

**U.S. DEPARTMENT OF COMMERCE
National Technical Information Service**

AD-A032 031

Design of a Retransmission Strategy for Error Control in Data Communication Networks

Massachusetts Inst of Tech Cambridge Electronic Systems Lab

Jul 76

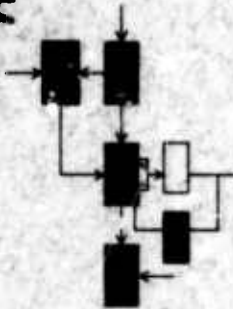
AD A032031

322100

July, 1976

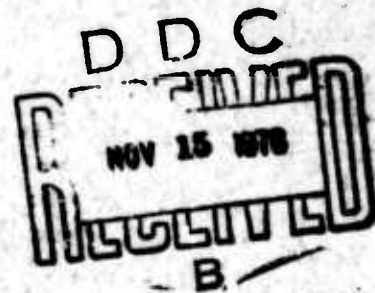
Report ESL-R-674

Grant NSF-ENG75-14103



DESIGN OF A RETRANSMISSION STRATEGY FOR ERROR CONTROL IN DATA COMMUNICATION NETWORKS

Seyyed J. Golstaani



REPRODUCED BY
NATIONAL TECHNICAL
INFORMATION SERVICE
U. S. DEPARTMENT OF COMMERCE
SPRINGFIELD, VA. 22161

DISTRIBUTION STATEMENT A

Approved for public release
Distribution Unlimited

Electronic Systems Laboratory

MASSACHUSETTS INSTITUTE OF TECHNOLOGY, CAMBRIDGE, MASSACHUSETTS 02139

Department of Electrical Engineering and Computer Science

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DESIGN OF A RETRANSMISSION STRATEGY FOR ERROR CONTROL IN DATA COMMUNICATION NETWORKS		5. TYPE OF REPORT & PERIOD COVERED Technical Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Seyyed J. Golestaani		8. CONTRACT OR GRANT NUMBER(s) ARPA Order No. 3045/5-7-75 ONR/N00014-75-C-1183
9. PERFORMING ORGANIZATION NAME AND ADDRESS Massachusetts Institute of Technology Electronic Systems Laboratory Cambridge, Massachusetts 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Program Code No. 5T10 ONR Identifying No. 049-383
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, Virginia 22209		12. REPORT DATE July, 1976
		13. NUMBER OF PAGES 113
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Office of Naval Research Information Systems Program Code 437 Arlington, Virginia 22217		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Error Detecting Codes Link Protocols		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) Several retransmission strategies with different degrees of flexibility for various applications in data communication between two terminals are analyzed and compared with each other. A method for proving the correctness of a strategy is developed and used in each case. For the Roll Back K scheme (used in constant block length and continuous transmission systems) this proof resulted in modifying the scheme into a form which does not require any space specified for the purpose of error control (like acknowledgement or repeat request)!		

20.

A new strategy for continuous transmission systems with variable block length (which is a rather general situation) is designed, using about one bit of error control data per block. It is an ordered retransmission scheme which assumes a constant round trip delay for the channel and takes advantage of this to specify block(s) in demand for retransmission. It has an advantage over other schemes designed earlier in the sense that it does not interpret an erased incoming block, as a block containing a repeat request (RQ), hence is more efficient. The strategy is then showed to be applicable to more specific cases such as constant block length and/or stop and wait transmission.

ACCESSION for	
NTIS	White Section ☒
DOC	Buff Section ☐
UNANNOUNCED	☐
JUSTIFICATION	
BY	
DISTRIBUTION/AVAILABILITY CODES	
Dist.	AVAIL. and/or SPECIAL
A	

July, 1976

Report ESL-R-674

DESIGN OF A RETRANSMISSION STRATEGY FOR ERROR CONTROL
IN DATA COMMUNICATION NETWORKS

by

Seyyed J. Golestaani

This report is based on the unaltered thesis of Seyyed J. Golestaani, submitted in partial fulfillment of the requirements for the degree of Master of Science and Electrical Engineer at the Massachusetts Institute of Technology in May, 1976. This research was conducted at the M.I.T. Electronic Systems Laboratory with partial support provided by the National Science Foundation under Grant NSF-ENG75-14103.

Electronic Systems Laboratory
Department of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

DESIGN OF A RETRANSMISSION STRATEGY FOR ERROR CONTROL
IN DATA COMMUNICATION NETWORKS

by

Seyyed Jamaaloddin Golestaani

SB, Arya-Mehr University of Technology, Tehran, Iran
(1973)

SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREES OF

MASTER OF SCIENCE

And

ELECTRICAL ENGINEER

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May, 1976

Signature of Author. *S. Jamaaloddin Golestaani*
Department of Electrical Engineering and
Computer Science, May 21, 1976

Certified by Thesis Supervisor

Accepted by.
Chairman, Departmental Committee on Graduate Students

DESIGN OF A RETRANSMISSION STRATEGY FOR ERROR CONTROL
IN DATA COMMUNICATION NETWORKS

by

Feyyed Jamaaloddin Golestaani

Submitted to the Department of Electrical Engineering
and Computer Science on May 21, 1976 in partial fulfillment
of the requirements for the Degrees of Master of Science
and Electrical Engineer

ABSTRACT

Several retransmission strategies with different degrees of flexibility for various applications in data Communication between two terminals are analyzed and compared with each other. A method for proving the correctness of a strategy is developed and used in each case. For the Roll Back K scheme (used in constant block length and continuous transmission systems) this proof resulted in modifying the scheme into a form which does not require any space specified for the purpose of error control (like acknowledgement or repeat request)!

A new strategy for continuous transmission systems with variable block length (which is a rather general situation) is designed, using about one bit of error control data per block. It is an ordered retransmission scheme which assumes a constant round trip delay for the channel and takes advantage of this to specify block(s) in demand for retransmission. It has an advantage over other schemes designed earlier in the sense that it does not interpret an erased incoming block, as a block containing a repeat request (RQ), hence is more efficient. The strategy is then showed to be applicable to more specific cases such as constant block length and/or stop and wait transmission.

THESIS SUPERVISOR: Robert G. Gallager
TITLE: Professor of Electrical Engineering

ACKNOWLEDGEMENTS

Professor Robert G. Gallager has been very helpful in supervising my research. I am greatly indebted to him for all his help and I would like to express my sincere gratitude to him.

I would like also to thank Mr. Arthur J. Giordani of Electronic Systems Laboratory for his help in drawing the diagrams of this thesis. Thanks also goes to Mrs. Sarah Tower for the typing.

TABLE OF CONTENTS

	<u>Page</u>
Title Page	1
Abstract	2
Acknowledgements	3
Table of Contents	4
List of Figures	8
 Chapter I INTRODUCTION	
1.1 Retransmission Strategy for Error Control	10
1.2 Historical Background and Description of the Problem	12
1.3 Outline and Preview	14
 Chapter II STOP AND WAIT STRATEGIES	
2.1 Noiseless Feedback Channel.....	17
2.2 Noisy Feedback Channel	20
2.2.1 Introduction	20
2.2.2 An Inadequate Strategy	23
2.2.3 Lynch's Design for the Noisy Feedback Channel.....	23
2.2.4 Ready and Acknowledge with Transition Signaling.....	24
2.3 Full Duplex Transmission	26

		<u>Page</u>
2.3.1	Decomposition Approach	26
2.3.2	Lynch's Scheme and Bartlett's Scheme for Full Duplex Transmission.....	29
2.4	Half Duplex Channel	30

Chapter III CONTINUOUS TRANSMISSION STRATEGY FOR CONSTANT BLOCK LENGTH

3.1	Introduction.....	33
3.2	Selective Retransmission Scheme	35
	3.2.1 Decomposition Approach.....	35
	3.2.2 Buffer Limitation in Decomposition Approach.....	38
3.3	Roll Back K Scheme.....	39
	3.3.1 Introduction.....	39
	3.3.2 Roll Back 2 Scheme.....	42
	3.3.3 Huristic Arguments for Roll Back 2 Scheme.....	44
3.4	Justification for Roll Back 2 Scheme....	47
	3.4.1 State Representation.....	47
	3.4.2 Justification	53
3.5	Retransmission Strategy with No Control Bit.....	54
3.6	Justification for the Schemes Mentioned in 2.2.3 and 2.2.4.....	57
	3.6.1 Lynch's Scheme.....	57
	3.6.2 Ready and Acknowledge with Transition Signaling.....	64

3.6.3	Comparison Between Bartlett's Scheme and Lynch's Scheme.....	66
-------	--	----

Chapter IV RETRANSMISSION STRATEGY FOR VARIABLE
BLOCK LENGTH AND CONTINUOUS TRANSMISSION
SYSTEMS

4.1	Introduction.....	70
4.2	How to Specify Blocks in Demand for Retransmission.....	73
4.2.1	The Method Adopted by IBM in SDLC.....	73
4.2.2	Getting Rid of the Count Numbers	75
4.2.3	Multiple Acknowledgments or Repeat Requests within one block.....	77
4.3	Developing a Retransmission Procedure....	80
4.3.1	Indication for a Retransmission Being Started.....	80
4.3.2	Erasures to be Taken as acknowledgements.....	83
4.4	Proposed Strategy.....	86
4.4.1	Appropriate Retransmission Procedure.....	86
4.4.2	Huffman Code for Error Control Signal.....	90
4.4.3	Lack of Synchronism after Erasures.....	92
4.5	Conclusion	98

	<u>Page</u>
4.5.1 Efficiency Considerations.....	98
4.5.2 Implementation of the Strategy for Specific Cases...	99
4.5.3 Suggestions for Further Research.....	103
BIBLIOGRAPHY.....	107

LIST OF FIGURES

	<u>Page</u>
Figure 2.1	18
Figure 2.2	22
Figure 2.3	25
Figure 2.4	27
Figure 2.5	28
Figure 2.6	31
Figure 3.1	34
Figure 3.2	36
Figure 3.3	41
Figure 3.4	43
Figure 3.5	46
Figure 3.6	49
Figure 3.7	51
Figure 3.8	58
Figure 3.9	59
Figure 3.10	60
Figure 3.11	62
Figure 3.12	
Figure 3.13	65
Figure 3.14	67
Figure 3.15	69
Figure 4.1	71
Figure 4.2	74

		<u>Page</u>
Figure	4.3	78
Figure	4.4	81
Figure	4.5	85
Figure	4.6	88
Figure	4.7	89
Figure	4.8	95
Figure	4.9	97
Figure	4.10	100
Figure	4.11	102
Figure	4.12	104
Figure	4.13	105

I. INTRODUCTION

1.1 Retransmission Strategy for Error Control

Since in practice there are no noiseless communication channels, we may say that the problem of errors introduced in signals after being passed through a channel is as old as signal communication itself. Naturally the first way to decrease this difficulty is to improve the performance of the channel by increasing its capacity and/or using better methods of modulation. However if the nature of the signal and its application is such that it is very sensitive to channel noise, we have to use methods of either error detection or correction.

In the case of data communication this can be done by using check bits for error correction. In the 1950's much effort was devoted to finding different ways of so doing (1). Theoretically by increasing both the number of check bits and information bits in each block. The probability of error can be made arbitrarily small if data rate is less than the capacity of the channel. In many types of digital data lines, as experience shows, errors tend to cluster together into bursts. Unfortunately, because of the wide variability of the duration of these bursts, error correction

does not appear to be practical on such channels if very low error probabilities are required.

Another technique for error control which has become standard practice is error detection plus retransmission upon request (Retransmission strategy). In this method each incoming block consisting of L bits of information is first encoded into an encoded block of the length $N = L + NC$ bits, and then sent. Usually the first L bits of such a block are the same as the incoming information block and the next NC bits are functions of the first part. These NC extra bits (check bits) provide means for the receiver to determine whether the received block is error-free or erroneous (which we refer to as an "Erasure"). We can decrease the probability of having undetected erasures to any arbitrary value by increasing NC (regardless of how large L is). For example a 255,231 code ($L=231, NC=24$) used in the Bell A-1 data system yields a figure of approximately one undetected erasure per 300 years (2). For this reason we assume throughout this thesis that there will be no undetected erasure in an error detecting system. Notice that having a large number of check digits in a block does not decrease efficiency of the channel necessarily because we can increase L at the same time.

After errors are detected in a received block, the receiver will ask the sender through a feedback channel (or even the same channel if not busy) to retransmit that block. Since error detection procedures are much simpler than error correction, implementation of retransmission strategies is relatively simple*. One important advantage of retransmission strategies over error correction codes is that when noise increases and channel capacity decreases, in the first case throughput of the system decreases (due to the more frequent retransmissions) while in the second case reliability decreases and throughput stays constant. Therefore retransmission strategy in a sense provides an automatic control of the throughput.

1.2 Historical Background and Description of the Problem

The concept of sending a feedback message to request retransmission of a word in error goes back at least to the 1950's, and considerable research was devoted to it around 1960 (2,3,8,11). Since 1968 many attempts have been made to design and implement networks to link computers in different locations. It is obvious that in this case blocks of information digits and the associated control

* It is also possible to implement retransmission strategies with a mixture of error correction and detection codes together. The receiver corrects errors if they are only a few and requests retransmission otherwise. As an example refer to Coding for Two-Way Channels (3).

information must be communicated exactly and without error, because having a few errors may cause the whole system to fail. Therefore in the past few years much research has been carried out concerning methods of queuing information sequences, routing messages, and achieving error control which are useful, efficient and reliable for use in computer networks.

There are already a number of retransmission strategies for error control in data communication networks, with different degrees of efficiency and flexibility for various applications. Our main objective is to design an efficient retransmission strategy for binary digital communication between two terminals * under full duplex, continuous transmission with variable block length. Existing designs for this case, such as IBM's SDLC (4) and the very similar proposed International Standard, HDLC (5) use at least 16 control bits in every block in transmission and hence are inefficient. We want to investigate whether another strategy can be designed using fewer control bits per block but having the same generality and flexibility and in fact we will come up with such a strategy.

*

The results however are easily adaptable to the non-binary case. Network communication (more than two terminals) will not be considered in this thesis and requires more research.

1.3 Outline and Preview

Although our major objective is to deal with the problem of retransmission strategies in a special case, we are going to consider the general problem. This is for two reasons: First of all it clarifies and discloses different aspects and difficulties of the ultimate problem which are not evident in the beginning, and therefore it helps to understand the final design and its importance more clearly. Secondly, by considering the general problem, we can present the results of previous investigations and make suggestions or useful modifications wherever possible.

This can be done best if we try to deal with the problem systematically, starting from the simplest case and ending with the most complicated one, rather than presenting different works chronologically. In order to approach the problem systematically, we distinguish the following sets of alternatives:

(a) Full duplex or half duplex communication channel:

The communication channel between two terminals can be made of two links or just a single link* .

* It can not be a simplex channel, because for the receiver to send acknowledgement (or request for retransmission) we always need to communicate both ways over the channel.

- (b) Full duplex or simplex transmission: Both of the terminals have messages to transmit over the channel or one of them works merely as a receiver*.
- (c) Constant or variable block length: A set of information digits together with appropriate protocol, error detection and control information which is dealt with and transmitted together as a message unit is called a "block". These blocks all may have the same length or may be of variable length.
- (d) Continuous or stop and wait transmission:
Transmission of blocks might be done continuously, or the transmitter might stop after transmitting each block and wait until the feedback signal comes, and then transmit the next block.

The first three sets of alternatives mentioned above depend on the requirements and characteristics of the system (terminals and channel). The fourth one is a result of the strategy adopted. There are some additional options on the strategy used which will be looked at later when the strategies are being described.

* By "merely receiver" we mean a terminal which does not transmit any information except for feedback control data which must be sent by each receiver in a system with a retransmission strategy.

In the next chapter, retransmission strategies for the stop and wait transmission case (which is the easier one) are studied. Chapter III is devoted to the analysis of the problem for the continuous transmission case but is limited to the constant block length systems. Two strategies designed in the past are studied and a simplified version for one of them (Roll Back K Scheme (2)) is proposed. A method for proving the correctness of retransmission strategies is discovered. Using this method all the strategies in chapter II and III are proved to function correctly. Having analyzed and understood the simpler cases, then the difficult task of retransmission strategies in the continuous transmission, variable block length case is considered in chapter IV. A flexible strategy, applicable for all the circumstances, using , on the average, approximately 1 control bit per block is developed. The retransmission procedure in the strategy turns out to be even more effective and efficient compared with those strategies designed and implemented for the simpler problems. This suggests using the same kind of concepts and the same strategy for the more simple cases as well.

II. STOP AND WAIT STRATEGIES

2.1 Noiseless Feedback Channel

To begin with we consider a very simple and even nonrealistic problem: Simplex transmission (from terminal A to B) over a full duplex channel where the forward path (from A to B) is noisy but the feedback path is noiseless (which doesn't exist in the real world).

As a result of having a noiseless feedback channel, the transmitter (terminal A) always receives error free control signals. The operation of the system in "stop and wait" mode can then be described as follows:

Terminal A each time sends one block which contains all the necessary protocol and error detecting data. Then it stops and waits until receiving a feedback signal from B. This signal, which is error free, tells A whether to go back and retransmit the last block or to send a new one. This information (in the feedback signal) can be actually put down in a single binary digit (bit) which we will refer to as the "verify bit" throughout this thesis. When this bit requests the transmitter for retransmission, we refer to it as RQ (repeat request), otherwise as OK (acknowledgment). Therefore in this case the feedback signal can be made of a single bit.

Although this scheme is described clearly enough, we are going to use a specific kind of diagram (adopted by us) to demonstrate the performance of the system

Over an arbitrary time period and an arbitrary pattern of erasures in the channel. Since this kind of diagram will be used in this report very often we will describe it here in detail: Figure 2.1 shows this diagram for the system and scheme under consideration. It is basically two parallel lines, each one of them representing the time axis for one terminal. Each block is shown on the top of the line of the transmitter located on the time interval of transmission. Each transmission is shown by an inclined line. For example line "L" represents transmission of block A1 from A to B. The arrow shows direction of transmission. Notice that for the receiver, block A1 is unknown till it receives the last bit of A1. It is exactly for this reason that we begin line "L" at the last moment of the transmission of A1. It should also be noticed that "L" is inclined rather than vertical i.e. there is a time difference between the beginning and end of "L" due to the delay in transmission (propagational delay plus processing delay). Therefore " τ " in Fig 2.1 represents the amount of this delay. In the same way line $\bar{L}$ shows the transmission of the feedback signal which in this case can just be a single bit. The content of each control signal is written along the line representing its transmission. Each transmission which is subject to error is shown by a dashed

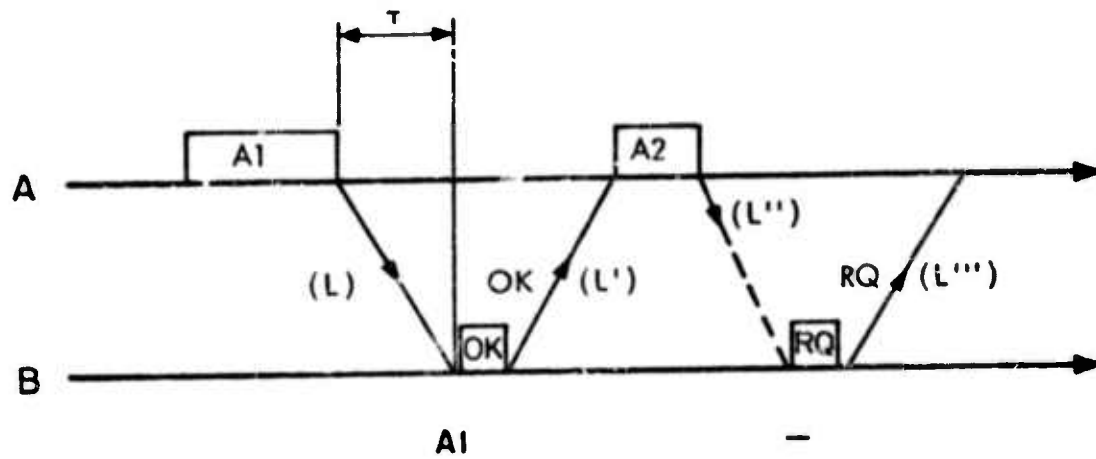


Fig . 2.1

line (L"). Whenever a terminal considers a received block as correct and prints it we write the name of that block beside the time axis of the terminal at the moment of printing; otherwise, we put a dash.

2.2 Noisy Feedback Channel

2.2.1 Introduction: Let us now investigate what kind of modification in the previous scheme needs to be done if the feedback channel is noisy. First of all since now the feedback signal is also subject to error, terminal A must have some way of determining whether the received feedback signal is erroneous or not. Therefore error detection codes must be used here as well, and the feedback signal can no longer be a single bit. It will contain at least $NC+1$ bits where NC is the number of necessary check digits.

Secondly we need to determine what decision must be made by terminal A if it turns out that the feedback signal contains error and thus does not represent the signal sent by B. Should terminal A interpret the erased feedback signal as an "OK" and therefore send the next block or should it interpret it as an "RQ" and so retransmit the previous one. Both interpretations are conceptually acceptable, since it has to assume something anyway. The only problem is that the strategy must be designed

so that if the interpretation turns out to be wrong, the mistake can be corrected. In the first case (i.e. if A considers the erased feedback as "OK" when it is really "RQ" and transmits the next block), there must be some way for A to find out about this mistake and hence go back a number of blocks to retransmit the appropriate one, and for B there must be some way of rearranging the received blocks in the correct order. In the second case (i.e. if A considers the erased feedback as "RQ" when it is really "OK" and retransmits the previous block), then only the receiver needs to be able to find out about this mistake and not print this block for the second time.

