to VAX1 (VAX 785) with message sizes ranging from 100 bytes to 4096 bytes. The results were recorded in both the NCP Line counters and the DTS counters. Tables E.1a to E.1c show the results from the counters. A typical example of this DTS Data Test is shown as follows:

```
$    RUN SYS$SYSTEM:NCP
         ZERO KNOWN LINE COUNTER
         SHOW KNOWN LINE COUNTER
         TELL VAX1"GILBERT password":: ZERO KNOWN LINE COUNTER
         TELL VAX1"GILBERT password":: SHOW KNOWN LINE COUNTER
$    RUN SYS$SYSTEM:DTSEND
         DATA/NODENAME=VAX1/PRINT/SIZE= variable /SEC=10
$    RUN SYS$SYSTEM:NCP
         SHOW KNOWN LINE COUNTER
         TELL VAX1"GILBERT password":: SHOW KNOWN LINE COUNTER
```

Message size	Number of messages	NO. of D.B. TX (RX)	Bytes Sent	Bytes Received
100	1948	1989 (1209)	261244	56065
300	1809	1845 (1238)	604261	57383
500	1473	1501 (922)	786592	42535
700	1497	1525 (950)	1098668	43827
900	1274	1301 (797)	1189963	37089
1100	1189	1225 (746)	1350323	34961
1300	1298	1324 (805)	1731509	37149
1450	1204	1231 (738)	1786969	34067
1480	864	1755 (920)	1338513	42439
1500	806	1643 (869)	1265149	40101
1700	658	1345 (718)	1163557	33155
1900	723	1478 (789)	1424417	36425
2100	757	1544 (832)	1641743	39205
2300	766	1561 (869)	1815217	40101
2500	804	1636 (928)	2065897	42811
2700	726	1482 (867)	2011139	40005
2900	744	1516 (896)	2209519	41339
3100	606	1845 (968)	1939915	44651
3300	567	1731 (883)	1931704	41041
3500	599	1823 (976)	2158524	45011
3700	483	1477 (786)	1839180	36275
3900	465	1425 (747)	1864512	35287
4096	561	1574 (850)	2165895	39219

Table E.1a DTS Data Test results

D.B. = data block
TX = transmitted
RX = received

Message size	Number of messages	NO. of D.B. TX (RX)	Data Block size (TX)ave	Data Block size (Rx)ave
100	1948	1989 (1209)	131.34	46.37
300	1809	1845 (1238)	327.51	46.35
500	1473	1501 (922)	524.04	46.13
700	1497	1525 (950)	720.43	46.13
900	1274	1301 (797)	914.65	46.54
1100	1189	1225 (746)	1102.30	46.86
1300	1298	1324 (805)	1307.78	46.14
1450	1204	1231 (738)	1451.64	46.16
1480	864	1755 (920)	762.68	46.13
1500	806	1643 (869)	770.02	46.14
1700	658	1345 (718)	865.09	46.17
1900	723	1478 (789)	963.75	46.17
2100	757	1544 (832)	1063.31	47.12
2300	766	1561 (869)	1162.86	46.14
2500	804	1636 (928)	1262.77	46.13
2700	726	1482 (867)	1357.04	46.14
2900	744	1516 (896)	1457.47	46.14
3100	606	1845 (968)	1051.44	46.13
3300	567	1731 (883)	1115.94	46.48
3500	599	1823 (976)	1184.05	46.11
3700	483	1477 (786)	1245.21	46.15
3900	465	1425 (747)	1308.43	47.23
4096	561	1574 (850)	1376.05	46.14

