To the Graduate Council:

I am submitting herewith a thesis written by Victoria Ivy Chung entitled "Performance Analysis of the RS232-C Switch on the MAP Network." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Asa O. Bishop, Jr., Major Professor

We have read this thesis
and recommend its acceptance:

Accepted for the Council:

Vice Provost
and Dean of The Graduate School

PERFORMANCE ANALYSIS OF THE RS232-C SWITCH ON THE MAP NETWORK

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Victoria Ivy Chung

May 1990

DEDICATION

This thesis is dedicated to my parents, brothers, and sisters. My parents' strong believe in education was my sustaining source of encouragement in college and higher education. Their endless support and confidence contributed to my being able to continue and complete both the undergraduate and graduate degrees.

ACKNOWLEDGMENTS

I would like to thank the Martin Marietta Corporation and Control Data Corporation for supporting me on the research assistantships. Furthermore, I would like to thank the Electrical and Computer Engineering Department of the University of Tennessee, Knoxville for the teaching assistantship that it offered.

I particularly like to thank Dr. Asa O. Bishop, Jr., Professor of the Electrical and Computer Engineering and major advisor of my graduate committee, for helping me to obtain the research assistantships from Martin Marietta Corporation and Control Data Corporation. He has given me precious suggestions and advice in my research. I also like to thank Dr. Robert E. Bodenheimer and Dr. Frederick W. Symonds, Professors of the Electrical and Computer Engineering Department, for their assistance and for serving on my graduate committee.

Finally, I would like to express my gratitude to Mr. Jason Hartman, who was the manager of the Manufacturing Automation Protocol (MAP) network of the Electrical and Computer Engineering Department, for his suggestions, assistance, and reviews. I greatly appreciate Mr. Rick Igou, an electrical engineer of the Martin Marietta Corporation, who provided me the valuable references and opinions. Last but not least, I am deeply grateful to Simon, who is my confidant, husband, and colleague, for offering me the assistance, advice, and encouragement on the research.

ABSTRACT

The issue of performance analysis plays a very important role in the computer communication arena today. In order to make a meaningful performance analysis on a network, specific conditions such as node and network loading, message length, and baud rate have to be established prior to making the measurements. This thesis was based on two nodes established on a well defined, experimental MAP Network of the Electrical and Computer Department at the University of Tennessee, Knoxville. The objective of this thesis is to define and implement a set of performance tests using test setups with different message lengths, baud rates, and loading conditions.

Due to the large number of RS232-C devices in the manufacturing environment, RS232-C devices communicating through a standard communication protocol were the focus of this thesis. The RS232-C devices were connected to a MAP broadband backbone. Baud rates of 1200, 2400, 4800, 9600, and 19200 bps were used. Network interface units such as the two-layer MAP multiplexers and the seven-layer MAP multiplexers were used for testing. The methodology of making the measurements under the conditions described above was presented. The results of the measurements were analyzed and documented with respect to network performance.

TABLE OF CONTENTS

CHAPTER	PAGE
1. INTRODUCTION	**1**
1.1. Communication as a Tool	1
1.2. The Birth of Communication Protocol	1
1.3. Issue of Performance	4
2. RS232-C SERIAL COMMUNICATION	**6**
2.1. Need for Serial Communication	6
2.2. RS232-C Terminology	7
2.3. Asynchronous Communication Format	12
2.4. Hardware in Asynchronous Communication	13
2.5. Polling Method	17
2.6. Interrupt Method	18
3. ORGANIZATION OF THE TEST	**19**
3.1. Test Architecture	19
3.2. Test Methodology	26
3.3. Test Hardware and Software	35
4. RS232-C IMPLEMENTATION AT 19.2 KBAUD	**37**

CHAPTER PAGE

5. RESULTS AND CONCLUSION..... 41

5.1. Test Results Taken Under 1200 to 9600 bps..... 41

5.2. Test Results Taken Under 19.2 kbps..... 49

5.3. Frame Format from the Network Analyzer..... 50

5.4. Conclusion..... 60

5.5. Future Work..... 61

LIST OF REFERENCES..... 63

VITA..... 66

LIST OF FIGURES

FIGURE **PAGE**

1-1. ISO/OSI Seven Layers Reference Model..... 3

2-1. Three Modes of Operation in Communication..... 8

2-2. DTE and DCE Configuration..... 10

2-3. Asynchronous Data Format..... 14

3-1. Model of the Overall Test Architecture..... 20

3-2. Two PCs Connected to the Same MAP Multiplexer... 21

3-3. Each PC Connected to a Separate MAP Multiplexer.. 23

3-4. Cable Configuration For Null Modem Mode..... 24

3-5. Cable Configuration For MAP Multiplexer..... 25

3-6. Setup 1 For Null Modem Mode..... 28

3-7. Setup 2 Which Included a Two-Layer MAP Multiplexer..... 29

3-8. Setup 3 Which Included 2 Two-Layer MAP Multiplexers..... 31

3-9. Setup 4 Which Included a Seven-Layer MAP Multiplexer..... 32

3-10. Setup 5 Which Included Two Seven-Layer MAP Multiplexers..... 34

5-1. Results For Two-layer MUX Under Different Message Lengths..... 45

FIGURE PAGE

5-2. Results For Seven-layer MUX Under Different Message Lengths 45
5-3. Results Under No Load Condition Using 256 char/msg 47
5-4. Results Under No Load Condition Using 1024 char/msg 47
5-5. Results Under Loading Condition Using 256 char/msg 48
5-6. Results Under Loading Condition Using 1024 char/msg 48
5-7. Results Under No Load Condition Using IBM PC-XT and TI PC at 256 char/msg 51
5-8. Results Under No Load Condition Using Everex at 256 char/msg 51
5-9. Results Under No Load Condition Using Everex or IBM/TI at 256 char/msg 52
5-10. Frame Format at the Data Link Layer 53
5-11. 802.4 Display Format of a Frame 55
5-12. The Data and Data Acknowledgment TPDUs 57
5-13. The Frames Captured by a Network Traffic Analyzer 58

CHAPTER 1

INTRODUCTION

1.1. Communication as a Tool

Communication is a tool for exchanging information among people. With the system of communication being unified between the plant floors and the controller rooms in the manufacturing environment, products may be systematically and effectively designed, manufactured, monitored, and tested. In order to have efficient ways of manufacturing high quality but low cost products to facilitate competition with foreign countries in the national and international markets, communication is a major issue in the manufacturing environments today. However, the existence of multivendor, multimedia, and multioperating system computer technologies makes inter-vendor communication difficult.

1.2. The Birth of Communication Protocol

Problems usually arise when one product is not compatible with another. To overcome these incompatibilities so that the cost associated with proprietary products can be eliminated, the

creation of the communication protocols such as MAP (Manufacturing Automation Protocol) and TOP (Technical and Office Protocol) were originated from General Motors and Boeing, respectively. These protocols are based on the International Standard Organization/Open Systems Interconnection (ISO/OSI) reference model. The OSI reference model provides a set of protocols for information exchange among application/users on a network. The effort to form the model was started in 1978 and the model was accepted by ISO officially in 1983 [1].

The model consists of seven layers of communication protocols. These layers are physical, data link, network, transport, session, presentation, and application. Each layer is obligated to perform certain functions so that the whole network can function as a coherent system. A tree of standards with specifications that are recognized internationally constitutes the model. The MAP specification is a subset of the OSI reference model [2]. See Figure 1-1 for the seven layers of the OSI reference model.

The ultimate objective of MAP is to form an applicable, universal, and flexible communication system in order to provide different vendors' products with compatible services. Furthermore, the communication system must have the flexibility for any improvements in technology and expansions when additional requirements of communications are realized and implemented.

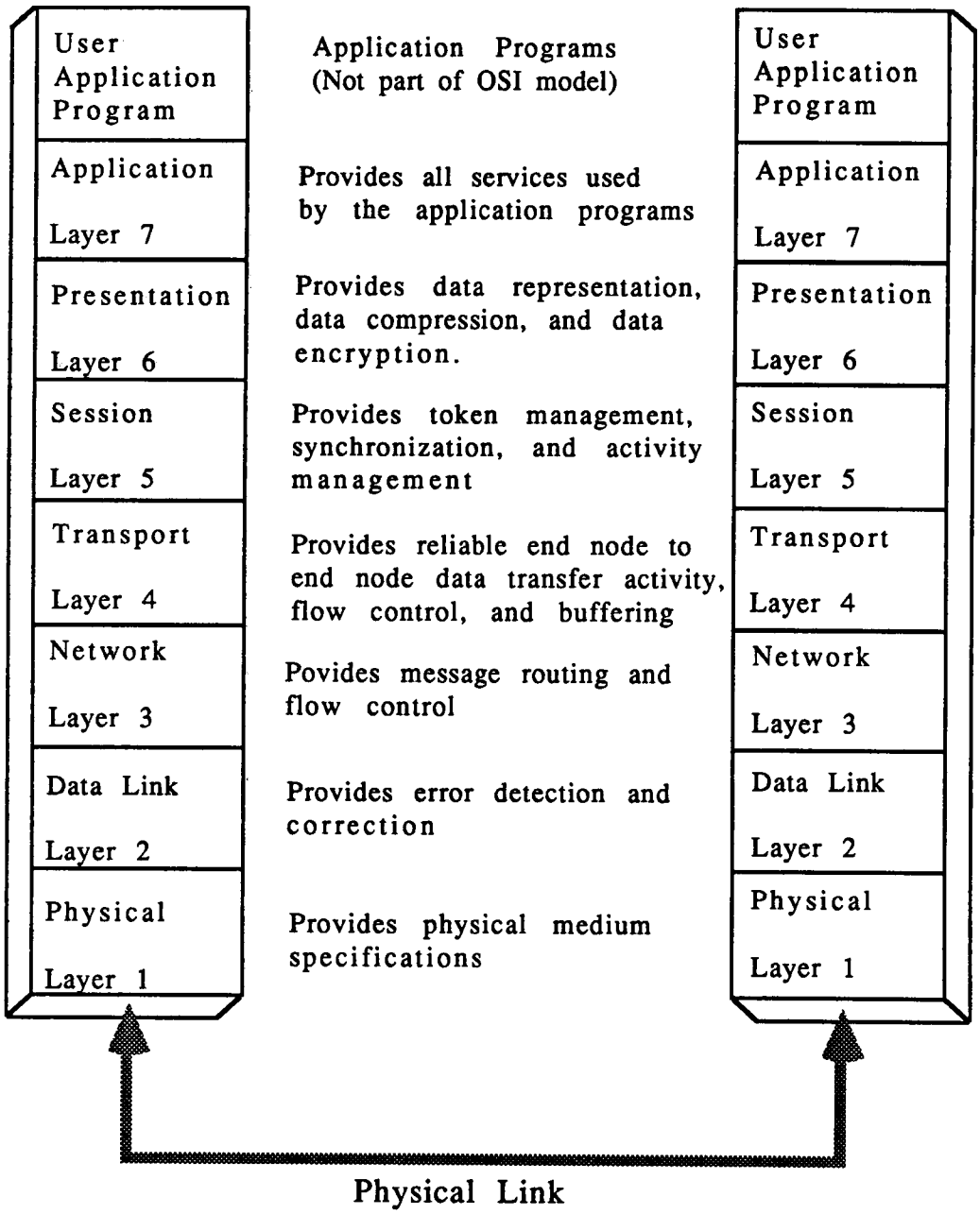


Figure 1-1. ISO/OSI Seven Layers Reference Model[1]

[1] MAP Specification Versions 2.1.A. & 2.2.(Michigan: General Motors Technical Center, 1986), p. 3-27.

1.3. Issue Of Performance

The ability of different vendors' products to communicate with one another only implies that they can interoperate. It does not imply any level of performance. The information on different levels of performance is a critical factor in vendor interoperation. For example, the network may interoperate with a frequent amount of errors occurring, or two devices may be able to interoperate, but they take a large amount of time to transfer a message under some traffic or load condition. Thus, performance analysis is very important in the study of the network [3].

There are two primary factors, transmission time and throughput, that define the performance of the network [4]. Transmission time is defined as the time required for a piece of information to travel from one application device to another. Data throughput is defined as the average number of data bits passed from one device to another in the network per unit of time (bits/second). In the process of transferring the message, delay in the form of overhead occurs. As a result, transmission time increases and throughput decreases.

Delays include propagation delays of the gates on the network adapter, processing delays of the application program (host's speed), packet creation using the protocol suite, and queueing delays of the packets at intermediate points between the source and destination [5]. Generally, the nature of the traffic

created by the user devices at the nodes in the network drastically affects the performance [6]. Thus, node and network loading is a major parameter that must be set as a condition for measurement.

The objective of this thesis is to define and implement a set of performance tests using test setups with different message lengths, baud rates, and loading conditions. The transfer of data between two RS232-C devices connected by null modems provides the basis for comparison of loading in the MAP backbone. The RS232-C devices are interfaced to the MAP network using a two-layer interface and a seven-layer interface. The overhead associated with the network interface, both two and seven-layer implementations of the ISO/OSI reference model, are compared with a native RS232-C network.

CHAPTER 2

RS232-C SERIAL COMMUNICATION

2.1. Need For Serial Communication

Normally, data travels in parallel from one device to another. Thus, the data bus used to convey bits of information has to be at least eight-bit wide (because a character is at least a byte, which is eight bits wide). This method of information transfer is widely used for parallel printers and other custom I/O features.

However, serial communication also has a place in the data exchange arena. During serial communication, bits of data travel serially from one device to another. For a single-byte character, each bit of the eight-bit data has to go through the same transmission line. Hence, serial communication has a much slower rate in transmitting a piece of information between devices. This kind of communication method is widely applied in telephone lines, in exchanging data between computers, and in some cases in sending data to the serial interfaced printers.