Seemingly, design of a strategy on the basis of the second interpretation is much simpler because of the fewer necessary actions in the case of a wrong interpretation. This is why all the people who have dealt with this problem have chosen to assume that each feedback signal erased in the channel is an "RQ". Therefore in this and the next chapter we will also make this assumption*.

*

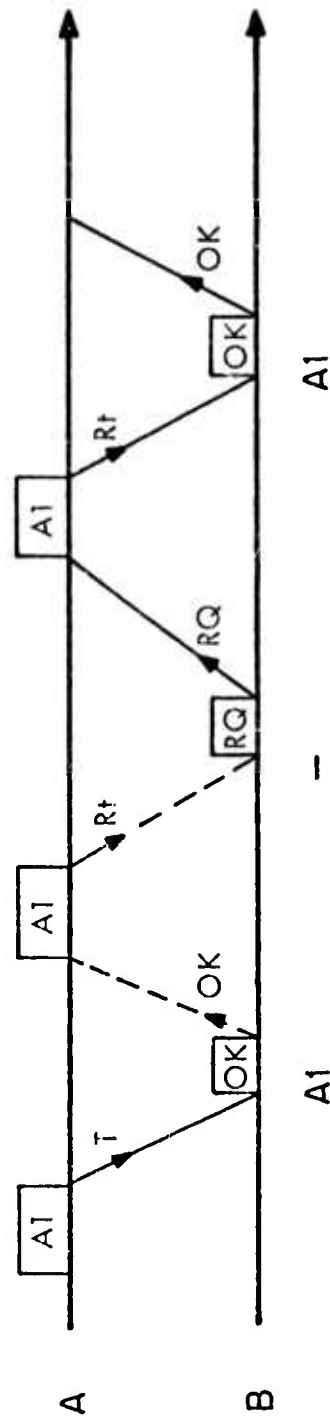
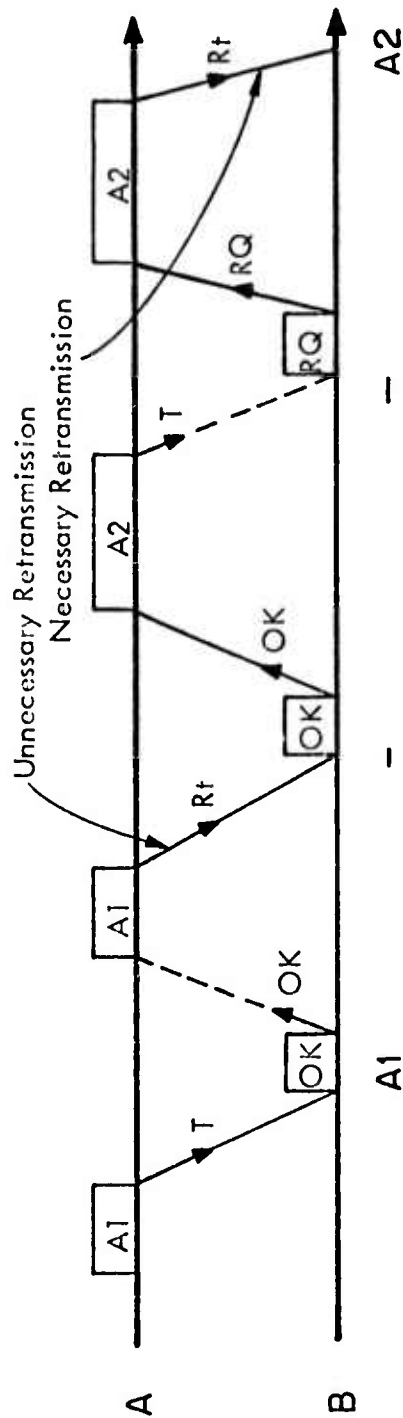
In chapter IV we will see however that this is a rather hasty conclusion. Our final design to be discussed later is based on the other interpretation, which makes it not only easier but even more efficient.

2.2.2 An Inadequate Strategy: One strategy (which is designed by a computer manufacturer as Lynch (6) mentions) suggests that terminal A (the transmitter) also reserves one control bit in each block to inform terminal B whether this block is a retransmission or not. The designer argues that if A receives an erased "OK" signal from B and therefore retransmits the previous block, then terminal B by observing the control bit in the received block concludes that it is a retransmission due to an erasure in the feedback channel and therefore does not print it for a second time (Fig. 2.2.a). It is very easy however to show that the argument is not complete and the strategy fails to operate successfully under some other erasure patterns like the one demonstrated in Fig 2.2.b. The argument is only concerned with a single erasure in the feedback channel, in which case the strategy works. In Fig 2.2.b we have two successive erasures where the result is a double print.

This example serves here to warn us that unless a strategy for retransmission is carefully checked out against all possible erasure patterns one can not consider it as a perfect one.

2.2.3 Lynch's Design for the Noisy Feedback Channel:

Lynch (6) has described another scheme for the foregoing



T : Transmission

Rt: Retransmission

Fig. 2.2

problem which works perfectly well and can overcome any combination of erasures in the channel. It is actually a modified version of the one described in 2.2.2. The control bit sent by terminal A with each block is used in a slightly different form. Instead of having static control symbols for retransmission and transmission, the control bit will be changed for each new transmission and will stay unchanged for retransmission. This kind of control bit will be referred to as "alternating bit" from now on. The receiver then accepts an error free received block as a new one and prints it if its alternating bit is different from the alternating bit in the most recent accepted block. A proof for complete operation of this scheme is given in 3.6.1 after we introduce the state representation of retransmission systems. We only present here a random incident of the channel as an example to clarify what was said before (Fig 2.3).

2.2.4 Ready and Acknowledge with Transition Signalling:

Bartlett (7) suggests that the feedback signal also instead of being a verify bit, can function as an alternating bit exactly in the same way that the alternating bit sent by A operates: It changes whenever B receives a new error free block and stays unchanged otherwise, i.e. whenever B receives an erroneous block or a previously received block , then terminal A

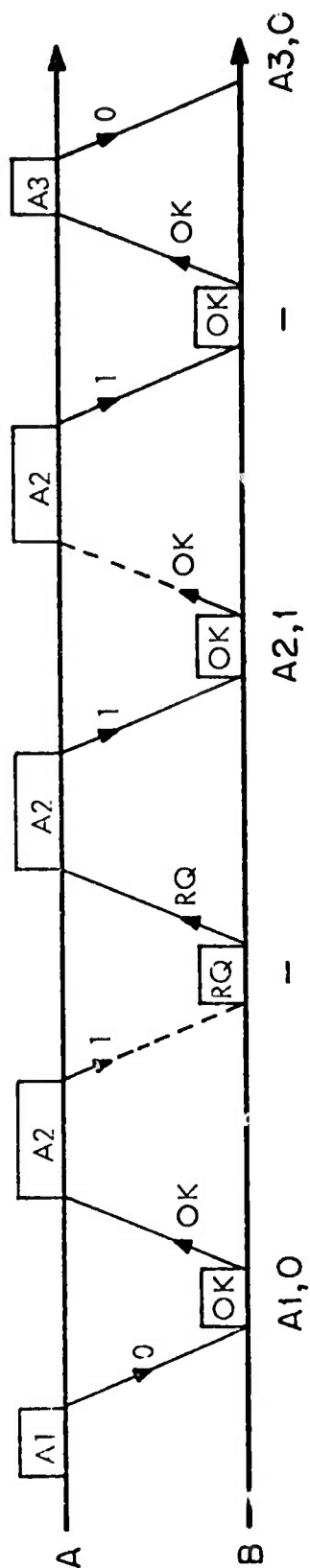


Fig. 2.3

sends a new block whenever it receives a new control signal and retransmits otherwise (Fig.2.4) this suggestion is actually made when discussing about a full-duplex transmission system to be dealt with later. Again a proof for this scheme together with a comparison between it and Lynch's scheme are presented in 3.6.2 and 3.6.3

2.3 Full Duplex Transmission

2.3.1 Decomposition Approach: This is now the time to take one more step forward and consider the full-duplex transmission case. Other assumptions made in Sec. 2.2 are kept the same, i.e. the channel of communication is full duplex and we consider stop and wait transmission only.

The first thing to mention about a full-duplex transmission system is that since now each terminal has its own blocks to transmit, there is no need for constructing special blocks to serve as feedback signals (for error control). In other words each terminal can send necessary control information with the blocks that it transmits. If it does so then the error detecting part of each block will cover the control bits as well and extra error detecting information is not necessary. (Figure 2.5)

The simplest way of approaching a solution for full duplex transmission system is to consider it as the composition of two simplex transmission systems,

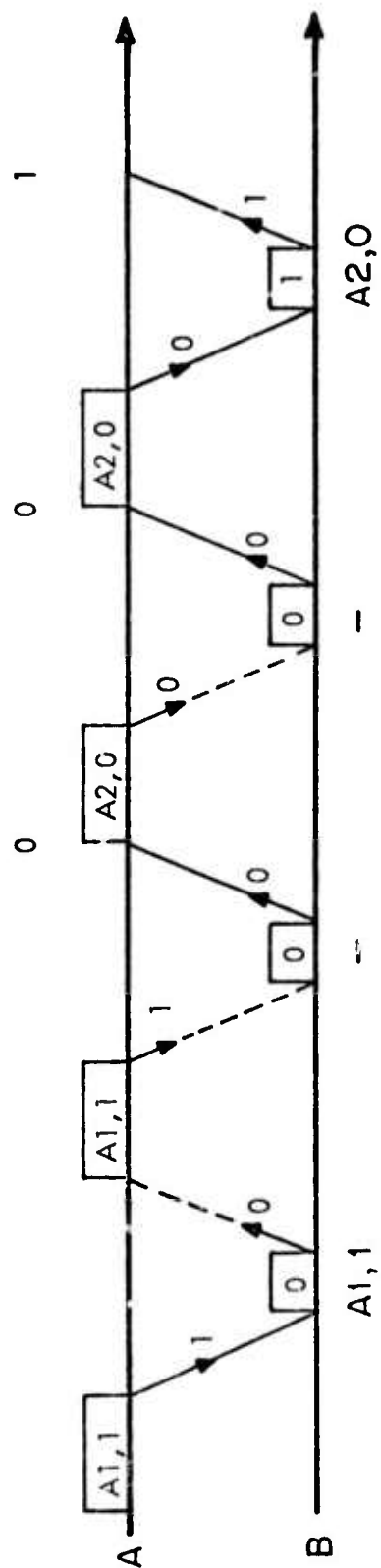


Fig. 2.4

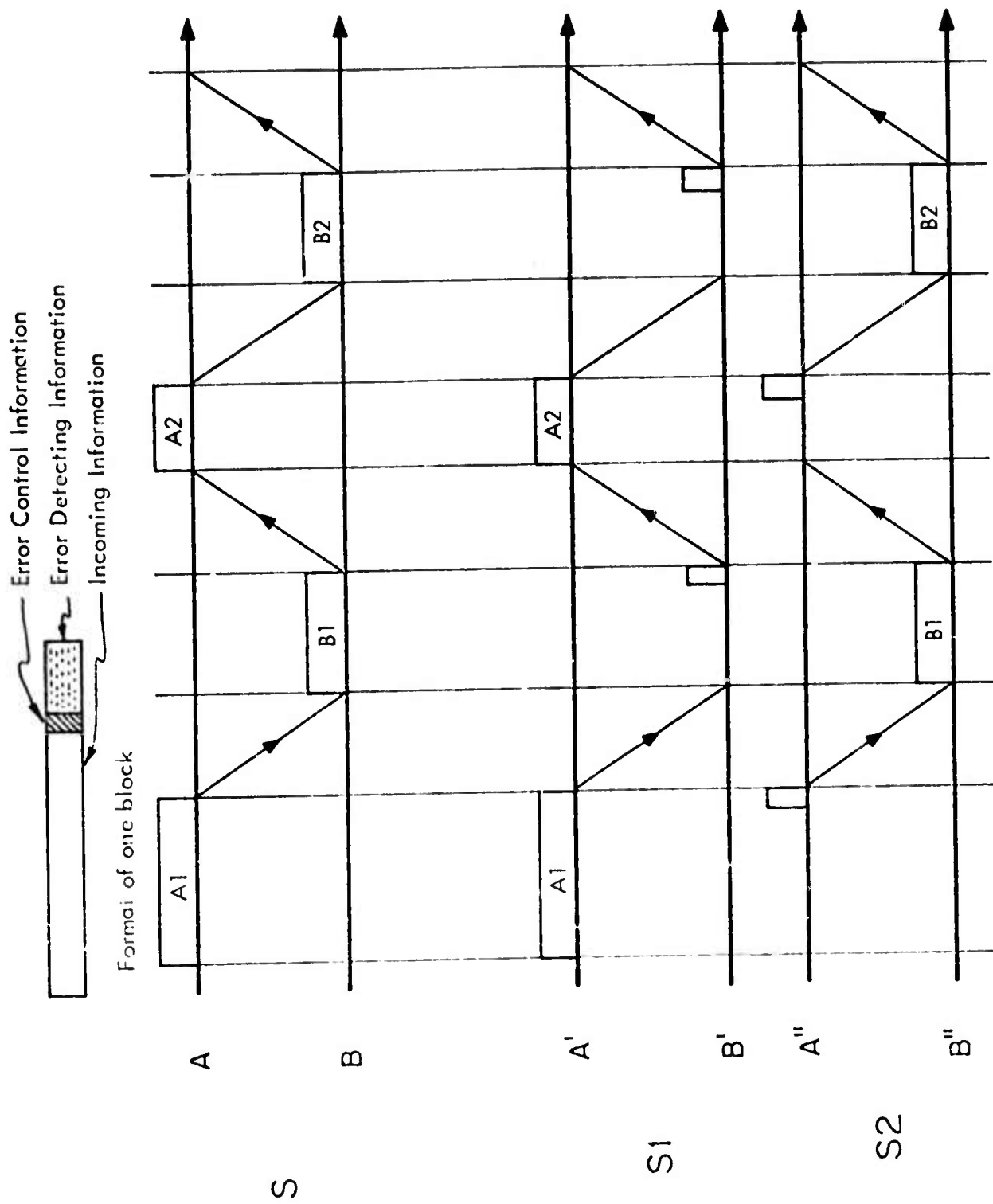


Fig. 2.5

each one using a full duplex channel. Fig 2.5 shows an example where the full duplex system S is decomposed into S1 and S2. If S1 and S2 operate correctly using some retransmission strategy, (for the simplex case), system S also operates satisfactorily. The strategy used by S in fact is a combination of two simplex strategies: Terminal A in S takes both the actions that A' does as a transmitter and A'' does as a receiver. It sends with each block the control data sent by A' (dt) and also the control data sent by A'' (dr). Terminal B also does the same thing. The receiver of a block then uses dt to decide whether it should print the block or not. It will use dr to decide about the next block it sends itself.

2.3.2 Lynch's Scheme and Bartlett's Scheme for Full

Duplex Transmission: It should be obvious from the foregoing discussion that the decomposed components of the full duplex system can use whatever scheme is adequate for simplex transmission. For example, they can adopt the scheme described in 2.2.3 or the one in 2.2.4. using the first one results in Lynch's strategy for full duplex transmission (6).

Fig. 2.6.a shows the full duplex transmission using Lynch's scheme. Here each terminal uses two control bits, one as dr (verify bit) and the other as dt (alternating bit)*.

Fig 2.6.b shows Bartlett's scheme where each one of the

* Turn to the next page

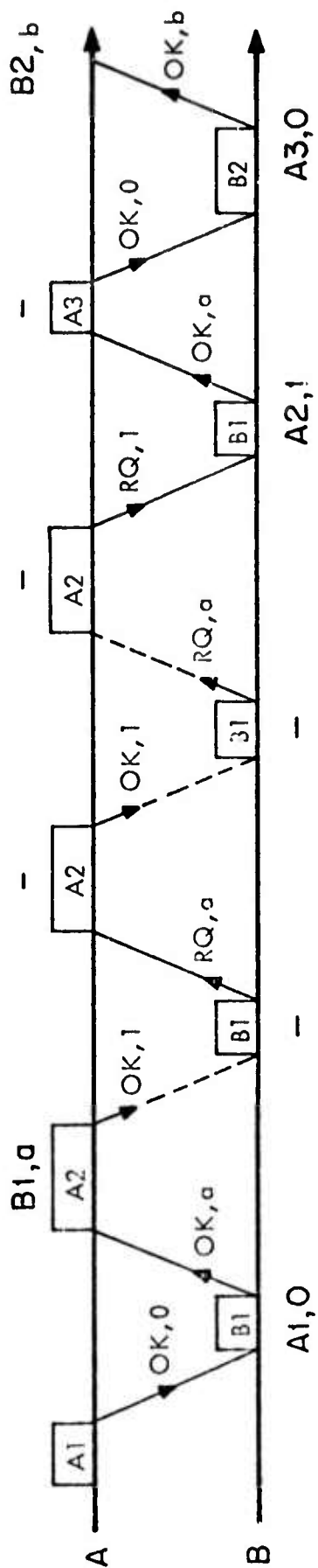
decomposed components of the system uses the scheme described in 2.2.4. Notice that in this case dr and dt conceptually are the same, i.e. both of them are a single alternating bit with similar functions. Therefore in Bartlett's scheme each terminal needs only to send a single bit as control data and is preferable over Lynch's scheme from this point. We will show in 3.6.3 that despite what Bartlett thinks his scheme works more efficiently for some particular erasure patterns and equally efficient for the others compared with Lynch's scheme. This becomes clear in Fig. 2.6 by comparing 2.6.a and 2.6.b which have similar erasure patterns. Notice that in Fig. 2.6 instead of 0 and 1, a and b are used as the values of a the alternating bit sent by B to prevent confusion.

2.4 Half-Duplex Channel:

In the foregoing sections we discussed several stop and wait strategies in full-duplex channel. Here we claim that these schemes can be used without any modification for variable block length as well as constant

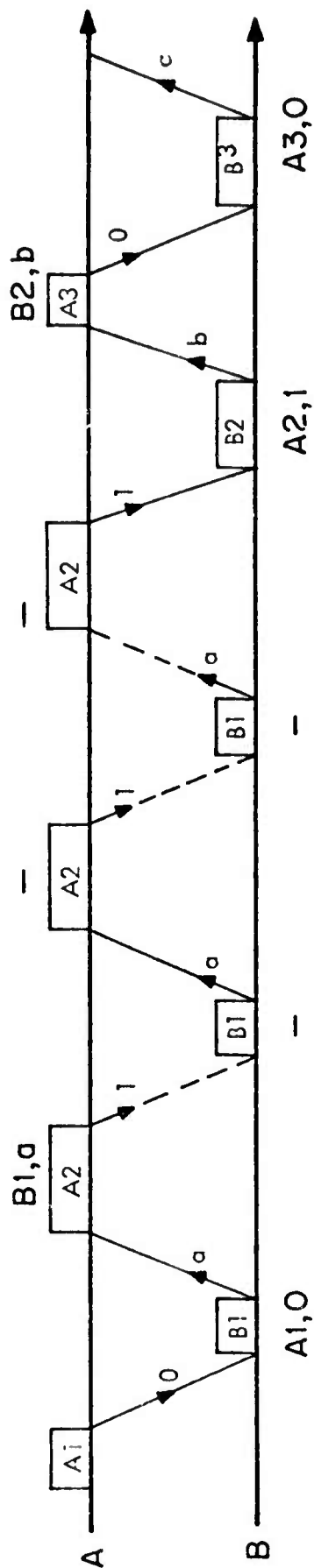
Refer to Previous Page*It should be mentioned however that by using Huffman codes we can decrease the average length of control data for Lynch's full duplex scheme to 1.5 bits/block. The Huffman code is as follows:

0	verify bit = OK	alternating bit = 0
10	verify bit = OK	alternating bit = 1
110	verify bit = RQ	alternating bit = 0
111	verify bit = RQ	alternating bit = 1



2.6.a

31



2.6.b

Fig. 2.6

block length and also for half duplex channels as well as full duplex ones! Figures 2.1 through 2.6 show these two facts clearly: First of all, they are drawn for the variable block length case and if the length of some blocks change, the nature of the problem doesn't change. The reason is that each terminal waits and doesn't transmit until the other one ends its transmission:hence, the period of waiting has no effect on the procedure.

A second fact about Fig 2.1 through 2.6 is that at each moment of time,at most, one terminal uses the channel for transmission. It never happens that both of the terminals use the channel simultaneously. This is why we don't necessarily need to use a full duplex channel. A half duplex one can serve in the system as well by reversing directions whenever necessary. Of course it introduces an extra delay at the moments of reversal, which is significant in many cases. This is why in practice, it is preferable to use a full duplex channel even for stop and wait strategies.

Having discussed these two points we are done with stop and wait strategies for all possible circumstances. In the next chapter we investigate the continuous transmission case.

III. CONTINUOUS TRANSMISSION STRATEGY FOR CONSTANT BLOCK LENGTH

3.1 Introduction

In a continuous retransmission system, the transmitter sends blocks continuously without waiting for control signals (Fig.3.1). For example, if block A2 sent by terminal A gets erased during transmission, the repeat request (RQ) signal for A2 will be received by A after it has transmitted some other blocks (in this case A3, A4, A5 and A6). Obviously then terminal A has to go back a number of blocks (here 5 blocks) and retransmit A2. After doing so it may also retransmit A3, A4, A5 and A6 to keep the correct order of transmission, or it may return to where it was before and begin its job from A7 to avoid unnecessary repeated transmissions. Whether terminal A adopts the first method or the second one depends upon the strategy implemented and the ability to detect end of block etc. We refer to the first method as "Ordered Retransmission" and to the second one as "Selective Retransmission".

In any case it is easy to conclude that for continuous retransmission strategies the nature of the problem for the constant block length case is very different from the variable block length case. This should become quite clear through this chapter and the next one.

