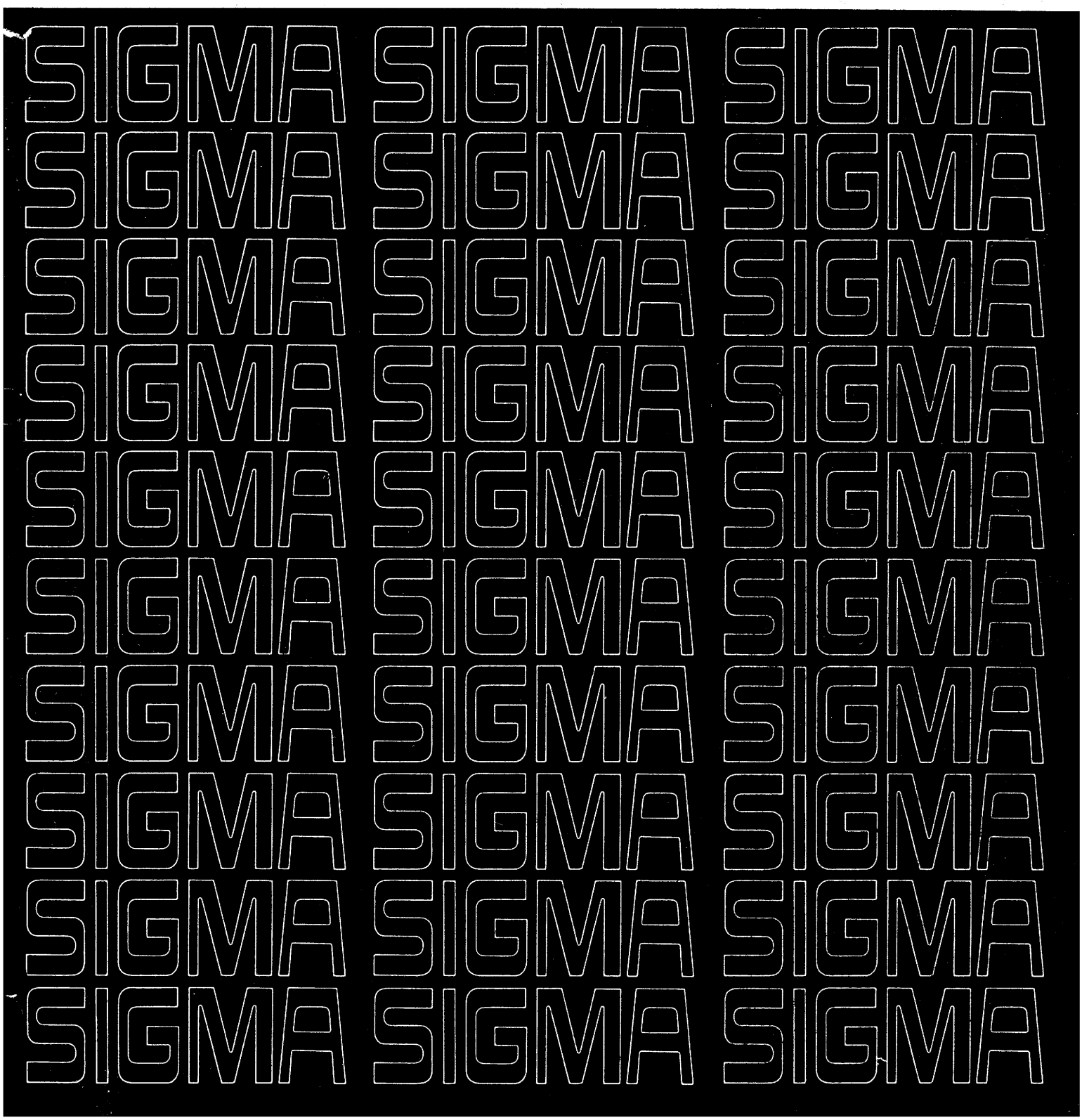
XDS

Xerox Data Systems

TECHNICAL MANUAL

CHARACTER ORIENTED COMMUNICATIONS EQUIPMENT MODEL 7611



XDS 901075A
$5.00

TECHNICAL MANUAL

CHARACTER ORIENTED COMMUNICATIONS EQUIPMENT

MODEL 7611

October 1969

This publication supersedes XDS 901075A
First Edition, dated June 1969

Xerox Data Systems 701 South Aviation Blvd., El Segundo, California 90245 (213) 772-4511, 679-4511

LIST OF EFFECTIVE PAGES

Total number of pages is 144, as follows:

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

TABLE OF CONTENTS (Cont.)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Cont.)

LIST OF TABLES

LIST OF TABLES (Cont.)

LIST OF RELATED PUBLICATIONS

The following publications contain information which is not included in this manual, but which is necessary for a complete understanding of the manual.

Publication Title	Publication No.
Sigma 2 Computer, Technical Manual	900630
Sigma 2 Computer, Reference Manual	900964
Sigma 5 Computer, Technical Manual	901172
Sigma 5 Computer, Reference Manual	900959
Sigma 7 Computer, Technical Manual	901060
Sigma 7 Computer, Reference Manual	900950
Sigma Computer Systems, Interface Design Manual	900973
Sigma 5 and 7 Character Oriented Communications Controller Test, Diagnostic Program Manual	901156
Sigma 2 Character Oriented Communications Controller Test, Diagnostic Program Manual	901168
Power Supply Models PX13, 52-PX13, PX14, PX15, Technical Manual	900001
Power Supply Model PT16, Technical Manual	901080
Power Supply Model PT18, Technical Manual	900866
Multiplexing Input/Output Processor Models 8271/8471 and 8272/8472, Technical Manual	901515
Sigma 5 and 7 Relocatable Diagnostic Program Loader, Diagnostic Program Manual	900972
Sigma 2 Relocatable Diagnostic Program Loader, Diagnostic Program Manual	901128

SECTION I
GENERAL DESCRIPTION

1-1 INTRODUCTION

This manual describes the XDS Character Oriented Communications Controller (COC) Model 7611 and optional equipment. The optional equipment consists of the High Speed Feature Model 7614 and the Dc Power Supply Model 7623. The manual is comprised of four sections that provide general information, programming information, principles of operation, and maintenance information and parts lists.

Technical manuals describing equipment associated with the COC are referenced in the list of related publications in the front matter of this manual.

The COC provides any Sigma series computer with the ability to communicate over both public and private transmission systems. The COC is intended for low to medium speed character oriented data transmissions and is optimized for character-by-character processing. However, the COC can be used for message processing with only a minimal loss of processor efficiency. A variety of speeds and formats are available as standard equipment. Special speeds, formats, and interfaces can be provided upon request.

The COC is particularly suited to applications in which each character (or short string) is subject to program examination and processing. It is also suited to time sharing applications which are geared to relatively slow human response capabilities. The COC is effectively decoupled from the CPU since real-time response to interrupts is not required and characters enter the computer memory without intervention from the CPU. This decoupling permits the CPU to perform other functions at the same time that messages from remote terminals are being stored in memory. The decoupling is made possible by use of the Multiplexing Input/Output Processor (MIOP), which provides buffering of all input data and processing of interrupts.

1-2 PHYSICAL DESCRIPTION

The COC is fully modular. Figure 1-1 shows the physical layout of the basic COC, which includes one Line Interface Unit (LIU) Model 7613, a Format Group Timing Unit Model 7612, and optional equipment. The LIU (address No. 0) that forms a part of the basic COC is contained in slots 6 through 32 of chassis A of the frame that houses the COC. As an option, seven additional LIU's, addresses 1 through 7, may be connected to the COC. Each LIU requires a separate chassis and is connected to the COC by cables. Each LIU may contain from one to eight send-receive positions. The number of positions needed depends upon the number of communications circuits served by the system. Each position is implemented by one Send Module Model 7615, one Receive Module Model 7616, and one Interface Module Model 7620, 7621, or 7622. Each send-receive position is capable of sending and receiving data at its interface. The interface conforms to EIA or MIL-STD-188D specifications or is a dc interface, depending upon the particular model of interface module that is used.

The send, receive, and interface modules can be arranged in any combination of simplex, half-duplex, or full duplex modes. For example: Both a send and a receive module are required if full duplex (simultaneous transmission in both directions), or half duplex (transmission in both directions but not simultaneous) operation is desired. However, only the send or the receive module, as applicable, is used if simplex operation (transmission capabilities in one direction only) is desired. Appropriate cables to the data set or other terminal device must also be used to meet the requirements of the type of operation. When dc interface module NT20 is used, a telegraphic relay module, KT12, is also used. Both of these modules are part of Relay Interface Unit Model 7620. Instead of connecting directly to the line, the interface module connects to the line through the relay module. The relay interface unit requires an additional source of dc power. This power may be supplied by either the customer or Power Supply Model 7623.

Locations of send, receive, and interface modules for each of the eight possible positions are shown in table 1-1. For example, the send, interface, and receive modules for position 3 plug into slots 21, 22, and 23, respectively.

The various transmission formats are categorized into format groups. A particular format requires specific types of send and receive modules. Table 1-2 lists the type of send and receive modules that are used for a desired format. For example, if format IV is desired, the send and receive modules used are LT50 and LT54, respectively. These modules are inserted in the appropriate slots as shown in table 1-1. Table 1-3 lists the types of interface modules with their respective numbers. The modules providing the desired interface types are inserted in the appropriate slots as shown in table 1-1.

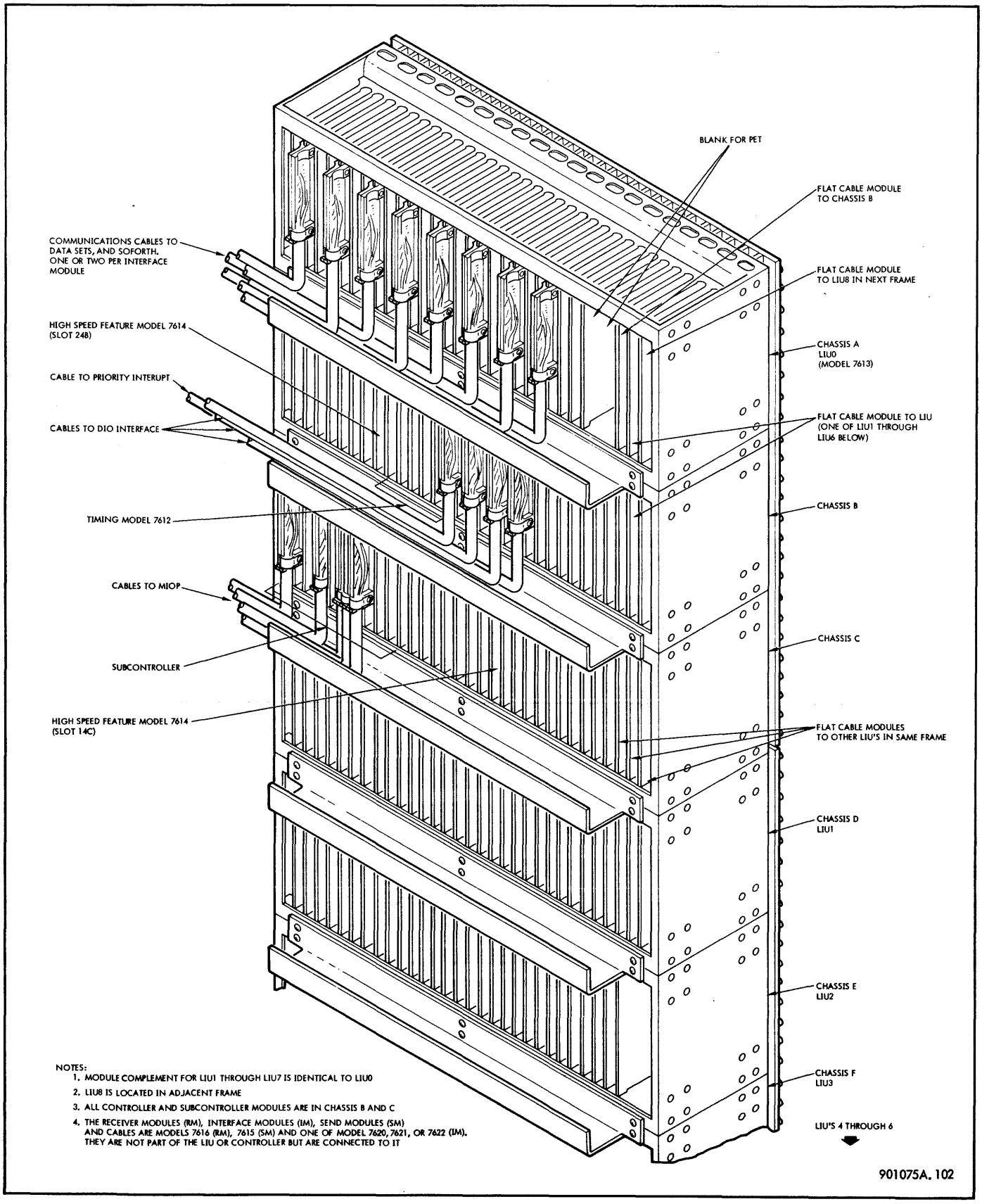


Figure 1-1. Character Oriented Communications Controller Model 7611 and Optional Equipment

The basic COC also contains the nine modules in slots 23 through 32 (slot 25 is vacant) of chassis C that comprise the subcontroller and the timing modules (CT17) in slots 20 through 24 of chassis B. The timing modules provide the bit timing source for all send and receive operations. Different configurations of timing modules are available as part of the basic COC. These configurations, formats, and speeds are as follows:

5-level/7.5-unit code at 60, 66, 75, and 100 words per minute (wpm)
7-level/9-unit code at 148 wpm
8-level/10-unit code at 150, 1200, and 1800 wpm
8-level/11-unit code at 100 wpm

Table 1-1. Send-Receive Modules, Position Locations

MODULE	POSITION NUMBER/SLOT							
	7	6	5	4	3	2	1	0
Send	9	12	15	18	21	24	27	30
Interface	10	13	16	19	22	25	28	31
Receive	11	14	17	20	23	26	29	32

Table 1-2. Module Types for Formats I through IV

FORMAT GROUP	MODULE TYPE	
	Send Module	Receive Module
I (5-level/7.5-unit code)	LT47	LT51
II (7-level/9-unit code)	LT48	LT52
III slow (8-level/10-unit code)	LT49	LT53
III fast (8-level/10-unit code)	LT49	LT53
IV (8-level/11-unit code)	LT50	LT54

Table 1-3. Interface Module Types

Type	Number
Dc (current)	NT20
EIA	NT21
MIL-STD-188D	NT32

Table 1-4 includes the crystal, frequency, speed, and slot in which the timing module must be inserted for each specified format.

Table 1-4. Correlation of Format, Speed, and Timing Module Locations

Format	Crystal 128131	Frequency	Speed	Slot
I	-010	1,474,560 Hz	60 wpm	21B
	-011	1,638,400 Hz	66 2/3 wpm	
	-012	1,843,200 Hz	75 wpm	
	-018	2,457,600 Hz	100 wpm	
II	-013	1,091,170 Hz	148 wpm	20B
III slow	-015	1,228,800 Hz	150 wpm	23B
III fast*	-015	1,228,800 Hz	1200 wpm	24B
	-012	1,843,200 Hz	1800 wpm	
IV	-014	1,802,240 Hz	100 wpm	22B

*Part of High Speed Feature Model 7614

Note

LIU0 can have only one format III timing source. If timing modules are installed in both slots (23B and 24B) that produce format III timing signals, LIU0 receives the format III fast clocks and LIU1 through LIU7 receive the format III slow clocks

1-3 FUNCTIONAL DESCRIPTION

The COC is the central element of the character-oriented communications system (figure 1-2). The COC provides a central source of timing and control and incorporates the functions of one LIU (address No. 0). Up to seven additional LIU's may be connected to a COC. An LIU contains from one to eight send circuits and from one to eight receive circuits. One COC, therefore, may have up to 64 send and 64 receive circuits, all of which may be simultaneously active. Since the maximum number of COC's that may be connected to a computer is 16, the total capacity of a system provides for two-way communication with up to 1024 data sets or other terminal equipments.

The COC connects to both the direct input/output (DIO) interface and the multiplexing input/output (MIOP) interface. Two levels of external priority interrupt are required, one for input and one for output. The receive interrupt (input) has the highest priority. All output data is transmitted from the CPU to the COC by way of the DIO interface, and all input data is

transmitted to core memory by way of the MIOP and its associated interface.

During a send (output) operation, the CPU transmits the character to the COC in parallel, loads it into a shift register, and transmits it serially on the send line. The characters are transmitted by the send circuits synchronously by bit but asynchronously by character. The CPU program controls transmission of the character to the COC, but a clock source contained in the COC controls sending each bit.

During a receive (input) operation, the CPU executes a start input/output (SIO) instruction to activate the COC. The sequence of events initiated by the SIO instruction permits the COC to make service requests of the MIOP. The character to be received is transmitted in serial form by the data set or by other terminal equipment to the receive circuits. When a character has been assembled in a receiver register, the COC requests service from the MIOP. During the ensuing service cycle, the input data is transmitted to core memory by way of the MIOP. Each character transmitted to the MIOP is accompanied by an external interrupt. The receiver continues to assemble characters until it is turned off by the program. The incoming characters are multiplexed into a common buffer area in core memory. The program is used to demultiplex the incoming data, to check for control characters and to assemble data into messages for further processing.

1-4 GENERAL SPECIFICATIONS

The general specifications of the communications controller and the line interface units are given in table 1-5.

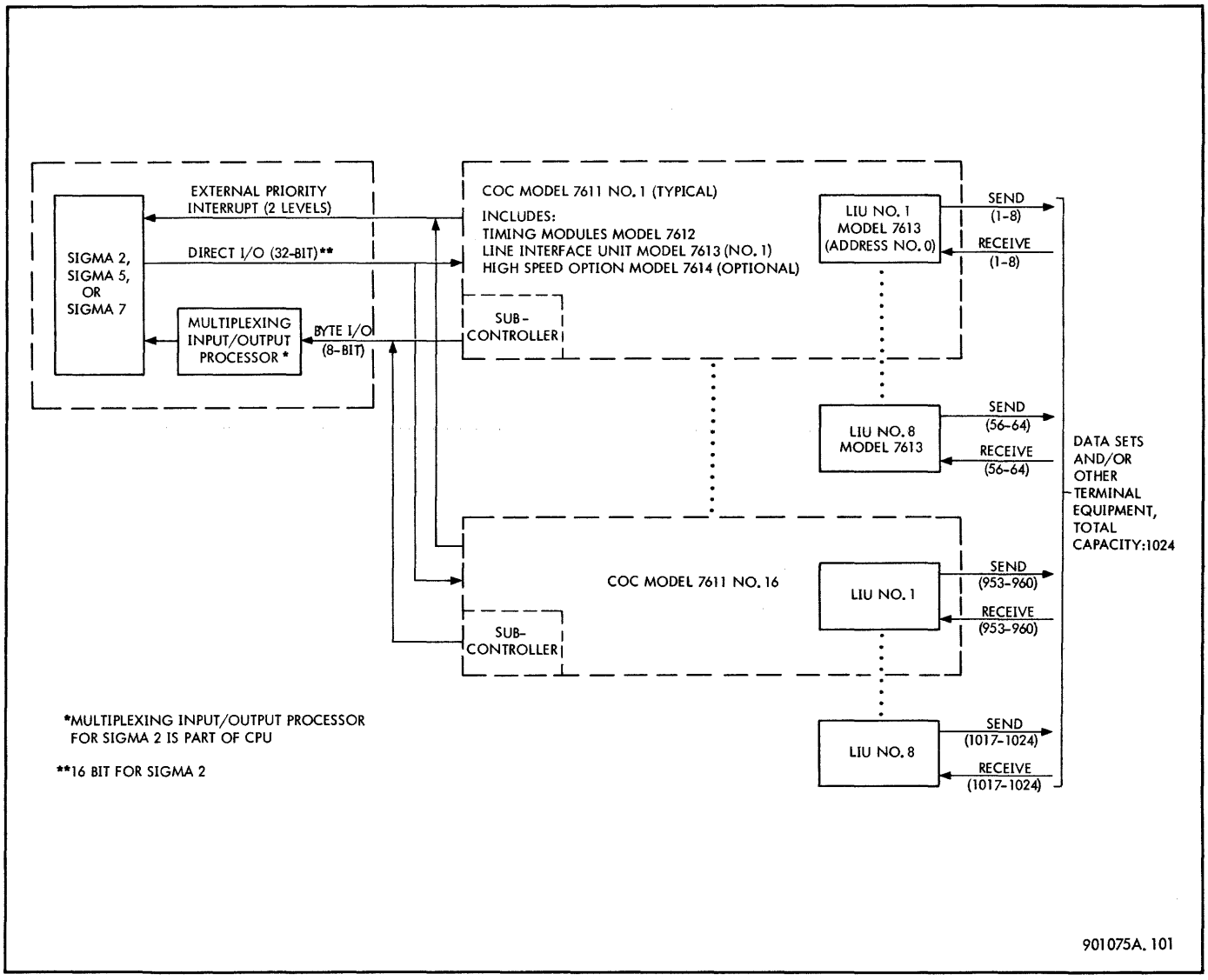


Figure 1-2. COC Optional Equipment, System Level Block Diagram

Table 1-5. General Specifications

Characteristic	Specification
Physical Characteristics	
Dimensions:	
Controller (standard Sigma three-row module case)	
Height	15-3/4 in.
Width	19 in.
LIU	
Height	5-1/4 in.
Width	19 in.
Weight	
Controller (3 chassis)	30 lb
LIU (1 chassis)	10 lb
Logic Signal Levels	
Internal to COC	True (one) : +4 Vdc False (zero) : 0 V (low impedance to ground)
External to COC (line)	EIA interface: True (mark): -5 Vdc False (space): +5 Vdc MIL-STD interface: True (mark): +6 Vdc False (space): -6 Vdc Dc interface: True (mark) : current False (space) : open line
Power Requirements (standard Sigma dc voltages)	
Complete COC (64 send and 64 receive circuits)	+4.0 Vdc (40A) +8.0 Vdc (20A) -8.0 Vdc (8.5A)
COC (including first LIU only)	+4.0 Vdc (12A) +8.0 Vdc (6A) -8.0 Vdc (1.5A)
LIU (not using MIL-STD-188D interface module)	+4.0 Vdc (4A) +8.0 Vdc (2A) -8.0 Vdc (1A)
LIU (using MIL-STD-188D interface module)	+4.0 Vdc (4A) +8.0 Vdc (2A) -8.0 Vdc (1A)
Maximum Number of Communications Circuits	1024 send 1024 receive

(Continued)

Table 1-5. General Specifications (Cont.)

Characteristic	Specification
Environmental Specifications	
Temperature	
Nonoperating	-40°C to 60°C (-40°F to 140°F)
Operating	5°C to 50°C (41°F to 122°F)
Relative humidity	
Nonoperating	No limits
Operating	10% to 95%
Vibration	
Nonoperating	
Unpacked	1G max at 5 to 1000 Hz
Packed	8G max at 5 to 1000 Hz
Operating	1G on vertical axis for 10 ms max
Altitude	
Nonoperating	20,000 ft max
Operating	10,000 ft max

SECTION II
OPERATION AND PROGRAMMING

2-1 GENERAL

The COC contains no controls or indicators other than the address switches contained on two switch comparator modules (LT26) and the ON-OFF switch on a logic element module (LT25). These switches are described in paragraphs 2-2 and 2-3. A brief description of the sequence of events that occur during an input/output operation are described in paragraph 2-4. The CPU instructions that are used at the IOP interface, the direct input/output (DIO) interface, and the meaning of the information received from the COC during the instructions are described beginning with paragraph 2-7. Additional information relating to instructions is provided in the following XDS reference manuals: Sigma 7 Computer, Reference Manual, publication number 900950, Sigma 5 Computer, Reference Manual, publication number 900959, and Sigma 2 Computer, Reference Manual publication number 900964.

2-2 ADDRESS SWITCHES

During certain instructions, the IOP places the device controller (D/C) address on the data lines. The D/C address is designated by setting eight toggle switches SWA0 through SWA7 (figure 2-1) located on Switch Comparator Module LT26 in slot 24 of the subcontroller chassis C. Switch SWA0 represents the most significant bit (MSB) of the address and switch SWA7 represents the least significant bit (LSB). For example, if the address assigned to the D/C is 3, switches SWA6 and SWA7 are set to the 1 position, and switches SWA0 through SWA5 are set to the 0 position. Similarly, a switch comparator module in slot 19 of chassis B is used for comparing the address assigned to the D/C with the addressed specified by the CPU on the direct input/output (DIO) interface lines. The switch settings on this module must reflect the D/C address used in conjunction with the DIO interface.

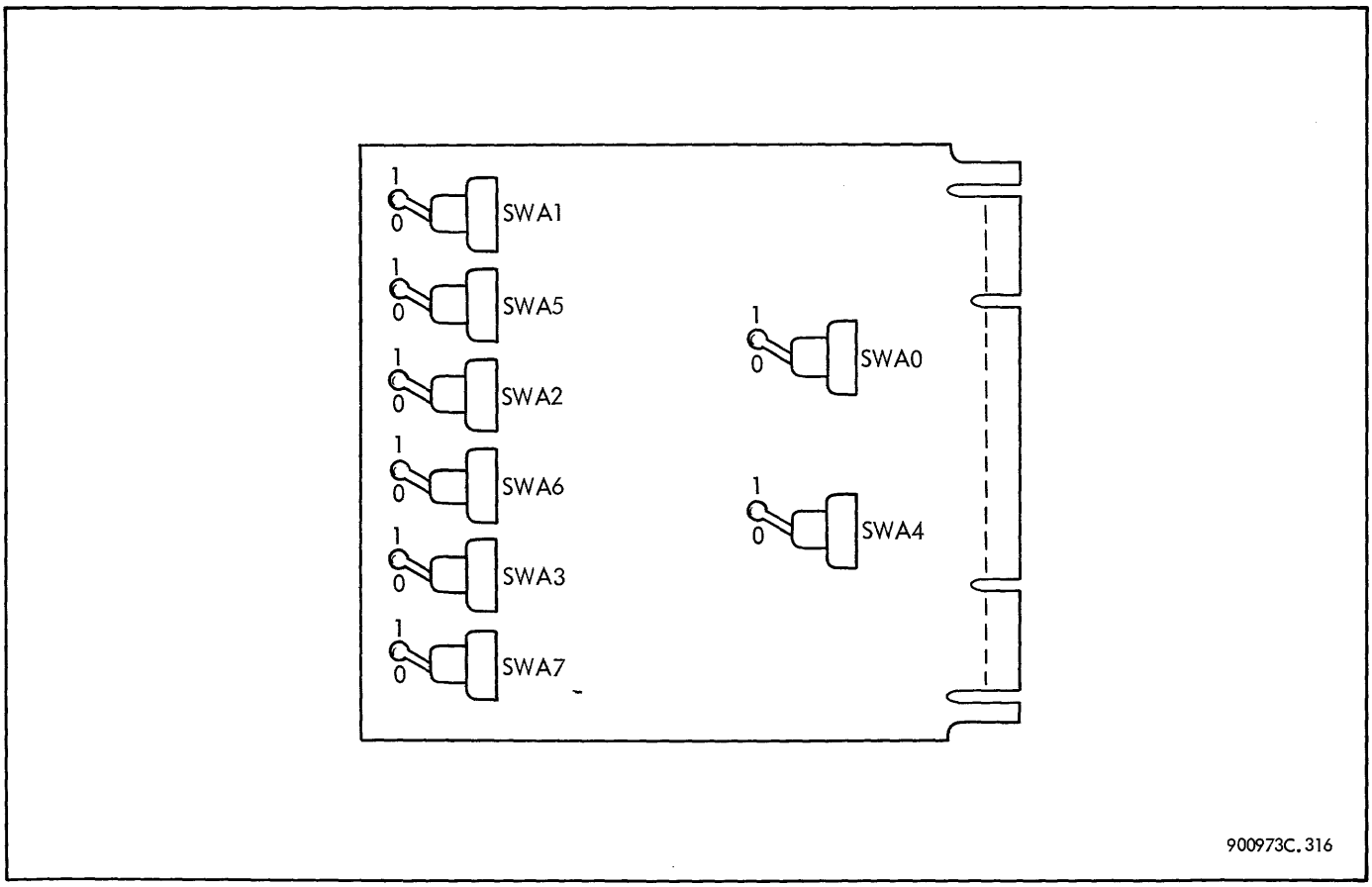


Figure 2-1. Switch Comparator Module LT26

2-3 ON-OFF SWITCH

The ON-OFF switch is located on module LT25 in slot 23 of chassis C of the subcontroller. This switch provides a means of both connecting the subcontroller to the MIOP interface and disconnecting the subcontroller from the MIOP interface in a transient-free manner. The subcontroller is connected to the MIOP when the switch is set to ON, and it is disconnected when the switch is set to OFF. The COC can be operated offline when it is disconnected from the MIOP, and online when it is connected to the MIOP. (See paragraph 3-31.)

2-4 OPERATION

Input and output processing are essentially independent of each other, and they are described separately in paragraphs 2-5 and 2-6.

2-5 Input Processing

Before the actual input operation is started, the program must set up buffer areas in core memory where the MIOP stores the input data. All input data is transferred by way of the MIOP interface; however, control functions are performed by way of the DIO interface.

An SIO instruction activates the COC and sets the address of the command doubleword for the COC into the MIOP fast-access memory subchannel associated with that COC. The COC responding to the SIO instruction requests service from the MIOP.

When the COC is connected to the MIOP for service, it specifies an order-out service cycle. This service cycle causes the MIOP to fetch the command doubleword from core memory and to set the address and the byte count from the doubleword into the appropriate subchannel. Data chaining operations and transfer in channel commands may be used in the command doubleword list to alternate between input buffer areas or to wrap around in a single area without additional program intervention.

After the initial setup has been completed, the receive modules are activated sequentially by way of the DIO interface by the Turn Receiver L On functions encoded in Write Direct instructions. All receive modules, whether activated or not, continuously maintain character synchronization with their input lines. When a receive module is activated by a Turn Receiver L On function, it assembles a new character or completes an assembly underway when activation occurred. When an active receive module completes assembly of a character, it requests service from its controller. The controller is continually scanning the receive modules for input service requests. When a request is found, the scanner stops, and the COC requests service from the MIOP. When the COC is connected to the MIOP for service, it specifies a data-in service cycle and transmits two bytes of data to the MIOP.

The MIOP stores these two bytes in the core memory location specified by the address contained in the subchannel associated with the COC. The first of these bytes contains the information character, and the second contains the line number and a control bit that indicates whether a long space has been received. Long Space is a space that is longer than a character time but is not necessarily an open line. It is a break (TTY) or an interrupt (KBD).