2.2. *RS232-C Terminology*

RS232-C is a recommended standard which describes how serial communication should behave. The data rate used to transmit the data can be range from 75 baud (bits per second) up to 19.2 kbaud (19200 bits per second). A maximum of 50 feet of cable is specified as a data path for a serial RS232-C connection [7].

There are three kinds of operations: simplex, half-duplex, and full-duplex. Simplex means that data can flow in only one direction between the source and destination. Thus, one station can only transmit, while the other station can only receive. In the half-duplex mode, data can flow from the source to destination and vice versa in a single route but not at the same time. This means that data travels in direction at one time, and the opposite direction at another time. Full-duplex means that data can flow in both direction between source and destination at the same time. Therefore, there is more than one mode to allow multiple directions in RS232-C environment [7]. Figure 2-1 illustrates the simplex, half-duplex, and full-duplex operations.

RS232-C consists of data signals, control signals, ground signals, pins, connectors, and many other components. It is an interface between data terminal equipment (DTE) and data communication equipment (DCE). A DTE is normally a terminal or a

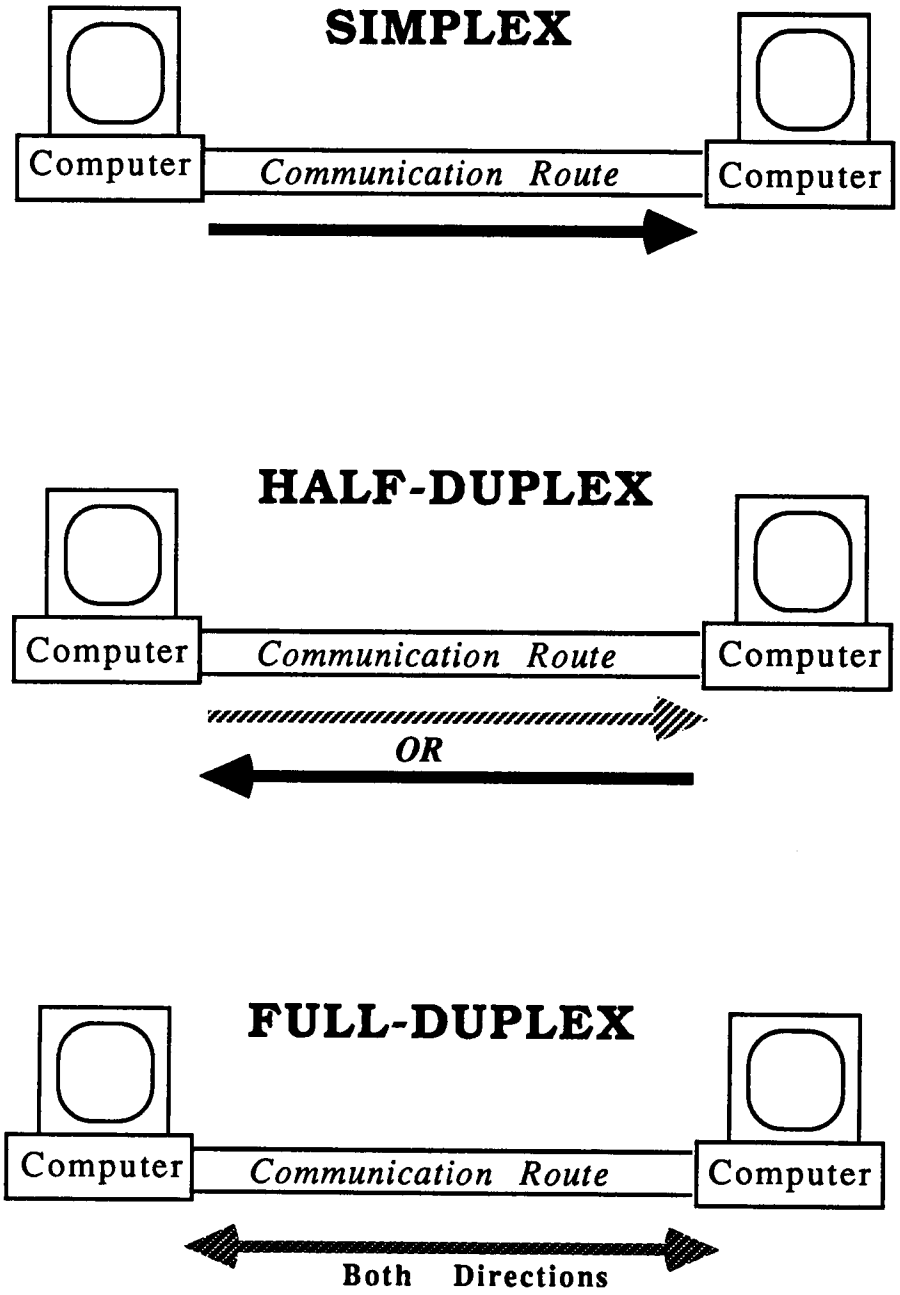


Figure 2-1. Three Modes of Operation in Communication

computer, while a DCE is normally a modem. Data signals such as transmitting (TD) and receiving (RD) lines are used for transmitting and receiving data respectively. Control signal such as data terminal ready (DTR) is used by DTE to inform DCE that it should be ready to answer the call. Usually, when the power is on, DTR is true, and a connection is ready to be made. If DTR is not true, then DCE will not prepare to answer the call. Data set ready (DSR) is then raised by the DCE when a connection has been made.

After a connection has been made, DTE raises request to send (RTS) to indicate its intention to send data. DCE then checks the other end to see if there is any asserted signal of data carrier detect (DCD). If no DCD is seen by DCE, then DCE raises clear to send signal (CTS) to indicate that DTE can start to send data to the other end. Otherwise, DCE will not assert any CTS signal. DTE starts to send data through its TD line and the data will be received by the RD line at other end of the connection. As long as there is data to be sent by the DTE, the RTS will remain high. When all data have been sent by the DTE, the RTS line as well as the CTS line will go low. Then, the DTE at either end can initiate the next RTS subsequently to control the line. The ground signals are the protective ground and signal ground which provide the grounding effect for the RS232-C cable [7]. Figure 2-2 shows the setup of the DTE and DCE. Table 2-1 shows the list of functions of the signals in an RS232-C environment.

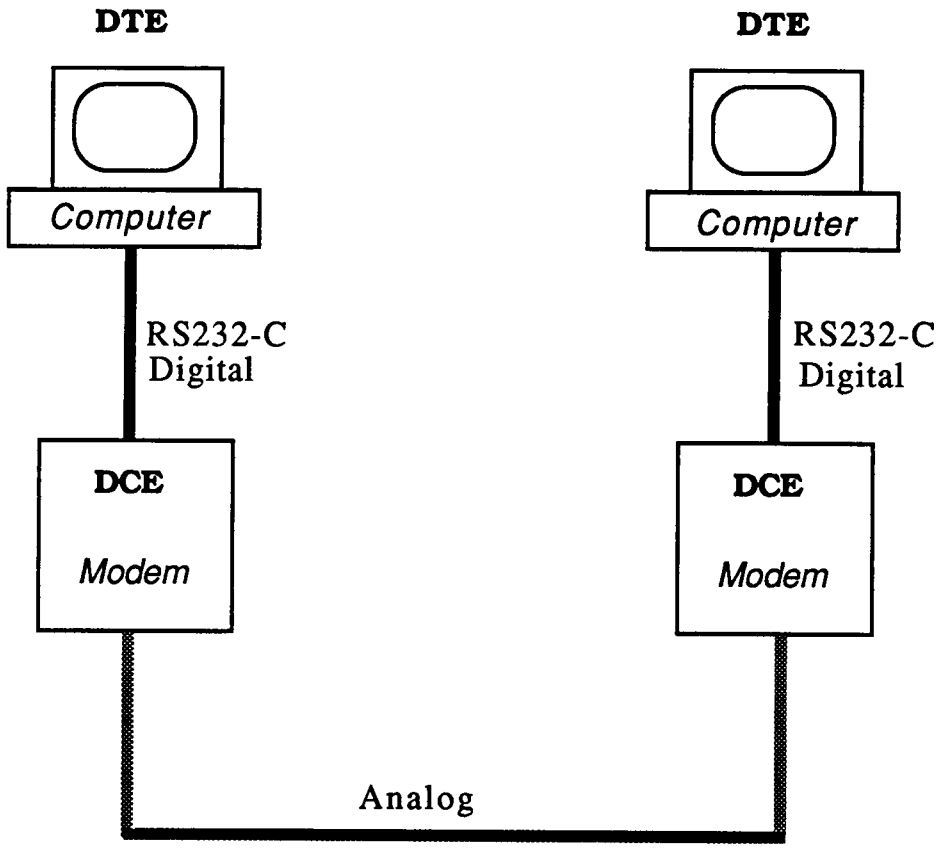


Figure 2-2. DTE and DCE Configuration

Table 2-1. Functions of the Signals of a Serial Port[1]

Name	Function	Pin Number
Protective Ground (PG)	Ground	1
Transmitted Data (TD)	Data	2
Received Data (RD)	Data	3
Request To Send (RTS)	Control	4
Clear To Send (CTS)	Control	5
Data Set Ready (DSR)	Control	6
Signal Ground (SG)	Ground	7
Data Carrier Detect (DCD)	Control	8
Data Terminal Ready (DTR)	Control	20

[1]Martin D. Seyer, RS-232 Made Easy (Englewood Cliffs, N.J. : Prentice-Hall, 1984), p.28.

2.3. *Asynchronous Communication Format*

Serial communication is asynchronous, which means that the length of time between characters is not uniform [8]. Therefore, asynchronous communication needs some kind of symbol or signal to inform the other end of the communication line that a correct format of data is coming and that it should ready itself to recognize and receive data from the sender. As a result, special bits or delimiters are used to denote the beginning and ending of a character in asynchronous communication. A stream of bits in ones (high level voltages) or zeroes (low level voltages) is transmitted and received serially from one computer to another. The high level voltage is called mark and the low voltage is called space. Marks are usually used to separate characters. When there is no data in the transmission line, the line is held in the mark position.

A character can be represented in five, six, seven, or eight bits. The character is begun by a start bit, which is a space, to denote the starting of a character. Then, it is followed by the bits representing the character. After that, a parity bit can be optionally appended to provide a checking scheme indicating whether the data is transmitted and received correctly. The parity bit can be odd, even, or none. When the parity bit is set to be even, the receiving end has to check to see if the number of ones in the

character including the parity bit is even. When the parity is set to be odd, then the receiver should check to see if the number of ones in the character including the parity bit received is odd to ensure that the data received is accurate. Otherwise, some bits among the character have been distorted during the transfer of the data stream. Note that when the number of bits of the character is specified to be eight, then there will not be a parity bit appended to the character. Finally, the character is followed by one or more stop bits, which are high level voltages. Figure 2-3 shows the format of the data stream in asynchronous communication. Note that both ends of the transmission line must use the same data format in order to have successful transmission.

2.4. Hardware In Asynchronous Communication

In the International Business Machines (IBM) or compatible personal computers, the Universal Asynchronous Receiver Transmitter (UART) is used as the asynchronous communication control. UART is a chip with special purposes to manage the asynchronous communication. There is a shift register inside the UART which shifts each data bit from the transmitter to the transmission line after the data is converted to serial format, or shifts each data bit from the line serially to the receiver in order to be received and be converted into parallel format for the other

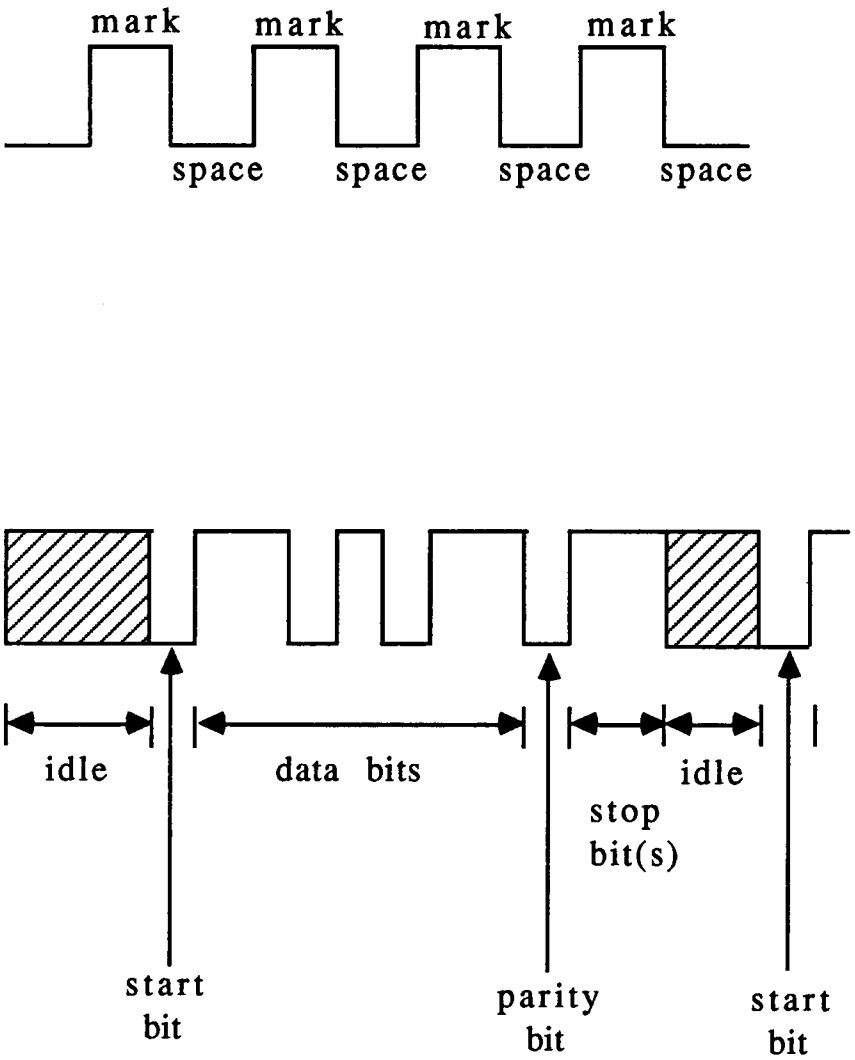


Figure 2-3. Asynchronous Data Format

parts of the computer to manipulate them. An UART supports one RS232-C port. Usually, UARTs reside on a board which contains up to two RS232-C ports. The ports are called COM1 and COM2.