Table E.1b DTS Data Test results

ave = on average

Message size	Number of messages	NO. of D.B. TX (RX)	Data Block size (TX)	Data Block size (Rx)ave
100	1948	1989 (1209)	131.34	46.37
300	1809	1845 (1238)	327.51	46.35
500	1473	1501 (922)	524.04	46.13
700	1497	1525 (950)	720.43	46.13
900	1274	1301 (797)	914.65	46.54
1100	1189	1225 (746)	1102.30	46.86
1300	1298	1324 (805)	1307.78	46.14
1450	1204	1231 (738)	1451.64	46.16
1480	864	1755 (920)	1480 (45.36)	46.13
1500	806	1643 (869)	1498 (42.04)	46.14
1700	658	1345 (718)	1498 (232.0)	46.17
1900	723	1478 (789)	1498 (429.5)	46.17
2100	757	1544 (832)	* (624.62)	47.12
2300	766	1561 (869)	* (827.72)	46.14
2500	804	1636 (928)	*(1027.54)	46.13
2700	726	1482 (867)	*(1216.00)	46.14
2900	744	1516 (896)	*(1416.94)	46.14
3100	606	1845 (968)	** (158.32)	46.13
3300	567	1731 (883)	** (351.82)	46.48
3500	599	1823 (976)	** (556.15)	46.11
3700	483	1477 (786)	** (739.63)	46.15
3900	465	1425 (747)	** (929.29)	47.23
4096	561	1574 (850)	**(1132.15)	46.14

Table E.1c DTS Data Test results

* = 1498 bytes
** = 2996 bytes

APPENDIX F

MEASURED STATISTICS OF THE EXPERIMENTAL NETWORK

Measured statistics:

\	DTS Counter		NCP LINE Counter	
	(VAX2)	IBM	VAX2	VAX1
Seconds since zero	180	180	196	210
Messages received	N/A	N/A	15033	24652
Bytes received	N/A	N/A	704535	33869240
Messages send	23598	6166	24822	15278
Bytes send	33037200	8632400	33895746	727305
Messages sent, multiple collision	N/A	N/A	4	0
Messages sent, single collision	N/A	N/A	14	1
Messages sent, initially deferred	N/A	N/A	1043	21

Measured statistics in one second:

VAX2 sent out 131.1 messages with size 1400 bytes to VAX1 per second. VAX1 sent out 76.7 messages with size 46.9 bytes to VAX2 per second. IBM sent out 34.3 messages with size 1400 bytes to uVAX per second.
For network 1: The total number of bytes sent per second was 3463.4 bytes. The total number of bits sent per second was 27706.9 bits. The channel utilization was 0.0028 or 0.28%
For network 2: The total number of bytes sent per second was 231497.7 bytes. The total number of bits sent per second was 1851982.2 bits. The channel utilization was 0.185 or 18.5%

TABLE F.1a Measured statistics of trial 1

Measured statistics:

\ \ \	DTS Counter		NCP LINE Counter	
	(VAX2)	IBM	VAX2	VAX1
Seconds since zero	180	180	192	199
Messages received	N/A	N/A	16992	27153
Bytes received	N/A	N/A	800343	35323343
Messages send	24559	6461	27084	17809
Bytes send	34382600	9045400	35315828	907400
Messages sent, multiple collision	N/A	N/A	10	3
Messages sent, single collision	N/A	N/A	24	4
Messages sent, initially deferred	N/A	N/A	1146	15

Measured statistics in one second:

VAX2 sent out 136.4 messages with size 1400 bytes to VAX1 per second.
VAX1 sent out 88.5 messages with size 47.1 bytes to VAX2 per second.
IBM sent out 35.9 messages with size 1400 bytes to uVAX per second.

For network 1:
The total number of bytes sent per second was 4559.8 bytes.
The total number of bits sent per second was 36478.4 bits.
The channel utilization was 0.036 or 3.6%

For network 2:
The total number of bytes sent per second was 241266.7 bytes.
The total number of bits sent per second was 1930133.3 bits.
The channel utilization was 0.193 or 19.3%

TABLE F.1b Measured statistics of trial 2

Measured statistics:

\	DTS Counter		NCP LINE Counter	
	(VAX2)	IBM	VAX2	VAX1
Seconds since zero	180	180	196	208
Messages received	N/A	N/A	15806	27475
Bytes received	N/A	N/A	738183	36856659
Messages send	25656	6109	26295	17637
Bytes send	35918400	8552600	36801201	888806
Messages sent, multiple collision	N/A	N/A	9	3
Messages sent, single collision	N/A	N/A	17	1
Messages sent, initially deferred	N/A	N/A	1123	23

Measured statistics in one second:

VAX2 sent out 142.5 messages with size 1400 bytes to VAX1 per second.
VAX1 sent out 80.6 messages with size 46.7 bytes to VAX2 per second.
IBM sent out 33.9 messages with size 1400 bytes to uVAX per second.