The receiver continues to assemble characters and to request service until it is deactivated by a Receiver Off function received by way of the DIO interface. The incoming characters are multiplexed into the common buffer area in core memory. Since data rates are very low, rate overruns are not likely to occur and are therefore not sensed.

With each character transmitted to the MIOP, a signal is sent on an external priority interrupt line (RINT). This signal is maintained until the acknowledge line (RINTACK) associated with that interrupt line goes true or until a reset or an HIO instruction addressed to the COC occurs.

XDS software is used to demultiplex the incoming data, to check for control characters, and to assemble data into messages for further processing. The program can also echo the characters to the data source for a full-duplex, verified operation.

2-6 Output Processing

Transmission of all output data and the associated control functions occur at the DIO interface. Before the first character of a message is sent to a line, the program must determine that a previous character transmission on that line is not in progress. This information may be already known or a test can be made using the Sense Transmitter L Status function.

If a Transmit On L function is sent to a line on which a previous transmission is in progress, the previous character and the new character are garbled. For subsequent characters it is known that a previous transmission is not in progress, since an interrupt is generated when transmission of each character is completed.

If a previous transmission is not in progress, the first character of the output message is sent to the line L

using the Transmit On L function of the Write Direct instruction. This function contains the character and the line number, L, over which the character is to be sent. The function is accepted, and transmission is attempted under all conditions.

If a data set is used and if it is not ready or if the transmitter is not installed, the character will not be successfully transmitted. Condition code bits CC3 and CC4 are controlled accordingly.

All output lines of a given format and speed use the same clock source for bit timing. The clocks occur at a rate of one per output bit except for the 5-level/7.5 unit formats which occur at a rate of two per output bit. Thus output characters are sent synchronously by bit. A character output starts on the next clock following receipt of the Transmit function; transmission is therefore asynchronous by character. For 5-level codes, a delay of not more than one half of a bit time is introduced because transmission is asynchronous by character; for other codes a delay of not more than one bit time.

After the character has been transmitted, the send module requests service from its controller, even if the transmission is unsuccessful. The controller scans its send modules for service requests. When the scanner detects a request, it stops and generates an output interrupt. The program then executes an Output Response function with a Read Direct instruction to determine the line number of the send module that is requesting service. The Read Direct instruction reads the state of the scanner flip-flops which encode the number of the send module that has requested service. The program then sends another character with a Transmit function or a long space with a Bend Long Space function. If no more characters are to be sent, a Stop Transmit function must be sent. The output scanner remains locked up until it is released by one of the above three Write Direct functions (Transmit, Send Long Space, or Stop Transmit).

The Send Long Space function of the Write Direct instruction causes the addressed send module to send spacing (zero) bits. As with any other output, the program must know or must determine that a previous transmission is not in progress before the Long Space function is initiated. When a send module starts sending space bits, it continues to do so until it receives a Stop Transmit function from the program.

When the module has been spacing for a full character time (the time required for start, data, and stop bits), it requests service of its controller. This service request results in an interrupt, the same as when transmission of a character has been completed. At this time another Long Space function may be sent, in which case the send module continues sending spaces. Another interrupt occurs after one character time has elapsed, or the send module may be deactivated by a Stop Transmit function.

The Stop Transmit function removes the true signal from the interrupt line and frees the send scanner. This function causes the addressed send module to send immediately a marking level (one bits) on its communication line. If a previous transmission is in progress it should not be sent because the last character will be garbled. The Stop Transmit function must always be used to terminate a long space output before a new character can be transmitted on that line. There must be a delay between the Stop Transmit and Transmit functions. The length of the delay depends on the device with which the line is communicating.

The program may send a Send Long Space function, immediately followed by a Stop Transmit function to a line that has finished sending a long space. If this is done, the line indicates "previous transmission in progress" as a status response until a full character time has elapsed from the time that Send Long Space was executed, even though the line marks from the time that the Stop Transmit function is executed.

2-7 PROGRAMMING CONSIDERATIONS

Operation of the COC involves use of the five I/O instructions associated with the MIOP interface and of the two I/O instructions associated with the DIO interface. (See the applicable reference manual and the Sigma Computer Systems, Interface Design Manual, publication No. 900973 for a general description of these instructions and of the MIOP and subcontroller functions generated in response to each instruction.)

All control and testing functions for both input and output are performed by Write Direct or Read Direct instructions over the DIO interface. All output data transfers are made using Write Direct instructions (DIO interface). All input data transfers are made by way of the MIOP interface with the control functions associated with the input operation using the DIO interface. Input data transfers from all receive modules in a COC share a single subchannel in the MIOP. The subchannel contains control information relating to the controller and to the subcontroller.

The COC is a single unit device controller, that is, the MSB of its address is always zero.

The COC response to both groups of instructions and the functions performed by these instructions are presented in paragraphs 2-8 through 2-34.

2-8 INPUT/OUTPUT INSTRUCTIONS (MIOP INTERFACE)

There are four input/output instructions besides the Read Direct and the Write Direct (DIO interface) that

apply to the COC. A fifth I/O instruction (Test Device) may be addressed to the COC, however, it is not used. All I/O instructions result in the setting of a condition code (CC1 and CC2) to denote the nature of the I/O response and status information (except AIO and TDV) relating to the COC.

The instructions are presented to the COC on five unique lines that are termed function indicators. These function indicators are AIO, HIO, SIO, TDV, and TIO; a sixth, ASC, is used to acknowledge a service call from the COC.

2-9 SIO Function Indicator

The start input/output (SIO) is used to start the COC in preparation for an input operation. Each SIO is accompanied by an address that is decoded by the subcontroller section of the COC. In response to a properly addressed SIO, the COC controls the condition code lines according to table 2-1 and places status information on the function response lines according to table 2-2.

If the controller is in the ready state, it advances to the busy state and makes a request for service (raises its service

Table 2-1. Condition Code Response for Instructions

CONDITION CODE BIT		INSTRUCTION	MEANING
CC1 (DORD)	CC2 (IORD)		
	1	SIO	SIO is successful
	1	HIO	COC was not busy when HIO was received
	1	TIO	SIO can be accepted
1		SIO, HIO, TIO, TDV	Address recognition
	1	TDV	Driven unconditionally by the COC
	1	AIO	Driven unconditionally by the COC
1 = Condition code light off			

Table 2-2. Status Response for SIO, HIO, and TIO Instructions

FUNCTION RESPONSE LINES								MEANING
0	1	2	3	4	5	6	7	
1	X	X	X	X	X	X	X	Interrupt pending
X	X	X	1	X	X	X	X	Automatic mode (always 1)
X	X	X	X	1	X	X	X	Unusual end
X	0	0	X	X	0	0	X	COC ready
X	1	1	X	X	1	1	X	COC busy
X	X	X	X	X	X	X	0	Not used (always 0)

Note

The COC does not provide status on the data lines when responding to an AIO. Status is provided on the function response lines only in response to SIO, HIO, and TIO instructions

call line). If the controller was already in the busy state when the SIO was received, it provides status and condition code information but otherwise ignores the SIO.

2-10 HIO Function Indicator

The halt input/output (HIO) is used to halt an input operation. Each HIO is accompanied by an address. In addition to providing status (table 2-2) and condition code information (table 2-1), the HIO resets the busy indicator and any existing external (receive only) or internal (MIOP) interrupt request. The controller returns to the ready state and can be started again by an SIO.

2-11 TIO Function Indicator

The test input/output (TIO) is used to test the status of the controller. Each TIO is accompanied by an address. In addition to providing status information (table 2-2), the controller also supplies condition code information (table 2-1). The operation of the controller is not affected by the TIO.

2-12 TDV Function Indicator

The test device (TDV) is not functional with the COC. If a TDV is addressed to the COC, the COC responds only by driving both condition code lines to indicate that an abnormal condition does not exist. The status byte is all zeros.

2-13 AIO Function Indicator

The acknowledge input/output interrupt (AIO) is used to determine which device controller requested an interrupt. The COC responds to this instruction, if it is requesting an internal interrupt and has priority, by placing its address on the function response lines and by driving both condition code lines. The COC does not send status information on the data lines.

The COC interrupts by way of the MIOP only if commanded by a terminal order. The AIO also resets the interrupt indicator if it is set.

2-14 ASC Function Indicator

The acknowledge service call (ASC) is generated by the MIOP in response to a service call from a controller. Upon receipt of an ASC, the COC connects to the MIOP interface lines for service if it is the highest priority controller with a service call pending. The COC specifies the type of service it wants by controlling the input/output request (IOR) and the data/order request (DOR) lines as shown below.

DOR	IOR	SERVICE
0	0	Data-in
0	1	Data-out
1	0	Order-in
1	1	Order-out

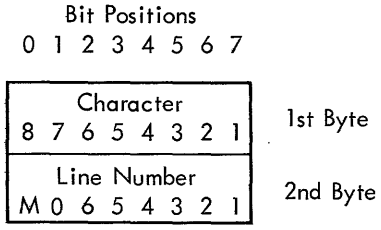
Data-out service does not apply to the COC, since output characters are transmitted by way of the DIO interface. The three types of service that are used are described briefly in paragraphs 2-15 through 2-17.

2-15 Order-Out Service

The first service cycle following a successful SIO is always specified by the controller as an order out. The controller makes no use of the order-out service cycle and only asks for it to permit the MIOP to perform the preliminary operations required for the data-in service that follows. The order received during the order-out service cycle is ignored by the COC.

2-16 Data-In Service

The controller requests a data-in service cycle when a character has been assembled by the receive module and is ready to be transferred to core memory. The COC inputs two bytes during each data-in service cycle. The first byte is the character; the second is the line number from which the character was received. The two bytes have the format shown below.



The least significant bits of the character and of the line number are in bit position seven of the bytes. The high order bits are ones for characters that are less than eight bits in length.

For five-bit characters (5-level/7.5-unit), a sixth bit is generated and is placed in bit position two of the first byte. This bit is initially zero and thereafter reflects the case shift that was last received. A one means that a figures code has been received since the last letters code. A zero means that a letters code has been received since the last figures code or that no figures code has been received since initialization by an I/O reset. The bit is changed after the figures or the letters character is sent to the MIOP.

Bit position one of the line number (adjacent to the M bit) is always zero. Bit M is used to signal long space detection. If M is a zero, the input byte is a valid character. If M is a one, a long space has been detected.

2-17 Order-In Service

The controller requests an order-in service cycle when the operational status byte is to be transmitted to the MIOP.

The COC asks for order-in service only as a result of a terminal order in which bit 1 (count done) or bit 3 (IOP halt) was true. During order-in service, the COC reports unusual end and channel end by driving data bits 3 and 4 respectively.

2-18 Terminal Orders

A terminal order automatically accompanies an order-in and an order-out service cycle whether or not there is anything to report to the controller. A terminal order may also be sent by the MIOP as part of a data-in service cycle when it wants the COC to perform a specific function. Specifically, terminal orders are sent to the COC to request an interrupt, to specify count done, or to specify IOP halt.

a. If bit zero (interrupt) of the terminal order is true, the COC generates an internal interrupt.

b. If bit one (count done) and/or bit 3 (IOP halt) of the terminal are true, the COC reports unusual end by driving data lines 3 and 4 during the ensuing order-in service cycle.

2-19 INPUT/OUTPUT INSTRUCTIONS (DIO INTERFACE)

The CPU is capable of communicating directly with the COC over the DIO interface by means of the Write Direct (WD) and the Read Direct (RD) instructions. The WD instruction is used to send the output character to the COC and to send control information to the COC. The RD instruction is used to obtain information from the COC. Since the same interface lines are used for both instructions, a selection line, Read Direct/Write Direct (RWD), which is controlled by the CPU distinguishes between the two instructions. The selection line is driven high for the WD and low for the RD.

There are always two words associated with an RD and WD instruction; the instruction word and the data word. These words are described in paragraphs 2-20 and 2-21.

2-20 Instruction Word

During execution of both the RD and the WD instructions, the CPU presents 16 bits of control information to the COC on input lines A00 through A15. This information comes from the effective address field of the instruction word and is interpreted as shown in figure 2-2. Bits 16 through 19 identify the mode of the instruction. Bits 20 through 23 must be zero. A mode number, 3, is assigned to the COC; therefore, bits 16 through 23 identify a COC as a recipient of an RD or a WD instruction if an X'30' is encoded in bits 16 through 23 (lines A00 through A07). Bits 24 through 27 identify one of the 16 possible COC's in a system. Bits 28 through 31 specify a function to be performed. The functions are described separately under the instruction with which they are associated.

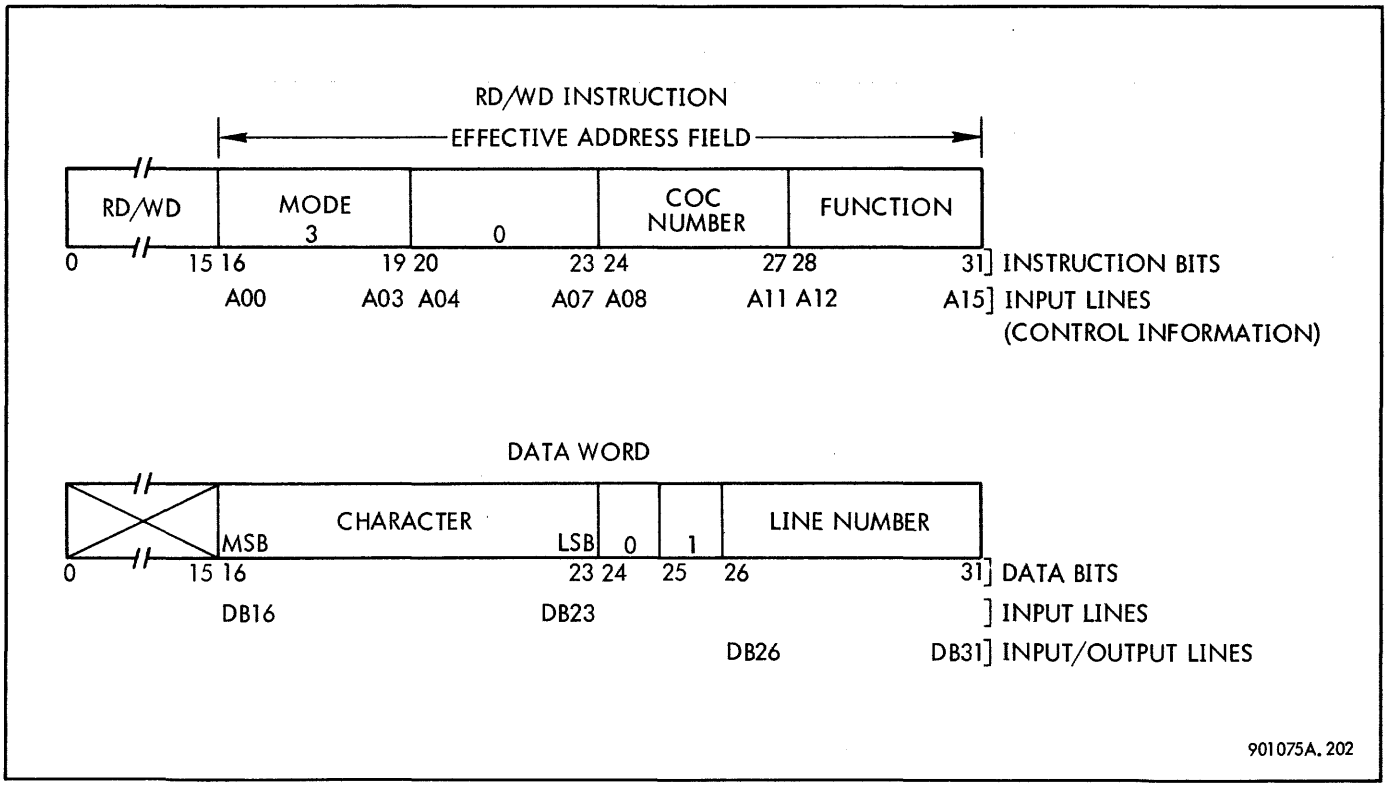


Figure 2-2. COC Interpretation of RD and WD Instructions

2-21 Data Word

During an output operation, a WD instruction is used to transmit a character to the COC. The character is encoded in bits 16 through 23 (input lines DB16 through DB23) of the data word. Bits 24 and 25 of the data word are used only for testing purposes during an RD or a WD instruction. Bits 26 through 31 (input/output lines DB26 through DB31) uniquely specify one of 64 send or 64 receive modules . Lines DB26 through DB31 are bidirectional. During a WD instruction, the CPU places the line number associated with a send or a receive module on these lines. During an RD instruction, the COC places the line number associated with a send or a receive module on these lines for transmission to the CPU.

2-22 Write Direct Functions

The functions encoded in bits 28 through 31 of the instruction word are described below. Figure 2-3 shows the format of the instruction word and the data word for each of the functions. Since the WD instruction is used to convey information to the COC, the COC does not transmit information to the CPU on the bidirectional lines. The COC does, however, send status information to the CPU on condition code lines CC3 and CC4. (See tables 2-3 and 2-4.) Condition code bits CC3 and CC4 in the Sigma 2 are the overflow and the carry bits, respectively. The condition code reflects the status of the addressed transmitter or the receiver resulting from the instruction for which status is returned. For example, CC4 is zero for a Transmit Data On L function, indicating that the present character is in progress. Condition code settings in response to WD instructions are described in paragraph 2-34.

2-23 SENSE RECEIVER L STATUS. This function is used to test the status of the selected receiver by way of the condition code bits.

2-24 TURN RECEIVER L ON. This function is used to set flip-flop FRON which is associated with receiver L. The state of this flip-flop is returned as part of the status in CC3 and CC4 for receiver functions. When the flip-flop is set, that receiver requests service from its controller when it completes assembly of a character. If the flip-flop is reset, that receiver does not request service.

2-25 TURN RECEIVER L OFF. This function is used to reset flip-flop FRON, which is associated with receiver L. See paragraph 2-24 for a complete description.

2-26 TURN RECEIVER L DATA SET OFF. This function causes the data terminal ready line to the associated data set to go false for approximately 80 milliseconds. As a result, the data set disconnects from the communication channel. This function is used only if data sets are employed.

Table 2-3. Condition Code for Receive

CONDITION CODE BITS		MEANING
CC3	CC4	
1	1	Receiver installed Data set ready Long space detected
1	0	Receiver installed Data set ready Receiver off and not long space
0	1	Receiver installed Data set ready Receiver on and not long space
0	0	Receiver not installed or data set not ready

Table 2-4. Condition Code for Send

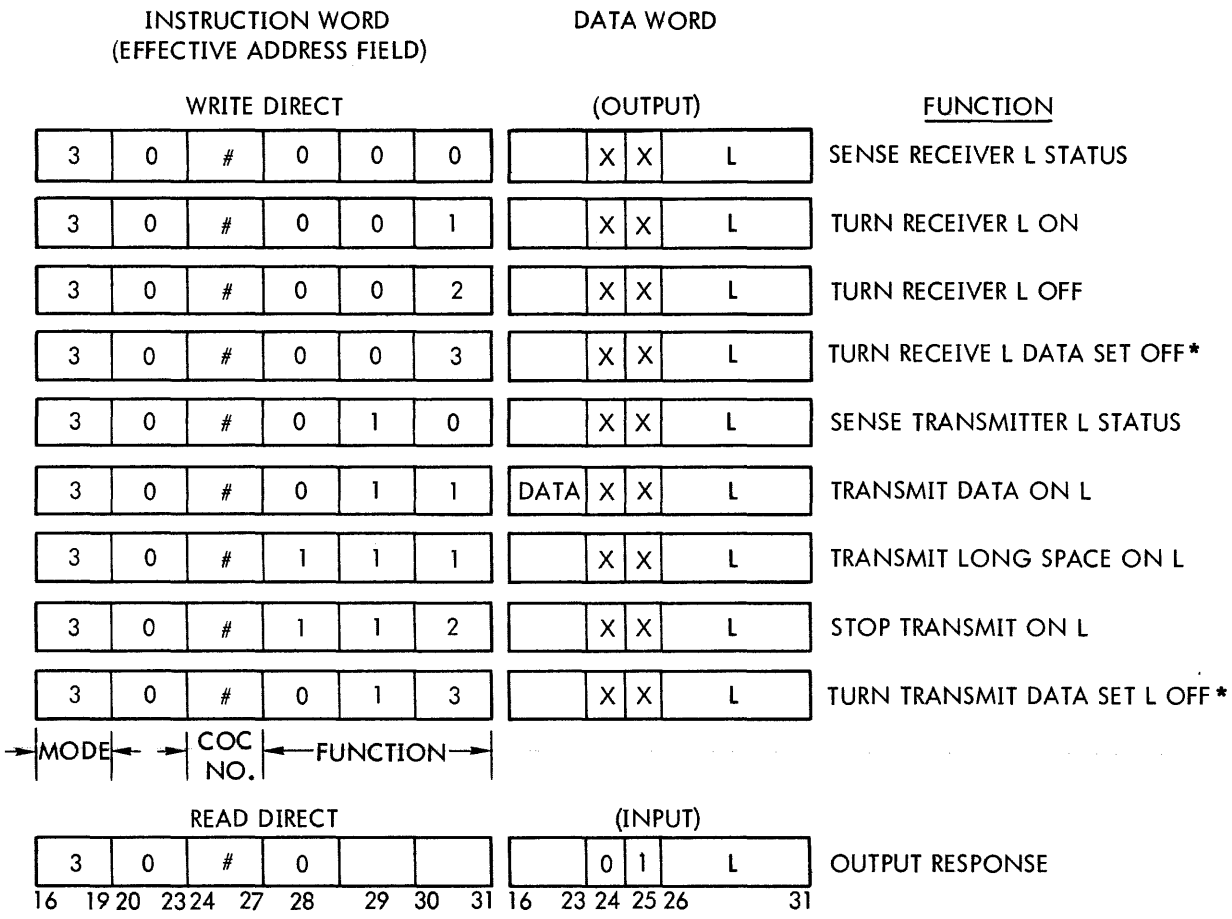
CONDITION CODE BITS		MEANING
CC3	CC4	
1	1	Transmitter installed and clear to send Previous character not in progress
1	0	Transmitter installed and clear to send Previous character in progress
0	1	Transmitter not installed or transmitter installed and not clear to send Previous character not in progress
0	0	Transmitter not installed or transmitter installed and not clear to send Previous character in progress

2-27 SENSE TRANSMITTER L STATUS. This function is used to test the status of the selected transmitter by way of the condition code bits.

2-28 TRANSMIT DATA ON L. This function causes the data character in the output word to be loaded into the send module and also causes transmission to be initiated.

2-29 TRANSMIT LONG SPACE ON L. This function causes the selected send module to start the long space transmission sequence. The send module sends spaces until it is terminated by a Stop Transmit function. A service request and the resulting interrupt is generated after transmission of the number of spaces equal to one character.

2-30 STOP TRANSMIT ON L. This function causes the selected module to move its output to the mark



*PROVIDED ONLY WHEN DATA SETS ARE EMPLOYED

#COMMUNICATIONS CONTROLLER NUMBER (X'0' THROUGH X'F')

DATA: CHARACTER TO BE TRANSMITTED

L: LINE NUMBER (X'0' THROUGH X'3F')

X: NOT INTERPRETED

901075A.203

Figure 2-3. Functions Encoded in RD and WD Instructions

state. It also causes the selected send module to stop generating service requests. Therefore, interrupts resulting from the service requests are stopped. The function does not affect the sequencing in the send module. For example, if a character transmission is started and is stopped by a Stop Transmit on L before the transmission is complete, the send module is not ready to send a new character until it completes the sequencing for output. If a status test is made before the sequencing is complete, the condition code bits indicate that a previous transmission is in progress.

2-31 TURN TRANSMIT DATA SET L OFF. This function causes the data terminal ready line to the associated data set to go false for approximately 20 milliseconds which results in the disconnection of the data set from the communication channel. This function is used only if data sets are employed.

2-32 Read Direct Functions

The only function encoded in an RD instruction that applies to the COC is Output Response. Figure 2-3 shows the format of the instruction word and the data word. Condition code bits CC3 and CC4 are zero for an RD instruction addressed to the COC.

2-33 OUTPUT RESPONSE. This function is issued only in response to an output service request interrupt. The low-order six bits of the input data word contain the line number, L, of the send module that is responsible for the interrupt.

2-34 Condition Code Status Response

The status response associated with a WD instruction is sent to the CPU by way of condition code lines CC3 and CC4. (These lines are the carry and the overflow lines, respectively, on the Sigma 2 CPU.) Condition code lines CC3 and CC4 are zero for an RD instruction addressed to the COC. Table 2-3 shows the condition codes and their meaning for receive only. Table 2-4 shows the condition codes and their meaning for send only.

SECTION III
PRINCIPLES OF OPERATION

3-1 GENERAL PRINCIPLES OF OPERATION

The general principles of operation describe the COC in terms of the overall functional areas which are shown in figure 3-1. The functional areas shown in the figure are described generally in this section and are described in greater detail under Detailed Principles of Operation. Also included in this section is an explanation of the transmission paths, codes and code formats that apply to the COC.

3-2 SUBCONTROLLER

Each device controller that operates with an IOP (or with an automatic I/O channel when used with Sigma 2) contains a subcontroller section that consists of nine modules. The subcontroller section is incorporated to interface with the IOP, since the device controllers are designed to operate with specific devices. In effect, the subcontroller makes all device controllers look alike to the IOP. The subcontroller provides the following capabilities.

a. All cable drivers and receivers required to connect the device controller to the IOP interface

b. Logic required to determine priority during ASC and AIO operations

c. Address selection switches and logic for comparing these switch outputs with the device number during SIO, TIO, TDV, and HIO operations

d. A service connect flip-flop that connects the device controller to the IOP interface when the controller requests service

e. Relay logic (under remote control) for connecting and disconnecting the subcontroller to the IOP interface during power on and power off operations

3-3 RECEIVE SCANNER

The receive scanner is a module 64 counter which sequentially generates all possible address combinations

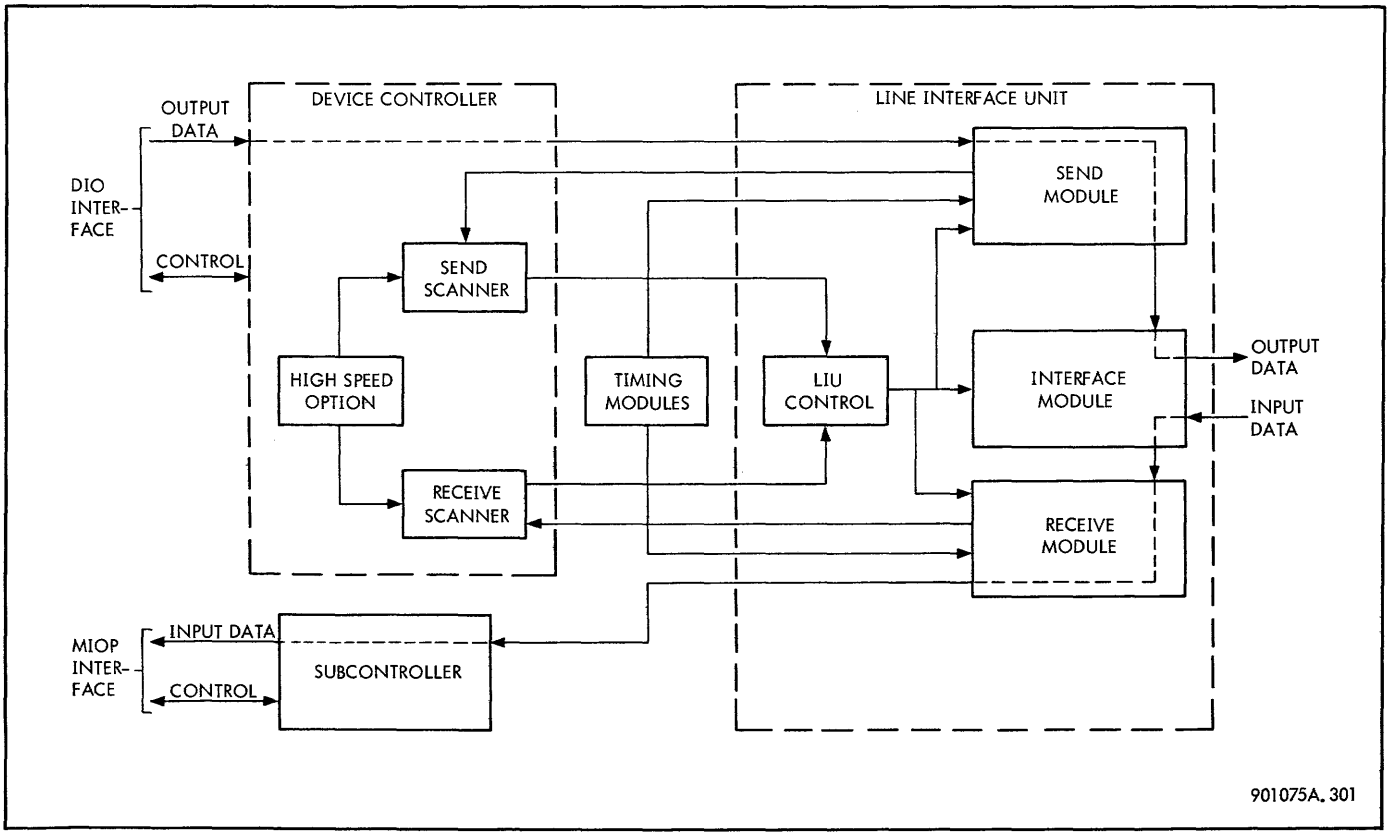


Figure 3-1. COC Overall Block Diagram (Simplified)

from 00 through 63 and then starts over. The purpose of the scanner is to scan sequentially all of the receive modules for a service request starting with LIU0 through LIU7. When a request is found, the scanner stops and the controller requests service from the MIOP.

The scanner consists of six flip-flops, three which specify one of eight LIU's and three which specify one of the eight receive modules in each LIU. The LIU number is decoded by gating in the COC. The receive modules in each LIU are selected by the manner in which three scanner signals and their complements are wired to the eight slots (receptacles) on the chassis in which the receive module is inserted.

When the address generated by the scanner coincides with the address of a receive module that is ready to input a character, the receive module causes the scanner to stop. While it is stopped, the character and the line number (receiver address) are transmitted to the MIOP. Since the address specified by the scanner coincides with the line number of the receive module, the MIOP obtains the line number by reading the outputs of the six scanner flip-flops.