There are ten registers which are single-byte and programmable for RS232-C ports status observations and data exchange operations. These registers are the Transmit Holding Register, the Receive Data Register, the Baud Rate Divisor Register for Least Significant Byte, the Baud Rate Divisor Register for Most Significant Byte, the Interrupt Enable Register, the Interrupt Identification Register, the Line Control Register, the Modem Control Register, the Line Status Register, and the Modem Status Register. In the Intel families of 8088 to 80286 microprocessors, the I/O address of COM1 starts at 3F8H, while COM2 starts at 2F8H. Then, the ten registers described above have addresses in the range of 3F8H to 3FEH for COM1 and 2F8H to 2FEH for COM2 [9]. The addresses for the ten registers are listed in Table 2-2.

The transmit holding register is used to hold data in the register before it is sent to the receiver. The receive data register is used to retain data after it is received by the receiver. In the register for baud rate divisor, the most and least significant bytes are used to store the divisor which can obtain the correct baud rate. The interrupt enable register is used to enable and disable interrupts generated by the serial adapter UART. The interrupt identification register is used to indicate the particular interrupt which has been invoked by the system. The line control register

Table 2-2. Registers for Accessing the Serial Port[2]

RS232-C Port Address		Register	ETC Decoding
Com1	Com2		At 3FBH
3F8H	2F8H	Transmit Buffer	Bit 7 = 0 (write)
3F8H	2F8H	Receive Buffer	Bit 7 = 0 (read)
3F8H	2F8H	Baud Divisor LSB	Bit 7 = 1
3F9H	2F9H	Baud Divisor MSB	Bit 7 = 1
3F9H	2F9H	Interrupt Enable	
3FAH	2FAH	Interrupt Identification	
3FBH	2FBH	Line Control	
3FCH	2FCH	Modem Control	
3FDH	2FDH	Line Status	
3FEH	2FEH	Modem Status	

[2] Robert Jourdain, Programmer's Problem Solver for the IBM PC, XT, & AT (New York: Brady, 1986), p. 369.

serves as a place to store the configuration information for communication and its content can also be read for assurance. The modem control register controls the RS232-C signals such as DTR, DSR, RTS, and CTS. The line status register is used to monitor the status of the line, while the modem status register is used to observe the status of the modem.

2.5. *Polling Method*

Polling is a method of RS232-C communication operation. The I/O status register is constantly checked to see if there is a character ready to be received before a character can be read from the receive data register. In this mode of transmitting a character, the I/O status register is also constantly tested to see if the transmit holding register is ready to send a character to the line. The drawback of this method is that instead of using the processing time to transmit or receive data from the RS232-C port at the speed of a microprocessor, delay results due to the constant checking of the I/O status register. Hence, at the speed of 300 baud, the microprocessor is able to check the I/O status register until it indicates that a character has arrived, and the CPU is able to put the character into the receive data register. Since the character may be printed on the screen or written to the disk before it goes back to check the I/O status register to see if there is any other

character to be captured, the character may be lost. At the rate of 9600 baud, the microprocessor will not be able to capture every character at RS232-C port on time.

2.6. Interrupt Method

Another method is used where interrupts can be established to initiate communication at the appropriate time without losing any data at higher baud rates. Interrupts can be programmed to be initiated. When the I/O device is available, it informs the CPU that an interrupt exists and is ready to be serviced. The CPU quickly halts its present task, saves its present status, and provides services to the interrupt. The data is then immediately received and saved in a temporary storage buffer before the next data is captured. In this way, no data is lost at any baud rate because each character is stored in the temporary buffer after being received from the RS232-C port. As a result, no time is wasted. When the interrupt routine is completed, the previous state is retrieved and the previous task is continued.

CHAPTER 3

ORGANIZATION OF THE TEST

3.1. Test Architecture

Two nodes have been established on the experimental Manufacturing Automation Protocol (MAP) Network in the Department of Electrical and Computer Engineering at the University of Tennessee, Knoxville. A MAP multiplexer was attached to each node. Figure 3-1 illustrates the model of the test architecture. Measurements were done using two RS232-C devices. Node and network loading was added to the network by other devices. The information transfer activity was done using five different setups to compare the results. The setups are listed as follows:

(1) Transfer data between two RS232-C devices using null modem.

(2) Transfer data between two RS232-C devices with both devices connected to the same two-layer MAP multiplexer on the MAP network (Figure 3-2).

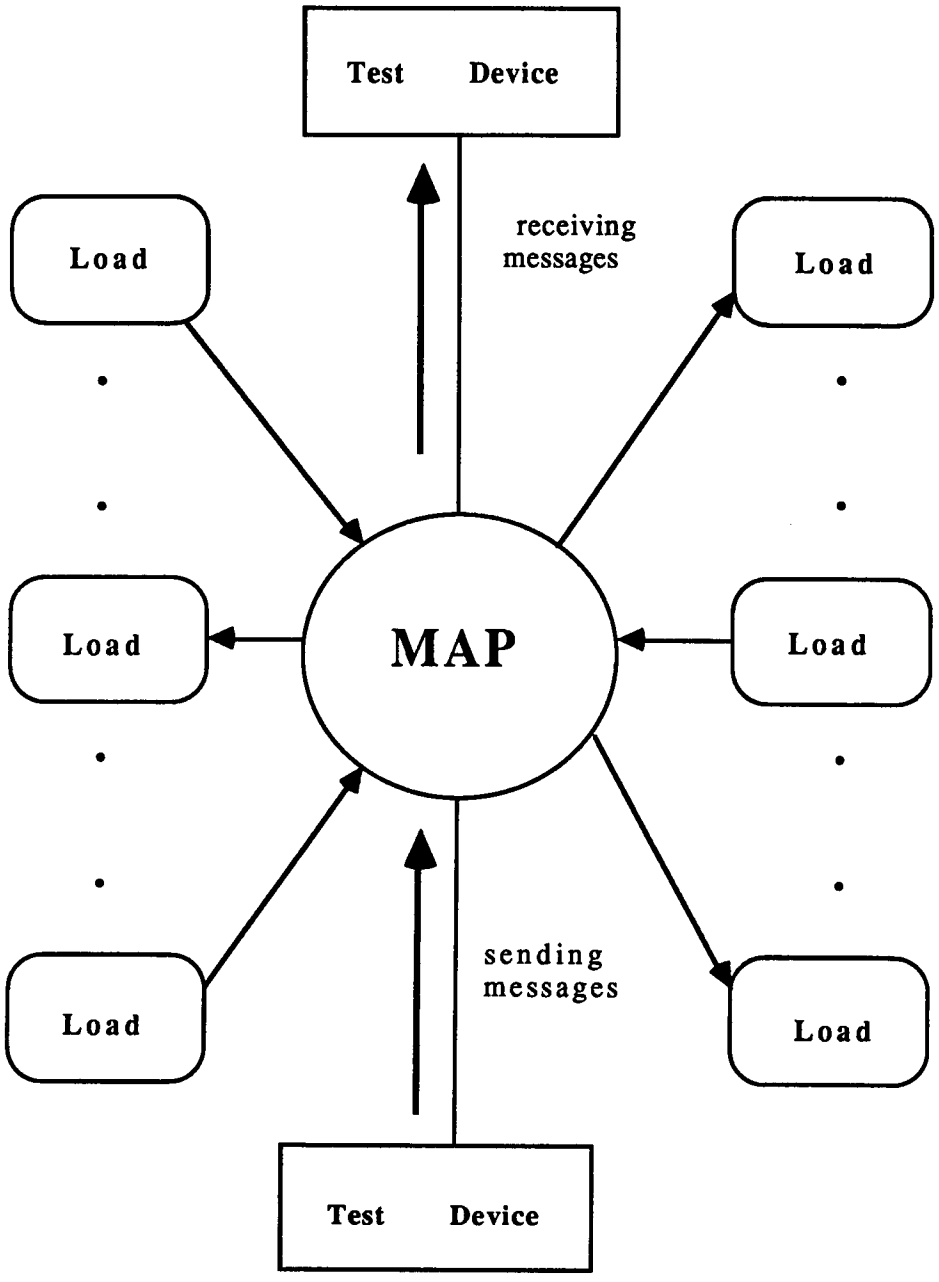


Figure 3-1. Model of the Overall Test Architecture

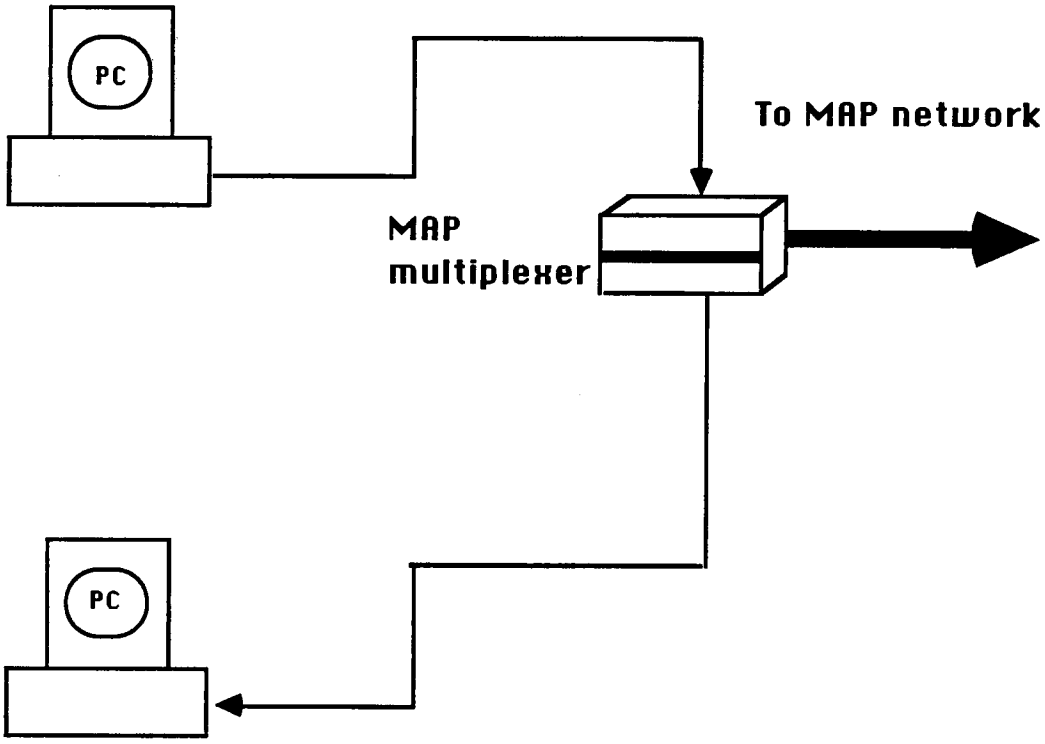


Figure 3-2. Two PCs Connected to the Same MAP Multiplexer

(3) Transfer data between two RS232-C devices with each device connected to a separate two-layer MAP multiplexer on the MAP network (Figure 3-3).

(4) Transfer data between two RS232-C devices with both devices connected to the same seven-layer MAP multiplexer on the MAP network (Figure 3-2).

(5) Transfer data between two RS232-C devices with each device connected to a separate seven-layer MAP multiplexer on the MAP network (Figure 3-3).

