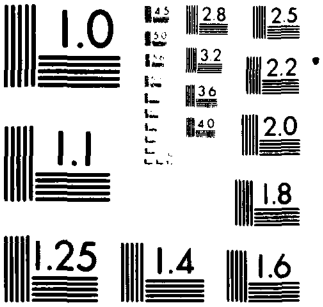
AD-A092 272 DELTA INFORMATION SYSTEMS INC JENKINTOWN PA F/G 9/2
USE OF A MICROPROCESSOR TO IMPLEMENT AN ADCCP PROTOCOL (FEDERAL--ETC(U)
JUL 80 S J URBAN, R SCHAPHORST DCA100-79-C-0050

UNCLASSIFIED NCS-TIB-80-7 NL

1 of 2
AD
A092272



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

USE OF A MICROPROCESSOR TO IMPLEMENT AN ADCCP PROTOCOL (FEDERAL STANDARD 1003)

July 1980

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER NCS TIB 80-7	2. GOVT ACCESSION NO. AD-A092272	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Use of a Microprocessor to Implement An ADCCP Protocol (Federal Std-1003).		5. TYPE OF REPORT & PERIOD COVERED FINAL
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Stephen J. Urban Richard Schaphorst		8. CONTRACT OR GRANT NUMBER(s) DCA100-79-C-0050
9. PERFORMING ORGANIZATION NAME AND ADDRESS Delta Information Systems, Inc. 259 Wyncote Road Jenkentown, PA 19046		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS National Communications System Office of Technology and Standards (NCS-TS) Washington, D.C. 20305		12. REPORT DATE JULY 1980
		13. NUMBER OF PAGES 110
14. MONITORING AGENCY NAME & ADDRESS(If different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution unlimited; approved for public release

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)

ADCCP	Federal Standard 1003
Unbalanced Normal Mode	
Balanced Asynchronous Mode	
Unbalanced Asynchronous Mode	
Protocol	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The objective of this program is to develop block diagrams, flow charts, and computer programming for the unblanced normal, balanced asynchronous and unbalanced asynchronous class of procedures in accordance with Federal Standard 1003 "Telecommunications: Synchronous Bit Oriented Data Link Control Procedures (Advanced Data Communication Control Procedures)". The purpose of this effort is to determine the feasibility of using the M6800 or similar microprocessor, to implement this type of protocol, and to obtain an estimate of memory and processor resources that would be required.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

2

TECHNICAL INFORMATION BULLETIN 80-7

6 USE OF A MICROPROCESSOR
TO IMPLEMENT AN ADCCP PROTOCOL
(Federal Standard 1003),

11 JULY 1980

10 [illegible] J./[illegible]
[illegible]

9 [illegible] 12 13 III

18 19 [illegible]

PREPARED BY:

15 [illegible]

Delta Information Systems, Inc
259 Wyncote Road
Jenkintown, PA 19046

APPROVED FOR PUBLICATION:

Marshall L. Cain

MARSHALL L. CAIN
Assistant Manager
Office of NCS Technology
and Standards

FOREWORD

Among the responsibilities assigned to the Office of the Manager, National Communications System, is the management of the Federal Telecommunication Standards Program which is an element of the overall GSA Federal Standardization Program. Under this program, the NCS, with the assistance of the Federal Telecommunication Standards Committee, identifies, develops, and coordinates proposed Federal Standards which either contribute to the interoperability of functionally similar Federal telecommunication systems or to the achievement of a compatible and efficient interface between computer and telecommunication systems. In developing and coordinating these standards a considerable amount of effort is expended in initiating and pursuing joint standards development efforts with appropriate technical committees of the Electronic Industries Association, the American National Standards Institute, the International Organization for Standardization, and the International Telegraph and Telephone Consultative Committee of the International Telecommunication Union. This Technical Information Bulletin (TIB), one of a series, is a companion document to NCS TIB 80-2 and has been prepared to inform interested Federal activities of the progress of these efforts. Any comments, inputs, or statements of requirements which could assist in the advancement of this work are welcome and should be addressed to:

Office of the Manager
National Communications System
ATTN: NCS-TS
Washington, D.C. 20305
(202) 692-2124

411 314

USE OF A MICROPROCESSOR TO IMPLEMENT AN ADCCP PROTOCOL (FEDERAL STD. 1003)

July, 1980

FINAL REPORT

Submitted to:

NATIONAL COMMUNICATIONS SYSTEMS
8th & S. Courthouse Rd.
Arlington, Virginia 22204

Contracting Agency:

DEFENSE COMMUNICATIONS AGENCY

Purchase Order: DCA 100-79-C-0050

Submitted by:

DELTA INFORMATION SYSTEMS, INC.

259 WYNCOTE ROAD

JENKINTOWN, PENNSYLVANIA 19046

USE OF A MICROPROCESSOR TO IMPLEMENT AN ADCCP PROTOCOL (FEDERAL STD. 1003)

1.0 INTRODUCTION 1-1

2.0 SYSTEM DESIGN CONSIDERATIONS 2-1

3.0 FUNCTIONAL FLOW CHARTS FOR UNBALANCED, NORMAL CLASS 3-1

4.0 DETAILED SOFTWARE DESIGN 4-1

4.1 OPERATING SYSTEM 4-2

4.2 DATA STRUCTURES 4-5

4.3 SUBROUTINE/FUNCTION DESCRIPTIONS 4-12

4.4 DETAILED FLOW CHARTS 4-15

5.0 MICROPROCESSOR CODING AND TESTING 5-1

6.0 DISCUSSION OF FEASIBILITY 6-1

6.1 MEMORY REQUIREMENTS 6-1

6.2 EXECUTION TIME 6-2

APPENDICES

A. SYNCHRONOUS PROTOCOL COMMUNICATIONS CONTROLLER--F6856 . . . . . A-1

B. LSI MICRO PACKET NETWORK INTERFACE--WD2501 . . . . B-1

1.0 INTRODUCTION

This document summarizes the work performed by Delta Information Systems, Inc. for the Office of Technology and Standards of the National Communications System, an organization of the U. S. Government, under Purchase Order DCA 100-79-C-0050. The Office of Technology and Standards, headed by National Communications System Assistant Manager Marshall L. Cain, is responsible for the management of the Federal Telecommunications Standards Program, which develops telecommunication standards whose use is mandatory by all Federal agencies. The objective of this program is to develop a block diagram, flow charts, and computer programming for the unbalanced normal, unbalanced asynchronous, and balanced asynchronous class of procedures in accordance with Federal Standard 1003. The purpose of this effort is to determine the feasibility of using the M6800 or similar microprocessor to implement this type of protocol, and to obtain an estimate of memory and processor resources that would be required. The Office of Technology and Standards will use the information to advise other Federal agencies who implement the standard and, when merged with the results of other studies, to evaluate the operational and economic impact of incorporating various options in Federal Standard 1003.

The effort necessarily has focussed on the software required to implement the protocol itself, and is by no means a total hardware/software system design that would be required to develop a complete system. Complete system development is, of course, beyond the scope of this program. However, there are at least two system design factors that may have a significant effect on system performance and on memory and processor resources that are required. These design factors include the type of LSI interface chip employed, and the implementation of the operating system required to control the concurrent software processes that make up the protocol. These factors are discussed in more detail in Section 2.0 along with a discussion of the block diagram of the overall system design.

Flow charts describing the software that makes up the protocol are included in Sections 3.0 and 4.0. The functional flow charts in Section 3.0 describe the protocol operations for the unbalanced normal class at the highest level and are largely independent of the hardware configuration. The detailed flow charts in Section 4.0 describe the protocol software processes in sufficient detail that code may be written with no major design decisions. These flow charts at this level are very hardware dependent.

A small portion of the code for the 6800 microprocessor has been written and is included in Section 5.0 The code was introduced into a 6800 microcomputer, provided by Delta Information Systems. The code in the computer was then tested to insure its validity. Finally Section 6.0 contains a discussion of the feasibility of using the 6800 to implement the ADCCP protocol. It is estimated that approximately 850 instructions are needed to implement the three classes of procedures in a logically configurable station (primary, secondary, or combined), with no optional functions, and that approximately 250 instructions are required for the operating system. Data transmission rates of up to 19.2 kilobit/sec. appear feasible for the configuration being considered.

2.0 SYSTEM DESIGN CONSIDERATIONS

The block diagram in Figure 2-1 shows a link with one primary/combined and one secondary/combined station communicating with each other by sending information in both directions. That is, either station may be a source or sink of data or both. Two-way simultaneous transmission is assumed. Although many secondary stations may communicate with one primary station, the objectives of this program can be met with no loss of generality, by assuming the existence of only one secondary station.

Each station, primary, secondary, or combined is made up of a microcomputer, an LSI interface to the link, and a user which supplies and uses the data to be communicated. The primary and secondary stations are physically very similar; operationally, of course, the primary must supervise and control a number of secondary stations, and thus it requires a larger data structure and somewhat more complicated code.

For the purpose of this program, the microcomputer can be assumed to be very basic--microprocessor, memory (RAM and ROM), interface chips, clock, etc. A design choice that has significant impact on the outcome of this program is the choice of the LSI interface. The purpose of the LSI interface is to convert the parallel data from the CPU to a continuous serial data stream for transmission. Simultaneously, it must convert received serial data to parallel data for the CPU. In addition, it must generate and verify the frame check sequence (FCS), stuff and delete 0's to distinguish FLAG or ABORT from data, insert and detect FLAG or ABORT, and insert interframe fill or idle link fill. Other functions may also be performed by

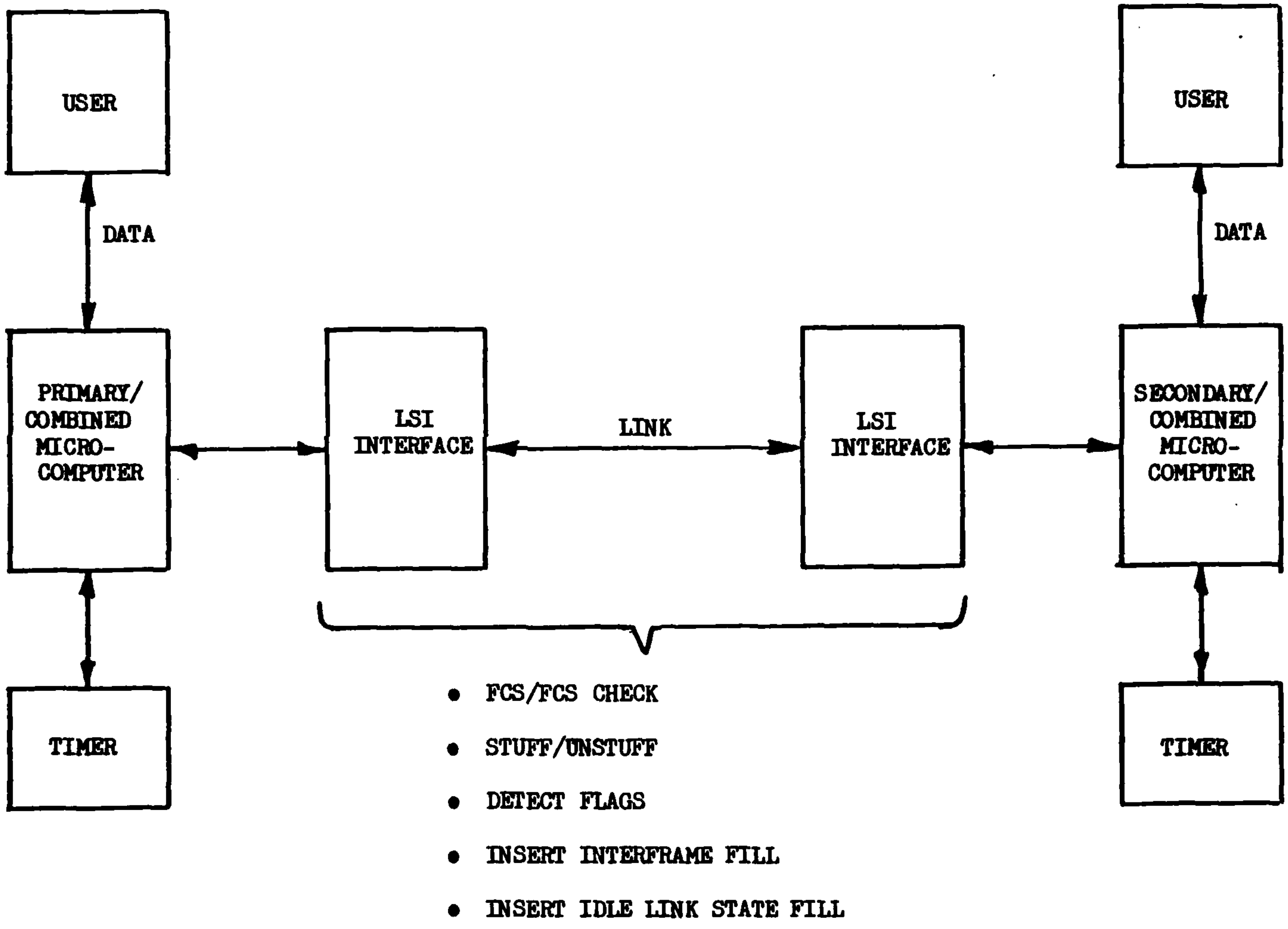


Figure 2-1 System Block Diagram

this interface.

Two different LSI chip specifications have been examined as possible candidates for the interface function in this particular study. These chips, which represent different approaches to the interface problem, are the F6856 and the WD2501. (Refer to Appendix A and Appendix B for a copy of their preliminary data sheets.)

The WD2501 interface chip controls frames of data by means of a direct memory access (DMA) technique, automatically transmitting/receiving flag, address, and control fields. Automatic retransmission of frames due to errors is also accomplished. The chip appears to be capable of implementing the unbalanced asynchronous class of procedures by itself.

The F6856 interface chip, on the other hand, controls bytes of data, in addition to performing the required function described above. This chip is capable of accommodating virtually all of the classes of procedures described in the standard, with the possible exception of the 32-bit frame check sequence. Since the chip sends and receives bytes of data, most of the processing must be done in the microcomputer.

The F6856 chip was selected for this program by mutual consent of the contractor and the government. The interface to the communications line requires additional logic such as a Federal Standard 1031 (Electronic Industries Association Recommended Standard 449) interface chip and a modem, but the choice of these has little impact on this program.

The data transmitted over the link must also be transmitted to the user. The interface/protocol required between the microcomputer and the user is also part of the system design. However, for this program, the protocol has not been defined. The interface, including the buffers to hold the data, is defined and described in Section 4.0.

3.0 FUNCTIONAL FLOW CHARTS FOR UNBALANCED, NORMAL CLASS

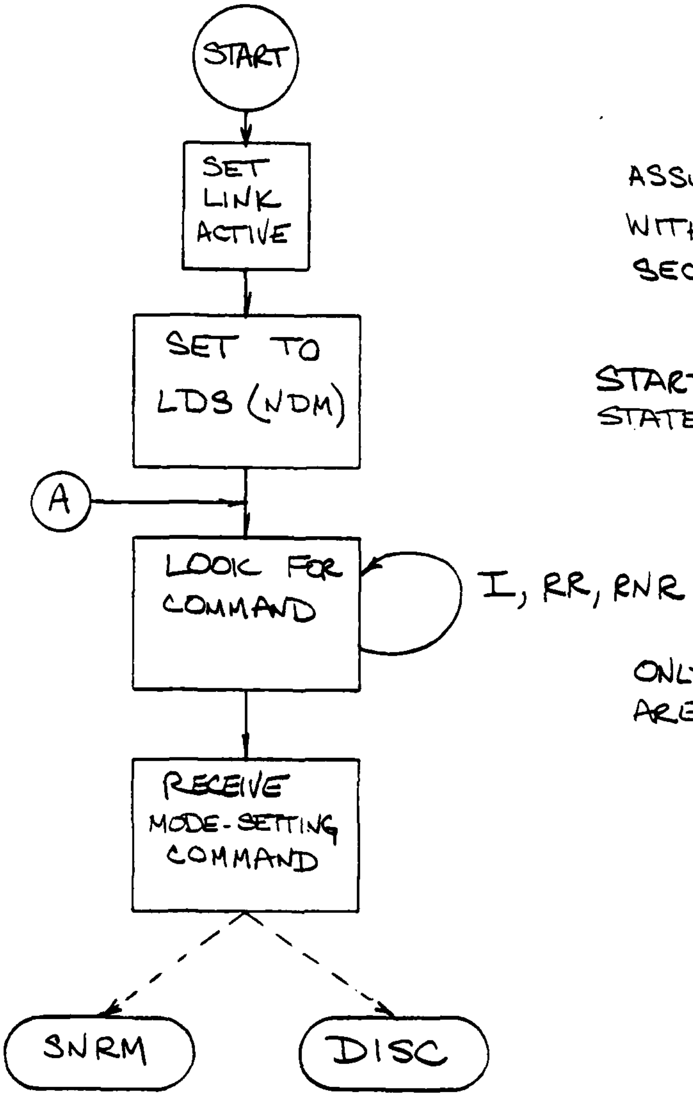
The functional flow charts describe the protocol operation at the highest level. That is, the frame is considered to be an entity, transmitted and received in "one piece." Operation is described in terms of gross system states and major parameters. The flow charts at this level are largely independent of the hardware configuration, and time constraints required for simultaneous two-way operation do not appear.

As an introduction to the detailed flow charts presented in Section 4, functional flow charts for the unbalanced normal, basic repetoire, class of procedures (Figures 3-1 through 3-9) are presented in this section. Since the functional flow charts for the three classes of procedure addressed in this report are very similar at this level of detail it is not necessary to include them here. Instead, the differences are more clearly shown in the detailed flow charts for the station, which may be logically configured to be a primary, secondary, a combined station operating, as appropriate, in normal respond mode, asynchronous respond mode, or asynchronous balanced mode.

The unbalanced normal class, basic repetoire must accommodate five received commands in the secondary station:

I	-	Information
RR	-	Receive Ready
RNR	-	Receive Not Ready
SNRM	-	Set Normal Response Mode
DISC	-	Disconnect

SECONDARY STATION – UNBALANCED NORMAL



ASSUME TWO-WAY SIMULTANEOUS WITH ONE PRIMARY AND ONE SECONDARY

START IN LOGICALLY DISCONNECTED STATE (NORMAL DISCONNECTED MODE)

ONLY MODE-SETTING COMMANDS ARE ACCEPTED

NOTE:

START ENTERED ON THE FOLLOWING CONDITIONS:

1) POWER ON
2) POWER-FAIL RECOVERY
3) MANUAL RESET
4) SWITCH FROM 'LOCAL' TO 'LINK' OPERATION

Figure 3-1 Functional Flow Chart A

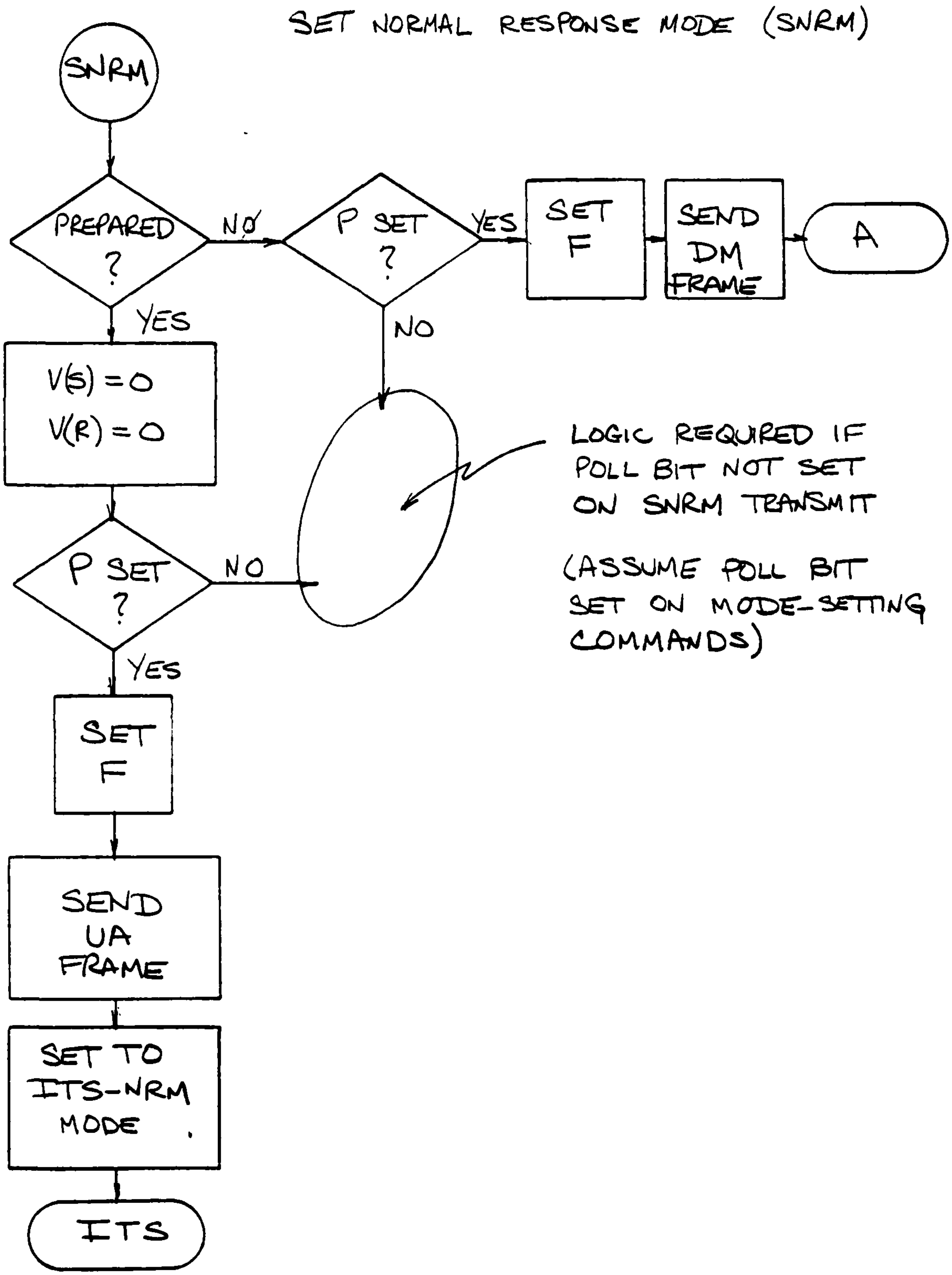

SET NORMAL RESPONSE MODE (SNRM)
SNRM
PREPARED ?
NO
P SET ?
YES
SET F
SEND DM FRAME
A
NO
YES
V(S) = 0
V(R) = 0
LOGIC REQUIRED IF POLL BIT NOT SET ON SNRM TRANSMIT
(ASSUME POLL BIT SET ON MODE-SETTING COMMANDS)
P SET ?
NO
YES
SET F
SEND UA FRAME
SET TO ITS-NRM MODE
ITS