For network 1:
The total number of bytes sent per second was 4273.1 bytes.
The total number of bits sent per second was 34184.8 bits.
The channel utilization was 0.034 or 3.4%

For network 2:
The total number of bytes sent per second was 247061.1 bytes.
The total number of bits sent per second was 1976488.9 bits.
The channel utilization was 0.198 or 19.8%

TABLE F.1c Measured statistics of trial 3

Measured statistics:

\ \ \	DTS Counter		NCP LINE Counter	
	(VAX2)	IBM	VAX2	VAX1
Seconds since zero	180	180	186	192
Messages received	N/A	N/A	13965	24798
Bytes received	N/A	N/A	653414	32507422
Messages send	22608	6500	23178	16363
Bytes send	31651200	9100000	32442661	833196
Messages sent, multiple collision	N/A	N/A	4	1
Messages sent, single collision	N/A	N/A	15	1
Messages sent, initially deferred	N/A	N/A	1024	25

Measured statistics in one second:

VAX2 sent out 125.6 messages with size 1400 bytes to VAX1 per second. VAX1 sent out 75.1 messages with size 48.9 bytes to VAX2 per second. IBM sent out 36.1 messages with size 1400 bytes to uVAX per second.
For network 1: The total number of bytes sent per second was 4339.6 bytes. The total number of bits sent per second was 34716.5 bits. The channel utilization was 0.035 or 3.5%
For network 2: The total number of bytes sent per second was 226395.5 bytes. The total number of bits sent per second was 1811164.4 bits. The channel utilization was 0.181 or 18.1%

TABLE F.1d Measured statistics of trial 4

Measured statistics:

\	DTS Counter		NCP LINE Counter	
	(VAX2)	IBM	VAX2	VAX1
Seconds since zero	180	180	190	197
Messages received	N/A	N/A	16555	29539
Bytes received	N/A	N/A	773777	39100685
Messages send	27204	6290	27560	19242
Bytes send	38085600	8806000	39001447	939201
Messages sent, multiple collision	N/A	N/A	5	1
Messages sent, single collision	N/A	N/A	12	2
Messages sent, initially deferred	N/A	N/A	1164	18

Measured statistics in one second:

VAX2 sent out 151.1 messages with size 1400 bytes to VAX1 per second. VAX1 sent out 87.1 messages with size 46.7 bytes to VAX2 per second. IBM sent out 34.9 messages with size 1400 bytes to uVAX per second.
For network 1: The total number of bytes sent per second was 4767.5 bytes. The total number of bits sent per second was 38140.1 bits. The channel utilization was 0.038 or 3.8%
For network 2: The total number of bytes sent per second was 260508.9 bytes. The total number of bits sent per second was 2084071.1 bits. The channel utilization was 0.208 or 20.8%

TABLE F.1e Measured statistics of trial 5

Measured statistics:

\	DTS Counter		NCP LINE Counter	
	(VAX2)	IBM	VAX2	VAX1
Seconds since zero	180	180	185	187
Messages received	N/A	N/A	16162	28623
Bytes received	N/A	N/A	753664	38198776
Messages send	26585	6558	26926	18472
Bytes send	37219000	9181200	38113658	890963
Messages sent, multiple collision	N/A	N/A	2	1
Messages sent, single collision	N/A	N/A	22	4
Messages sent, initially deferred	N/A	N/A	1225	25

Measured statistics in one second:

VAX2 sent out 147.7 messages with size 1400 bytes to VAX1 per second. VAX1 sent out 87.4 messages with size 46.6 bytes to VAX2 per second. IBM sent out 36.4 messages with size 1400 bytes to uVAX per second.
For network 1: The total number of bytes sent per second was 4764.5 bytes. The total number of bits sent per second was 38116.1 bits. The channel utilization was 0.038 or 3.8%
For network 2: The total number of bytes sent per second was 257778.9 bytes. The total number of bits sent per second was 2320010.0 bits. The channel utilization was 0.232 or 23.2%

TABLE F.1f Measured statistics of trial 6

APPENDIX G

MEASURED STATISTICS OF THE TERMINAL SERVERS

	VAX 8650 (VAX2)		Terminal Servers (TS)		
	frame size (bytes)	Transmit rate (frames/sec)	frame size (bytes)	Transmit rate (frames/sec)	TS Name
	104.34 80.27 60.40 60.17 64.18	1.83 1.96 6.50 3.08 3.49	64.15 64.16 64.03 64.06 50.07	1.38 1.45 4.20 2.02 2.43	TS1001 TS1002 TS1004 TS1007 TS2001
Average SD	73.87 18.92	3.37 1.89	61.29 6.27	2.30 1.15	

SD=standard deviation

The probability for the terminal servers replying to VAX2 is 0.68.

	VAX 785 (VAX1)		Terminal Servers (TS)		
	frame size (bytes)	Transmit rate (frames/sec)	frame size (bytes)	Transmit rate (frames/sec)	TS Name
	69.59 78.61 73.58 80.08 76.90	3.59 2.86 5.16 3.73 1.18	64.05 64.06 64.02 64.05 64.81	2.67 2.11 3.93 2.90 0.86	TS1011 TS1012 TS1013 TS1014 TS1015
Average SD	75.75 4.21	3.30 1.45	64.20 0.34	2.49 1.13	