3-4 SEND SCANNER

The send scanner is a module 64 counter which sequentially generates all possible address combinations from 00 through 63 and then starts over. The purpose of the send scanner is to scan sequentially all of the send modules starting with LIU0 through LIU7 in search of one that has completed transmission of a character and is ready to request service. When one such send module has been found, the send scanner stops and the send external interrupt is raised.

The send scanner consists of six flip-flops, three which specify one of eight LIU's, and three which specify one of the eight send modules in each LIU. The LIU address is decoded by a combination of gating in the COC and chassis back wiring on each LIU. Ribbon cables connect the LIU's to the COC. The three scanner signals that represent the send module address are sent to all LIU's. The send module address is decoded by the LIU control logic and is routed to the appropriate send module.

When the address generated by the send scanner coincides with the address of a send module that has completed the output of a character, the scanner stops, and the send interrupt line is raised. In response to the interrupt, the CPU issues an RD instruction in which the output response function is encoded. During the RD instruction, the outputs of the six send scanner flip-flops are placed on the DIO interface data lines for transmission to the CPU. After the CPU has the address of the send module wanting service, the CPU must issue a WD instruction addressed to that send module before the send scanner can resume scanning.

The send scanner stops every time that an RWD instruction is addressed to the COC, even if the instruction is to control a receive function. This is done because the same lines that are used to address the send modules from the send scanner are also used to address the receive modules when controlling receive functions. The send scanner is stopped in order not to bypass any send service requests.

3-5 HIGH SPEED FEATURE

The High Speed Feature Model 7614 permits speeds of either 1200 or 1800 bits per second (bps) to be used for format III modules of LIU address zero (the first LIU). When the option is installed, the high-speed timing pulses are routed to format III modules of LIU address zero, and the low-speed timing pulses are routed to format III modules in all other LIU's. In addition, this option modifies the scanners so that the high-speed lines receive service more often than the low-speed lines. Thus, if a controller has the high-speed option, it can drive a maximum of eight high-speed input lines and eight high-speed output lines, as well as 56 low-speed input lines and 56 low-speed output lines. The manner in which the send scanner and the receive scanner are modified when the high speed option is installed is described in paragraphs 3-6 and 3-7.

3-6 Send Scanner Modification

The send scanner sequentially scans the eight send module positions of LIU No. 1 (address LIU0), then the eight positions of LIU No. 2 (address LIU1), and continues until the eight send modules of LIU No. 8 (address LIU7) have been scanned. The scanner then starts over when all 64 possible addresses have been generated in sequence. When the high speed option is installed, the scanner is reset to 63 (addressing the last module in LIU7) each time that a send module, not in LIU0, is serviced. (Serviced means that a Transmit, Stop Transmit, or Send Long Space function has been directed to the module in question.)

These modifications to the scanner provide a faster scanning rate for the high speed modules (LIU0) since these lines are always scanned following each service of lines in any other LIU.

3-7 Receive Scanner Modification

The receive scanner sequentially scans the eight receive module positions of LIU No. 1 (address LIU0), then the eight positions of LIU No. 2 (address (LIU1), and continues until the eight receive modules of LIU No. 8 (address LIU7) have been scanned. The scanner starts over when all 64 possible addresses have been

generated in sequence. When the high speed option is installed, the scanner is reset to 63 (addressing the last module in LIU7) each time that a receive module not in LIU0 is serviced. (Serviced means that a character is sent from that receiver to the MIOP.)

3-8 LINE INTERFACE UNIT

The Line Interface Unit (LIU) Model 7613 can accommodate eight send modules Model 7615 and eight receive modules Model 7616. The first LIU is included in the communications controller. Up to seven additional LIU's can be cable-connected (via ribbon cable) to the controller.

Each LIU contains the logic and the electronics necessary to operate eight send/receive positions. Each send/receive position accommodates a send module, a receive module, and an interface module. These can be arranged in any combination of simplex, half-duplex, or full-duplex modes.

3-9 Send Module

The Send Module Model 7615 accepts output data, one byte at a time, from the controller and accepts timing signals from the Timing Module Model 7612 associated with its unique format group. The send module serializes the data and transmits it with appropriate start and stop bits. Each send module is functionally independent of all other send modules in the system. Thus, the transmission operation is asynchronous on a per-line basis. Send modules are classified according to the transmission format used. See table 1-2 for the send module to use with a given format.

3-10 Receive Module

The Receive Module Model 7616 accepts input data as an asynchronous bit stream, strips off start and stop bits, assembles the characters, and transmits them to the computer memory by way of the controller and the MIOP. Each receive module is functionally independent of all other receive modules in the system. Receive modules are classified according to the speed and the transmission format used. See table 1-2 for the receive module to use with a given format. All receive modules of a given format group derive their bit timing from a common, shared timing source in their associated controller.

3-11 Interface Module

An interface module (Models 7620, 7621, and 7622) contains the line interface circuits and some control circuits for connecting one send module and one receive module to their data sets, communication lines, or relays. Since each interface module connects to one send and to one receive line, the send and receive lines are paired. Each member of a pair must use the same model interface module (Models 7620, 7621, or 7622) for the other member of that pair. The models shown above satisfy the interface specification and standard as follows:

a. Model 7621 (NT21): Electronic Industries Association (EIA) voltage interface

b. Model 7622 (NT32): Military Standard (MIL-STD-188D) voltage interface

c. Model 7620 (NT20): Relay driver/receiver interface (dc)

The cabling to the communications network (data sets, current lines, or other terminal devices) is attached to the interface modules by connectors. The cabling used must conform to the type of operation, that is, full-duplex, half-duplex, or simplex.

When the NT20 module is used, the KT12 relay module is also used. Both the KT12 module and the NT20 module are part of Relay Interface Unit Model 7620.

The relay interface unit requires a separate source of power. This power can be furnished by Power Supply Model 7623 when it is not furnished by the customer.

3-12 Line Interface Unit Control Logic

The LIU control logic that accommodates all eight send, receive, and interface modules in each LIU is contained on two LT46 control buffer modules. These two modules contain logic that decodes the address of the send, receive, and interface modules, and also decodes the function encoded in the RD or the WD instruction.

3-13 TIMING MODULES

Timing modules (Format Group Timing Unit Model 7612) provide bit timing for all send and receive operations. Timing modules (CT17) are available in nine different configurations to accommodate the speeds and the formats shown in table 1-4. Thus, one controller can meet the format and the speed requirements for most types of terminal devices. If a new device with a different speed or format is connected to the COC, the timing module, the send module, and the receive module can easily be changed to accommodate the new device.

3-14 TRANSMISSION PATHS, CODES, AND CODE FORMATS

The types of transmission paths, codes, and code

formats applicable to the COC are described in paragraphs 3-15 through 3-19.

3-15 Transmission Paths

Three modes of operation for transmission of data are possible with the COC: simplex, half-duplex, and full-duplex.

3-16 SIMPLEX OPERATION. During simplex operation, transmission capabilities are possible in either a send or a receive direction, but not both. If send only capabilities are desired, the receive module for that send/receive position is not required. Likewise, if receive only capabilities are desired, the send module for that send/receive position is not required.

3-17 HALF-DUPLEX OPERATION. Half-duplex operation is one in which transmission is possible in both directions but not simultaneously. Half-duplex operation requires the same equipment in the COC as the full-duplex operation. The transmission capabilities (half-duplex or full-duplex) are determined by the terminal equipment.

3-18 FULL-DUPLEX OPERATION. Full-duplex operation is one in which transmission is possible in both directions simultaneously. The simultaneous transmission of data is possible because of the connection to the MIOP interface for input data and to the DIO interface for output data, as well as the independently operated send and receive logic in the COC. For example, a teletype printer can be printing a received message while the operator is sending a different message by way of its keyboard. Full-duplex operation also permits an immediate accuracy check of input information. The message being typed on the keyboard can be immediately echoed by the CPU and, therefore, can be printed simultaneously.

3-19 Codes and Code Formats

The codes used by various keyboard printers are a function of the number of bits used to represent a character. In this case, the number of bits are referred to as levels. The eight-level code is the one best suited to digital data processing work; however, five-level and seven-level codes are used by some equipment.

Characters are transmitted serially, bit by bit. A true bit is designated as a marking bit, and a false bit as a spacing bit. Each character that is transmitted must be accompanied by start and stop bits in addition to the character bits.

Each bit period (pulse period), including the character bits and the start and the stop bits, are referred to as units. The number of units is the total number of pulse periods required for transmission of a character, including the start, stop, and data bits. The number of levels refers to the number of data bits alone. For example, the 5-level/7.5-unit format requires one unit for start, five units for data, and one and one-half for stop. The 8-level/11-unit format requires one unit for start, eight units for data, and two units for stop. See table 1-2 for the send and the receive module to use for a specific format.

PULSE

When no characters are being transmitted by a particular send/receive position, that transmission system is normally in the marking condition; that is, the send module in the COC is continually sending marking bits, and the receive module in the COC is continually receiving marking bits from the line. When the COC transmits a character, the send module first transmits a start bit, which is always a space, and follows with the data bits, which are, in turn, followed by the stop bit or bits (always a mark).

If the terminal device has been in the quiescent state (receiving marking bits), the receipt of a space bit alerts that device to the coming character and synchronizes the terminal device with the COC. Figure 3-2 shows a character, including the start bits and stop bits, that conforms to the 8-level/11-unit code format.

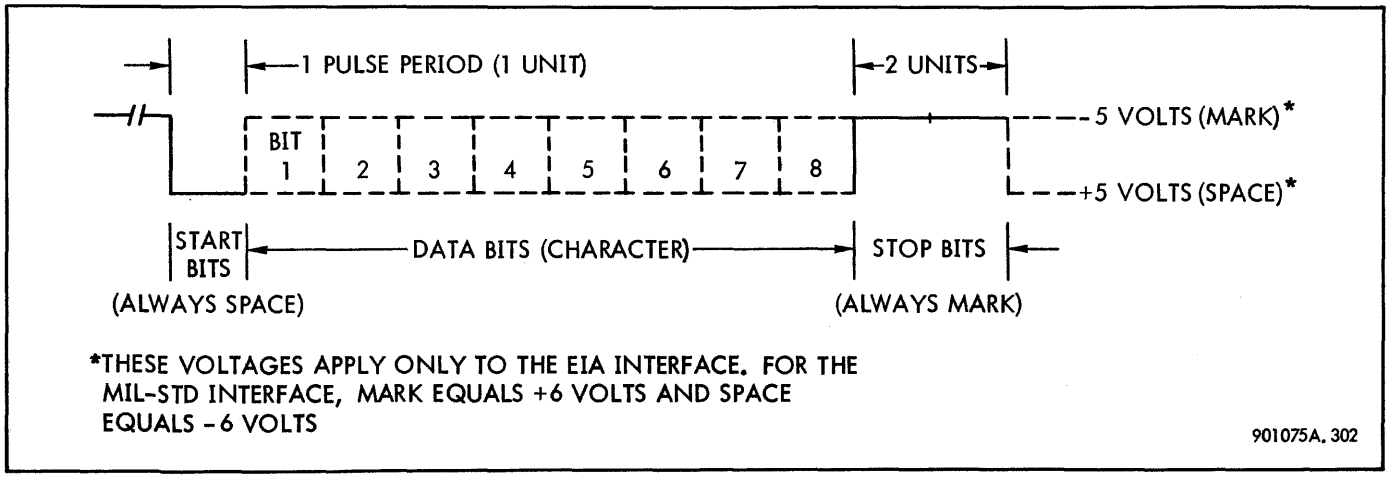


Figure 3-2. Start, Stop, and Data Units in the Eight-Level/11-Unit Code Format

After the stop bits, which are always marking, have been sent, transmission of the character is complete. A new character may be sent immediately by transmitting a space (start bit) during the pulse period immediately following the stop of the previous character. If a new character does not follow, marking bits follow the stop bits.

The same operation occurs when characters are being received by the COC. When a character is transmitted from the COC, the send module generates the start bit (space), clocks out the data bits in series, and, finally, generates the proper stop signal. The output of the send module is level-shifted by the interface circuits to match the terminal device. The voltage values provided by the EIA interface module for the mark and the space signals are: mark = -5 volts, and space = +5 volts. The voltage values provided by the MIL-STD interface module for the mark and the space signals are: mark = +6 volts, and space = -6 volts. Mark and space for the dc interface module are: mark = current flow, and space = no current flow (open circuit). On the return path, the receiver module strips off the start bit and clocks the data bits into a shift register. When the stop signal is received, the data is made available at the shift register outputs.

3-20 DETAILED PRINCIPLES OF OPERATION

The detailed principles of operation are divided into two categories: a description of register and functional logic groups, and a glossary of terms and signals.

3-21 COC REGISTERS AND FUNCTIONAL LOGIC GROUPS

To facilitate discussion of the registers and logic groups, the COC has been divided into functional sections. The resulting sections are: the subcontroller, the controller, the line interface units, and the timing signal generator. The controller (not including the LIU) is divided into the send logic and the receive logic. The LIU is divided into four subsections, the LIU control logic, the send module, receive module, and the interface module. Figure 3-3 is a block diagram showing each register and logic group in each subsection. Both the subcontroller and the timing signal generator logic are represented by single blocks, each of which are shown separately in figures 3-4 and 3-44. These detailed descriptions are keyed to the applicable overall block diagram (figures 3-3, 3-4 or 3-44). Each separate description and its related logic diagram applies to a specific block on the block diagram. The descriptions describe the functions performed by that logic, and provide the sources and the destinations of signals associated with each block.

3-22 Device Subcontroller

The Device Subcontroller (DS) functional logic groups are shown in figure 3-4 and are described below.

3-23 CABLE RECEIVER-DRIVERS. The cable receiver-drivers (figure 3-5) provide amplification and impedance matching between the I/O system and the other circuits within the subcontroller and the COC.

Signals that are received from the MIOP through a cable receiver enter the DS with the letter R attached to the signals' reference designators. For example, function strobe FS becomes FSR at the output of its cable receiver. Likewise, signals that are generated in the COC and are sent to the MIOP through cable drivers have the letter D affixed to their reference designators. For example, interrupt call IC, the output of a cable driver, becomes ICD at the input to the cable driver.

3-24 ADDRESS SELECTION AND RECOGNITION. The address selection and recognition logic (figure 3-6) consists of eight toggle switches and associated gates and inverters. The eight toggle switches are set to the address assigned to the COC and generate signals SWA0 through SWA7. When the COC is being addressed during an SIO, TIO, TDV, or HIO function, the address logic compares the address presented to the COC on the data lines (DA0R through DA7R) with the address specified by the eight toggle switches. Signals DCA and DCA47 are generated if the two addresses match and if the service connect flip-flop in the DS is reset (signal FSC false). (Signals DCA and DCA47 are physically tied together.) Signal DCA is used as an enable signal in both the DS function logic and the COC.

During an AIO or an ASC function, signals SWA0 through SWA7 are applied to the function response lines (FR0 through FR7) for purposes of supplying the MIOP with the COC address.

3-25 SERVICE-INTERRUPT PRIORITY AND SERVICE CONNECT LOGIC. The service-interrupt priority logic (figure 3-7) gates service requests and interrupt requests between the controller and the I/O system. When a service request or an interrupt request is acknowledged by the I/O system, the service-interrupt logic determines whether the COC has priority and, accordingly, actuates the appropriate circuits in the controller. The service-interrupt logic is described first during an AIO (interrupt) function, and then during an ASC (service call) function.

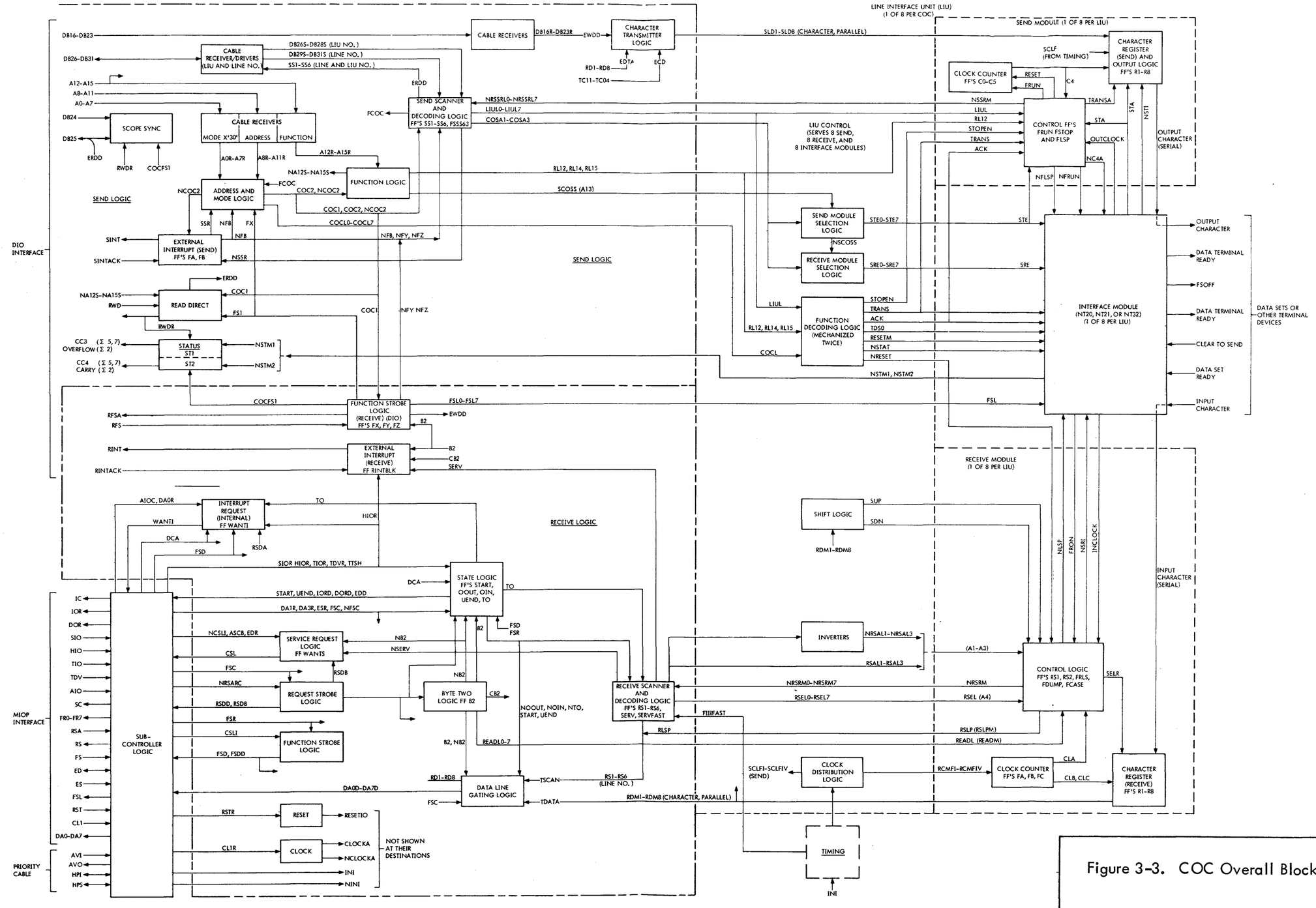


Figure 3-3. COC Overall Block Diagram

901075A.303

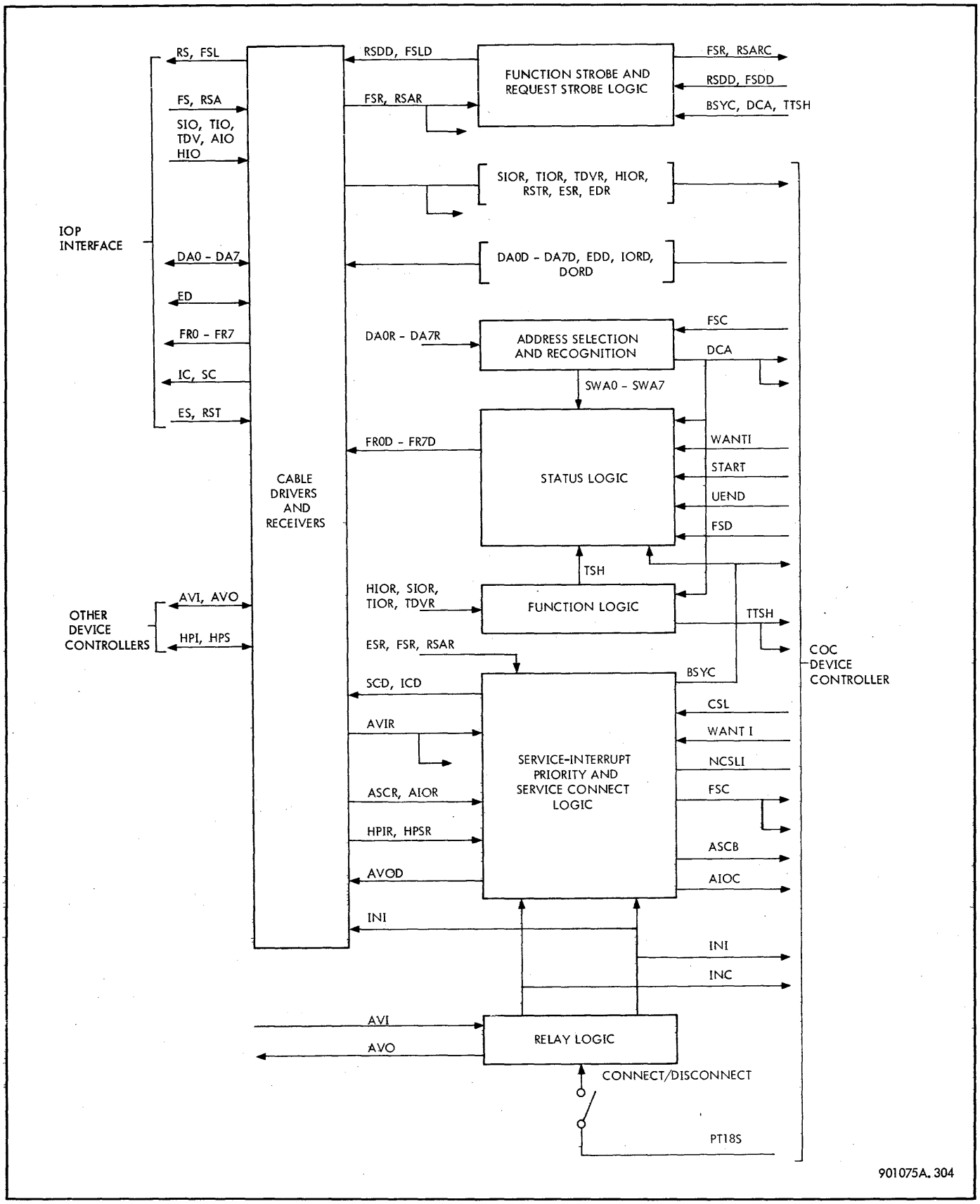


Figure 3-4. Device Subcontroller, Block Diagram

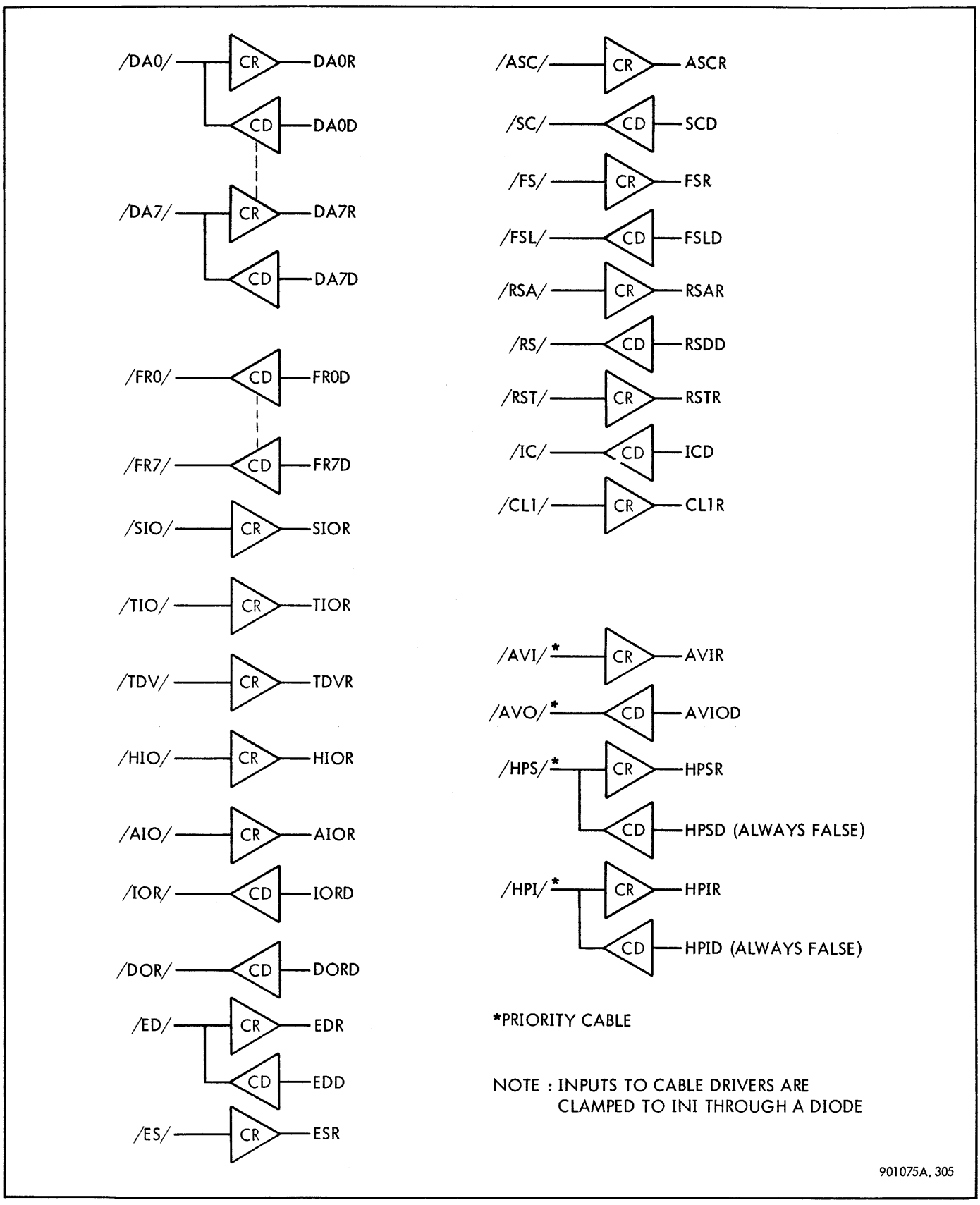


Figure 3-5. Cable Receiver-Driver, Logic Diagram

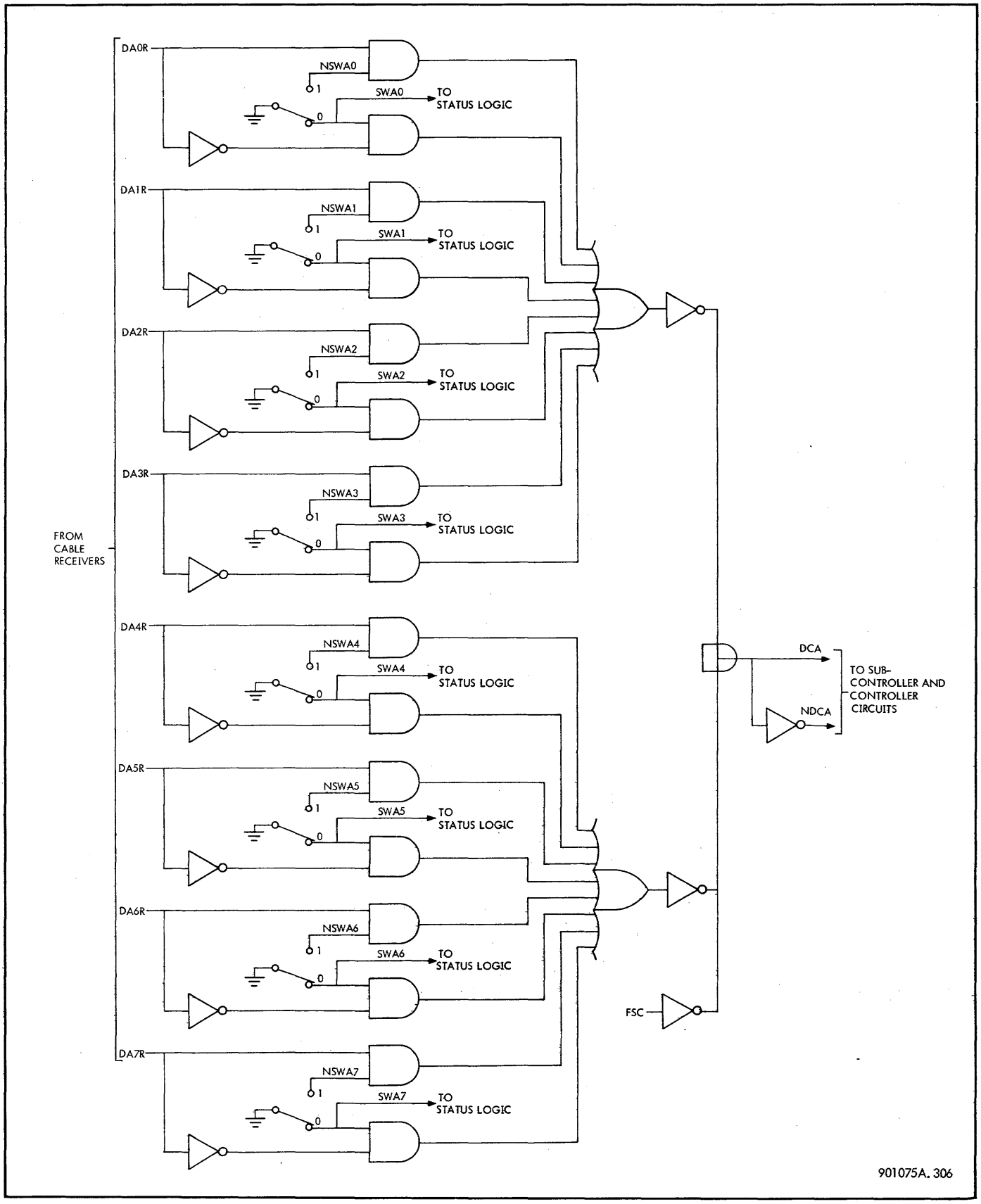


Figure 3-6. Address Selection and Recognition, Logic Diagram

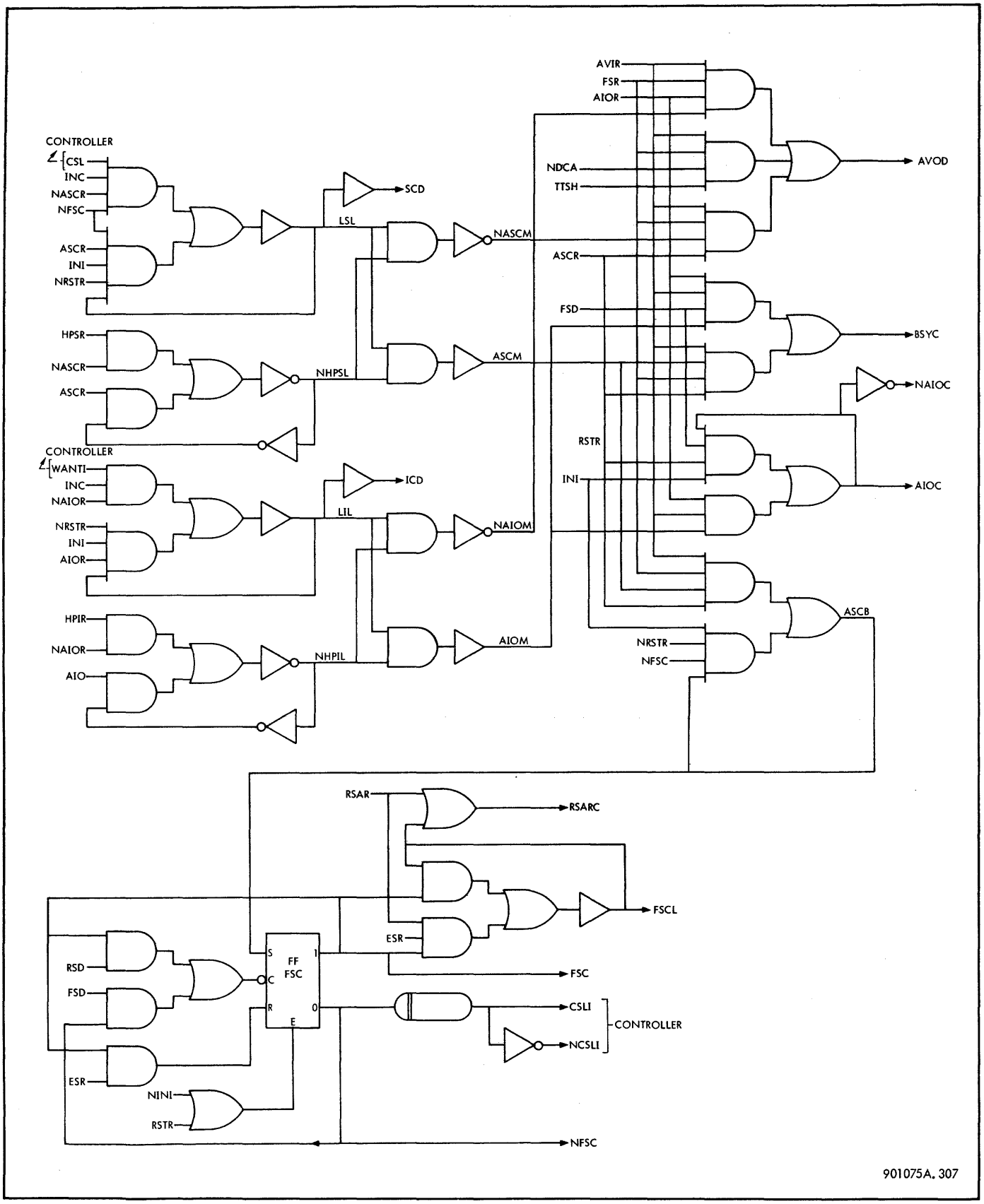


Figure 3-7. Service-Interrupt Priority and Service Connect Logic

3-26 INTERRUPT CALL. When the COC receives a terminal order that requests the COC to interrupt, the controller generates signal WANTI. This is the only way that the COC can generate an interrupt to the MIOP. Latch LIL in the DS goes true if the AIO function indicator line is not being driven by the MIOP (signal NAIOR true), and if power has been applied to the DS (signal INC true).