Figure 3-2 Functional Flow Chart B

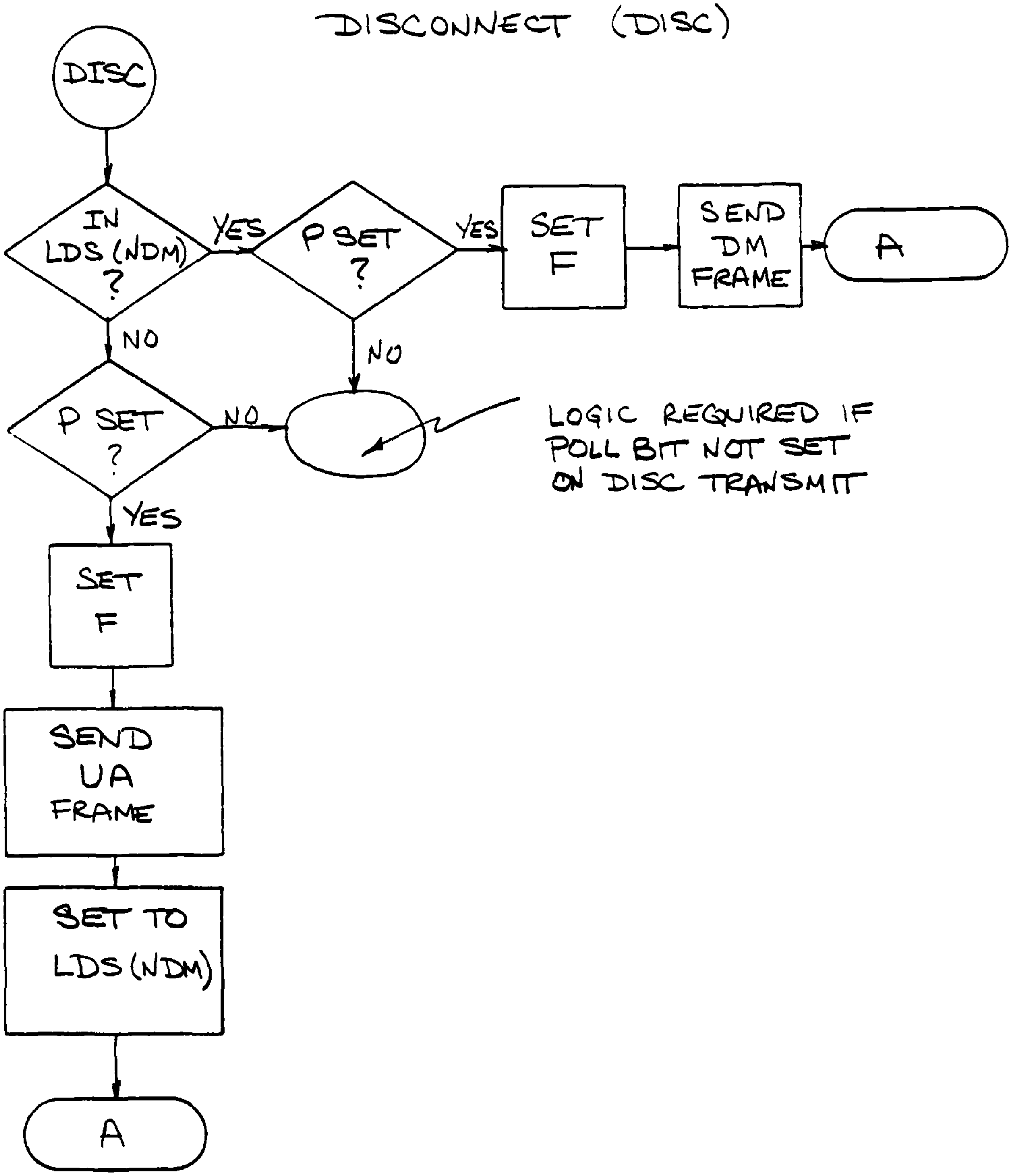


Figure 3-3 Functional Flow Chart C

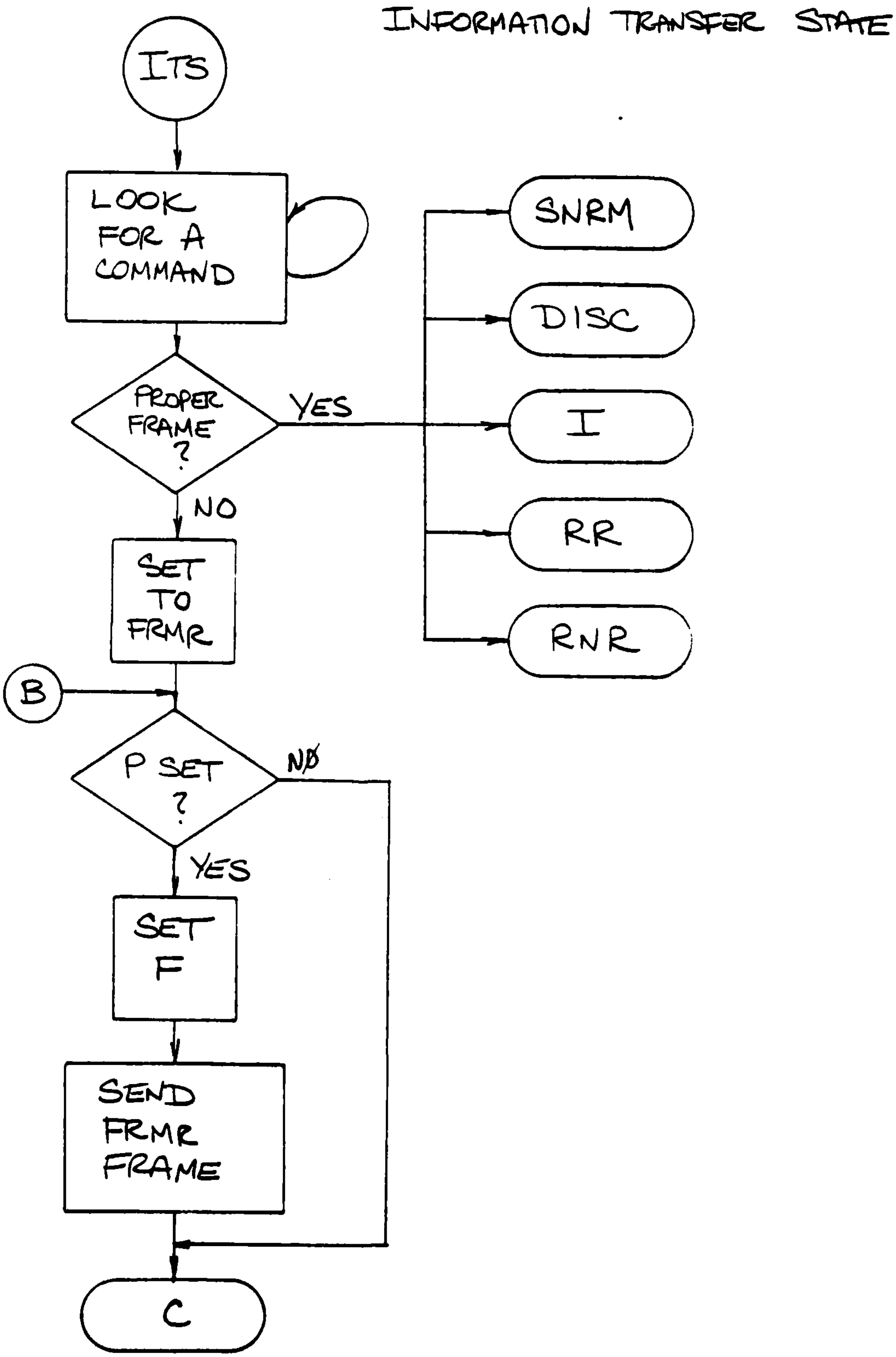


Figure 3-4 Functional Flow Chart D

FRAME REJECT

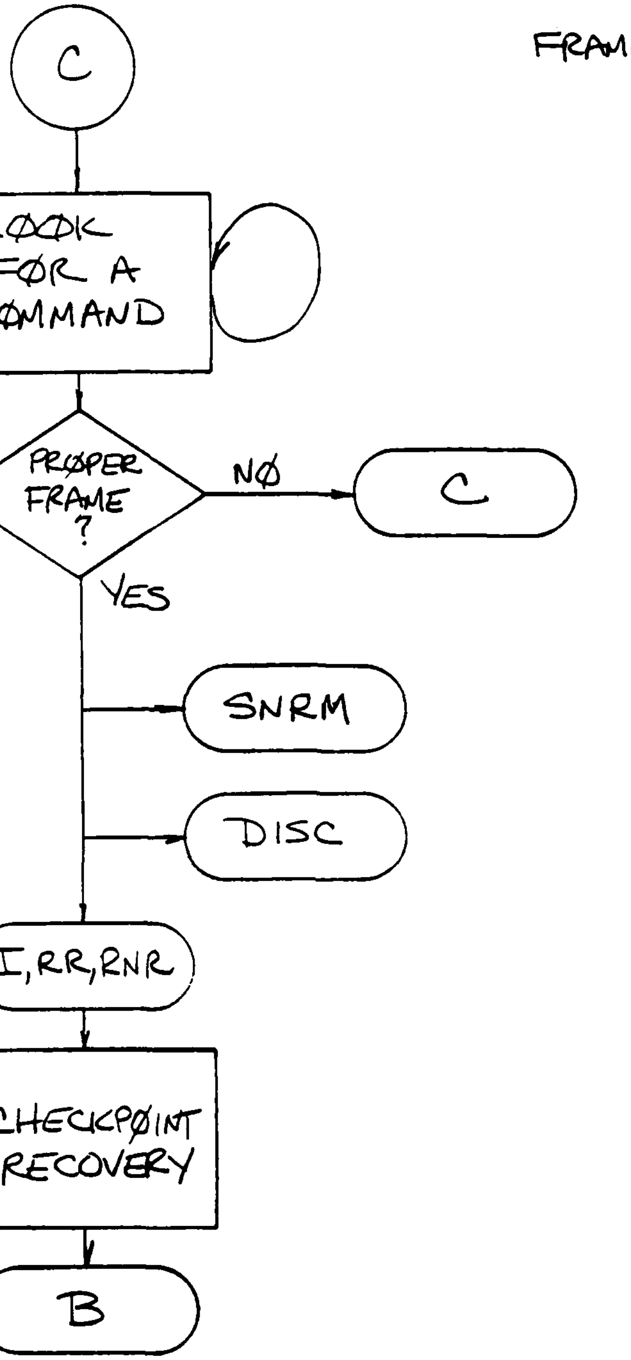


Figure 3-5 Functional Flow Chart E

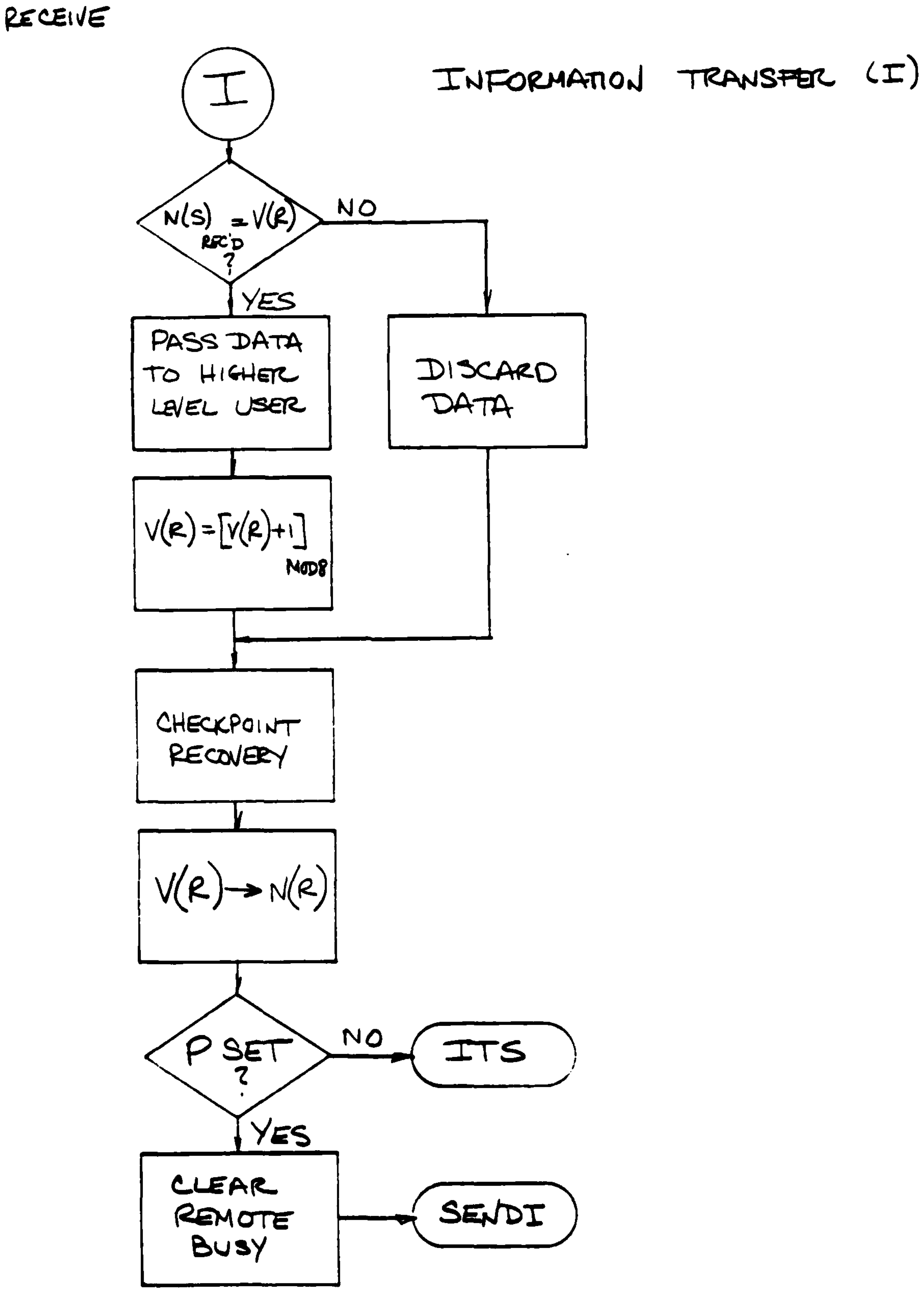


Figure 3-6 Functional Flow Chart F

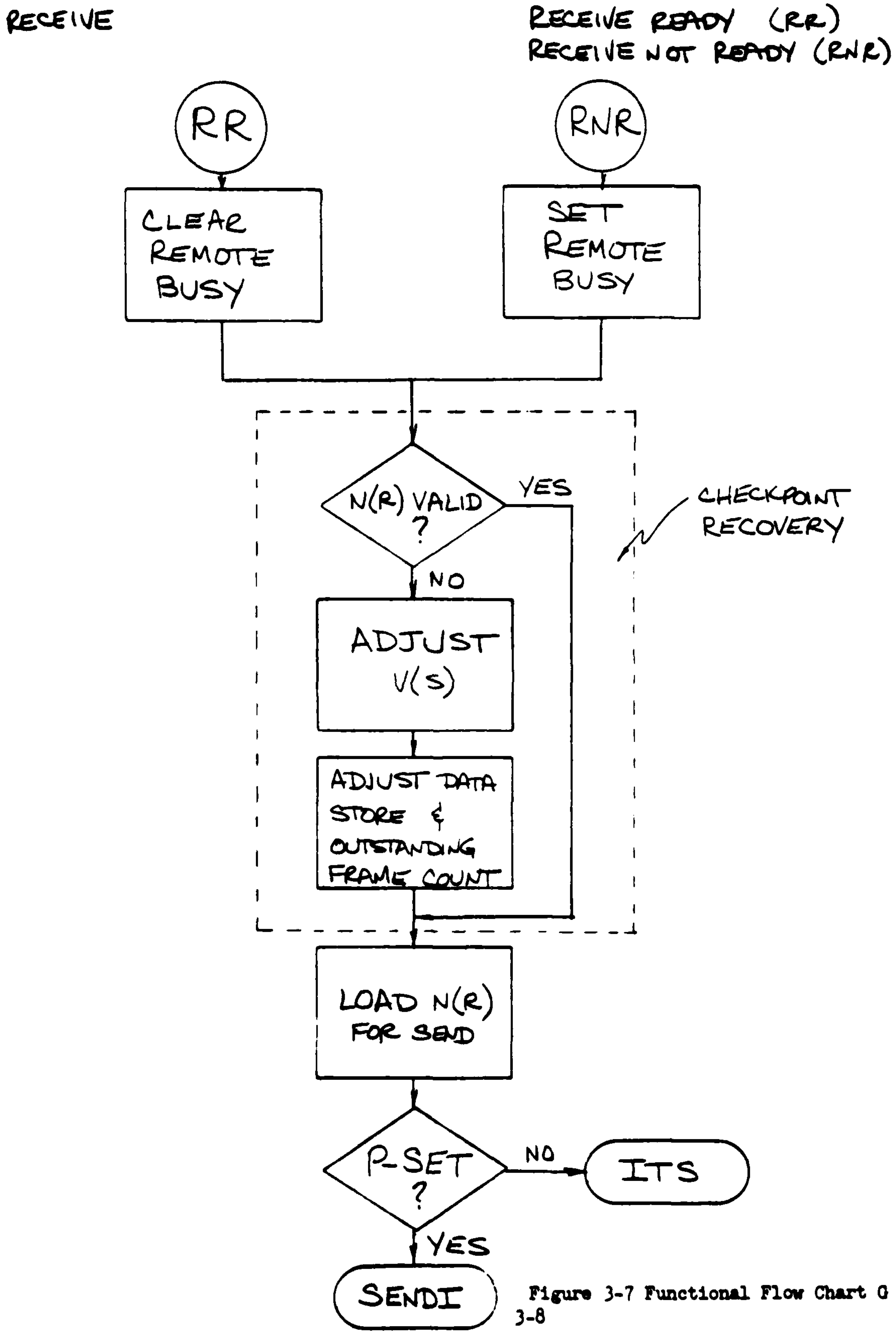


Figure 3-7 Functional Flow Chart G

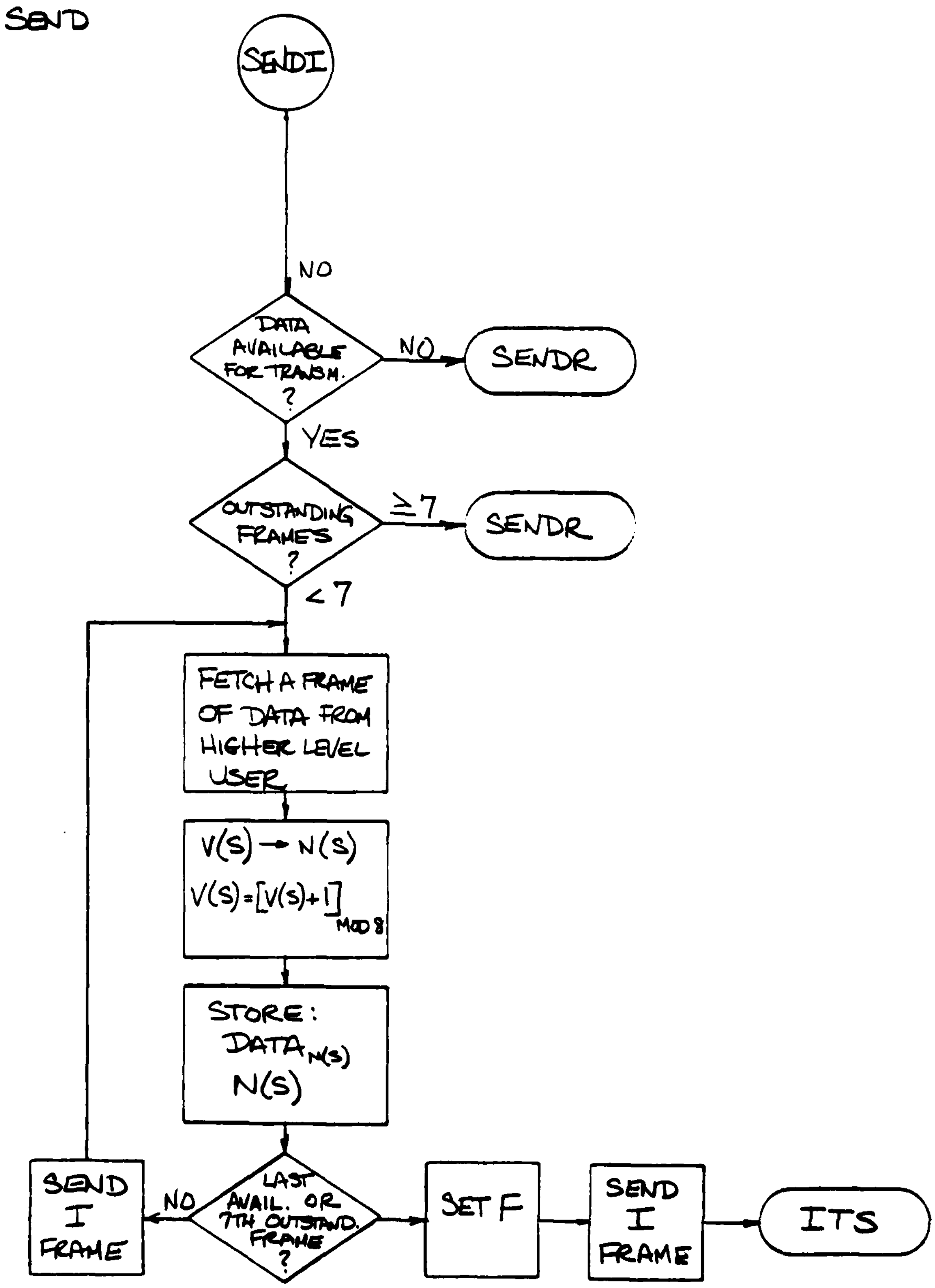


Figure 3-8 Functional Flow Chart H

SEND

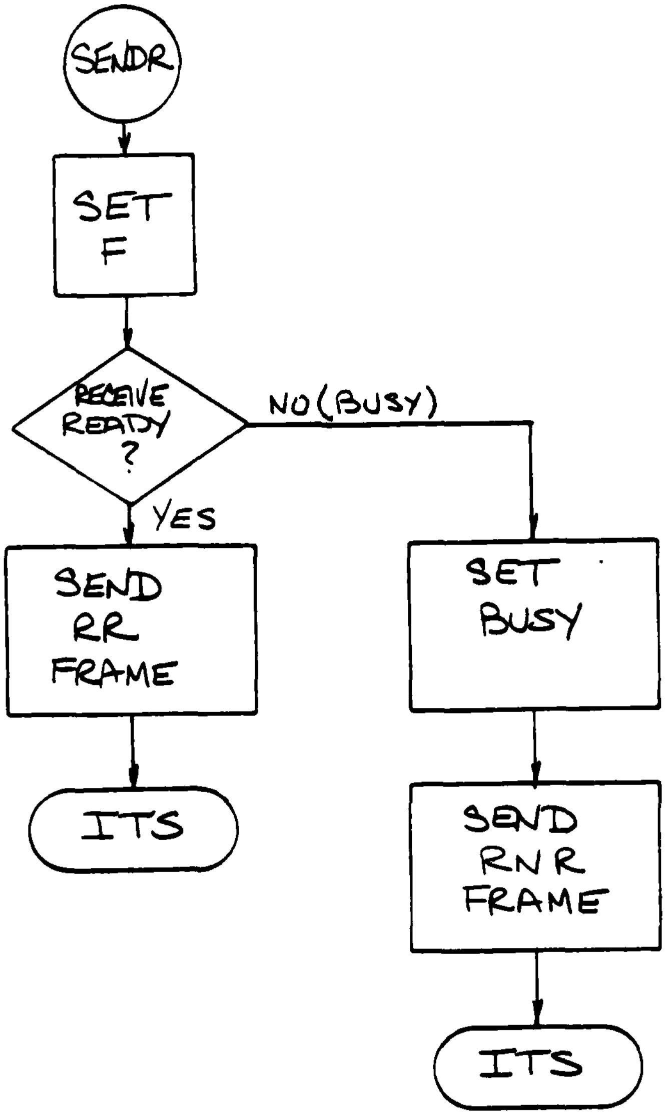


Figure 3-9 Functional Flow Chart I

The secondary station may transmit six responses:

I	- Information
RR	- Receive Ready
RNR	- Receive Not Ready
UA	- Unnumbered Acknowledgement
DM	- Disconnected Mode
FRMR	- Frame Reject

The secondary station operates in one of three major states which are mutually exclusive:

(1) LDS (NDM) - Logically disconnected state (normal disconnected mode)

(2) ITS (NRM) - Information transfer state (normal respond mode)

(3) FRMR - Frame reject state

Other major variables required by the secondary station are:

REMOTE BUSY - true if RNR received;
false if RR received or P-bit set

RECEIVER BUSY - true if not prepared to receive information;
false otherwise

P-BIT - Poll bit

F-BIT - Final bit

V(S) - Send Variable (next I-frame to be transmitted)

V(R) - Receive Variable (expected sequence number of next received I-frame)

N(S) - Send sequence number (I-frame sequence number)

N(R) - Receive sequence number (station transmitting N(R) has correctly received all I-frames up to and including N(R)-1)

The functional flow charts are described briefly in the following paragraphs. Refer once again to Figures 3-1 through 3-9. On start-up, the secondary station enters the logically disconnected state. In this state, only mode-setting commands are accepted by the station. If the statio is ready to accept commands, it responds to an SNRM command with a UA response frame and enters the information transfer state (ITS). If not ready, it sends a DM frame. The response to a DISC command is similar.

Upon entering the ITS, the receiver looks for one cf the five valid commands. If a valid command is received, appropriate action is taken. If an invalid command is received, the frame reject (FRMR) state is entere and the cause of the rejected frame is reported to the primary station via the FRMR response. The only way to recover from the frame reject state is to receive an SNRM or DISC command. In frame reject state, the I, RR, and RNR commands are monitored to perform checkpoint recover only; that is, the received N(R) is monitored to verify those frames that have been received correctly by the primary station.

If a valid I-frame is received, the N(S) is checked, and if valid, the data is passed to the user; if not valid, checkpoint recovery is performed and, if the poll bit is set, the secondary station may transmit I-frames if available. If not available, the station responds with RR or RNR in response to the poll and proceeds to monitor incoming frames. If a valid RR or RNR frame is received, checkpoint recover is performed.

4.0 DETAILED SOFTWARE DESIGN

In this section the software design is presented in sufficient detail to allow the objectives of this program to be met: that is, the feasibility of using the 6800 and an estimate of memory and throughput can be obtained. The design covers the major aspects of a logically configurable station; functions that allow the station to operate as a primary, secondary, a combined station in unbalanced normal, unbalanced asynchronous or balanced modes are included. Operation as a secondary/combined station is emphasized, since FED-STD-1003 does not cover many of the primary/combined station procedures for managing the link. These are left to the system designer. The software design includes a description of a minimal operating system to handle concurrent processes, major data structures, major software routines, and a set of detailed flow charts.

The detailed flow chart, together with associated data structures, describes the protocol software processes in sufficient detail so that code may be generated with no major design decisions. The flow chart at this level is hardware dependent, and must take into consideration the time constraints imposed by the concurrent software processes associated with the implementation of the protocol. That is, frames are not really transmitted and received in "one piece" in two-way simultaneous operation, but are transmitted and received a character at a time concurrently.

The protocol is made up of four major concurrent software processes, each of which is an example of the classic producer/consumer problem. In this problem, one process produces items and then deposits them into a buffer. A second process consumes the items by taking them from the buffer. The processes must be coordinated so that the consumer does not run ahead of the producer, and that the producer does not write over records before the consumer has had a chance to read them. For the protocol problem, two concurrent processes are involved in communicating data between the LSI interface and the microprocessor; the LSI chip deposits bytes in its buffer as the producer, and the MPU reads this data as the consumer. Conversely, the microprocessor writes data into a buffer as the producer, to be read by the LSI chip as the consumer and transmitted over the link. A similar pair of processes serves to implement the interface between the microprocessor and the higher level user. For this effort, emphasis is placed on the interface between the MPU and the LSI protocol chip. This requires two main processes to be running at the same time--transmit and receive. The operating system that manages these processes is presented in the following paragraphs. Although it is explained in terms of the transmit and receive processes, the approach is general and any number of processes can be added.

4.1 Operating System

The design of the operating system (OS) is important because it can have a significant impact on the time required to switch the processor among concurrent processes and to handle interrupts. The approach taken makes use of the "standard" WAIT and SIGNAL primitives together with event variables. No attempt has been made to design a complete operating system; only those routines required to handle the concurrent processes are included.

Each software process is defined to be in one of three states:

ACTIVE (RUNNING) STATE - Executing computer instructions

BLOCKED STATE - Waiting for the occurrence of an event in another process

READY STATE - Waiting for processor to run

State Transitions are illustrated in the state diagram of Figure 4-1.

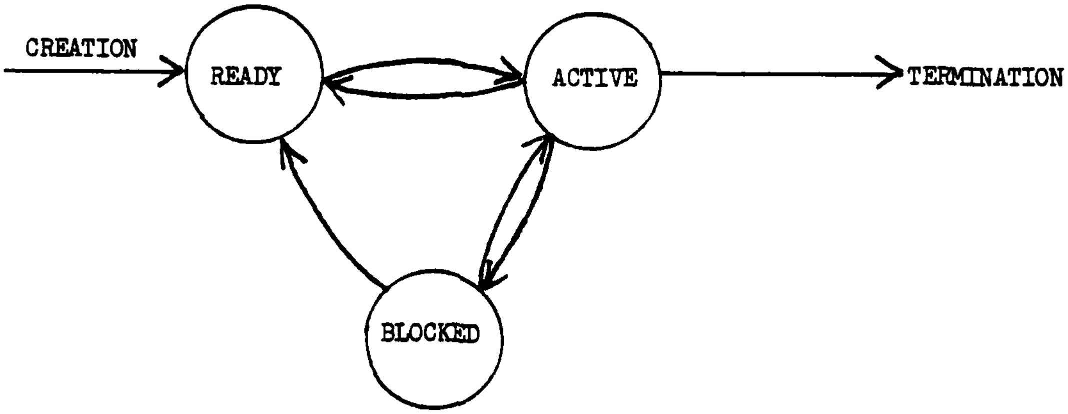


Figure 4-1 Process State Diagram

Processes may be created or terminated by other processes, and each process is defined to be either running or on the blocked queue or the ready queue depending on its state. Associated with each process is a process control block (PCB) which contains an area to save the CPU's registers when the process is interrupted, a pointer to the next PCB in the queue, and the process' priority.

If the process currently running becomes blocked, it is changed from the ACTIVE state to the BLOCKED state via the WAIT routine. For example, if the receive process is executing instructions and wishes to obtain a byte of data from the LSI interface chip buffer, the process tests the event variable RDAFLG to determine whether or not the byte of data is available. If available, the process continues; if not, the WAIT routine is called to save the receive process registers, insert the receive process into the blocked queue, get a new process from the ready queue, restore the registers of the new process, and run the new process.

The receive process continues where it left off after an interrupt from the LSI chip signals that a byte of data is available, the interrupt service routine services the interrupt, and the receive process is moved to the running state via the SIGNAL routine. The SIGNAL routine removes the receive process from the blocked queue and restores it to the running state. Any process is blocked and restored in this way by the WAIT and SIGNAL routines respectively, and by the appropriate interrupt service routine. Refer to Section 4.4 for the flow charts of the WAIT and SIGNAL routines.

The 6800 has but one interrupt input that is maskable. This means that unless some additional hardware is used, all interrupt lines must be logically ORed and fed to the CPU via the single interrupt input, $\overline{\text{IRQ}}$. This requires that the interrupt service routine poll the various devices that might cause an interrupt to determine what device actually did cause the interrupt. The order in which the devices are polled determines the interrupt priority. The following five events cause interrupts from the 6856 protocol chip:

RDA - Received data available

ROVR - Receiver overrun (data was not read from buffer before new byte was loaded)

TOR - Transmitter overrun

TBMT - Transmitter buffer empty

TUR - Transmitter underrun (data was not loaded in transmitter buffer in time to transmit)

In addition, two timers are assumed to be part of the design, one to provide a time slicing function to interrupt a running process periodically to give the CPU to a different process, and a time-out timer to to indicate an overdue response. These two functions may be provided by one timer and appropriate software or by two separate timers. Refer to Section 4.4 for the interrupt service routine flow charts.

4.2 Data Structures

This section outlines the data structures, including variables, arrays, buffers, etc. in order to more easily understand the detailed flow charts that follow and to estimate the amount of random access memory (RAM) that is required to implement the protocol. Main state variables are as follows:

STATION TYPE - PRIMARY, SECONDARY, OR COMBINED (mutually exclusive)

OPERATIONAL STATE - has 3 values, mutually exclusive:

LDS - Logically disconnected state

ITS - Information transfer state

FRMREJ - Frame reject state (for secondary/combined)

Logically disconnected state has two mutually exclusive modes:

NDM - normal disconnected mode

ADM - asynchronous disconnected mode

Information transfer state has three mutually exclusive modes:

NRM - normal respond mode

ARM - asynchronous respond mode

ABM - asynchronous balanced mode

Other major variables are:

REMOTE BUSY - true if RNR received
false if RR received or P/F bit set in
received I-frame

RECEIVER BUSY - true if not prepared to receive information;
false otherwise

P-BIT - Poll bit

F-BIT - Final bit

V(S) - Send Variable (next I-frame to be transmitted)

V(R) - Receive Variable (expected sequence number of next received I-frame)

N(S) - Send Sequence Number (I-frame sequence number)

N(R) - Receive Sequence Number (station transmitting N(R) has correctly received all I-frames up to and including N(R) - 1)

Operating System variables include:

RDAFLG - Receive data available event variable

TBMT - Transmitter buffer empty event variable

RUNNING
BLOCK } Pointers to beginning of each queue
READY

PCB - Each process control block contains:

condition code register
accumulator A
accumulator B
index register (upper and lower)
program counter (upper and lower)
pointer to next PCB
priority

A number of buffers are required for such things as the received control word, transmitted control word, frame type, etc. Next, consider the data buffer required to transmit/receive information between CPU and USER. Assume that a separate buffer is required for transmit and receive, and that each buffer can hold up to eight I-frames of data. These buffers are accessed via tables shown in Figure 4-2. Each frame to be transmitted via the LSI chip has a starting address for the data and a length in bytes of the data part of the frame. If the frame was transmitted with the poll/final bit set, this is recorded. The "acknowledge" variable is used to indicate whether a frame has been deposited by the USER for transmission, whether it has been transmitted, and finally, whether it has been acknowledged by the receiving station. In the example shown, six frames have been deposited by the USER for transmission, three have been transmitted ending with a final bit (SECONDARY - NRM), and the first frame has been acknowledged. Three frames remain to be transmitted. The receiver look-up table performs a similar function for data received from the LSI chip. Each frame is assembled byte-by-byte and the frame length is incremented. When the frame has been correctly received (valid FCS and N(S)) the frame is tagged as verified and may be read by the USER.

The buffers and associated variables required for LSI interface chip operation are shown in Figures 4-3, 4-4, and 4-5. The Mode Control Register (MCR) contains control information common to both receiver and transmitter. The SAR contains the secondary station address. The TCR is loaded by the MPU to control the transmitter, and the TDB contains the byte to be transmitted. The Receiver Status Register (RSR) is read by the MPU to determine the status of the byte received in the Receive Data Buffer (RDB). The RCR contains control information for the receiver and the TSR supplies transmitter status. Refer to Appendix A for a detailed

TPACK

FRAME NUMBER	STARTING ADDRESS	FRAME LENGTH	FINAL BIT SET	ACKNOWLEDGEMENT
0	ADDR0	1028	0	1
1	ADDR1	1028	0	0
2	ADDR2	512	F	0
3	ADDR3	512		-1
4	ADDR4	512		-1
5	ADDR5	512		-1
6				8
7				8

TRANSMITTER LOOK-UP TABLE

RPACK

FRAME NUMBER	STARTING ADDRESS	FRAME LENGTH	FRAME VERIFIED
0			
1			
2			
3			
4			
5			
6			
7			

RECEIVER LOOK-UP TABLE

Figure 4-2 Transmitter/Receiver Look-up Table

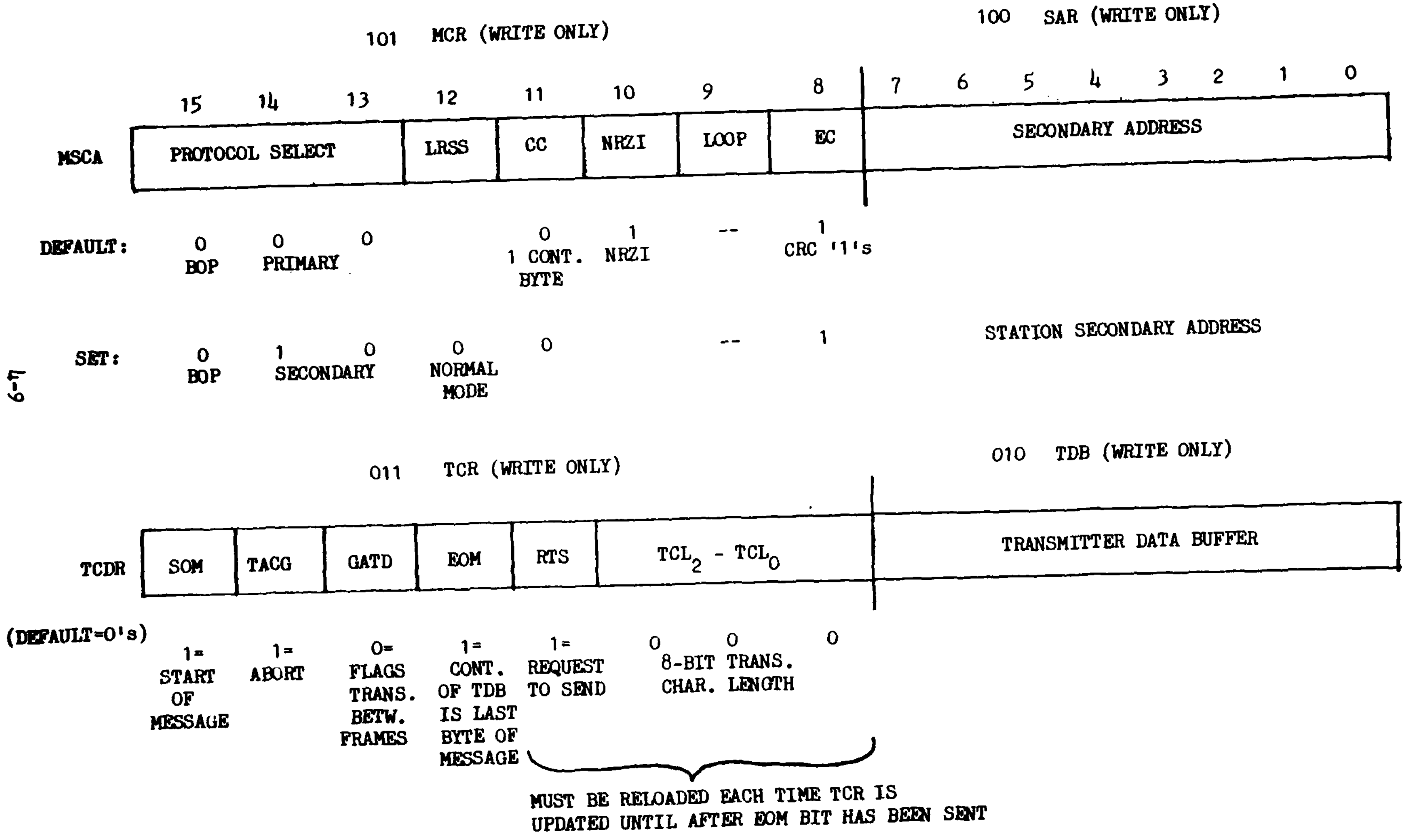


Figure 4-3 Mode Control, Secondary Address, Transmitter Control, and Tranmitter Data Registers

001 RSR (READ ONLY)

000 RDB (READ ONLY)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSDR	ROVR	RDA	REOM	ABGA	RDL_2 - RDL_0			RERR	RECEIVER DATA BUFFER							
	1 = RECEIVER OVERRUN	1 = RECEIVED DATA AVAILABLE	1 = REC'D FLAG OR ABORT (REC'D EOM)	1 = REC'D ABORT IF RERR = 1 = " GO AHEAD " " = 0	RECEIVER LAST CHARACTER LENGTH			1 = CRC ERROR (ASSERTED AT END OF FRAME)								

ALL BITS OF RSR EXCEPT RDA ARE RESET ON READ; RDA IS RESET WHEN RDB IS READ

DATA IS PASSED TO RDB ONLY ON ADDRESS MATCH

Figure 4-4 Receiver Status Register and Receive Data Buffer

	111 RCR (WRITE ONLY)								110 TSR (READ ONLY)							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RCTS (DEFAULT=0's)	DTR	MISC	NOT USED		CRC	RE	RCL_1 - RCL_0		TUR	TBMT	TOR	NOT USED		CTS	CD	DSR
	DATA TERMINAL READY = 1				CRC SELECTED = 1	RECEIVER ENABLE = 1	RECEIVER CHARACTER LENGTH 00 = 8 BITS		TRANSMITTER UNDERRUN = 1	TRANSMITTER BUFFER EMPTY = 1	TRANSMITTER OVERRUN = 1			CLEAR TO SEND = 1	CARRIER DETECT = 1	DATA SET READY = 1

TOR, TUR ARE RESET WHEN TSR IS READ

TBMT IS RESET WHEN TDB IS LOADED

Figure 4-5 Receiver Control Register, Transmitter Status Register

description of receiver/transmitter operation and flow charts for the F6856 LSI interface chip.

4.3 Subroutine/Function Descriptions

This section describes the functions or software modules that make up the protocol. Figure 4-6 contains a table of all the significant software modules organized by station type (primary, secondary, or common to both) and by whether the module is associated primarily with a transmit or receive process. Those modules contained in the operating system have been discussed previously. Some of the modules are the main routines for processes, namely INIT, RCV, SENDI, and IDLE. INIT is the initialization process, RCV is the process that receives and processes the address and control bytes of the received frame, SENDI is the main information transmitting process, and IDLE is a simple process that runs when all other processes are blocked. DUMMY is a process that is never run, but serves as a place for the IDLE process to point to and has the lowest priority. Those modules that are associated only with either the primary/combined or secondary/combined stations are mostly mode-setting or mode acknowledging functions and are relatively simple. The remaining routines are discussed in the following paragraphs.

RCNTRL - Extracts a provisional P/F bit, N(R) and N(S) if applicable from the control byte and determines the frame type by matching the masked control byte against a table of implemented commands and responses (11 all told). The frame type is then checked for validity against current mode and station type. Output is frame type if valid.

SECONDARY/COMBINED

RECEIVE FUNCTIONS	TRANSMIT FUNCTIONS
SNRM	
SARM	TR-UA
SABM	TR-DM
DISC	
RESET	
FRMRREJ	TR-FRMR

PRIMARY/COMBINED

RECEIVE FUNCTIONS	TRANSMIT FUNCTIONS
	TR-SNRM
UA	TR-SARM
DM	TR-SABM
	TR-DISC
FRMR	TR-RSET

BOTH

RECEIVE FUNCTIONS	TRANSMIT FUNCTIONS	OPERATING SYSTEM
RCV	SENDI	WAIT
I	SENDR	SIGNAL
RR		INTERRUPT SERVICE
RNR		IDLE
CHKPNT		DUMMY
RCNTRL		
GETBYT	SNDBYT	

INIT

Figure 4-6 Software Modules

CHKPNT - Performs checkpoint recovery. The received N(R) is checked to determine if N(R) points to an outstanding frame. If not, FRMR is called. If the poll (or final) bit is set, a check is made to determine if N(R) - 1 points to the frame transmitted with final (poll) bit set. If not, V(S) and appropriate elements of TPACK array are adjusted.

I - Reads Information frame. The information part of the frame is checked for length, message is stored in buffer. Send sequence number, N(S), is checked against receive variable, V(R). Checkpoint recovery is performed.

RR & RNR - Receives the Receive Ready, RR, or Receive Not Ready, RNR, command (response). Performs checkpoint recovery.

SENDI - Transmits information frame a byte at a time.

SENDR - Transmits RR or RNR as specified.

GETBYT - Macro that reads receive buffer if data is available and stores the data in a location specified in the macro argument. Interrupts are disabled for the duration of the macro to prevent flags or data from changing while reading. If no data is available, the process is blocked via WAIT.

SNDBYT - Macro that loads transmitter buffer. Similar operation to that in GETBYT.

FRMRIJ - Changes state of secondary/combined station to frame reject, assembles FRMR information field and activates FRMR transmit process.

TR-FRMR - FRMR transmit process terminates any other transmit process except TR-DM or TR-UA, sends the FRMR frame and terminates itself.

FRMR - Receives a FRMR frame (implies primary/combined station) and takes appropriate mode-setting action.

4.4 Detailed Flow Charts

The detailed flow charts are shown in Figures 4-7 through 4-24.