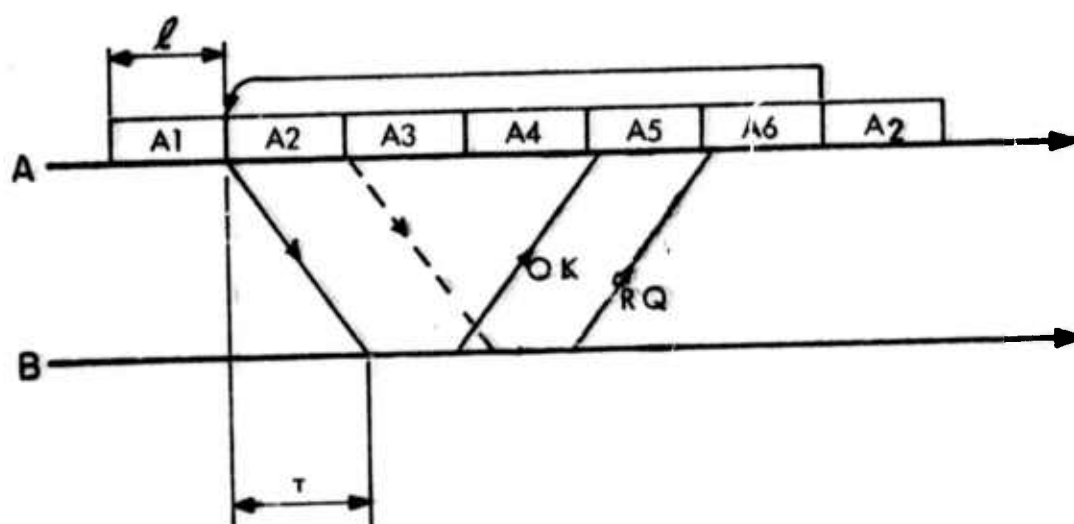


Fig. 3.1

This is why we have distinguished these two cases from each other and deal only with the first one in this chapter. The case of variable block length which is the main objective of this research is left for chapter IV.

Furthermore the channel of communication has to be full duplex for continuous transmission systems, because if one terminal sends information through a half-duplex channel continuously, then there remains no way for the other terminal to transmit anything.

To the best of our knowledge in this case two different approaches have been made previously for the design of retransmission schemes. One of them results in a "Selective Retransmission" scheme and the other one gives an "Ordered Retransmission" one. In this chapter we will discuss both of them, while in chapter IV we will introduce a third kind of approach.

3.2 Selective Retransmission Scheme

3.2.1 Decomposition Approach: The first approach is decomposition of the system into a number of systems operating with a stop and wait scheme. Apparently this approach was made first by Metzener and Morgan (8) while Nourani (9) describes it in more detail. The illustration made in Fig. 3.2 is probably the best way of explaining this idea. Here a full duplex continuous

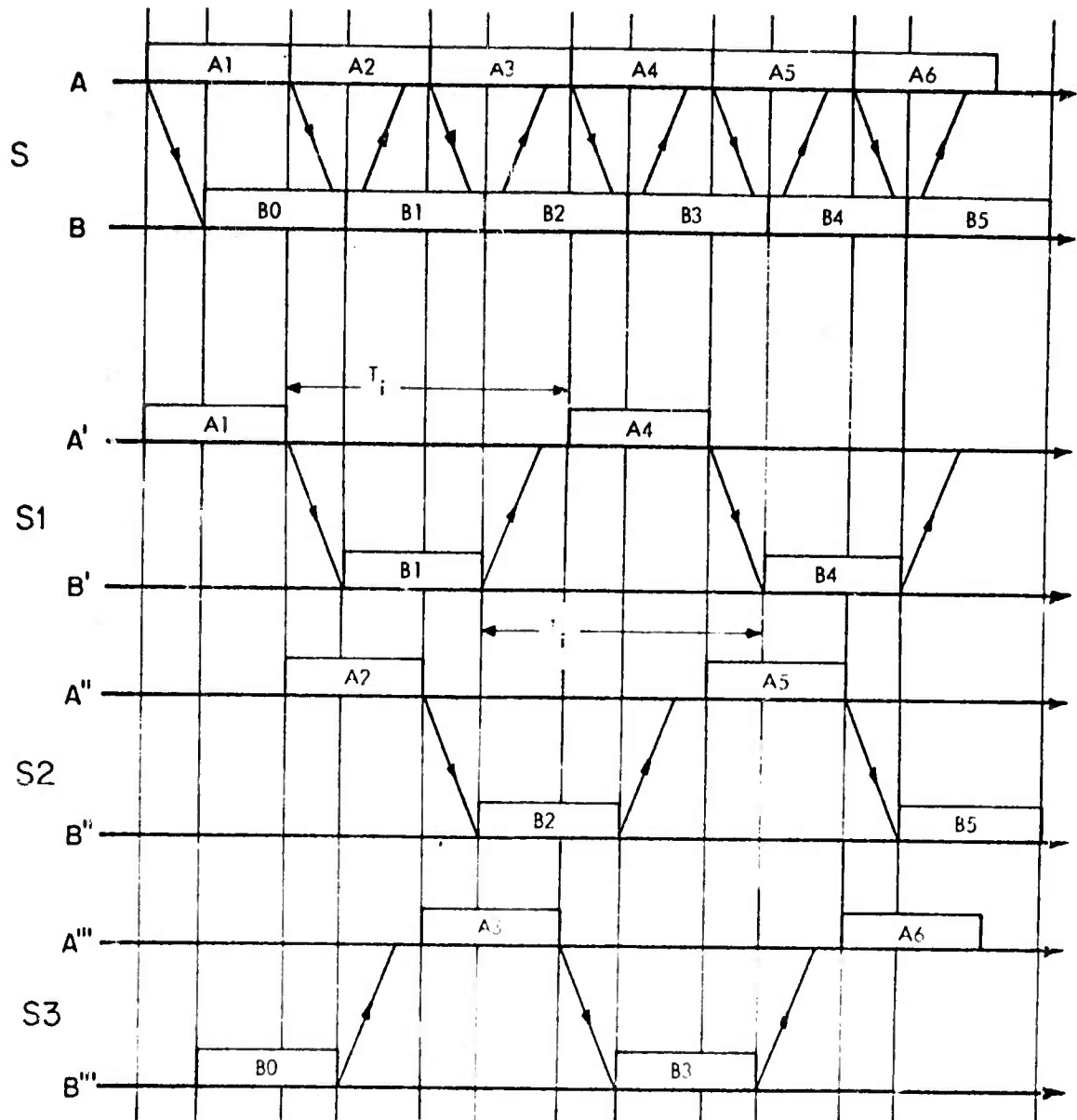


Fig. 3.2

transmission system with constant block length is treated as three simplex system merged together, where each one of these systems operates in Stop and Wait mode.

To understand Fig. 3.2 better, we can suppose that terminal B separates its stream of blocks $(B_1, B_2, \dots, B_n)$ into three groups: $(B_1, B_4, \dots, B_{l+3i})$, $(B_2, B_5, \dots, B_{l+3i+1})$ and $(B_3, B_6, \dots, B_{l+3i+2})$. If terminal A also does a similar separation, then they can exchange their first groups of blocks on the basis of a stop and wait strategy. The idle period of each terminal $-T_i-$ therefore will be at least $l + 2\tau$. If $l + 2\tau$ is less than $2l$ i.e. if 2τ is less than l then T_i can be fixed at $2l$. Now as Fig 3.2 shows they can use this idle interval T_i to communicate the other two groups of blocks, (once again with the stop and wait strategy). Notice that in general we can use the same kind of trick when the round trip delay $-2\tau-$ is larger than l by dividing up the stream of blocks into n groups, where n is the smallest integer larger than $2 + \frac{2\tau}{l}$. Also notice that there is not restriction on the stop and wait scheme used for communication of each group. For example; Lynch's or Bartlett's scheme described in 2.3.2 could also be used. Obviously in the second case the control information is a single bit for each block, while the first case requires two control

bits. As we have pointed out earlier, using Bartlett's scheme is preferable to Lynch's not only because it uses fewer control bits, but also because it overcomes the erasure patterns faster.

Another thing to mention here is that this scheme is a "selective retransmission strategy" since groups are transmitted independantly, and the transmitter retransmitts only the errased block, not those which appear after it.

3.2.2 Buffer Limitation in Decomposition Approach:

There is an important problem associated with the above scheme: Since A and B communicate each group of blocks independently, if more erasures occur on the channel for one group than for the other, the transmission of the first group proceeds more slowly than the second one. Therefore although the order of blocks within each group will be the same in transmitter and receiver, the over all stream of blocks is received in a different order than that of presentation to the transmitter. The receiver however is potentially able to rearrange the blocks in the proper way because, considering the receiving interval of different groups, it can first separate these groups from each other and then combine them together by periodically choosing one

block from each group. The receiver then needs to store each incoming block in a buffer until the time it can pick it up. Since in practice we have limited size buffers, there is some possibility that the buffer gets saturated by received blocks from some groups, where, from one of the groups nothing has arrived due to the consequent erasures in that group. Although it can be proved that by increasing the size of buffer, probability of such an event can be made arbitrarily small, still it doesn't vanish and will occur in the long run of the system. One solution to the problem then is to send some artificial RQ's in some groups to make an artificial balance in the number of erasures in different groups.

3.3 Roll Back K Sche. *

3.3.1 Introduction: This scheme is neither a "selective retransmission scheme" nor is it based on decomposing the system into some simpler ones. The whole stream of blocks is treated together and the transmitter in the case of erasure, not only resends the erased block, but also resends those which appear right after that (Fig 3.1). Therefore it is an "ordered" retransmission strategy.

* Apparently the basic idea of this scheme was first suggested by Wozencraft (3). Schmitt et al then developed it for implementation (2). Readers are advised to refer to (2) for a detailed discussion about the system design. We only emphasize here, those aspects of design which are of interest for our purpose.

The scheme suggests a single verify bit for error control. As it was pointed out in chapter II, each terminal upon receiving an erased block can not determine it has actually contained an RQ or an OK and it has to assume something. This scheme again, like those discussed in chapter II, takes each erased block as one with RQ.

Fig. 3.3 shows that the number K in a roll back K strategy depends on the ratio of τ -delay of line-and l -length of block. Another factor which affects K is the time shift between the start of transmission of the two terminals and the location of the control bit in each block. It can be shown easily that the smallest value for K is obtainable when: i) the control bit is located just before the check digits which are at the end of a block and ii) the time shift between terminals is such that one of them (either A or B) ends receiving a block when it inserts the control bit of its own block. The situation shown in Fig 3.3 fulfills both of these conditions. Under these conditions K is the smallest integer greater than $1 + \frac{tc+2\tau}{l}$, where tc is the length of check digits in a block. In the following discussion, for simplicity, we assume that $\frac{tc+2\tau}{l} < 1$ so that K turns to be 2. This happens when $2\tau < l - tc$ i.e. when the round trip delay of the line (2τ) is less

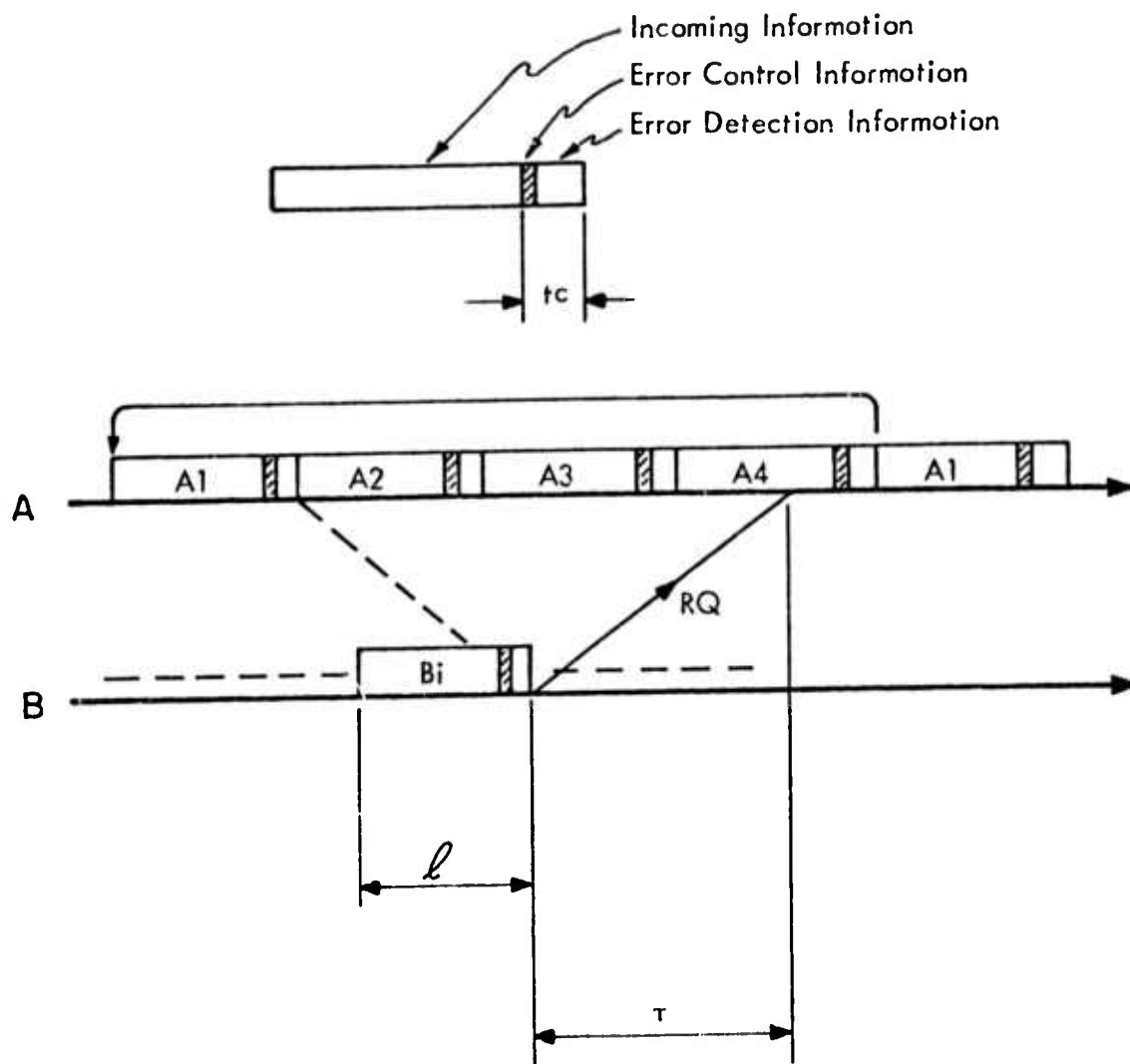


Fig. 3.3

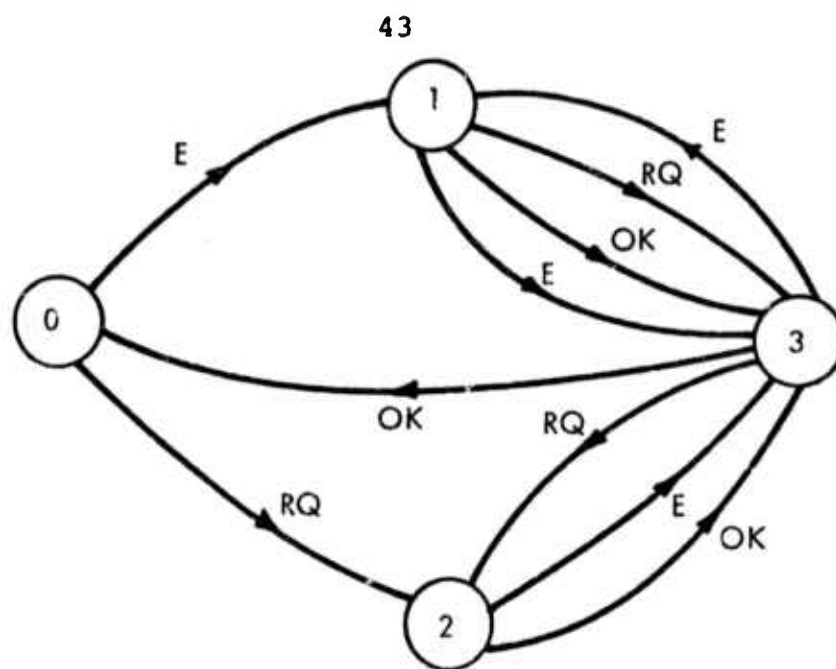
than the length of the information field of each block
($l - t_c$)*.

3.3.2 Roll Back 2 Scheme:

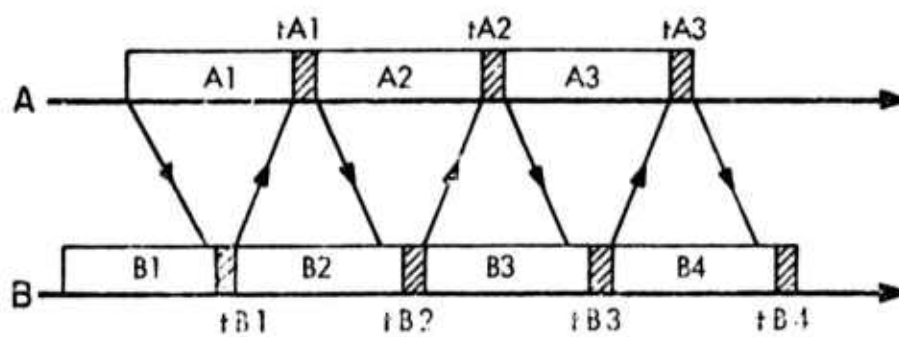
In this method there are four possible states for each terminal which determine different actions that a terminal needs to take. Fig 3.4.a shows how the state of a terminal changes in different situations. As far as error control is concerned, each received block can be one of the following three cases: It is either error free with the verify bit being acknowledgement (OK) or error free containing a repeat request (RQ) or it is erroneous block (E). We assume that each terminal changes its state immediately after reception and detection of a new block. Accordingly in Fig. 3.4.b, $tA_1, tA_2, tA_3, \dots$ are the moments of state transition for A and $tB_1, tB_2, tB_3, \dots$ are such moments for B. We also refer to all decisions and actions made by a terminal at these moments. For example at tA_1 terminal A after determining its new state decides about the control bit to insert in block A1, about accepting (printing) the block which is just received (B1), and about the next message to start sending (P).

State 0 is the desired state for a terminal

* Control bits are considered here to be among check digits. Also the detecting time for a block in the receiver must be considered as a part of delay.



3.4.a



3.4.b

Fig. 3.4

(when there is no erasure for sometime). A terminal at state 0 accepts the received block, inserts OK and begins transmitting a new block. At state 1 or 2 the received block isn't accepted and the terminal goes back 2 and begins to retransmit from that point. If the state is 1 it inserts RQ, if the state is 2 it inserts OK. At state 3 a terminal doesn't accept the received block, inserts an OK and keeps on transmitting without going back. The reason being that state 3 always appears right after state 1 or 2 (Fig.3.4.a). Therefore a terminal in state 3 has already gone back for retransmission.

3.3.3 Huristic Arguments for Roll Back 2 Scheme:

Although we are finished with describing the roll back 2 scheme it isn't yet clear why it works! Nevertheless there are some dark or even strange points in the design which are seemingly unreasonable to the reader. One of these points which can be answered right away is why a terminal doesn't print the received block at state 2 (when it contains an RQ but is error free)! The answer is because, as we said earlier, an erased block is interpreted to contain an RQ. Therefore if a terminal sends RQ because it has received an erasure, it means that it is retransmitting (like state 1). The other terminal, however, if it hadn't actually sent RQ previously shouldn't print a block for the second time.

There are other questions which can not be answered this way. For example why states 1 and 2 are to transit to the same state (3). Is it a correct simplification? Why does state 3 change back to 1 or 2 when a new E or RQ comes in? Unfortunately the authors (2) haven't presented any argument concerning the correctness of their design nor have they described their way of tackling this problem and their approach to this simple and nice looking design. They have only chosen a number of error patterns to show that this scheme can overcome them successfully. But this wasn't enough for us because we thought that understanding the central idea behind this design would help us to develop a similar scheme for the variable block length problem. This expectation turned out to be wrong but it caused us to develop a type of state diagram which can rigorously prove the correctness of this design. After that, we even could simplify the foregoing design to some easier form. To the best of our knowledge no real proof has been previously given*.