LIL = WANTI NAIOR INC
+ LIL AIOR INI NRSTR

The interrupt line (IC) is raised when signal LIL goes true.

IC = ICD = LIL

Latch LIL is held true by the term WANTI NAIOR INC until the interrupt is acknowledged. At this time, the term LIL AIOR INI NRSTR goes true (AIO function indicator true) and sets latch LIL. Signal INI is true if power has been applied to the DS. Signal NRSTR is true if the reset signal from the MIOP is false.

When the MIOP raises the function strobe (FS), the highest priority DS examines signal LIL to determine if the AIO can be accepted. The available out (AVO) signal is sent to the next lower priority controller if the AIO cannot be accepted, that is, if the DS has no interrupt pending, or if a lower priority device has raised the high priority interrupt line HPIL (either of these conditions cause signal NAIOM to be true).

AVOD = NAIOM AIOR FSR AVIR + ...

NAIOM = NLIH

Signal AVIR is true at all times in the highest priority controller. In all other controllers, signal AVIR is true only after signal AVO has been received from higher priority controllers. The action of sending signal AVO sequentially from higher to lower priority controllers continues until AVO is true in a DS in which signal AIOM is also true. At this point, AVOD is inhibited, and function strobe acknowledge FSL is raised.

FSL = FSLD = BSYC + ...

BSYC = AVIR AIOR FSD AIOM + ...

Signal BSYC indicates that available input (AVI) has been received from a higher priority controller, that the AIO function indicator has been raised (AIOR), that the function strobe has been received (FSD), and that this is the highest priority controller with an interrupt pending (AIOM). Signal BSYC is also applied to the status logic, where it enables the device controller address to be placed on the function response lines.

The DS also generates signal AIOC which indicates that its interrupt call is being acknowledged.

AIOC = AVIR AIOR NIOM
+ AIOC FSDD INI NRSTR

The controller continues to hold flip-flop WANTI true until the DS raises AIOC. Signal AIOC, in conjunction with FSD, resets flip-flop WANTI. Signal AIOC also enables the sending of condition code information during the AIO function.

After signal FSL is received by the MIOP, the MIOP drops signals FS and AIO, concluding the AIO function.

3-27 SERVICE CALL. When a service call is required, the service request logic in the controller raises signal CSL which causes latch LSL to go true (figure 3-7).

LSL = CSL INC NFSC NASCR
+ LSL NFSC NRSTR ASCR INI

Signals NFSC and NASCR are true until the MIOP responds to the service request that is being generated. The service call line (SC) is driven true when signal LSL is true.

SC = SCD = LSL

The MIOP responds to the service call (when it is ready) by raising the ASC function indicator line (signal ASCR true). This causes latch LSL to set. When the MIOP raises the function strobe (FS), the highest priority DS examines signal LSL to determine if the ASC can be accepted. If the highest priority DS is not requesting an interrupt (signal LSL false), signal AVO is sent to the next lower priority controller.

AVOD = NASCM ASCR FSR AVIR + ...

NASCM = NLSL + ...

The action of sending AVO sequentially from higher to lower priority controllers continues until AVO reaches a DS in which signal ASCM is true. At this time, AVOD is inhibited and FSLD is raised.

FSL = FSLD = BSYC + ...

BSYC = AVIR ASCR FSR ASCM + ...

Signal BSYC indicates that AVI (AVIR) has been received from a higher priority controller or from the MIOP, that the ASC function indicator has been raised (ASCR), and that the function strobe has been received from the MIOP (FSR), and that a lower priority

controller is not driving high priority service line HPS. If HPS is true, ASCM is false, in which case AVO is sent to the next lower priority controller. Otherwise, the function strobe acknowledge line (FSL) is raised. Signal BSYC is also applied to the status logic where it enables the device controller address to be placed on the function response lines.

In response to FSL, the MIOP drops FS. The fall of FS causes the service connect flip-flop, FSC, in the DS to set as described below. Signal ASCB, which is true if ASCM is true, is applied to the set side of flip-flop FSC.

S/FSC = ASCB

C/FSC = FSR NFSC + ...

ASCB = AVIR ASCR FSR ASCM
+ ASCB NFSC INI NRSTR

The term AVIR ASCR FSR ASCM causes ASCB to go true. After ASCB is true, it is latched by the term ASCB NFSC INI NRSTR. The flip-flop is set, when FS (FSR) which is applied to the clock input of flip-flop FSC goes false. When flip-flop FSC is set, the controller is connected to the MIOP for service and proceeds with the service operation by raising request strobe RS and the associated signal lines. The raising of RS is a function of the receive circuits in the controller and goes high when these circuits furnish signal RSDD to the DS. The service connect flip-flop remains set until the MIOP drives the end service line, ES. Signal ES (ESR) is applied to the reset side of flip-flop FSC, and signal RSDD is applied to its clock input. When RSDD falls, FSC is reset, and the controller is disconnected from the MIOP (the service cycle is concluded).

R/FSC = ESR FSC

C/FSC = RSDB FSC

RS = RSDD = RSDC = RSDB = RSDA

Signal CSL, which is applied to the DS to initiate a service request, goes false shortly after flip-flop FSC is set, unless the operation is halted by an HIO function. The delay circuit input is connected to the zero output (NFSC) of the service connect flip-flop so that when it is set (NFSC goes false) the output of the delay circuit, CLSI, goes false approximately 50 nanoseconds later, at which time CSL goes false. Similarly, when FSC is reset, CLSI goes true, however, the minimum delay is 100 nanoseconds. The purpose of the delay is to inhibit controller logic switching transients from appearing on the service call line.

3-28 STATUS LOGIC. The status logic (figure 3-8) is used to control function response lines FR0 through FR7. During an SIO, HIO, or TIO function, the status logic gates status information to the FR-lines. The meaning of the status bits is given in table 2-2. During SIO, HIO, or TIO functions which are addressed to this controller, signal TSH is supplied by the function logic in the DS. Signal FSD is also supplied by the controller. In the DS, FSD is applied to a buffer. The resultant output is signal BFSD. Signal FSD (BFSD) drops when the MIOP drops function strobe FS. Signals TSH and BFSD are used as enable signals by the function logic to gate the status information to the FR-lines as shown in the following equations.

FR0D = TSH BFSD WANTI + ...

FR1D = TSH BFSD START + ...

FR2D = TSH BFSD START + ...

FR3D = TSH BFSD + ...

FR4D = TSH BFSD UEND + ...

FR5D = TSH BFSD START + ...

FR6D = TSH BFSD START + ...

FR7D = TSH BFSD GRD + ...

During AIO and ASC functions, the status logic gates the device controller address to the FR-lines. During these two functions, signal BSYC is supplied by the service-interrupt priority logic. This signal is used as an enable signal to gate the device controller address, supplied by switches SWA0 through SWA7 in the address selection logic to the FR-lines.

FR0D = BSYC SWA0 + ...

FR1D = BSYC SWA1 + ...

FR2D = BSYC SWA2 + ...

FR3D = BSYC SWA3 + ...

FR4D = BSYC SWA4 + ...

FR5D = BSYC SWA5 + ...

FR6D = BSYC SWA6 + ...

FR7D = BSYC SWA7 + ...

3-29 FUNCTION LOGIC. The function logic consists of the circuits shown in figure 3-9. This logic samples function indicator lines SIO, HIO, TIO, and TDV, from the MIOP, and the address recognition

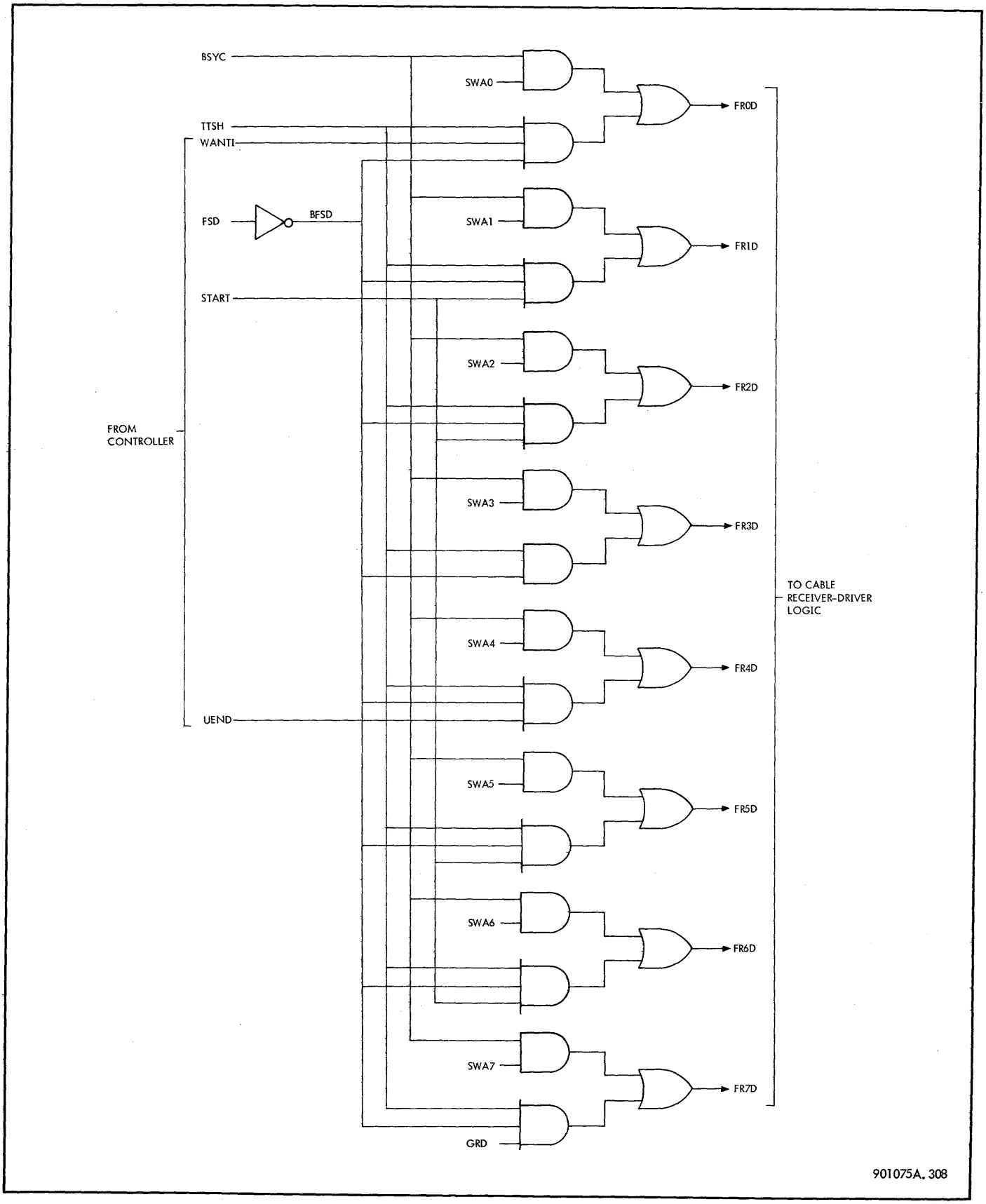


Figure 3-8. Status Logic, Logic Diagram

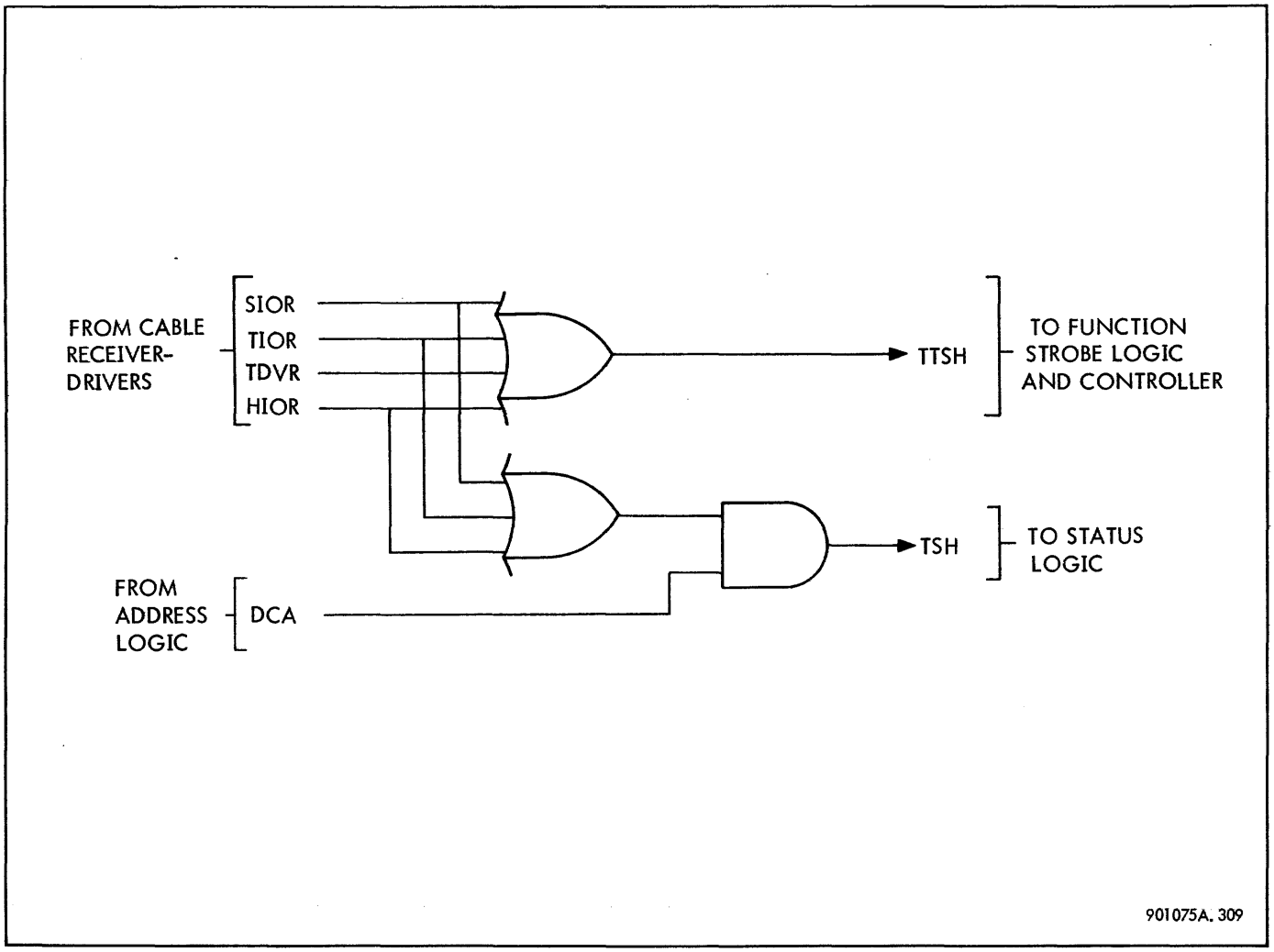


Figure 3-9. Function Logic, Logic Diagram

signal DCA generated by the address selection and recognition logic. Based on the state of these signals, the function logic generates signals TTSH and TSH. Signal TTSH is true if any one of the function indicator lines SIO, HIO, TIO, or TDV is true.

$$TTSH = SIOR + HIOR + TIOR + TDVR$$

Signal TTSH is applied to the receive logic in the controller to provide condition code information in response to the instruction and to the function strobe logic in the subcontroller to generate the function strobe acknowledge signal.

Signal TSH is true if any one of function indicator lines SIO, TIO, or HIO is true, and if signal DCA (address recognition) is true.

$$TSH = DCA\ (SIOR + TIOR + HIOR)$$

Signal TSH is applied to the status logic where it is used to gate the status information to the FR-lines. Since signal TSH does not include a TDV function, no status information is sent to the MIOP during execution of a TDV instruction.

3-30 FUNCTION STROBE AND REQUEST STROBE LOGIC. The function strobe logic (figure 3-10) receives the function strobe, FS, from the MIOP in conjunction with a function indicator. Under proper conditions (for example, proper address received and mode received) the function strobe logic returns the function strobe acknowledge signal, FSL, to the MIOP.

The request strobe logic sends the request strobe, RS, to the MIOP whenever an exchange of information is required (when the controller is connected to the MIOP for service). In response to the request strobe, the MIOP performs the required operation, and then sends the request strobe acknowledge signal, RSA, to the controller.

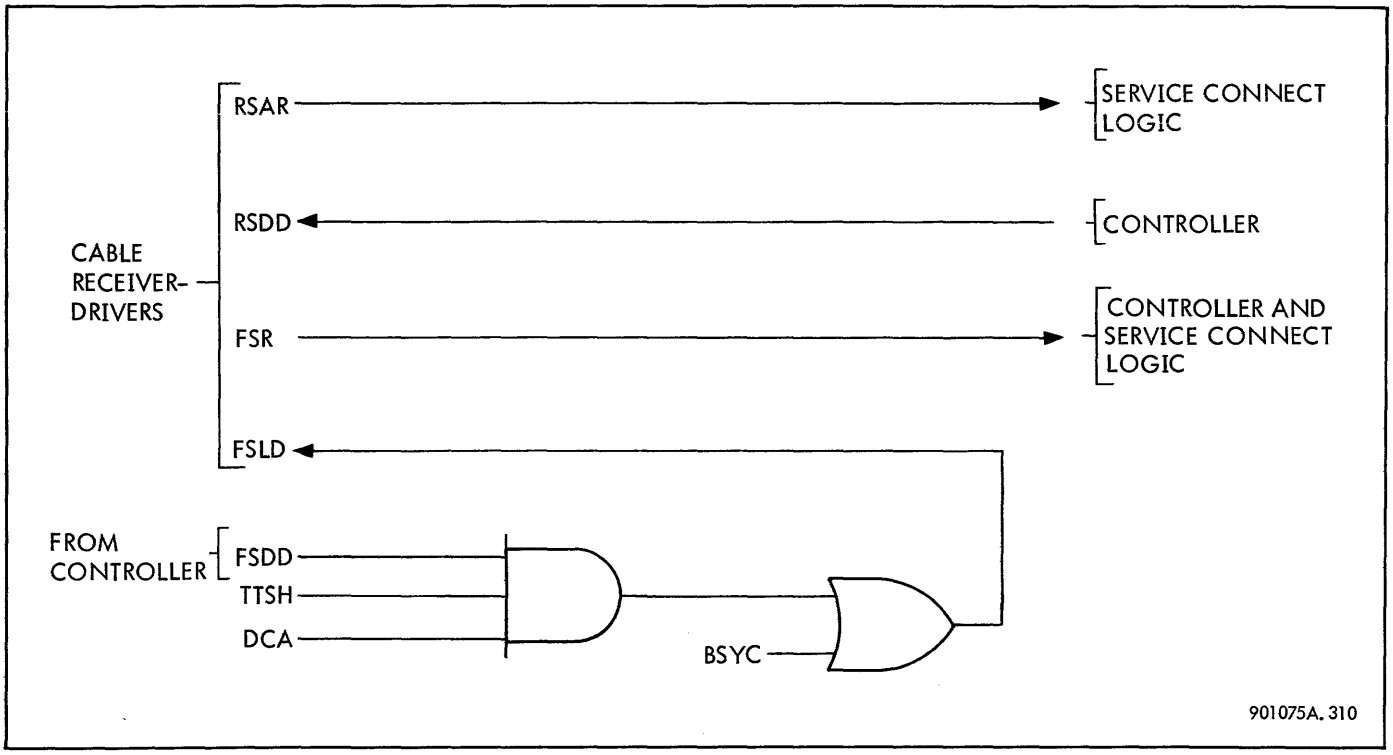


Figure 3-10. Function Strobe and Request Strobe, Logic Diagram (MIOP Interface)

The MIOP raises FS shortly after it raises any of the function indicators. Signal FS (FSR) is applied to the service-interrupt priority logic in the DS and to the controller. During an AIO or ASC function, the service-interrupt logic determines whether the COC has priority. If the COC has priority, busy signal BSYC is generated. This signal is applied to the status logic to gate the controller address to the FR-lines. At the same time, signal BSYC is applied to the FS logic to generate acknowledge signal FSL. Signal FSL follows BSYC.

$$FSL = FSLD = BSYC + \ldots$$

During the other functions (SIO, HIO, TIO, and TDV), the acknowledge signal is generated when signal DCA is received from the address logic, when signal TTSH is received from the function logic, and when signal FSDD is received from the function strobe logic (part of the receive logic in the controller).

$$FSL = FSLD = DCA\ TTSH\ FSDD + \ldots$$

The request strobe logic receives signal RSDD from the receive logic in the controller every time that the request strobe is to be generated. The logic in the controller that generates RSDD receives signal RSARC (NRSARC) from the request strobe logic. Signal RSARC must be false to generate RSDD. Signal RSARC is true if the request strobe acknowledge, RSA (RSAR), is being received from the MIOP or if the service connect flip-flop is being reset.

$$RSARC = RSAR + FSCL$$

$$RSAR = RSA$$

$$FSCL = FSC\ RSAR\ ESRFSC + FSCL\ FSC$$

Signal NRSARC inhibits switching transients from appearing on the request strobe line if flip-flop FSC is slow in resetting. Request strobe generation ceases when flip-flop FSC is reset.

3-31 RELAY LOGIC. The relay logic (figure 3-11) contains the circuits and the relays for connecting and disconnecting the device controller from the MIOP interface in a transient-free manner when power is applied or is removed. The signal that starts the operation originates in the controller, although the subcontroller contains the circuits that perform the connect and disconnect sequencing.

In addition, the relay logic provides a direct path from the input of the cable receiver that receives AVI (from a higher priority DS) to the output of the cable driver that drives AVO (to a lower priority DS) when power is not applied to this controller. This path is through relay contacts that are held open when power

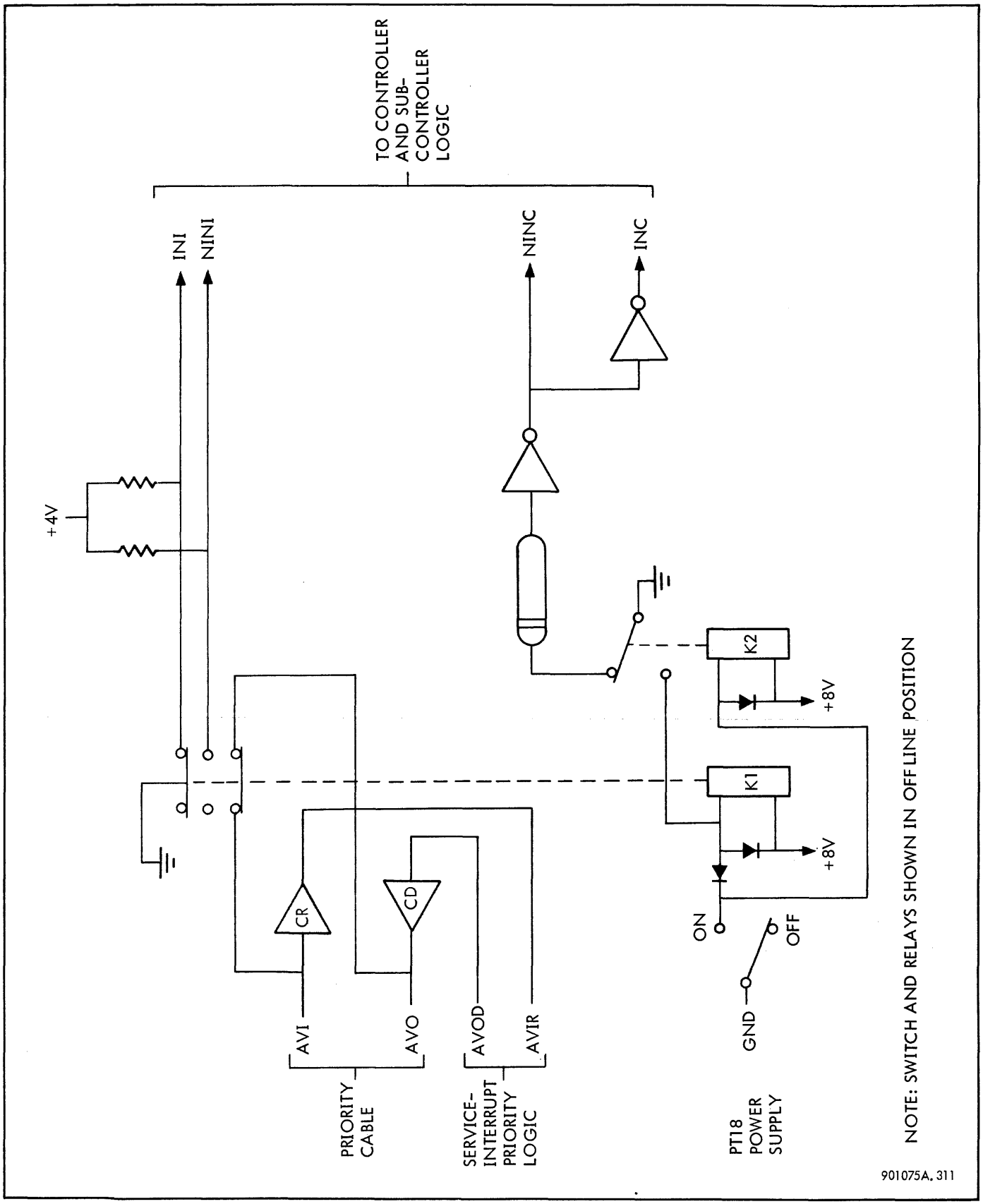


Figure 3-11. Relay Logic, Logic Diagram

is on and are held closed when power is off. When power is on (relay contacts open), AVO is sent to the next lower priority DS only if this DS does not have priority when AVI is received.

Signals INI, NINI, INC, and NINC, which are generated by the relay logic, are applied to the cable drivers, the service-interrupt priority logic, and the service connect logic in the DS as well as to the controller. The sequence of operations that occurs when the DS is connected and is disconnected from the MIOP are described in paragraphs 3-32 and 3-33. (See figure 3-12.)

3-32 Connect Principles of Operation. When the DS is to be connected to the MIOP interface, a ground source, originating in the PT18 power supply, is applied through an ON-OFF switch to the relay coils in the relay logic. The ground source is not applied until the output of the power supply is at its rated value. The ON-OFF switch (located on the LT25 module in slot 23C) must be set to ON to apply the ground. The sequence of events that connects the DS is as follows:

a. Approximately 4.5 ms after the ground source is applied, a set of relay contacts close, and signal NINI is grounded.

b. Approximately 0.5 ms later, signal INI goes true, and the short circuit between lines AVI and AVO is removed.

c. Approximately 120 μs after signal INI goes true, signal INC goes true, and signal NINC is grounded.