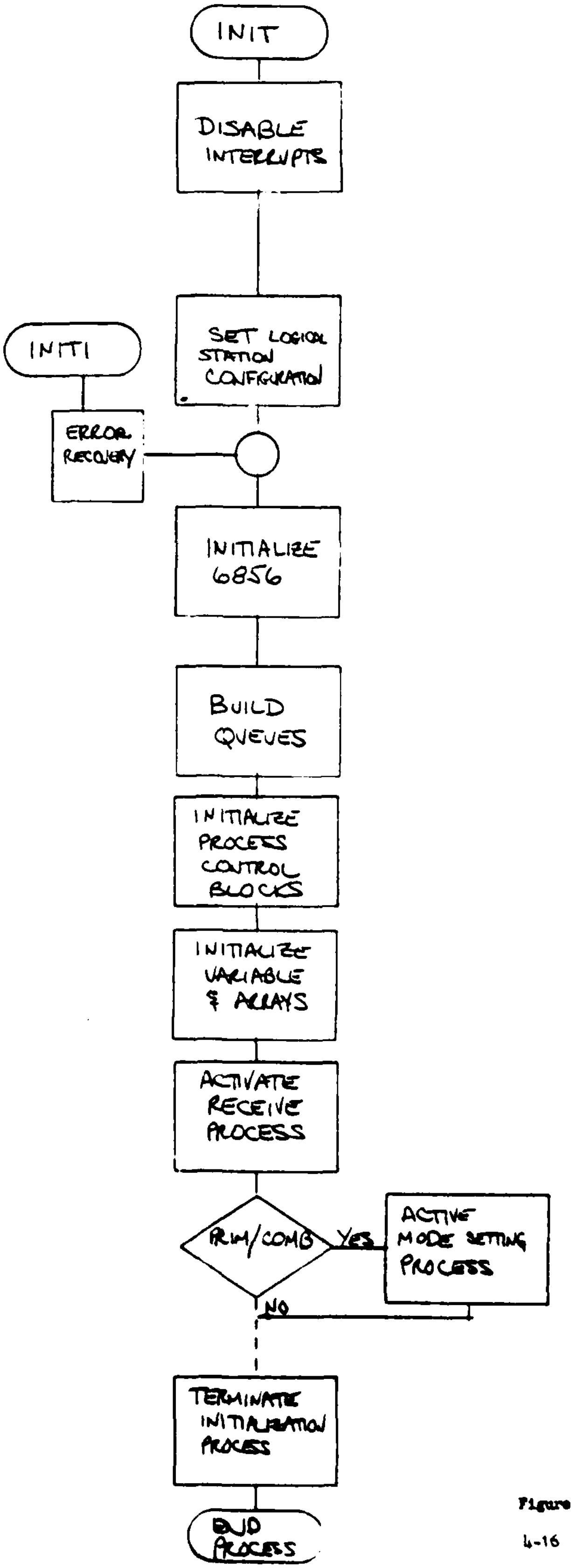
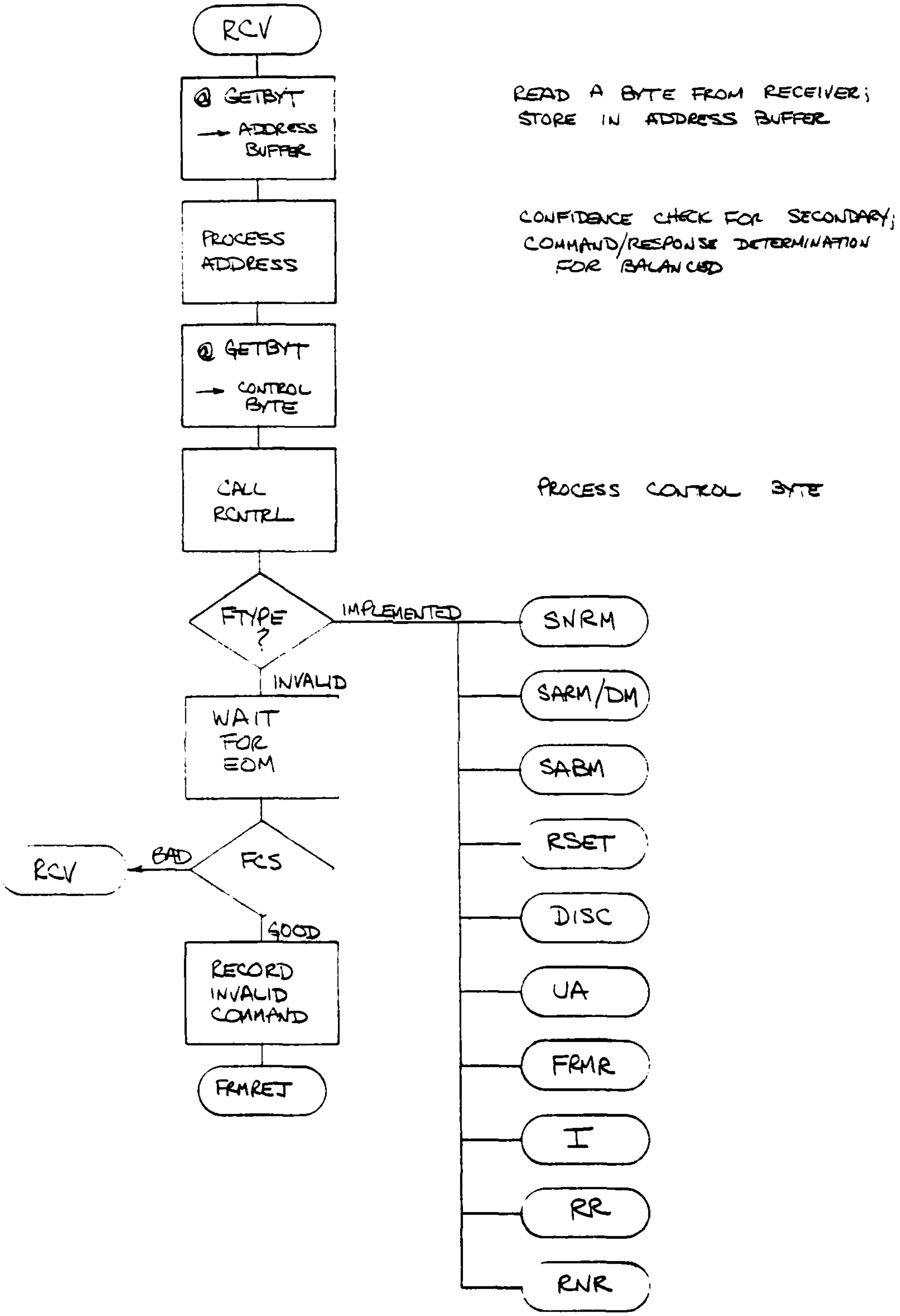

INIT
DISABLE INTERRUPTS
SET LOGICAL STATION CONFIGURATION
INITI
ERROR RECOVERY
INITIALIZE 6856
BUILD QUEUES
INITIALIZE PROCESS CONTROL BLOCKS
INITIALIZE VARIABLE & ARRAYS
ACTIVATE RECEIVE PROCESS
PRIM/COMB
YES
ACTIVE MODE SETTING PROCESS
NO
TERMINATE INITIALIZATION PROCESS
END PROCESS

Figure 4-7 Detailed Flow Chart A



READ A BYTE FROM RECEIVER; STORE IN ADDRESS BUFFER

CONFIDENCE CHECK FOR SECONDARY; COMMAND/RESPONSE DETERMINATION FOR BALANCED

PROCESS CONTROL BYTE

Figure 4-8 Detailed Flow Chart B

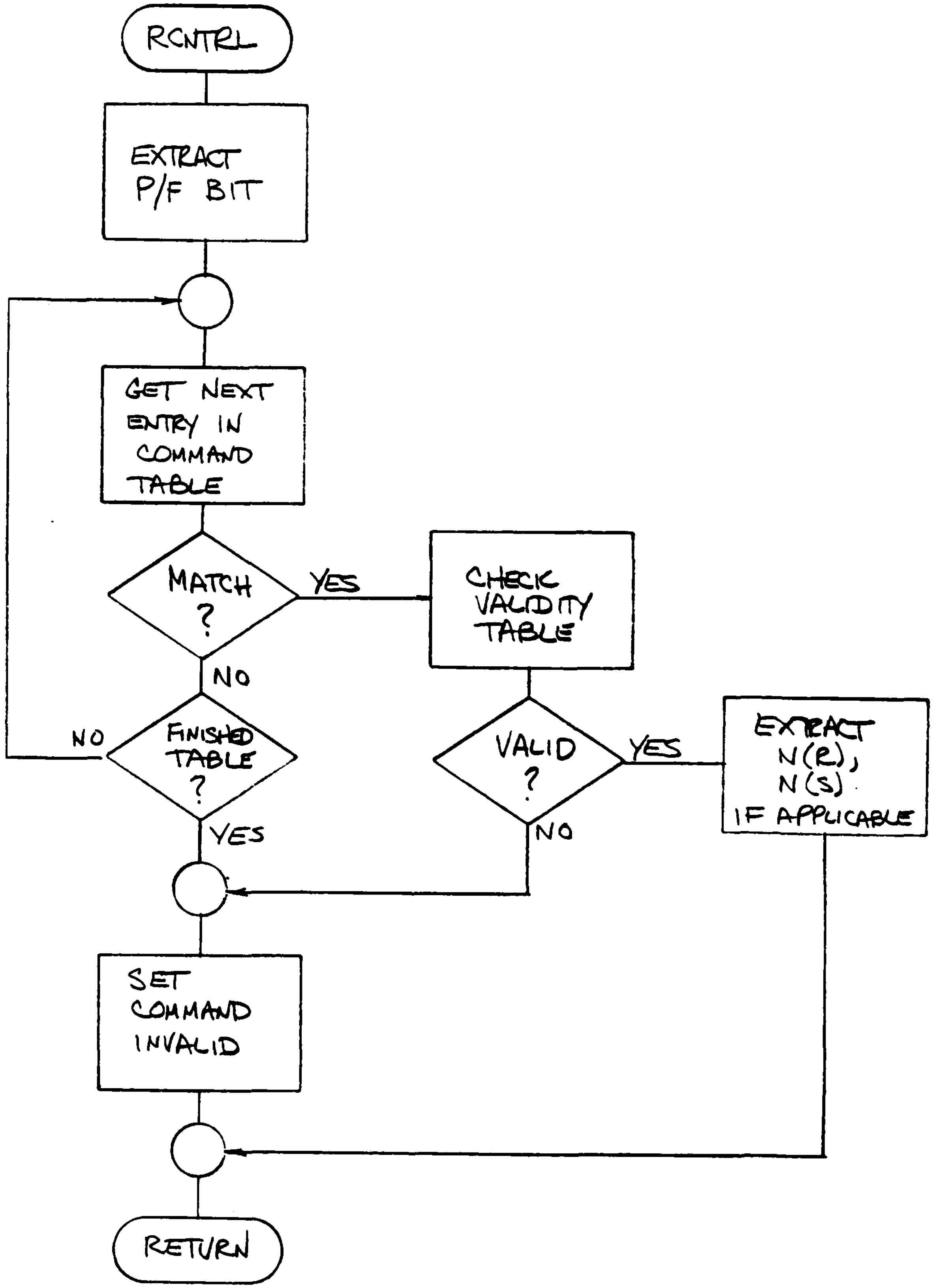


Figure 4-9 Detailed Flow Chart C

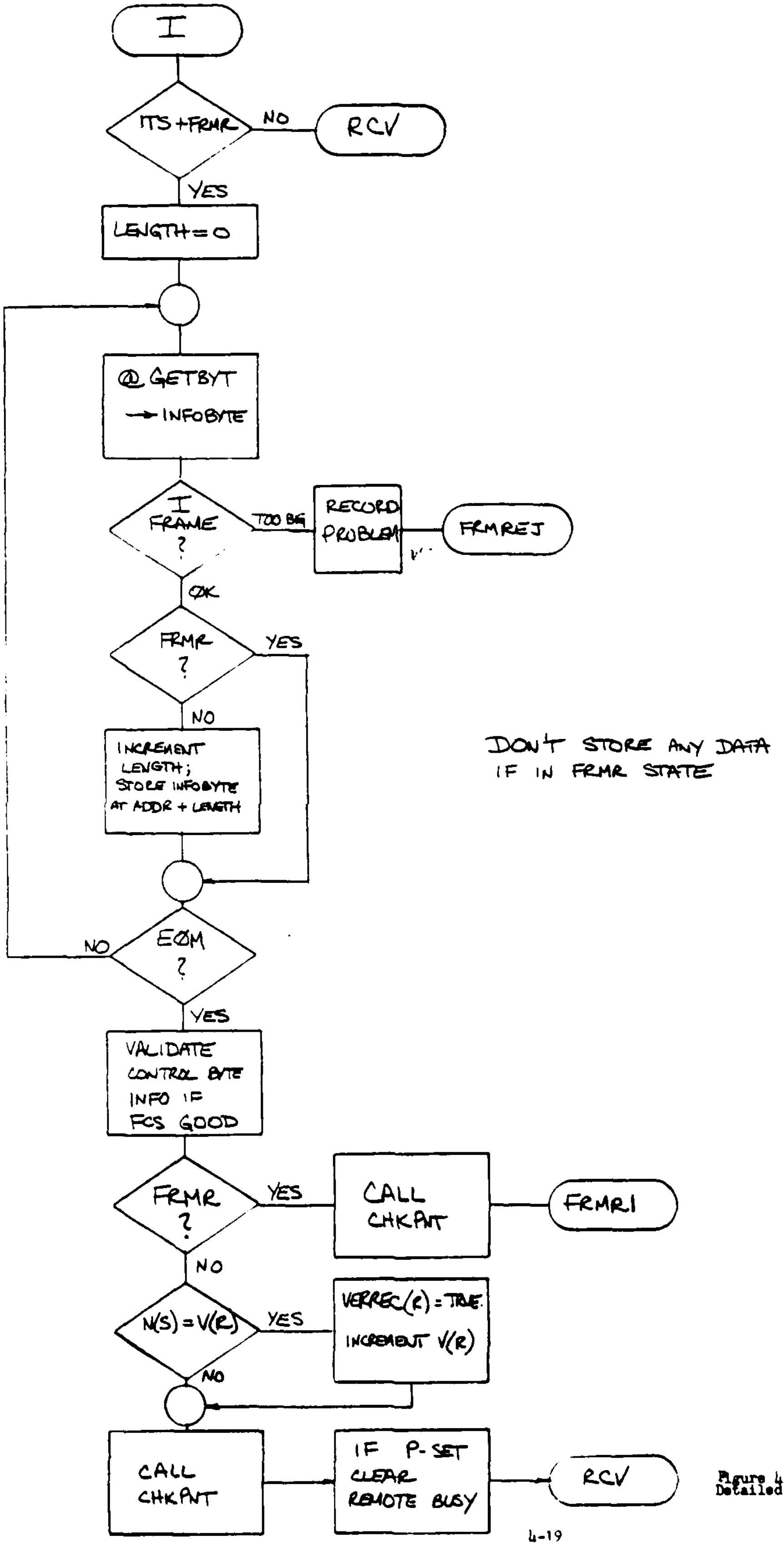
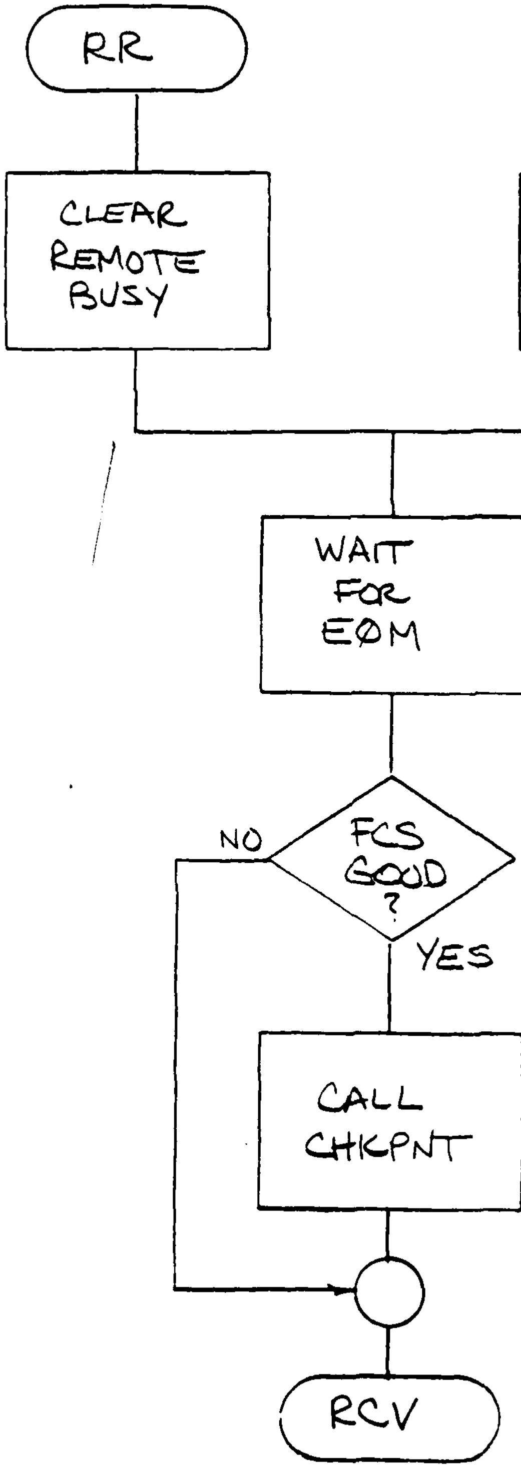


Figure 4-10
Detailed Flow Chart D



DON'T PROCEED UNLESS FCS IS GOOD

Figure 4-11 Detailed Flow Chart E

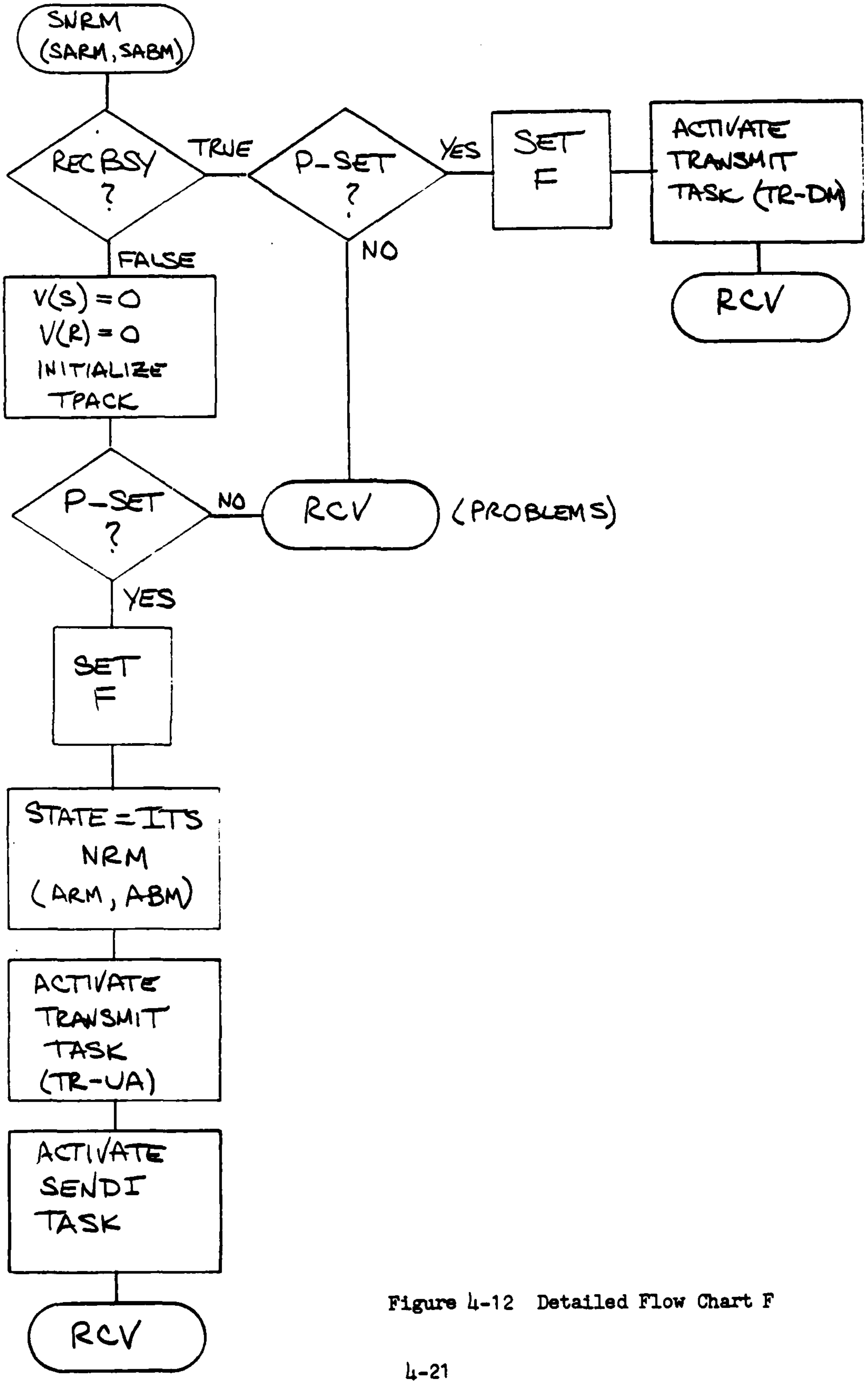


Figure 4-12 Detailed Flow Chart F

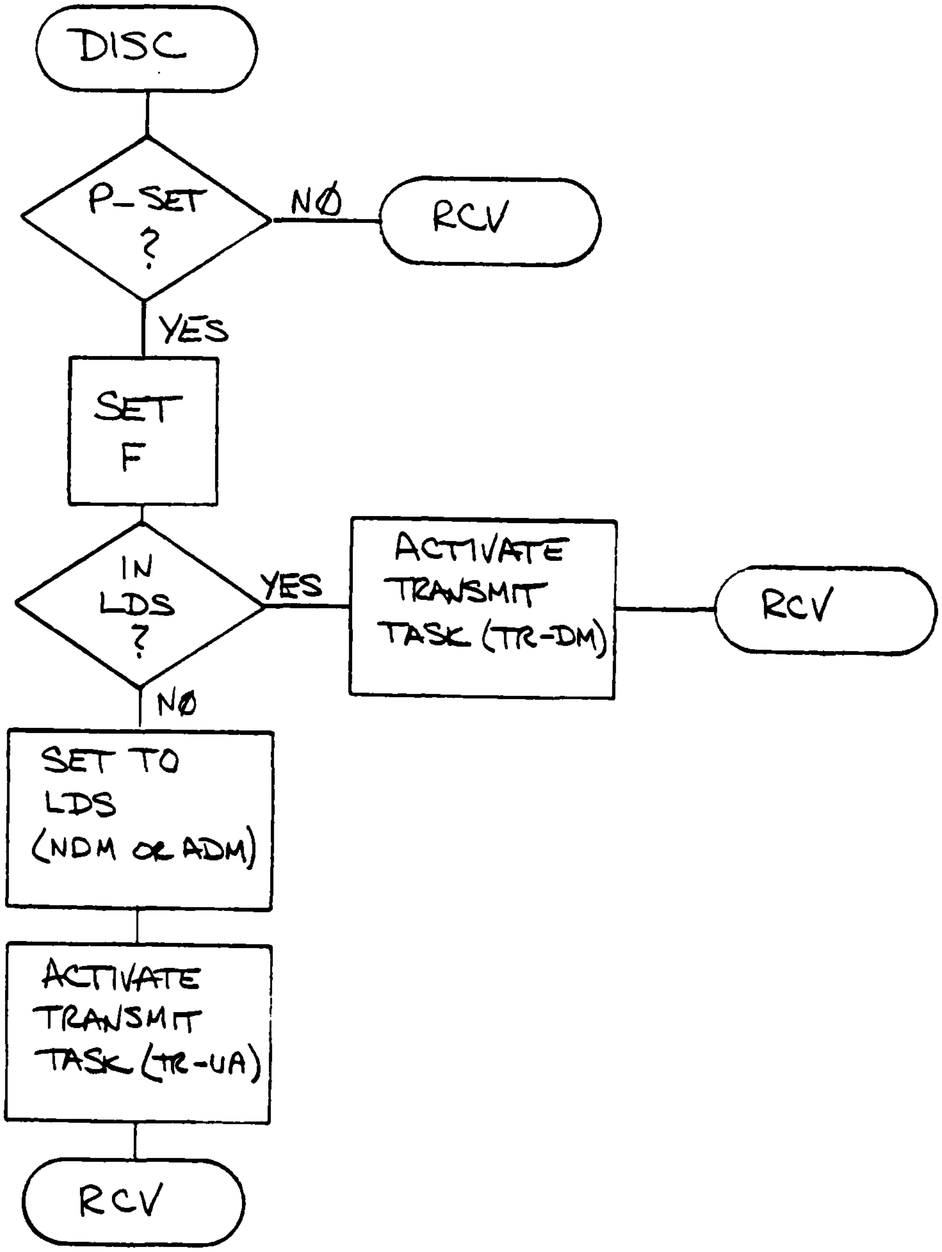
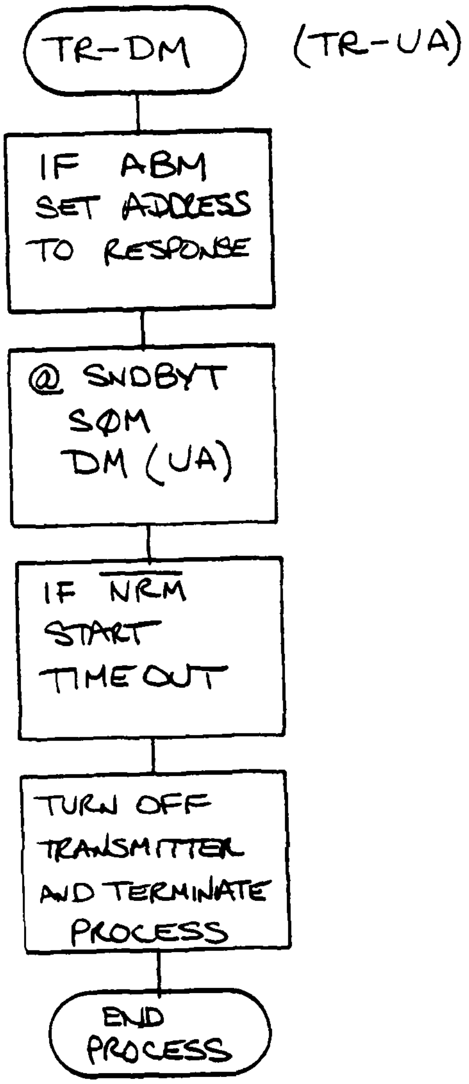


Figure 4-13 Detailed Flow Chart G


TR-DM
(TR-UA)
IF ABM SET ADDRESS TO RESPONSE
@ SNDBYT SØM DM (UA)
IF $\overline{NRM}$ START TIMEOUT
TURN OFF TRANSMITTER AND TERMINATE PROCESS
END PROCESS

IF ABM SET 6856
TO SECONDARY

TURN ON TRANSMITTER
& SEND DM (OR UA)

Figure 4-14 Detailed Flow Chart H

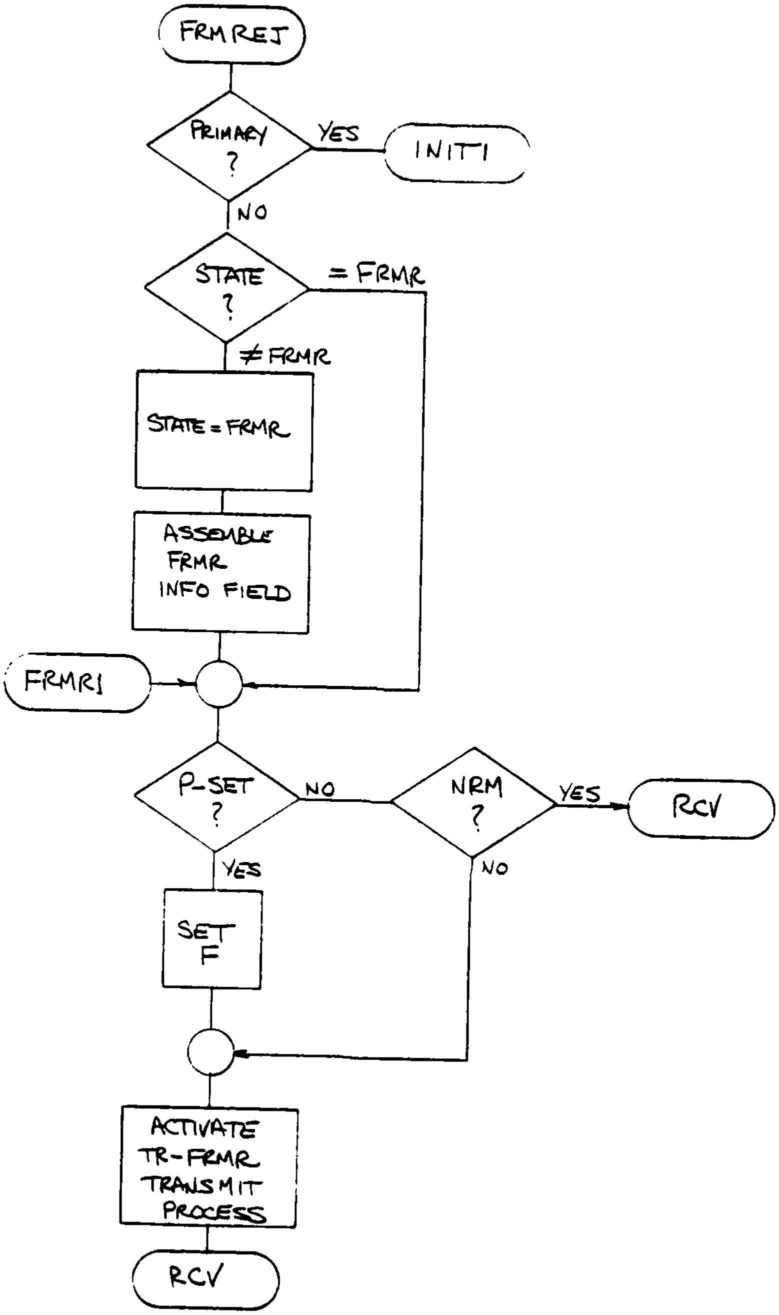


Figure 4-15 Detailed Flow Chart I

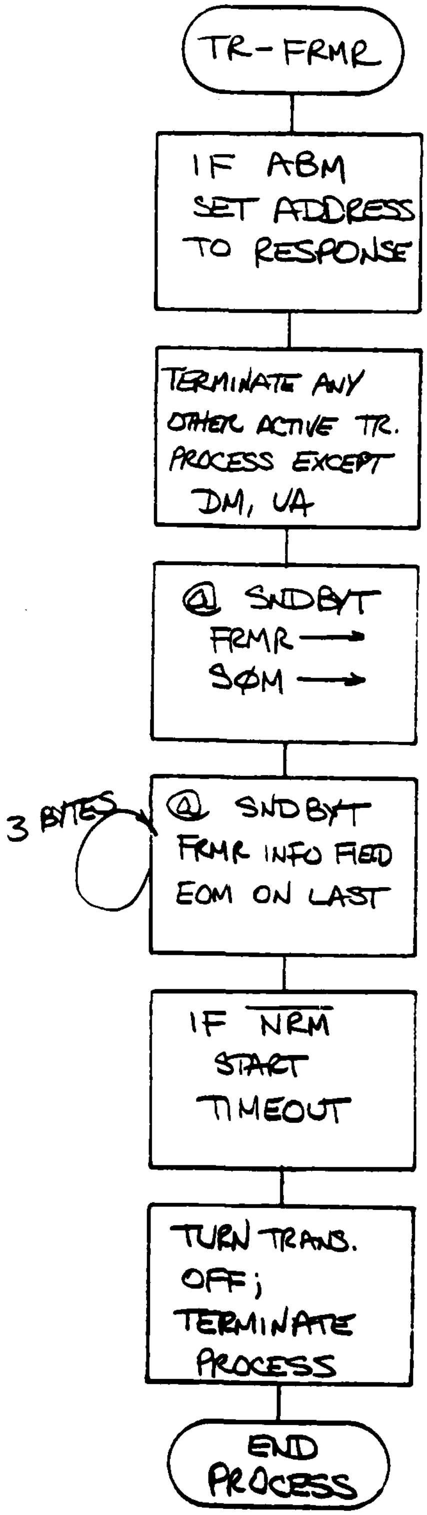

TR-FRMR
IF ABM SET ADDRESS TO RESPONSE
TERMINATE ANY OTHER ACTIVE TR. PROCESS EXCEPT DM, UA
@ SNDBYT
FRMR →
SØM →
3 BYTES
@ SNDBYT
FRMR INFO FIELD
EOM ON LAST
IF $\overline{\text{NRM}}$ START TIMEOUT
TURN TRANS. OFF; TERMINATE PROCESS
END PROCESS