We will present our proof in the next section but

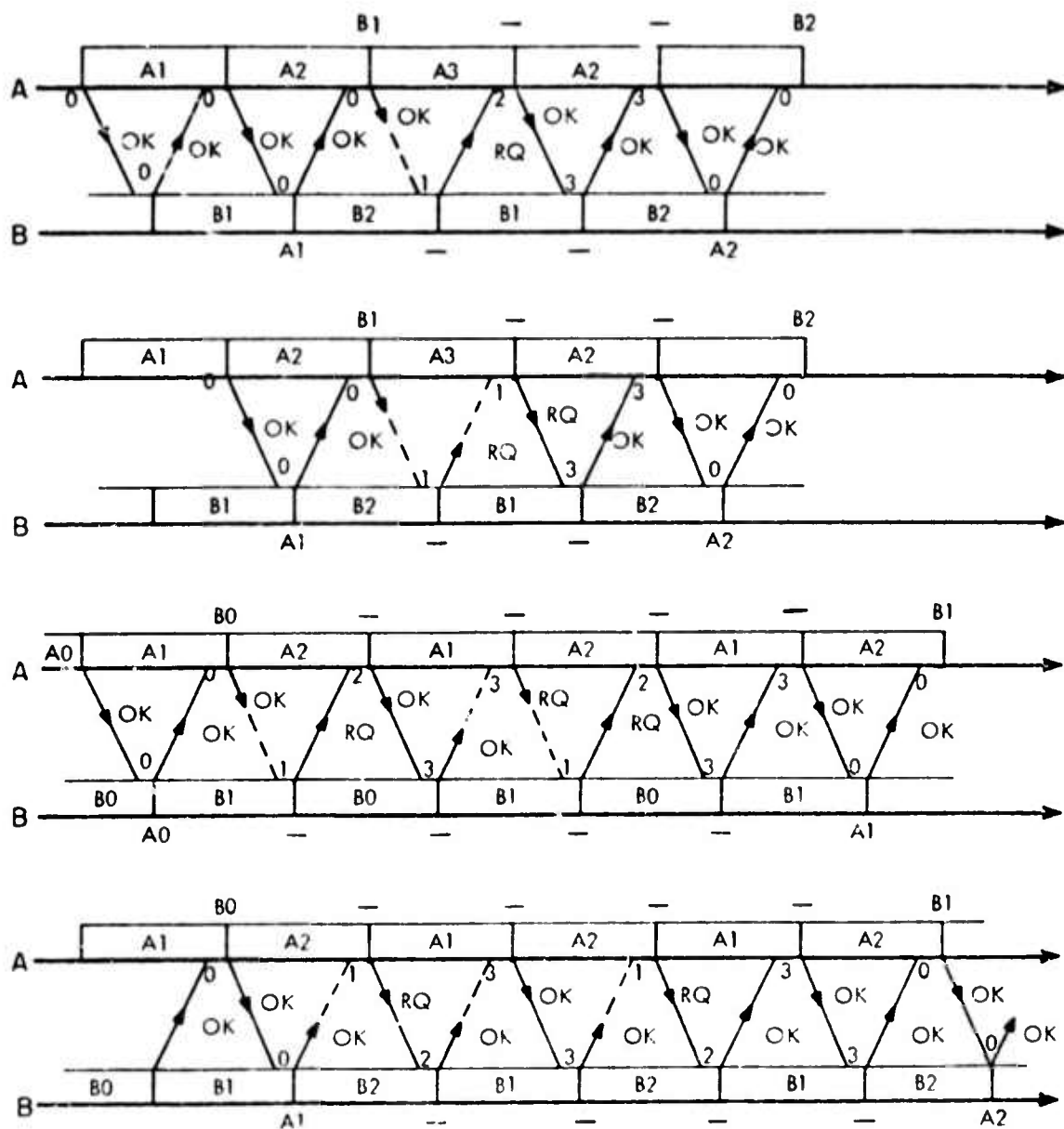
* There are however some heuristic arguments concerning it. Fray (10) for example tries to justify and analyse the scheme by considering lengthy sequences of possible events in the system.

before that it is instructive to see how the scheme treats different error patterns in the channel. Fig 3.5 demonstrates several examples of the system performance and is very useful for such a purpose.

3.4 Justification for Roll Back 2 Scheme

3.4.1 State Representation: Since there are an infinite number of erasure patterns which can happen in a channel (as the channel usage period tends to 0), we can not argue about the successfulness of the scheme for all these cases separately. Our approach is then to consider a limited number of system's situations which are enough to represent all the possibilities occurring in the system. We refer to these situations as the states of the system. Notice however that the state of the system is different from the state of a single terminal which was described earlier.

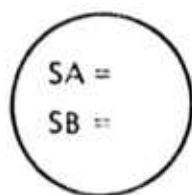
Since the system is composed of two terminals and a channel we can imagine that SA and SB, the states of terminal A and B, when put together make up at least some part of the system's state i.e. $S = (SA, SB, X)$. We try now to explore the unknown element X: since SA changes at the moments when A receives some block i.e. at $tA_1, tA_2, tA_3, \dots$ (Fig 3.4.b) and SB changes at $tB_1, tB_2, tB_3, \dots$ we can conclude that the system's state transitions happen at all of these moments i.e. at $tB_1, tA_1, tB_2, tA_2, \dots$



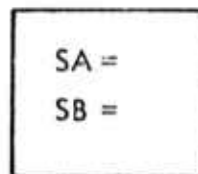
Hint: The state of each terminal is shown beside its time axis at the moment of state transition.

Fig. 3.5

(Fig.3.4.b). We refer to these moments as t_i where $t_1 = t_{B1}$, $t_2 = t_{A1}$, $t_3 = t_{B2}$, $t_4 = t_{A2}$, ... Next let us ask what we expect from the state of the system! We would like the state at t_{i+1} (S_{ti+1}) to be only a function of the state at t_i (S_{ti}) and the occurrence in the channel, which is either erased or error free transmission. Notice now that if, for example, $S_{Ati} = S_{Bti} = 0$ and the next occurrence in the channel is an erasure, we can not say whether S_{Ati} changes to 1 or S_{Bti} , unless we know whether t_{i+1} is a transition moment for A or B. This is the third element (X) of the state S. In other words, X alternates periodically between two states XA and XB and says whether the last transition was at terminal A or at B. Fig. 3.6 shows how we represent these 3 elements of the state. Probably this is the most expressive way of saying $X = XA$ or $X = XB$. This completes our discussion of drawing and determining the state diagram of the system. However we have not discussed our major goal in this section i.e. proving or disproving that a strategy works correctly. A strategy is correct if each block is printed once, correctly and in order. In order to be able to check for this we need to add something more to the state diagram. We need to demonstrate there, which blocks are sent at each transmission and which blocks are printed



$$x = x_A$$



$$x = x_B$$

Fig. 3.6

at the receivers.

Fig. 3.7 shows the state diagram for the roll back 2 scheme. The two parallel lines representing transmissions in the system are drawn this time downward on the left. Transition moments t_i are indicated behind them. In front of each transition moment the new states of the system are shown by the two symbols  or . Beside these symbols at their right the name of the printed block is written. If there is no block accepted and printed a dash is put down. Those branches which represent an erroneous transmission are drawn from a state to the the left with dashed lines, while others showing an error free transmission are drawn to the right with solid lines. Beside these lines at the right side, the name of the block which is being transmitted and the content of the control bit (OK or RQ) is written. The diagram is started from a normal state $SA = SB = 0$ and is contained with two error free transmissions in order to have a relaxed background for demonstrating the disturbances of an erroneous channel.

Notice that beside each state symbol at the left side the name of a block is written and underlined. For example beside  att_0 we have A2. This means that terminal A has decided to send A2 in its next transmission,

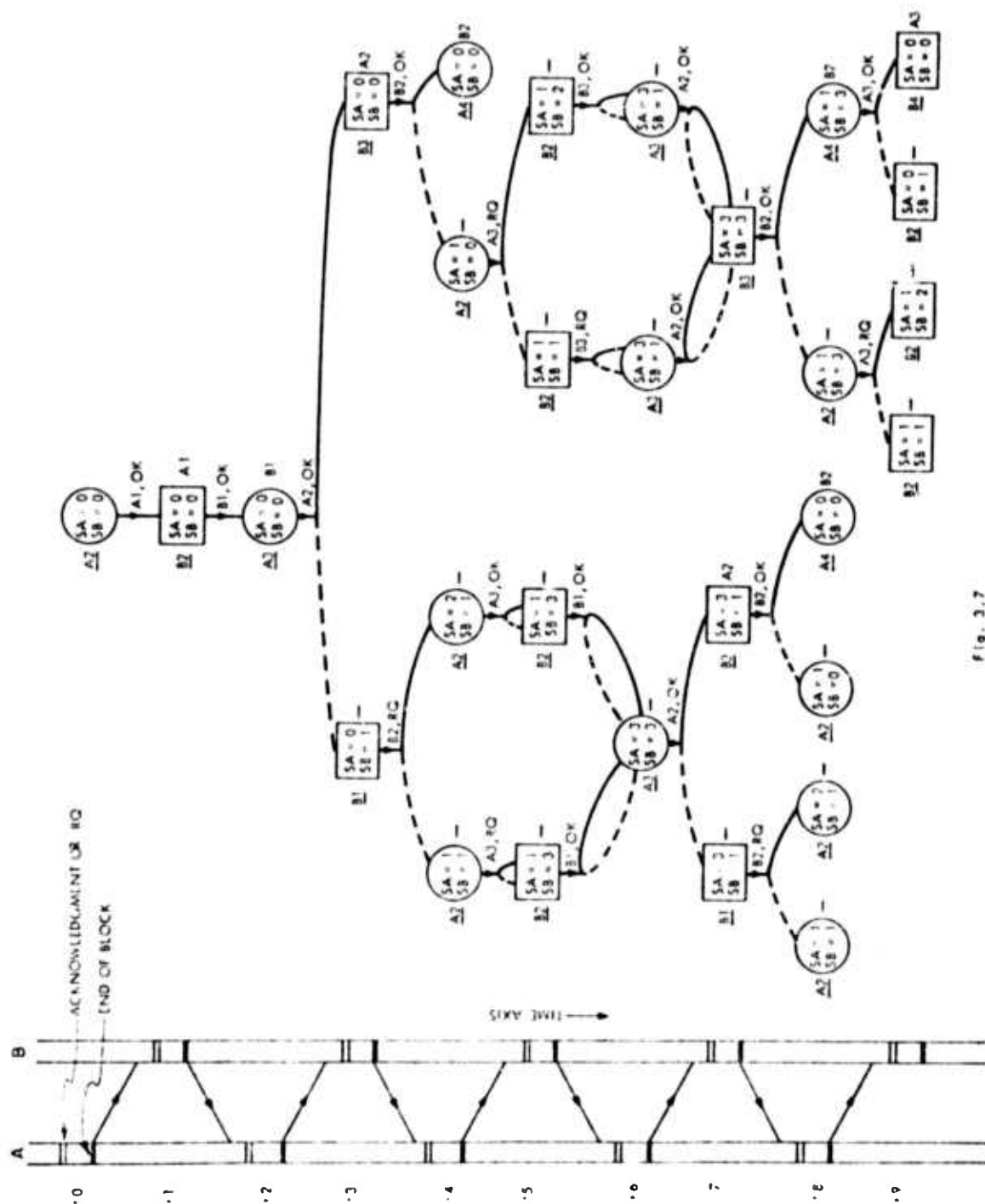


Fig. 3.7

while A1 is being transmitted right now. Notice that the branch going out of this symbol shows transmission of A1 not A2. The decision for sending A2 isn't carried out until two time slots later at t_2 . The reason for this delay in acting upon decisions is made clear in Fig 3.4.b where, for example, terminal A at t_{A1} inserts a verify bit for the just received block (B1) inside block A1. The transmission of A1 is finished after t_{A1} and then A can start transmitting A2, which is represented by a line after t_{A1} and t_{B2} , i.e. two time slots later at t_{A2} .

Let us see now what happens when a terminal receives an E. One example is state $\begin{matrix} SA=0 \\ SB=1 \end{matrix}$ at t_3 . Previously terminal B at t_1 had sent B1 and decided to send B2. If there was no E received it would decide to send B3 at t_5 . If an E is received, however, terminal B goes back 2 and decides to send B1. Again notice that at t_3 the previous decision for sending B2 is being carried out and B1 will be sent at t_5 . This is why we see B1 written beside and at the left of the symbol $\begin{matrix} SA=0 \\ SB=1 \end{matrix}$ at t_3 meaning that B1 will be sent at t_5 and B_2 will be retransmitted after that.

Whenever a state is repeated in the diagram no branches are required coming from that state. For example, at t_8 we see four states $\begin{pmatrix} SA=1 \\ SB=1 \end{pmatrix}, \begin{pmatrix} SA=2 \\ SB=1 \end{pmatrix}, \begin{pmatrix} SA=1 \\ SB=0 \end{pmatrix}, \begin{pmatrix} SA=0 \\ SB=0 \end{pmatrix}$ which

are repeated at higher levels of the diagram (respectively at t_4 , t_4 , t_4 and t_2). Therefore these four pathes are stopped at t_8 .

3.4.2 Justification: In order to justify the foregoing scheme we should go back to Fig 3.7 and look at all the directed pathes in the diagram to make sure no improper printing has occurred in terminal A or B. This is because each directed path of the diagram shows a possible sequence of events (transmissions, errasures and printings) in the system. Notice however that it is enough just to check for that part of a directed path which is shown in the diagram but we should rather more carefully investigate to see what happens when we go from the end of a branch to an equal state at a higher level.

As an example consider the path starting from the top of the diagram and going to the state $\begin{pmatrix} SA=0 \\ SB=0 \end{pmatrix}$ at level t_8 and then turning back to the same state at level t_2 . The sequence of printed blocks in the first part of this path (until level t_8) are $A1, B1, -, -, -, A2, B2$. The strategy is correct if after t_8 terminal A prints $A3, A4, \dots$ and terminal B prints $B2, B4, \dots$. We might however make a hasty judgement from Fig 3.7 that since after t_8 we go back to the equivalent state at t_2 , the next sequence of printed blocks will be $A2, A3, \dots$ for A and $B2, B3, \dots$ for B. hence the strategy is not correct. The judgement

is not correct because although the states $\begin{pmatrix} SA=0 \\ SB=0 \end{pmatrix}$ at t_8 and t_2 are equivalent, the content of buffers and the blocks in transmission are not the same. At t_8 , A_3 is the current block sent by A and B_2 is the last sent by B, while at t_2 , A_2 and B_1 are the current one and the last one sent by A and B. Considering this difference we see that after going to the equivalent state at t_2 , the sequence of printed blocks is in accordance with those blocks printed earlier. Going through the same kind of argument for other directed paths, one can prove that the strategy works correctly.

3.5 Retransmission Strategy with No Control Bit

At this point we want to introduce a rather strange retransmission scheme for the problem under consideration in this chapter(i.e. constant block length, continuous transmission case) which uses no portion of a block specifically for sending error control data. This scheme is actually a simplified version of the Roll Back 2 (or K) scheme. We deduced the possibility for this simplification only after finding the state representation of the Roll Back 2 scheme.

Careful investigation of the state diagram in Fig. 3.7 tells us that there is no difference basically between an RQ received by a terminal and an E (errasure).

This fact becomes quite clear if one observes that as long as we are only concerned with the output of the system (i.e. the result of communication) which is nothing but the sequence of printed blocks at A and B, and not interested in the state variations of the system (which is just a matter of modeling). It makes no difference if a block containing RQ gets erased in the process of transmission. For example consider state $\begin{pmatrix} SA=1 \\ SB=1 \end{pmatrix}$ at t_4 which sends an RQ but results the same state and no printing at t_5 both if transmission be erroneous or error free. The same thing is true about $\begin{pmatrix} SA=1 \\ SB=1 \end{pmatrix}$ at t_5 . Or consider $\begin{pmatrix} SA=0 \\ SB=1 \end{pmatrix}$ at t_3 , which sends an RQ as the control bit. The sequence of states is different if A receives this RQ or an erasure but still the system reaches to the same final state three time slots later at t_6 and there is no printed block in this interval for both cases.

At this point one comes naturally to the question of why bother at all with sending RQ's. Can not a terminal just send a previously erased block whenever it wants to request retransmission instead of wasting one bit for error control? The foregoing argument is conceptually enough to prove this conclusion, but still it may seem awkward and even impossible if the Roll Back 2 scheme is not well understood yet. For example one may ask why should we waste the whole of a block by intentionally

erasing it, instead of inserting a single RQbit. This will be answered if one notices that in the Roll Back 2 scheme a block will not be accepted and printed either if it is erased or if it contains an RQ. Therefore, it makes no difference if we send an erased block or one with RQ. Furthermore, notice that the decision for sending an RQ or OK is made at a moment when the information part of a block is already being transmitted (Fig 2.4.b) and only check digits are remained for transmission.

Another point is that although we convey the error control information without assigning any specific bit (or bits) to it, we are using the check digits field indirectly for this purpose.

It must be noticed also that when a terminal sends a pre-erased block (as RQ), there are some error patterns which can correct the block if they get into it. By making a suitable change in the check digits field of a block (when we want to send RQ) the probability of such undesired events can be decreased to something comparable with the probability of having undetected erasure which is negligible and assumed to be zero. The format of a suitable change depends on the error detection code implemented.

The state diagram for a terminal is simpler for this strategy than for the roll back 2 scheme because

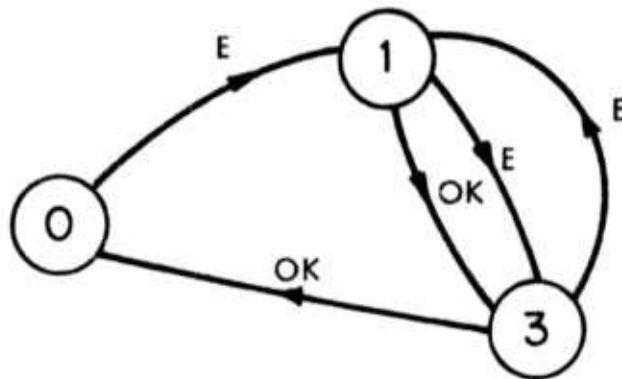
in this case instead of 3 possibilities for a block there are only two possibilities, i.e. E (erased block) and OK (error free block). State 2 therefore is omitted from Fig. 3.4.b. The new state diagram for one terminal is illustrated in Fig. 3.8, while the state diagram for the system is shown in Fig. 3.9. Notice that for a Roll Back K scheme $K \geq 2$ (discussed at (2)) this simplification is still possible. Fig. 3.10 compares the state diagram for one terminal in a Roll Back K scheme (described in (2)) and the simplified version.

3.6 Justification for the Schemes Mentioned in 2.2.3&2.2.4

3.6.1 Lynch's scheme: We now want to deviate from our discussion about continuous transmission schemes and go back to the two schemes suggested by Lynch and Bartlett for a simplex wait and stop transmitting system. Our purpose is to use the previous state representation method for justifying these two schemes. A separate justification for the full duplex schemes of Section 2.3.2 is not then necessary. Again to our best of knowledge no other proof is stated for them.*

In order to find a state representation for Lynch's scheme, first we need to introduce a state diagram

* Nourani (9) has presented a number of schemata to justify these two schemes, but his argument seems to be inadequate.



state 0 : Print received block, send OK, transmit next block
state 1 : Do not print, send erasure(E), go back 2
state 3 : Do not print, send OK, send next block