When INI and INC have reached the true state, the DS is connected to the MIOP interface, and the service call, the interrupt call, and the cable driver lines become active.

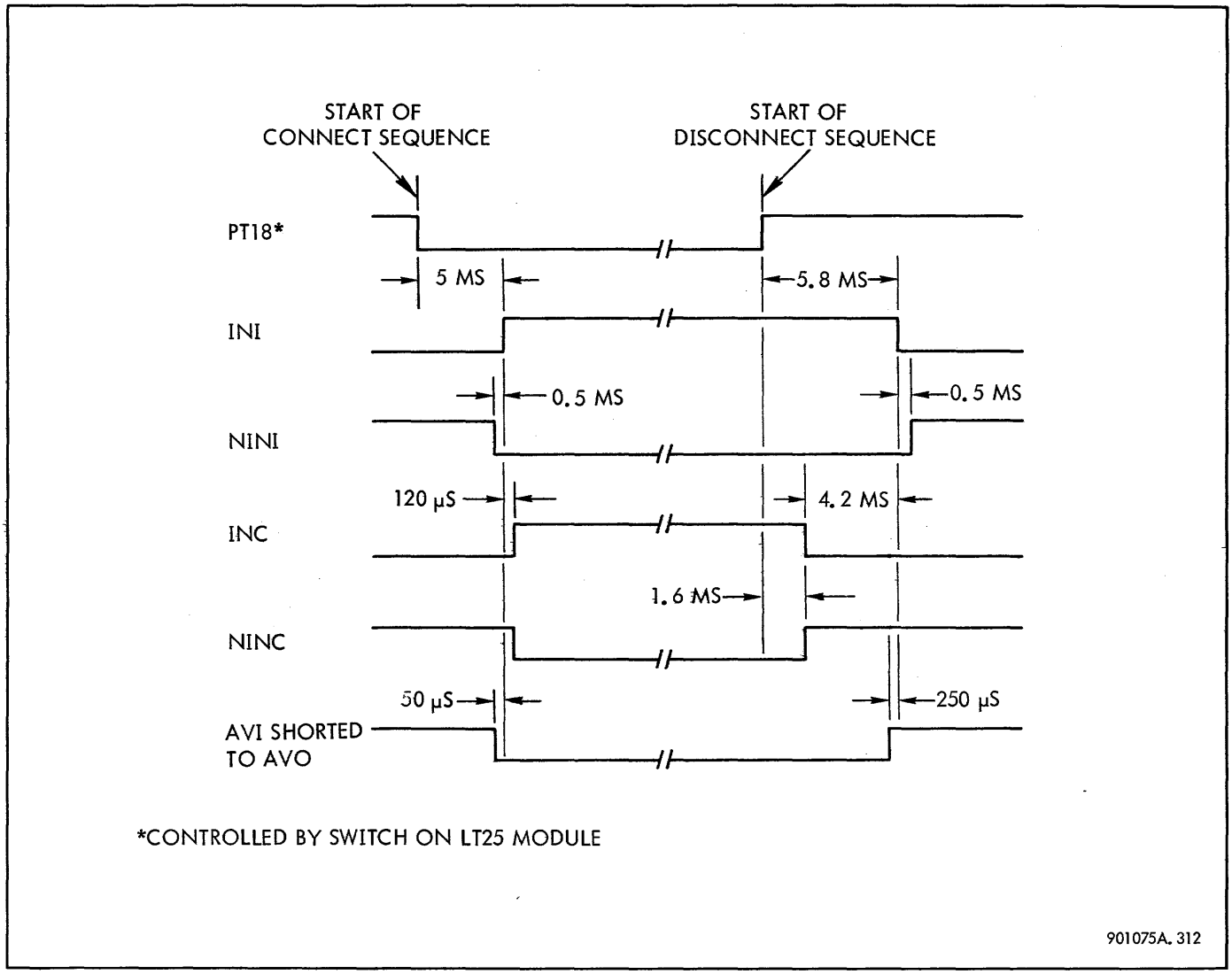


Figure 3-12. Connect-Disconnect, Timing Diagram

3-33 Disconnect Principles of Operation. The DS is disconnected from the MIOP interface when the ON-OFF switch is set to OFF, or the ground source is removed at the PT18 power supply. The sequence of events that disconnects the DS is as follows:

a. Approximately 1.6 ms after the ground is removed, all service and interrupt calls to the MIOP are inhibited by grounding signal INI and by letting NINC go true through relay and transistor logic.

b. Approximately 4.2 ms after signal INC is grounded, signal INI becomes grounded through a set of relay contacts. At approximately the same time, the AVO line is shorted to the AVI line through a second set of relay contacts. The timing of the two sets of contacts can vary by as much as 250 μs.

c. Approximately 0.5 ms after signal INI is grounded, signal NINI is allowed to go true. Signal INI is applied to the cable driver inputs at the MIOP interface to disable them.

As a result of the disconnect operation, the DS is effectively disconnected from the MIOP interface and, since AVI is shorted to AVO, the priority chain for the rest of the controllers is not broken.

3-34 Device Controller

The registers and the logic elements that form the controller (not including the LIU) are separated into two primary groups, the send logic and the receive logic. A detailed description of each register and group of logic elements is contained under the appropriate headings in paragraphs 3-35 and 3-46. The reset logic, described in paragraph 3-45 (with the send logic descriptions), applies equally to the send and to the receive circuits.

3-35 RECEIVE LOGIC. Each block in figure 3-3 that forms part of the receive logic is described in the following paragraphs.

3-36 Receive Scanner and Decoding Logic. This logic (figure 3-13) consists of six receive scanner flip-flops (RS1 through RS6) that function as a module 64 counter, two control flip-flops (SERV and FASTSERV), input gating that controls the counter, and output gating that sends the LIU number and the line number to the LIU's and to the MIOP.

Each count of the counter specifies the address of an input line on which data is received from a data set or from another terminal device. The address of each receiver module in the LIU is the same as the input line from which it receives the data.

The scanner operates in the normal mode or in the high speed mode, depending on whether the high speed option is installed. When operating in the normal mode, the scanner searches for a receiver module that has assembled a character and is ready to request service. The receiver requests service by dropping signal NRSRM, which is applied to the scanner enable logic. When signal NRSRM from any receiver in any LIU goes low, the scanner is stopped. The receiver can drop this line only when it is receiving its address from the scanner. Therefore, when the scanner is stopped, its count is equal to the line number and receiver requesting service.

The signal that enables the scanner is RSRUN. Signal RSRUN is applied to both the set and the reset inputs of flip-flop RS1, which is the first flip-flop in the counter chain. The counter is a simple ripple counter; that is, each flip-flop in the counter applies its output signal to the clock input of the following flip-flop.

S/RS1 = NRS1 RSRUN

R/RS1 = RSRUN

C/RS1 = CLOCK

M/RS1 = RSS63

E/RS1 = GND

S/RS2 = NR2

R/RS2 = .

C/RS2 = RS1

M/RS2 = RSS63

E/RS2 = RESET

⋮ ⋮

S/RS6 = NRS6

R/RS6 = .

C/RS6 = RS5

M/RS6 = RSS63

E/RS6 = RESET

The clock signal that is applied to the clock input of flip-flop R1 is the one MHz clock signal from the MIOP.

When none of the receiver modules in an LIU are requesting service, the output signal NRSRM from each LIU is high. The eight signals from the eight LIU's are designated NRSRM0 through NMRMS7 in the

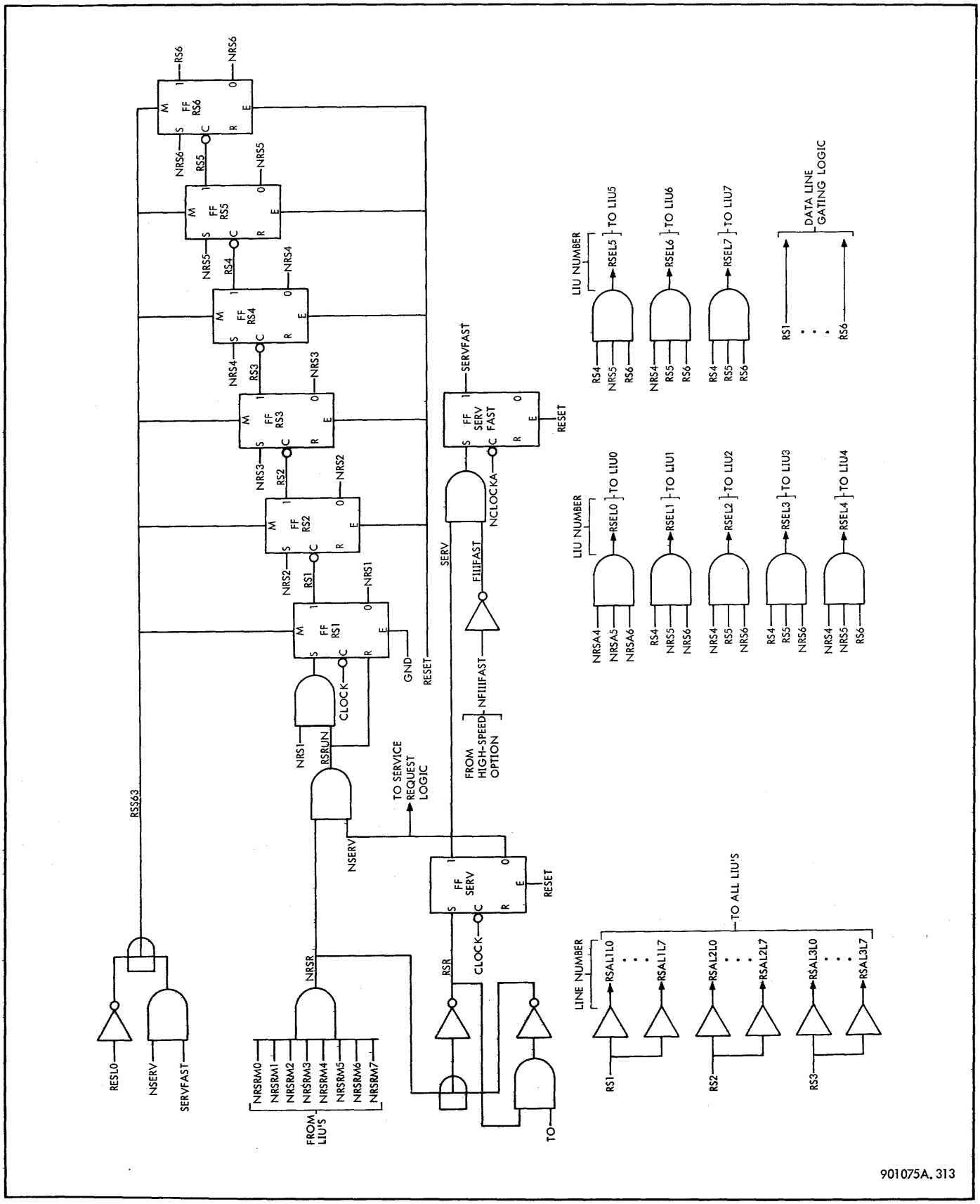


Figure 3-13. Receive Scanner and Decoding Logic, Logic Diagram

controller. For example, NRSRM0 is the receiver service request line for the eight receiver modules in LIU0. The eight signals (NRSRM0 through NRSRM7) are applied to an AND gate in the controller so that when none of the receivers are requesting service, NRSR, the output of the AND gate is true.

NRSR = NRSRM0 NRSRM1 NRSRM2 NRSRM3
NRSRM4 NRSRM5 NRSRM6 NRSRM7

Signal NRSR is gated with signal NSERV to generate signal RSRUN that permits the counter to scan.

RSRUN = NRSR NSERV

Signal NSERV, generated by flip-flop SERV in the reset state, is used to synchronize the start of the scanner after completion of a service request. Flip-flop SERV is in the set state approximately when a receiver module requests service. It is set by the first clock after the rise of RSR and is reset by the first clock after the fall of RSR.

S/SERV = RSR

R/SERV = NRSR

C/SERV = CLOCK

When a receiver module is ready to request service, it drops signal NRSRM (equal to one of signals NRSRM0 through NRSRM7 in the controller) when the scanner generates the receiver's address. This causes NRSR and RSRUN to go false and to stop the scanner. During the time that the scanner is stopped, it sends the address of the interrupting receiver module to the MIOP. The scanner output is applied to the character receive logic in the controller which, in turn, controls the data lines in the subcontroller. Sending the receiver address (line number) is controlled by enabling signal TSCAN generated by the character receive logic.

The decoding logic decodes scanner signals RS4 through RS7 and their complements to generate signals RSEL0 through RSEL7. Each of signals RSEL0 through RSEL7 is hard wired to its respective LIU. For example, signal RSEL0 is applied only to LIU0, signal RSEL1 is applied only to LIU1, and so forth. (Signals RSEL0 through RSEL7 are designated as RSEL in their respective LIU's.)

Each LIU address signal is true for eight counts, since the LIU address is a function of the three MSB's of the counter. Each of the eight counts, generated by RS1 through RS3, designates a separate line number (receiver number) in the LIU specified by one of the LIU address signals. The three line number address signals, RS1 through RS3, are wired to all eight LIU's through three groups of buffers with eight buffers to a group. For example, RS1 is applied to the group of buffers that produces output signals RSAL1L0 through RSAL1L7. Signal RSAL1L0 is applied to LIU0; signal RSAL1L1 is applied to LIU1 and so forth. The same is true for RS2 (produces RSAL2L0 through RSAL2L7) and RS3 (produces RSAL3L0 through RSAL3L7). In each LIU, the three signals are designated RSAL1 through RSAL3. The receiver modules in each LIU are addressed by signals RSAL1 through RSAL3 and by their complements. The complements are produced by inverters in each LIU.

The logic equations for RSEL0 through RSEL7 are as follows:

RSEL0 = NRS4 NRS5 NRS6

RSEL1 = RS4 NRS5 NRS6

RSEL2 = NRS4 RS5 NRS6
.
.
.
RSEL7 = RS4 RS5 RS6

The logic equations for RSAL1L0 through RSAL7L0, RSAL2L0 through RSAL2L7, and RSAL3L0 through RSAL3L7 are as follows:

RSAL1L0 = RS1
⋮
RSAL1L7 = RS1

RSAL2L0 = RS2
⋮
RSAL2L7 = RS2

RSAL3L0 = RS3
⋮
RSAL3L7 = RS3

When the high speed option is installed in the COC, the normal sequencing of the scanner is modified. Each time that a receiver module which is not in LIU0 is serviced, the scanner is preset to address 63. This is the address of the last receiver module in LIU7. This operation permits LIU0 to be scanned at a higher rate than the other LIU's. No presetting takes place if the receiver module being serviced is in LIU0. Thus, LIU0 is the only LIU that can accommodate the high speed option. Receiver address 63 may initiate a service request before the scanner is stepped to address zero; however, operation of the high speed option is not appreciably affected.

Presetting the scanner to 63 is accomplished by gating signal FIIIFAST and signal SERV in an AND configuration

to the set input of the service fast flip-flop (SERVFAST). Signal FIIIFAST is supplied by the high speed option and is true when a CT17 module is installed in slot B24. Flip-flop SERVFAST is set one-half clock time after flip-flop SERV has been set. The clock used by SERVFAST is the complement of the clock used by SERV.

S/SERVFAST = SERV FIIIFAST

C/SERVFAST = NCLOCKA

When signal NRSR goes true, indicating the end of a service request, RSR goes false causing flip-flop SERV to reset at the next clock time. At this point, RSRUN goes true enabling the scanner. If the high speed option has not been installed, the scanner continues to count from its previous address on the first clock time following the reset of flip-flop SERV. With the high speed option installed, however, signal SERVFAST causes signal receive scanner set to 63 (RSS63) to go true before the scanner starts. Signal RSS63 presets the scanner to address 63 before the first count clock (CLOCK) appears at the clock input of flip-flop RS1.

RSS63 = SERVFAST NSERV NRSEL0

If the receiver being serviced is in LIU0, signal NRSEL0 is false. This disables RSS63 and prevents any modification of the scanner when processing service requests in LIU0.

3-37 Service Request Logic. The service request logic consists of flip-flop wants service, WANTS, and the associated input and output gating shown in figure 3-14. When a service request to the MIOP is to be made, the service request logic generates signal CSL and sends it to the DS. The service-interrupt priority logic (paragraph 3-27) then generates the service call. The service request logic receives inputs from the DS and from the send and receive logic in the controller.

During an input operation, the receiver module in the LIU is turned on by a turn receiver on function by way of a WD instruction. This permits the receiver to assemble an input character and then to request service. The request for service causes flip-flop SERV to set (see paragraph 3-36). Signal NSERV is applied to the clock input of flip-flop WANTS. When flip-flop SERV is setting, the falling edge of signal NSERV causes flip-flop WANTS to set (see figure 3-15), which in turn causes CSL to go true.

S/WANTS = NWANTS

C/WANTS = NWANTS NSERV + ...

CSL = (WANTS + ...) CSLI

As a result of the service call, the COC is connected to the MIOP for service. During the process of being connected (the acknowledge service call sequence), the acknowledge service call signal from the MIOP causes signal ASCB in the DS to go true. Signal ASCB drops when the service connect flip-flop in the DS is set (COC connected to service). The fall of ASCB resets flip-flop WANTS, because the reset input of WANTS is floating.

R/WANTS = .

C/WANTS = ASCB INI

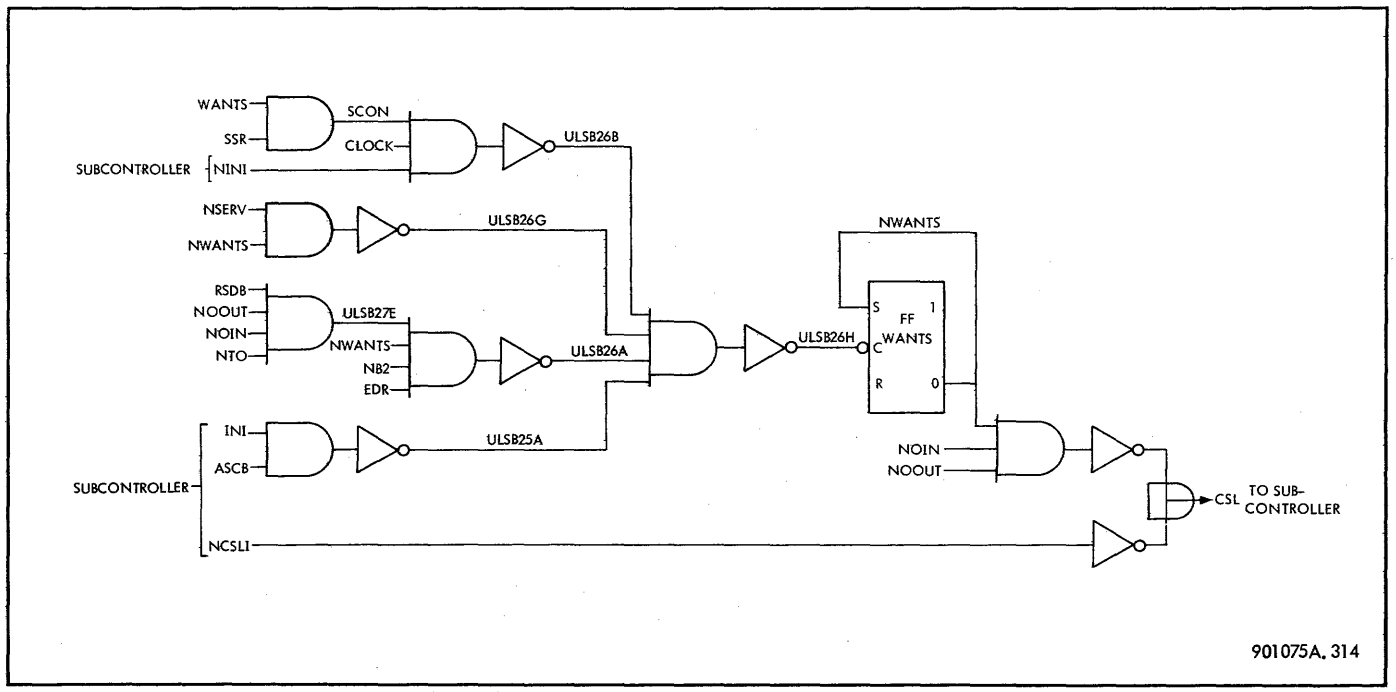


Figure 3-14. Service Request Logic, Logic Diagram

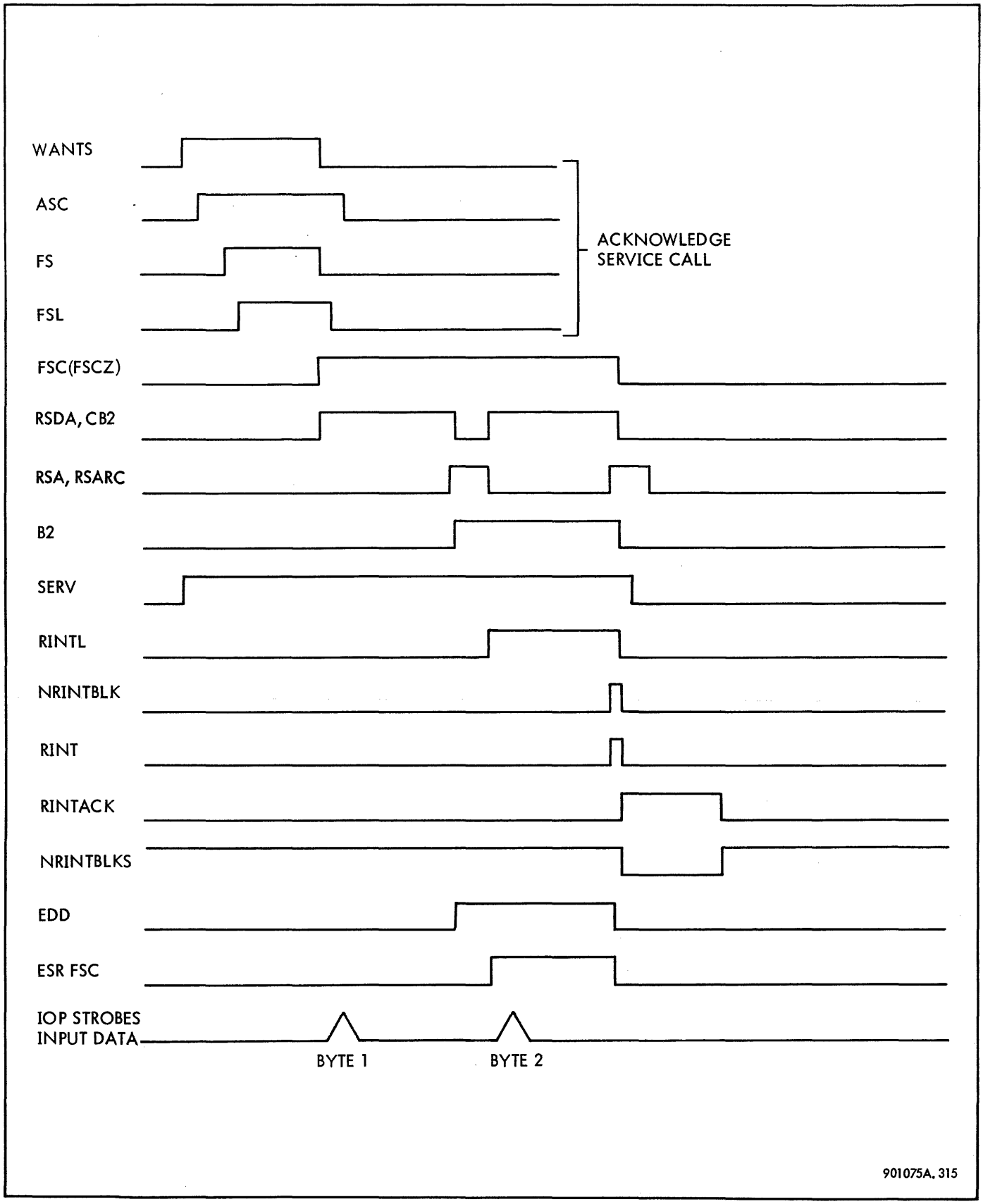


Figure 3-15. Input Operation, Timing Diagram

The COC sends two bytes of data to the MIOP on the data lines during the time that the COC is connected to the MIOP for service. The first byte is the character received by the receiver module in the LIU, and the second byte is the line number specified by the receive scanner in the controller receive logic. (See paragraph 2-16.)

Logic is also provided for controlling flip-flop WANTS when operating with the Perpherial Equipment Tester (PET). When operating with the PET, the COC must be operated offline by setting the ON-OFF switch in location 25 of chassis C (subcontroller) to OFF. (See paragraph 3-32.) When this switch is set to OFF, signal NINI is true. Signal NINI is gated with signal service call on (SCON), and the clock signal to clock flip-flop WANTS.

Signal CSL is raised to request service for an order-out or an order-in service cycle. During an SIO function, flip-flop OOUT is set so that the service request following the SIO is an order out. Signal OOUT causes CSL to go true.

CSL = (OOUT + ...) CLSI

Following a terminal order in which count done or IOP halt has been specified, the controller specifies an order-in service cycle by setting flip-flop OIN. Signal OIN causes CSL to go true.

CSL = (OIN +...) CLSI

The purpose of signal CSLI is to prevent switching transients from appearing on the service call line. (See paragraph 3-27.)

3-38 Byte Two Logic. This logic consists of flip-flop byte two (B2) and associated input gating (see figure 3-16). The purpose of B2 is to define the byte being transferred to the MIOP during a data-in service cycle as the first or the second byte of the service cycle. Two bytes are input during each data-in service cycle. The first byte is the character, and the second byte is the line number from which that character was received. See paragraph 2-16 for the formats of the two bytes.

The byte two logic receives signals from the state logic and from the request strobe logic. Signals generated by the B2 logic are used by other controller circuits including the data line gating logic, the end data (state) logic, the receive interrupt logic, the function strobe (DIO) logic, and from the receive module in the LIU that is inputing the character.

The state of flip-flop B2 is changed only during a data-in service cycle. Shortly after the service connect flip-flop (FSC) is set during the acknowledge service call sequence of the data-in service cycle, signal RSDA (RS at the interface) is generated by the request strobe logic. (See figure 3-14.) The request strobe permits the MIOP to read the data on the data lines. Flip-flop B2 is initially in the reset state. When the first request strobe of the service cycle is generated, signal NB2 is used by the data line gating logic to gate the character on to the data lines. After the MIOP has input the character, it sends the request strobe acknowledge signal. This causes the controller to drop RSDA. When RSDA drops, flip-flop B2 sets.

S/B2 = NB2

C/B2 = CB2

CB2 = NOOUT NOIN NTO RSDA

Signals NOOUT, NOIN and NTO are true during the data portion of a data-in service cycle.

When the MIOP drops the first request strobe acknowledge signal (RSA), the request strobe logic in the controller generates the second request strobe of the service cycle. Flip-flop B2 is now true, and signal B2 is used by the data line gating logic to gate the line number on to the data lines. The second request strobe permits the MIOP to strobe the data lines for the line number. Shortly thereafter, the MIOP raises RSA in response to the second RS. This causes RSDA to drop. The dropping of RSDA resets flip-flop B2 so that it is ready for the next data-in service cycle.

R/B2 = .

C/B2 = CB2

Signal B2 is used by the end data logic to generate signal EDD. This signal informs the MIOP that the last byte of the service cycle is being transferred.

Signal B2 is applied to the inputs of eight buffers (READL0 through READL7). Each buffer output is sent to its respective LIU where the signal becomes READM at each of the receiver modules in the LIU's. The fall of signal B2 (and therefore of READM) sets flip-flop FDUMP in the receiver being serviced. Signal FDUMP terminates the receiver module's request for service, since the fall of B2 indicates that the two bytes for this service cycle have been transferred to the MIOP.

Signals B2 and CB2 are used by the receive external interrupt logic to generate a receive interrupt at the conclusion of the two-byte transmission to the MIOP.

3-39 Receive External Interrupt Logic. This logic consists of flip-flop, receive interrupt block RINTBLK (NNRINTBLK) and other logic elements shown in figure 3-17. The purpose of this logic is to generate an external interrupt signal (receive interrupt RINT) after every two-byte transmission to the MIOP that results

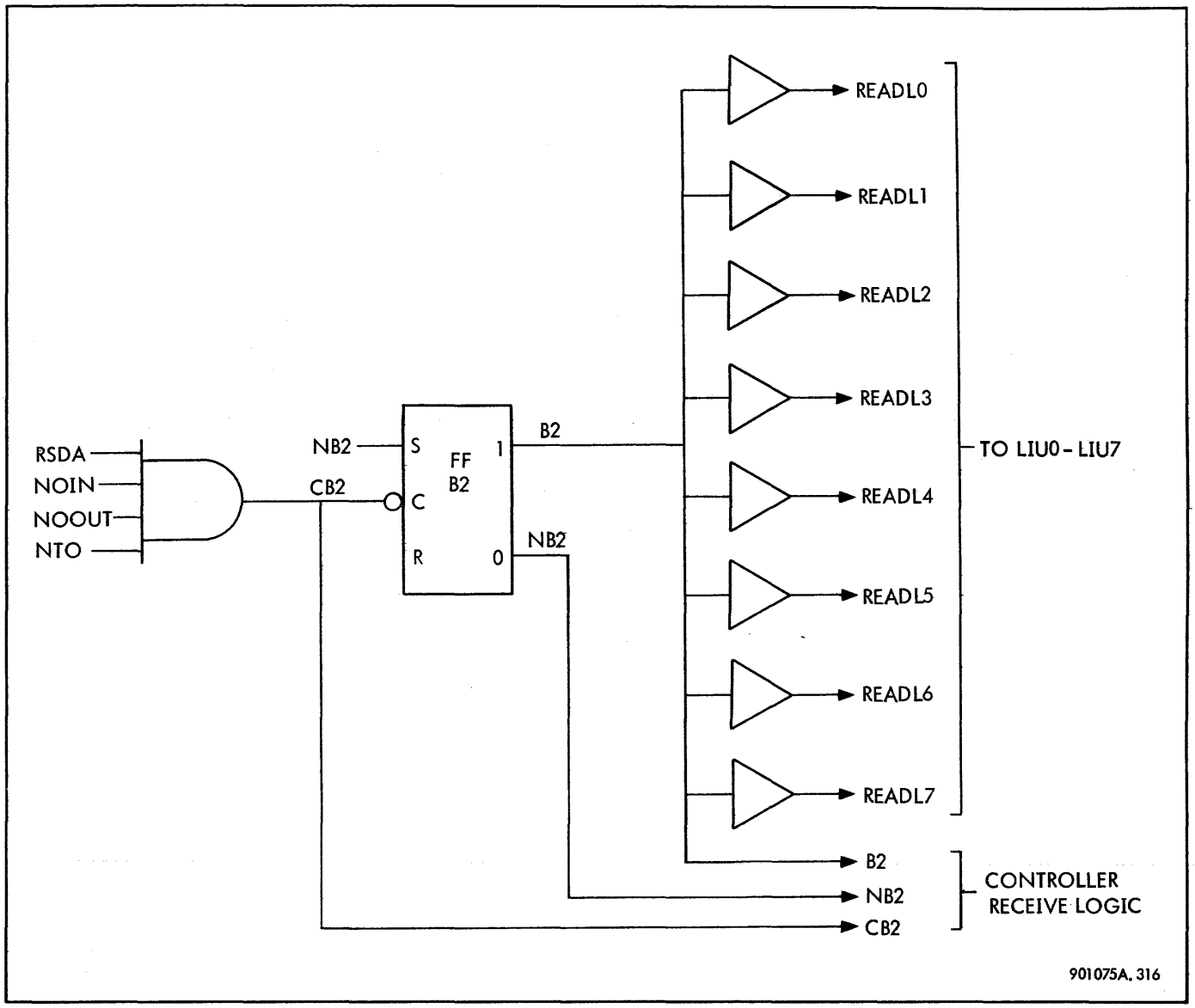


Figure 3-16. Byte Two Logic, Logic Diagram

from a service request. This logic also receives the acknowledge signal (receive interrupt acknowledge RINTACK) from the CPU. When signal RINT is generated, it is sent to the CPU through a cable driver and is held true until the acknowledge signal, RINTACK, is received, a reset occurs, or an HIO addressed to the controller is received. The external receive interrupt logic receives signals from the CPU, the subcontroller, the receive scanner, state logic, and from B2 logic in the controller.