Figure 4-16 Detailed Flow Chart J

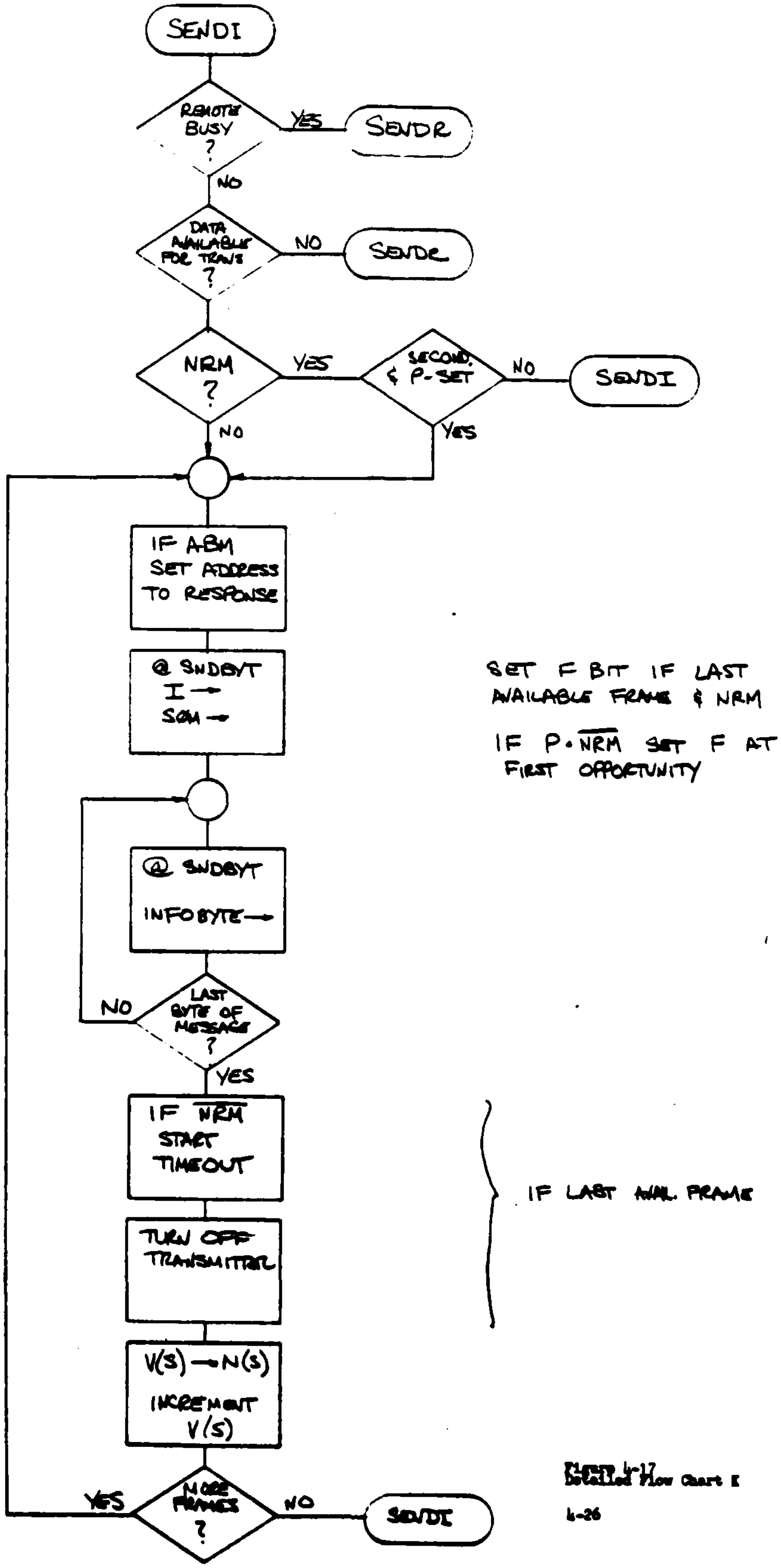


Figure 4-17
Detailed Flow Chart E

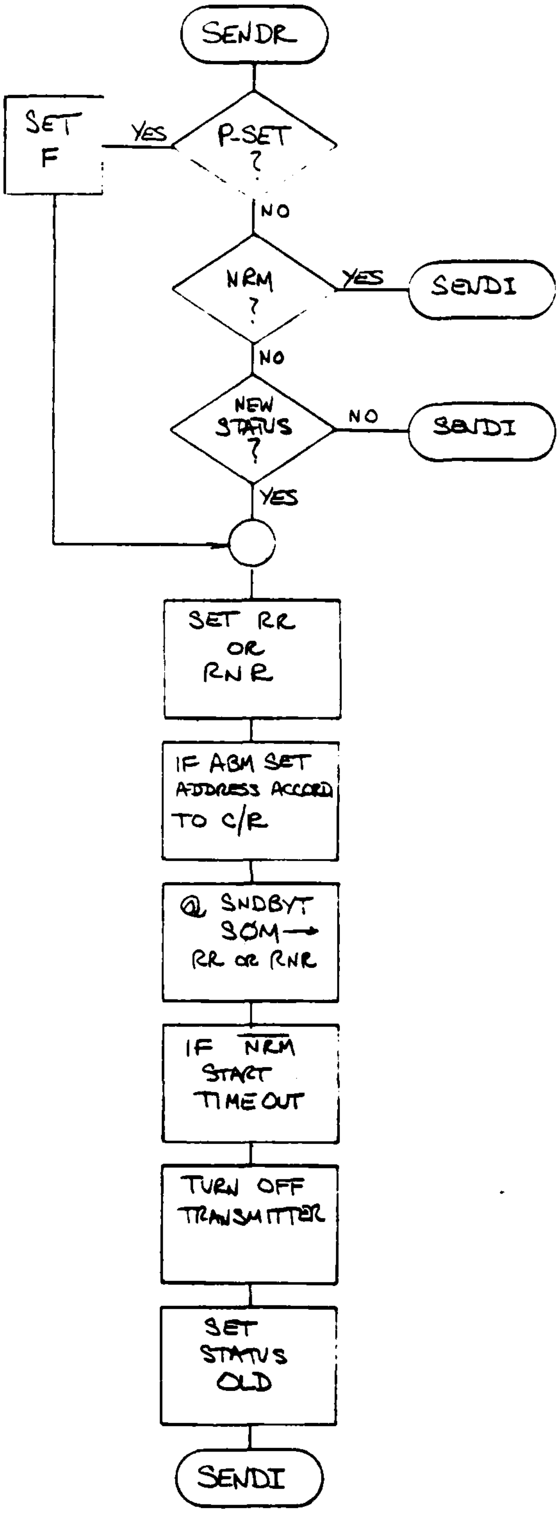
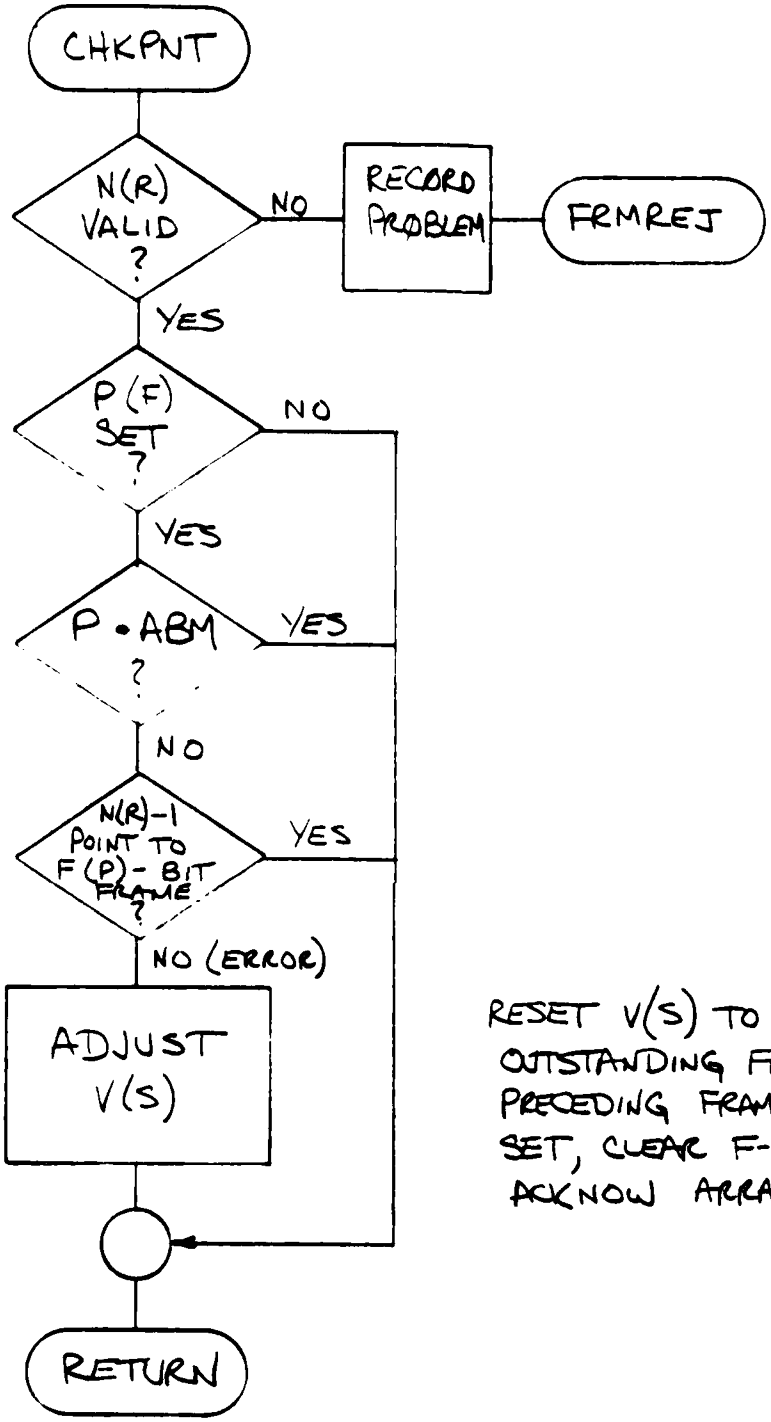


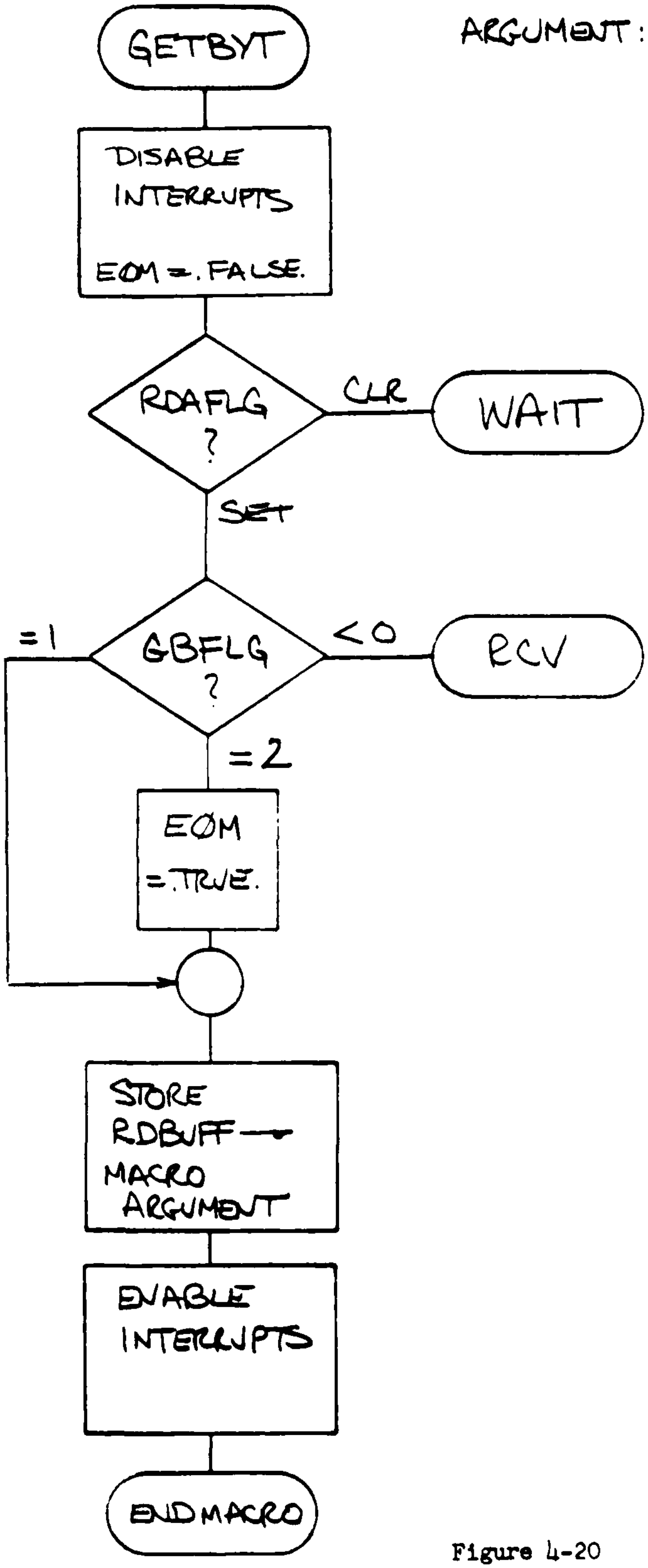
Figure 4-18 Detailed Flow Chart L



DOES N(R)-1 POINT TO AN OUTSTANDING FRAME ?

RESET V(S) TO EARLIEST OUTSTANDING FRAME PRECEDING FRAME WITH F-BIT SET, CLEAR F-BIT ARRAY, ACKNOW ARRAY = -1

Figure 4-19 Detailed Flow Chart M



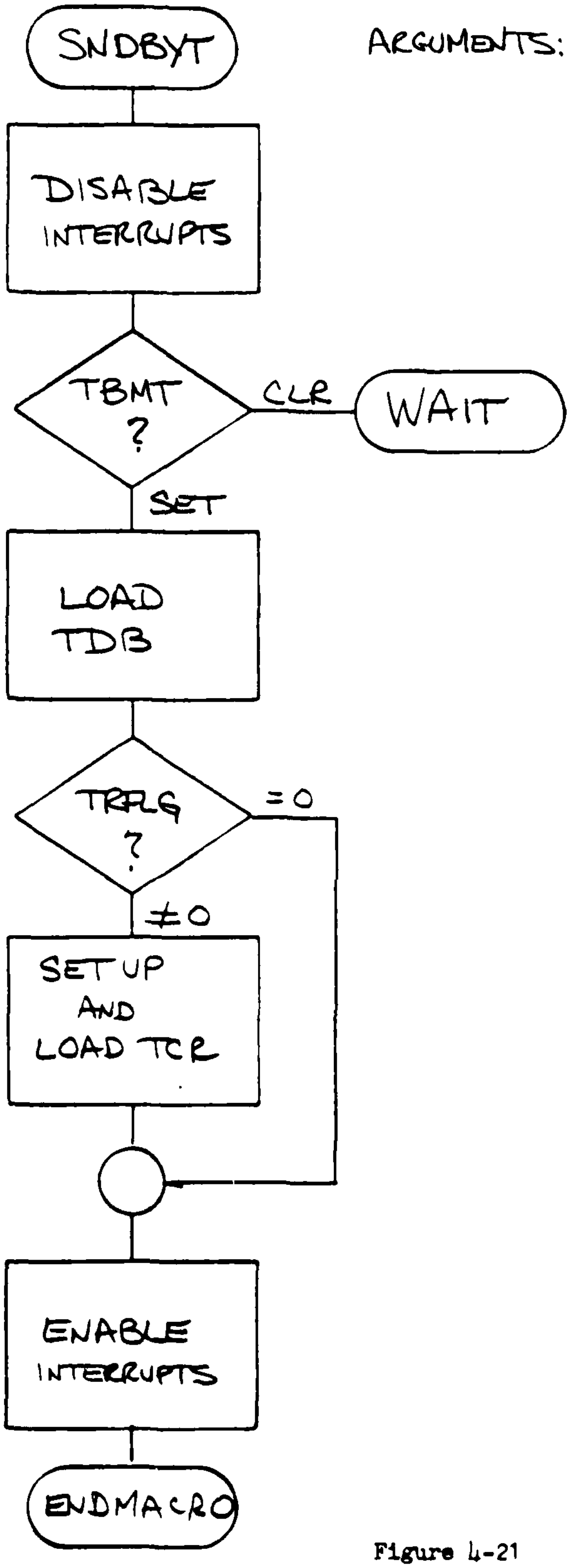
ARGUMENT: BYTE TO BE READ INTO

PREVENT DATA CHANGE
WHILE READING

IF NO BYTE AVAILABLE
GO TO NEXT PROCESS
IN READY QUEUE

IF ERRORS, GO TO
START OF RECEIVE
PROCESS

Figure 4-20
Detailed Flow Chart N



ARGUMENTS: BYTE TO BE TRANSMITTED
TRFLG

IF TRANSMITTER BUFFER NOT EMPTY, GO TO NEXT PROCESS IN READY QUEUE

SET UP FOR:
EOM,
SOM,
ABORT

Figure 4-21
Detailed Flow Chart 0

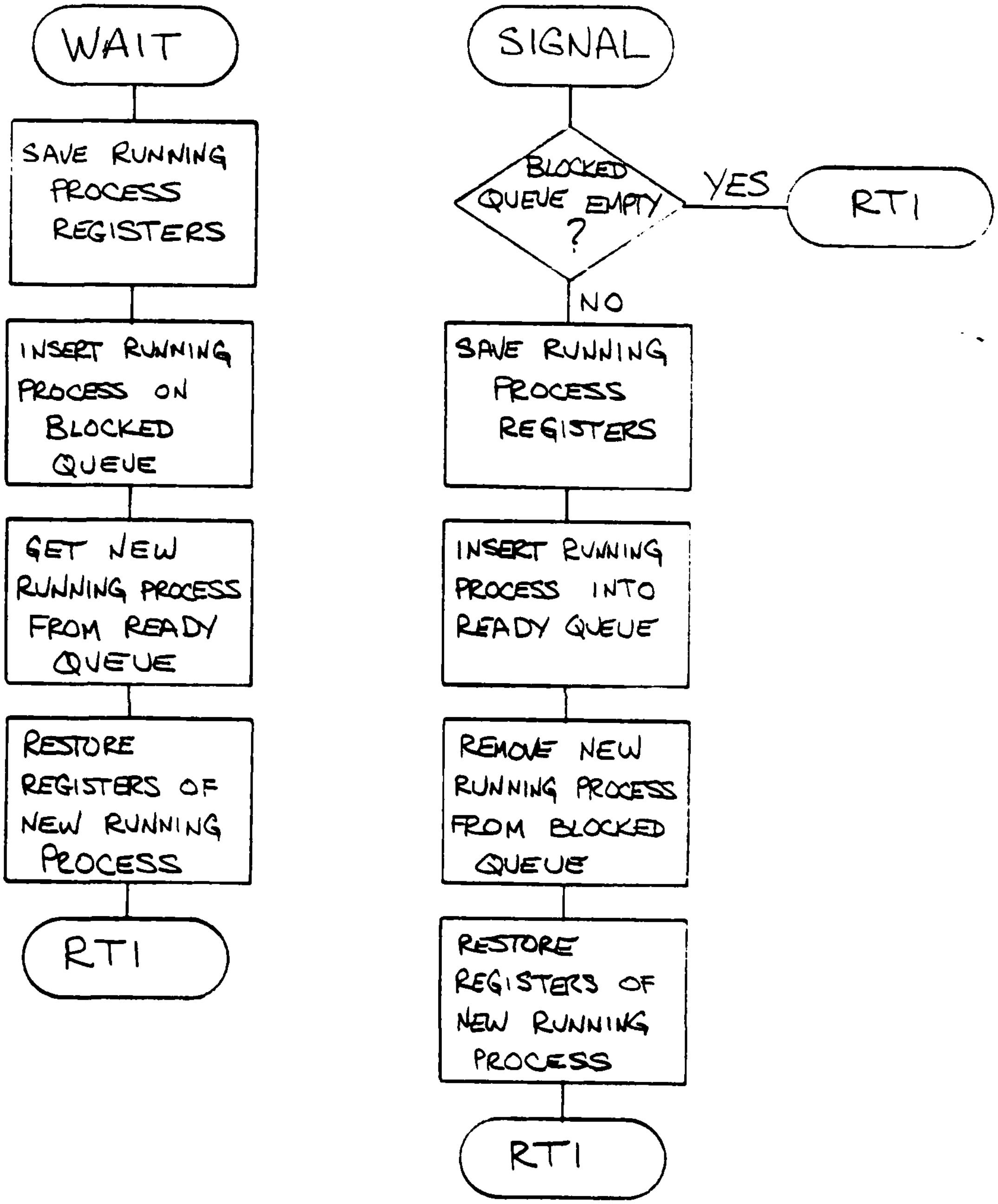

WAIT
SAVE RUNNING PROCESS REGISTERS
INSERT RUNNING PROCESS ON BLOCKED QUEUE
GET NEW RUNNING PROCESS FROM READY QUEUE
RESTORE REGISTERS OF NEW RUNNING PROCESS
RTI
SIGNAL
BLOCKED QUEUE EMPTY ?
YES
RTI
NO
SAVE RUNNING PROCESS REGISTERS
INSERT RUNNING PROCESS INTO READY QUEUE
REMOVE NEW RUNNING PROCESS FROM BLOCKED QUEUE
RESTORE REGISTERS OF NEW RUNNING PROCESS
RTI

Figure 4-22 Detailed Flow Chart P

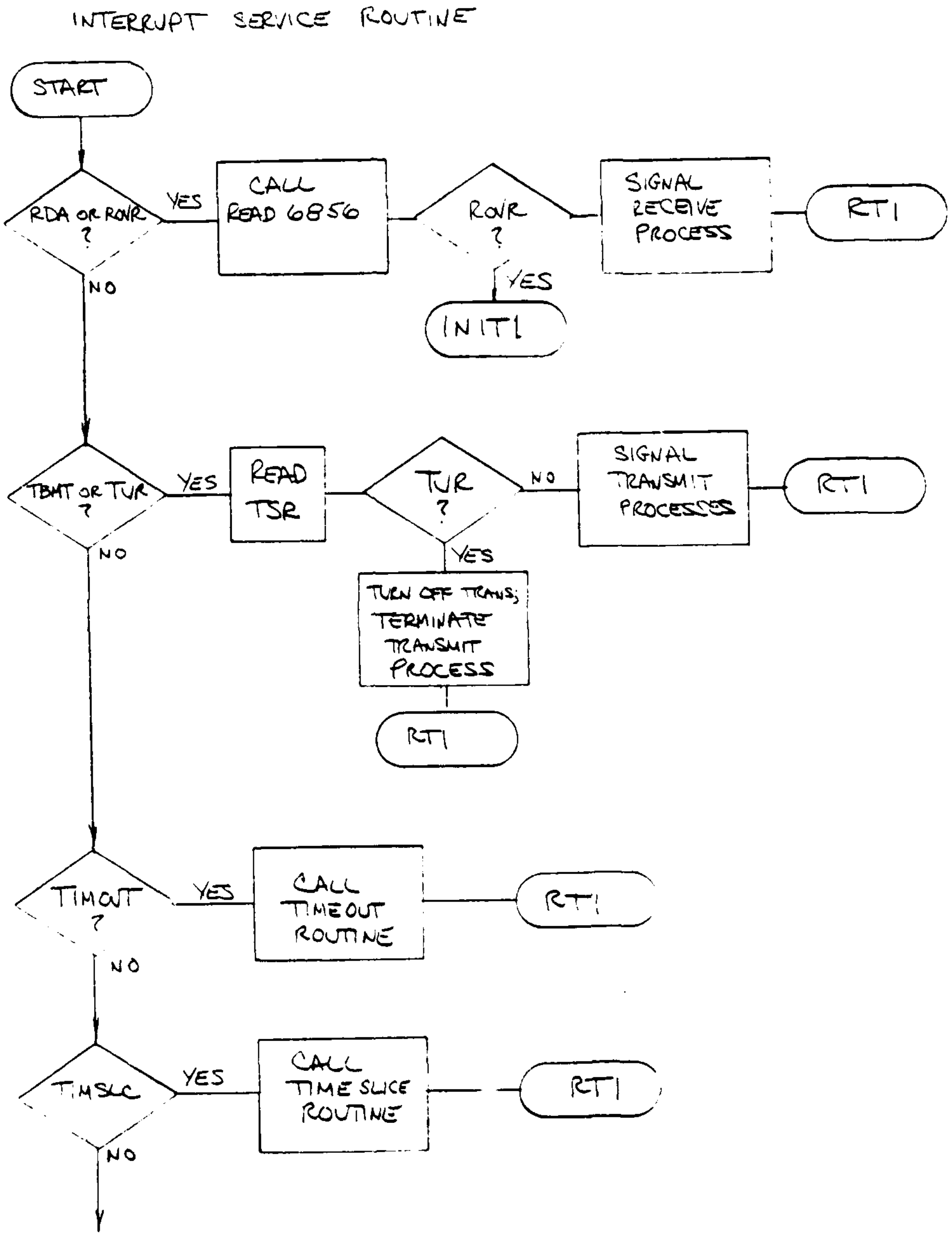


Figure 4-23 Detailed Flow Chart Q

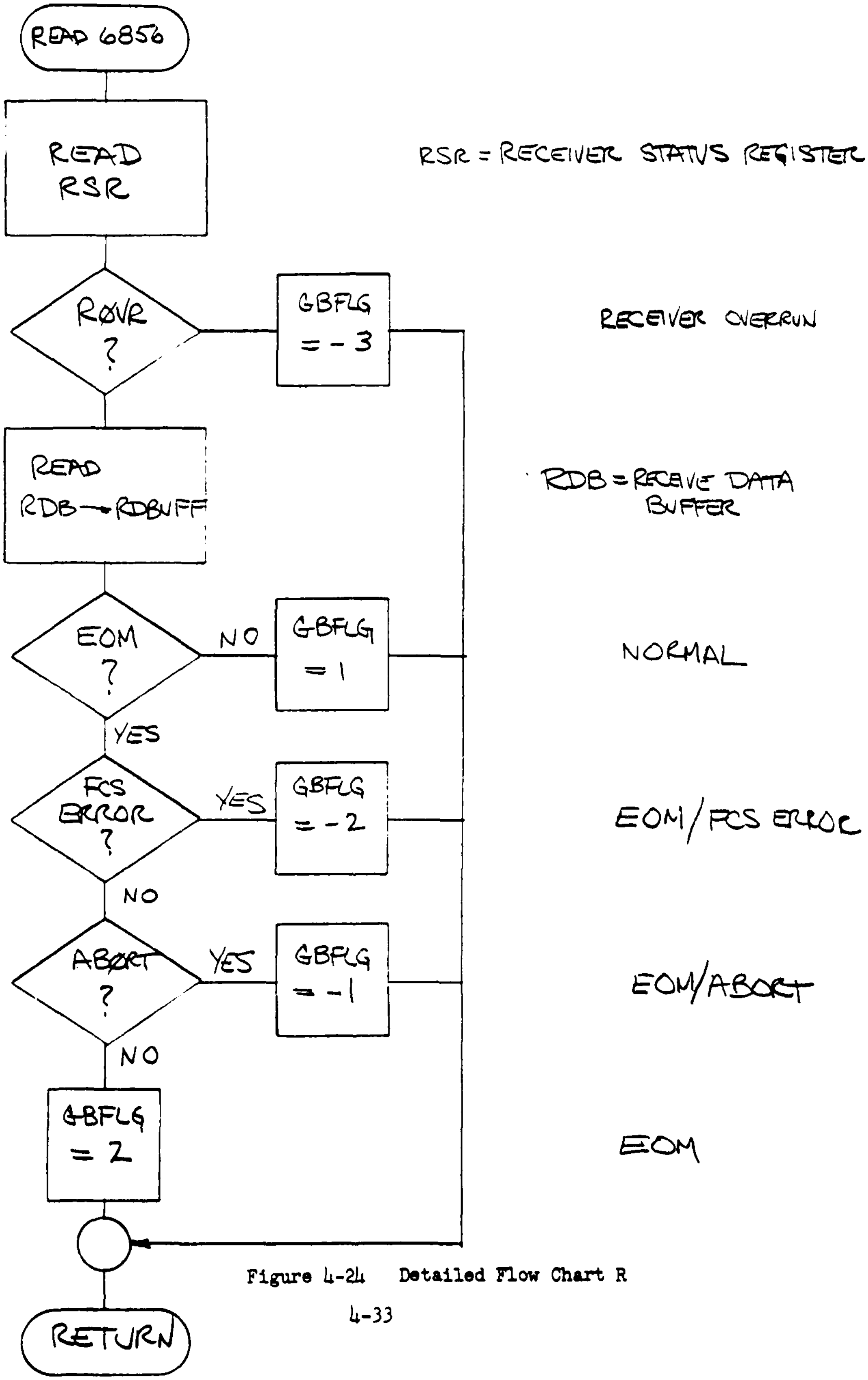


Figure 4-24 Detailed Flow Chart R

5.0 MICROPROCESSOR CODING AND TESTING

5.1 Microprocessor Code

The RCNTRL subroutine has been programmed for the 6800 microprocessor. The code is shown in Figure 5-1. The subroutine examines the control field of the received command and determines which of the eleven valid commands of the set belonging to the three classes of procedures (basic repertoire) has been transmitted. The command is checked against a validity table to determine whether it is valid for the station configuration (primary or secondary, UN or UA or BA). The result is returned in the variable FTYPE. FTYPE is set to -1 if an invalid or un-implemented command is received. The poll/final bit is extracted if the frame type is I, RR, or RNR. N(S) is extracted from I frames. The subroutine requires 94 bytes for instructions plus 22 bytes for tables for a total of 116 bytes in ROM. In addition to the arguments, 2 bytes of temporary storage are used (RAM). The subroutine requires 312 machine cycles through its longest path. This could be reduced considerably by using an associative memory technique for determining the frame type instead of the table search used in the present subroutine. The associative approach becomes more efficient as more commands are added to the repetoire.

5.2 Test Program

Testing of 6800 code was accomplished on a 6800-based microcomputer using a DEC LA-36 printer/terminal. The microcomputer includes a TINY BASIC interpreter (firmware) which was used to facilitate programming of the routine to test the RCNTRL subroutine.

```
ADDR INST   MNEM R  OPER
0800 B60701 LDA  A    0701
0803 16     TAB
0804 C410   AND  B # 10
0806 F70702 STA  B    0702
0809 16     TAB
080A CE0A0B LDX    # 0A0B
080D C4EF   AND  B # EF
080F E100   CMP  B X 00
0811 271E   BEQ       1E
0813 09     DEX
0814 8C0A03 CPX    # 0A03
0817 26F6   BNE      -0A
0819 C40F   AND  B # 0F
081B E100   CMP  B X 00
081D 2712   BEQ       12
081F 09     DEX
0820 8C0A01 CPX    # 0A01
0823 26F6   BNE      -0A
0825 C401   AND  B # 01
0827 E100   CMP  B X 00
0829 2706   BEQ       06
082B 5F     CLR  B

ADDR INST   MNEM R  OPER
082C 5A     DEC  B
082D F70700 STA  B    0700
0830 39     RTS
0831 F60705 LDA  B    0705
0834 E410   AND  B X 10
0836 27F3   BEQ      -0D
0838 FF0860 STX       0860
083B F60861 LDA  B    0861
083E F70700 STA  B    0700
0841 C103   CMP  B # 03
0843 2E19   BGT       19
0845 16     TAB
0846 C4E0   AND  B # E0
0848 54     LSR  B
0849 54     LSR  B
084A 54     LSR  B
084B 54     LSR  B
084C 54     LSR  B
084D F70703 STA  B    0703
0850 F60700 LDA  B    0700
0853 C101   CMP  B # 01
0855 2E07   BGT       07

ADDR INST   MNEM R  OPER
0857 16     TAB
0858 C40E   AND  B # 0E
085A 54     LSR  B
085B F70704 STA  B    0704
085E 39     RTS
085F 00     ***
0860 0A     CLV
```

Figure 5-1 RCNTRL Code

A sample of the test results for this subroutine is shown in Figure 5-2. As shown, the operator is asked to enter the decimal equivalent of a control field. First, the operator entered 182, corresponding to:

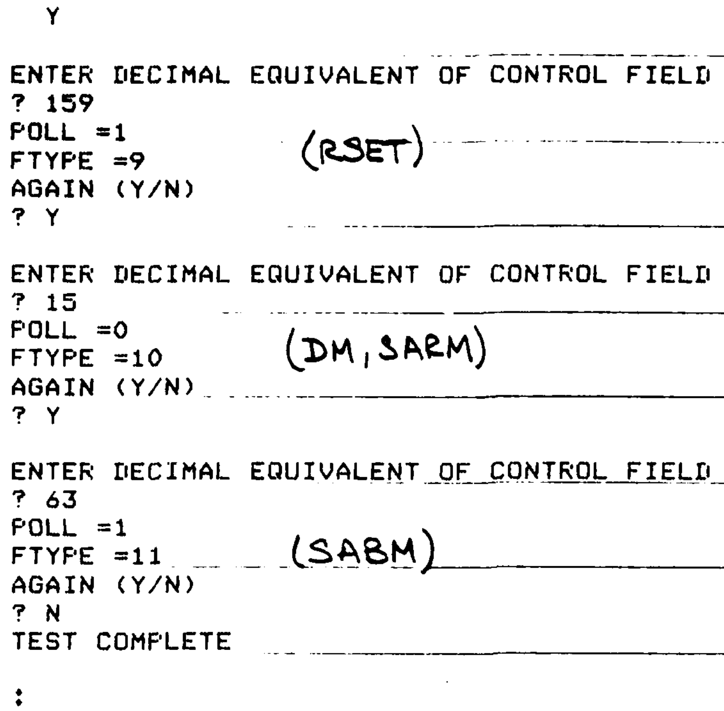
$$\underbrace{1\quad 0\quad 1}_{\text{N(R)}}\quad \underset{\text{P/F}}{\underset{\downarrow}{1}}\quad \underbrace{0\quad 1\quad 1}_{\text{N(S)}}\quad \underset{\text{I-FRAME}}{\underset{\uparrow}{\overset{\overset{\text{LSB}}{\downarrow}}{0}}}$$

The main program loads this byte, that the operator has entered, into the variable called CNTFLD and calls the RCNTRL subprogram. Upon return, the main program prints the values of the poll bit, frame type, N(R) and N(S), if applicable, produced by RCNTRL. The operator is then asked for another value to test. Decimal equivalents of eleven different frame types are shown. For this example the station configuration is set to allow all eleven commands to be valid in order to test the subroutine. Figure 5-3 illustrates the results for the station set to unbalanced normal (UN), secondary with the same control byte inputs. As shown, all commands are invalid except for I, RR, RNR, SNRM and DISC.

An exhaustive test also was made on the RCNTRL subroutine, and the test results are included in Figures 5-4 and 5-5. Since the control field contains 8 bits, 256 possibilities exist, and it was convenient to modify the main program described above to loop through all of the possible values that the control field may have. The result of this test was as expected: Of the 256 fields, starting with 0, every other field was a valid I-frame. There were 16 each of the RR and RNR frames (8 values of N(R) with P-bit "1" and "0") and just 2 each of the remaining commands. The remaining possibilities were marked (correctly) as invalid. Figure 5-4 shows the results for all eleven commands valid, as previously: Figure 5-5 gives the results for the station configured as UN secondary.