Two RS232-C devices were designated to be the application devices. One of the devices was designated to be the sending device and the other to be the receiving device. Also, each device was connected to a designated port depending on each setup. The RS232-C devices were configured in null modem mode when a null modem was used. For all other cases when either two or seven-layer MAP multiplexers were used, the RS232-C devices were configured with the MAP multiplexer. See Figure 3-4 for the configuration of the RS232-C devices using null modem mode and Figure 3-5 for the configuration of RS232-C devices to be used with a MAP multiplexer. Furthermore, a set of selected lengths of messages (ranging from 128 to 1024 characters per message) and

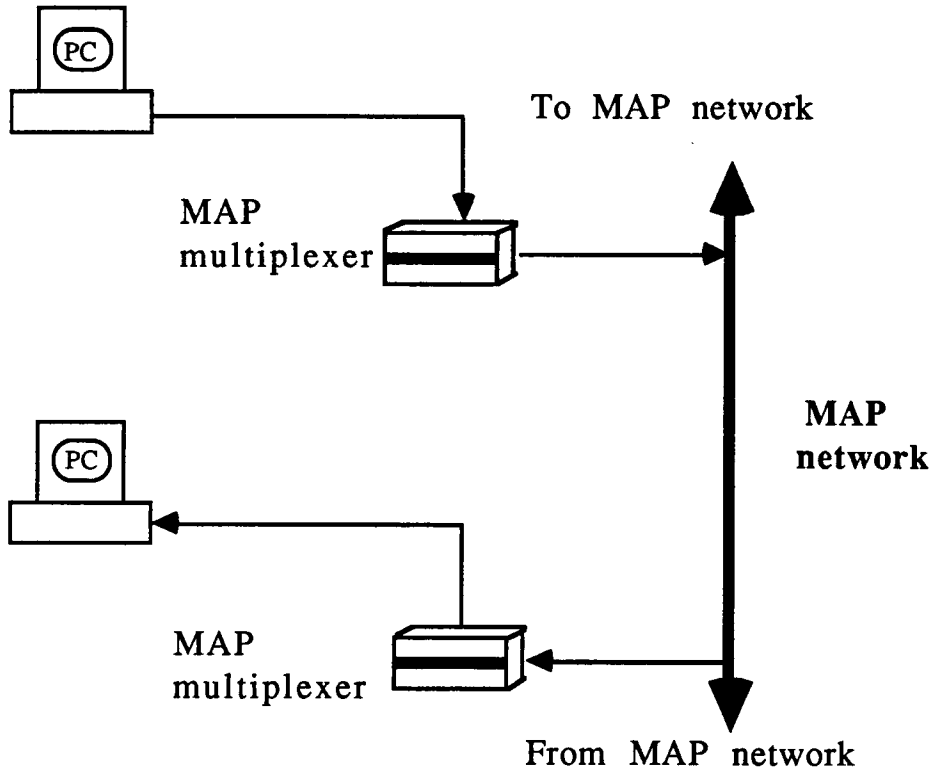


Figure 3-3. Each PC Connected to a Separate MAP Multiplexer

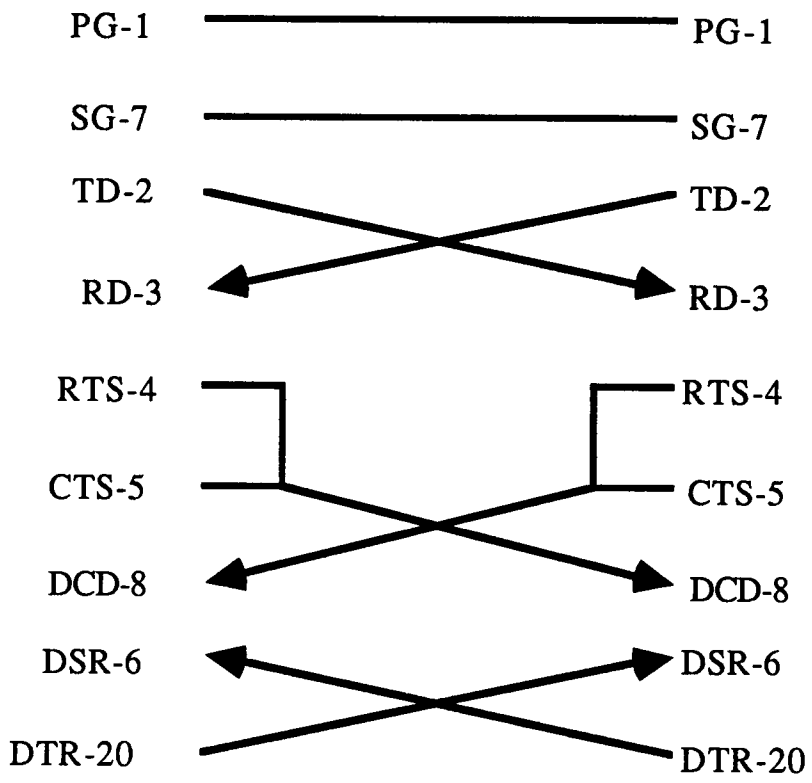


Figure 3-4. Cable Configuration For Null Modem Mode[1]

[1]Martin D. Seyer, RS-232 Made Easy (Englewood Cliffs, N.J.: Prentice-Hall, 1984), p. 64.

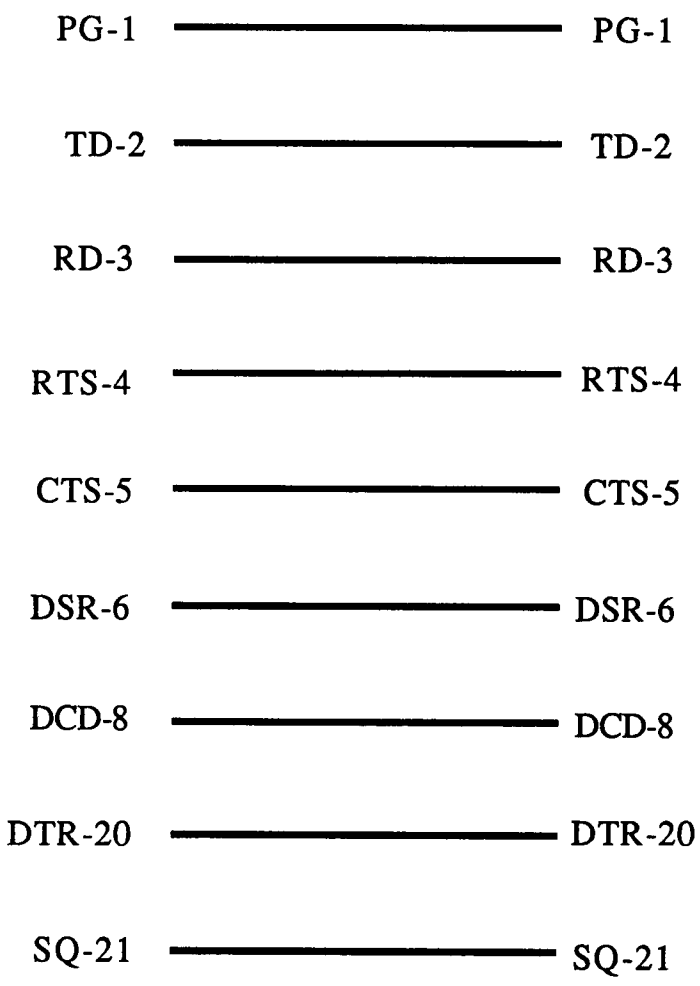


Figure 3-5. Cable Configuration For MAP Multiplexer[2]

[2] Martin D. Seyer, RS-232 Made Easy (Englewood Cliffs, N.J.: Prentice-Hall, 1984), p.65.

baud rates (ranging from 1200 to 9600 baud) were designated to be used for each setup.

3.2. Test Methodology

Two RS232-C devices communicating over a network consisting of no other devices and using a null modem was the reference for comparison of each setup. Setup 1 established a base load for a node on a RS232-C network. For the purpose of this set of tests, the base load was considered a no load condition. This base load existed in each of the setups 1 through 5. Loading in setups 2 through 5 was compared with the loading of setup 1 which established the no load condition of this series of tests.

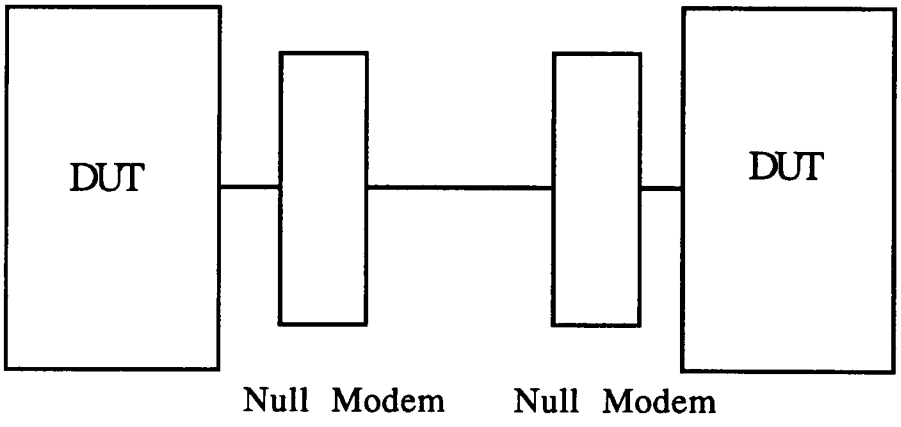
- **Setup 1**

When the null modem mode was used, selected lengths of messages were sent from one device to the other device with each chosen baud rate. Measurement of transmission time started when a message transmission began on the sending device and ended when the message was received by the receiving device. Several baud rates: 1200, 2400, 4800, and 9600 bps were selected to be used with each length of message for the transfer activity. The

transmission time was a direct function of the RS232-C network interface. Any loading due to the node itself was considered in the base load. See Figure 3-6 for this setup.

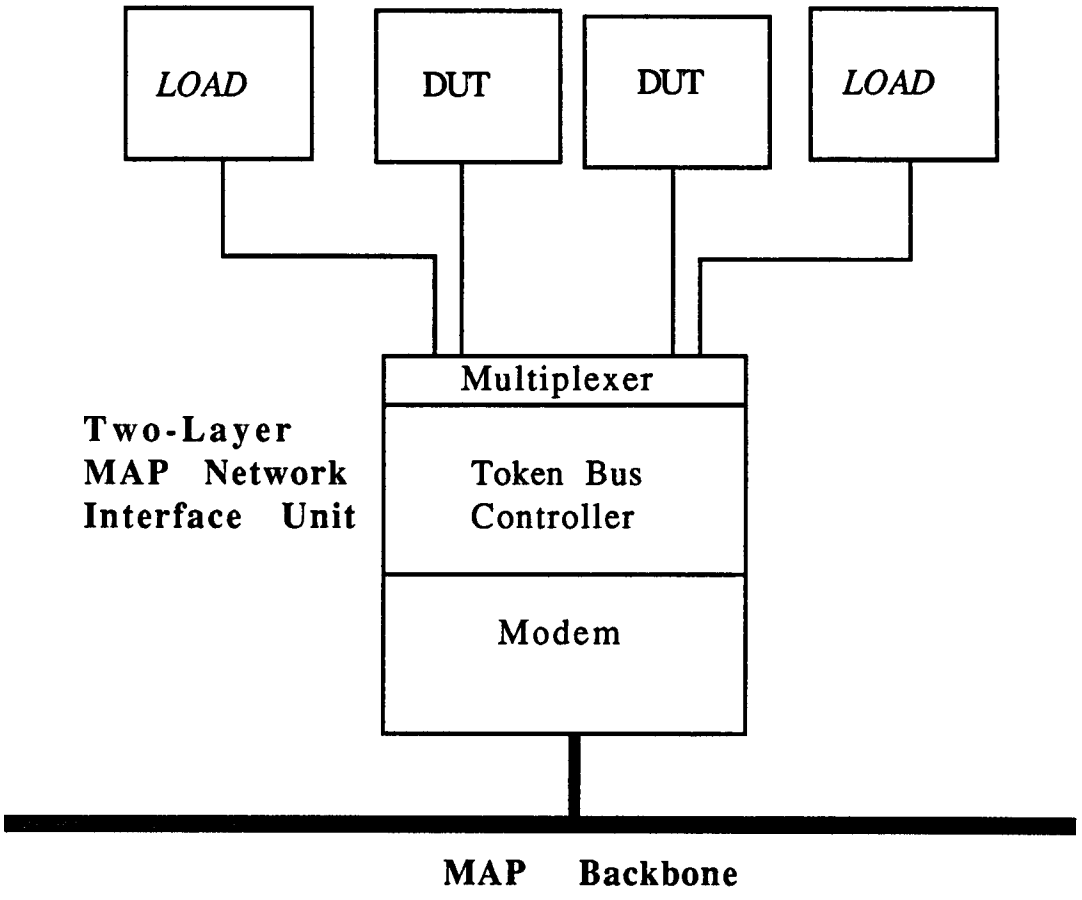
- **Setup 2**

When both RS232-C devices were connected to the same two-layer MAP multiplexer, each device was attached to a different port of the same MAP multiplexer. This setup was used under no load and load conditions. The same set of message size, baud rates, and timing procedure for the transmission time from the null modem mode were used for no load and load conditions. Under load condition, the load was created by sending 1024 characters per message at 9600 baud from a port of the two-layer MAP multiplexer to the other port of the same multiplexer back and forth repetitiously. Since there were eight ports on a MAP multiplexer, two of the ports were connected to the application devices for measurement, six ports were left to be used to create three traffic paths where each path was established by the transfer activity between two devices at two ports. The switching time for the multiplexer also contributed to the delay through the network. See Figure 3-7 for this setup.



• DUT -- Device Under Test

Figure 3-6. Setup 1 For Null Modem Mode



• DUT -- Device Under Test

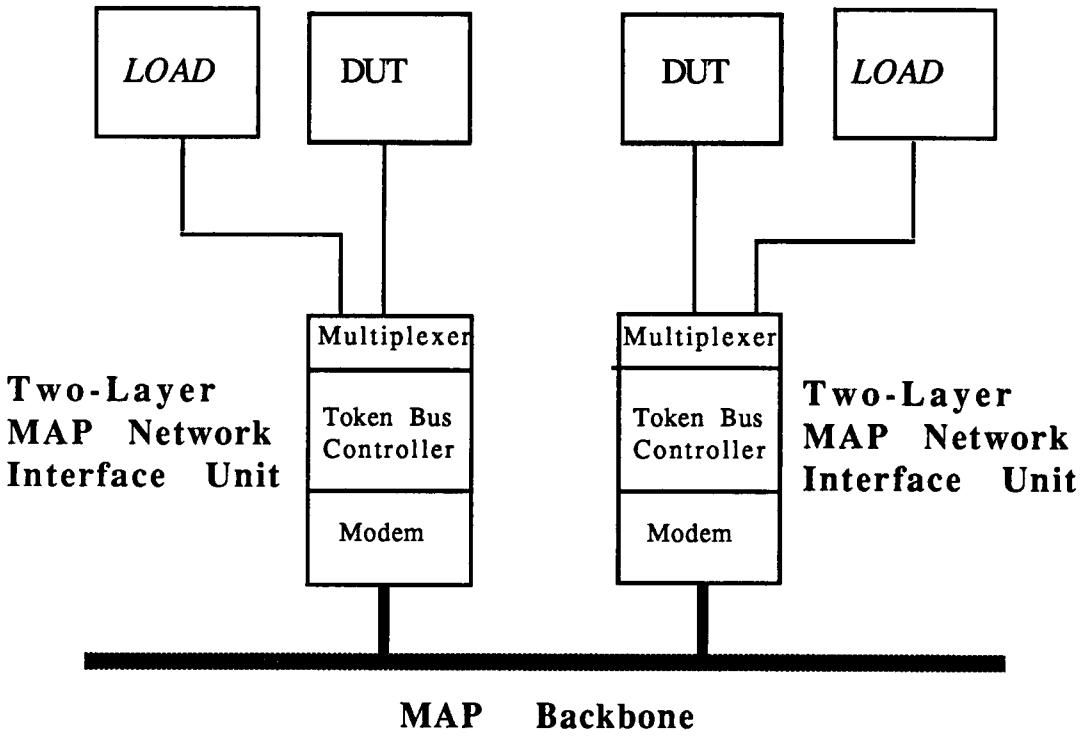
Figure 3-7. Setup 2 Which Included a Two-Layer MAP Multiplexer

- **Setup 3**

In the case of the setup with each RS232-C device connected to a different two-layer MAP multiplexer, each device was attached to a port of a different two-layer MAP multiplexer. No load and load conditions were used in this setup. In the no load condition, the same procedure and parameters from the null modem mode were used. However, the load condition was different from the one described in Setup 2. The load was created in the network by four traffic paths. Each path sent 1024 characters per message at 9600 baud back and forth from one device attached to one two-layer MAP multiplexer to another device attached to the other two-layer MAP multiplexer. The delay through the network included the switching time for the two multiplexers. See Figure 3-8 for this setup. Any addition of the load could be quantified by the difference of timing with reference to the results taken under the specified loading conditions.