Generation of the interrupt starts when signal CB2 goes high during transmission of the second byte of data to the MIOP, that is, when flip-flop B2 in the byte two logic is set. (See figure 3-15.) Signals B2 and CB2 cause RINTL to go true if the reset signal is not true and if an HIO is not addressed to this controller. When RINTL goes true, it is latched by the term RINTL NRINTBLKS.

$$\text{RINTL} = \text{CB2 B2} + \text{RINTL NRINTBLKS}$$

Signal NRINTBLKS is true during the time that the receive interrupt acknowledge signal (RINTACK) is false. When acknowledge signal RINTACK goes true, signal NRINTBLKS is dropped, which disables latch RINTL.

$$\text{RINTBLKS} = \text{RINTACK} + \text{RINTBLKS SERV}$$

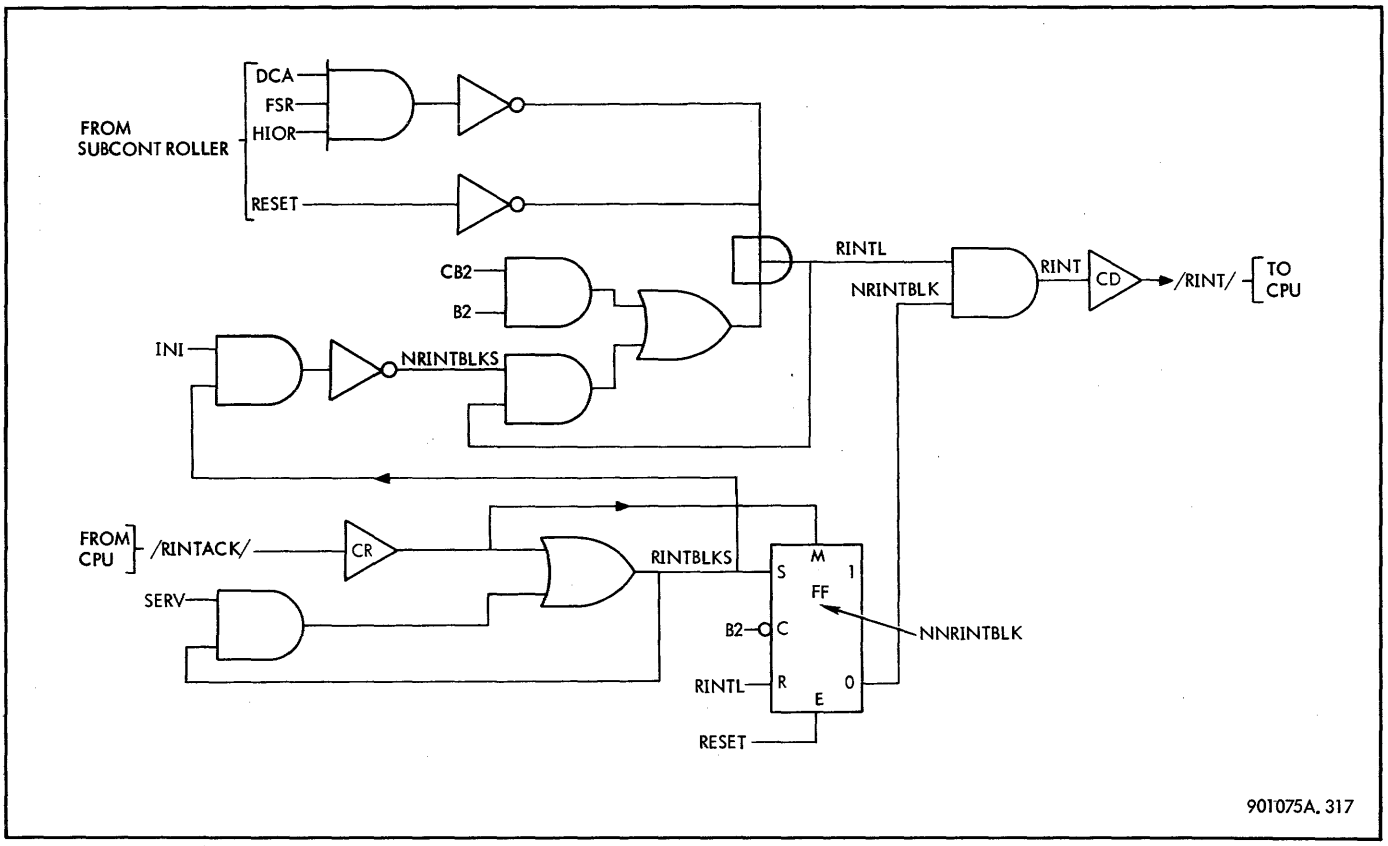


Figure 3-17. Receive External Interrupt Logic, Logic Diagram

The receive interrupt block flip-flop (NNRINTBLK) is used to enable or to disable receive interrupt signal RINT. This flip-flop remains set from the time that the previous receive interrupt acknowledge signal (RINTACK) was received and is not reset until RINTL is raised in preparation for a new interrupt. At that time, the fall of B2 resets flip-flop NNRINTBLK.

R/NNRINTBLK = RINTL

C/NNRINTBLK = B2

When flip-flop NNRINTBLK is reset (signal NRINTBLK true), the receive interrupt line is raised.

RINT = RINTL NRINTBLK

Shortly after RINT is raised, the CPU sends acknowledge signal RINTACK. Signal RINTACK is applied to the mark input of flip-flop NNRINTBLK causing signal NRINTBLK to go false and to drop the interrupt line RINT. Additional interrupts are inhibited until acknowledge signal RINTACK is dropped by the CPU.

M/NNRINTBLK = RINTACK

Signal SERV, generated by the receive scanner logic, is incorporated as part of the latch (RINTBLKS) that supplies the signal to the set input of flip-flop RINTBLK. This prevents the flip-flop from resetting at the fall of B2 if a service request is in progress. Signal NRINTBLKS is held false which disables latch RINTL. This logic prevents an interrupt from being generated and being sent to the CPU while data is being transferred to the MIOP. The interrupt cycle is concluded when signal RINTACK is dropped, causing signal NRINTBLKS to go true. This is the last signal to be restored in preparation for a new interrupt.

3-40 Internal Interrupt Request Logic. This logic consists of flip-flop WANTI and of the input gating shown in figure 3-18. When an internal interrupt request is to be made by way of the MIOP, the internal interrupt request logic generates signal want interrupt, WANTI, which is sent to the subcontroller. The subcontroller then raises the interrupt call line, IC, to the MIOP. Signal WANTI is also used by the status logic in the subcontroller to indicate that an interrupt is pending (table 2-2) and by other logic in the receive logic section of the controller. The internal interrupt request logic receives signals from the subcontroller and from other logic areas of the receive logic section of the controller.

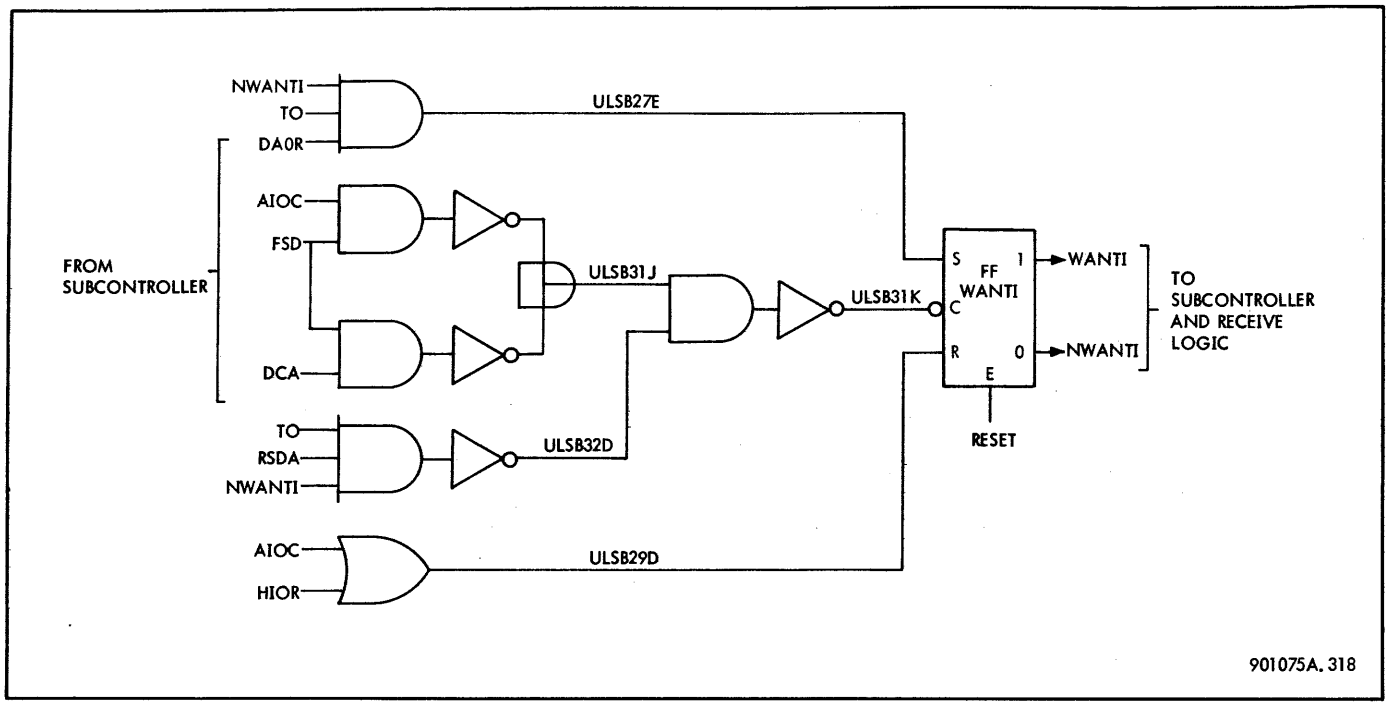


Figure 3-18. Internal Interrupt Request Logic, Logic Diagram

The COC makes an internal interrupt request only when it is given instruction by way of a terminal order. The MIOP tells the COC to interrupt by driving data line DA0 (DA0R) during a terminal order (TO). Flip-flop WANTI is set when the COC drops request strobe RS (RSDA) that was generated for the terminal order.

S/WANTI = ULSB27F

USLB27F = NWANTI TO DA0R

C/WANTI = ULSB31K

ULSB31K = TO RSDA NWANTI + ...

Flip-flop WANTI remains set until the interrupt is acknowledged by an AIO instruction, or until an HIO instruction is addressed to the COC.

R/WANTI = ULSB29D

ULSB29D = AIOC + HIOR

C/WANTI = ULSB31K

ULSB31K = FSD (AIOC + DCA) + ...

Signal DCA is true if this COC is being addressed during the HIO instruction. The AIO instruction does not include an address. In either case, the flip-flop is reset when function strobe FS (FSD) is dropped.

Signal NWANTI is used by the state logic to code line IOR (IORD) when status is being sent to the MIOP and to enable setting flip-flop START when an SIO is being received.

3-41 Data Line Gating Logic. This logic consists of the logic required to gate information on to data lines DA0 through DA7. Flip-flop FSCX, which simulates the service connect flip-flop during testing operations, is also included. (See figure 3-19.) The information to be gated on to the data lines consists of the character received from the receiver module in the LIU that initiated the input service cycle and of the line number associated with that receiver. The line number is obtained from the receiver scanner flip-flops, RS1 through RS6.

The first of the two bytes of data to be input is the character RD1 through RD8. Signal TDATA, controlled by signal NB2, enables gating the character. Signal NB2, received from the byte two logic, is true during the first byte of each data-in service cycle and is false during the second byte of each service cycle.

DA0D = DA0C = TDATA START RD8
+ TSCAN RLSP

DA1D = DA1C = TDATA RD7

DA2D = DA2C = TDATA RD6
+ TSCAN RS6

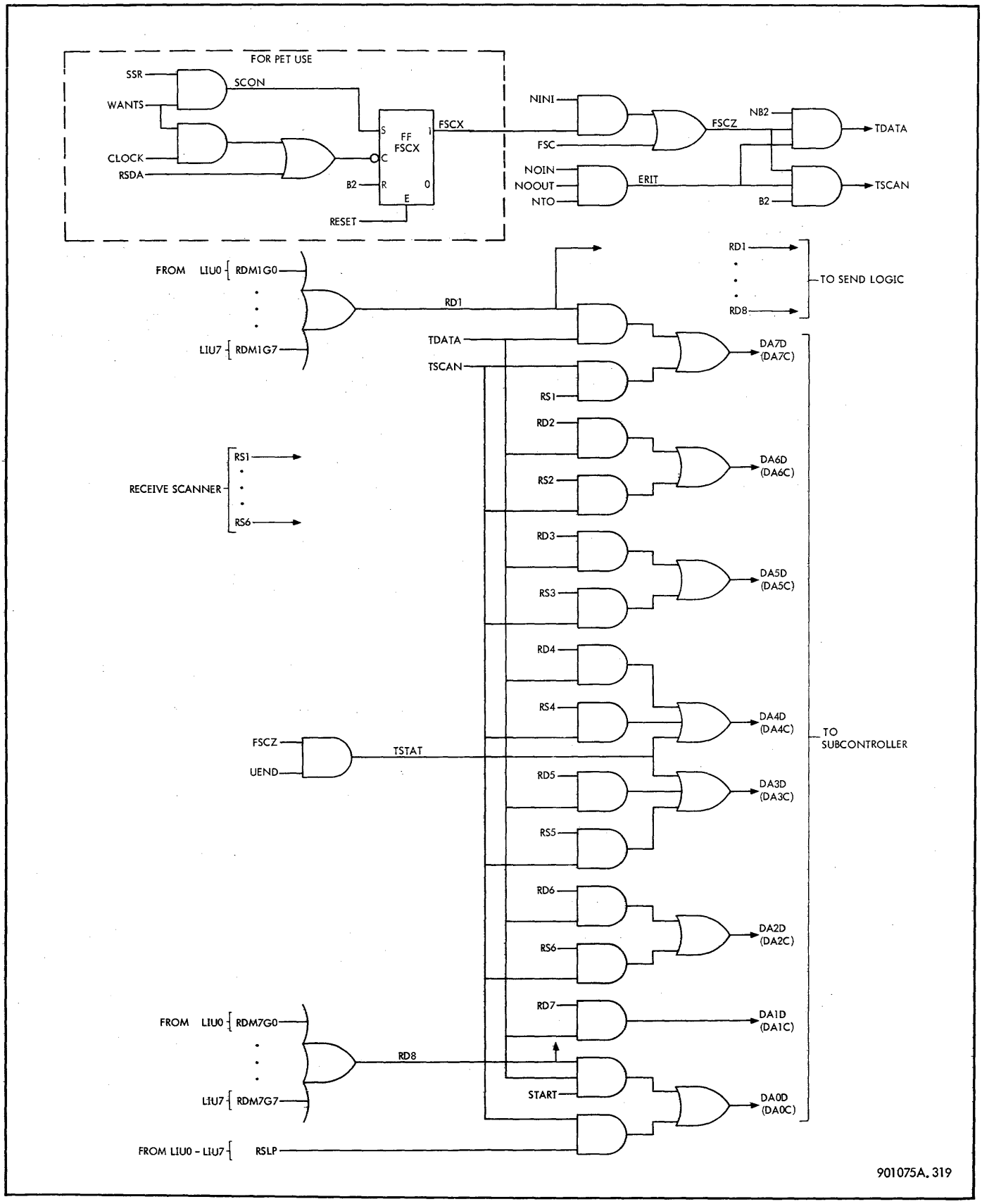


Figure 3-19. Data Line Gating Logic, Logic Diagram

DA3D = DA3C = TDATA RD5
+ TSCAN RS5
+ FSCZ UEND

DA4D = DA4C = TDATA RD4
+ TSCAN RS4
+ FSCZ UEND

DA5D = DA5C = TDATA RD3
+ TSCAN RS3

DA6D = DA6C = TDATA RD2
+ TSCAN RS2

DA7D = DA7C = TDATA RD1
+ TSCAN RS1

The data bits of the character, RD1 through RD8, are derived from the eight data output lines on the receiver modules. The designation G0 through G7 represents the eight LIU's. For example, signal RDMLG0 represents bit 1 of the character being received from LIU0.

RD1 = RDM1G0 + RDM1G1 + RDM1G2 + RDM1G3
+ RDM1G4 + RDM1G5 + RDM1G6 + RDM1G7

⋮ ⋮ ⋮ ⋮ ⋮

RD8 = RDM8G0 + RDM8G1 + RDM8G2 + RDM8G3
+ RDM8G4 + RDM8G5 + RDM8G6 + RDM8G7

Signals RD1 through RD8 are also gated to the send lines (character transmitter logic) during testing operations using the PET.

The second of the two bytes of data to be input is the line number, RS1 through RS6. Signal TSCAN enables gating of the line number. Signal TSCAN is controlled by signal B2, which is true during the second byte.

In addition, the unusual end flip-flop, UEND, drives DA3C and DA4C during an order-in service cycle to specify channel end and unusual end, respectively.

The enable signals, TDATA and TSCAN, are controlled by signals ERIT and FSCZ. Signal ERIT is true only during the data portion of a data-in service cycle; that is, it is not true during the terminal order transmission if a terminal order is included. Signal FSCZ is true if either the service connect flip-flop FSC or flip-flop FSCX is true. Flip-flop FSC applies when the controller is operated online, and FSCX applies when the controller is used offline with PET.

3-42 Request Strobe Logic. The request strobe logic consists of the gating shown in figure 3-20. Signal RSDD is generated by the request strobe logic and is sent to the subcontroller where it becomes request strobe (RS) after passing through a cable driver. Input signals to the request strobe logic are supplied by the subcontroller. When operating offline, signals FSCX and NRSARCX are supplied by the receive logic in the controller. When operating with the PET (offline), request strobes may be generated as desired.

The request strobe logic generates the first request strobe of the service cycle immediately after the service connect flip-flop (FSC) in the subcontroller is set. In response to each RS sent to the MIOP, the MIOP delays and then returns request strobe acknowledge RSA. When RSA is received by the subcontroller, it drops signal NRSARC, thus causing the request strobe logic to drop RS. The MIOP immediately drops RSA when it senses that RS is low. This process continues as long as flip-flop FSC is set. The MIOP concludes the service cycle by raising the end service line (ES). Signal ES is applied to the reset input of the FSC flip-flop, which resets when RSDD (applied to the clock input of FSC) drops. (Signal RSDD is RSD delayed by four inverters.)

During order-in and order-out service cycles, the request strobe logic generates two request strobes. The first is for the order byte and the second is for the terminal order which is always included as part of an order-in/order-out service cycle. During a data-in service cycle the request strobe logic generates two request strobes (the first to input the character and the second to input the line number), and generates a third if a terminal order is to be included.

Signal RSDA is applied to the clock input of a number of flip-flops in the controller (see figure 3-20). In each case, the fall of the signal triggers the flip-flop.

3-43 Function Strobe Logic (MIOP Interface). The function strobe logic consists of the AND gate and the inverters shown in figure 3-21. The function strobe logic receives signals FSR and CSLI from the subcontroller and sends signals FSDD and FSD to the subcontroller. In addition, signal FSD is applied to the clock input of flip-flops OOUT, START, UEND, and WANTI in the receive logic. See paragraph 3-30 for a description of the use of the function strobe (FS) and the function strobe acknowledge (FSL) signals.

The MIOP raises the function strobe signal shortly after it raises any of the function indicator lines. In the case of the SIO, HIO, TIO, and TDV functions, the MIOP sends an address on the data lines. All of these signals are received by the subcontroller. If the subcontroller recognizes its address, it generates the acknowledge signal when it receives signal FSDD from the function strobe logic in the controller. The function strobe logic generates signal FSD when it receives FSR and CLSI from the subcontroller.

FSD = FSR CLSI

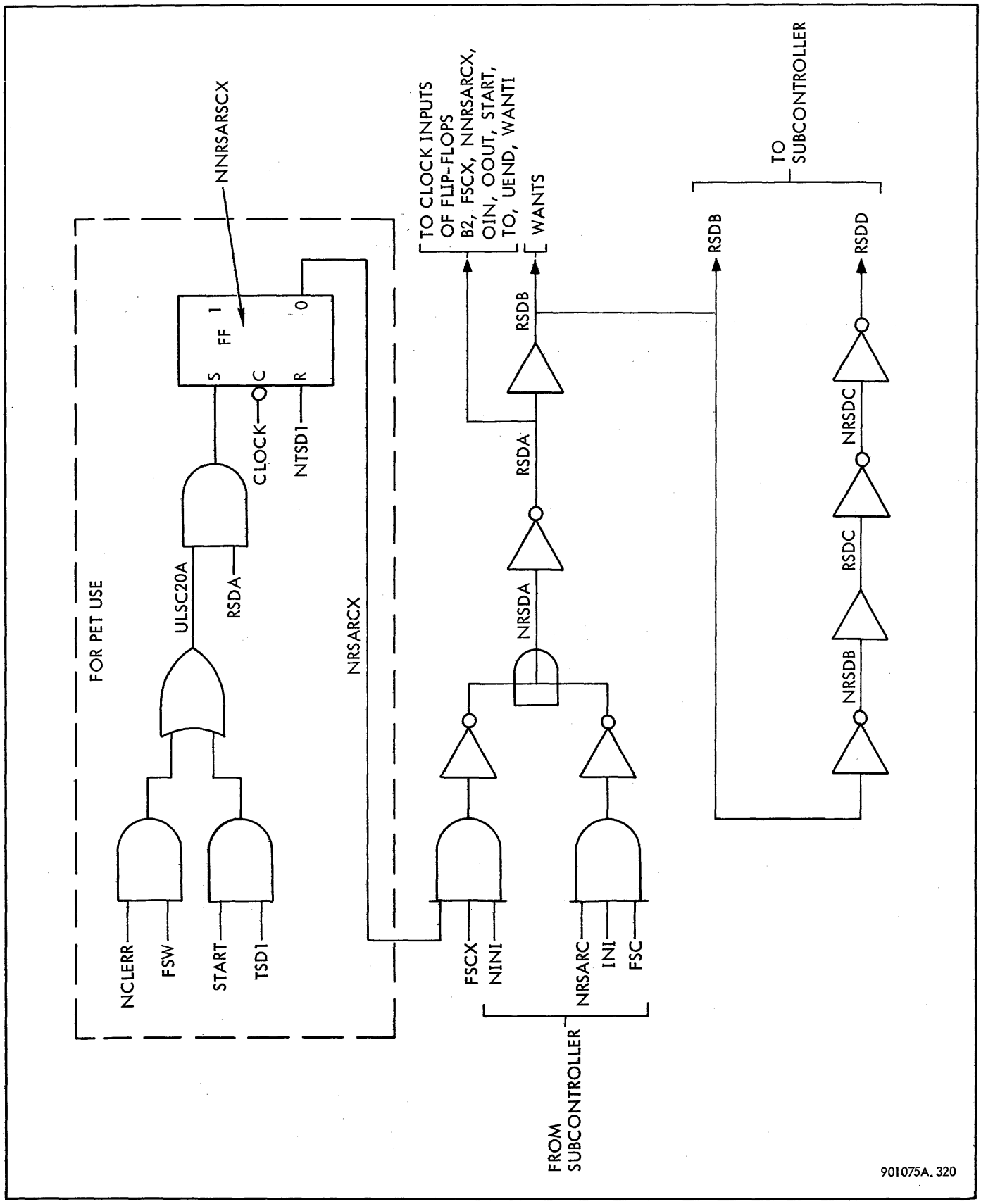


Figure 3-20. Request Strobe Logic, Logic Diagram

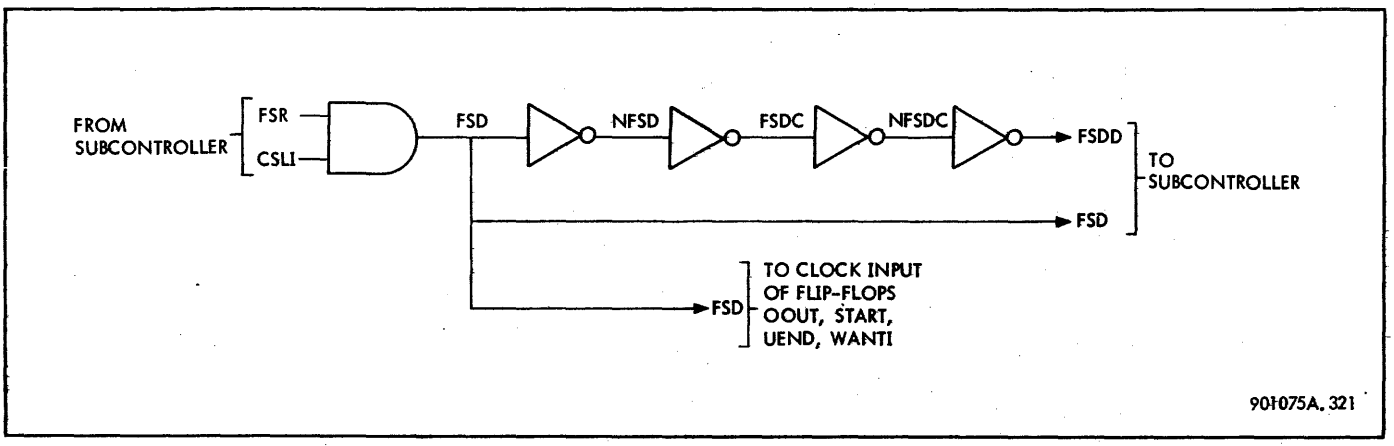


Figure 3-21. Function Strobe Logic, Logic Diagram

Signal FSR, the output of a cable receiver, goes true when FS is received, and signal CLSI is true as long as the service connect flip-flop is false. Signal FSD is converted to FSDD (FSD delayed) after passing through four inverters.

FSDD = NFSDC = FSDC = NFSD = FSD

The subcontroller uses FSD (BFSD through a buffer) to gate the status information on to the FR-lines.

3-44 State Logic. The state logic consists of the five flip-flops and the gating shown in figure 3-22. This logic controls the various states in which the controller may operate, and also provides for the generation of the end data signal and the condition code information. The flip-flops that apply to each state and the logic that generates the condition code and the end data signals are described in the following paragraphs.

a. START FLIP-FLOP. The START flip-flop, when set, defines the busy state, and when reset, it defines the ready state. The device controller enters the busy state only when an SIO is addressed to the controller and when the controller is not requesting an interrupt by way of the MIOP.

S/START = STARTS

STARTS = SIOR NWANTI NFSC

C/START = ULSB29E

ULSB29E = DCA FSD + ...

Signal NFSC is true (service connect flip-flop reset) when the controller is in the ready state. Flip-flop START is armed when the SIO is received (SIOR from the subcontroller) if signal WANTI (interrupt request) is false. Flip-flop START is set when signal FSD goes false. It goes false after the COC has acknowledged the function strobe from the MIOP. Signal DCA, generated in the subcontroller, is true if the SIO is addressed to this device controller and if the controller is not connected for service.

The COC may enter the ready state (reset START flip-flop) for the following two conditions: (a) An HIO is received and the device controller is not connected for service (NFSC). (b) The device controller is connected for service (FSC), signal end service (ESR) has been received, and flip-flop UEND is set.

R/START = ULSB29F

ULSB29F = HIOR NFSC
+ ESRFSC UEND

ESRFSC = ESR FSC

C/START = ULSB29E

ULSB29E = FSC RSDA OIN + ...

Flip-flop UEND is true during the order-in service cycle following a terminal order in which count done or IOP halt has been specified. The clock signal is supplied by the fall of FSD if flip-flop START is being reset by the HIO or by the fall of RSDA if RSDA is being reset at the conclusion of an order-in service cycle.