Fig. 3.8

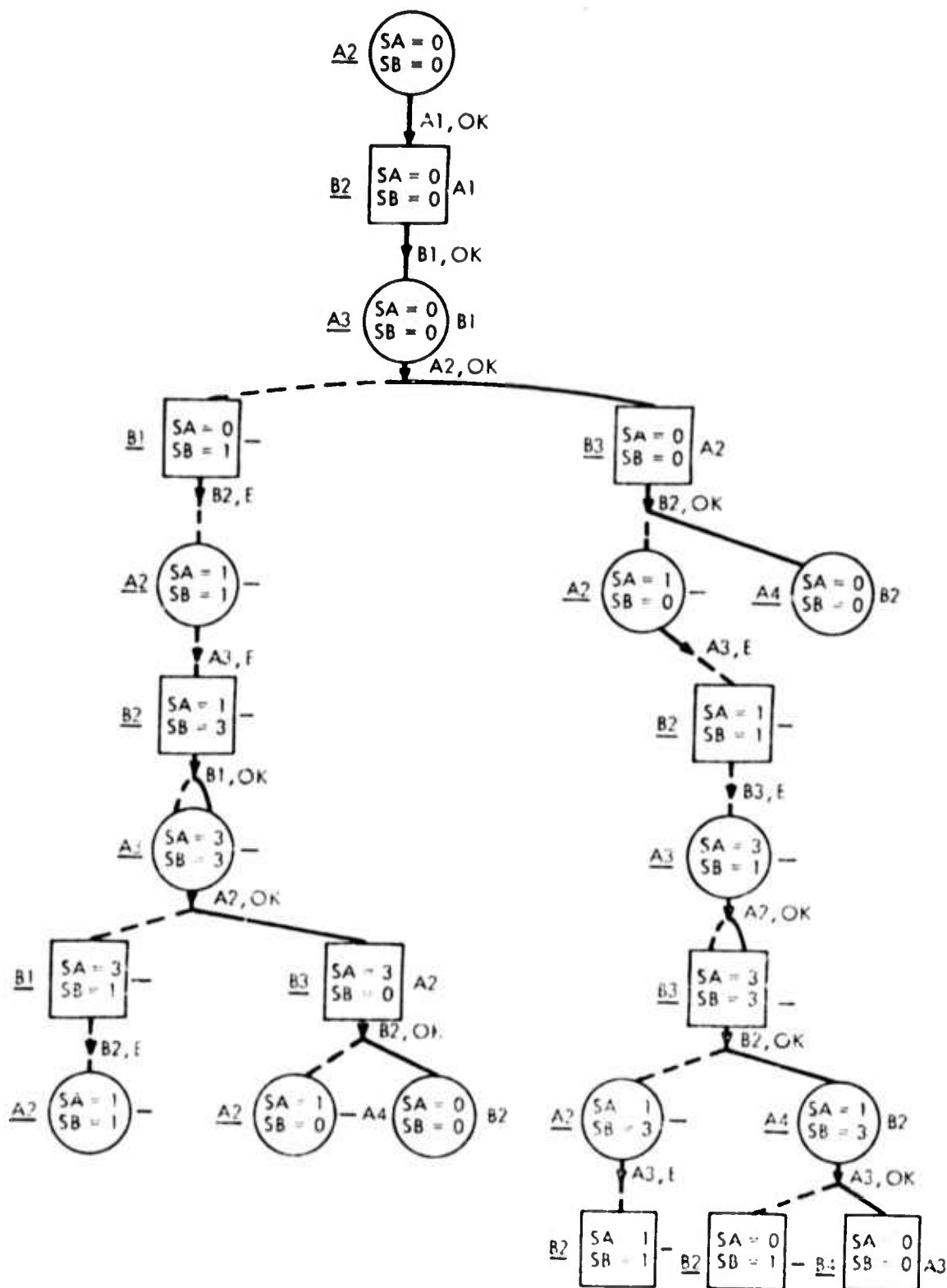
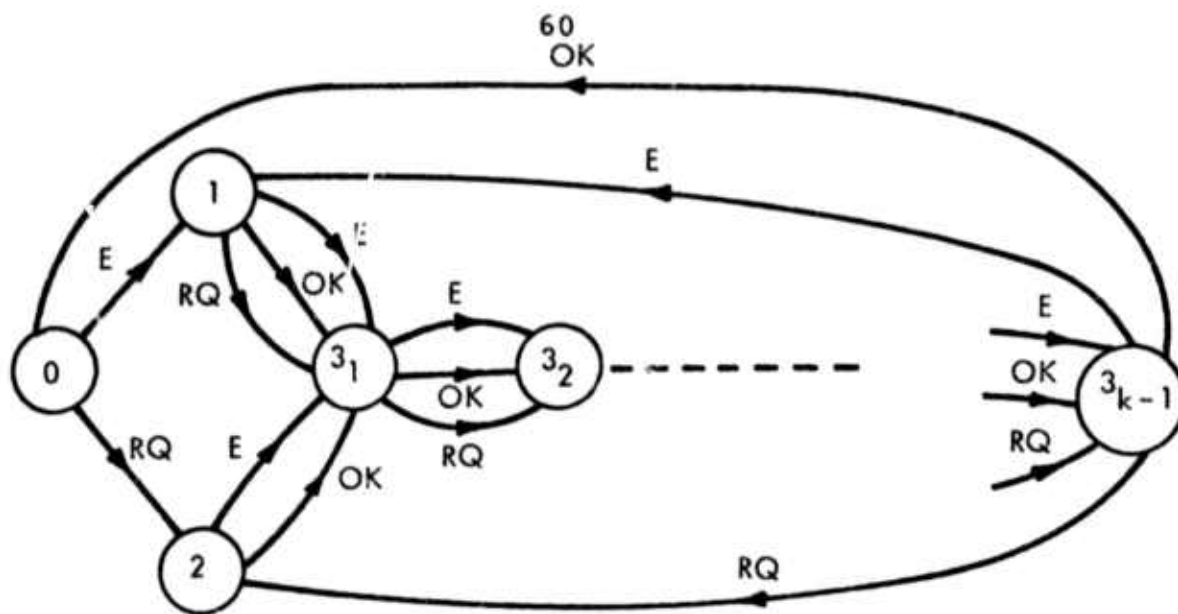
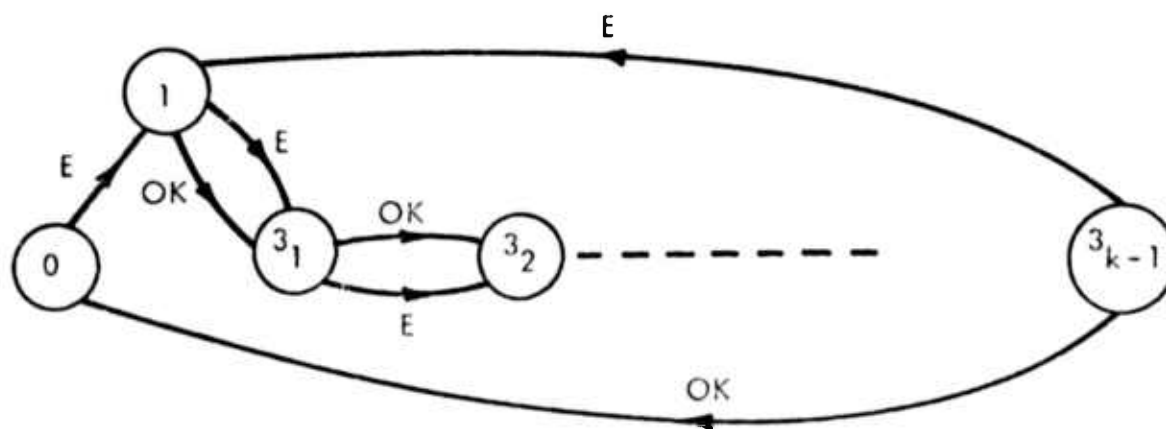


Fig. 3.9



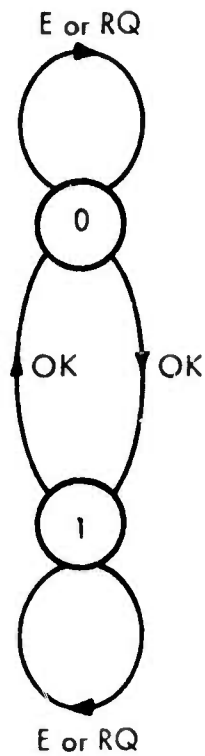
3.10.a: Roll Back K Scheme



3.10.b : Simplified Version

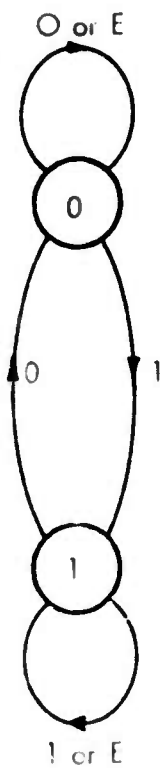
Fig. 3.10

for the two terminals in the system i.e. for A as the transmitter and B as the receiver. Since terminal A, in order to send the appropriate alternating bit, has to remember the previously sent alternating bit, we can refer to this information as the state of the transmitter. On the other hand terminal B in order to decide about a new received block needs to remember the alternating bit of the previously accepted block. This information also can be considered as the state of B. Fig 3.11 then shows the state transitions of A and B together with a table of decisions made by a terminal only in terms of its state and input. We mean by the "input", an erasure (E) or the control data of an error free signal (0 or 1). By the "state" we mean the state before it changes due to the new input. Having done this we can then easily draw the state diagram of the system as a whole (Fig. 3.12). Notice that the symbols and method of demonstration is exactly the same as in Fig. 3.7 except that here the problem is less complicated because it is concerned with a wait and stop transmission. For example in Fig. 3.12 whatever decision a terminal makes is carried out immediately. Furthermore, one terminal works merely as a receiver and another one as a transmitter. Having this diagram and going through a similar argument as used in



Previous State	Input	Action
0	OK	Send a new block, set the Altr. bit to 1
1	OK	Send a new block, set the Altr. bit to 0
0	RQ or E	Retransmit, set the Altr. bit to 0
1	RQ or E	Retransmit, set the Altr. bit to 1

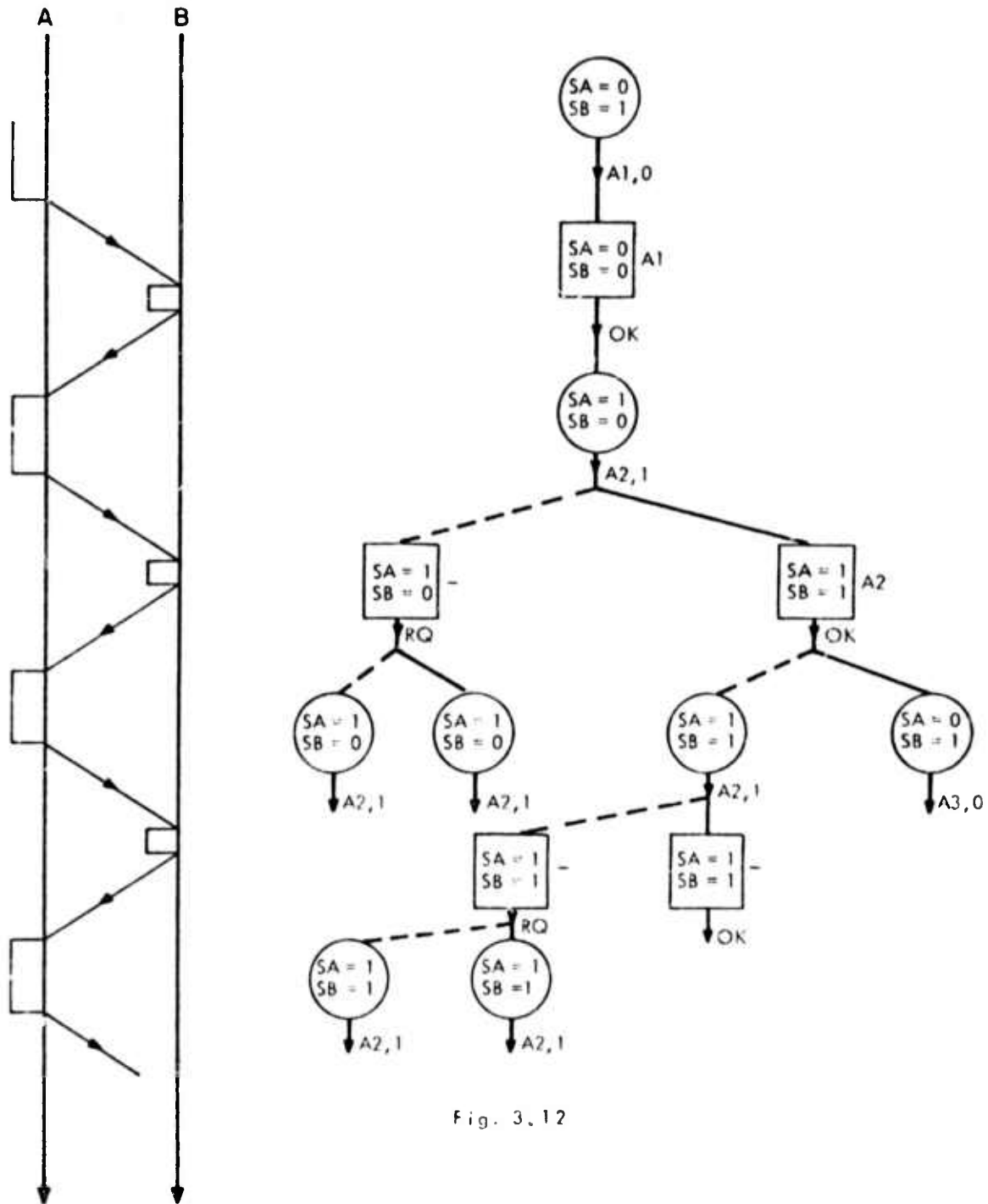
Terminal A : Transmitter



Previous State	Input	Action
0	0	Do not print, send OK
1	0	print, send OK
0	1	print, send OK
1	1	Do not print, send OK
0 or 1	E	Do not print, send RQ

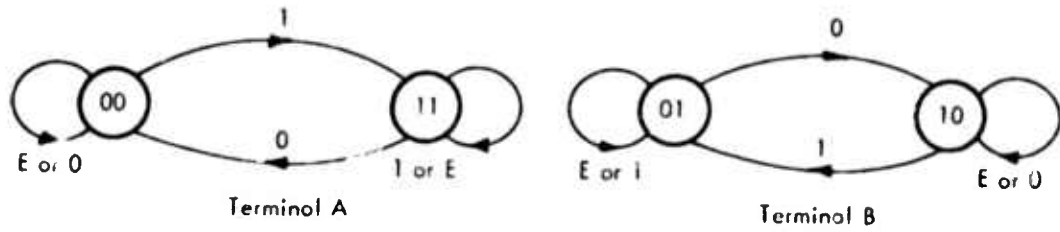
Terminal B : Receiver

Fig. 3.11

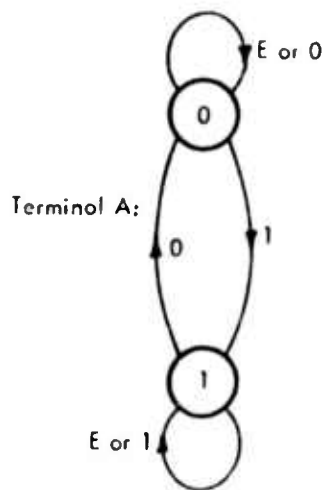


section 3.4.2, the reader can justify Lynch's Scheme easily for himself.

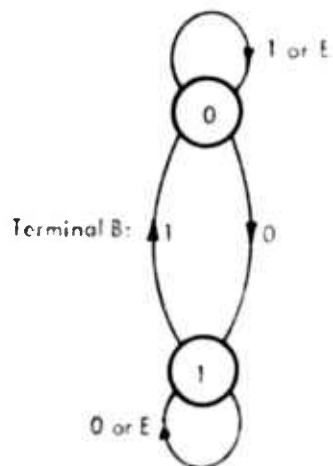
3.6.2 Ready and Acknowledge with Transition Signaling: We repeat now the same kind of things done in 3.6.1. First of all since in this case both of the terminals use alternating bits as control information, there are two bits of information to be memorized by each terminal and hence to be considered as the terminal's state: for example, terminal A needs to know about the previously sent alternating bit in order to send next the appropriate one. It must also know about the previously received alternating bit in order to interpret correctly the next incoming alternating bit. The same thing is true about terminal B. Therefore, we represent the state of a terminal by two binary digits, which are respectively the alternating bits previously sent (S1) and received (S2) by the terminal. Fig. 3.13.a illustrates the state transitions of the system with such a definition both for the receiver (B) and the transmitter (A). Notice that for one terminal S1 and S2 will stay the same always and for the other they will remain different. The reason is that if a terminal receives an erasure, S1 doesn't change because its going to retransmit. S2 also doesn't change because a new control bit isn't received. The same thing is true, if the terminal



3.13.a



Previous State	Input	Action
0	0 or E	Retransmit, set the Altr. bit to 0
0	1	Send a new block, set the Altr. bit to 1
1	0	Send a new block, set the Altr. bit to 0
1	1 or E	Retransmit, set the Altr. bit to 1



Previous State	Input	Action
0	0	Print, set the Altr. bit to 1
0	1 or E	Do not print, set the Altr. bit to 0
1	0 or E	Do not print, set the Altr. bit to 1
1	1	Print, set Altr. bit to 0

3.13.b

Fig. 3.13

receives an error free block that contains a nonchanged control bit. Again none of the state components change. But if an error free block comes in with a new control bit S2 changes and the terminal sends a new block with a new control bit so S1 changes also. Therefore, S1 and S2 either change together or stay unchanged.

Therefore, both for the receiver and the transmitter there is no need to represent the state with two bits and just a single bit is enough. Choosing the first component of state (S1) as the new state (S') Fig 3.13b shows the State transitions and the decisions made by a terminal (both for A and B) in terms of its state and input. After doing this, drawing the state diagram for the system and using it to justify Bartlett's scheme is a trivial matter. The diagram is shown in Fig. 3.14 while the justification is again left to the reader.

3.6.3 Comparison Between Bartlett's Scheme and Lynch's Scheme:

At this point we want to see which one of the schemes proposed by Lynch and Bartlett are more effective and efficient in overcoming erasure patterns. First we will do this comparison between the simplex schemes discussed earlier. Referring to figures 3.12 and 3.14, one can see that if the first erasure in an erasure pattern occurs when A transmits something, then both of the schemes react to it in the same way. This similarity exists

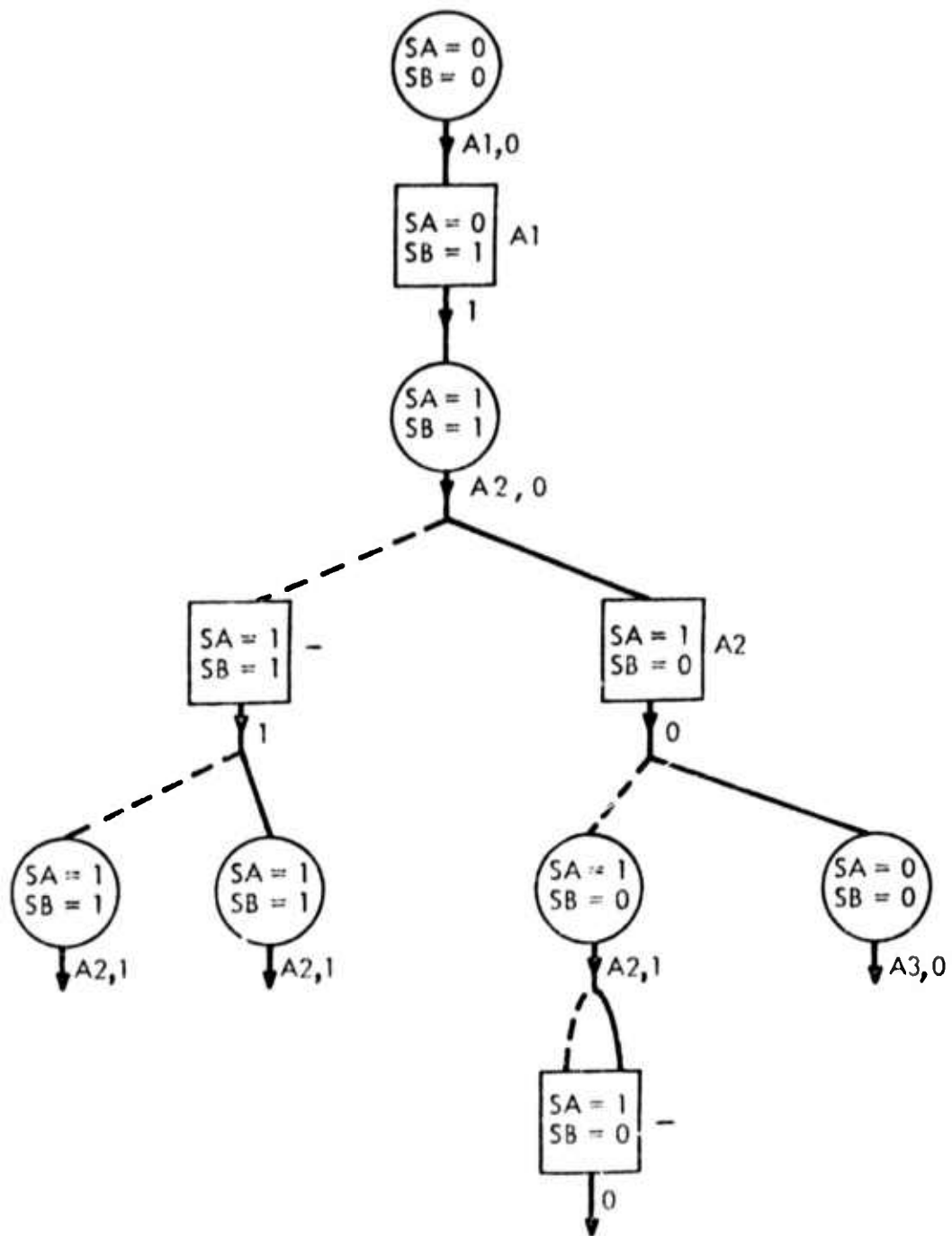
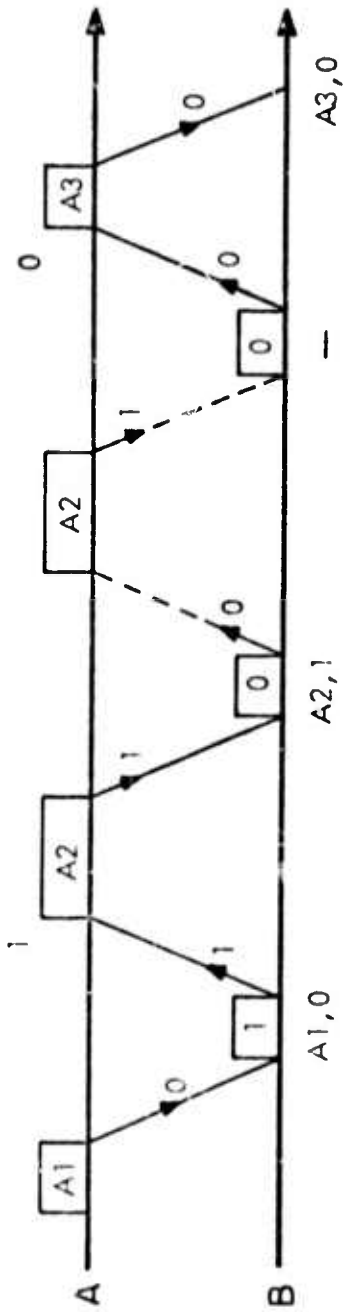


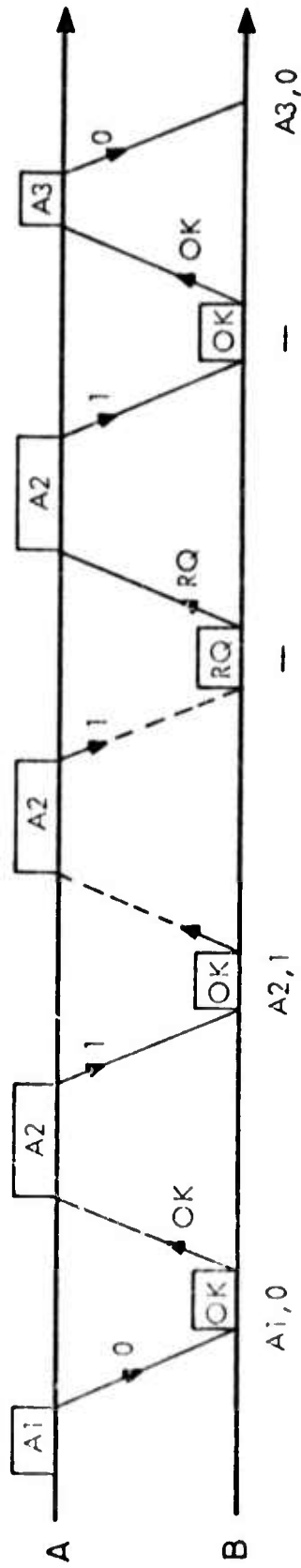
Fig. 3.14

for all other erasure patterns except for one case, despite the different nature of the two schemes. The two schemes react differently only if there are two consecutive erasures, the first one from B to A and the second one from A to D. In this case the scheme of Fig 3.14 overcomes the erasure pattern faster than Lynch's scheme. (Fig. 3.15)

This advantage of course holds true when we extend the problem to the full duplex transmission case. Fig. 2.6 shows this fact clearly. Therefore in the full duplex transmission case, Bartlett's scheme is preferable over Lynch's scheme both because of the above reason and because of using fewer error control bits (as indicated in Section 2.3.2).