SD=standard deviation

The probability for the terminal servers replying to VAX1 is 0.75.

```
NCP>CON NODE TS1001

Console connected (press CTRL/D when finished)


DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1
Enter username>

Local> show counter

                        * ETHERNET COUNTERS *

Seconds Since Zeroed:              27092   Excessive Collisions:      0
Bytes Received:                  5173914   Carrier Check Failed:      0
Bytes Sent:                      2408971   Frames Too Long:           0
Frames Received:                   49588   Heartbeat Absent:          0
Frames Sent:                       37555   Late Collisions:           0
Multicast Bytes Rcv'd:            645215   Data Underrun:             0
Multicast Bytes Sent:               6968   Block Check Error:         0
Multicast Frames Rcv'd:             4359   Framing Error:             0
Multicast Frames Sent:                52   Data Overrun:              0
Frames Sent, Deferred:               155   System Buffer Unavailable: 0
Frames Sent, 1 Collision:              6   User Buffer Unavailable:   0
Frames Sent, 2+ Collisions:            5

                         * SERVER COUNTERS *

Messages Received:                 44832   Duplicates Received:       1
Messages Transmitted:              36936   Illegal Messages Rcv'd:    0
Messages Re-Transmitted:               0   Illegal Slots Rcv'd:       0
Non-Circuit Messages Rcv'd:          564   Duplicate Node Count:      0
Non-Circuit Messages Xmt'd:          454

Local> logoff
Local -020- Logged out port 9
```

```
NCP>CON NODE TS1002

Console connected (press CTRL/D when finished)

DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1

Enter username>

Local> show counter


                          * ETHERNET COUNTERS *

Seconds Since Zeroed:              27090   Excessive Collisions:        0
Bytes Received:                  4272664   Carrier Check Failed:        0
Bytes Sent:                      2512573   Frames Too Long:             0
Frames Received:                   53229   Heartbeat Absent:            0
Frames Sent:                       39162   Late Collisions:             0
Multicast Bytes Rcv'd:            645326   Data Underrun:               0
Multicast Bytes Sent:               6298   Block Check Error:           0
Multicast Frames Rcv'd:             4360   Framing Error:               0
Multicast Frames Sent:                47   Data Overrun:                0
Frames Sent, Deferred:               176   System Buffer Unavailable:   0
Frames Sent, 1 Collision:              9   User Buffer Unavailable:     0
Frames Sent, 2+ Collisions:            6

                           * SERVER COUNTERS *

Messages Received:                 48345   Duplicates Received:         7
Messages Transmitted:              38594   Illegal Messages Rcv'd:      0
Messages Re-Transmitted:               0   Illegal Slots Rcv'd:         0
Non-Circuit Messages Rcv'd:           94   Duplicate Node Count:        0
Non-Circuit Messages Xmt'd:           48

Local> logoff
Local -020- Logged out port 9
```

```
NCP>CON NODE TS1004

Console connected (press CTRL/D when finished)

DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1

Enter username>
Local> show counter

                         * ETHERNET COUNTERS *

Seconds Since Zeroed:             27079   Excessive Collisions:         0
Bytes Received:                10634106   Carrier Check Failed:         0
Bytes Sent:                     7282104   Frames Too Long:              0
Frames Received:                 176071   Heartbeat Absent:             4
Frames Sent:                     113732   Late Collisions:              0
Multicast Bytes Rcv'd:           645392   Data Underrun:                0
Multicast Bytes Sent:              6030   Block Check Error:            0
Multicast Frames Rcv'd:            4361   Framing Error:                0
Multicast Frames Sent:               45   Data Overrun:                 0
Frames Sent, Deferred:              495   System Buffer Unavailable:    0
Frames Sent, 1 Collision:            24   User Buffer Unavailable:      0
Frames Sent, 2+ Collisions:          18

                          * SERVER COUNTERS *

Messages Received:               171622   Duplicates Received:         17
Messages Transmitted:            113601   Illegal Messages Rcv'd:       0
Messages Re-Transmitted:              0   Illegal Slots Rcv'd:          0
Non-Circuit Messages Rcv'd:           0   Duplicate Node Count:         0
Non-Circuit Messages Xmt'd:           0

Local> logoff
Local -020- Logged out port 9
```

```
NCP>CON NODE TS1007

Console connected (press CTRL/D when finished)


DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1
Enter username>

Local> show counter

                          * ETHERNET COUNTERS *

Seconds Since Zeroed:              27072   Excessive Collisions:        0
Bytes Received:                  5022714   Carrier Check Failed:        0
Bytes Sent:                      3506304   Frames Too Long:             0