During the time that flip-flop START is set, signal START is sent to the subcontroller where it is used as part of the status response during an SIO, TIO, or HIO. (See table 2-2.) Signal START is used by the data line gating logic in the controller to enable RD8 (from the receive module) on to line DA0 for transmission to the MIOP. Signal NSTART is used in the logic that generates signal IORD when condition code information is being sent to the MIOP during instruction execution. During an SIO, if the controller is not requesting an interrupt (NWANTI), and if the COC is

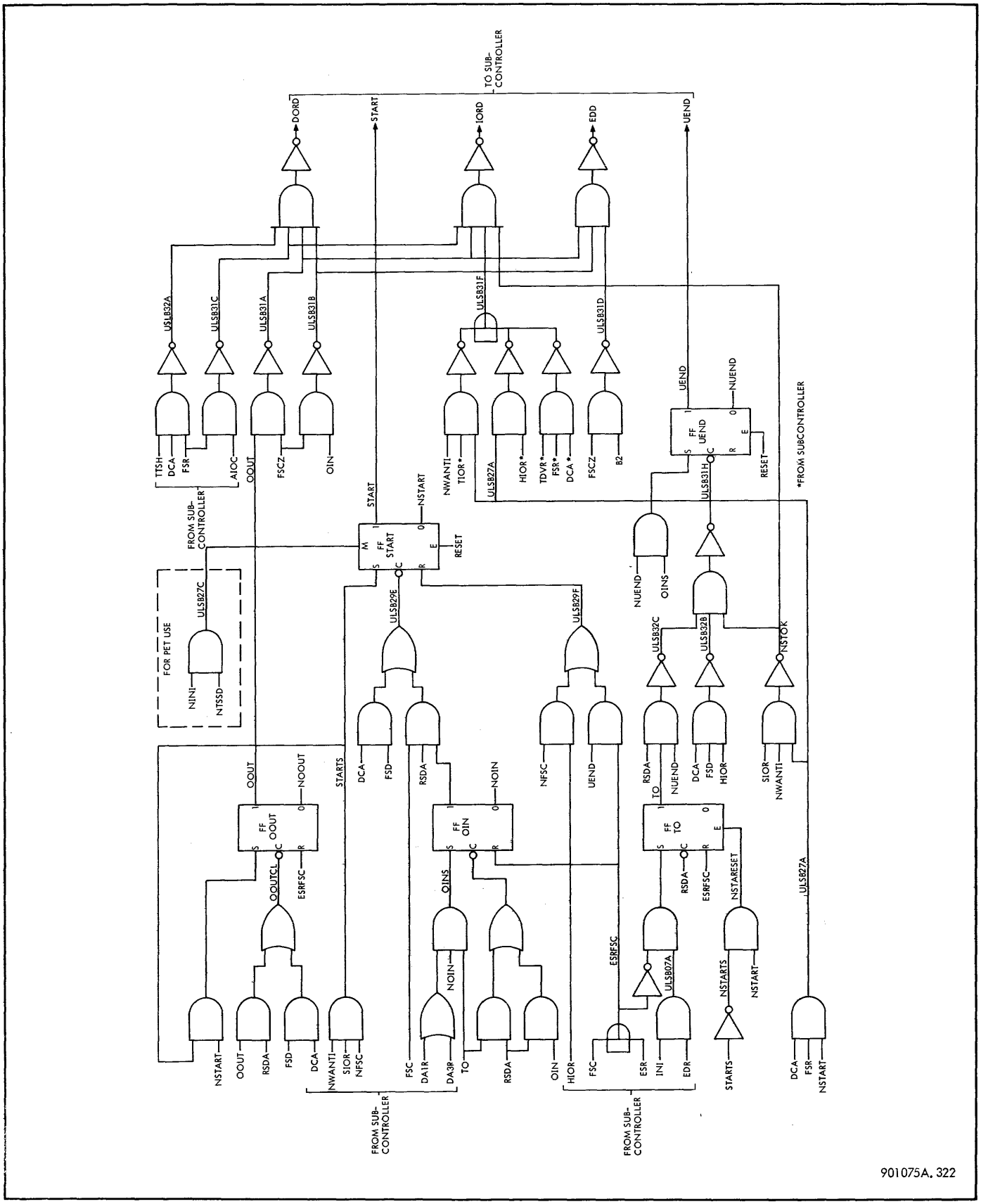


Figure 3-22. State Logic, Logic Diagram

ready (NSTART), signal IORD is driven true. Signal IORD (by way of the MIOP) controls condition code indicator CC1 to indicate that the SIO is successful.

IORD = NSTOK + ...

NSTOK = SIOR NWANTI ULSB27A

ULSB27A = DCA FSR NSTART

During an HIO, signal IORD indicates whether or not the COC is busy by sensing the state of the START flip-flop.

IORD = HIOR ULSB27A

ULSB27A = NSTART FSR DCA

Signal START and its complement are used by other circuits in the state logic, that is, on the reset input of flip-flop TO and on the set input of flip-flop OOUT.

b. ORDER-OUT FLIP-FLOP. When the order out flip-flop (OOUT) is set, it defines the order-out service cycle. This service cycle is always the first service cycle following a successful SIO. Although it serves no purpose (other than generating CSL for the service call) in the COC, it is required by the MIOP to prepare for the ensuing input/output operation. The order that the COC receives from the MIOP during the order out service cycle is ignored.

Flip-flop OOUT is set at the same time as flip-flop START and uses the same clock signal. It is set when an SIO is received if an interrupt request is not pending; that is, if signal WANTI is false.

S/OOUT = STARTS NSTART

C/OOUT = OOUTCL

OOUTCL = FSD DCA + ...

Flip-flop OOUT is reset at the conclusion of the terminal order that is always included as part of the order-out service cycle. During the order-out service cycle, the COC issues two request strobes (RSDA in the controller) which are for the order transfer and for the terminal order transfer. After the terminal order information has been placed on the data lines by the MIOP, the MIOP drives the end service line (ESR in the controller). Signal ESR is gated with signal FSC (which is true if the SIO is successful) to generate signal ESRFSC. Signal ESRFSC is applied to the reset input of the OOUT flip-flop. Dropping the request strobe that was issued for the terminal order causes flip-flop OOUT to reset.

R/OOUT = ESRFSC

ESRFSC = ESR FSC

C/OOUT = OOUTCL

OOUTCL = OOUT RSDA + ...

Signal OOUT is used to generate signals DORD, IORD, and EDD. Both signal DORD and signal IORD are driven true to specify the service cycle as an order out.

IORD = ULSB31A + ...

ULSB31A = OOUT FSCZ

DORD = ULSB31A + ...

When operating online, signal FSCZ is true when the service connect flip-flop is set. When operating offline, signal FSCZ is true when flip-flop FSCX is set. Flip-flop FSCX is used to simulate the service connect flip-flop.

Signal OOUT is applied to the end data logic (EDD) to specify end data to the MIOP. Even though the controller specifies end data, the MIOP does not terminate the service by driving the end service line true until the controller issues another request strobe for the terminal order.

EDD = ULSB31A + ...

Signal OOUT, gated with RSDA, is also used as a clock input to provide the clock signal for resetting flip-flop OOUT.

Signal NOOUT is applied to the following logic groups in the controller: service request, byte two, receive external interrupt request, and data line gating logic.

c. ORDER-IN FLIP-FLOP. When the order in flip-flop (OIN) is set, it defines the order-in service cycle and generates the service call signal. The order-in service cycle can only be entered following a terminal order in which the MIOP has specified a count done or an IOP halt. Count done is specified when the MIOP drives data line 1 (DA1R in the controller), and the IOP halt is specified when the MIOP drives data line 3 (DA3R in the controller). Flip-flop OIN is set with the fall of the request strobe (RSDA) that was issued for the terminal order. During the time that OIN is true, the operational status byte, consisting only of unusual end is placed on the data lines by way of the data line gating logic in the controller.

S/OIN = OINS

OINS = DA1R DA3R TO NOIN

C/OIN = TO RSDA + ...

Signal OIN is used to generate signals DORD and EDD. Signals DORD true and IORD false define the service cycle as order-in.

DORD = ULSB31E

USLB31E = OIN FSCZ

Signal OIN is applied to the end data logic to specify end data to the MIOP. Similar to the order-out service cycle, the controller issues one more request strobe for the terminal order, even though the terminal order specifies end data.

Flip-flop OIN is reset on the fall of the request strobe that is issued for the terminal order that follows the order-in service cycle.

R/OIN = ESRFSC

C/OIN = RSDA OIN + ...

d. TERMINAL ORDER FLIP-FLOP. When the terminal order flip-flop (TO) is true, it defines the current transmission (always from MIOP to device controller) as a terminal order. The terminal order may specify count done (data line 1) or IOP halt (data line 3). Count done applies only to a data in service cycle; however, IOP halt applies to any of the service cycles.

The MIOP holds the end service line false if the MIOP wants to send a terminal order during any service cycle in which end data is specified (either by the MIOP or the COC). This condition arms the TO flip-flop, which is set with the fall of RSDA. Signal RSDA is applied to the clock input. During the terminal order transmission, the MIOP drives the end service line. Signal ESRFSC is applied to the reset input of the terminal order flip-flop. The flip-flop resets when the request strobe that was issued for the terminal order falls.

S/TO = NESRFSC ULSB07A

ULSB07A = INI EDR

R/TO = ESRFSC

C/TO = RSDA

Signals TO and NTO are used in addition to the state logic, by the following logic groups in the controller: receive scanner, service request, internal interrupt request, byte two, and data line gating logic.

e. UNUSUAL END FLIP-FLOP. The unusual end flip-flop, UEND, is set every time that the controller enters the order-in state. Unusual end and channel end (both are true when UEND is true) are the only information conveyed by way of the operational status byte during the order-in service cycle. The COC reports unusual end and channel end because the same conditions that cause the COC to enter the order-in service cycle (either count done or IOP halt) are received from the MIOP during a terminal order. The same signal (OINS) that arms the OIN flip-flop is also used to arm the UEND flip-flop.

S/UEND = NUEND OINS

C/UEND = RSDA TO NUEND + ...

Flip-flop UEND is set with the fall of the request strobe (RSDA). Signal RSDA is issued for the terminal order following the last data-in transmission.

The UEND flip-flop remains set until it is reset by an HIO, a successful SIO, or the RESET signal. The RESET signal is applied to the erase (E) input, and causes immediate resetting of the flip-flop. The two other conditions (SIO or HIO) control the resetting of UEND by way of the clock input. Since the reset input is floating (always true), the flip-flop resets when the clock signal falls, which was true because of the SIO or HIO.

R/UEND = .

E/UEND = RESET

C/UEND = HIOR DCA FSD
+ SIOR NWANTI NSTART DCA FSR

In the case of the HIO, flip-flop UEND is reset with the fall of signal FSD, which generates the function strobe acknowledge signal during the HIO function. Signal DCA indicates address recognition during both the SIO and the HIO functions.

In the case of the SIO, flip-flop UEND is reset with the fall of signal FSR (function strobe from the MIOP) if an interrupt request is not being generated (signal NWANTI true) and if the controller is not in the busy state (signal NSTART true).

During the time that flip-flop UEND is true, signal UEND is sent to the status logic in the subcontroller, where it is sent to the MIOP as part of the status information in response to an instruction. Signal UEND also drives DA3D and DA4D true in the data line gating logic in the controller during an order-in service cycle. Signal DA3D drives data line 3 (channel end) and signal DA4D drives data line 4 (unusual end) at the MIOP interface by way of the subcontroller. Signal UEND is also applied to the reset input of flip-flop START so that the COC returns to the ready state at the conclusion of the order-in service cycle.

f. IORD AND DORD LOGIC. This logic furnishes condition code information during instruction execution

and specifies the type of service cycle required during the order-in/order-out and the data-in operations. Data-out operations are not used by the COC. Signals IORD and DORD drive the IOR and DOR lines at the MIOP interface by way of the subcontroller. Line IOR true at the MIOP/COC interface causes line NCOND2 at the MIOP/CPU interface to go true. As a result condition code light CC2 on the processor control panel on the CPU remains off. Similarly, line DOR true at the MIOP/COC interface causes line NCOND1 at the MIOP/CPU interface to go true, and the condition code light CC1 on the processor control panel remains off. (See table 2-1 for the meaning of the condition codes.)

Signal DORD is controlled in the following manner to specify the condition code. During an SIO, HIO, TIO, or TDV function that is addressed to the COC, DORD goes true when the function strobe (FSR) is received from the MIOP.

$$DORD = TTSH\ DCA\ FSR + AIOC\ FSR + \ldots$$

During an AIO function, signals AIOC and FSR cause DORD to go true. Address recognition (DCA) does not apply during an AIO function.

Signal IORD is controlled in the following manner to specify the condition code. During a TDV function addressed to the COC, IORD goes true when the function strobe is received from the MIOP.

$$IORD = TDVR\ DCA\ FSR + HIOR\ NSTART\ DCA\ FSR + TIOR\ NWANTI\ NSTART\ DCA\ FSR + AIOC\ FSR + \ldots$$

During an HIO function addressed to the COC, IORD goes true when the function strobe is received from the MIOP if flip-flop START is false. This indicates that the COC was not busy when the HIO was received. During a TIO function addressed to the COC, IORD goes true when the function strobe is received from the MIOP if both flip-flop WANTI and flip-flop START are false. This condition indicates that the COC was not busy and was not requesting an interrupt when the TIO was received and can therefore accept an SIO. During an AIO function, IORD goes true when AIOC and FSR are received; that is, IORD is driven true unconditionally during an AIO as it is during a TDV addressed to the COC.

Signals IORD and DORD are controlled in the following manner to specify the type of service cycle required by the COC. (See paragraphs 2-14 through 2-17 for information pertaining to the use of the service cycles.) An order-out service cycle is specified when both IORD and DORD are true. Both are driven true by signal OOUT if the COC is connected for service (FSCZ true). An order-in service cycle is specified when only DORD is true. Signal DORD is driven true by signal OIN if the COC is connected for service. A data-in service cycle is specified when both IORD and DORD are false (signals OOUT and OIN are absent).

$$DORD = FSCZ\ (OOUT + OIN) + \ldots.$$

$$IORD = FSCZ\ OOUT + \ldots$$

g. END DATA LOGIC. Signal EDD is generated by the state logic and is sent to the subcontroller where it controls the end data line (ED) to the MIOP. This line may also be controlled by the MIOP. Since the COC is a two-byte device, EDD is driven true by signal byte two (B2) during a data-in service cycle. Signal EDD is also driven true by signals OIN and OOUT during the order-in and order-out service cycles.

$$EDD = FSCZ\ (B2 + OIN + OOUT)$$

3-45 Function Strobe Logic (DIO Interface). The function strobe logic consists of flip-flops FX, FY, FZ, a cable driver and receiver at the DIO interface, and the additional logic shown in figure 3-23. This logic is used with read/write direct instructions at the DIO interface in a similar manner to the function strobe logic associated MIOP interface that is used with the five function indicators at the MIOP interface. The function strobe logic (DIO interface) is used during both input and output operations.

In response to function strobe RFS during a read direct or a write direct instruction addressed to the COC, the controller generates function strobe acknowledge RFSA, which is returned to the CPU. In addition, signals FSL0 through FSL7 and EWDD are generated. Signals FSL0 through FSL7 are sent to LIU0 through LIU7, respectively. Signal EWDD is sent to the send logic in the controller where it is used to gate the character during a send operation.

Other input signals to the function strobe logic (in addition to RFS received at the DIO interface) are received from the subcontroller and from the send logic in the controller. Logic is also incorporated for off-line operation with the PET.

If a write direct or a read direct instruction with the proper mode is addressed to the COC, the send logic generates signal SCOCE. (See figure 3-24 for a timing diagram of the function strobe and of the associated signals.) Shortly after raising the address lines, the CPU raises RFS (RFSR). Signal RFSR gated with signal INI (from the subcontroller) causes signal FS1 to go true. FX is set immediately since FS1 is applied to the mark (M) input of flip-flop FX. At the same

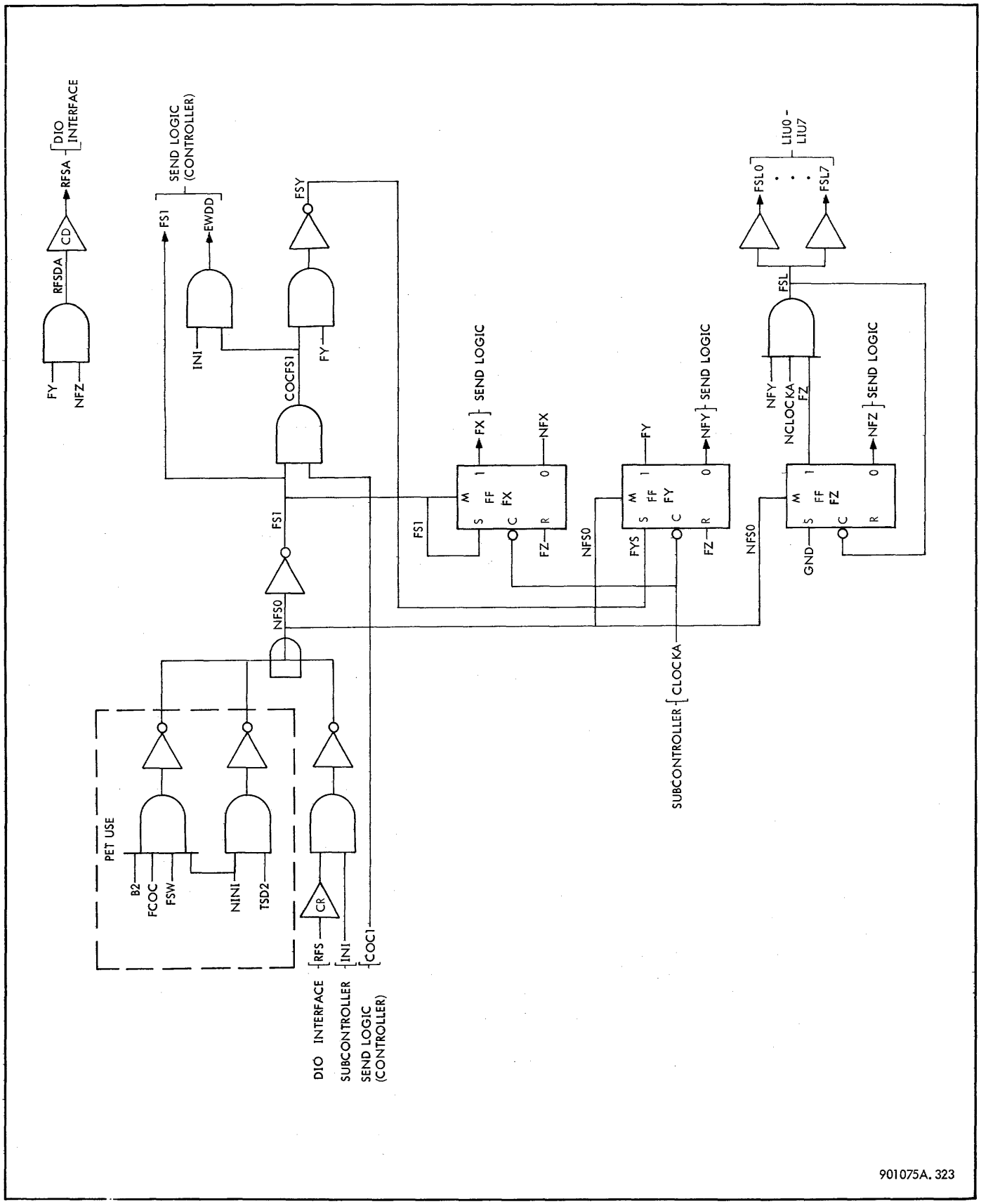


Figure 3-23. Function Strobe Logic, Logic Diagram (DIO Interface)

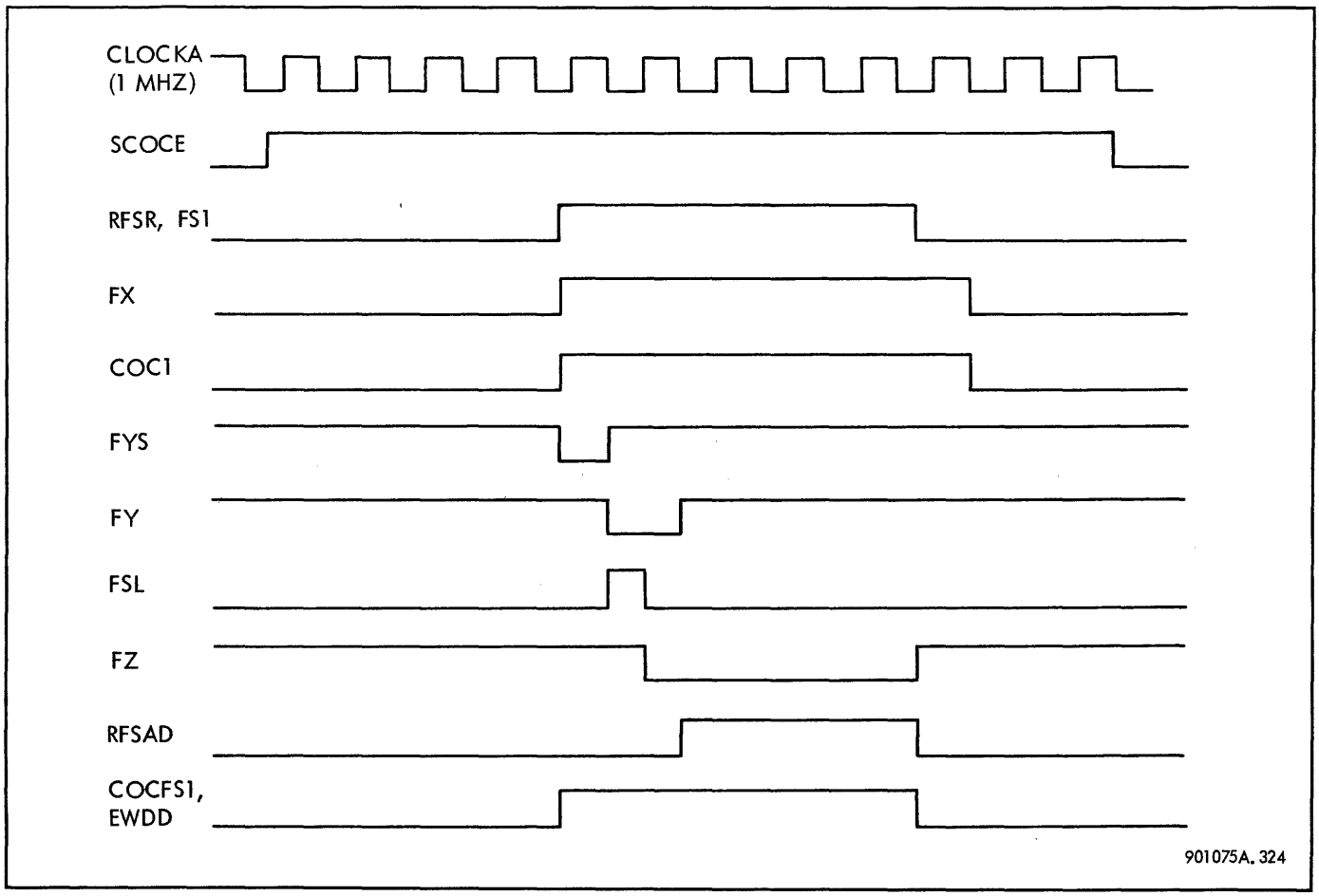


Figure 3-24. Function Strobe Timing Diagram (DIO Interface)

time, signal EWDD is generated and is sent to the send logic in the controller.

M/FX = FS1

FS1 = RFSR INI

EWDD = COCFS1 INI

Signal FX is sent to the send logic in the controller where it is gated with SCOCE to produce signal COC1. Signal COC1 signifies that the address has been recognized along with the read direct or the write direct instruction.

COC1 = SCOCE FX INI

Signal NFS0, which is normally true until RFSR is received, is applied to the M input of flip-flops FY and FZ, causing them to assume the set state normally.

M/FY = NFS0

M/FZ = NFS0

When FS1 goes true, FYS goes false. This allows flip-flop FY to reset at the next CLOCKA time, since the true signals were removed from the M input (NFS0) and from the S input (FYS), and since the signal applied to the R input (FZ) is true.

S/FY = FYS

R/FY = FZ

C/FY = CLOCKA

When FY resets, signal FYS again goes true, thus arming the flip-flop for the next CLOCKA time.

Signal FSL (applied to the clock input of flip-flop FZ) goes high when flip-flop FY resets.

FSL = FZ NFY NCLOCKA

At the fall of NCLOCKA, flip-flop FZ resets, which causes FSL to drop. When CLOCKA falls following

the reset of flip-flop FZ, flip-flop FY sets, enabling the function strobe acknowledge RFSAD, which is sent to the CPU as signal RFSA through a cable driver.

RFSAD = FY NFZ

Signal RFSAD remains high until the CPU drops RFS (RFSR). When RFSR drops, flip-flop FZ sets and flip-flop FY remains set.

M/FY = NFS0

M/FZ = NFS0

Signal FZ is applied to the reset input of flip-flop FX, causing FX to reset with the fall of signal CLOCKA.

R/FX = FZ

C/FX = CLOCKA

When signal FX goes false, signal COC1 (generated in the controller) goes false and concludes the function strobe cycle.

If the function strobe is raised and if there is no address recognition (SCOCE low), flip-flop FX sets and then resets at the conclusion of the function strobe. Since COC1 stays low, no attempt is made to raise the function strobe acknowledge signal.

3-46 SEND LOGIC. Each block in figure 3-3 that forms part of the send logic is described in the following paragraphs.

3-47 Send Scanner and Decoding Logic. This logic (figure 3-25) consists of six send scanner flip-flops, SS1 through SS6, that function as a modulo 64 counter, two control flip-flops (FC OC and FSSS63), input gating that controls the counter, and output gating that sends the LIU number and the line number to the LIU's and to the CPU by way of the DIO interface. (See figure 3-26 for timing.)

The function of the send scanner is to scan sequentially all 64 send modules starting with number 00 in LIU0 and ending with number 63 in LIU7. When the counter reaches the address of module 63, it starts again with address 00. The addresses of the send modules in the LIU's are the same as the lines to which they feed information.

The scanner operates in the normal mode or in the high speed mode, depending on whether the high speed option is installed. In the normal mode, the scanner is searching for a send module that has completed transmission of a character and that is ready to request service. A send module requests service by dropping signal NSSRM (NSSRL0 through NSSRL7 in the LIU's), which is applied to the send scanner enable logic. Signals NSSRL0 through NSSRL7 originate on the eight LIU's and are normally high until service is requested. For example, signal NSSRL0 designates the send service request line for the eight send modules in LIU0. When there is a request for service, the send module drops its signal causing NSSR to fall.

NSSR = NSSRL0 + NSSRL1 + NSSRL2 + NSSRL3 + NSSRL4 + NSSRL5 + NSSRL6 + NSSRL7

When signal NSSRM from any send module in any LIU goes low, the scanner is stopped. The send module can drop this signal only when it is receiving its address from the scanner. Therefore, when the scanner is stopped, its count is equal to the line number and to the send module number that is requesting service. Shortly after the scanner is stopped, the send external interrupt logic raises the interrupt line (SINT).

The signal that enables the scanner to run, SSRUN, is applied to both the set and the reset input terminals of flip-flop SS1, which is the first flip-flop in the counter chain. The counter is a simple ripple counter; each flip-flop in the counter (except SS6) applies its output signal to the clock input of the following flip-flop.

S/SS1 = NSS1 SSRUN

R/SS1 = SSRUN

C/SS1 = CLOCK

M/SS1 = SSS63

E/SS1 = GND

S/SS2 = NSS2

R/SS2 = .

C/SS2 = SS1

M/SS2 = SSS63

E/SS2 = GND

. .

. .

. .

S/SS6 = NSS6

R/SS6 = .