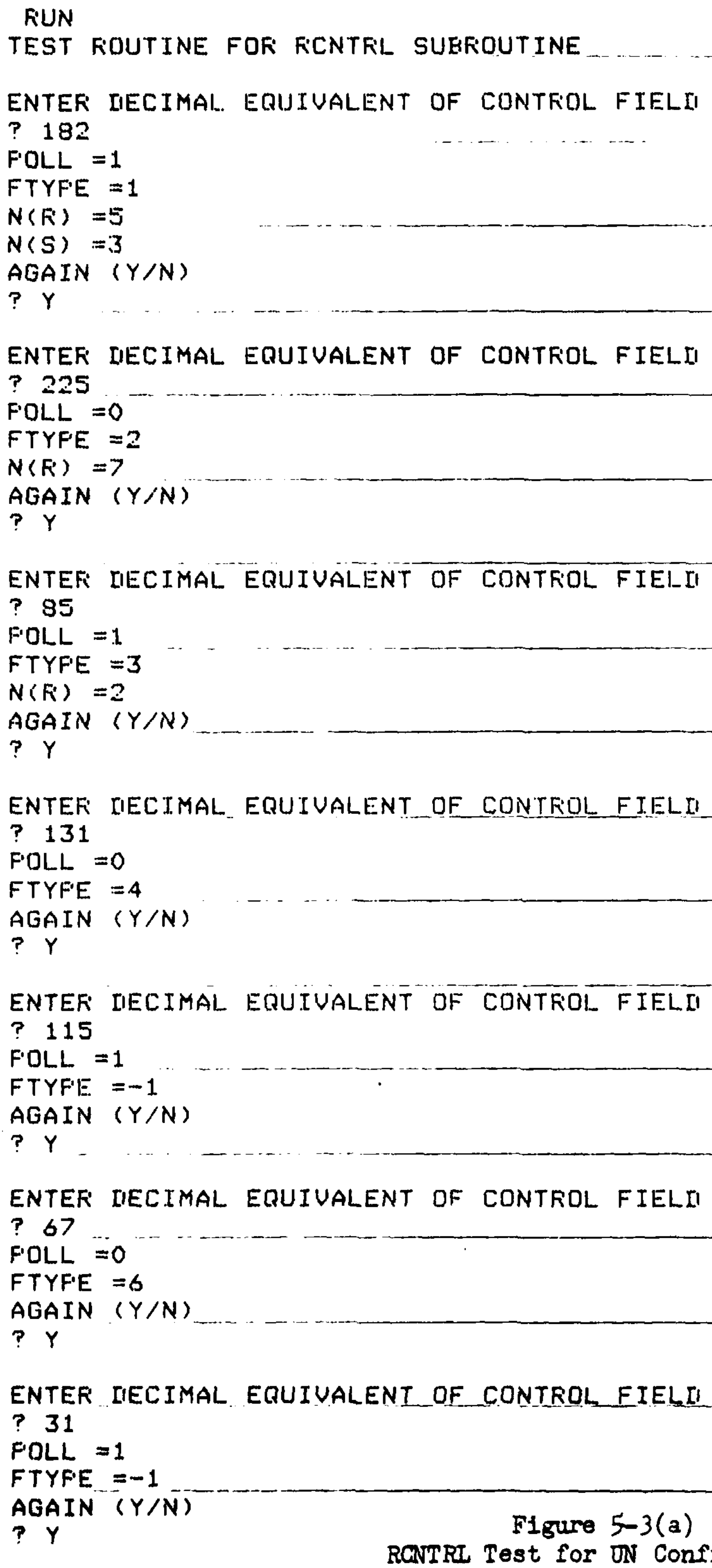
```
 RUN
TEST ROUTINE FOR RCNTRL SUBROUTINE

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 182
POLL =1
FTYPE =1        (I)
N(R) =5
N(S) =3
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 225
POLL =0
FTYPE =2        (RR)
N(R) =7
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 85
POLL =1
FTYPE =3        (RNR)
N(R) =2
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 131
POLL =0         (SNRM)
FTYPE =4
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 115
POLL =1
FTYPE =5         (UA)
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 67
POLL =0          (DISC)
FTYPE =6
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 31
POLL =1
FTYPE =10         (DM, SARM)
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 135
POLL =0
FTYPE =8          (FRMR)
AGAIN (Y/N)
?
```

Figure 5-2(a)
RCNTRL Basic Repetoire
Test Results



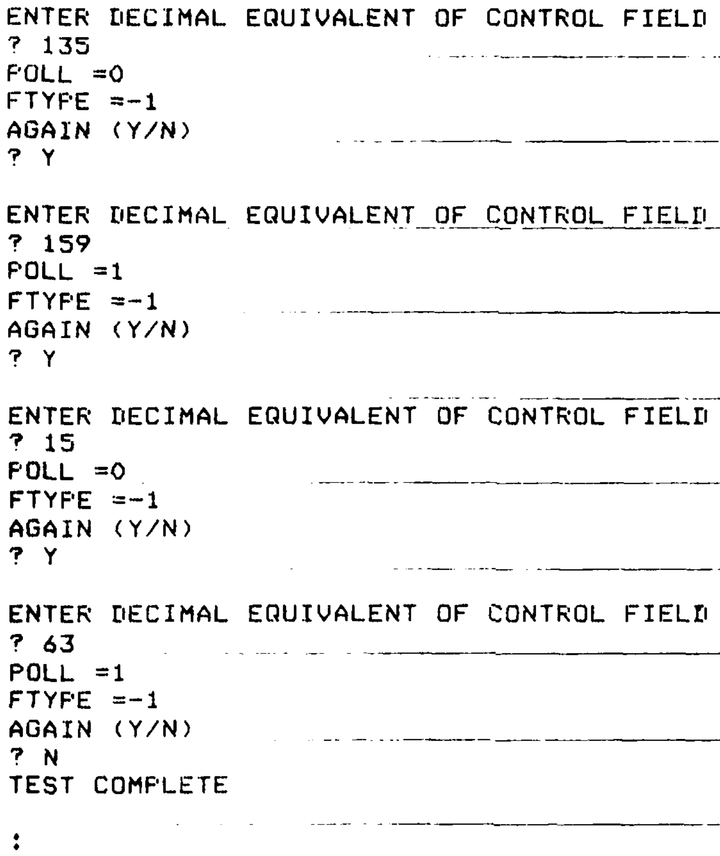
```
  Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 159
POLL =1
FTYPE =9            (RSET)
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 15
POLL =0
FTYPE =10           (DM, SARM)
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 63
POLL =1
FTYPE =11           (SABM)
AGAIN (Y/N)
? N
TEST COMPLETE

:
```

Figure 5-2(b)
RCNTRL Basic Repetoire Test Results



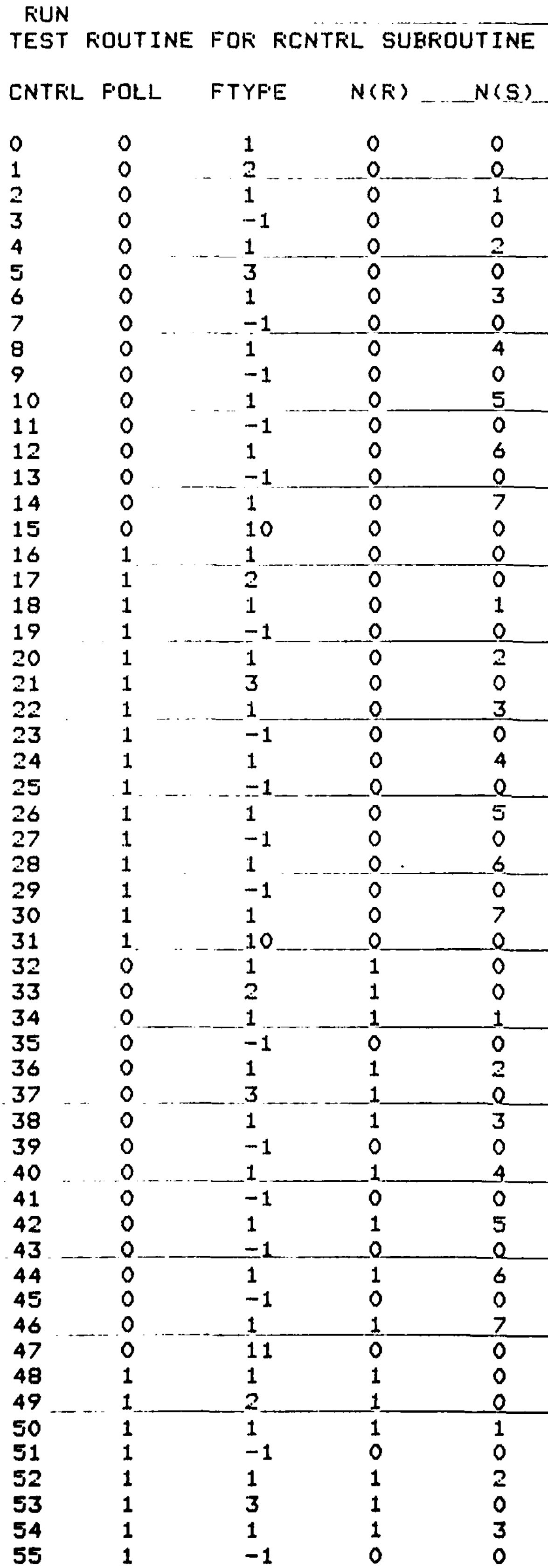
```
 RUN
TEST ROUTINE FOR RCNTRL SUBROUTINE

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 182
POLL =1
FTYPE =1
N(R) =5
N(S) =3
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 225
POLL =0
FTYPE =2
N(R) =7
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 85
POLL =1
FTYPE =3
N(R) =2
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 131
POLL =0
FTYPE =4
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 115
POLL =1
FTYPE =-1
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 67
POLL =0
FTYPE =6
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 31
POLL =1
FTYPE =-1
AGAIN (Y/N)
? Y
```

Figure 5-3(a)
RCNTRL Test for UN Configuration



```
ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 135
POLL =0
FTYPE =-1
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 159
POLL =1
FTYPE =-1
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 15
POLL =0
FTYPE =-1
AGAIN (Y/N)
? Y

ENTER DECIMAL EQUIVALENT OF CONTROL FIELD
? 63
POLL =1
FTYPE =-1
AGAIN (Y/N)
? N
TEST COMPLETE

:
```

Figure 5-3(b)
RCNTRL Test for UN Configuration



```
 RUN
TEST ROUTINE FOR RCNTRL SUBROUTINE

CNTRL POLL   FTYPE    N(R)    N(S)

0       0      1        0       0
1       0      2        0       0
2       0      1        0       1
3       0     -1        0       0
4       0      1        0       2
5       0      3        0       0
6       0      1        0       3
7       0     -1        0       0
8       0      1        0       4
9       0     -1        0       0
10      0      1        0       5
11      0     -1        0       0
12      0      1        0       6
13      0     -1        0       0
14      0      1        0       7
15      0      10       0       0
16      1      1        0       0
17      1      2        0       0
18      1      1        0       1
19      1     -1        0       0
20      1      1        0       2
21      1      3        0       0
22      1      1        0       3
23      1     -1        0       0
24      1      1        0       4
25      1     -1        0       0
26      1      1        0       5
27      1     -1        0       0
28      1      1        0       6
29      1     -1        0       0
30      1      1        0       7
31      1      10       0       0
32      0      1        1       0
33      0      2        1       0
34      0      1        1       1
35      0     -1        0       0
36      0      1        1       2
37      0      3        1       0
38      0      1        1       3
39      0     -1        0       0
40      0      1        1       4
41      0     -1        0       0
42      0      1        1       5
43      0     -1        0       0
44      0      1        1       6
45      0     -1        0       0
46      0      1        1       7
47      0      11       0       0
48      1      1        1       0
49      1      2        1       0
50      1      1        1       1
51      1     -1        0       0
52      1      1        1       2
53      1      3        1       0
54      1      1        1       3
55      1     -1        0       0
```

Figure 5-4(a)
RCNTRL Exhaustive Test-- All Station Configurations (Table A)

56	1	1	1	4
57	1	-1	0	0
58	1	1	1	5
59	1	-1	0	0
60	1	1	1	6
61	1	-1	0	0
62	1	1	1	7
63	1	11	0	0
64	0	1	2	0
65	0	2	2	0
66	0	1	2	1
67	0	6	0	0
68	0	1	2	2
69	0	3	2	0
70	0	1	2	3
71	0	-1	0	0
72	0	1	2	4
73	0	-1	0	0
74	0	1	2	5
75	0	-1	0	0
76	0	1	2	6
77	0	-1	0	0
78	0	1	2	7
79	0	-1	0	0
80	1	1	2	0
81	1	2	2	0
82	1	1	2	1
83	1	6	0	0
84	1	1	2	2
85	1	3	2	0
86	1	1	2	3
87	1	-1	0	0
88	1	1	2	4
89	1	-1	0	0
90	1	1	2	5
91	1	-1	0	0
92	1	1	2	6
93	1	-1	0	0
94	1	1	2	7
95	1	-1	0	0
96	0	1	3	0
97	0	2	3	0
98	0	1	3	1
99	0	5	0	0
100	0	1	3	2
101	0	3	3	0
102	0	1	3	3
103	0	-1	0	0
104	0	1	3	4
105	0	-1	0	0
106	0	1	3	5
107	0	-1	0	0
108	0	1	3	6
109	0	-1	0	0
110	0	1	3	7
111	0	-1	0	0
112	1	1	3	0
113	1	2	3	0
114	1	1	3	1
115	1	5	0	0
116	1	1	3	2
117	1	3	3	0
118	1	1	3	3
119	1	-1	0	0
120	1	1	3	4
121	1	-1	0	0

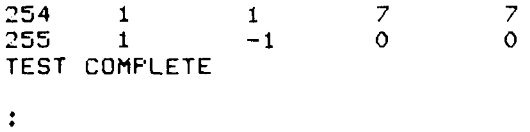
Figure 5-4(b)
RCNTRL Exhaustive Test-- All Station Configurations (Table B)

122	1	1	3	5
123	1	-1	0	0
124	1	1	3	6
125	1	-1	0	0
126	1	1	3	7
127	1	-1	0	0
128	0	1	4	0
129	0	2	4	0
130	0	1	4	1
131	0	4	0	0
132	0	1	4	2
133	0	3	4	0
134	0	1	4	3
135	0	8	0	0
136	0	1	4	4
137	0	-1	0	0
138	0	1	4	5
139	0	-1	0	0
140	0	1	4	6
141	0	-1	0	0
142	0	1	4	7
143	0	9	0	0
144	1	1	4	0
145	1	2	4	0
146	1	1	4	1
147	1	4	0	0
148	1	1	4	2
149	1	3	4	0
150	1	1	4	3
151	1	8	0	0
152	1	1	4	4
153	1	-1	0	0
154	1	1	4	5
155	1	-1	0	0
156	1	1	4	6
157	1	-1	0	0
158	1	1	4	7
159	1	9	0	0
160	0	1	5	0
161	0	2	5	0
162	0	1	5	1
163	0	-1	0	0
164	0	1	5	2
165	0	3	5	0
166	0	1	5	3
167	0	-1	0	0
168	0	1	5	4
169	0	-1	0	0
170	0	1	5	5
171	0	-1	0	0
172	0	1	5	6
173	0	-1	0	0
174	0	1	5	7
175	0	-1	0	0
176	1	1	5	0
177	1	2	5	0
178	1	1	5	1
179	1	-1	0	0
180	1	1	5	2
181	1	3	5	0
182	1	1	5	3
183	1	-1	0	0
184	1	1	5	4
185	1	-1	0	0
186	1	1	5	5
187	1	-1	0	0

Figure 5-4(c)
RCNTRL Exhaustive Test-- All Station Configurations (Table C)

188	1	1	5	6
189	1	-1	0	0
190	1	1	5	7
191	1	-1	0	0
192	0	1	6	0
193	0	2	6	0
194	0	1	6	1
195	0	-1	0	0
196	0	1	6	2
197	0	3	6	0
198	0	1	6	3
199	0	-1	0	0
200	0	1	6	4
201	0	-1	0	0
202	0	1	6	5
203	0	-1	0	0
204	0	1	6	6
205	0	-1	0	0
206	0	1	6	7
207	0	-1	0	0
208	1	1	6	0
209	1	2	6	0
210	1	1	6	1
211	1	-1	0	0
212	1	1	6	2
213	1	3	6	0
214	1	1	6	3
215	1	-1	0	0
216	1	1	6	4
217	1	-1	0	0
218	1	1	6	5
219	1	-1	0	0
220	1	1	6	6
221	1	-1	0	0
222	1	1	6	7
223	1	-1	0	0
224	0	1	7	0
225	0	2	7	0
226	0	1	7	1
227	0	-1	0	0
228	0	1	7	2
229	0	3	7	0
230	0	1	7	3
231	0	-1	0	0
232	0	1	7	4
233	0	-1	0	0
234	0	1	7	5
235	0	-1	0	0
236	0	1	7	6
237	0	-1	0	0
238	0	1	7	7
239	0	-1	0	0
240	1	1	7	0
241	1	2	7	0
242	1	1	7	1
243	1	-1	0	0
244	1	1	7	2
245	1	3	7	0
246	1	1	7	3
247	1	-1	0	0
248	1	1	7	4
249	1	-1	0	0
250	1	1	7	5
251	1	-1	0	0
252	1	1	7	6
253	1	-1	0	0

Figure 5-4(d)
RCNTRL Exhaustive Test-- All Station Configurations (Table D)



```
254     1       1       7       7
255     1      -1       0       0
TEST COMPLETE

:
```

Figure 5-4(e)
RCNTRL Exhaustive Test-- All Station Configurations (Table E)

```
 RUN
TEST ROUTINE FOR RCNTRL SUBROUTINE

CNTRL POLL    FTYPE     N(R)     N(S)

0       0        1        0        0
1       0        2        0        0
2       0        1        0        1
3       0       -1        0        0
4       0        1        0        2
5       0        3        0        0
6       0        1        0        3
7       0       -1        0        0
8       0        1        0        4
9       0       -1        0        0
10      0        1        0        5
11      0       -1        0        0
12      0        1        0        6
13      0       -1        0        0
14      0        1        0        7
15      0       -1        0        0
16      1        1        0        0
17      1        2        0        0
18      1        1        0        1
19      1       -1        0        0
20      1        1        0        2
21      1        3        0        0
22      1        1        0        3
23      1       -1        0        0
24      1        1        0        4
25      1       -1        0        0
26      1        1        0        5
27      1       -1        0        0
28      1        1        0        6
29      1       -1        0        0
30      1        1        0        7
31      1       -1        0        0
32      0        1        1        0
33      0        2        1        0
34      0        1        1        1
35      0       -1        0        0
36      0        1        1        2
37      0        3        1        0
38      0        1        1        3
39      0       -1        0        0
40      0        1        1        4
41      0       -1        0        0
42      0        1        1        5
43      0       -1        0        0
44      0        1        1        6
45      0       -1        0        0
46      0        1        1        7
47      0       -1        0        0
48      1        1        1        0
49      1        2        1        0
50      1        1        1        1
51      1       -1        0        0
52      1        1        1        2
53      1        3        1        0
54      1        1        1        3
55      1       -1        0        0
56      1        1        1        4
```

Figure 5-5(a)
RCNTRL Exhaustive Test Results--
UN Secondary (Table A)

57	1	-1	0	0
58	1	1	1	5
59	1	-1	0	0
60	1	1	1	6
61	1	-1	0	0
62	1	1	1	7
63	1	-1	0	0
64	0	1	2	0
65	0	2	2	0
66	0	1	2	1
67	0	6	0	0
68	0	1	2	2
69	0	3	2	0
70	0	1	2	3
71	0	-1	0	0
72	0	1	2	4
73	0	-1	0	0
74	0	1	2	5
75	0	-1	0	0
76	0	1	2	6
77	0	-1	0	0
78	0	1	2	7
79	0	-1	0	0
80	1	1	2	0
81	1	2	2	0
82	1	1	2	1
83	1	6	0	0
84	1	1	2	2
85	1	3	2	0
86	1	1	2	3
87	1	-1	0	0
88	1	1	2	4
89	1	-1	0	0
90	1	1	2	5
91	1	-1	0	0
92	1	1	2	6
93	1	-1	0	0
94	1	1	2	7
95	1	-1	0	0
96	0	1	3	0
97	0	2	3	0
98	0	1	3	1
99	0	-1	0	0
100	0	1	3	2
101	0	3	3	0
102	0	1	3	3
103	0	-1	0	0
104	0	1	3	4
105	0	-1	0	0
106	0	1	3	5
107	0	-1	0	0
108	0	1	3	6
109	0	-1	0	0
110	0	1	3	7
111	0	-1	0	0
112	1	1	3	0
113	1	2	3	0
114	1	1	3	1
115	1	-1	0	0
116	1	1	3	2
117	1	3	3	0
118	1	1	3	3
119	1	-1	0	0
120	1	1	3	4
121	1	-1	0	0
122	1	1	3	5

Figure 5-5(b)
RCNTRL Exhaustive Test Results--
UN Secondary (Table B)

123	1	-1	0	0
124	1	1	3	6
125	1	-1	0	0
126	1	1	3	7
127	1	-1	0	0
128	0	1	4	0
129	0	2	4	0
130	0	1	4	1
131	0	4	0	0
132	0	1	4	2
133	0	3	4	0
134	0	1	4	3
135	0	-1	0	0
136	0	1	4	4
137	0	-1	0	0
138	0	1	4	5
139	0	-1	0	0
140	0	1	4	6
141	0	-1	0	0
142	0	1	4	7
143	0	-1	0	0
144	1	1	4	0
145	1	2	4	0
146	1	1	4	1
147	1	4	0	0
148	1	1	4	2
149	1	3	4	0
150	1	1	4	3
151	1	-1	0	0
152	1	1	4	4
153	1	-1	0	0
154	1	1	4	5
155	1	-1	0	0
156	1	1	4	6
157	1	-1	0	0
158	1	1	4	7
159	1	-1	0	0
160	0	1	5	0
161	0	2	5	0
162	0	1	5	1
163	0	-1	0	0
164	0	1	5	2
165	0	3	5	0
166	0	1	5	3
167	0	-1	0	0
168	0	1	5	4
169	0	-1	0	0
170	0	1	5	5
171	0	-1	0	0
172	0	1	5	6
173	0	-1	0	0
174	0	1	5	7
175	0	-1	0	0
176	1	1	5	0
177	1	2	5	0
178	1	1	5	1
179	1	-1	0	0
180	1	1	5	2
181	1	3	5	0
182	1	1	5	3
183	1	-1	0	0
184	1	1	5	4
185	1	-1	0	0
186	1	1	5	5
187	1	-1	0	0
188	1	1	5	6

Figure 5-5(c)
RCNTRL Exhaustive Test Results--
UN Secondary (Table C)

189	1	-1	0	0
190	1	1	5	7
191	1	-1	0	0
192	0	1	6	0
193	0	2	6	0
194	0	1	6	1
195	0	-1	0	0
196	0	1	6	2
197	0	3	6	0
198	0	1	6	3
199	0	-1	0	0
200	0	1	6	4
201	0	-1	0	0
202	0	1	6	5
203	0	-1	0	0
204	0	1	6	6
205	0	-1	0	0
206	0	1	6	7
207	0	-1	0	0
208	1	1	6	0
209	1	2	6	0
210	1	1	6	1
211	1	-1	0	0
212	1	1	6	2
213	1	3	6	0
214	1	1	6	3
215	1	-1	0	0
216	1	1	6	4
217	1	-1	0	0
218	1	1	6	5
219	1	-1	0	0
220	1	1	6	6
221	1	-1	0	0
222	1	1	6	7
223	1	-1	0	0
224	0	1	7	0
225	0	2	7	0
226	0	1	7	1
227	0	-1	0	0
228	0	1	7	2
229	0	3	/	0
230	0	1	7	3
231	0	-1	0	0
232	0	1	7	4
233	0	-1	0	0
234	0	1	7	5
235	0	-1	0	0
236	0	1	7	6
237	0	-1	0	0
238	0	1	7	7
239	0	-1	0	0
240	1	1	7	0
241	1	2	7	0
242	1	1	7	1
243	1	-1	0	0
244	1	1	7	2
245	1	3	7	0
246	1	1	7	3
247	1	-1	0	0
248	1	1	7	4
249	1	-1	0	0
250	1	1	7	5
251	1	-1	0	0
252	1	1	7	6
253	1	-1	0	0
254	1	1	7	7

Figure 5-5(d)
RCNTRL Exhaustive Test Results--
UN Secondary (Table D)

```
255    1        -1       0        0
TEST COMPLETE

:
```

Figure 5-5(e)
RCNTRL Exhaustive Test Results -- UN Secondary (Table E)

6.0 DISCUSSION OF FEASIBILITY

As discussed previously, one of the objectives of this program is to determine the practicality of using a microprocessor, such as the M6800, to implement the unbalanced normal, unbalanced asynchronous, and balanced asynchronous class of procedures. Two major factors affecting the feasibility are the number of instructions required to implement the protocol, and the time necessary to execute these instructions. The total number of instructions has a significant effect on the cost of developing a processor-based system, and the throughput (or baud-rate over the communication line in this instance) is determined by the execution speed through critical paths on the program. These factors are discussed below.

6.1 Memory Requirements

The number of instructions required to implement the protocol can be approximated by examining the detailed flow charts in Section 4.0. This number is estimated to be 850 instructions. Note that this does not include code for an operating system or code required to manage the concurrent processes discussed previously. Approximately 250 instructions will be required for the OS, including a task multiplexer, depending on the hardware design and desired features.

Memory is also required for variable storage (approximately 220 bytes) and for the data buffers fcr sending and receiving messages. Two eight-message buffers would require 16 times the number of bytes in a message.

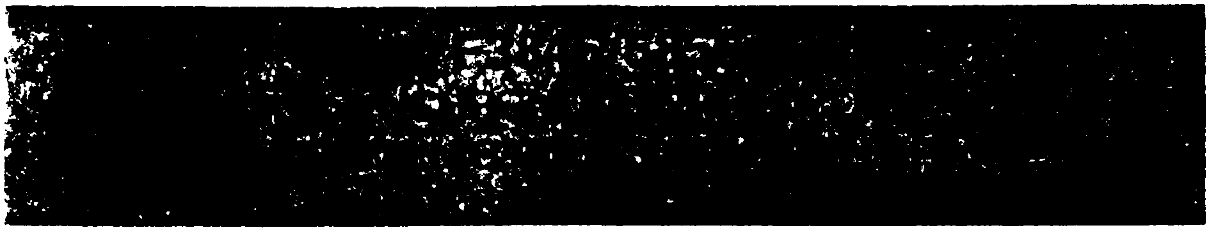
6.2 Execution Time

The speed at which the microprocessor can execute the protocol in real-time depends to a large extent on the actual hardware/software design: The hardware design can be "standard" or it can include many processes accomplished in hardware (such as the F6856). For the purpose of this program, the standard approach with the aid of the F6856 is assumed. The software design must address the time-critical portions of the simultaneous transmit/receive processes to ensure that these critical processes may be serviced in real time. For this program, no attempt has been made to optimize these processes, since a thorough analysis is required to determine just what is "critical." However, some rough estimates can be made based on the current state of the design.

Assuming a MPU rate of 1 cycle/microsec. it appears that a 9.6 or 19.2 kilobit/sec. transmission rate would not be too difficult. That is, a 19.2 kilobit/sec rate is equivalent to 400 microseconds per byte transmitted, which is approximately 100 instructions. It should be possible to implement the critical parts of the send/receive process using between 100 and 200 instructions. A more thorough analysis might reveal that 100 kilobit/sec rate may be possible, but certainly difficult. A faster MPU and additional hardware might be required. Another tradeoff that can be made is memory for speed; that is, table look-up may be used in some cases to reduce the number of instructions required to be executed.

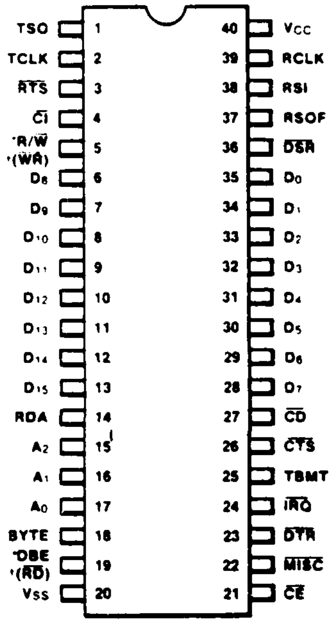
APPENDIX A

SYNCHRONOUS PROTOCOL COMMUNICATIONS CONTROLLER - F6856



DESCRIPTION – The F3846/F6856 Synchronous Protocol Communications Controller (SPCC) is a monolithic n-channel MOS-LSI circuit designed to satisfy the major interface requirements for Bit Oriented (BOP) and Byte Control Protocols (BCP). The SPCC converts parallel data from the CPU to a continuous serial data stream for transmission. Simultaneously, it converts received serial data to parallel data for the CPU. The SPCC is organized to interface with either an 8 or 16-bit bidirectional data bus, is fully TTL compatible and operates from a single +5 V supply.