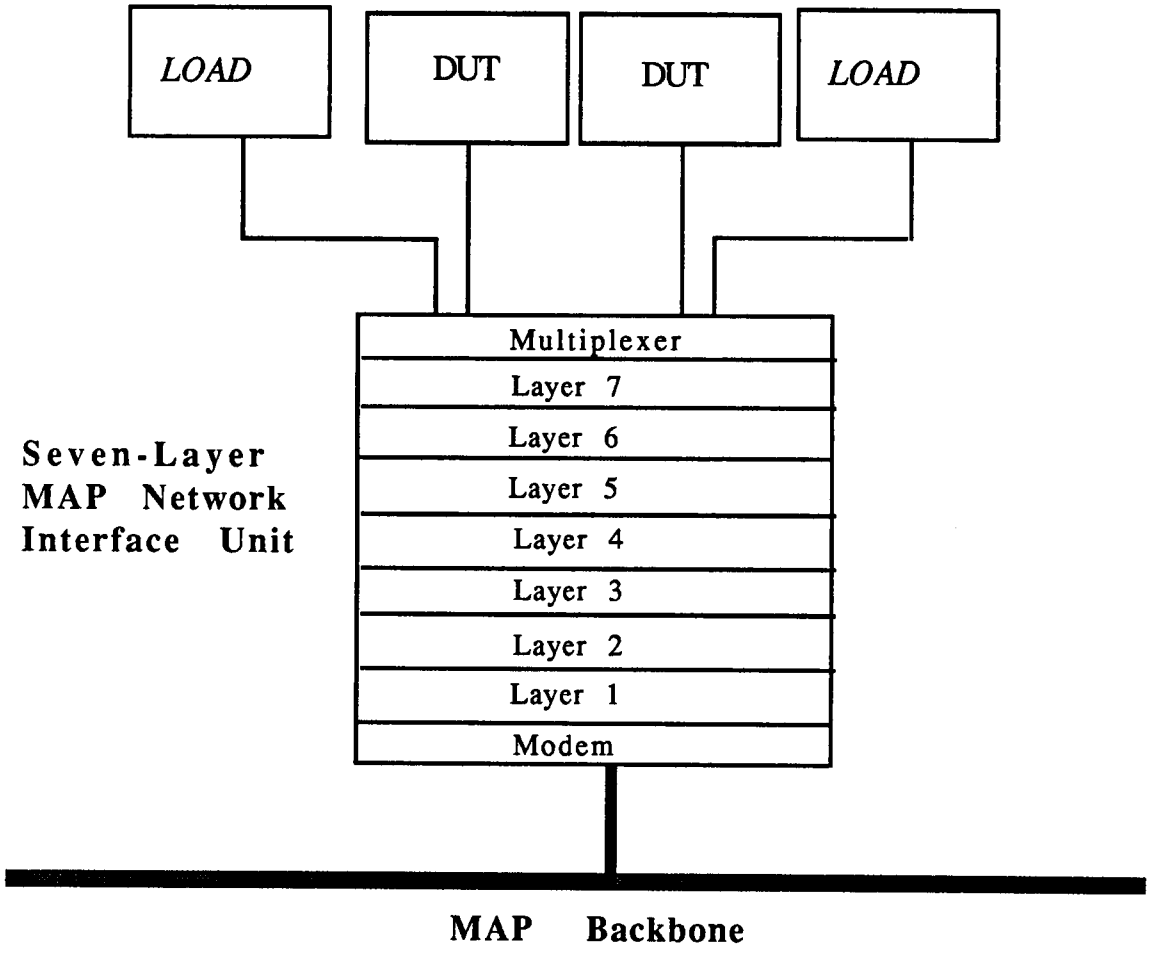
- **Setup 4**

All conditions were the same as described in Setup 2 for no load and load condition, except that this setup used a seven-layer MAP multiplexer instead of two-layer MAP multiplexer. See Figure 3-9 for this setup.



- DUT -- Device Under Test

Figure 3-8. Setup 3 Which Included 2 Two-Layer MAP Multiplexers



- DUT -- Device Under Test

Figure 3-9. Setup 4 Which Included a Seven-Layer MAP Multiplexer

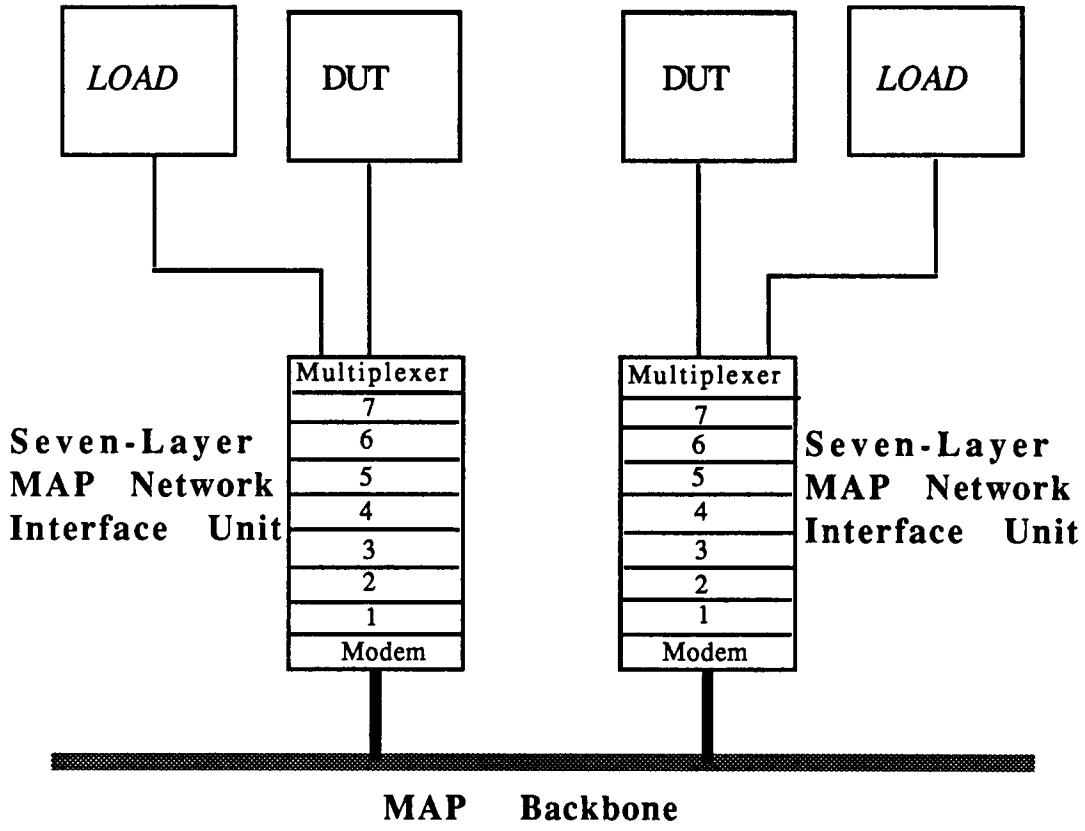
- **Setup 5**

All conditions were the same as described in Setup 3 for no load and load condition, except that this setup used 2 seven-layer MAP multiplexers instead of 2 two-layer MAP multiplexers. See Figure 3-10 for this setup.

For all the message transfers in all the setups described above, each message was sent back and forth between the two sending and receiving devices ten times. Thus, there were twenty occurrences of one transfer of the message for each measurement. The transmission time was obtained by averaging each measurement over twenty transmissions instead of using a single transmission. There were several definitions covering when to start and stop the clock for the measurement of the transmission time and throughput. One had to define the method in order to establish the procedure. The clock was triggered by the first bit of data sent out to the network through the RS232-C cable, and stopped when all the data was received by the designated device. Since there was a start bit, one stop bit, eight data bits, and no parity bit, the throughput was obtained by dividing the message size in bits, which was ten times the number of characters per message, by the time of transmission for one message.

Throughput = message size / time for one message

Message size = 10 bits per char. * no. of chars.



- DUT -- Device Under Test

Figure 3-10. Setup 5 Which Included Two Seven-Layer MAP Multiplexers

3.3. Test Hardware and Software

The network components include:

(1) An IBM Personal Computer XT™* which was used as one of the measuring device. The IBM PC-XT could be either the sending or receiving device.

(2) A Texas Instruments Professional Computer which could be used as another measuring device. The TI PC could be either the sending or receiving device. All the node and network loading was also created using TI Professional Computers.

(3) The two and seven-layer MAP multiplexers were used in the MAP network.

(4) RS232-C cables were used with an average length of 25 feet.

(5) The MAP 2.1 broadband network was used.

(6) A headend remodulator was used.

---

* IBM PC-XT™ is a trademark of the International Business Machines Corporation.

(7) A protocol analyzer was used.

Each device had a program that acted as a dumb terminal which could receive and send messages of variable lengths and baud rates to the other device. The source for this program was Barkakati[3] [10]. The program was modified in order to use it for this project. Since each PC had its own system clock, there was a problem of synchronization of the system clock. This problem was resolved by using only one clock for the timing. Since the system clock used resided in the sending PC, the message that was sent to the receiving PC had to be sent back to the sending PC in order to know when to complete the timing measurement. Then, the measurement of the transmission time for the round trip was divided by two to get the single trip measurement. The timing was measured using the software DOS clock of the IBM PC/XT™.

[3] Nabajyoti Barkakati, MS-DOS Papers (Indianapolis: Howard W. Sams & Company,1988), pp. 477-533.

CHAPTER 4

RS232-C IMPLEMENTATION AT 19.2 KBAUD

The performance tests using 19.2 kbaud were carried out using the Everex Personal Computers. To set the baud rate of 19.2 kbps, the baud rate divisor registers 3F8H and 3F9H (2F8H and 2F9H) were to be set to a number that would divide the system clock of 1,190,000 cycles/seconds to obtain the expected baud rate. For instance, setting the baud rate divisor registers to twelve would obtain the baud rate of 9600 (1,190,000/12), while setting the baud rate divisor registers to six would obtain the baud rate of 19200. To use the registers 3F8H and 3F9H (2F8H and 2F9H) as the baud rate divisor, the line control register 3FBH (2FBH) had to be set to one at bit 7. After inputting the proper baud rate divisor to obtain the desired baud rate, bit 7 of the line control register was then set to zero. The line control register was used to set the proper data format for the serial communication. See Table 4-1 for the bit setting of the line control register.

A load could be created by using two serial ports in one personal computer. A message could be sent from one port of a

Table 4-1. Bit setting of the Line Control Register[1]

Bit	Parameters
0 - 1	Character Length 00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits
2	Number of Stop Bit 0 = 1 1 = 1.5 if character length is 5 1 = 2 if otherwise
3	Parity 1 = parity bit is set 0 = parity bit is not set
4	Parity Type 0 = odd 1 = even

[1] Robert Jourdain, Programmer's Problem Solver for the IBM PC, XT, & AT (New York: Brady, 1986), p.373.

personal computer and received by another port of the same computer. This could be done by setting up interrupts for the receiving port to prepare to receive any message. Then, the polling method was used on the sending port in order to make sure that the port was ready to send the message.

BIOS interrupt (INT 14H) was used for the polling method. First, BIOS function 0 was used to initialize the data format of the serial communication. The DX register was used to determine the use of the desired serial port (COM1, DX = 0 or COM2, DX = 1). The parameters such as word length, number of stop bits, parity, and baud rate were stored in the AL register for initialization. See Table 4-2 for the parameters setting in the AL register for the BIOS function.

BIOS function 1 of INT 14H was used to transmit data. Again, the DX register was used to set the communication path to the desired serial port. The AL register was used to store the character to be transmitted.

Once the BIOS function 0 was used to initialize the port parameters, the BIOS function 1 was issued to send the message. The message had to be completely transmitted before the microprocessor could accept the interrupt for receiving the message at the other port.

Table 4-2. Parameters Setting for BIOS Function[2]

Bit	Parameters
0 - 1	Word Length 10 = 7 bits 11 = 8 bits
2	Number of Stop Bits 0 = 1 1 = 2
3 - 4	Parity Type 00 = none 10 = none 01 = odd 11 = even
5 - 7	Baud Rate 000 = 110 bps 001 = 150 bps 010 = 300 bps 011 = 600 bps 100 = 1200 bps 101 = 2400 bps 110 = 4800 bps 111 = 9600 bps

[2] Robert Jourdain, Programmer's Problem Solver for the IBM PC, XT, & AT (New York: Brady, 1986), p.371.