Frames Received:                   83474   Heartbeat Absent:            4
Frames Sent:                       54738   Late Collisions:             0
Multicast Bytes Rcv'd:            645125   Data Underrun:               0
Multicast Bytes Sent:               5719   Block Check Error:           0
Multicast Frames Rcv'd:             4361   Framing Error:               0
Multicast Frames Sent:                43   Data Overrun:                0
Frames Sent, Deferred:               313   System Buffer Unavailable:   0
Frames Sent, 1 Collision:             14   User Buffer Unavailable:     0
Frames Sent, 2+ Collisions:           15

                          * SERVER COUNTERS *

Messages Received:                 79016   Duplicates Received:         0
Messages Transmitted:              54600   Illegal Messages Rcv'd:      0
Messages Re-Transmitted:               0   Illegal Slots Rcv'd:         0
Non-Circuit Messages Rcv'd:            0   Duplicate Node Count:        0
Non-Circuit Messages Xmt'd:            0

Local> logoff
Local -020- Logged out port 9
```

```
NCP>CON NODE TS2001

Console connected (press CTRL/D when finished)

DECserver 200 Terminal Server V1.0 (BL20) - LAT V5.1

Please type HELP if you need assistance
Enter username>

Local> show counter

DECserver 200 V1.0 BL20        LAT Protocol V5.1       Uptime:  57  04:20:06

Seconds Since Zeroed:          27088    Frames Sent, 1 Collision:       12
Bytes Received:              6065811    Frames Sent, 2+Collisions:      15
Bytes Sent:                  3296756    Send Failures:                   0
Frames Received:               94508    Send Failure Reasons:    000000000
Frames Sent:                   65841    Receive Failures:                0
Multicast Bytes Rcv'd:        645189    Receive Failure Reasons:000000000
Multicast Bytes Sent:           5588    Unrecognized Destination:        0
Multicast Frames Rcv'd:         4362    Data Overrun:                    0
Multicast Frames Sent:            44    User Buffer Unavailable:         0
Frames Sent, Deferred:           282    System Buffer Unavailable:       0
Messages Received:             89919    Duplicates Received:            11
Messages Transmitted:          65572    Messages Re-Transmitted:         0
Solicitations Accepted:           17    Illegal Messages Rcv'd:          0
Solicitations Rejected:            0    Illegal Slots Rcv'd:             0
Multiple Node Addresses:           0    Illegal Multicasts Rcv'd:        0

Local> logoff
Local -020- Logged out port 9
```

```
NCP>CON NODE TS1011

Console connected (press CTRL/D when finished)

DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1
Enter username>

Local> show counter

                              * ETHERNET COUNTERS *

Seconds Since Zeroed:              27065   Excessive Collisions:        0
Bytes Received:                  6754289   Carrier Check Failed:        0
Bytes Sent:                      4630155   Frames Too Long:             0
Frames Received:                   97063   Heartbeat Absent:            0
Frames Sent:                       72286   Late Collisions:             0
Multicast Bytes Rcv'd:            645205   Data Underrun:               0
Multicast Bytes Sent:               7336   Block Check Error:           0
Multicast Frames Rcv'd:             4364   Framing Error:               0
Multicast Frames Sent:                56   Data Overrun:                0
Frames Sent, Deferred:               518   System Buffer Unavailable:   0
Frames Sent, 1 Collision:             62   User Buffer Unavailable:     0
Frames Sent, 2+ Collisions:           63

                              * SERVER COUNTERS *

Messages Received:                 92470   Duplicates Received:         9
Messages Transmitted:              72003   Illegal Messages Rcv'd:      0
Messages Re-Transmitted:               1   Illegal Slots Rcv'd:         0
Non-Circuit Messages Rcv'd:          326   Duplicate Node Count:        0
Non-Circuit Messages Xmt'd:          163

Local> logoff
Local -020- Logged out port 9
```

```
NCP>CON NODE TS1012

Console connected (press CTRL/D when finished)

DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1
Enter username>

Local>

                    * ETHERNET COUNTERS *

Seconds Since Zeroed:             27066   Excessive Collisions:        0
Bytes Received:                 6084757   Carrier Check Failed:        0
Bytes Sent:                     3661894   Frames Too Long:             0
Frames Received:                  77401   Heartbeat Absent:            0
Frames Sent:                      57158   Late Collisions:             0
Multicast Bytes Rcv'd:           645317   Data Underrun:               0
Multicast Bytes Sent:              7205   Block Check Error:           0
Multicast Frames Rcv'd:            4364   Framing Error:               0
Multicast Frames Sent:               55   Data Overrun:                0
Frames Sent, Deferred:              370   System Buffer Unavailable:   0