CONNECTION DIAGRAM
DIP (TOP VIEW)



*6856 Designation
†3846 Designation

- **F6800 AND 8080 BUS COMPATIBLE**
- **DC TO 1M BPS DATA RATE**
- **LINE CONTROL PROTOCOLS**
  BIT ORIENTED PROTOCOLS (BOP): SDLC, ADCCP, HDLC
  BYTE CONTROL PROTOCOLS (BCP): BISYNC, DDCMP AND OTHER BCP
- **BIT ORIENTED PROTOCOLS**
  AUTOMATIC DETECTION AND GENERATION OF SPECIAL CONTROL SEQUENCES, I.e., FLAG, ABORT, GO-AHEAD
  ZERO INSERTION AND DELETION
  PRIMARY OR SECONDARY STATION SELECT
  GLOBAL ADDRESS
  AUTOMATIC EXTENDED ADDRESS
  ONE OR TWO CONTROL BYTES
  DATA CHARACTER LENGTH OF FIVE TO EIGHT BITS WITH 1 TO 8-BIT RESIDUAL LAST CHARACTER
  CCITT-CRC ERROR DETECTION
  IBM RETAIL STORE LOOP MODE
- **BYTE CONTROL PROTOCOL – BISYNC**
  SPECIAL CHARACTER GENERATION: DLE, SYNC
  SPECIAL CHARACTER DETECTION: DLE, SYNC, SOH, STX, ITB, ETB, ETX
  USASCII OR EBCDIC
  NON-TRANSPARENT MODE AND TRANSPARENT MODE
  8-BIT CHARACTER LENGTH
  AUTOMATIC FILL CHARACTER INSERTION WITH SELECTABLE STRIPPING
  CCITT OR CRC-16 ERROR DETECTION
- **BYTE CONTROL PROTOCOLS – DDCMP AND OTHER**
  PROGRAMMABLE SYNC CHARACTERS
  5 TO 8-BIT CHARACTER LENGTH
  SELECTABLE CRC ERROR DETECTION
  AUTOMATIC FILL CHARACTER INSERTION WITH SELECTABLE STRIPPING
- **DIRECTLY ADDRESSABLE PARAMETER CONTROL REGISTERS: MODE, SYNC/ADDRESS, TRANSMITTER CONTROL AND RECEIVER CONTROL**
- **SEPARATE ADDRESSABLE STATUS AND DATA REGISTERS FOR RECEIVER AND TRANSMITTER**
- **MODEM HANDSHAKE SIGNALS: RTS, CTS, DTR, DSR AND CARRIER DETECT (CD)**
- **NRZ OR NRZI (ZERO COMPLIMENTING)**
- **FULL OR HALF DUPLEX OPERATION**
- **SELF TEST LOOP MODE**
- **8 OR 16-BIT BIDIRECTIONAL 3-STATE DATA BUS**
- **TTL COMPATIBLE**
- **SINGLE +5 V SUPPLY**
- **40-PIN PACKAGE**

464 ELLIS STREET, MOUNTAIN VIEW, CALIFORNIA, 94042 (415) 962-5011/TWX 910-379-6435

FAIRCHILD

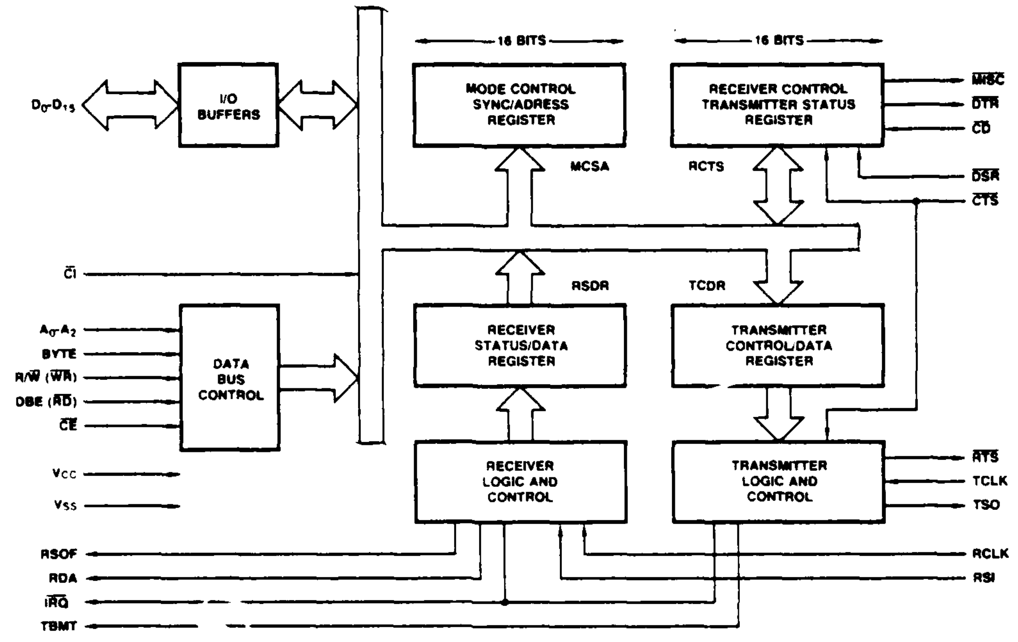

16 BITS
16 BITS
D_0-D_{15}
I/O BUFFERS
MODE CONTROL SYNC/ADRESS REGISTER
RECEIVER CONTROL TRANSMITTER STATUS REGISTER
MISC
DTR
CD
MCSA
RCTS
DSR
CTS
CI
RSDR
TCDR
A_0-A_2
BYTE
R/W (WR)
DBE (RD)
CE
DATA BUS CONTROL
RECEIVER STATUS/DATA REGISTER
TRANSMITTER CONTROL/DATA REGISTER
Vcc
Vss
RECEIVER LOGIC AND CONTROL
TRANSMITTER LOGIC AND CONTROL
RTS
TCLK
TSO
RSOF
RDA
IRQ
TBMT
RCLK
RSI

Fig. 1 BLOCK DIAGRAM

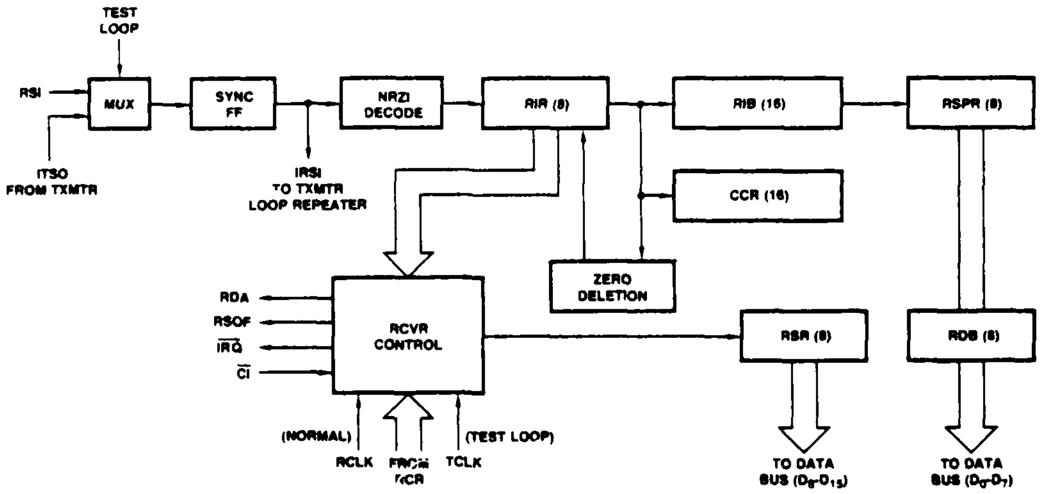


Fig. 2 RECEIVER DATA PATH

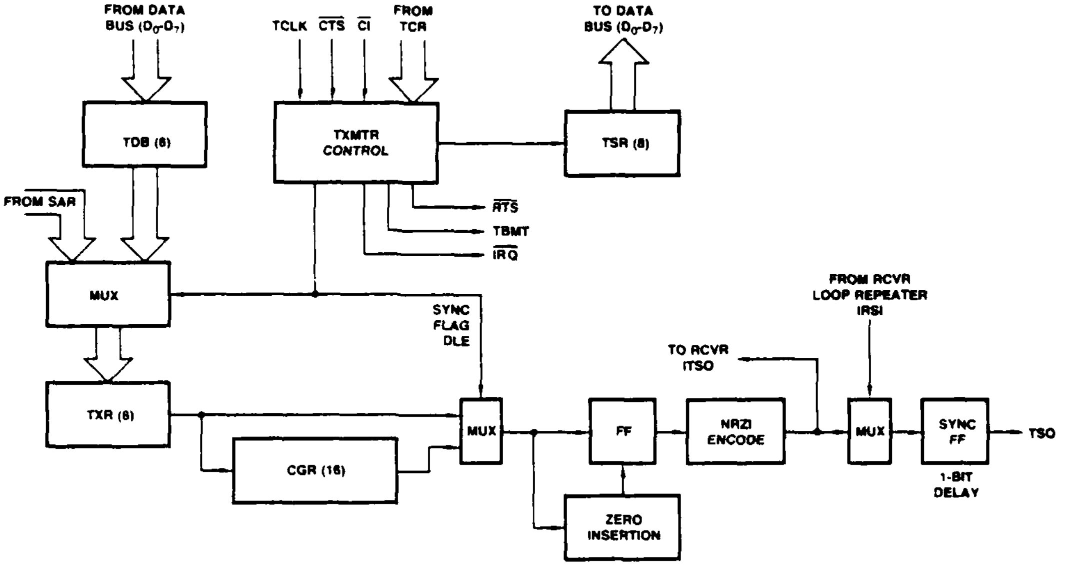


Fig. 3 TRANSMITTER DATA PATH

INPUT/OUTPUT DESIGNATIONS

NAME	TYPE	FUNCTION
D_0-D_{15}	I/O	DATA BUS: D_0-D_{15} contain bidirectional data status and control information to and from the CPU. D_0-D_7 may be Wired-OR to D_8-D_{15} for use as an 8-bit data bus.
A_0-A_2	I	REGISTER ADDRESS: A_0-A_2 select internal data, status and control registers. The internal registers may be selected as eight or 16 bits. See Register Address section.
BYTE	I	BYTE. A HIGH level indicates an 8-bit data bus. A LOW level indicates a 16-bit bus
$\overline{CE}$	I	CHIP ENABLE: A LOW level enables a data bus transfer with DBE
*R/$\overline{W}$	I	READ/$\overline{WRITE}$. A HIGH level allows data from the addressed register to be output to the data bus. A LOW level allows data from the bus to be loaded into the addressed register
*DBE	I	DATA BUS ENABLE: A strobe on this input causes information transfer between the data bus and the addressed register when the $\overline{CE}$ input is LOW
$\overline{CI}$	I	CHIP INITIALIZE: A LOW level initializes the internal control registers and timing.
RCLK	I	RECEIVER CLOCK: RCLK provides timing for the receiver logic. RCLK frequency is the same as the received baud rate.
RSI	I	RECEIVED SERIAL INPUT: RSI is the received serial data. Data changes on the positive going edge of RCLK
TCLK	I	TRANSMITTER CLOCK: TCLK provides timing for the transmitter logic. TCLK frequency is the same as the transmitted baud rate
TSO	O	TRANSMITTER SERIAL OUTPUT: TSO is the transmitted serial data. Data changes on the positive going edge of TCLK.
RDA	O	RECEIVER DATA AVAILABLE: A HIGH level indicates an assembled character is in the Receiver Buffer. RDA is reset on the trailing edge of DBE when the Receiver Buffer is read by the CPU
RSOF	O	RECEIVED SYNC OR FLAG: RSOF is HIGH for one receiver clock period each time a received SYNC or FLAG is detected.
TBMT	O	TRANSMITTER BUFFER EMPTY: A HIGH level indicates the device is ready to receive new data and/or control information from the CPU. TBMT is reset on the trailing edge of DBE when the Transmitter Buffer is loaded.

INPUT/OUTPUT DESIGNATIONS (Cont'd.)

NAME	TYPE	FUNCTION
$\overline{\text{IRQ}}$	O	INTERRUPT REQUEST A LOW level indicates an error has occurred as a result of a Receiver Overrun (ROVR) or Transmitter Underrun (TUR) ROVR occurs if the CPU fails to read data from the Receiver Buffer before it is overwritten by the next assembled character. TUR occurs if the CPU fails to load the Transmitter Buffer within one character time after TBMT goes HIGH. $\overline{\text{IRQ}}$ is reset on the trailing edge of DBE when the Receiver Status is read for an ROVR or the Transmitter Status for a TUR.
$\overline{\text{DTR}}$	O	DATA TERMINAL READY The $\overline{\text{DTR}}$ output is general purpose in nature It can be set LOW by programming the appropriate bit of the Receiver Control Register
$\overline{\text{DSR}}$	I	DATA SET READY The $\overline{\text{DSR}}$ input is general purpose in nature It can be tested by the CPU by reading the Transmitter Status Register
$\overline{\text{CD}}$	I	CARRIER DETECT The $\overline{\text{CD}}$ input is general purpose in nature It can be tested by reading the Transmitter Status Register
$\overline{\text{RTS}}$	O	REQUEST TO SEND: $\overline{\text{RTS}}$ is used with $\overline{\text{CTS}}$ to enable the transmitter It may be set LOW by programming the appropriate bit of the Transmitter Control Register
$\overline{\text{MISC}}$	O	MISCELLANEOUS: The $\overline{\text{MISC}}$ output is general purpose in nature It can be set LOW by programming the appropriate bit of the Receiver Control Register.
$\overline{\text{CTS}}$	I	CLEAR TO SEND $\overline{\text{CTS}}$ is used with $\overline{\text{RTS}}$ to enable the transmitter It can be tested by reading the Transmitter Status Register
V_{CC}	I	POWER SUPPLY INPUT +5 V
V_{SS}	I	GROUND 0 V reference
†$\overline{\text{RD}}$	I	READ PULSE Pulse (negative) on this input with address and $\overline{\text{CE}}$ transfers the addressed data register contents to the data bus.
†$\overline{\text{WR}}$	I	WRITE PULSE Pulse (negative) on this input with address and $\overline{\text{CE}}$ transfers the data bus information to the addressed register

*Pin label for F6856
†Pin label for F3846

ERROR CONTROL

BOP	A Frame Check Sequence (FCS) is transmitted/received as a 16-bit character following the last data character of a frame. The CRC polynomial used to generate/check the FCS is CRC-CCITT ($X^{16} + X^{12} + X^5 + 1$) with the dividend preset to "0" or "1"s.
BISYNC	A Block Check Character (BCC) is transmitted/received as a 16-bit character following an ITB, ETB or ETX character. The CRC polynomial used to generate/check the BCC is either CRC-16 ($X^{16} + X^{15} + X^2 + 1$) or CRC-CCITT with the dividend preset to "0"s.
BCP	A BCC, twice the data character length is transmitted/received following the last data character of a message if CRC is selected. The CRC polynomial used to generate/check the CRC changes with character length. These polynomials are listed below 5 Bit $X^{10} + X^9 + X^3 + X^2 + 1$ 6 Bit $X^{12} + X^{11} + X^3 + X^2 + 1$ (CRC-12) 7 Bit $X^{14} + X^{12} + X^4 + X^2 + 1$ 8 Bit CRC-16 or CRC-CCITT The dividend is always preset to "0"s.

REGISTER DEFINITIONS

	BITS	DESCRIPTION
ADDRESSABLE		
MCSA Mode Control Sync/Address	16	The upper eight bits (MCR) contain mode control information common to the receiver and transmitter. The lower eight bits (SAR) contain the programmed SYNC character in BCP or the secondary address in BOP. It is not used in Bisync mode.
TCDR Transmitter Control and Data Register	16	The upper eight bits (TCR) contain control information specifically for the transmitter. The lower eight bits (TDB) contains the data character to be transmitted.
RCTS Receiver Control and Transmitter Status Register	16	The upper eight bits (RCR) contain control information specifically for the receiver. The lower eight bits (TSR) contain transmitter and modem status information.
RSDR Receiver Status and Data Register	16	The upper eight bits (RSR) contain receiver status information. The lower eight bits (RDB) contains the assembled received character.
INTERNAL RECEIVER		
RIR Receiver Input Register	8	RIR, RIB, RSPR are used for character assembly and CCR is used to check for received CRC error.
RIB Receiver Input Buffer	16	
RSPR Receiver Serial to Parallel Register	8	
CCR CRC Check Register	16	
INTERNAL TRANSMITTER		
TXR Transmitter Shift Register CGR CRC Generation Register	8	TXR is used to convert parallel data from TDB to a serial output. CGR generates the transmitted CRC check sequence.

SHORT FORM REGISTER FORMAT

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCSA	PROTOCOL SELECT			LRSS	CC	NRZI	LOOP	EC	SYNC/SECONDARY ADDRESS							
								MCR ←—	—→ SAR							
TCDR	SOM	TACG	GATD	EOM	RTS	TCL_2-TCL_0			TRANSMITTER DATA							
								TCR ←—	—→ TDB							
RCTS	DTR	MISC	NOT USED		CRC	RE	RCL_1-RCL_0		TUR	TBMT	TOR	NOT USED		CTS	CD	DSR
								RCR ←—	—→ TSR							
RSDR	ROVR	RDA	REOM	ABGA	RDL_2-RDL_0			RERR	RECEIVED DATA							
								RSR ←—	—→ RDB							

SPECIAL CHARACTERS

CHARACTER	BIT PATTERN	FUNCTION
BOP		Frame
FLAG	01111110	Message
ABORT	11111111 Generated 11111110 Detected	Terminate a message prematurely
GA	11111110	Close frame in Store Loop Mode
ADDRESS	SAR	Secondary station address
BISYNC		
SYNC	00010110 USASCII 00110010 EBCDIC	Start a message and fill character
PAD	11111111	End of frame pad
DLE	00010000	Data link escape
SOH	00000001	Start of heading
STX	00000010	Start of text
ITB	00011111	End of intermediate transmission block
ETB	00010111 USASCII 00100110 EBCDIC	End of transmission block
ETX	00000011 ETX	End of transmission
BCP		
SYNC	SAR	Start a message and fill character
PAD	11111111	End of frame pad, selectable fill character for DDCMP

FUNCTIONAL DESCRIPTION – The SPCC is functionally partitioned into receiver, transmitter, addressable registers and data bus control. Figure 1 is a block diagram of the SPCC. Figures 2 and 3 show the data flow in the receiver and transmitter respectively.

RECEIVER OPERATION

GENERAL – The Mode Control Sync Address Register (MCSA) must be programmed prior to starting receiver operation. The receiver may then be enabled and the character length established by programming the receiver control register (RCR). Once the receiver is enabled, data on the RSI input will be serially shifted into the Receiver Input Register (RIR). Data is decoded from NRZI to NRZ as it is continuously monitored (on a bit-for-bit basis) for a match with the FLAG (BOP) or SYNC (Bisync or BCP) character. The RSOF output is set HIGH for one RCLK clock period when a match occurs. The receiver then operates as described below for each mode of operation.

BOP OPERATION – A flow chart of BOP receiver operation is shown in Figure 4. The receiver starts assembling characters and accumulating the CRC immediately after the detection of a FLAG. It also continues to search for additional FLAG, ABORT or GA characters on a bit for bit basis. Zero deletion (to remove "0"s added to the data stream after five consecutive "1"s to distinguish data from FLAG, ABORT and GA) is implemented in the RIR after the FLAG detection logic.

Assembled characters are shifted through the Receiver Input Buffer (RIB) into the Receiver Serial-to-Parallel Register (RSPR) and transferred to the Receiver Data Buffer (RDB). The RDA output and status bit are set HIGH each time data is transferred to RDB. Receiver data should be read by the CPU before the next character is assembled to prevent an overrun, resulting in loss of data. The $\overline{IRQ}$ output will go LOW and the ROVR status bit will be set if an overrun occurs.

Character length assembly is set at eight bits per character at the start of each frame. It remains at eight bits until the address and control fields (See Figure 5) have been processed. Character length switches to the programmed length at the start of the information field, if any, until the closing FLAG, ABORT or GA is detected. The length of the address field is determined by monitoring the least significant bit (LSB) of each address character for a logic "1". The last character of the address field has a "1" in the LSB. The length of the control field is one or two bytes as programmed in the MCR.

Character assembly and CRC accumulation are stopped when a closing FLAG, ABORT or GA is detected. REOM, ABGA (if the closing character was an ABORT or GA), RDL_0-RDL_2 (indicating length of last character) and RERR (if the accumulated CRC is incorrect) status bits are set. The last character is transferred to RDB and the RDA output is set HIGH.

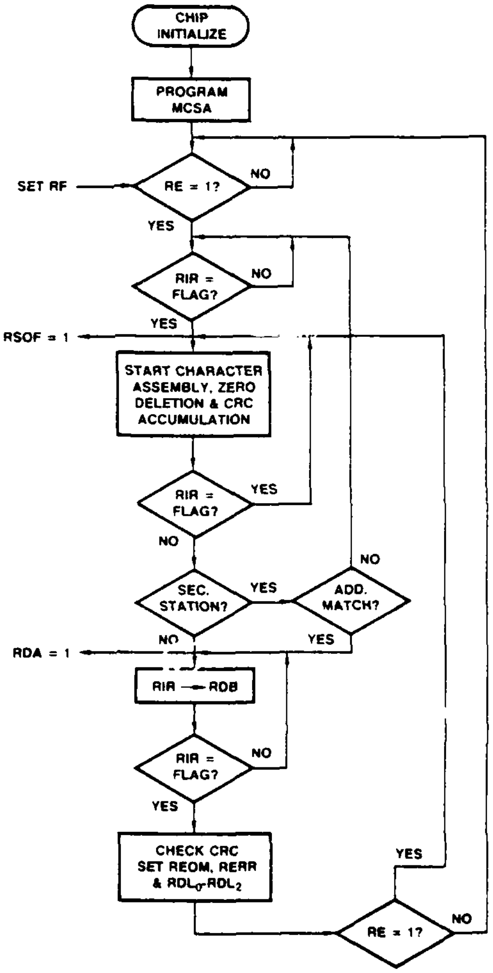


Fig. 4 BOP RECEIVE

FLAG	ADDRESS FIELD	CONTROL FIELD	INFORMATION FIELD (IF ANY)	FCS	FLAG
	n 8-BIT BYTES	1 OR 2 8-BIT BYTES	0 to m BITS	16 BITS	

INCLUDED IN CRC ACCUM.

Fig. 5 BOP MESSAGE FORMAT

The CRC accumulation includes all characters following the opening FLAG through the frame check sequence (FCS). The contents of the CRC Check Register (CCR) are checked at the close of a frame if CRC is selected. If an error is detected, RERR status bit is set. Neither the FCS nor the closing FLAG are assembled and passed on to the CPU.

The receiver may be turned off after the status and last characters are read by the CPU by resetting the RE bit of RCR or it can be left active to receive additional frames.

The closing FLAG of one frame may be used as the opening FLAG of the next frame. Character assembly of the next frame starts with the first non-FLAG character. If the frame was closed with an ABORT or GA, an opening FLAG must be detected before character assembly of the next frame is started.

All receiver status bits except RDA are reset after the Receiver Status Register (RSR) is read by the CPU. The RDA output and status bit are reset when RDB is read by the CPU.

If secondary address is selected, the first non-FLAG character of a frame is compared to the contents of the SYNC/Address Register. Data for the frame is not passed on to the CPU if no address match occurs. When GLOBAL address is selected, an all '1s' address results in an address match.

LOOP REPEATER OPERATION – Loop Repeater Mode is a special case of BOP. Receiver operation is the same as for BOP except the NRZI decode logic is disabled, frames may be terminated by a GO-AHEAD or FLAG, and received data and GA are routed to the transmitter. RCLK and TCLK should be tied together in this mode.

BISYNC OPERATION – A flow chart of Bisync receiver operation is shown in Figure 6. Characters in Bisync mode may be either EBCDIC or USASCII as programmed in the MCR. Character length defaults to eight bits. The eighth bit, when USASCII is programmed may be used for odd parity by the CPU. It is ignored in the recognition of the USASCII characters.
Character assembly starts after receipt of two continuous SYNC characters and continues until the receiver is turned off by resetting the RE bit of RCR. Assembled characters are shifted through the RIB to the RSPR and transferred to the RDB. The RDA output and status bit are set HIGH each time a character is transferred to the RDB. All characters which match the SYNC character in non-transparent mode and DLE SYNC pairs (if not immediately preceded by an odd number of DLE's) in transparent mode are *excluded from* the RDB. However, the RSOF output goes HIGH for one RCLK clock period each time a SYNC character is detected.

Data must be read by the CPU each time the RDA output goes HIGH before the next character is assembled to prevent an overrun, resulting in loss of data. The $\overline{\text{IRQ}}$ output goes LOW and the ROVR status bit is set if an overrun occurs.

The receiver always starts operation in the non-transparent mode. It switches to transparent mode if a DLE STX character pair is received. The receiver will then remain in transparent mode until a DLE ITB, DLE ETB or DLE ETX (if not immediately preceded by an odd number of DLE's) character pair is received.

CRC accumulation begins after the first non-SYNC character if the first character is an SOH or STX. It begins after the second non-SYNC character and enters transparent mode if the first two non-SYNC characters are DLE STX. SYNC characters in non-transparent mode or DLE SYNC pairs in transparent mode are excluded from the CRC accumulation. The first DLE of a DLE DLE sequence and the DLE of DLE ITB, DLE ETB or DLE ETX sequences are not included in the accumulation. The CRC is checked for 0000 remainder after receipt of an ITB, ETB or ETX in non-transparent mode or DLE ITB, DLE ETB or DLE ETX in transparent mode. The REOM and RERR (a non-zero remainder is detected) status bits are set when the closing character is transferred to the RDB and RDA is set HIGH. The block check character (BCC) following the closing character is passed to the CPU as the next two characters. If the closing character was an ETB or ETX, the receiver should be reset by dropping the RE bit of RCR. If the closing character was an ITB, CRC accumulation and character assembly will start again on the first character following the BCC.

All receiver status bits except RDA are reset each time RSR is read by the CPU. The RDA output and status bit are reset each time RDB is read by the CPU.

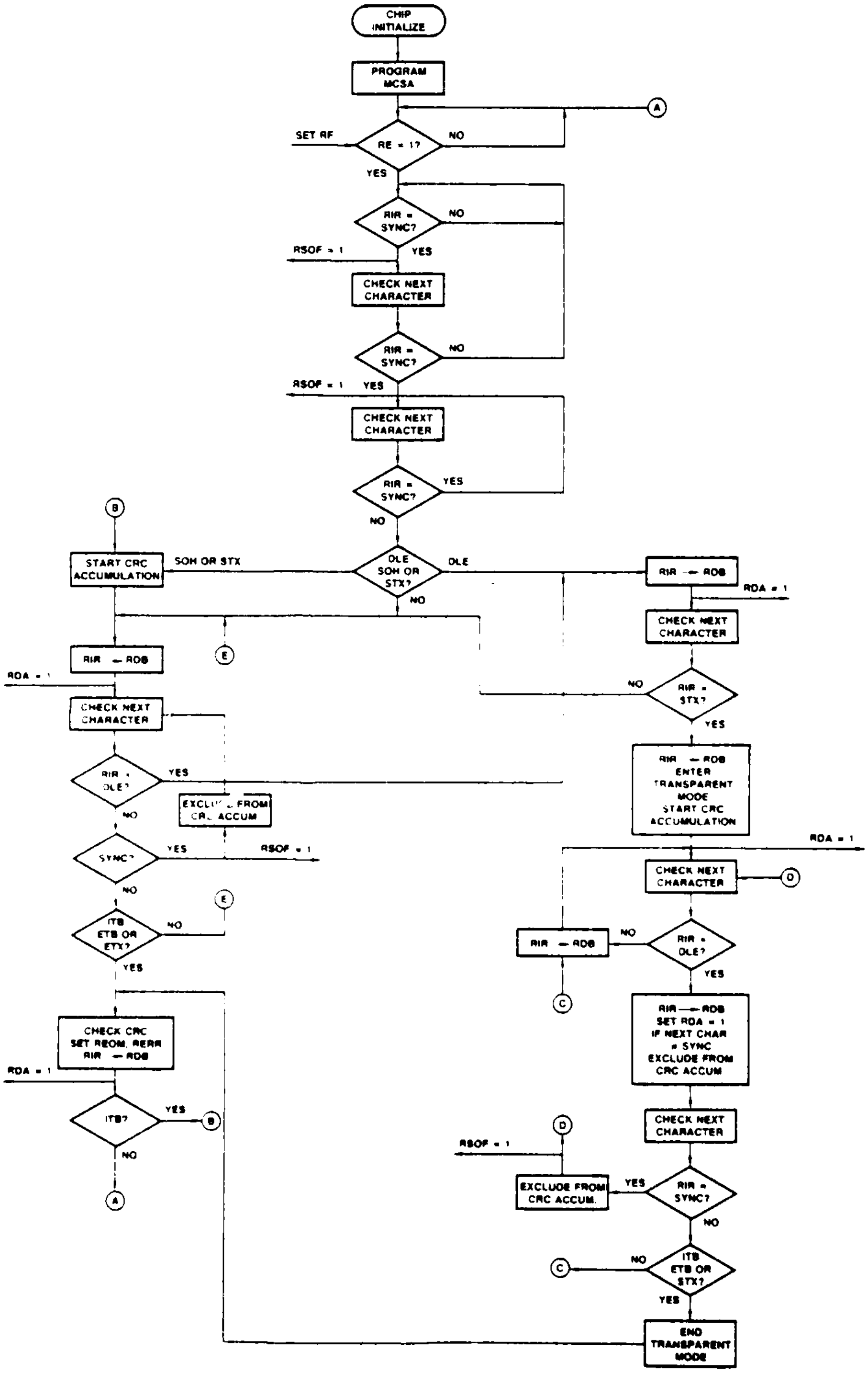
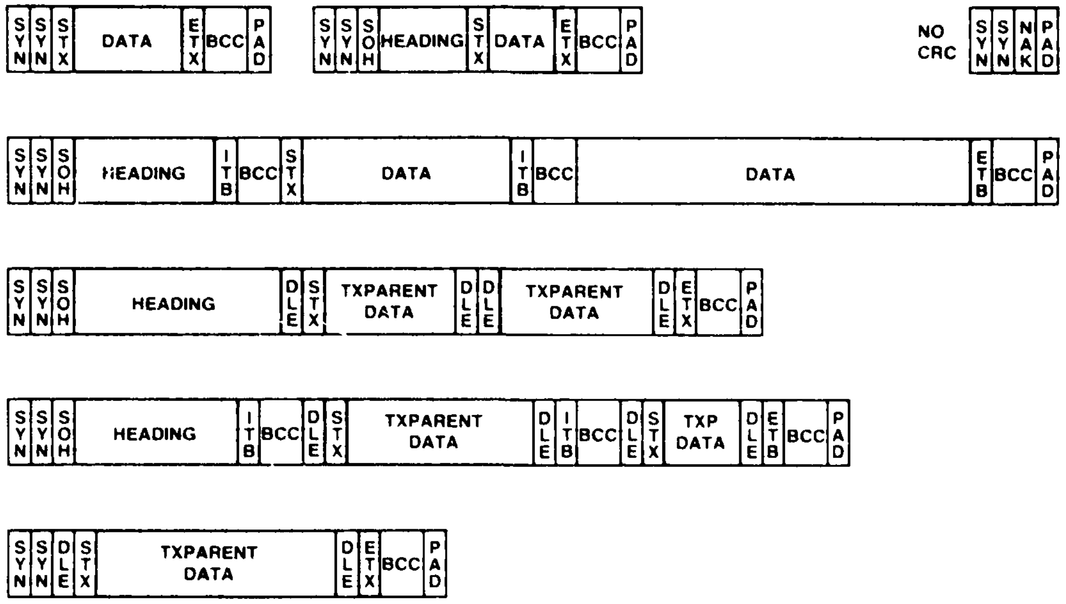


Fig. 6 BISYNC RECEIVE


SYN SYN STX DATA ETX BCC PAD
SYN SYN SOH HEADING STX DATA ETX BCC PAD
NO CRC
SYN SYN NAK PAD
SYN SYN SOH HEADING ITB BCC STX DATA ITB BCC DATA ETB BCC PAD
SYN SYN SOH HEADING DLE STX TXPARENT DATA DLE DLE TXPARENT DATA DLE ETX BCC PAD
SYN SYN SOH HEADING ITB BCC DLE STX TXPARENT DATA DLE ITB BCC DLE STX TXP DATA DLE ETB BCC PAD
SYN SYN DLE STX TXPARENT DATA DLE ETX BCC PAD

Shaded area included in CRC accumulation

Fig. 7 BISYNC MESSAGE FORMAT

BCP OPERATION - The flow diagram for BCP mode other than BISYNC is shown in Figure 8. The SYNC character is programmed in Sync Address Register (SAR). All characters, including the SYNC character are the length specified in the Receiver Control Register (RCR).

Character assembly starts after receipt of two contiguous SYNC characters and continues until the receiver is turned off by resetting the RE bit of RCR. Assembled characters are shifted through the RIB to the RSPR and transferred to the RDB. The RDA output and status bit are set HIGH each time an assembled character is transferred to the RDB. All characters which match the SYNC character are excluded from the RDB, if SYNC strip has been programmed. Only leading SYNC characters are excluded from the RDB if sync stripping has not been programmed. However, the RSOF output goes HIGH for one RCLK clock period each time a SYNC character is detected.

Data must be read by the CPU each time the RDA output goes HIGH before the next character is assembled. If not, an overrun will occur resulting in loss of data. The $\overline{IRQ}$ output goes LOW and the ROVR status bit is set if an overrun occurs.

CRC accumulation begins with the first non-SYNC character and includes all subsequent characters if sync strip is not programmed. The CRC accumulation will include only non-SYNC characters if sync strip is programmed. The CRC accumulation is checked each character time and the RERR status bit is set if the remainder does not equal "0" or reset if the remainder equals "0." Since there is not defined end of message character, the REOM status bit is not set. The CPU must determine when the end of message occurs and check the RERR status at that time. If an error free message has been received, RERR will be "0" for one character time. RE should be dropped, thereby resetting the receiver, after the last character has been read.

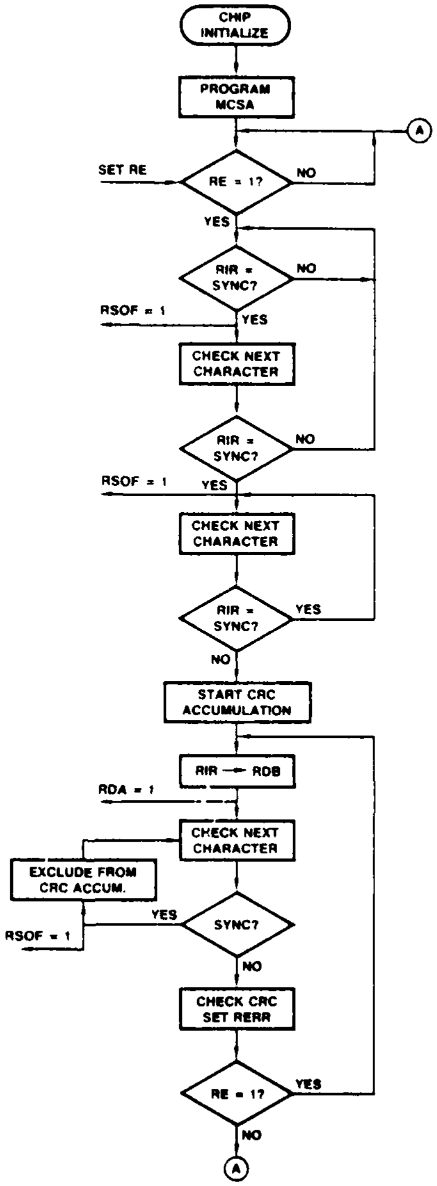


Fig. 8 BCP RECEIVE

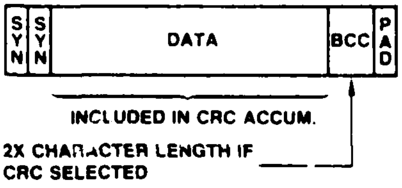


Fig. 9 BCP MESSAGE FORMAT

TRANSMITTER OPERATION

GENERAL – The Mode Control Sync/Address Register (MCSA) must be programmed prior to starting transmitter operation. The RTS bit of the Transmitter Control Register (TCR) must be set to turn on the transmitter. The SOM bit of TCR may also be set at this time and the Transmitter Data Buffer (TDB) loaded with the first character of the message. When $\overline{RTS}$ has been loaded into TCR, the $\overline{RTS}$ output goes LOW. The TSO output is held HIGH (marks) until the $\overline{CTS}$ input *goes LOW. Two* SYNC or FLAG characters are then outputted on TSO, if SOM has been set. Otherwise TSO will continue to output marks until SOM is set and the first character is loaded into TDB. Transmitter operation after the two SYNC or FLAG characters have been outputted depends on the mode of operation. Note, RTS and transmitter character length must be reloaded each time TCR is updated until after the EOM (end of message) bit has been set.

BOP OPERATION – Character length in BOP mode always starts at eight bits per character each frame. It remains eight bits until the address and control fields have been transmitted. It then switches to the programmed length at the start of the information field, if any, until the last character has been transmitted. Character length switches back to eight bits for the transmission of the Frame Check Sequence (FCS) and the closing FLAG.

A flow diagram for BOP transmitter operation is shown in Figure 10. The secondary address is transmitted after the initial two FLAGs. The secondary address comes from the Sync/Address Register (SAR) if the device is programmed as a secondary station or from the TDB if the device is programmed as a primary. If the secondary address came from SAR, it is followed in the transmission by the character from TDB. Characters are transferred in parallel from SAR or TDB to the Transmitter Shift Register (TXR) and serially shifted, LSB first, out the TSO output. The TBMT output and status bit are set HIGH each time data is transferred from TDB. The CPU must update TCR, if required, and load TDB with the next character. An underrun occurs if this is not done within one character time. If an underrun occurs, the TUR status bit is set and an ABORT (11111111) is transmitted. The output is held at a mark until SOM is set for a new message. A transmitter overrun occurs if TDB is updated before TBMT goes HIGH. An overrun can result in the misinterpretation or loss of the character in TDB. The TOR status bit is set when an overrun occurs.

The least significant bit (LSB) of each character, starting with the secondary address is examined. The first character with an LSB = "1" denotes the last character of the address field. The next one or two characters *(programmed in MCR) are the control* field. The character length switches to the programmed length in TCR after the last character of the control field unless that character was the end of message.

The CPU must set the EOM bit of TCR when loading the last character of the message. Character length may be changed at this time to allow transmission of a residual last character. The character in TDB is followed by the FCS (if CRC is selected) and a closing FLAG when EOM is set. The transmitter may be turned off by resetting RTS after TBMT goes HIGH or it may remain active. The closing FLAG of one frame may be used as the opening FLAG of the next frame by setting SOM and loading TDB after TBMT goes HIGH. If the transmitter is left active and SOM has not been set, FLAG characters are transmitted between frames if the GATD bit of TCR equals "0" or marks if GATD equals "1".

A message may be terminated at any time with an ABORT by setting the TACG bit of TCR. This causes the TSO output to go immediately to a mark condition until SOM is set.