3.15.a: Ready and Acknowledge with transition Signaling.



3.15.b: Lynch's Simplex Transmission Scheme.

Fig. 3.15

IV. RETRANSMISSION STRATEGY FOR VARIABLE BLOCK LENGTH

CONTINUOUS TRANSMISSION SYSTEMS

4.1 Introduction

We start our discussion by considering a typical situation which might occur in a variable block length continuous transmission (VBLCT) system (Fig. 4.1). For the sake of generality we have assumed that the length of blocks might be widely variable. As an example notice that in Fig. 4.1 terminal B receives four blocks sent by terminal A (A1,A2,A3,A4) while transmitting only one block (B3). The acknowledgement or repeat request for each of these blocks must be sent in block (B3). Different situations are illustrated for blocks B2 and B4. Terminal B while transmitting B2, receives no block and while sending B4, receives only one block (A5).

For all the other cases which we analyzed previously, this variability does not occur and in fact there exists always a one to one correspondence between blocks sent from A to B and those transmitted by B toward A. In the stop and wait transmission systems the reason for this is that each terminal, after sending one block*, stops and waits for a return block and only then resumes transmission. In continuous transmission systems also, when the blocks all have the same length, this one to one correspondence

* A pure control message is also considered as a block.

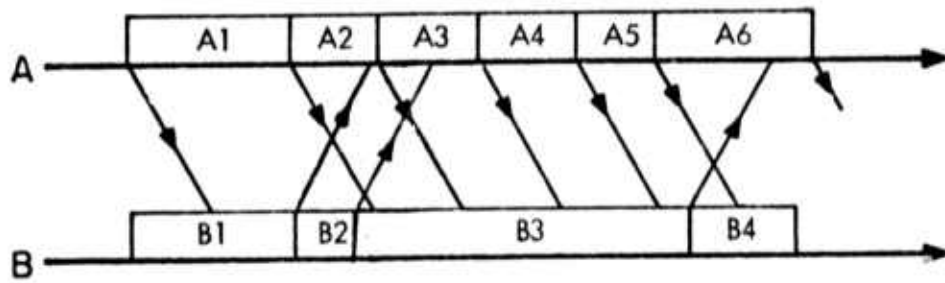


Fig. 4.1

has to occur.

Because of the lack of a one to one correspondence between blocks in the two directions, none of the strategies discussed earlier can be used for a VBLCT system. This raises the following two questions about a retransmission strategy for VBLCT systems.

- a) In the previous problems, since there always existed a one to one correspondence between blocks going and coming back, the receiver could associate a given acknowledgement or repeat request with the appropriate block. How does one solve the problem of specifying the blocks in demand for retransmission at the present case?
- b) How does one then develop a retransmission procedure suitable for the system?

Sections 4.2 and 4.3 are devoted to the discussion of questions a and b. In Section 4.4 we describe details of our proposed strategy. The applicability of the strategy to the cases discussed in Chapters II & III, is shown in Section 4.5.

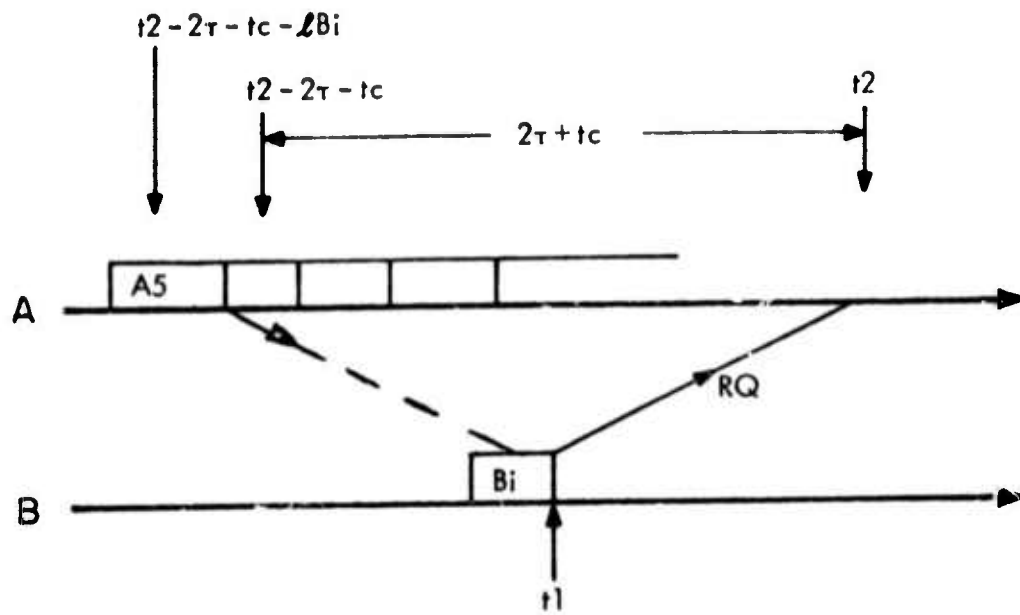
4.2 How to Specify Blocks in Demand for Retransmission

4.2.1 The Method Adopted by IBM in SDLC*

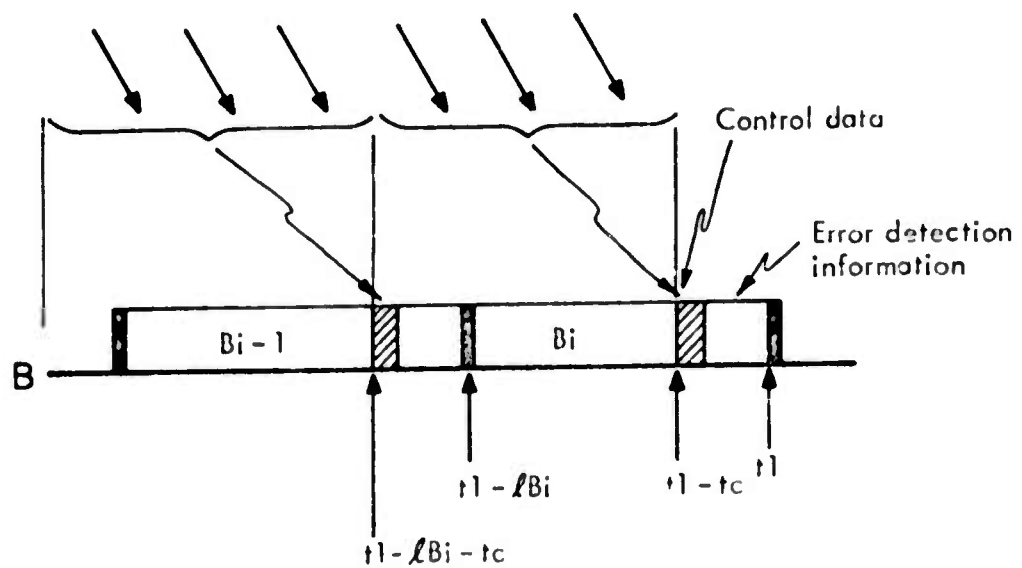
Consider the case demonstrated in Fig. 4.2.a. Terminal B should send a repeat request signal with block B_i . But how does it tell terminal A that the request is concerned with block A5? The solution implemented by IBM is as follows:

Each terminal assigns a count number to its outgoing blocks and inserts this number (NS) into the block itself so that the receiver knows which block is being received. Each terminal also keeps track of the count numbers of consecutive correctly received blocks (NR) and sends the most recent NR with each new transmission. Therefore every block contains two numbers NS and NR. NS being its own count number, and NR specifying the most recent block received by the transmitter. NS and NR both change cyclically between 0 and N. These two numbers are used then to specify which block must be retransmitted. The magnitude of N therefore must be large enough to guarantee that no confusion can arise in the process of controlling the system. In other words N should be large enough so that if a new block with count number $i (0 \leq i < N)$ is sent, all of the affairs of the previous block i have been taken

* Refer to (4) for a detailed description of SDLC.



4.2.a



4.2.b

Fig. 4.2

care of. N in SDLC happens to be 7. Therefore NS & NR together engage 6 bits of space. There are two other bits in each block reserved for other control purposes.

4.2.2 Getting Rid of the Count Numbers:

Although using count numbers NS & NR and inserting them in each block simplifies the problem to a great extent, it also decreases the efficiency of the system because the control field of each block is as long as 8 bits. One question we might ask ourselves at this point is whether transmitting these count numbers is necessary.

Going back to Chapters II & III we see that we have never felt the necessity of transmitting such numbers. Looking more carefully into the example of Fig. 4.2.a, we can see that since the round trip delay of the channel is constant (2τ), terminal A at the moment of receiving a block (say t_2) knows that this has been sent at $t_1 = t_2 - 1$. The repeat request signal in this block then must be related to a block received by B between $t_1 - t_c$ & $t_1 - l_{Bi} - t_c$, where l_{Bi} is the length of B_i and t_c is the length of error detecting information field*.

* Notice that when a block (say B_i) is being transmitted, all the information including the error control data must be inserted before the detecting information (check digits) field starts. Therefore if the transmission of B_i is started at $t_1 - l_{Bi}$ and finished at t_1 , the error control data transmission can not be postponed to after $t_1 - t_c$. Since the control data of B_{i-1} is also inserted before the check digit field starts ($t_1 - l_{Bi} - t_c$), the error control data of B_i relates to blocks received between $t_1 - t_c$ & $t_1 - l_{Bi} - t_c$ (Fig. 4.2.b)

Therefore the block in demand for retransmission is sent by A between $t_2 - 2\tau - t_c$ and $t_2 - 2\tau - t_c - 2B_i$. Now if there is only one block sent by A which terminates in this interval as in the example of Fig. 4.2.a, then the RQ received by A must indicate that block.* We see that without using any count number, the repeat request by itself specifies the block in demand.

As can be seen now, in SDLC the main effort is to make the problem as simple as possible at the cost of using more control bits. Developing a retransmission scheme by taking advantage of the known channel round trip delay is not a new idea. We have been using this idea implicitly since Chapter II. The only thing new here is that the variable block length problem makes the idea more explicit.

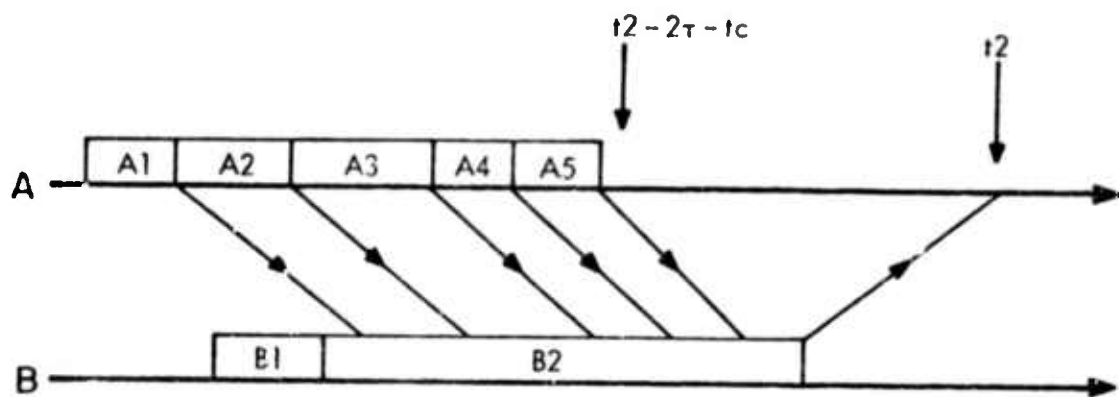
Although our claim of round trip delay being constant seems obvious, let us consider it more critically. The delay of a channel consists of two parts: Propagational delay and Processing delay. The first one is proportional to the physical distance between A and B, thus as long as A and B are static with respect to each other, it is constant. Satellite communication is the only case that this condition gets violated unless the satellite is static with respect to the earth.

* We will consider the case of many blocks terminating in this interval in the next section.

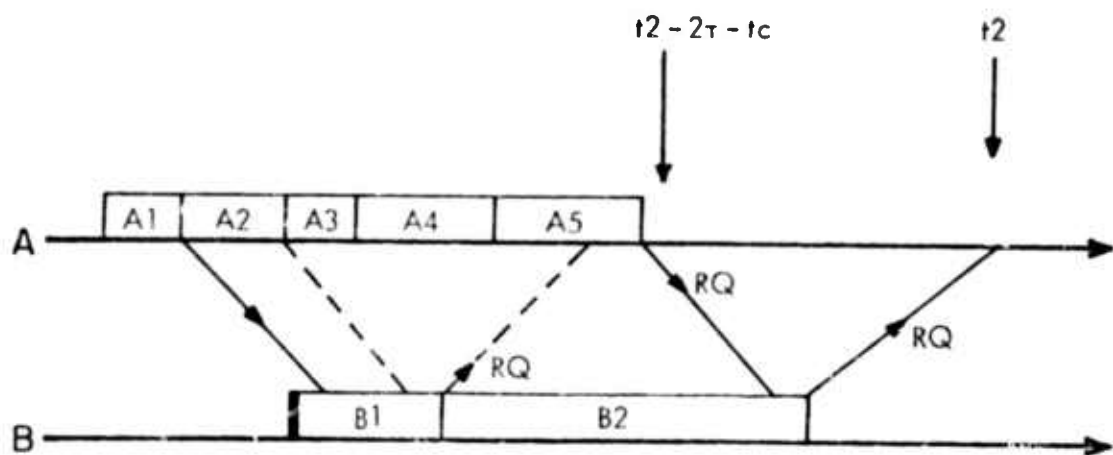
The processing delay (which includes decoding time) might be variable and a function of block length and other parameters. However since the processing delay is known to the receiver itself, this variability should not be considered as violating our arguments. The simplest solution for a receiver is to make the processing delay constant by introducing artificial delays in the processor. Perhaps a better way is to reconsider the value of τ dynamically. It should be added that these considerations are meaningful and necessary only when the decoding time is large compared with the length of a single bit.

4.2.3 Multiple Acknowledgments or Repeat Requests within one Block:

The previous section considered only cases such as in Fig 4.2, where each OK or RQ is concerned with one block. Here we want to consider the problem of multi-acknowledgement (repeat request) as demonstrated in Fig. 4.3. Block B2 in Fig 4.3.a should contain the acknowledgement or repeat request for blocks A1 through A5. Notice again that the error control data carried by B2 will not be detected by A before t_2 , regardless of how and where this data is encoded within B2. Therefore, we always insert all the error control data at the end of the block, just before the check digit field, to give it the chance



4.3.a



4.3.b

Fig. 4.3

of being more up to date*.

Another case where a multi-acknowledgment (repeat request) is necessary is illustrated in Fig 4.3.b. A2 is erased during transmission. Accordingly an RQ is sent within B1. The RQ itself gets erased. The feedback from A then is not satisfactory and B realizes that a new RQ needs to be sent. Therefore B2 should contain a repeat request code indicating A2.

The appropriate format of the error control signal within B2 will be different for an ordered retransmission strategy from a selective one. In the first case, if, for example, blocks A2 and A4 in Fig 4.3.a or Fig 4.3.b are erased, terminal A should go back to the beginning of block A2 and keep on transmitting from that point. Therefore the control signal within B2 needs only to specify block A2. This might be done by sending number i (in this case $i=4$) which means to terminal A that it should start counting backward those blocks sent before $t_2 - 2\tau - t_c$ i.e. A5, A4, A3, A2, A1, ...and then selects the i^{th} one (here fourth one, which is A2) and starts retransmission from it (A2). We choose the convention RQ, i for this kind of repeat request and refer to it generally as "SRQ" (specified repeat request).

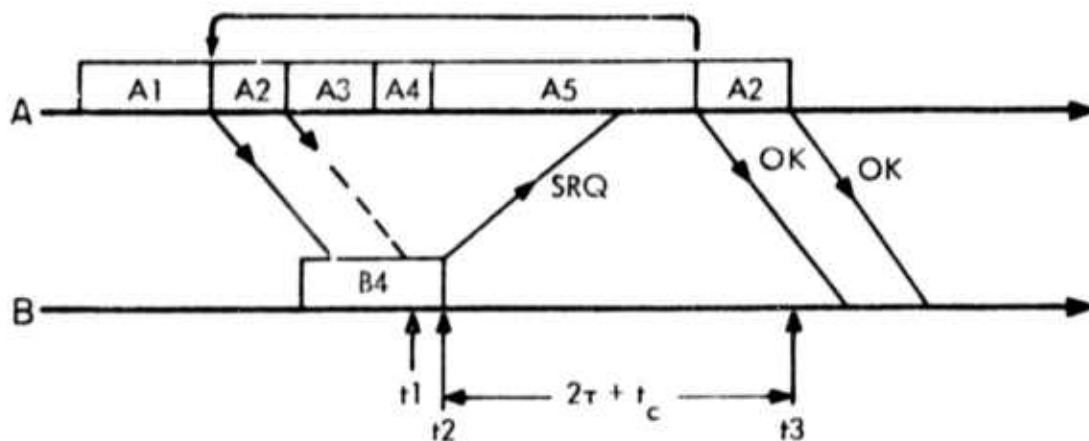
* Notice that the parameter t_c used in last section, then must be the length of check digits and error control data together (Fig. 4.2.b).

For a selective retransmission scheme however, both blocks A2 and A4 need to be specified. This can be done by sending numbers 4 for A2 and 2 for A4. Therefore the SRQ requires more bits compared with the previous case. The final decision about preferring one of these over the other will be made later, since it depends also on some other factors. We will come back later to this issue to describe Huffman coding for SRQ (and of course OK) and also to discuss another aspect of the problem not considered yet (sec 4.4).

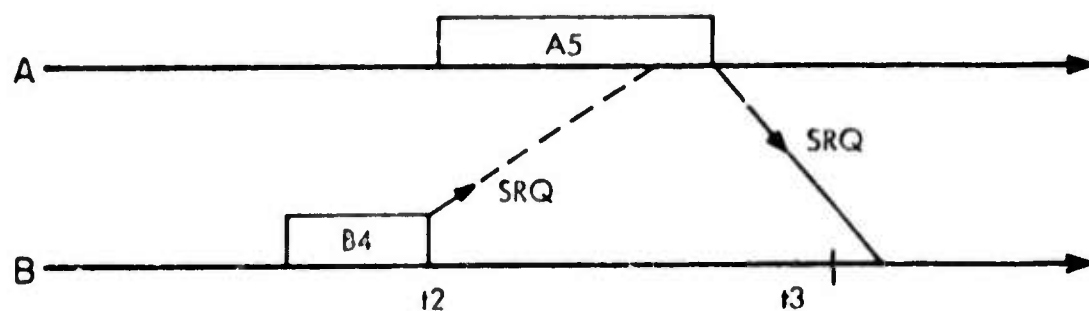
4.3 Developing a Retransmission Procedure

4.3.1 Indication for a Retransmission Being Started

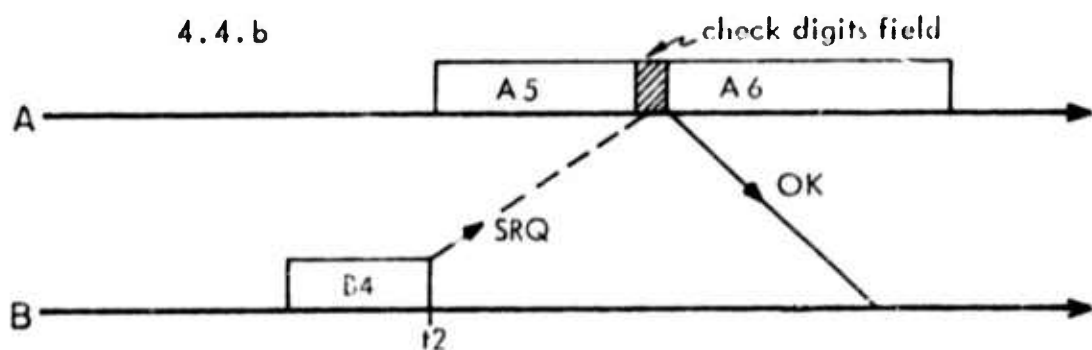
Having at hand a suitable and efficient method for indicating erased blocks to the transmitter, we proceed to discover some difficulties which might face us during a retransmission period. Consider the situation of Fig. 4.4.a. Terminal B receives an erased block at t_1 and sends an SRQ within B4. Block B4 might reach A correctly or get erased on the way. In the first case (Fig. 4.4.a) terminal A goes back and retransmits A2. On the other side B, after sending SRQ at t_2 waits until $t_2 + 2t_{tc} = t_3$ and then looks forward to start receiving a duplicate of A2. But how can it make sure that SRQ is received correctly by A and is not erased itself, as in the case of Fig 4.4.b.