CHAPTER 5

RESULTS AND CONCLUSION

The results were divided into two sections. The first section consisted of the results taken using the baud rates ranging from 1200 to 9600 bps. The second sections consisted of results taken using 19.2 kbps. Generally, the throughputs were plotted against the baud rates. Thus, throughputs and their corresponding baud rates were related.

5.1. Test Results Taken Under 1200 To 9600 bps

The following results were summarized under no load condition. With reference to the results taken under null modem mode, the percentage decreased in throughputs under different setups for the different lengths of messages were listed as follows:

• For a message of **128** characters, throughput was decreased by 12.7% using a two-layer MAP multiplexer and 17.3% using a seven-layer MAP multiplexer at 9600 bps. At 4800 bps, throughput was decreased by 12% using a two-layer MAP multiplexer and 17% using a seven-layer MAP multiplexer.

• For a message of **256** characters, throughput was decreased by 6.3% using a two-layer MAP multiplexer and 9.65% using a seven-layer MAP multiplexer at 9600 bps. At 4800 bps, throughput was decreased by 6% using a two-layer MAP multiplexer and 9% using a seven-layer MAP multiplexer.

• For a message of **512** characters, throughput was decreased by 3.2% using a two-layer MAP multiplexer and 4.5% using a seven-layer MAP multiplexer at 9600 bps. At 4800 bps, throughput was decreased by 3% using a two-layer MAP multiplexer and 4% using a seven-layer MAP multiplexer.

• For a message of **1024** characters, throughput was decreased by 1.8% using a two-layer MAP multiplexer and 3% using a seven-layer MAP multiplexer at 9600 bps. At 4800 bps, throughput was decreased by 1.5% using a two-layer MAP multiplexer and 2.8% using a seven-layer MAP multiplexer.

Furthermore, there was an 0.8% decrease in the throughput taken using 2 two-layer MAP multiplexers as compared to the results taken under the setups using a single two-layer MAP multiplexer. However, there was a 1% decrease in throughput taken using 2 seven-layer MAP multiplexers compared to the

results taken under the setups using a single seven-layer MAP multiplexer.

Under load condition, the throughput was decreased by:

- 5% compared to the results taken under no load condition for a message length of 128 characters at 9600 and 4800 bps.

- 3.2% compared to the results taken under no load condition for a message length of 256 characters at 9600 and 4800 bps.

- 2.2% compared to the results taken under no load condition for a message length of 512 characters at 9600 and 4800 bps.

- 1% compared to the results taken under no load condition for a message length of 1024 characters at 9600 and 4800 bps.

For each length of message under no load or load condition, the throughput at 2400 bps was about half of that at 4800 bps using either a two-layer or a seven-layer MAP multiplexer. The percentage decrease at 1200 bps was about half of that at 2400 bps using either a two-layer MAP multiplexer or a seven-layer MAP multiplexer.

Overall, 1024 chars/msg gave the highest throughput while 128 chars/msg gave the lowest throughput in all the setups and under different conditions. Figure 5-1 and Figure 5-2 show that the longer the message length, the higher the throughput. The increase of throughput at longer message lengths were due to the batch processing of characters.

Among all of the five setups, the transmission time for null modem mode was the shortest and the throughput was the best regardless of the baud rates and the message lengths due to the direct connection between two application devices. The throughput for the null modem setup was used to define a baseline for comparison with other setups.

By comparing the results for the setups that involved only the two-layer MAP multiplexer, Setup 2 had a higher throughput and shorter transmission time than that of the Setup 3 under the condition when either no load or loading existed. The message, coming out from one device at one MAP multiplexer, had to wait to be switched and travel out to the network path before it could reach the receiving device at another MAP multiplexer, while the message coming from the sending device at a port of a MAP multiplexer in Setup 2 only needed to go through the MAP layers within the switch before it reached the other port at the same MAP

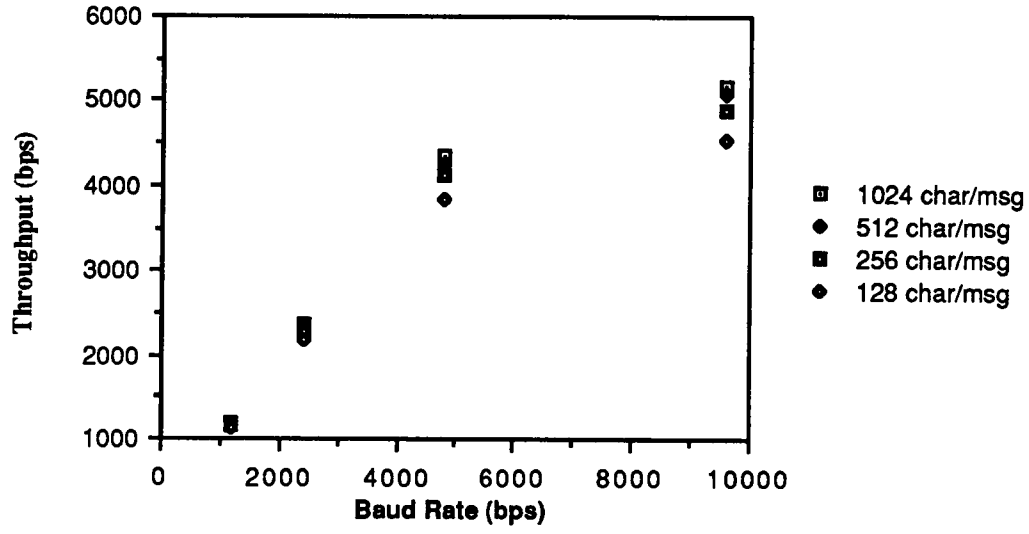


Figure 5-1 Results For Two-layer MUX Under Different Message Lengths

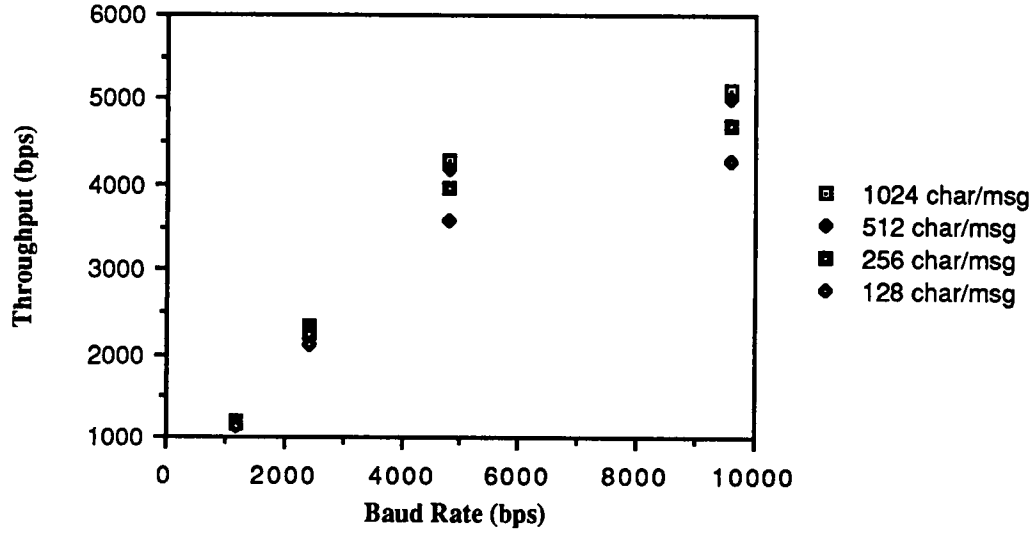


Figure 5-2 Results For Seven-Layer MUX Under Different Message Lengths

multiplexer. Setup 5 had longer transmission time and lower throughput than that of Setup 4 for the reason described above.

Generally, the setups that used the two-layer MAP multiplexer had better throughput and shorter transmission time than that of the seven-layer MAP multiplexer. The seven-layer MAP multiplexer required more transmission time because a message needed to travel through all seven layers in the seven-layer MAP multiplexer. On the other hand, shorter delay resulted in the two-layered MAP multiplexer since data only needed to pass through two layers. See Figure 5-3 and Figure 5-4 for illustration for no load condition. Figure 5-5 and Figure 5-6 show that the addition of traffic by other loads in the network besides the traffic created by the testing devices did affect the performance.

Even though throughput seemed to increase in proportion to the baud rate, notice that the throughput started to level off at 4800 baud when using the IBM PC-XT™ and TI PC as the measuring devices. This roll-off region was caused by the delay resulted from the way the network was constructed. Since each message had to go through each layer of the network during the communication process, flowing speed of data deteriorated at higher speed like 4800 bps.

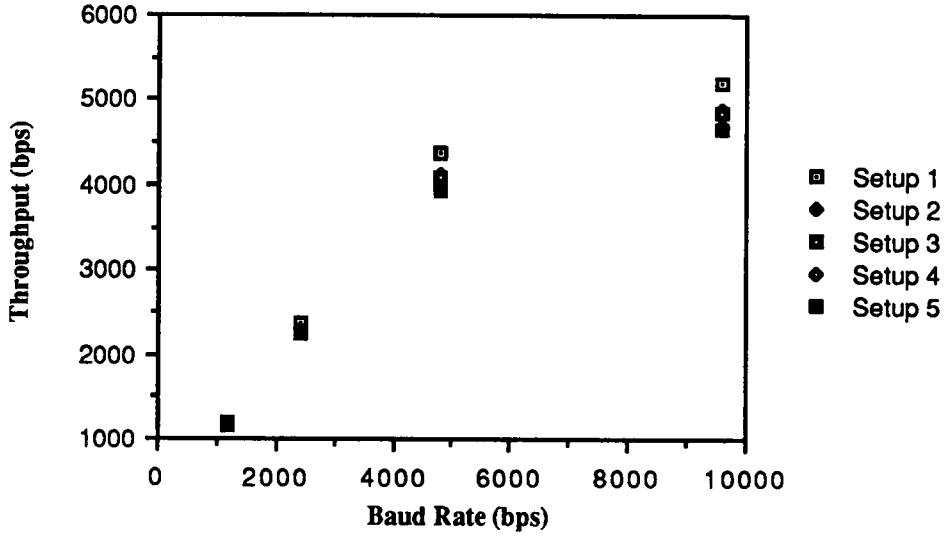


Figure 5-3 Results Under No Load Condition Using 256 char/msg

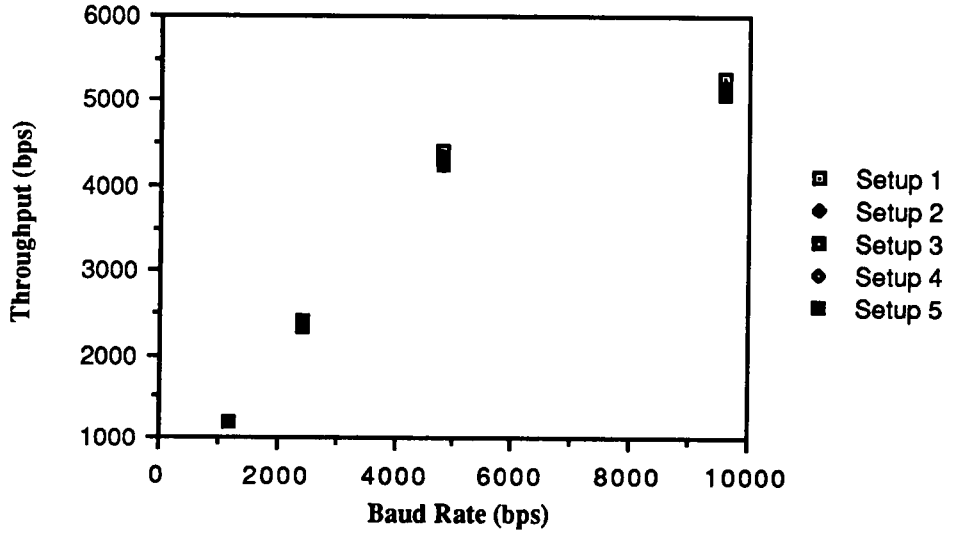


Figure 5-4 Results Under No Load Condition Using 1024 char/msg

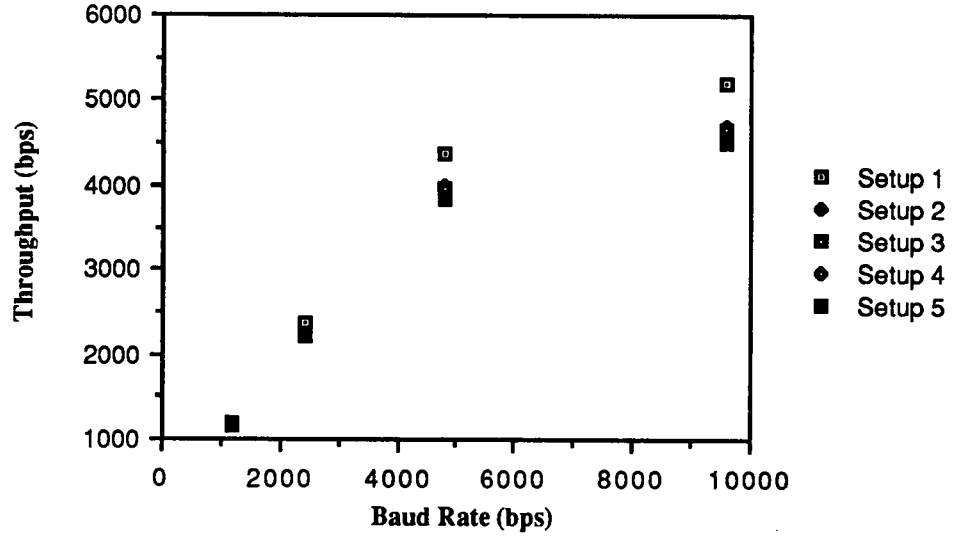


Figure 5-5 Results Under Loading Condition Using 256 char/msg

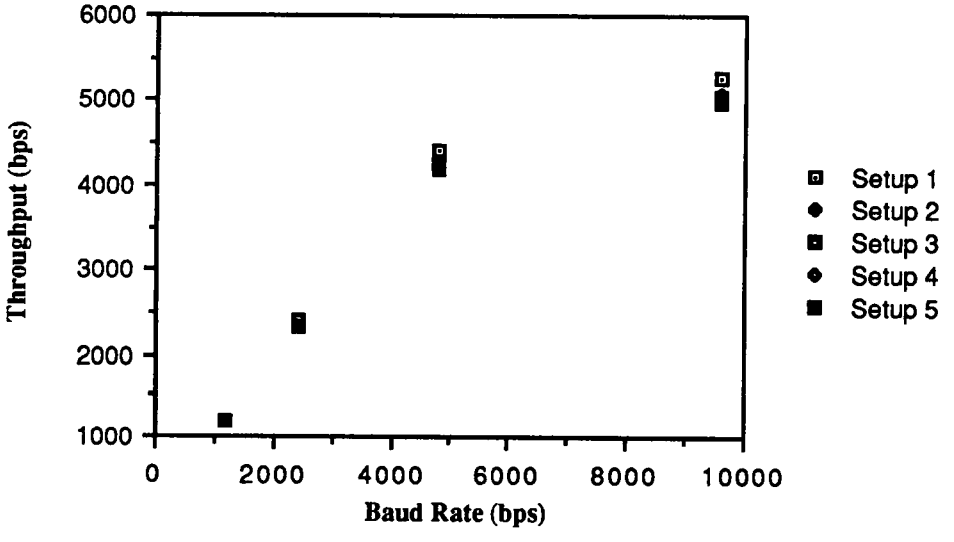


Figure 5-6 Results Under Loading Condition Using 1024 char/msg

5.2. *Test Results Taken Under 19.2 kbps*

The percentage decrease of throughput compared with the null modem mode and a two-layer MAP multiplexer at 19.2 kbps were as follows:

- For the message length of 128 characters, there was a 30% decrease in throughput.
- For the message length of 256 characters, there was a 18% decrease in throughput.
- For the message length of 512 characters, there was a 9% decrease in throughput.
- For the message length of 1024 characters, there was a 5% decrease in throughput.

The percentage decrease of throughput compared with the null modem mode and a seven-layer MAP multiplexer at 19.2 kbps were as follows:

- For the message length of 128 characters, there was a 33% decrease in throughput.
- For the message length of 256 characters, there was a 20% decrease in throughput.
- For the message length of 512 characters, there was a 11% decrease in throughput.
- For the message length of 1024 characters, there was a 6% decrease in throughput.

By comparing Figure 5-7 and Figure 5-8, there was a relative increase in throughput at the baud rates of 1200 to 9600 bps in Figure 5-8. Furthermore, additional throughputs at 19.2 kbps were provided in Figure 5-8 but not in Figure 5-7 because the setups that used the TI PCs and IBM PC-XT™ in Figure 5-7 had the output limited to 9600 bps due to their clock speeds. The setups which involved the Everex Step-16 PCs in Figure 5-8 had a clock speed of 16 MHZ. Figure 5-7 and Figure 5-8 were combined together in Figure 5-9 to give a better comparison.

5.3. Frame Format From the Network Analyzer

The frames passing through the MAP network were defined by the IEEE 802.4 standards, which specified a token bus network. According to the IEEE 802.4 standards, data link layer provided the frames with the minimum amount of information to be transmitted over the network successfully.

As shown in Figure 5-10, each frame consisted of the preamble, the start delimiter, the basic frame, the Frame Check Sequence (FCS), and the end delimiter. The preamble, the start delimiter, FCS, and the end delimiter were appended to the frame before the frame was sent out over the network by the Token Bus

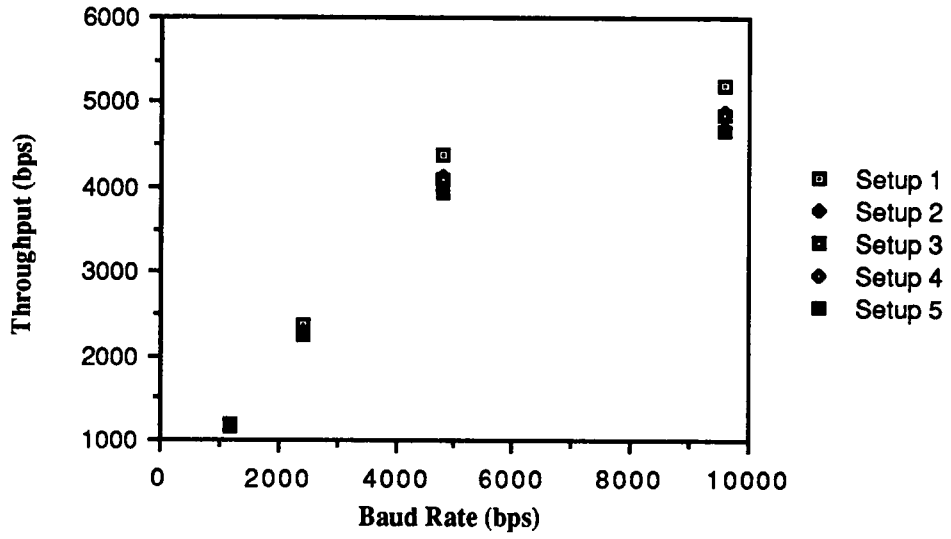


Figure 5-7 Results Under No Load Condition Using IBM PC-XT and TI PC at 256 char/msg

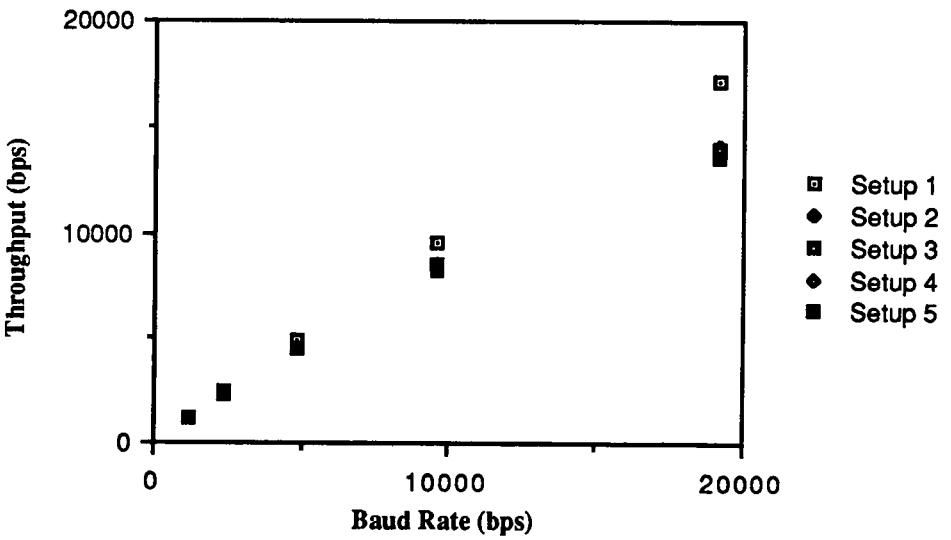


Figure 5-8 Results Under No Load Condition Using Everex at 256 char/msg

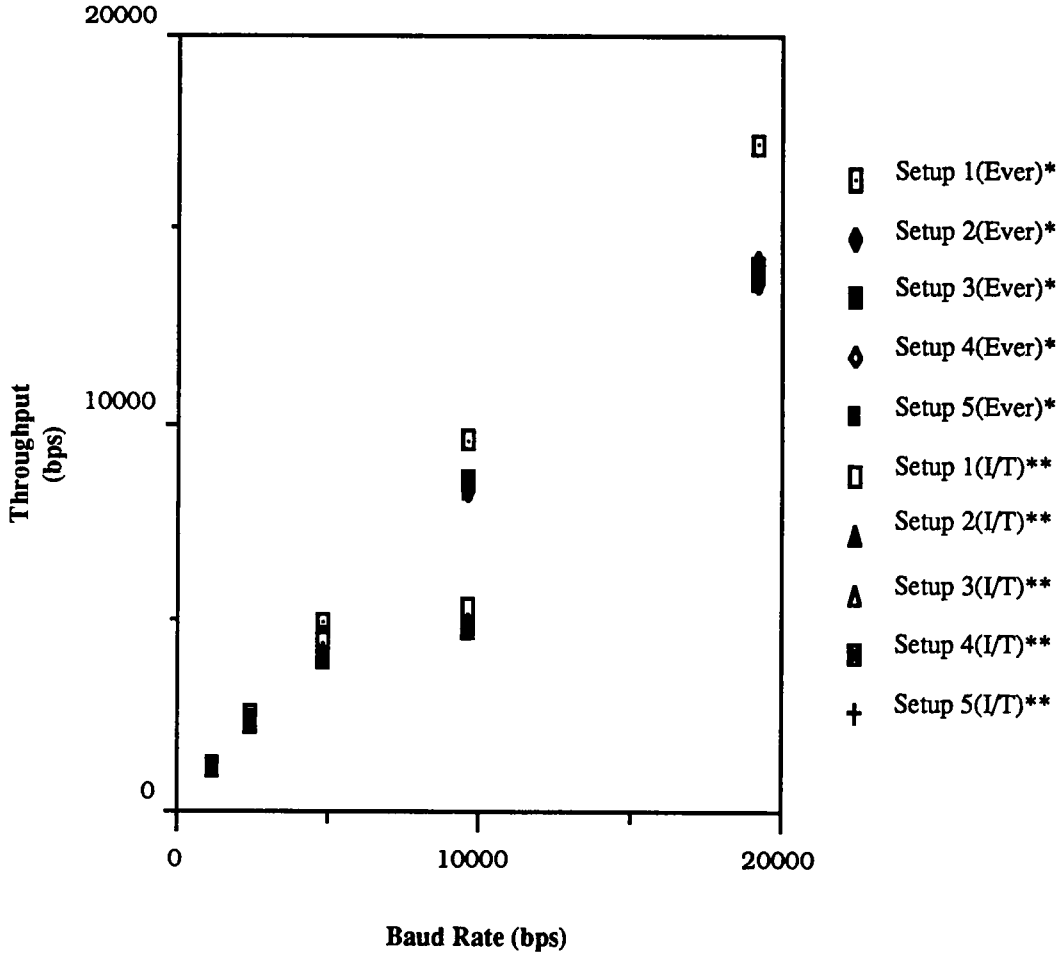


Figure 5-9 Results Under No Load Condition Using Everex or IBM/TI at 256 char/msg

* Using 16 MHZ Everex Personal Computers

** Using 4.77 MHZ IBM PC-XT and 5 MHZ TI PC

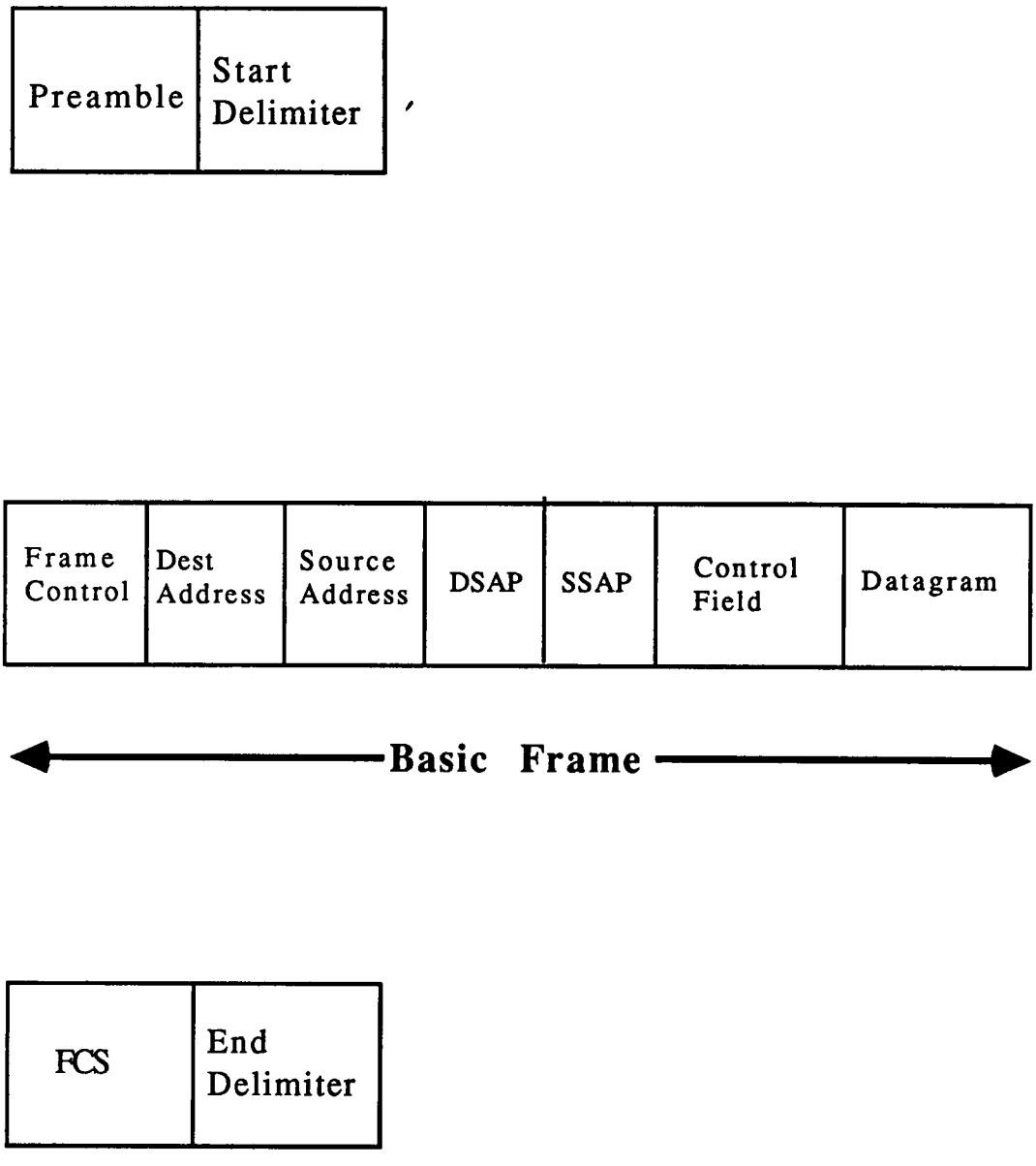


Figure 5-10. Frame Format at the Data Link Layer[1]

[1]Industrial Networking Incorporated, MSS-30 Network Traffic Analyzer User's Manual (1987), p.3-4.