Data transmitted on the TSO output is continuously monitored for five consecutive "1s." A "0" is inserted in the data stream each time this condition occurs. This insures a data character will not be interpreted as a FLAG, ABORT or GA at the received end.

TUR and TOR status bits are reset whenever the Transmitter Status Register (TSR) is read. The TBMT output and status bit are reset when TDB is loaded.

CRC accumulation begins with the first non-FLAG character and includes all subsequent characters up to and including the last data character. The accumulated CRC is then transmitted as the FCS following the last data character, if CRC is selected

LOOP REPEATER OPERATION – Loop Repeater Mode is a special case of BOP. The primary station in the loop should be programmed for normal BOP primary operation. The GATD bit of TCR is used to initiate a polling sequence. When this bit is set, marks are transmitted after the closing FLAG of a frame. The last "0" of the closing FLAG and the next seven "1s" are interpreted down loop as a GO-AHEAD. The end of the polling sequence is detected when the ABGA (received GA) bit of the RSR is set.

Down-loop stations should be programmed as BOP secondary, loop repeater (LRSS = "1" in MCR). In this mode, data received at the RSI input is delayed one bit *time and outputted on TSO. When data is to*

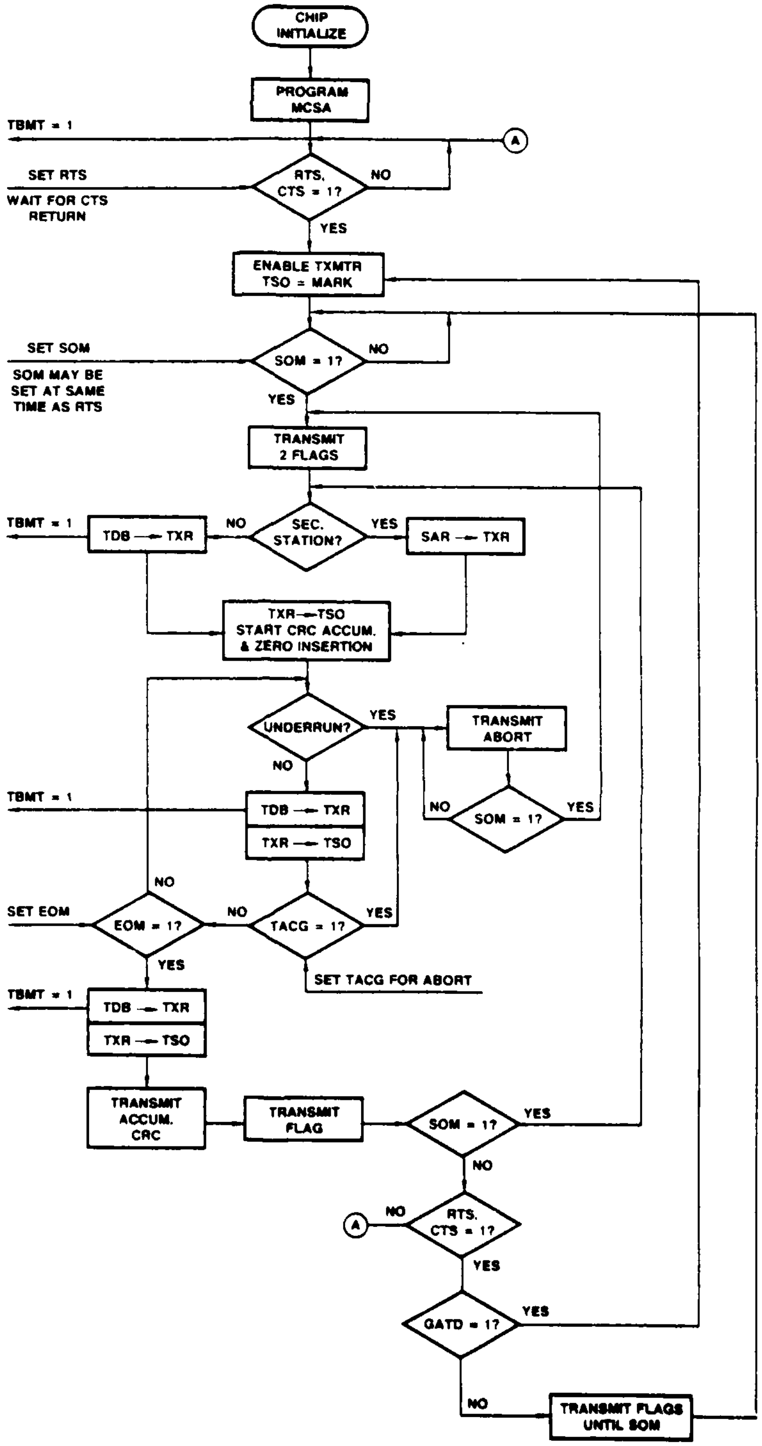


Fig. 10 BOP TRANSMIT

be transmitted in this mode, the CPU should set RTS and SOM and load the first character into TDB. CTS is ignored in this mode. The transmitter waits for a received GA. When a received GA is detected, the seventh "1" is changed to a "0," creating a FLAG. This prevents down-loop station from receiving a GA, reserving the line for the transmitting station. The TBMT output and status bit are set and transmitter operation proceeds in normal BOP operation except the NRZI encode logic is disable.

When the last character and FCS have been transmitted, the message is terminated with a GA. TSO switches back to RSI delayed one bit time. Down-loop stations may then capture the line by detecting the GA.

RCLK and TCLK should be tied together in this mode.

BISYNC OPERATION – A flow diagram for Bisync transmitter operation is shown in Figure 11. Character length for Bisync mode defaults to eight bits per character. The transmitter always assumes non-transparent mode unless forced to transparent by the CPU.

The message format following the initial SYNC pair depends on the action of the CPU. If the Transmitter Data Buffer (TDB) has not been loaded with the first character of the message, SYNC characters are outputted on TSO until a TDB load. This can occur only with an 8-bit data bus, since TCR and TDB are loaded simultaneously for a 16-bit data bus. The character from TDB, when available, is transferred to the Transmitter Shift Register, (TSR) and serially shifted out the TSO output. The character in TDB is preceded with a contiguous DLE when GATD (transmit DLE) is set. GATD bit is cancelled after it has been internally processed. The first occurrence is interpreted as a DLE STX command and the transmitter begins transparent mode operation. The transmitter will remain in transparent mode until the end of message.

The TBMT output and status bit are set HIGH each time data is transferred from TDB. The CPU should update TCR, if required, and load TDB with the next character. An underrun occurs if this is not done within one character time and the TUR status bit is set and SYNC characters (or DLE SYNC pairs in transparent mode) are transmitted until TDB is updated. A transmitter overrun occurs if TDB is updated before TBMT goes HIGH. An overrun can result in the misinterpretation or loss of the character in TDB. The TOR status bit is set when an overrun occurs.

The EOM bit of TCR, GATD (if in transparent mode) and TACG (if the accumulated CRC is to be transmitted as the Block Check Character) should be set when the last character is loaded into TDB. The last character must be an ITB, ETB or ETX if CRC is used. A 16-bit BCC, if selected, is transmitted following the last character. The last character is followed by marks for a minimum of one character time if no BCC is transmitted.

A second block of data may be transmitted immediately following the BCC by setting SOM and loading TDB after TBMT goes HIGH. The transmitter may be turned off at this time by resetting RTS. The transmitter transmits marks following the BCC for a minimum of one character time if SOM is not set.

CRC accumulation begins after the first non-SYNC character for non-transparent mode, or after the second non-SYNC character if the message starts in transparent mode. The CRC continues up to and including the last character. SYNC characters or DLE SYNC pairs caused by a transmitter underrun are not included. Forced DLE characters in transparent mode are not included. The forced DLE of a DLE STX pair which occurs after the start of the message is included. See Figure 7.

TUR and TOR status bits are reset whenever the Transmitter Status Register (TSR) is read. The TBMT output and status bit are reset when TDB is loaded.

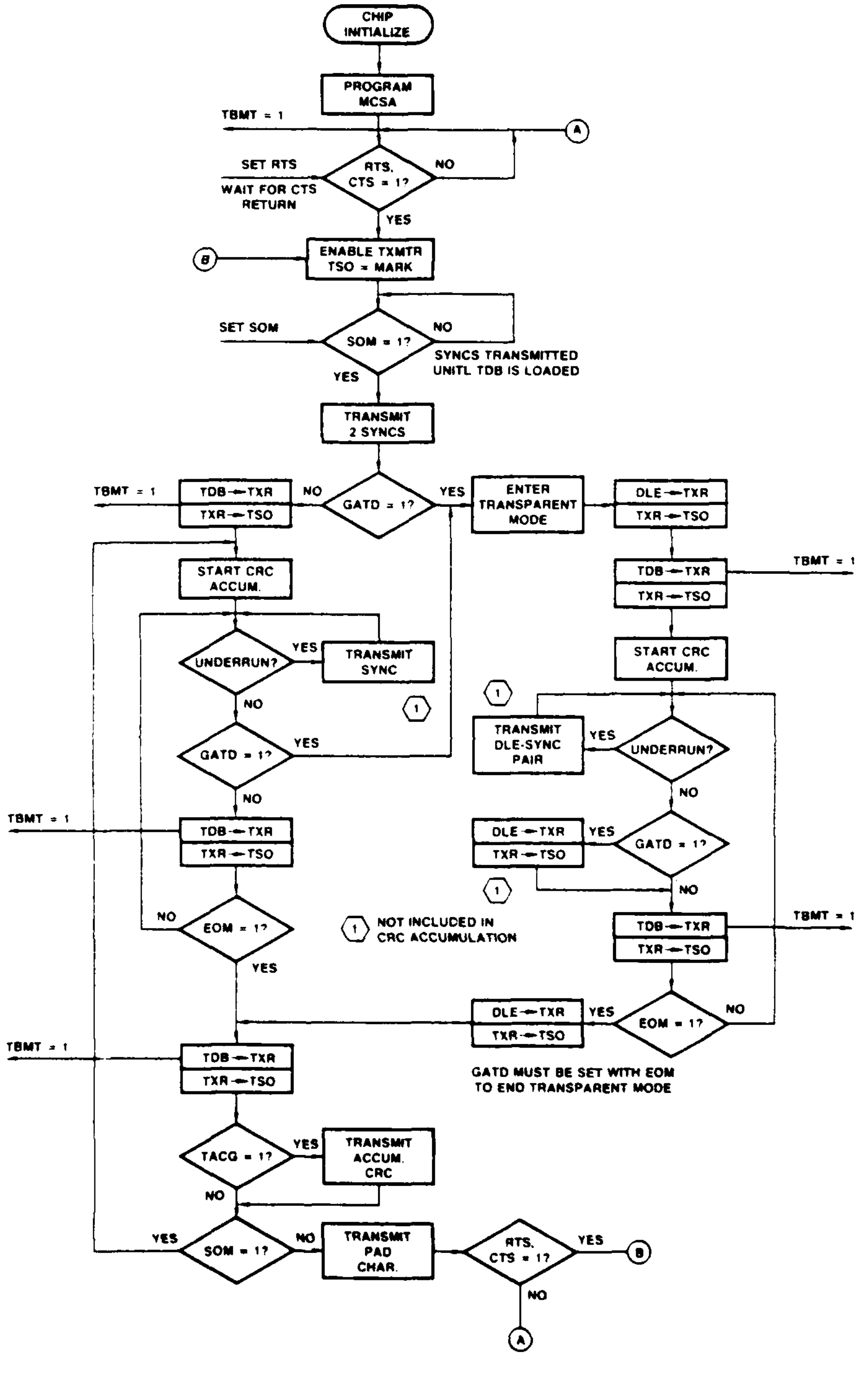


Fig. 11 BISYNC TRANSMIT

BCP OPERATION – The flow diagram for BCP mode other than Bisync is shown in Figure 12. The SYNC character is programmed in the Sync/Address Register (SAR). All characters are the length specified in the Transmitter Control Register (TCR).

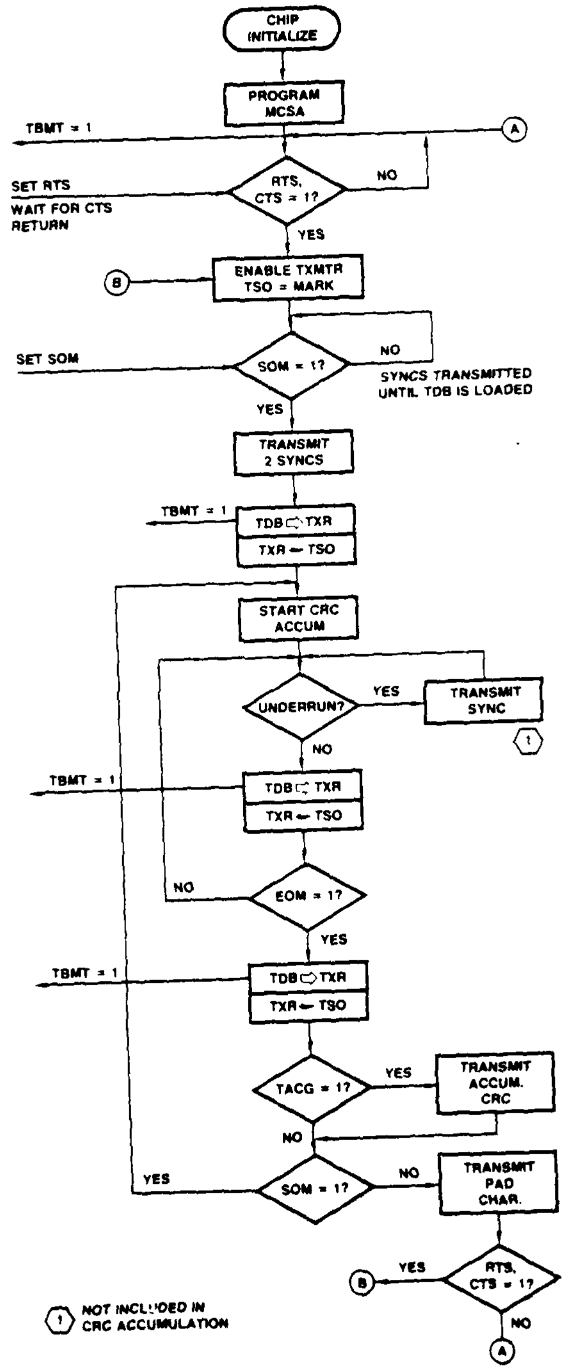


Fig. 12 BCP TRANSMIT

The message format following the initial SYNC pair depends on the action of the CPU. If the Transmitter Data Buffer has not been loaded with the first character of the message, SYNC characters are transmitted until a TDB load. This can occur only with an 8-bit data bus, since TCR and TDB are loaded simultaneously for a 16-bit data bus. The character from TDB, when available, is transferred to the Transmitter Shift Register (TSR) and serially shifted out the TSO output. The TBMT output and status bit are set HIGH each time data is transferred from TDB. The CPU should update TCR, if required, and load TDB with the next character. An underrun occurs if this is not done within one character time and the TUR status bit is set and SYNC characters (marks, if sync stripping is not programmed) are transmitted until TDB is updated. A transmitter overrun occurs if TDB is updated before TBMT goes HIGH. An overrun can result in the misinterpretation or loss of the character in TDB. The TOR status bit is set when an overrun occurs.

The EOM bit of TCR and TACG (if the accumulated CRC is to be transmitted as the Block Check Character) should be set when the last character is loaded into TDB. The last character is followed by a BCC and a pad character if CRC is selected, or the pad character only if CRC is not selected. The transmitter may be turned off by resetting $\overline{RTS}$ after TBMT goes HIGH.

CRC accumulation (See Error Control) includes all non-SYNC characters. The CRC Generation Register (CGR) in BCP mode is defined as twice the character length.

TUR and TOR status bits are reset whenever the Transmitter Status Register (TSR) is read. The TBMT output and status bit are reset when TDB is loaded.

DATA BUS CONTROL (6856)

The CPU uses the Register Address (A_0-A_2), Byte Select (BYTE), Chip Enable ($\overline{CE}$), Read/Write ($R/\overline{W}$), and Data Bus Enable (DBE) inputs to control information transfer on the data bus. The Byte Select input specifies a 16-bit data bus when BYTE = "0" or an 8-bit data bus when BYTE = "1." For an 8-bit data bus, D_0 through D_7 may be Wired-OR with the corresponding pins D_8 though D_{15}.

A read operation ($R/\overline{W}$ = "1") is initiated on the leading edge of DBE. The other control inputs (A_0-A_2, BYTE, $\overline{CE}$ and $R/\overline{W}$) must be stable before the leading edge of DBE (see Dynamic Characteristics). Any unused bits in the addressed register are "0." D_8-D_{15} contain receiver status when TSR is read using a 16-bit bus. Status bits are reset on the trailing edge of DBE, when the appropriate register is read.

Data is loaded into the addressed register on the trailing edge of DBE for a write ($R/\overline{W}$ = "0") operation. The other control inputs must be stable prior to the leading edge of DBE. TBMT is reset on the trailing edge of DBE when TCDR (16-bit bus) or TDB (8-bit bus) is addressed.

DATA BUS CONTROL (3846)

Bus control for the F3846 has the same characteristics as the F6856 with $\overline{RD}$ only for read rather than DBE = "1" and $R/\overline{W}$ = "1" and $\overline{WR}$ only for write rather than DBE = "1" and $R/\overline{W}$ = "0."

REGISTER ADDRESSES

	$R/\overline{W}$	A_0	A_1	A_2	REGISTER	$\overline{RD}$	$\overline{WR}$
BYTE = "0"	1	X	0	0	RSDR	0	1
16-BIT	0	X	1	0	TCDR	1	0
DATA BUS	0	X	0	1	MCSA	1	0
	1	X	1	1	$RCTS_L$ (TSR)	0	1
	0	X	1	1	$RCTS_U$ (RCR)	1	0
BYTE = 1	1	0	0	0	$RSDR_L$ (RDB)	0	1
8-BIT DATA	1	1	0	0	$RSDR_U$ (RSR)	0	1
BUS D_0-D_7	0	0	1	0	$TCDR_L$ (TDB)	1	0
WIRE OR'ED	0	1	1	0	$TCDR_U$ (TCR)	1	0
TO D_8-D_{15}	0	0	0	1	$MCSA_L$ (SAR)	1	0
	0	1	0	1	$MCSA_U$ (MCR)	1	0
	1	0	1	1	$RCTS_L$ (TSR)	0	1
	0	1	1	1	$RCTS_U$ (RCR)	1	0

PROGRAMMING

The Mode Control Sync Address Register (MCSA) is a directly addressable write only register used to configure the SPCC for the user's specific data communications environment. MSCA should be programmed after initialization and prior to initiating data transmission or reception. It may be changed at any time that both the receiver and transmitter are disabled. The default mode (after initialization) is BOP primary with one byte control field, NRZI encoding, 8-bit character length and error control using CRC-CCITT preset to "1s." The lower byte, sync/address, is not used in BOP primary mode.

The Transmitter Control and Data Register (TCDR) is a directly addressable write only register which controls the format of the transmitted data. The lower byte (TDB) contains the data characters to be transmitted. The upper byte (TCR) contains control information relating specifically to the data being transmitted. TCDR may be updated whenever the TBMT output is HIGH. The default mode for this register is all '0s' corresponding to transmitter disabled.

The upper byte (RCR) of the Receiver Control and Transmitter Status Register (RCTS) is a directly addressable write only register which contains control information specifically related to the receipt of data and the $\overline{DTR}$ and $\overline{MISC}$ general purpose outputs. Those bits which control the received character length should not be changed while the receiver is enabled. The default value of RCR is all 0s corresponding to receiver disabled and general purpose outputs at a HIGH level.

Specific definition of the format of the addressable registers is given in the following section. Address information is given in the Data Bus Control section.

ADDRESSABLE REGISTER FORMAT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PROTOCOL SELECT			LRSS	CC	NRZI	LOOP	EC	SYNC/SECONDARY ADDRESS							

MODE CONTROL SYNC/ADDRESS REGISTER (MSCA) – Write Only

BIT	NAME	MODE	FUNCTION
0-7	SAR		Sync/Address Register
		BOP	Secondary Address for secondary station mode
		Bisync	Not used
		BCP	SYNC Character
8	EC		Error Control
		BOP	0 = CCITT preset to all '0's 1 = CCITT preset to all '1's
		Bisync	0 = CRC-16 preset to all '0's 1 = CCITT preset to all '0's
		BCP	Same as Bisync for 8-bit characters length ONLY
9	LOOP	All	Self test loop mode. TSO loop to RSI internally
10	NRZI	All	0 = NRZ data 1 = NRZI, zero complementing
11	CC	BOP	0 = 1 control byte, 1 = 2 control bytes
		Bisync	Not used
		BCP	Not used
12	LRSS		Loop Repeater/Sync Strip
		BOP	0 = Normal mode 1 = Loop repeater mode
		Bisync	Not used
		BCP	0 = Tx Mark for FILL character Strip Leading SYNC's only 1 = Tx SYNC for FILL character Strip all SYNC's
13-15		All	Protocol Select 15 14 13 0 0 0 BOP, Primary 0 1 0 BOP, Secondary 0 1 1 BOP, Second, Global 1 0 0 BCP 1 1 0 Bisync – USASCII 1 1 1 Bisync – EBCDIC 0 0 1 Reserved 1 0 1 Reserved

ADDRESSABLE REGISTER FORMAT (Cont'd.)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SOM	TACG	GATD	EOM	RTS	$TCL_2 - TCL_0$			TRANSMITTER DATA BUFFER							

TRANSMITTER CONTROL AND DATA REGISTER (TCDR) – Write Only

BIT	NAME	MODE	FUNCTION
0-7	TDB	All	Transmitter Data Buffer
8-10	TCL_0-TCL_2		Transmitter Character Length
		BOP/BCP	8 9 10
			0 0 0 8-bits
			1 0 0 1
			0 1 0 2
			1 1 0 3
			0 0 1 4
			1 0 1 5
			0 1 1 6
			1 1 1 7
		BISYNC	Character length automatically 8-bits
11	RTS	All	Request to Send. "0" = "1" on $\overline{RTS}$ output; "1" = "0" on $\overline{RTS}$ output.
12	EOM	All	End of Message. "1" defines character in TBD as last data character of message. This bit is self-cancelling.
13	GATD		Go-ahead/Transmit DLE
		BOP	"0" = FLAGs transmitted between frames
			"1" = Marks transmitted between frames
		BISYNC	"1" = Transmit DLE character ahead of character in TDB. Enter transparent mode.
		BCP	Not used.
14	TACG		Transmit Abort/CRC Generate
		BOP	"1" = Transmit Abort
		BISYNC/	"0" = No CRC on transmitted message
		BCP	"1" = Transmit Block Check Character after last data character
15	SOM	All	Start of Message. Initiates start of message causing SYNCs or FLAGs to be transmitted. This bit is self-cancelling.

ADDRESSABLE REGISTER FORMAT (Cont'd.)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DTR	MISC	NOT USED		CRC	RE	RCL_1 - RCL_0		TUR	TBMT	TOR	NOT USED		CTS	CD	DSR

RECEIVER CONTROL AND TRANSMITTER STATUS REGISTER (RCTS) – Read/Wnte

BIT	NAME	MODE	FUNCTION
0	DSR	All	Data Set Ready. Equals "1" when $\overline{DSR}$ input is LOW
1	CD	All	Carner Detect. Equals "1" when $\overline{CD}$ input is LOW
2	CTS	All	Clear to Send. Equals "1" when $\overline{CTS}$ input is LOW
3-4			Not used
5	TOR	All	Transmitter Overrun. "1" = CPU updated TCDR before the SPCC was ready
6	TBMT	All	Transmitter Buffer Empty. "1" = CPU may load new data and/or Control information in TCDR.
7	TUR	All	Transmitter Underrun. "1" = CPU failed to load TDB in time. Abort is transmitted in BOP mode When TUR occurs fill characters are transmitted in BISYNC or BCP. TUR occurs along with a LOW level of $\overline{IRQ}$ output.
8-9	RCL_0-RCL_1	All	Receiver Character Length 8 9 0 0 8-bits 1 0 5 0 1 6 1 1 7
10	RE	All	Receiver Enable. "1" enables receiver
11	CRC	All	"0" = No CRC (Transmit/Receive) "1" = CRC selected
12-13			Not used
14	MISC	All	Miscellaneous. "0" = "1" on $\overline{MISC}$ output; "1" = "0" on $\overline{MISC}$ output.
15	DTR	All	Data Terminal Ready. "0" = "1" on $\overline{DTR}$ output; "1" = "0" on $\overline{DTR}$ output.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ROVR	RDA	REOM	ABGA	RDL_2 - RDL_0			RERR	RECEIVER DATA BUFFER							

RECEIVER STATUS AND DATA REGISTER (RSDR) – Read Only

BIT	NAME	MODE	FUNCTION
0-7	RDB	All	Receiver Data Buffer
8	RERR	All	Received Error. "1" = CRC error occurred on received message. Asserted when last character is in RDB.
9-11	RDL_0-RDL_2	BOP only	Received Last Character Length. Corresponds to the number of bits in last character 000 = 8 bits, 100 = 1 bit, 010 = 2 bits, etc.
12	ABGA	BOP only,	Abort/Go-Ahead Corresponds to received Abort if RERR = "1" or go-Ahead if RERR = "0"
13	REOM	 BOP BISYNC	Received End of Message "1" = Received FLAG, Abort or Go-Ahead "1" = Received ITB, ETB, or ETX (preceded by DLE in transparent mode)
14	RDA	All	Received Data Available. "1" indicates valid data available in RDB
15	ROVR	All	Receiver Overrun. "1" indicates CPU failed to read data in RDB before next character was assembled. Accompanied by a LOW on $\overline{IRQ}$ output.

ABSOLUTE MAXIMUM RATINGS

Operating Temperature	
Ceramic	−55°C to +125°C
Cermet	−55°C to +125°C
Plastic	0°C to +70°C
Storage Temperature	−65°C to +150°C
Supply Voltage	−0.3 V to +7.0 V
Input/Output Voltage	−0.3 V to +10 V
Input Voltage	−0.3 V to +15 V
Output Voltage	−0.3 V 'o +10 V

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages.

ELECTRICAL CHARACTERISTICS: Over the Operating Temperature Range

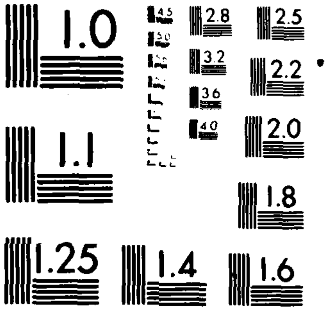
SYMBOL	PARAMETER	LIMITS			UNIT	CONDITIONS
		MIN	TYP	MAX		
V_{CC}	Supply Voltage	4.5	5.0	5.5	V	
	Input Voltage					
V_{IL}	Input LOW	−0.3		0.8	V	
V_{ILC}	Clock LOW	−0.3		0.8	V	
V_{IH}	Input HIGH	2.0		V_{DD}	V	
V_{IHC}	Clock HIGH	2.4		V_{DD}	V	
	Output Voltage					
V_{OL}	Output LOW			0.45	V	I_{OL} = 3.2 mA
V_{OH}	Output HIGH	2.4			V	I_{OH} = −600 μA
	Leakage Current:					
I_{LI}	Input Leakage			10	μA	
I_{LO}	Output Leakage			±10	μA	
I_{DD}	Supply Current			120	mA	V_{DD} = 5.5 V
	Capacitance					
C_I	Input			10	pF	Measured at 27°C
C_O	Output			15	pF	and 1 MHz
C_{IO}	Bus In			20	pF	

AD-A092 272 DELTA INFORMATION SYSTEMS INC JENKINTOWN PA F/G 9/2
USE OF A MICROPROCESSOR TO IMPLEMENT AN ADCCP PROTOCOL (FEDERAL--ETC(U)
JUL 80 S J URBAN, R SCHAPHORST DCA100-79-C-0050

UNCLASSIFIED NCS-TIB-80-7 NL

2 of 2
AD
A092272

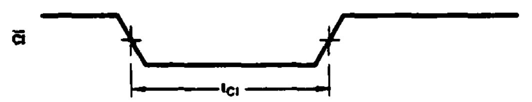
END
DATE
FILMED
1 81
DTIC


1.0
4.5
5.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

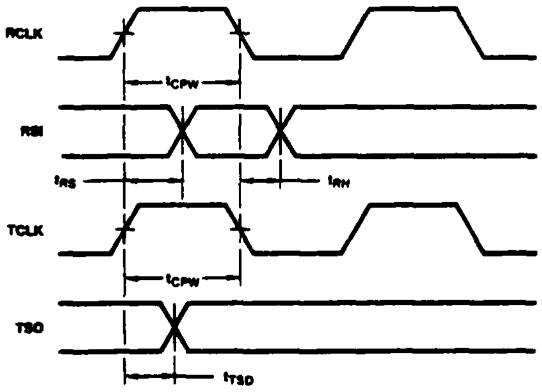
DYNAMIC CHARACTERISTICS: Over the Operating Temperature Range, V_{CC} = 5 V ±10%
C_L = 100 pF for D_0-D_{15}, C_L = 50 pF for all other outputs

SYMBOL	PARAMETER	LIMITS			UNITS
		MIN.	TYP.	MAX.	
	Set-up and Hold Time				
t_{ACS}	Address/Control Set-up	100			ns
t_{ACH}	Address/Control Hold	100			ns
t_{DS}	Data Bus Set-up (Write)	200			ns
t_{DH}	Data Bus Hold (Write)	100			ns
t_{RS}	RSI Set-up			200	ns
t_{RH}	RSI Hold	100			ns
	Pulse Width				
t_{CI}	CI	450			ns
$t_{DBE(\overline{RD})}$	DBE/$\overline{RD}$, $\overline{WR}$	450			ns
	Delay Time				
t_{DD}	Data Bus (Read)			250	ns
t_{TSD}	Transmit Serial Data			200	ns
t_{BRD}	Bus Release			150	ns
f	Clock Frequency			1.0	MHz
t_{CPW}	Clock Pulse Width	400			ns

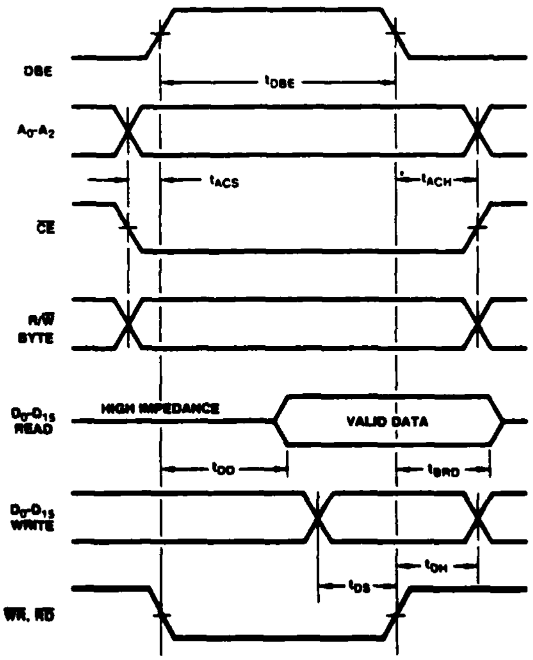
CHIP INITIALIZE



CLOCK AND SERIAL DATA

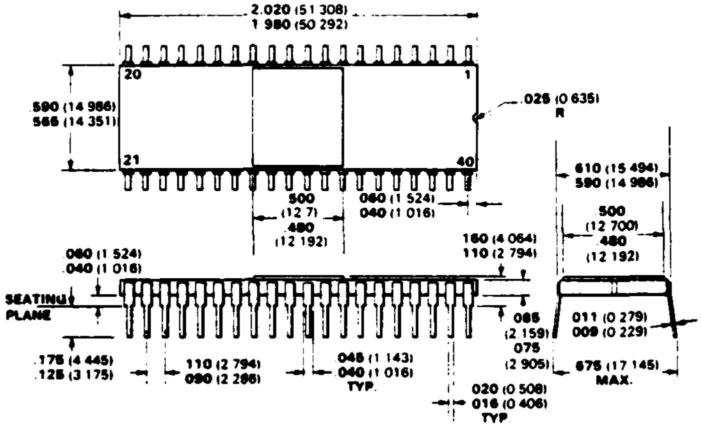


READ/WRITE DATA BUS



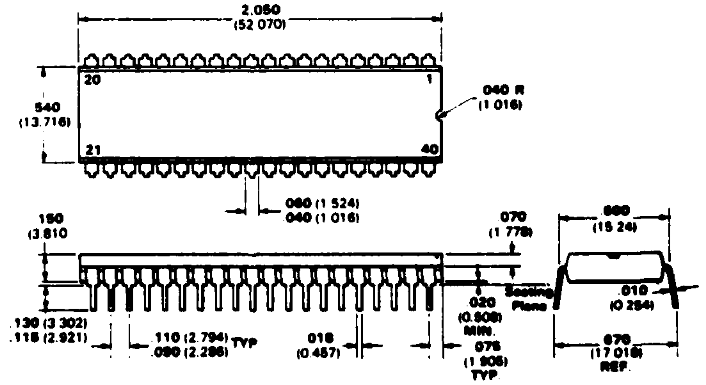
PACKAGE OUTLINE

40-PIN CERAMIC DUAL IN-LINE SIDE-BRAZED



NOTES:
All dimensions in inches (bold) and millimeters (parentheses)
Pin material nickel gold-plated kovar
Cap is kovar
Base is ceramic
Package weight is 6.5 grams

40-PIN PLASTIC DUAL IN-LINE PACKAGE



NOTES:
All dimensions in inches (bold) and millimeters (parentheses)
Pins are tin-plated kovar
Package material is plastic
Pins are intended for insertion in hole rows on 0.600" (15.42) centers
They are purposely shipped with "positive" mmisalignment to facilitate insertion.