Frames Sent, 1 Collision:            45   User Buffer Unavailable:     0
Frames Sent, 2+ Collisions:          41

                    * SERVER COUNTERS *

Messages Received:                72972   Duplicates Received:         8
Messages Transmitted:             57040   Illegal Messages Rcv'd:      0
Messages Re-Transmitted:              0   Illegal Slots Rcv'd:         0
Non-Circuit Messages Rcv'd:           0   Duplicate Node Count:        0
Non-Circuit Messages Xmt'd:           0

Local>
Local -020- Logged out port 9
```

```
NCP>CON NODE TS1013

Console connected (press CTRL/D when finished)

DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1

Enter username>
Local> show counter

                          * ETHERNET COUNTERS *
Seconds Since Zeroed:              27062    Excessive Collisions:         0
Bytes Received:                 10265834    Carrier Check Failed:         0
Bytes Sent:                      6804626    Frames Too Long:              0
Frames Received:                  139520    Heartbeat Absent:             0
Frames Sent:                      106282    Late Collisions:              0
Multicast Bytes Rcv'd:            645089    Data Underrun:                0
Multicast Bytes Sent:               4847    Block Check Error:            0
Multicast Frames Rcv'd:             4363    Framing Error:                0
Multicast Frames Sent:                37    Data Overrun:                 0
Frames Sent, Deferred:               755    System Buffer Unavailable:    0
Frames Sent, 1 Collision:             76    User Buffer Unavailable:      0
Frames Sent, 2+ Collisions:           86

                           * SERVER COUNTERS *

Messages Received:                135092    Duplicates Received:          40
Messages Transmitted:             106182    Illegal Messages Rcv'd:       0
Messages Re-Transmitted:               0    Illegal Slots Rcv'd:          0
Non-circuit Messages Rcv'd:            0    Duplicate Node count:         0
Non-circuit Messages Xmt'd:            0

Local> logoff
Local -020- Logged out port 9
```

```
NCP>CON NODE TS1014

Console connected (press CTRL/D when finished)

DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1
Enter username>

Local> show counter

                         * ETHERNET COUNTERS *

Seconds Since Zeroed:            27060   Excessive Collisions:        0
Bytes Received:                8077940   Carrier Check Failed:        0
Bytes Sent:                    5029284   Frames Too Long:             0
Frames Received:                100877   Heartbeat Absent:            0
Frames Sent:                     78520   Late Collisions:             0
Multicast Bytes Rcv'd:          645317   Data Underrun:               0
Multicast Bytes Sent:             5240   Block Check Error:           0
Multicast Frames Rcv'd:           4364   Framing Error:               0
Multicast Frames Sent:              40   Data Overrun:                0
Frames Sent, Deferred:             487   System Buffer Unavailable:   0
Frames Sent, 1 Collision:           54   User Buffer Unavailable:     0
Frames Sent, 2+ Collisions:         49

                          * SERVER COUNTERS *

Messages Received:               95931   Duplicates Received:        12
Messages Transmitted:            77892   Illegal Messages Rcv'd:      0
Messages Re-Transmitted:             0   Illegal Slots Rcv'd:         0
Non-circuit Messages Rcv'd:        272   Duplicate Node count:        0
Non-circuit Messages Xmt'd:        146

Local> logoff
Local -020- Logged out port 9
```

```
NCP>CON NODE TS1015

Console connected (press CTRL/D when finished)


DECserver 100 Terminal Server V1.2 (BL12) - LAT V5.1
Enter username>

Local> sh counter

                        * ETHERNET COUNTERS *

Seconds Since Zeroed:            27065   Excessive Collisions:       0
Bytes Received:                2463068   Carrier Check Failed:       0
Bytes Sent:                    1510317   Frames Too Long:            0
Frames Received:                 32028   Heartbeat Absent:           0
Frames Sent:                     23303   Late Collisions:            0
Multicast Bytes Rcv'd:          645269   Data Underrun:              0
Multicast Bytes Sent:             6157   Block Check Error:          0
Multicast Frames Rcv'd:           4365   Framing Error:              0
Multicast Frames Sent:              47   Data Overrun:               0
Frames Sent, Deferred:             159   System Buffer Unavailable:  0
Frames Sent, 1 Collision:           11   User Buffer Unavailable:    0
Frames Sent, 2+ Collisions:          8

                        * SERVER COUNTERS *

Messages Received:               23917   Duplicates Received:        3
Messages Transmitted:            19523   Illegal Messages Rcv'd:     0
Messages Re-Transmitted:             0   Illegal Slots Rcv'd:        0
Non-Circuit Messages Rcv'd:        548   Duplicate Node Count:       0
Non-Circuit Messages Xmt'd:        277

Local> logoff

Local -020- Logged out port 9

NCP> exit
```