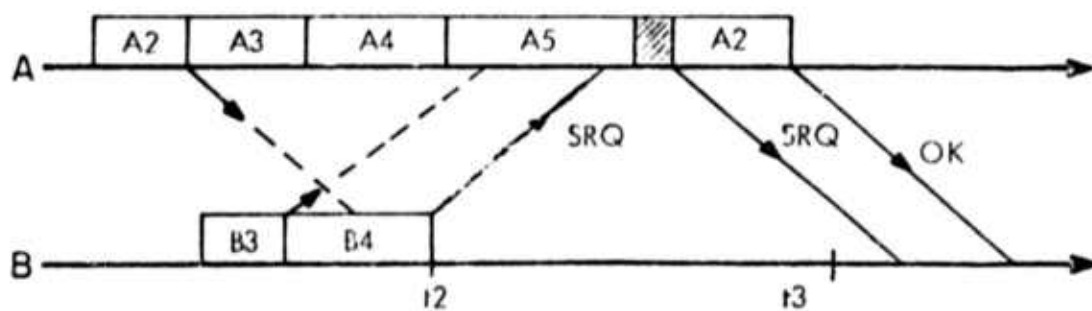
4.4.a



4.4.b



4.4.c

4.4.d
Fig. 4.4

Probably a good indication is the fact that if it gets erased, terminal A sends an SRQ (Fig. 4.4.b) which reaches to A after t_3 . Therefore one might conclude that if the first error control signal which B detects after t_3 is OK then the next block is the duplicate of A2. Fig. 4.4.c shows a contradictory situation however: Block B4 reaches A when A is inserting check digits of A5. Therefore A5 does not contain control data for B4. Accordingly even if A5 gets erased during the transmission as in Fig 4.4.c, A5 can still contain an acknowledgment. This results in a wrong impression since B considers A6 as being A2.

Therefore in such a case another indication must be looked for by B to make sure that the desired retransmission is made by terminal A. We see that the previously described indication is not always effective due to the variability of the block length.

As a more tricky situation consider Fig. 4.4.d, where B3 is also erased but B4 containing SRQ is error free. If an ordered retransmission strategy is used by the system then the SRQ in block A5 just specifies block B3 meaning that the retransmission is required from block B3, and does not clarify whether B4 also was erased or not, hence whether the block next to A5 is A2 or A6. This example shows that the described indication of a retransmission being made is not only ineffective in special block

length arrangements (which is known to A and B and hence can be modified for these cases) but in general is ambiguous. This second difficulty is not restricted to the case of an ordered retransmission scheme. In fact, by considering different erasure patterns and block length arrangements, one can easily find various difficulties for a selective retransmission case.

Notice that it is possible to think of a solution and a better indication in the case of each example, but our objective is to come up with a solution applicable to all of the situations. Unfortunately the number of possible situations here is not limited despite those cases considered in chapter II and III*.

The puzzle can be solved only by using a new concept, which, although simple and trivial, is very effective: Each terminal, when it goes back for retransmission, inserts a special character in the first block meaning to the other terminal that a retransmission has started. We refer to this character as R_t (Retransmission).

4.3.2 Erasures to be Taken as Acknowledgments (OK's):

We observed in section 2.3.2 that using a static code (verify bit) to specify that a retransmission has

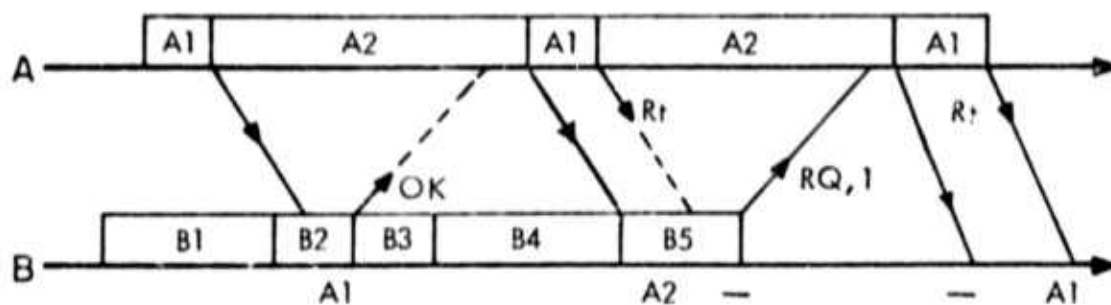
* We could represent the operation of systems in chapter II and III with finite state diagrams. In fact such a representation turns out to be impossible for a case in which we have an infinite number of block length arrangements.

been made, is not effective and results in ambiguity. Notice that this conclusion was made in a case where each erasure is considered as having an RQ. Unfortunately the same thing is true here with the difference that now we have a much more complicated system. Thus those cases in which the scheme fails to work are more numerous. Fig. 4.5.a shows an example similar to Fig. 2.2.b. Terminal A assumes that the erasure contains an RQ. A second erasure after that causes the other terminal to have a double print.

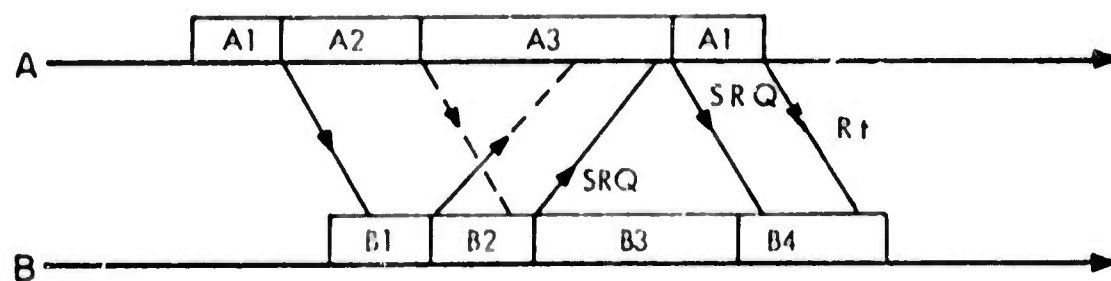
Fig. 4.5.b shows another example which causes confusion. Terminal A after receiving an erasure (B1) decides to retransmit A1 after A3. The SRQ within B2 however requests for retransmission of A2. A, naturally first resends A1 with Rt. Terminal B then considers the next coming block as A2 unless it can conclude from the SRQ in A3 that A is going first to retransmit A1. To give a terminal the ability of making this kind of conclusion first of all, requires too much preparation and secondly it will be limited to certain cases.

The idea of using an alternating bit (or code) is not useful here as it was in Chapter 11 (Lynch's scheme), because there is no regularity and periodicity for the block length arrangement in this case.

There is however another solution. Even though we chose to consider every erasure as an RQ in section 2.2.1,



4.5.a



4.5.b

Fig. 4.5

there is no reason to restrict ourselves forever to this choice. We will see that at least in the present case it is simpler not to take each erasure as an RQ. The next section describes our proposed strategy with such an attitude.

4.4 Proposed Strategy

4.4.1 Appropriate retransmission procedure:

1) Each terminal retransmits only when it receives an error free block containing an SRQ. In this case it goes back to the specified block and starts an ordered retransmission. It also sends Rt. with the first block it resends.

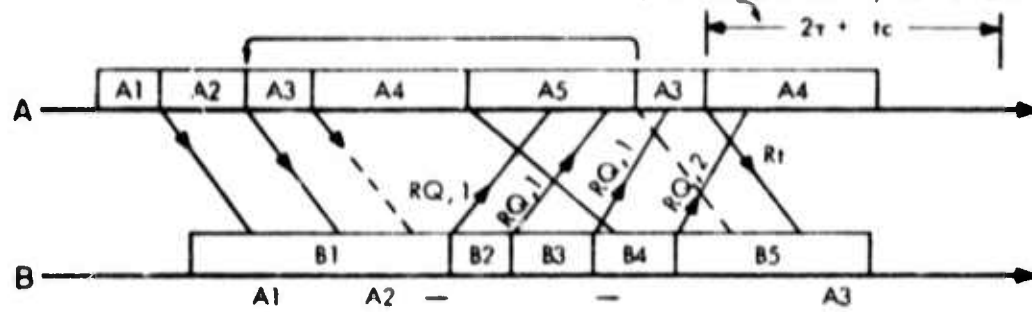
2) A terminal starting a retransmission does not take any action upon other SRQ's which arrive less than $2\tau + t_c$ seconds after the end of the first retransmitted block. (The reason will become clear soon)

3) Each terminal that receives an erased block sends an SRQ at the first opportunity. This erasure of course has no effect on the sequence of out going blocks from the terminal. The erased block and those following it before an Rt gets detected will not be printed. An Rt is not expected within those blocks coming in less than $2\tau + t_c$ seconds after SRQ is sent. When a block containing Rt comes in after $2\tau + t_c$ seconds then the terminal starts printing the incoming blocks again.

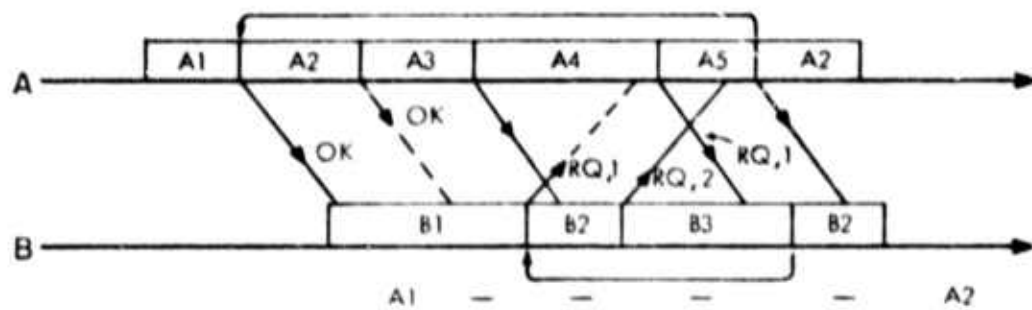
4) A terminal, after sending an RQ and before receiving an Rt, has the option to send additional SRQ at arbitrary times. The only thing is that the SRQ must specify the correct block (Fig. 4.6.a). Doing so is useful only to make sure that if one RQ gets erased on the way the other one can still cause a retransmission (Fig. 4.6.b). Notice that if after sending an SRQ a terminal does not detect Rt in the first block which starts receiving after $2t + t_c$ seconds, it means that the SRQ is not detected (Fig. 4.6.c). In such a situation it is necessary (not arbitrary) to send another SRQ. This is also true if the first block received after $2t + t_c$ seconds is erased (Fig. 4.6.d). Remember also that if one SRQ is detected the additional SRQ's do not have any undesired effect due to the restriction in part 2.

Fig. 4.7 illustrates two new examples of what might happen in the system and serves to clarify the foregoing scheme. As can be seen now, in this scheme, the transmission of blocks in one direction proceeds, regardless of the erasures in the other direction and this is a major advantage of this scheme. The freedom of sending additional SRQ's is another advantage. There is a trade off in choosing the number of SRQ's to be sent. First the greater the number of SRQ's, the faster in average the other terminal starts to retransmit, and less time will be lost

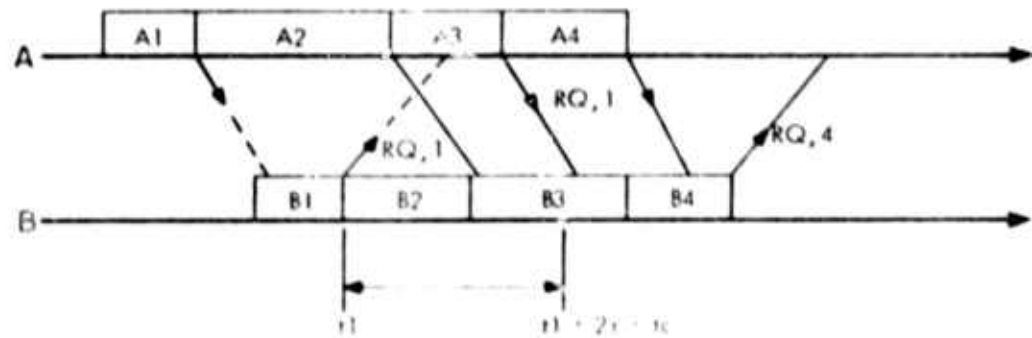
A does not consider any new SRQ in this interval



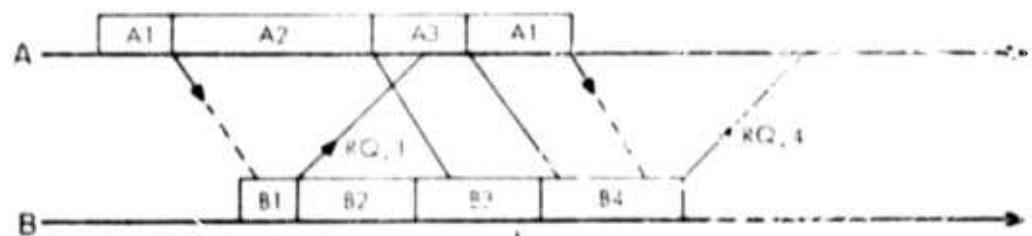
4.6.a



4.6.b



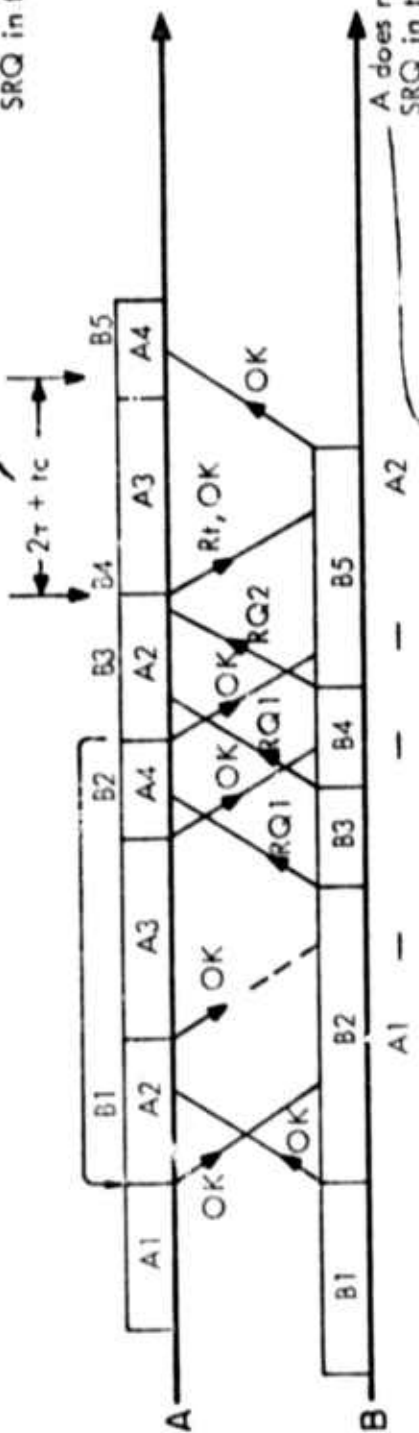
4.6.c



4.6.d

Fig. 4.6 $2\tau + t_c$

A does not consider any new SRQ in this interval.



89

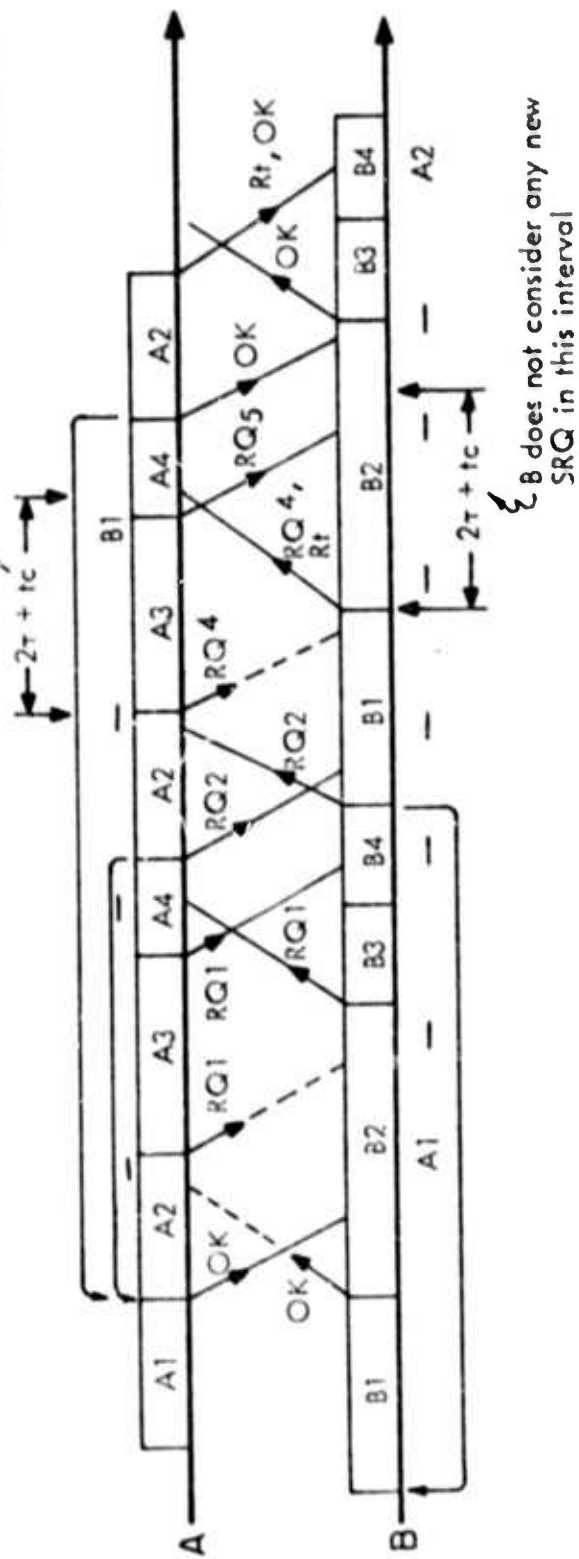


Fig. 4.7

for unnecessary retransmission of blocks. (Notice that since the retransmission is ordered, several error free blocks may be retransmitted). On the other hand each SRQ takes a number of bits in a block, and, as we see in the next section, as an SRQ points to a further block it becomes longer. Therefore using additional SRQ's takes more and more room in each block. In general we conclude as a rule of thumb that since it is very unlikely to have two consecutive erasures, it is better not to send additional SRQ's unless it becomes necessary. For special cases we might reconsider this conclusion.

4.4.2 Huffman Code for error control signal

The error control part of every block in the foregoing scheme has two items of information: First it specifies whether a retransmission is started (Rt) or not. Second it specifies the blocks in demand for retransmission if any (SRQ or OK). Accordingly the following possibilities exist for an error control signal:

OK

OK & Rt

RQ_i, $i = 1, 2, 3, \dots$

RQ_i & Rt, $i = 1, 2, 3, \dots$

Having the probability assignment for this set of outcomes, one can easily find it's Huffman Code which has the smallest average length. In fact, in the present

case, since the probability of having an erasure in the channel and hence having RQ and Rt is very small (perhaps less than 0.01), one can think of the following code as having an average length very close to that of the Huffman code.

OK	0
OK,RT	10
RQ1	1101
RQ2	1111
RQ3	1110
RQ1,Rt	110001
RQ2,Rt	110011
RQ3,Rt	110010
RQi	1100000 $\overbrace{\dots 0 \dots}^{1-3} 1$
RQi,Rt	1100001 $\overbrace{\dots 0 \dots}^{1-3} 1$

Of course knowing how often an RQi happens mainly depends on the block length variability and error statistics of the channel. For example if 90% of the blocks are 50 bits long and 10% of them 500 bits, then we will have almost as many RQ1 as RQ5. On the other hand, if blocks in 95% of the cases are 200 bits long and in the rest of the cases smaller than 200 bits, then almost all the SRQ's are RQ1 and rarely RQ2. RQi ($i > 2$) almost never will be used. In the described code it is assumed

that most of the RQ's are RQ1, RQ2 and RQ3. The important conclusion one can get from this is that despite introducing an extra character (Rt) in the scheme, the average length of the error control code is slightly larger than 1 (because erasures happen rarely). Therefore the price we are paying for having blocks with variable length is negligible.

One important thing which should be indicated at this point is that since we are using a variable length code for error control data, then it will vary a few bits. Some new considerations then are necessary, since $2^{\tau} + t_c$ is something we have used several times as a constant.

4.4.3 Lack of Synchronism after Erasures

In this chapter we have made an assumption implicitly; A terminal always recognizes the start and the end of an incoming block, regardless of the block being error free or not. This assumption is of course correct for the constant block length case. For the variable block length problem, however, it is not valid if the protocol information about the length of block is encoded into the block itself (probably all of the cases). In these cases the assumption is violated because as the block gets erased, the information about its length also gets lost. Therefore we cannot send an RQi because i is unknown. For

example, in Fig. 4.8 terminal B at some time after t_1 finds out that the block after A1 is erased. It does not know how long it is and how many other blocks have come in before t_2 . Therefore i is unknown to B.

One possible solution is to define i as the number of bits between t_1 and $t_2 - t_c$. Therefore the other terminal after receiving RQi at t_3 first goes back $2T + t_c$ seconds and then counts backward i bits to find the start of the erased block. This solution is very simple; however one might think that since i gets very large, (perhaps over a thousand bits for the first SRQ) it is inefficient. If we implement constant length codes for i , then it will be longer than 10 bits, in addition to the fact that constant block length puts a limit on the size of i , unless we increase the length of code for i exceeding some limit. The following error control code is one example. (Notice it is not a Huffman code but has an average length close to it).