APPENDIX B

LSI MICRO PACKET NETWORK INTERFACE - WD2501

WESTERN DIGITAL
CORPORATION

PRELIMINARY

LSI MICRO PACKET NETWORK INTERFACE (μPAC) WD2501 SHORT FORM DATA SHEET

FEATURES

- Packet Switching Controller Compatible with CCITT Recommendation X.25, Level 2, LAP.
- Programmable Primary Timer (T1) And Retransmission Counter (N2)
- Programmable A-Field Which Provides A Wider Range Of Applications Than Defined By X.25. These Include: DTE-To-DTE Connection, Multipoint, And Loop-Back Testing
- Direct Memory Access (DMA) Transfer: Two Channels; One For Transmit And One For Receive. Send/Receive Data Accessed By Indirect Addressing Method. No External Address Latches Required. Sixteen Output Address Lines.
- Zero Bit Insert And Delete
- Automatic Appending and Testing Of FCS Field
- Computer Bus Interface Structure: 8 Bit Bi-Directional Data Bus. CS, WE, RE-Four Input Address Lines
- DC To *1.6M Bits/SEC Baud Rate
- TTL Compatible
- 48 Pin Dual In-Line Packages
- Pin-for-pin compatible with WD2511 (LAPB.)

* Higher Baud Rates Available By Special Order

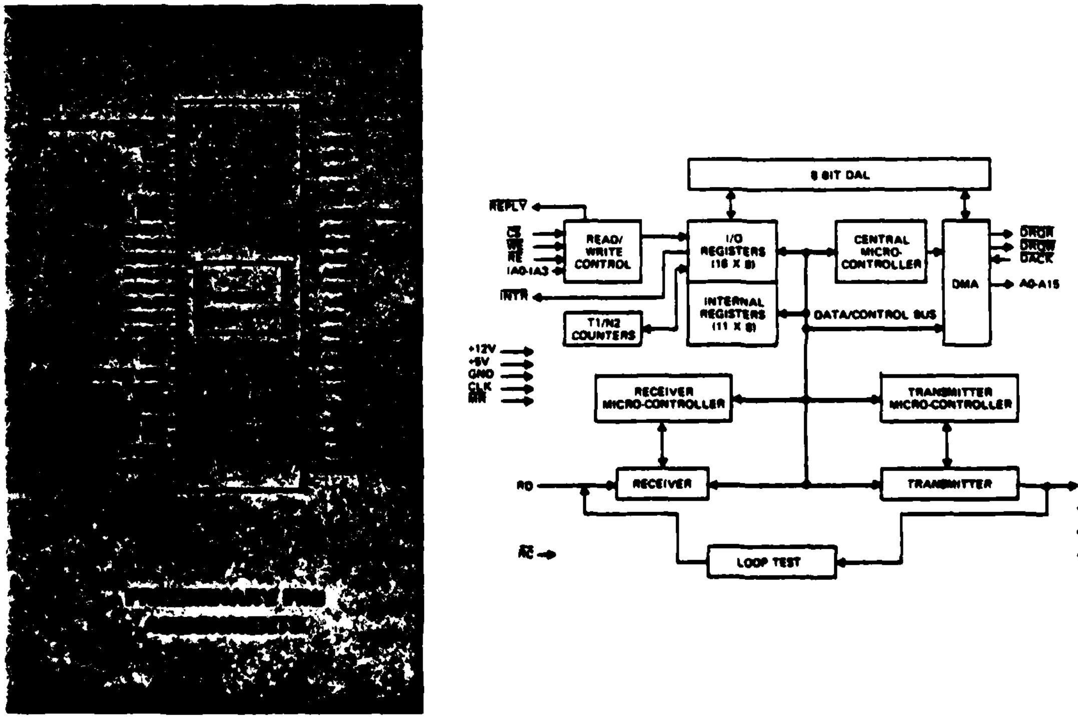
APPLICATIONS

X.25 PACKET SWITCHING CONTROLLER
PART OF DTE OF DCE
PRIVATE PACKET NETWORKS

GENERAL DESCRIPTION

The WD2501 is a MOS/LSI device which handles bit-oriented, full-duplex serial data communications with DMA, which conforms to CCITT X.25 with programmable enhancements.

The device is fabricated in N-Channel silicon gate MOS technology and is TTL compatible on all inputs and outputs.



WD 2501 BLOCK DIAGRAM

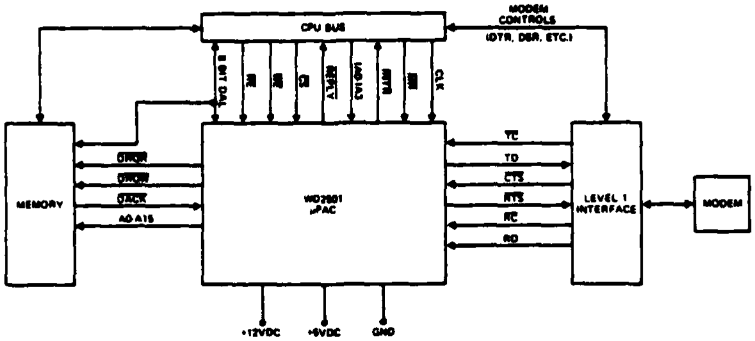
INTERFACE SIGNAL DESCRIPTION

*PIN NUMBER	SYMBOL	NAME	FUNCTION
48	VCC	Power Supply	+5VDC power supply input
42	VDD	Power Supply	+12VDC power supply input
18	VSS	Ground	Ground
6	CLK	Clock	Clock input used for internal timing. Must be square wave from 1.0 to 3.0 mHz.
7	$\overline{MR}$	Master Reset	Initialize on active low. All registers reset to zero, except control bits MDISC and $\overline{LINK}$ are set to 1. $\overline{DACK}$ must be stable high before MR goes high.
4	$\overline{CS}$	Chip Select	Active low chip select for CPU control of I/O registers.
8-15	DAL0-DAL7	Data Access Lines	An 8 bit bi-directional three-state bus for CPU and DMA controlled transfers.
5	$\overline{RE}$	Read Enable**	The contents of the selected register are placed on DAL when $\overline{CS}$ and $\overline{RE}$ are low.
3	$\overline{WE}$	Write Enable	The data on the DAL are written into the selected register when $\overline{CS}$ and $\overline{WE}$ are low. $\overline{RE}$ and $\overline{WE}$ must not be low at the same time.
2	$\overline{REPLY}$	Reply	An active low output to indicate that either a $\overline{CS}\bullet\overline{WE}$ or $\overline{CS}\bullet\overline{RE}$ input is present.
43	$\overline{INTR}$	Interrupt Request	An active low interrupt service request output, and returns high when Status Register #1 is read.
47-44	IA0-IA3	Address Lines In	Four address inputs to the 2501 for CPU controlled read/write operation with registers in the 2501. If ADRV = 0, these may be tied to A0 - A3.
26-41	A0-A15	Address Lines Out	Sixteen address outputs from the 2501 for DMA operation. If the control bit ADRV is 1, the outputs are TTL drives at all times. If ADRV is 0, the ouputs are 3-state, and are HI-Z whenever $\overline{DACK}$ is high. (ADRV is in Control Register #1.)
23	$\overline{DRQR}$	DMA Request Read	An active low output signal to initiate CPU bus request so the 2501 can output onto the bus.

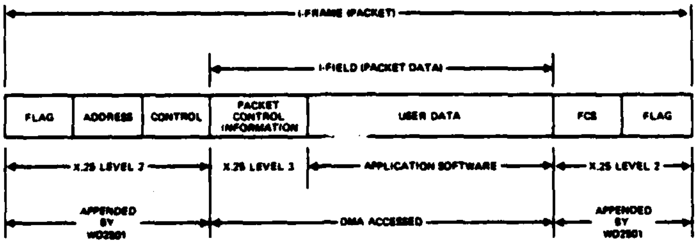
*PIN NUMBER	SYMBOL	PIN NAME	FUNCTION
24	$\overline{DRQW}$	DMA Request Write	An active low output signal to initiate CPU bus request so that data may be written into the 2501. $\overline{DRQW}$ and $\overline{DRQR}$ cannot be low at the same time.
25	$\overline{DACK}$	DMA Acknowledge	An active low input from the CPU in response to $\overline{DRQR}$ or $\overline{DRQW}$. $\overline{DACK}$ must not be low if $\overline{CS}$ and $\overline{RE}$ are low or if $\overline{CS}$ and $\overline{WE}$ are low.
21	TD	Transmit Data	Transmitted serial data output
16	RD	Receive Data	Receive serial data input
22	$\overline{TC}$	Transmit Clock	A 1X clock input. TD changes on the falling edge of $\overline{TC}$.
17	$\overline{RC}$	Receive Clock	If the NRZI control bit is 0, this is a 1X clock input, and RD is sampled on the rising edge of $\overline{RC}$. If the NRZI control bit is 1, this is a 32X clock input. Data is sampled according to the Digital Phase Locked Loop (DPLL). Adjustment of the sample is by quadrant. The sampling may be monitored by the $\overline{RCO}$ output.
19	$\overline{RTS}$	Request-To-Send	An open collector (drain) output which goes low when the 2501 is ready to transmit either flags or data. May be hard-wired to ground.
20	$\overline{CTS}$	Clear-To-Send	An active low input which signals the 2501 that transmission may begin. If high, the TD output is forced high. May be hard-wired to ground.

• PIN NUMBERS ARE PRELIMINARY

•• Throughout this document, the term "read" refers to data out of the 2501 and "write" refers to data going into the 2501.



SYSTEM CONNECTION



According to the X.25 protocol, there are three types of frames: supervisory (S-frame), unnumbered (U-frame), and information (I-frame). All S- and U-frames are generated and tested by the 2501. The user's CPU handles only the I-field of I-frames, which are packets.
NOTE: X.25 Level 1 is the physical interface.

FRAME FORMAT CONTROL

The WD2501 is controlled and monitored by sixteen I/O registers.
Control, status, and error bits will be referred to as CR, SR, or ER, respectively, along with two digits. For example, SR16 refers to status register #1 and bit 6, which is "XBA".

REG. #	IA3	IA2	IA1	IA0	REGISTER	REGISTER GROUPING
.0	0	0	0	0	CR0	OVERALL CONTROL AND MONITOR
1	0	0	0	1	CR1	
2	0	0	1	0	*SR0	
3	0	0	1	1	*SR1	
4	0	1	0	0	*SR2	
5	0	1	0	1	*ER0	
6	0	1	1	0	*CHAIN MONITOR	RECEIVER MONITOR
7	0	1	1	1	*RECEIVED C-FIELD	
8	1	0	0	0	T1	TIMER
9	1	0	0	1	N2/T1	
A	1	0	1	0	TLOOK H1	DMA SET-UP
B	1	0	1	1	TLOOK LO	
C	1	1	0	0	CHAIN/LIMIT	
D	1	1	0	1	(UNUSED)	
E	1	1	1	0	XMT COMMAND "E"	"A" FIELD
F	1	1	1	1	XMT RESPONSE "F"	

*CPU READ ONLY. (Write not possible)

CONTROL,STATUS,ERROR REGISTERS

REGISTER	7	6	5	4	3	2	1	0
CR0	0	0	0	ACTIVE/ $\overline{\text{PASSIVE}}$	LOOP TEST	0	RECR	MDISC
CR1	0	0	0	ADRV	RRT1	0	0	SEND
SR0	NA2	NA1	NA0	RNRR	NB2	NB1	NB0	RNRX
SR1	[1]PKR	[1]XBA	[1]ERROR		NE2	NE1	NE0	
SR2	T1OUT	$\overline{\text{IRTS}}$	REC IDLE				RANC	$\overline{\text{LINK}}$
ER0	ER07	ER06	ER05	ER04	ER03	ER02	ER01	ER00

[1] Causes Interrupt ($\overline{\text{INTR}}$ Goes Low).

BIT	DESCRIPTION
CR07	Unused control bits, like CR07, should be 0.
CR04	This bit will cause the 2501 to initiate link set-up if CR04 = 1, or to wait for a link set-up from the remote device if CR04 = 0.
CR03	The LOOP TEST bit will connect the transmitted data output to the receiver input. The receiver input pin, RD, is gated-out. The "E" and "F" registers of the A-field should be equal.
CR01	This bit is RECR which defines the CPU's receiver buffer as Ready (CR01 = 1) or as Not Ready (CR01 = 0). If RECR = 0, this bit indicates that the CPU has a temporary inability to accept more I-frames, or packets, and the 2501 will transmit an RNR S-frame.
CR00	MDISC is a mandatory disconnect command. MDISC will cause a logical disconnect in the DTE/DCE link. No DMA accessed data may be transferred as long as MDISC = 1. After Master Reset ($\overline{\text{MR}}$ pin transition from low to high), MDISC will be set. The 2501 will neither transmit nor accept received data until MDISC = 0.
CR14	The ADRV bit (CR14) is the control for the 16 bit output addresses (A0-A15). If ADRV = 0, the outputs are 3-state and are in HI-Z, except when $\overline{\text{DACK}}$ goes low. If ADRV = 1, the outputs are always low impedance (TTL), and are high when $\overline{\text{DACK}}$ is high.

BIT	DESCRIPTION
CR13	RRT1 will cause the 2501 to transmit an RR (RECR = 1) or RNR (RECR = 0) at T1 intervals provided the 2501 is not sending a command or waiting for an acknowledgement.
CR10*	The SEND bit (CR10) is used to command the 2501 to send the next packet or packets. If SEND = 1, the 2501 will read from TLOOK the BRDY bit of the next segment for transmission. If BRDY = 0, the 2501 will clear SEND and no action occurs. If BRDY = 1, the 2501 will then read TSADR and TCNT, followed by the transmission of that buffer. After transmission, the 2501 clears BRDY of the segment just transmitted, and reads BRDY of the next segment. If 1, the next segment is transmitted. If 0, the SEND bit is cleared, and transmission of packets is stopped.
SR07-SR05*	NA2-NA0. Next block of transmitted data to be Acknowledged.
SR04	RNRR. An RNR has been received.
SR03-SR01*	NB2-NB0. Next block to be transmitted.
SR00	RNRX. As a result of RECR (CR01) = 0, an RNR has been transmitted.
SR17	The PKR bit stands for Packet Received. This means that a packet has been received error-free and in correct sequence according to the received N (S) count. The data (I-field) has been placed in the CPU's RAM memory. NE is advanced. The three interrupt-causing bits are SR17, SR16, and SR15. Any of the three will cause an interrupt request ($\overline{INTR}$ goes low) when that bit goes to a 1. After SR1 is read, all three bits are reset to 0, and $\overline{INTR}$ returns high.
SR16	The XBA bit means that a previously transmitted Block, or Blocks, have been acknowledged by the remote device. Upon acknowledgement, the ACK'ED bit is set to "1" for each segment in TLOOK which was acknowledged.
SR15	The ERROR bit indicates: 1) An error has occurred which is not recoverable by the 2501, or 2) A significant event has occurred. The "significant events" are: change in link status (link-up or down), the 2501 is progressing to the next segment in a chained receive buffer, or one direction of the link has been reset. The exact nature of the reason for the ERROR bit is given in ER0.
SR13-SR11*	NE2-NE0. Next Expected packet segment number of RLOOK.
SR27	T1OUT bit means that timer T1 has timed-out. This bit returns to 0 when T1 is re-started.
SR26	$\overline{IRTS}$ stands for the Internal Request-To-Send bit, and indicates that the transmitter is attempting (successful or not) to send either data or flags. If the $\overline{RTS}$ pin is not tied to ground or WIRE-ORED with another signal, then $\overline{IRTS} = \overline{RTS}$.
SR25	RECIDLE indicates that the 2501 has received at least 15 contiguous 1's.
SR22	This bit is used to override CR05. It is possible for the 2501 to be programmed for LAP (CR05 = 0), but actually receive a link set-up for LAPB, or vice versa. SR22 shows which is actually being used. (Early versions of the 2501 are LAP only.)
SR21	RANC means that the Received Address field is Not Correct. Either the A-field was from "E" but should have been "F" or vice versa. A CMDR will be transmitted if link was in the information transfer phase. NOTE: If an A-field is neither "E" nor "F", the entire packet is disregarded and not brought into memory by DMA. No action is taken.
[illegible]	If the link is established, $\overline{LINK}$ = 0. If the link is logically disconnected, $\overline{LINK}$ = 1.

*See "Memory Access Method" Section

ERROR REGISTER (ER0)

ER07	ER06	ER05						
0	0	0	EROO = NOSFR ER01 = ROR ER02 = TUR ER03 = RPKNR ER04 = RLNR					
0	0	1	ER04	ER03	ER02	ER01	ER00	
			0	0	0	0	1	LINK is up. (Was down)
			1	0	0	0	0	Received DISC while LINK up.
			0	0	1	0	0	DISC sent, sent SARM sent N2 times without UA.
			0	0	0	1	0	DISC sent, REC IDLE for T1xN2.
0	1	0	CHAIN STATUS ER00 = GNCS ER01 = CNR					
1	0		LINK RESET RECEIVED if ER05 - ER00 = 000000 LINK RESET TRANSMITTED if ER05 - ER00 = non-zero ER00 similar to W ER01 similar to X ER02 similar to Y ER03 similar to Z ER05 means received F = 1, but did not send P = 1 ERO4 means I-frame was sent N2 times without acknowledge					
1	1		COMMAND REJECT RECEIVED if ER05 - ER00 = 000000 TRANSMITTED if ER05 - ER00 = non-zero ER00 = W ER01 = X ER02 = Y ER03 = Z ER04 = Z1					

NOTES:

1. Whenever a command reject (CMDR) is received, the I-field will have been placed in appropriate memory by DMA, and a link reset SARM will be transmitted. The NB is not advanced.
2. Definitions of W,X,Y,Z as stated in CCITT X.25. Z1 indicates received N(S) is invalid (not part of X.25).

TERMS USED IN ERROR REGISTER

GNCS Going to Next Chain Segment

RLNR RLOOK Not Ready. REC RDY bit of next segment is 0.

ROR Receiver Over-Run. The Receiver Register (RR) had a character to load into the FIFO, but the FIFO was full.

RPKNR Received Packet but Memory Block was Not Ready.

TUR Transmitter Under-Run. The Transmitter Register (TR) needed a character from the Transmitter Holding Register (THR), but the THR was not ready.

NOSFR No S-frame received for T1 x N2. Used only if RRT1 = 1.

MEMORY ACCESS METHOD

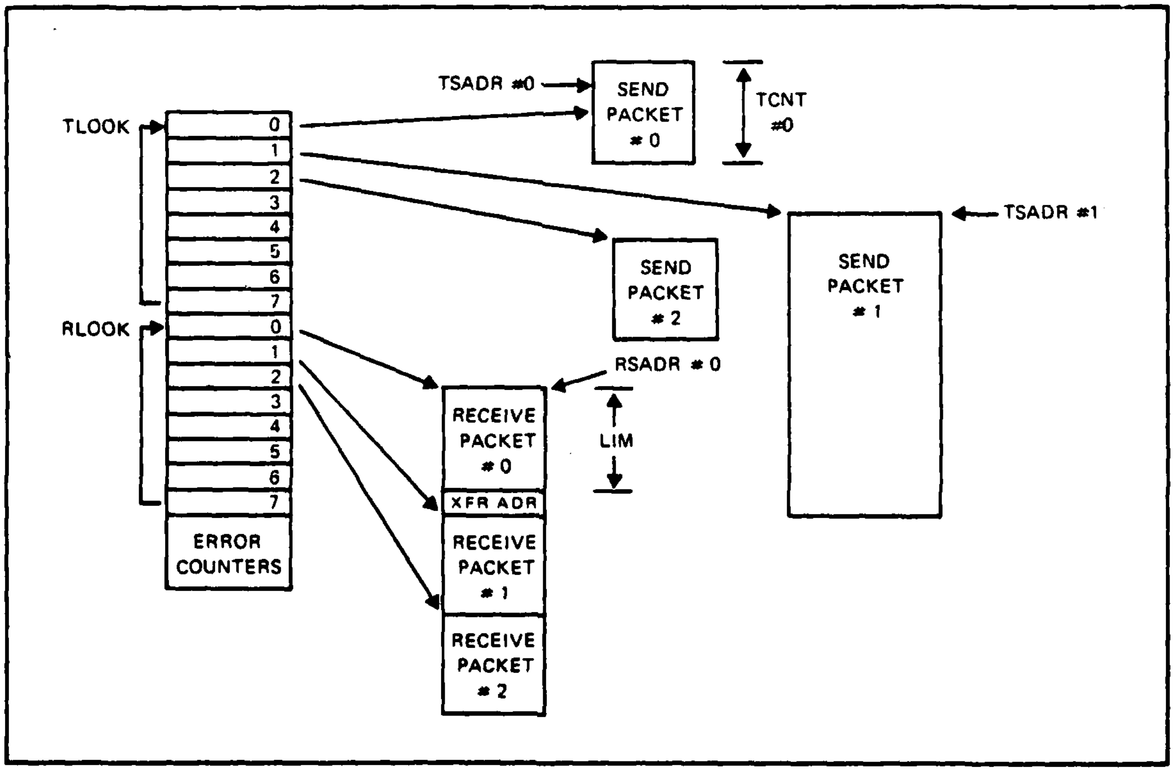
The memory access method, which includes DMA, is designed to take full advantage of the bit-oriented protocol which allows up to 7 I-frames to be outstanding (i.e., unacknowledged) in each direction of a communications link. The memory access method used two "look-up" tables: One for transmit and one for receive. These tables contain addresses and control for the individual send/receive packets. Thus, packet data are DMA addressed indirectly. This method is best suited for most software applications.

The 16 bit starting address for the look-up table TLOOK is loaded into the 2501 by the CPU. (I/O Registers "A" and "B"). RLOOK must immediately follow TLOOK in contiguous fashion. TLOOK and RLOOK are in the RAM memory external to the 2501. There are a total of 8 segmented control sections for each table. Each segment contains eight bytes. Four bytes are used for data memory starting address and length, two bits of one byte are used for control, one byte defines variable bit length and residual, and the other two bytes are open for user definition.

In transmit, the 2501 will have read from TLOOK the starting address and length of the first packet to be transmitted. The 2501 will automatically transmit the flag, address, and control fields. Next, the information field data will be transmitted using DMA from the "SEND #0 PACKET" memory. At the end of the information field, the 2501 will automatically send the FCS and closing Flag. The 2501 will then move on to the next packet.

If retransmission of one or more (up to seven) packets becomes necessary, the 2501 will automatically retrace the previous transmissions through the TLOOK table. The user's CPU software does not become involved in the retransmission. However, an ERROR COUNTER is incremented. (See Error Counter Section.)

To receive, each frame is checked for correct address and FCS fields and for type of control field. If the frame is a packet, the information field is placed in the assigned memory location in a method similar to that used in transmit. After the packet is received error-free and in proper N(S) sequence count, an interrupt is generated, and the 2501 is ready for the next packet which will be placed in the next location.



MEMORY ACCESS SCHEME

"DEADLY EMBRACE" PREVENTION

A "deadly embrace" can occur when two processors reach a state where each is waiting for the other. In this case, the two orocessors are the user's CPU and the micro-controller inside the 2501. Therefore, to prevent the "deadly embrace", the following rule is obeyed by the 2501 and should also be obeyed by the user's CPU. This rule applies to TLOOK, RLOOK, and to the I/O registers. The Error Counters do not apply to this rule.

> RULE: If a bit is set by the CPU, it will not be set by the 2501, and vice versa. If a bit is cleared by the 2501, it will not be cleared by the CPU, and vice versa.

As an example, the BRDY bit in the TLOOK segment is set by the CPU, only, but cleared by the 2501, only.

ERROR COUNTERS

Following continguously after RLOOK is ten 8 bit error counters. The 2501 will increment each counter at the occurrence of the defined event. However, the 2501 will not increment past 255 (all 1's). The CPU has the responsibility of clearing each counter. The first counter past RLOOK is #1, etc.

ERROR COUNTER	COUNT
1	Received Frames with FCS Error
2	Received Short Frames (less than 32 bits)
3	Number of times T1 ran-out (completed)
4	Number of I-Frame Retransmissions
5	REJ Frames Received
6	REJ Frames Transmitted
7	Invalid Commands Received
8	Invalid Responses Received
9	Number of frames which I-field exceeded total Limit.
10	Number of Null Packets Received

BYTE # IN SEGMENT	7	6	5	4	3	2	1	0
1	ACK'ED	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	BRDY
2	TSADR HI							
3	TSADR LO							
4	SPARE				TCNT HI			
5	TCNT LO							
6*	SBL2	SBL1	SBL0	BL1	RES2	RES1	RES0	BL0
7	SPARE FOR USER DEFINITION							
8	SPARE							

TLOOK SEGMENT

BYTE # IN SEGMENT	7	6	5	4	3	2	1	0
1	FRCML	SPARE	SPARE	SPARE	SPARE	SPARE	SPARE	REC RDY
2	RSADR HI							
3	RSADR LO							
4					RCNT HI			
5	RCNT LO							
6*	SBL2	SBL1	SBL0	BL1	RES2	RES1	RES0	BL0
7	SPARE FOR USER DEFINITION							
8	SPARE							

*Byte #6 defines variable bit length and residual bits.

RLOOK SEGMENT

BRDY means that the transmit buffer is ready. The 2501 will send the block only after the CPU makes BRDY = 1. (BRDY is used in conjunction with the SEND bit.) At the completion of the transmission, the 2501 will make BRDY = 0, and then read the BRDY of the next segment.

After transmitting a packet, an acknowledgement must be received from the remote device. The acknowledgement is contained in the received N (R) count of an I-frame, RR frame, or RNR frame. Upon acknowledgement, the 2501 will make ACK'ED = 1, and generate a block-acknowledged interrupt. Before assigning a new block to a segment in TLOOK, the CPU must make sure that the previous block which used that segment number has been acknowledged.

REC RDY informs the 2501 that the receive buffer is ready. The 2501 will not receive a packet into a buffer referenced by a particular segment until REC RDY = 1. If the 2501 progresses to a segment which has REC RDY = 0, an error interrupt will be generated.

After receiving an error-free packet in proper sequence, the 2501 will set FRCML, clear REC RDY, and generate a Packet Received Interrupt. The 2501 will also write the value of the binary length of the received packet in RCNT HI and RCNT LO. The NE count is advanced. The 2501 will acknowledge received packets at the first opportunity. This will be in either the next transmitted I-frame, or by an RR frame if RECR = 1, or by an RNR frame if RECR = 0. (RECR is in CRO.)

In the address bytes, HI represents the upper 8 bits and LO represents the lower 8 bits. In the count bytes, HI represents the upper 4 bytes.

TSADR is the starting address of the buffer to transmit, and TCNT is the binary count of the number of characters in the I-field.

RSADR is the starting address of the receive buffer. After successfully receiving the packets, the 2501 will write the value of RCNT which is the binary count of the number of characters in the I-field.

Whether the 2501 accesses a look-up table or a memory block, a DMA Cycle is required for each access.

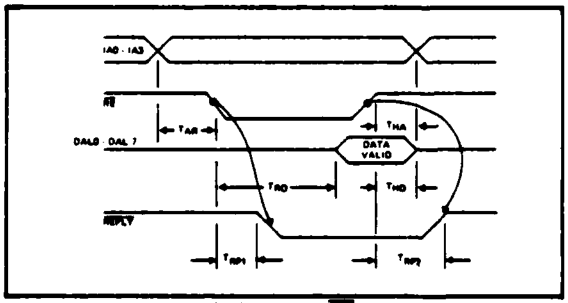
TLOOK AND RLOOK POINTERS

There are three 3-bit counters for the status of the segments in TLOOK and RLOOK. Status Register #0 (SRO) contains counters NA and NB which are used in conjunction with TLOOK. NB is the segment number of the next block to be transmitted, and is advanced at the end of each DMA transmission. NA is the value of the segment of the next block which will be acknowledged. If all transmitted blocks have been acknowledged, then NA = NB.

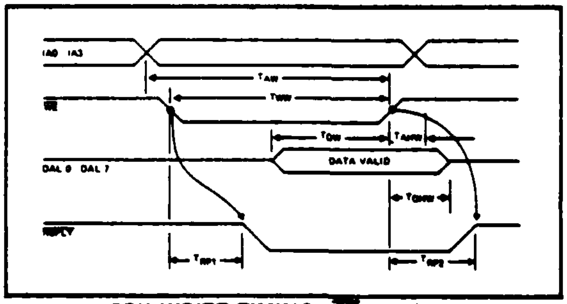
In SR1 is a 3-bit counter, NE, used with RLOOK. NE is the value of the segment number where the next received packet will be placed.

PRELIMINARY TIMING SPECIFICATIONS

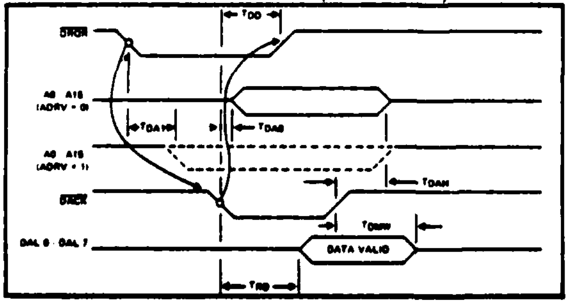
SYMBOL	PARAMETER	MIN. (NS)	MAX. (NS)	COMMENT
T_{AR}	Input Address Valid to $\overline{RE}$	0		
T_{RD}	Read Strobe (or $\overline{DACK}$ Read) to Data Valid	200 375		C (DAL) = 50 pf C (DAL) = 100 pf
T_{HD}	Data Hold Time from Read Strobe		80	
T_{HA}	Address Hold Time from Read Strobe	80		
T_{AW}	Input Address Valid to Trailing Edge of $\overline{WE}$	200		
T_{WW}	Minimum $\overline{WE}$ Pulse	200		
T_{DW}	Data Valid to Trailing Edge of $\overline{WE}$ or Trailing Edge of $\overline{DACK}$ for DMA Write	100		
T_{AHW}	Address Hold Time after $\overline{WE}$	80		
T_{DHW}	Data Hold Time after $\overline{WE}$ or after $\overline{DACK}$ for DMA Write	80		
T_{DA1}	Time from $\overline{DRQR}$ (or $\overline{DRQW}$) to Output Address Valid if ADRV = 1		80	C (ADDRESS) = 100 pf
T_{DA0}	Time from $\overline{DACK}$ to Output Address Valid if ADRV = 1		360	C (ADDRESS) = 100 pf
T_{DD}	Time from Leading Edge of $\overline{DACK}$ to Trailing Edge of $\overline{DRQR}$ (or $\overline{DRQW}$)		200	C ($\overline{DRQ}$) = 50 pf
T_{DAH}	Output Address Hold Time from $\overline{DACK}$		120	
T_{DMW}	Data Hold Time from $\overline{DACK}$ for DMA Read		80	
T_{RP1}	$\overline{REPLY}$ Response Leading Edge		160 240	C_{LOAD} = 50 pf C_{LOAD} = 100 pf
T_{RP2}	$\overline{REPLY}$ Response Trailing Edge		200 260	C_{LOAD} = 50 pf C_{LOAD} = 100 pf



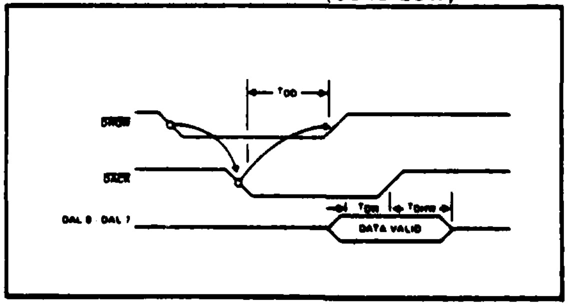
CPU READ TIMING ($\overline{CS}$ IS LOW)



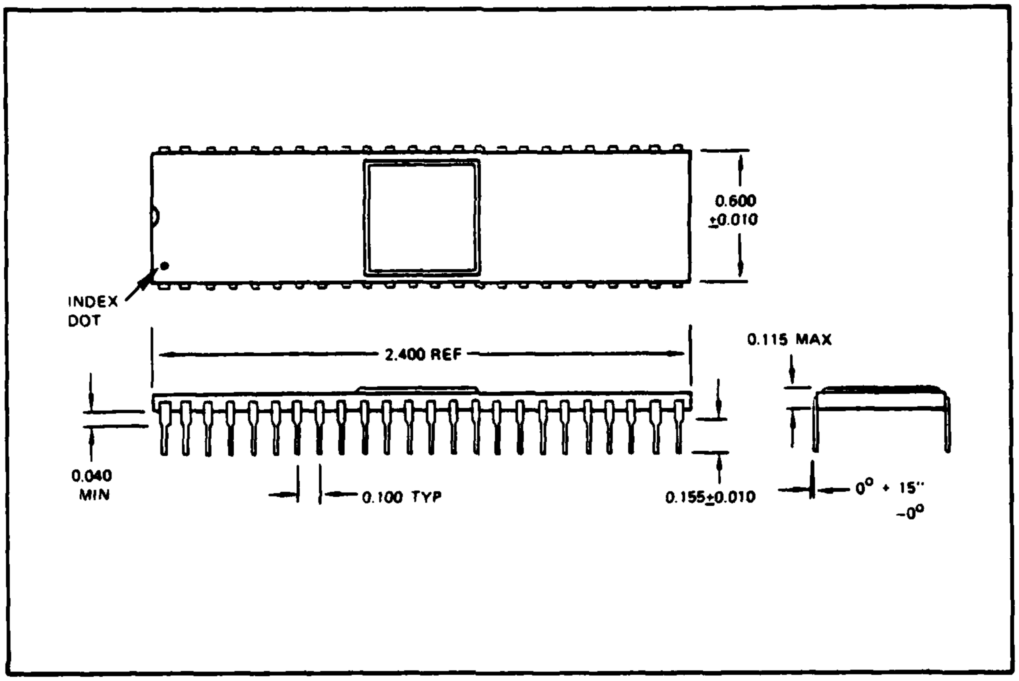
CPU WRITE TIMING ($\overline{CS}$ IS LOW)



DMA READ TIMING



DMA WRITE (A0-A15 SAME AS DMA READ)



WD2501 CERAMIC PACKAGE

This is a preliminary specification with tentative device parameters and may be subject to change after final product characterization is completed.

Information furnished by Western Digital Corporation is believed to be accurate and reliable. However, no responsibility is assumed by Western Digital Corporation for its use; nor any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Western Digital Corporation. Western Digital Corporation reserves the right to change said circuitry at any time without notice.

DAT

FILM