Handler (TBH). However, the preamble, the start delimiter, and the end delimiter were discarded by the TBH before the network analyzer buffer stores any of the frames detected. Thus, the preamble, the start delimiter, and the end delimiter were not included in the display of the network analyzer.

The basic frame consisted of Frame Control, Destination Address, Source Address, Destination Service Access Point (DSAP), Source Service Access Point (SSAP), Control Field, and Datagram. The Frame Control code consisted of 8 bits, which were displayed in a two-character format. Then, the Destination Address, as well as the Source Address, consisted of 48 bits each. Each field of the DSAP, SSAP, and Control Field contained 8 bits of information. The FCS was made up of 32 bits. Figure 5-11 shows a frame in the 802.4 format. The Data field consisted of user data and the control information appended by the Transport, InterNet, Session, and Case layers in the MAP specification. If a layer was not needed in the process of communication, then the subsequent layer continued to add its control information to the frame [11].

The OSI Transport Protocol Data Unit (TPDU) consisted of the length field for the length of the fixed and variable header, the header depending on which type of TPDU was used, the variable header, and the user data field. There were ten different types of TPDU, but only the Data (DT), and the Data Acknowledgment TPDU were needed for the network interface unit used in this research.

Frame Control	Destination Address	Source Address
8 bits	48 bits	48 bits

DSAP	SSAP	Control Field	Data	FCS	End Delimiter
8 bits	8 bits	8 bits	N bits	32 bits	8 bits

Figure 5-11. 802.4 Display Format of a Frame[2]

[2] Industrial Networking Incorporated, MSS-30 Network Traffic Analyzer User's Manual (1987), p.3-5.

Figure 5-12 illustrates the Data and Data Acknowledgment TPDUs. The LI (Length Indicator) field had the length of the total header field (fixed and variable) expressing in bytes, not including the LI field itself. The field followed by the LI field was the fixed part of the header which consisted of a unique code for each TPDU depending on which TPDU was used. The Destination reference field identified the destination for transport connection. This field was needed since the Transport layer could multiplex more than one transport connection over one network connection. Hence, the Destination reference was used to identify a particular TPDU with the correct transport connection. The variable part of the header was divided into different fields with each field consisting of a 1-byte type field, a 1-byte length field, and the data itself. This was an option field that was not always used. The user data field consisted of data specified by the user [12].

Figure 5-13 illustrates the frames going from one MAP multiplexer to another MAP multiplexer captured by a protocol analyzer. The sequence number, *Sqnce#*, denoted the sequence of frames captured by the network analyzer. The *Time received* was the time that the frames was captured. The P7 (Data Frame Priority 7) was the Frame Control field. The Destination Address (either 080060002183 or 080060002149) and the Source Address (either 080060002183 or 080060002149) then followed. The two columns of FE were the DSAP (Destination Service Access Point) and

Data TPDU Format

LI	1111	0000	Destination reference	User Data

Data Acknowledgment TPDU Format

LI	0110	0000	Destination reference	TPDU expected	Variable part

Figure 5-12. The Data and Data Acknowledgment TPDUs[3]

[3] Andrew S. Tanenbaum, Computer Networks (Englewood Cliffs, N.J.: Prentice-Hall, 1988), p.425.

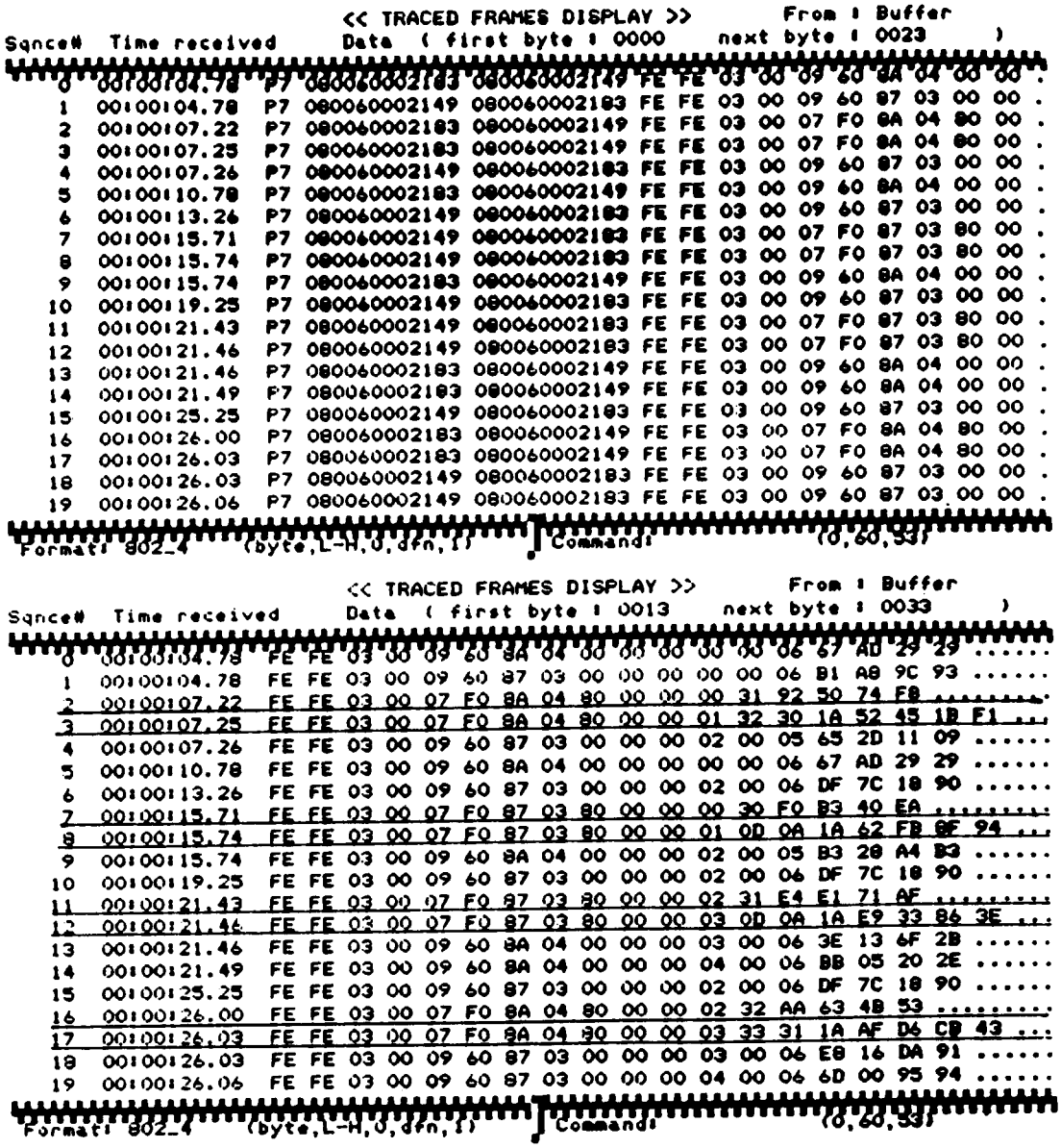


Figure 5-13. The Frames Captured by a Network Traffic Analyzer

the SSAP (Source Service Access Point), respectively. The next field was the Control Field which was usually defined by 03 hex. The one-byte field 00 indicated that the Internet layer was inactive. Then, the field which consisted of either 07 hex or 09 hex denoted the length of the Transport layer information in bytes. The subsequent field described the TPDU (Transport Protocol Data Unit) type. 'F0' indicated a Data Acknowledgment TPDU was present, while '60' indicated that a Data TPDU existed.

The next 16-bit field denoted the Destination Reference (either was 8A 04 hex or 87 03 hex). The fourth byte after the Destination Reference, which was immediately before the User Data, was the sequence number of the frame. This sequence number was used to put the frames that contained User Data pertaining to a particular Destination Reference into sequence. It started from 00 hex. Only the subsequent frames contained User Data pertaining to the same Destination Reference were sequenced in the consecutive order. If two frames were used to contain the User Data sent to the same Destination Reference, then two consecutive sequence numbers were used to put these frames in order. For example, Sqnce# 2 (the first underlined frame) had sequence number 00 hex which was used for the frame that contained User Data 31 hex with 8A 04 hex as the Destination Reference. Then 01 hex in Sqnce# 3 (the second underlined frame) was used for the frame that contained User Data 32 hex and 30 hex using the same Destination Reference. The frames that contained User Data pertaining to a

different Destination Reference were sequenced in different order. After the User Data, four bytes of Frame Check Sequence always trailed at the end.

5.4. Conclusion

The transfer of longer messages resulted in a higher throughput rate than the transfer of shorter messages on the MAP network. This difference in throughput using different message lengths was due to the overhead appended on each packet after going through each layer in the network interface unit.

The greater the functionality of the MAP multiplexer (two versus seven-layer implementation in the network environment), the worse the performance. For example, the two-layer MAP multiplexer would only allow messages to be transferred in the two-layer devices environment. This limited the functionality of the network adapter but its performance was better than that of the seven-layer MAP multiplexer. A seven-layer MAP multiplexer could work in any environment for communicating with different seven-layer devices. However, its performance was not as good as a two-layer MAP multiplexer.

According to the results gathered under the conditions specified, the decrease in throughputs using the MAP multiplexers

were larger at higher baud rates and shorter messages with respect to null modem mode.

One had to keep in mind that the messages used were stated in block lengths. Therefore, the MAP multiplexer provided an acceptable way of transferring blocks of information between RS-232-C devices and MAP devices using a full seven layer implementation.

5.5. *Future Work*

This research can be used as a kernel for future development. This kernel, which uses the different setups described above, can act as a network analyzer that measures the amount of load existing on a MAP network and determine the optimum and maximum load conditions that the network can handle. Mathematical models can also be developed on a MAP network that uses different network devices besides the multiplexers so that study of the traffic condition using different devices can be obtained too.

The performance measurements can be further applied to the MAP network with the traffic created by passing data between any host system and a manufacturing model controlled by different vendors' programmable logic controllers, data from a temperature broadcast system, and data between PCs which are directly

connected to the MAP network without going through any RS232-C multiplexer. Additional tests with each setup should be run with block lengths of 8 or fewer characters for the purpose of establishing throughput loading associated with header and control information.

LIST OF REFERENCES

LIST OF REFERENCES

[1] Schwartz, Mischa, Telecommunication Networks: Protocols, Modeling and Analysis, Addison-Wesley, Reading, Mass., 1987.

[2] MAP Specification Versions 2.1.A & 2.2, General Motors Technical Center, Michigan, 1986.

[3] Larkin, Anne-Marie and Miller, Arthur R., "Testing -- Proving the Reality of MAP," Ultratech Conference Proceedings, vol. 1, Society of Manufacturing Engineers, Long Beach, CA., September 1986.

[4] Marathe, Madhav V. and Smith, Robin A., "Performance of a MAP Network Adapter," IEEE Network, vol. 2, no. 3, May 1988.

[5] Hammond, Joseph L. and O'Reilly, Peter J. P., Performance Analysis of Local Computer Networks, Addison-Wesley, Reading, Mass., 1986.

[6] Suppan-Borowka, Jurgen, "Analyzing the Performance of the MAP Backbone," Ultratech Conference Proceedings, vol. 1, Society of Manufacturing Engineers, Long Beach, CA., September 1986.

[7] Seyer, Martin D., RS-232 Made Easy, Prentice-Hall, Englewood Cliffs, N.J., 1984.

[8] Sargent III, Murray and Shoemaker, Richard L., The IBM Personal Computer From the Inside Out, Addison-Wesley, Reading, Mass., 1984.

[9] Jourdain, Robert, Programmer's Problem Solver for the IBM PC, XT, & AT, Brady, New York, 1986.

[10] Barkakati, Nabajyoti, MS-DOS Papers, Howard W. Sams & Company, Indianapolis, Indiana, 1988.

[11] Industrial Networking Incorporated, MSS-10 Network Traffic Analyzer User's Manual, 1987.

[12] Tanenbaum, Andrew S., Computer Networks, Prentice-Hall, Englewood Cliffs, N.J., 1988.

VITA

VITA

Victoria Ivy Chung was born in Jambi, Indonesia. She went to elementary and secondary schools in Singapore before coming to the United States of America for college education. She became a U.S. citizen in 1989. She graduated with a Bachelor of Science degree in Electrical Engineering in August 1987 at the University of Tennessee, Knoxville. She continued to study for the Master of Science degree in the same field in September 1987.

She went to the International Business Machines Corporation (IBM) in Charlotte, North Carolina to co-op for four quarters, and had opportunities to work in the printer development and testing areas. Computer networking was her research area during her graduate study. She is working for NASA, Langley Research Center in Hampton, Virginia after graduating with the M.S.E.E. degree. She is a member of the IEEE, Eta Kappa Nu Electrical Engineering Honor Society, and Tau Beta Pi Engineering Honor Society.