OK		0	
RQi	$i < 2^{12}-2$	1	$\overbrace{1 \dots 1}^{12 \text{ bits}}$
RQi	$i \geq 2^{12}-1$	1	$\overbrace{11 \dots 1}^{12 \text{ bits}}$

Notice that it is necessary that each retransmission starts with a flag (a special long code) to let the

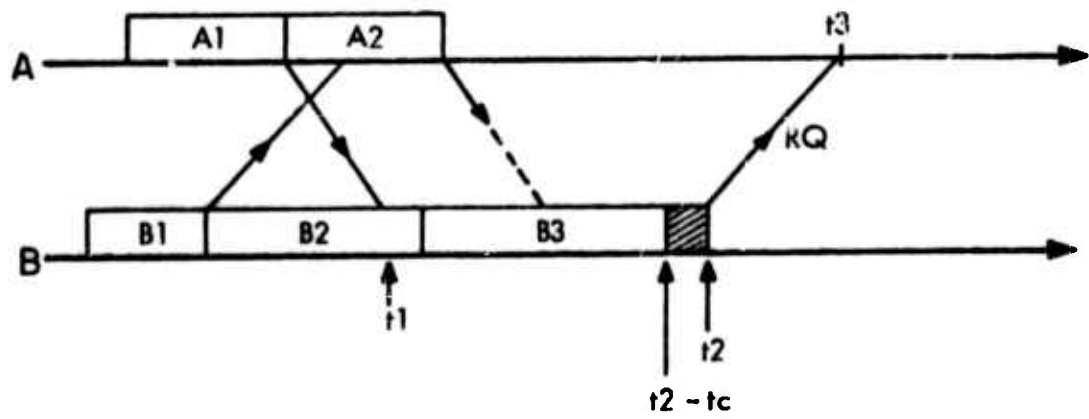


Fig. 4.8

receiver maintain the synchronism again*. In this case, the flag plays the role of R_t as well. This is why R_t has not appeared in the above code.

Despite what we thought earlier, the average length of code is very close to 1 since the probability of a block being erased, presumably is not much larger than 0.01. A modification to the above solution is to use the smallest possible block length as the unit of measuring the time distance between t_1 and $t_2 - t_c$ (Fig. 4.8). This measurement then will be approximate and by setting the approximation to be always rounded up or rounded down, there remains no ambiguity for the other terminal to determine the block in demand for retransmission. Notice however that this modification does not have any appreciable effect in decreasing the average length of the error control signal. It is only from a conceptual point of view important to know about different ways of defining "i".

A more interesting method is to use a similar kind of idea to that implemented by SDLC (Sec. 4.2.1). Each terminal assigns a count number cyclically varying between 0 and N to its outgoing blocks. This number

* In the same time we have to prevent the content of a block from having a sequence of bits similar to the flag, by using "insertion" technique. The error control code described right now also must be subject to insertion.

is not sent with the block itself contrary to SDLC.

If initially the two terminals are informed of the number of the first block they receive, as long as there is no erasure, each terminal knows the count number of incoming blocks. When an erasure happens, then this count number provides means for specifying the erased block (Fig. 4.9). Again N must be as large as to make sure no ambiguity might happen in specifying a block. The block length variability is the main factor in determining the suitable value of N . Notice that the method described here is different from SDLC by the important fact that a count number is rarely sent.

An important point is that it is more efficient that a terminal after receiving an SRQ goes back immediately and retransmits instead of first completing the block currently under transmission (as we have done in this chapter), because the synchronism is going to be lost any way.

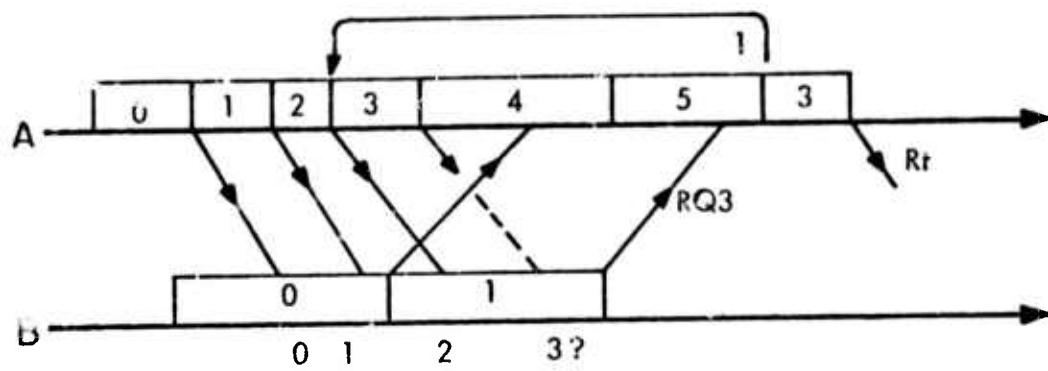


Fig. 4.9

4.5 Conclusion

4.5.1 Efficiency Considerations : We showed that in the foregoing strategy, the average length of error control data is very close to 1, if we implement an efficient coding scheme. From this view point the strategy is very efficient and comparable with the strategies studied in previous chapters. The Roll Back K scheme uses one control bit per block. Its simplified version turned out not to require any control bit. Lynch's scheme and Bartlett's scheme for full duplex and stop and wait transmission use respectively 2 bits and 1 bit per block. The decomposition scheme of section 3.2 requires 2 bits or 1 bit of control data per block depending on the scheme it uses for its decomposed stop and wait transmission systems. Considering the generality of the proposed strategy in this chapter, approximately one bit of control data used per block there is a good figure. There is, however, another dominant factor: that of evaluating the maximum through put yielded by the strategy. How effectively and quickly does it overcome erasure patterns, compared with the other schemes?

To answer this question remember that in the schemes of the previous chapters, each erased block is taken as an RQ and hence causes a retransmission by the receiver.

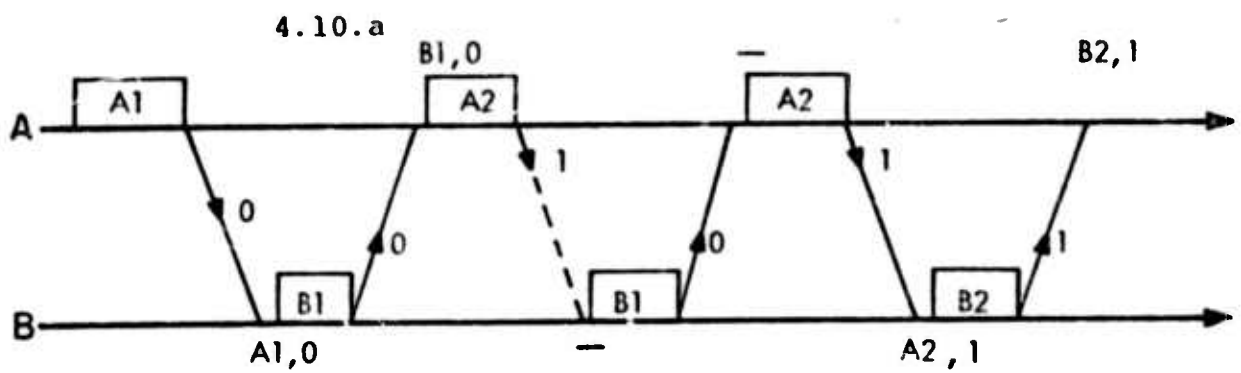
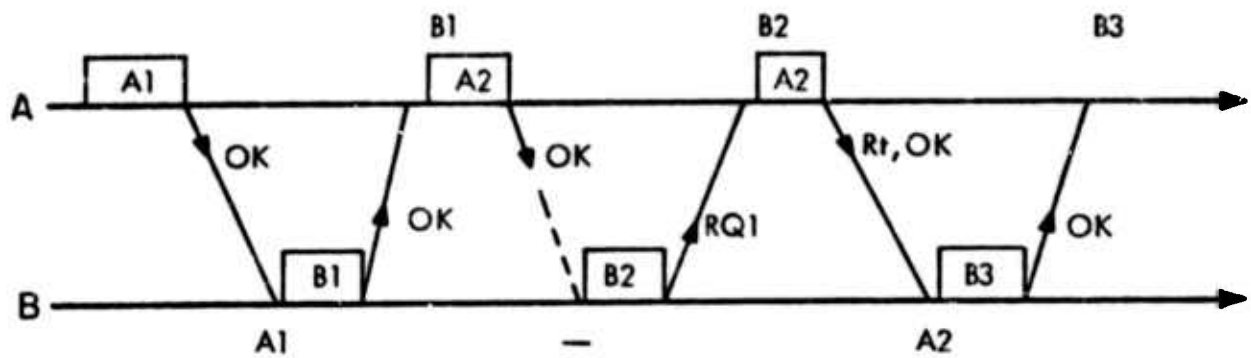
This is undesired in almost all of the cases, because if a block is erased (say from A to B), as we have pointed out earlier; it is very unlikely that the previous block from B to A was also erased*. Therefore, most often, taking an erasure as an RQ causes some wastage of the channel use. The only place that we have chosen to take an erasure as OK is in the strategy of this chapter. This advantage suggests that we implement the same strategy for the more specific cases like stop and wait transmission etc.

4.5.2 Implementation of the Strategy for Specific Cases:

Stop and wait transmission is one case in which our strategy can be implemented easily. Fig. 4.10.a shows the performance of the system when a single erasure occurs. To see its advantage compare it with Fig. 4.10.b where Bartlett's scheme is implemented. (Lynch's scheme yields the same performance).

Implementation of the strategy for the constant block length and continuous transmission case, however, causes some difficulties. The coding scheme used for error

* If the probability of having an erasure is ϵ ($\epsilon < 1$) and if the length of blocks are much larger than burst noise length (so that the channel can be approximated as a memoryless channel regarding block erasures, although it may have memory with respect to bit errors), then the probability of having two consecutive erasures or two erasures with one error free block in between, will be something in the order of ϵ^2 . In the same way, the probability of all other erasure patterns is negligible compared with a single erasure.



4.10.b
Fig. 4.10

control data is a variable length code and hence should be changed to a constant length code in order not to change the length of block. In order to avoid encoding error control information into long codes, then we need to put an upper limit on the value of i in an RQ_i . If we restrict i to be less than 8 then the error control code will be 4 bits long*, plus the fact that we have to take some emergency action, when i gets larger than 7.

There is a trade off now in determining whether this strategy is more efficient than the Roll Back K scheme. The number of control bits in this strategy is more while it does not waste channel usage by making unnecessary retransmissions. Block length and the erasure statistics of the channel must be known before choosing one of these strategies over the other one.

Fig. 4.11 shows the performance of the new strategy for several erasure patterns. Notice that for a single erasure pattern only two blocks are retransmitted while in a Roll Back 2 scheme, 4 retransmissions are required. A further improvement will result if we try to use a selective retransmission scheme, since in this case only one retransmission is required in the case of a single erasure pattern. Although in general we have not designed

* One bit to show whether it is $Rt.$ (retransmission) or not, the rest for indicating the number i (0-7).

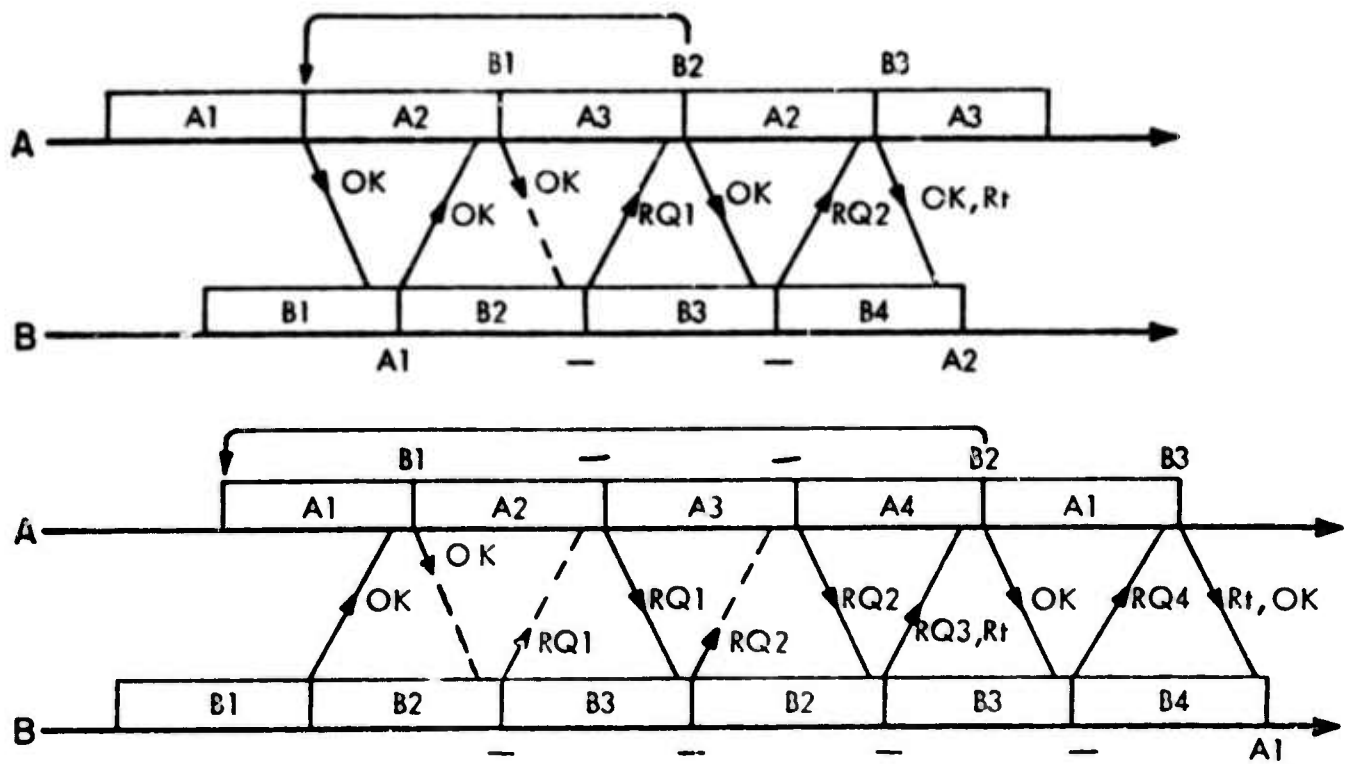


Fig. 4.11

a selective scheme (with taking an erasure as OK), in the present case we can do so. We can implement the selective scheme studied in section 3.2, while using the new strategy for each one of its stop and wait decomposed transmission systems. This will yield excellent performance, requiring only one retransmission in the case of a single erasure pattern (Fig. 4.12). Notice however that the problem of having several error control bits per block still holds true. Another thing to mention is that the implementation of a selective retransmission scheme is more difficult due to the greater buffering and control equipments requirements.

To complete our discussion in this part, Fig. 4.13 shows the implementation of the new strategy in another situation we have not looked at yet: half duplex continuous transmission (with variable or constant block length). As can be seen, since the receiver sends the error control data immediately after receiving each new block the situation is somewhat simpler.

4.5.3 Suggestions for Further Research

The following are some promising questions which can be the subject of new investigations:

- 1) All that we have said so far is concerned with a communication channel between two terminals. What kind of modifications need to be made and/or what new strategies

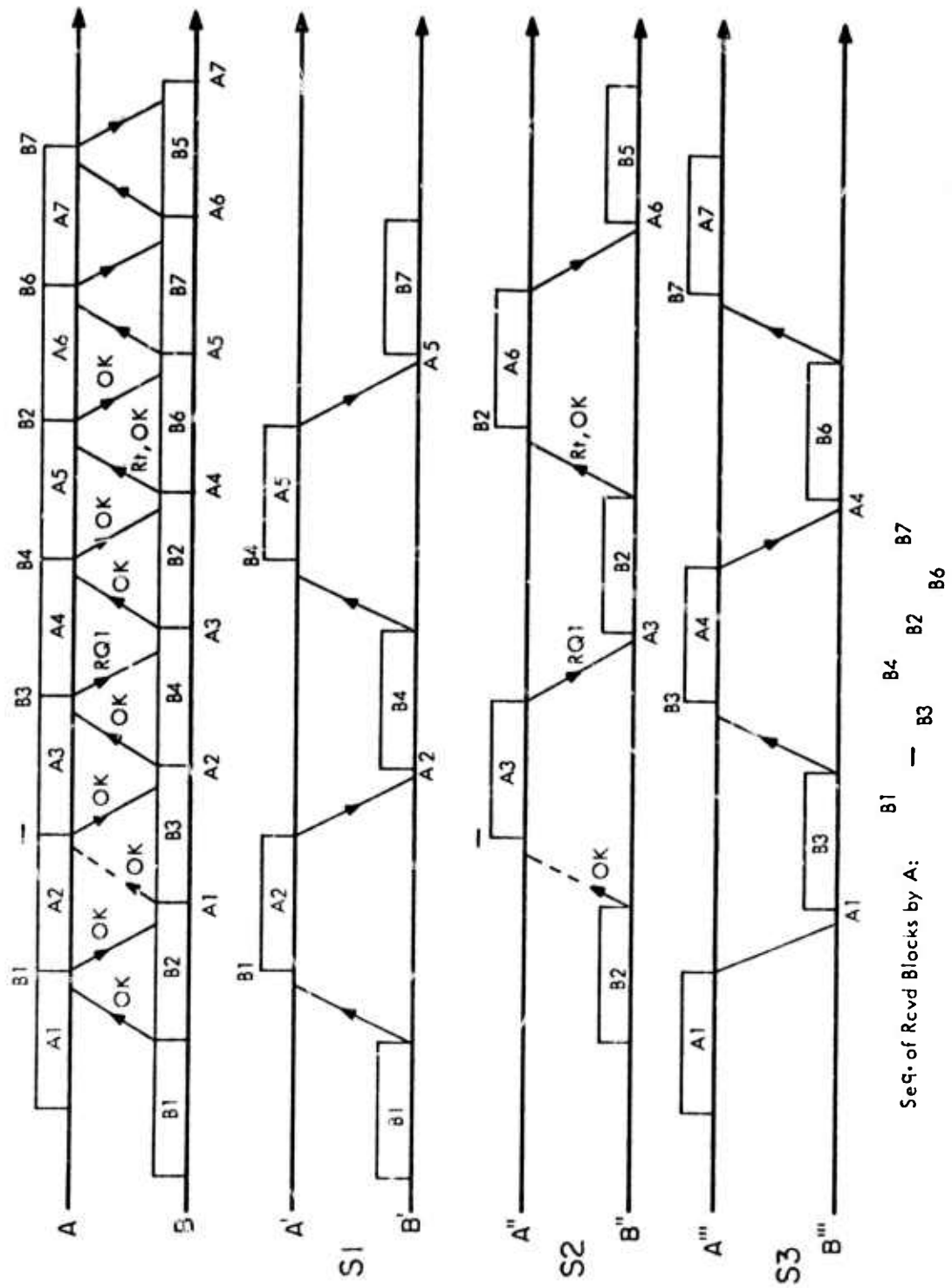


Fig. 4.12

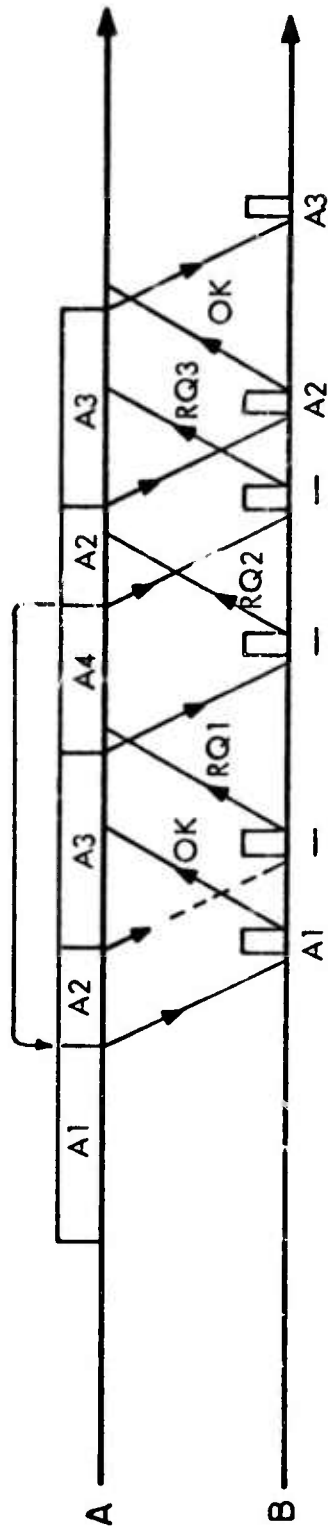


Fig. 4.13

should be designed for implementing a system with more than two terminals (Network Communication).

2) Can the ordered strategy proposed in this chapter, be modified somehow to result in a selective strategy for the VBLCT systems? What kind of modification should be made?

BIBLIOGRAPH

- (1) Gallager, R.G., Information Theory and Reliable Transmission: Wiley Press, 1969, Chapter 6.
- (2) Reiffen, B., Schmidt, W.G. and Yudkin, H.L., "The Design of an Error-Free Data Transmission System for Telephone Circuits", Trans. AIEE on Communication and Electronics, July 1961.
- (3) Wozencraft, J.M. and Horstein, M., "Coding for Two-Way Channels", Fourth London Symposium on Information Theory, London, England, August 1960.
- (4) Kersey, J.R., "Synchronous Data Link Control", IBM Data Communications, 1974.
- (5) Davies, D.W. and Barber, D.L.A., Communication Networks for Computers, Wiley, 1973, pp. 234-235.
- (6) Lynch, W.C., "Reliable Full-Duplex File Transmission over Half-Duplex Telephone Lines", Communication of the ACM, No. 5, May 1969.
- (7) Bartlett, K.A., Scantlebury, R.A., Wilkinson, P.T., "Transmission over Half-Duplex Links", Communication of the ACM, No. 5, May 1969.

- (8) Metzener, J.J., Morgan K.C., "Coded Feedback Communication Systems", Proceedings National Electronics Conference, 1960.
- (9) Nourani, F., "Derivation of a Testing Method for Repeat Request Logic", S.M. Thesis, Elec. Eng. Dep., M.I.T., 1973.
- (10) Benice, R.J., Frey, A.H., "An Analysis of Retransmission Systems", IEEE Trans. on Communication Technology, Dec. 1964.
- (11) Moore, J.B., "Constant Ratio-Code and Automatic-RQ on Transoceanic HF Radio Services", Trans. I.R.E., Vol. CS-8, No. 1, March 1960.