C/SS6 = SS5

M/SS6 = SSS63

E/SS6 = GND

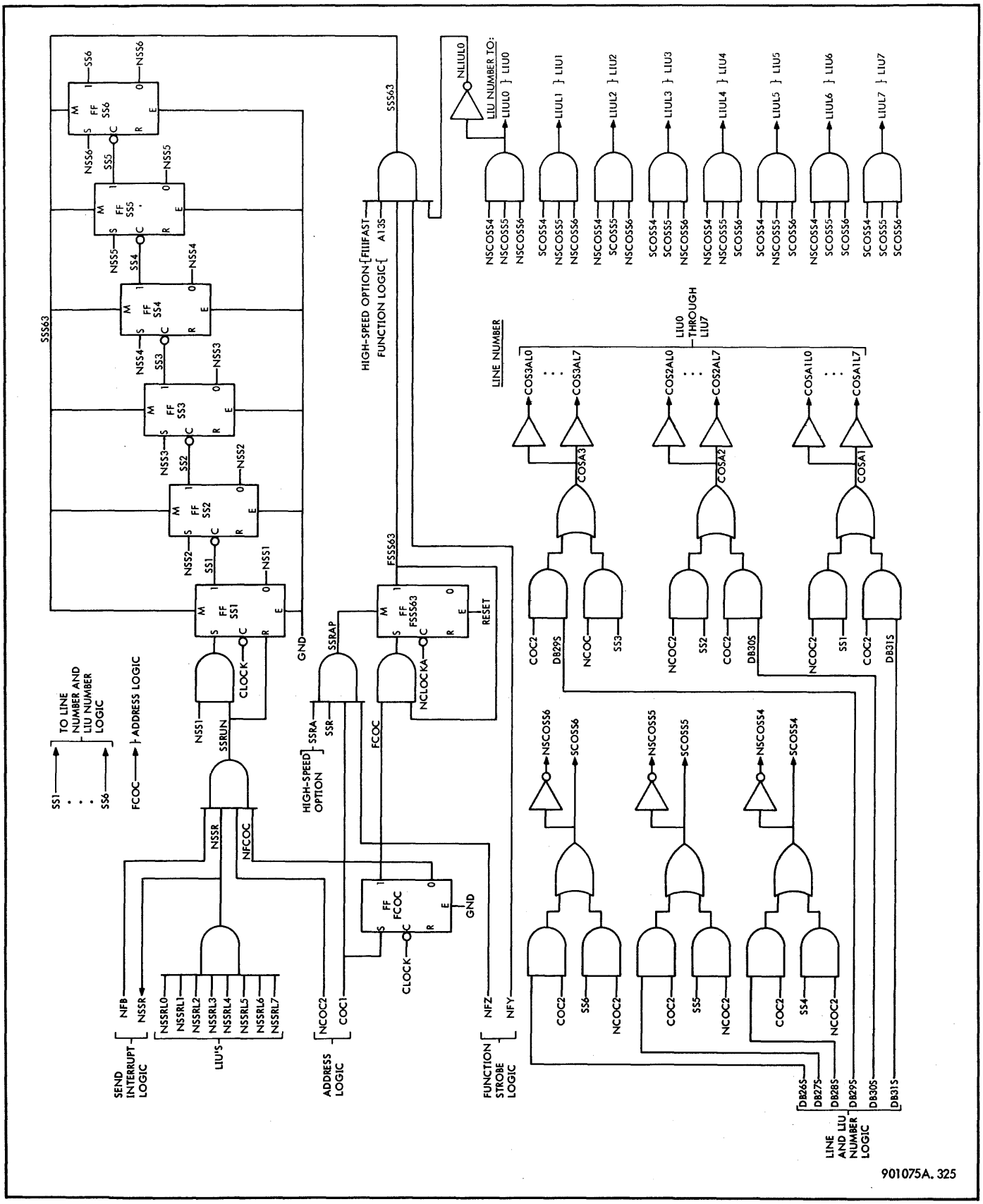


Figure 3-25. Send Scanner and Decoding Logic, Logic Diagram

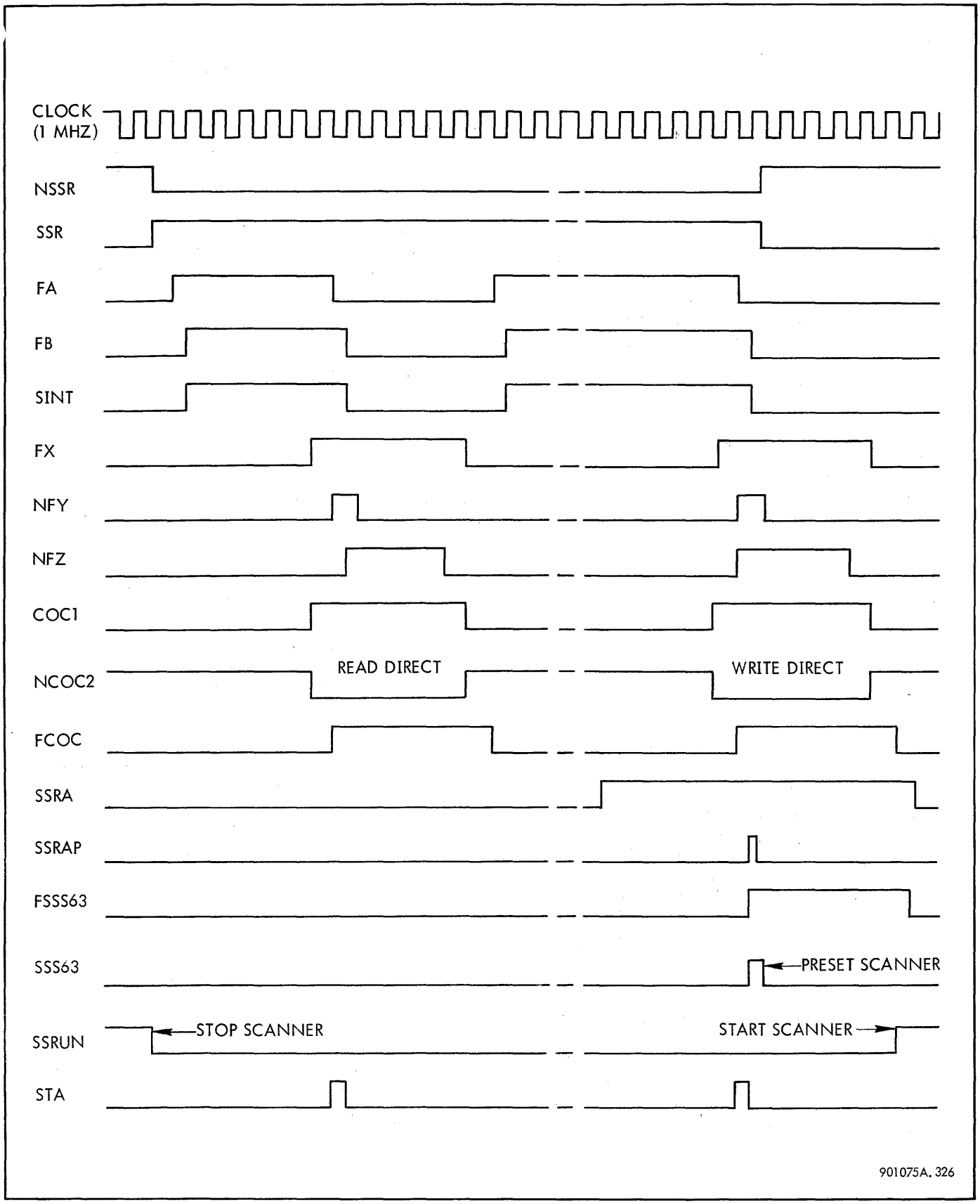


Figure 3-26. Send Logic, Timing Diagram

The logic equation for enable signal SSRUN is:

SSRUN = NSSR NCOC2 NFCOC NFP

Signal NCOC2 is true when a read/write direct instruction has not been directed to this COC. During this time, the send scanner is free to run in search of a service request.

NCOC2 = NCOC1

If a read/write direct instruction is executed, signal NCOC2 is dropped which forces the send scanner to stop. The scanner stops even if the instruction is related to a receive function. This is done because the lines that define the LIU number (SCOSS4 through SCOSS6), the send module, and the line number (COSA1 through COSA3) are also used to address the receive modules when performing receive functions. Thus, the send scanner is stopped in order not to bypass any send service requests.

Signal NFCOC in the SSRUN equation is the zero output flip-flop FCOC. This flip-flop is used to synchronize the starting of the scanner after completion of a read/write direct instruction.

S/FCOC = COC1

R/FCOC = .

C/FCOC = CLOCK

M/FCOC = GND

E/FCOC = GND

Signal COC1 is raised when the CPU acknowledges the send service request (send interrupt SINT) by issuing a read direct instruction with the output response function. The clock following the rise of COC1 sets flip-flop FCOC, causing NFCOC to drop . This further inhibits scanner enable signal SSRUN. At the completion of the read/write direct instruction, signal COC1 falls allowing flip-flop FCOC to reset with the fall of the next clock. The send service request line NSSR is still false; therefore, the scanner is unaffected by FCOC at this time. Signal COC1 is raised a second time when the CPU executes a write direct instruction, following the read direct instruction, to instruct the send module and to clear the send service request line. Flip-flop FCOC is also set and reset a second time, since it is controlled by signal COC1. During the second time that FCOC is true, the LIU that caused the scanner to stop raises its respective service request line and causes signal NSSR to go true. Signal SSRUN, therefore, goes true when flip-flop FCOC resets and enables the scanner to run.

Signal NFB, in the SSRUN equation, is generated by the send interrupt logic. This signal prevents the scanner from running while the send interrupt line is activated.

When the high speed option is installed in the COC, the normal sequencing of the scanner is modified. Each time that a send module requests service which is not in LIU0, the scanner is preset to address 63. This represents the last send module in LIU7. If the scanner count does not represent the address of a send module in LIU0, all addresses from the present address to 63 are skipped. No presetting takes place if the scanner is addressing a send module in LIU0. Thus, LIU0 is the only LIU that can accommodate the high speed option.

Flip-flop send scanner set to 63 (FSSS63) is set by signal SSRAP every time that a write direct instruction is executed by the CPU in response to a send service request (interrupt).

S/FSSS63 = FCOC FSSS63

M/FSSS63 = SSRAP

R/FSSS63 = .

C/FSSS63 = NCLOCKA

Flip-flop FSSS63 is initially set when signal SSRAP, applied to the M input goes true. It remains true until the fall of NCLOCKA following the fall of FCOC at the S input.

SSRAP = SSR SSRA NFZ COC1

Signal SSRA is the send service request acknowledge signal, which is raised during the write direct instruction following the rise of SSR. Since signal SSRA is received from the high speed option, it is generated only if the option is present.

SSRA = N(NDB29S SS3 + DB29S NSS3)
N(NSCOSS4 SS4 + DB28S NSS4)
N(NSCOSS5 SS5 + DB27S NSS5)
N(NSCOSS6 SS6 + DB26S NSS6)
N(NDB31S SS1 + DB31S NSS1)
N(DB30S NSS2 + NDB30S SS2)
N(SSRAGO SSRASTOP)

SSRAGO = N(NA14S A15S)

SSRASTOP = N(A14S NA15S)

Signal SSRAGO enables SSRA, and signal SSRASTOP disables SSRA. Signal SSRAGO is true when the function encoded in the write direct instruction is transmit data or transmit long space. Signal SSRASTOP is true when the function encoded in the write direct instruction is a stop transmit.

Signal SSR is the output of an inverter that is controlled by signal NSSR at its input. Signals NFZ and COC1

are raised during both the read direct and the write direct instructions. It is not until the write direct instruction, however, that NFZ and COC1 become useful. At this time, signal SSRA is raised. This completes the requirements of signal SSRAP.

Signal SSS63 is applied to the M inputs of the six scanner flip-flops to preset them to address 63.

SSS63 = FSSS63 A13S NFY FIIIFAST NLIU0

Signal A13S, received from the function logic, is high during the time that the send circuits are activated. Signal A13S implies that one of the transmit functions is encoded in the instruction. Signal NFY, received from the function strobe logic, is high for one CLOCKA time, which limits the amount of time that SSS63 is true. Signal FIIIFAST, generated by the high speed option, is continuously true when the option is installed. Signal NLIU0 is the output of an inverter that is controlled by signal LIU0. Normally, signal NLIU0 is not generated (not connected); however, when the high speed option is installed, a connection is made that causes NLIU0 to be true whenever the send scanner is addressing any LIU except LIU0. Therefore, if the send module that is being serviced is in LIU0, no presetting of the scanner takes place.

There are two groups of gates. One group is used to gate three signals (COSA1 through COSA3) that represent the line number to the control logic in all LIU's. The control logic in each LIU decodes the three signals in order to select one of the eight send-receive positions in its LIU. Each of the three signals are distributed to the eight LIU's through separate buffer amplifiers, the output of which carries the designation (L0 through L7) of the LIU that it serves. For example, signals COSA1L2 through COSA3L2 are applied only to LIU2.

The other group of gates is used to decode the LIU number. Eight signals are generated, LIUL0 through LIUL7. Each gate is wired only to its respective LIU (LIU0 through LIU7). Even though the line number (send-receive position) is sent to all LIU's, only the position in the LIU specified by the LIU number responds.

Both groups of gates can receive inputs from the send scanner when the COC is not executing a read/write instruction (NCOC2) or from the DIO interface, by way of the line number and the LIU number logic, when the COC is executing a read/write instruction (COC2). When the LIU and the line number are received from the CPU via the DIO interface, they may apply to either a send or receive module. However, when the send scanner is supplying the LIU and the line number to the LIU (send scanner running), they apply only to a send module.

Decoding and gating the send scanner outputs are similar to decoding and gating the receive scanner outputs. That is, the three LSB's of the scanner define the line number, and the three MSB's define the LIU number.

The LIU number is generated by creating signals SCOSS4, SCOSS5, and SCOSS6 and their complements. The complement of each signal is generated by the use of three inverters (NSCOSS4 through NSCOSS6).

SCOSS4 = SS4 NCOC2 + DB28S COC2

SCOSS5 = SS5 NCOC2 + DB27S COC2

SCOSS6 = SS6 NCOC2 + DB26S COC2

Signals SCOSS4, SCOSS5, and SCOSS6 and their complements are decoded to generate signals LIUL0 through LIUL7.

LIUL0 = NSCOSS4 NSCOSS5 NSCOSS6

LIUL1 = SCOSS4 NSCOSS5 NSCOSS6

LIUL2 = NSCOSS4 SCOSS5 NSCOSS6

. .

. .

. .

LIUL7 = SCOSS4 SCOSS5 SCOSS6

The line number (module number) is generated by creating signals COSA1, COSA2, and COSA3.

COSA1 = SS1 NCOC2 + DB31S COC2

COSA2 = SS2 NCOC2 + DB30S COC2

COSA3 = SS3 NCOC2 + DB29S COC2

These signals are applied simultaneously to all LIU's through buffer amplifiers. The outputs of the buffer amplifiers are COSA1L0 through COSA1L7 (driven by COSA1), COSA2L0 through COSA2L7 (driven by COSA2), and COSA3L0 through COSA3L7 (driven by COSA3).

3-48 LIU and Line Number Logic. This logic consists of the cable receivers-drivers and of the gating shown in figure 3-27. During execution of a read/write direct instruction, this logic receives the LIU number (encoded on lines DB26 through DB28 and the line number (encoded on lines DB29 through DB31). After passing through cable receivers at the DIO interface, the input signals become DB26R through DB31R and are applied to gating that generates signals DB26S through DB31S. These signals are then decoded by the same logic that decodes and distributes the LIU and the line numbers that are generated by the send scanner.

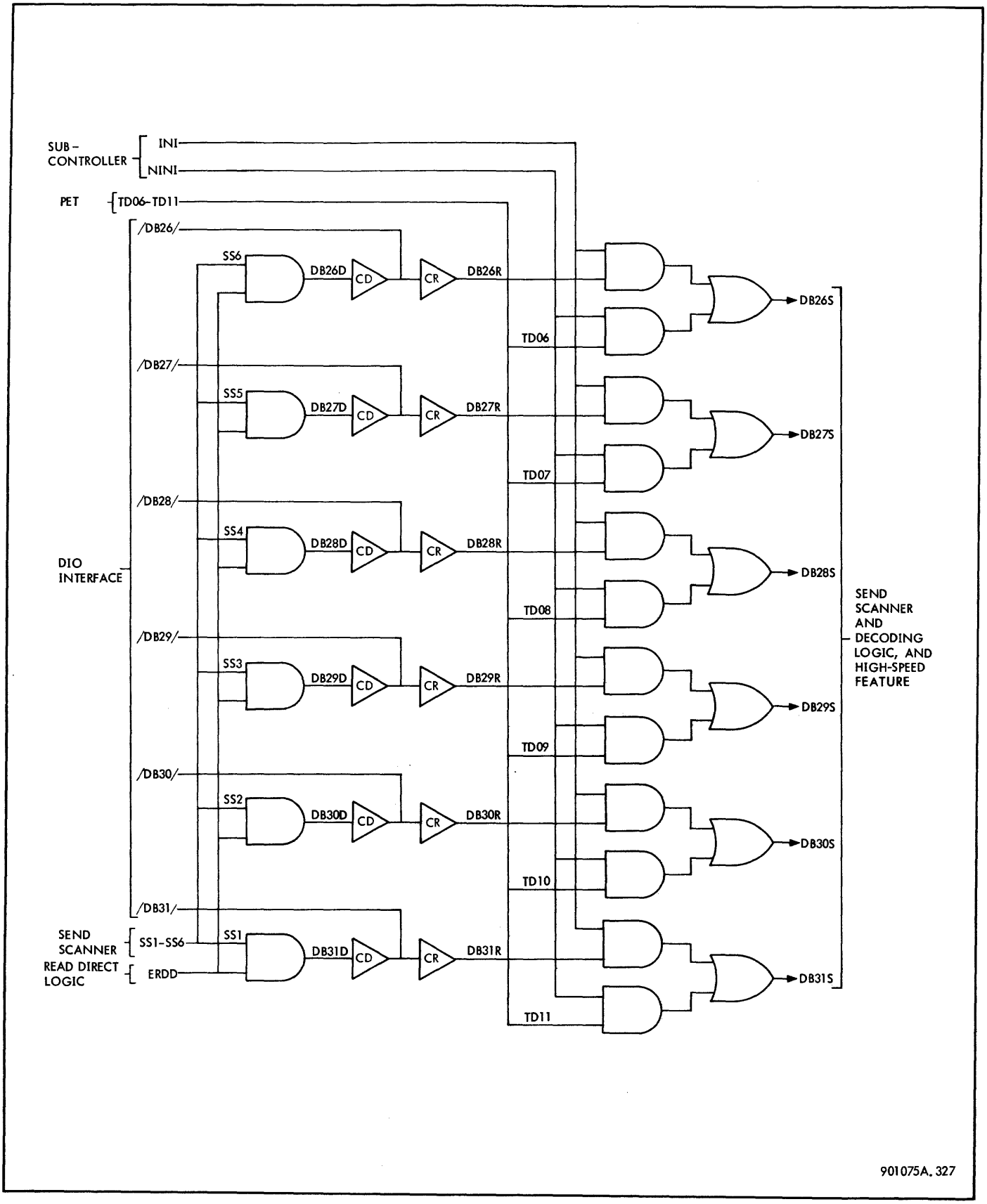


Figure 3-27. LIU and Line Number Logic, Logic Diagram

DB26S = DB26R INI + TD06 NINI
⋮ ⋮ ⋮
DB31S = DB31R INI + TD11 NINI

When used offline with the PET (signal NINI true), signals TD06 through TD11 are supplied by the PET in lieu of the LIU and the line number supplied by the CPU.

This logic also consists of the required cable drivers and the gating to send the LIU and the line number specified by the send scanner to the CPU. This occurs during a read direct instruction which is executed by the CPU to determine which send module is requesting service via a send interrupt SINT. A send module requests service when it has completed a previous transmission and is ready to be instructed by the CPU.

The send scanner signals, SS1 through SS6, are gated with signal ERDD to generate signals DB26D through DB31D which are applied to cable drivers at the DIO interface. Signal ERDD is true only during a read direct instruction.

DB26D = ERDD SS6
⋮ ⋮
DB31D = ERDD SS1

3-49 Character Transmitter Logic. This logic consists of the cable receivers and the gating shown in figure 3-28. This logic gates the character received during a write direct instruction to the send module in the LIU specified by the LIU and the line number logic. During offline operation using the PET, signals TC04 through TC11 can be generated by switches on the PET to simulate the character, or the character being received by the receive module in the LIU can be turned around and sent to the send module by way of the character transmitter logic. Signal ECD, generated by one position of a PET switch, gates the simulated character, and signal EDTA, generated by the other position of the switch, gates the character received to the send module.

When the CPU is the source of the character being transmitted, signals DB16R through DB23R (output of the receivers) are gated by signal EWDD to generate signals SDB1 through SDB8. Signal DB16R is the MSB, and signal DB23R is the LSB of the character. Signal EWDD, generated by the function strobe logic, is true when the function strobe (RFS) is received at the DIO interface.

SDB1 = DB23R EWDD + RD1 EDTA + TC11 ECD
⋮ ⋮ ⋮ ⋮
SDB8 = DB16R EWDD + RD8 EDTA + TC04 ECD

Each bit of the character, SDB1 through SDB8, is distributed to the eight LIU's through a separate buffer amplifier. For example, the LSB of the character (SDB1) is applied to eight buffers. The outputs of these buffers are SDL1L0 through SDL1L7. Signal SDL1L0 is applied to LIU0, signal SDL1L1 is applied to LIU1, and so forth. The same is true for the other seven bits of the character.

SDL1L0 = SDB1
. .
. .
. .
SDL1L7 = SDB1
. .
. .
. .
SDL8L0 = SDB8
. .
⋮ ⋮
SDL8L8 = SDB8

3-50 Address and Mode Logic. The address and the mode logic consists of a switch comparator module and of gating shown in figure 3-29. The switch comparator module (LT26) is identical to the one used at the MIOP interface. This module compares the COC address encoded on lines A08 through A11 during a read direct or a write direct instruction. Signal SCOCE is true, if, at that time, the mode number encoded on lines A00 through A03 is an X'3' (A00 and A01 false, A02 and A03 true), and if A04 through A07 are false. The above information is encoded in the effective address field of the read/write direct instruction as shown in figure 2-2. The logic equations for SCOCE follow:

SCOCE = NA00R NA01R A02R A03R Mode
NA04R NA05R NA06R NA07R
(A08R NS3-1 + NA08R S3-1)
(A09R NS2-1 + NA09R S2-1)
(A10R NS4-1 + NA10R S4-1)
(A11R NS1-1 + NA11R S1-1) } Address

Signal SCOCE is gated with signals INI and FX to generate COC1. Signal INI is true during online operation, and FX is true after the COC has received the function strobe (via the DIO interface) from the CPU. (See figures 3-24 and 3-26 for timing diagrams.)

COC1 = SCOCE INI FX + ...

Logic is also incorporated by using the PET to generate signal COC1 when operating offline.

Signal COC1 is applied to eight buffers for purposes of generating signals COCL0 through COCL7, which are distributed to LIU0 through LIU7, respectively. These signals appear as COCL in the LIU control logic of each LIU. Signal COCL is gated with the LIU number (LIUL) in each LIU to generate an enabling signal that permits decoding of the function encoded in the instruction. The enabling signal is also applied to the interface module in the addressed LIU so that the addressed send

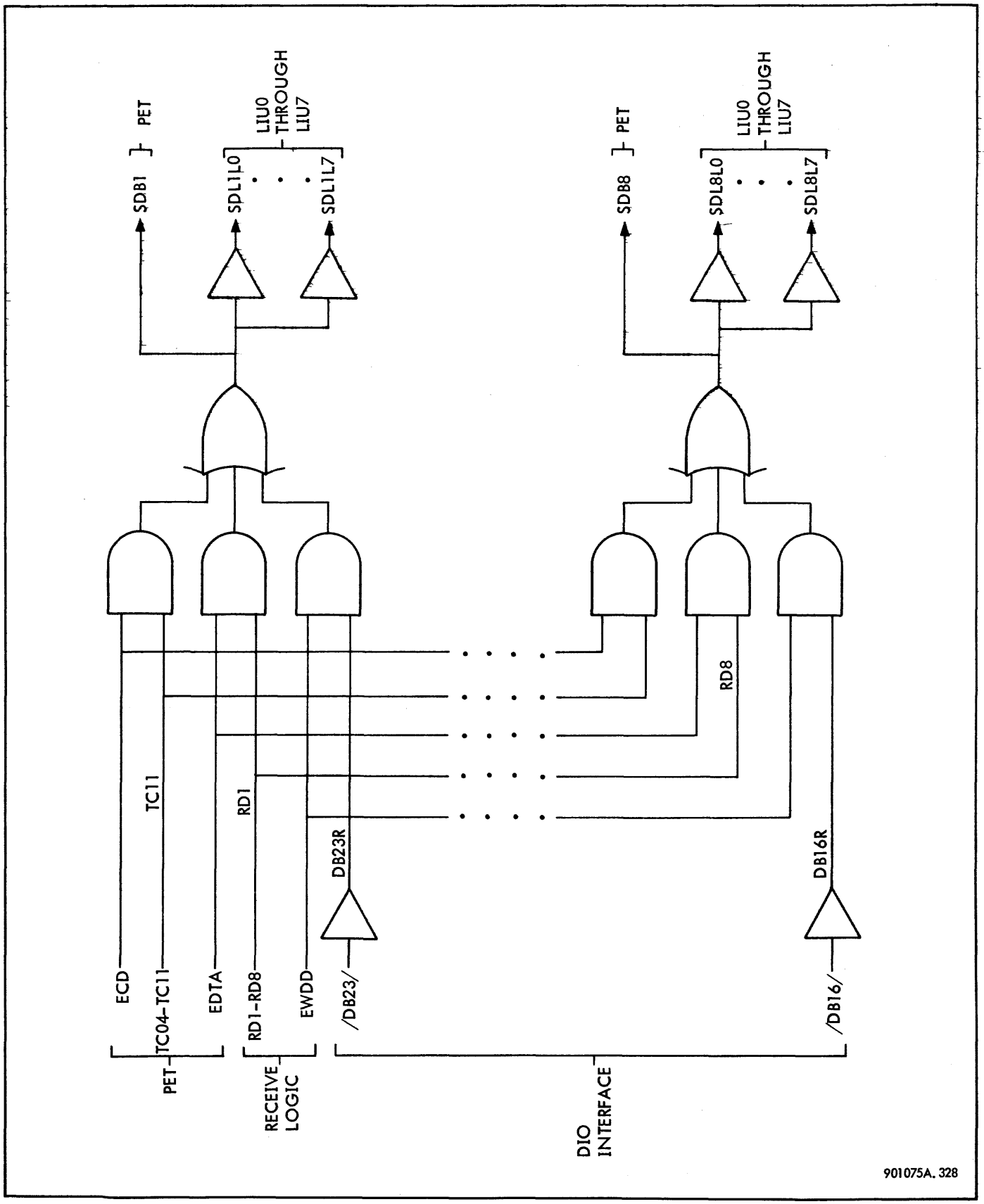


Figure 3-28. Character Transmitter Logic, Logic Diagram

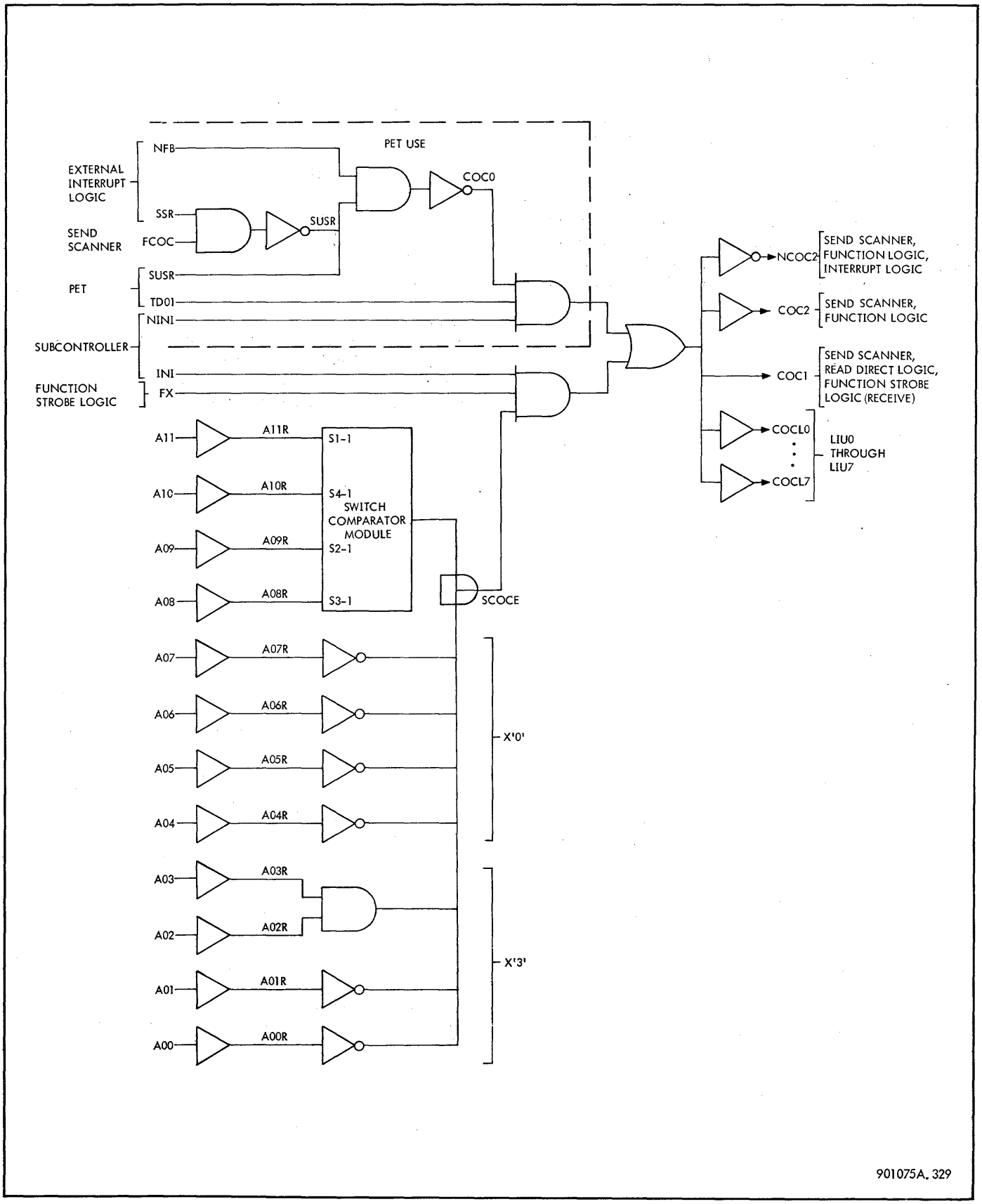


Figure 3-29. Address and Mode Logic, Logic Diagram

or receive module in that LIU can generate status information. The status information is returned to the CPU on the condition code lines. This procedure applies to both send and receive operations.

Signal COC1 is used by the send scanner and by the read direct logic in the send logic section and by the function strobe logic in the receive logic section.

Signal COC1 is also applied to a buffer amplifier to generate signal COC2 and is applied to an inverter to generate signal NCOC2. Both of these are used by the send scanner and the function logic. In addition, NCOC2 is used by the external interrupt logic associated with the send circuits.

COCL0 = COC1
⋮ ⋮
COCL7 = COC1

3-51 Function Logic. The function logic consists of the gating shown in figure 3-30. The primary purpose of this logic is to receive the function encoded on lines A12 through A15 at the DIO interface during a read or write direct instruction, and to distribute it, undecoded, to the LIU's. Decoding the function is accomplished by the control logic only in the LIU selected by the address logic.

Signals A12 through A15 at the interface become A12R through A15R at the output of the cable receivers. When operating online, these signals are gated with signal INI from the subcontroller to generate signals A12S through A15S.

A12S = INI A12R + NINI TD02
⋮ ⋮ ⋮
A15S = INI A15R + NINI TD05

When operating offline, signals TD02 through TD05 from the PET are gated with signal NINI from the subcontroller to generate signals A12S through A15S, respectively.

Signals A12S through A15S are applied to inverters to generate signals NA12S through NA15S. Signals A12S through A15S and their complements are used by the read direct logic, and by the send scanner when the high speed option is installed.

Signals A12S, A14S, and A15S each control a set of buffer amplifiers. Signal A12S generates signals RL12L0 through RL12L7; signal R14S generates signals RL14L0 through RL14L7; and signal A15S generates signals RL15L0 through RL15L7. One signal from each of the aforementioned groups is applied to the appropriate LIU.

RL12L0 = A12S
⋮ ⋮
RL12L7 = A12S

RL14L0 = A14S
⋮ ⋮
RL14L7 = A14S

RL15L0 = A15S
⋮ ⋮
RL15L7 = A15S

Signal A13S is gated with signal COC2 received from the address and mode logic to generate signal SCOSS. Signal COC2 implies that a read or a write direct instruction with the proper mode and the address was directed to this COC and that the function strobe was received. If signal SCOSS is true (A13S true), the instruction applies to a send operation. If signal SCOSS is false (A13S false), the instruction applies to a receive operation. Signal SCOSS is also generated by NCOC2, which is true when a read/write instruction is not being executed by the COC.

Signal SCOSS is generated by NCOC2 when the COC is not executing a read or a write direct instruction so that the send module address, generated by the send scanner, can be decoded by the LIU control logic. This permits the send module to make a service call (resulting in an interrupt) after it has transmitted a character or a long space. Upon receipt of the interrupt, the CPU can take appropriate action; that is, it can send another character, or long space, or can cause the send module to stop making service requests. The service request made by the send module causes the send external interrupt logic in the controller to generate the interrupt. This requirement (service call and interrupt) does not apply to a receive module, since the receipt of a new character from the line causes a service call to be generated and sent to the MIOP.

SCOSS = A13S COC2 + NCOC2

Signal SCOSS is distributed to the eight LIU's through eight buffer amplifiers that generate signals SCOSSL0 through SCOSSL7. Each of these signals is applied to its respective LIU where it is used to enable the decoding of the send or the receive module address (signals COSA1 through COSA3).

SCOSSL0 = SCOSS
⋮ ⋮
SCOSSL7 = SCOSS

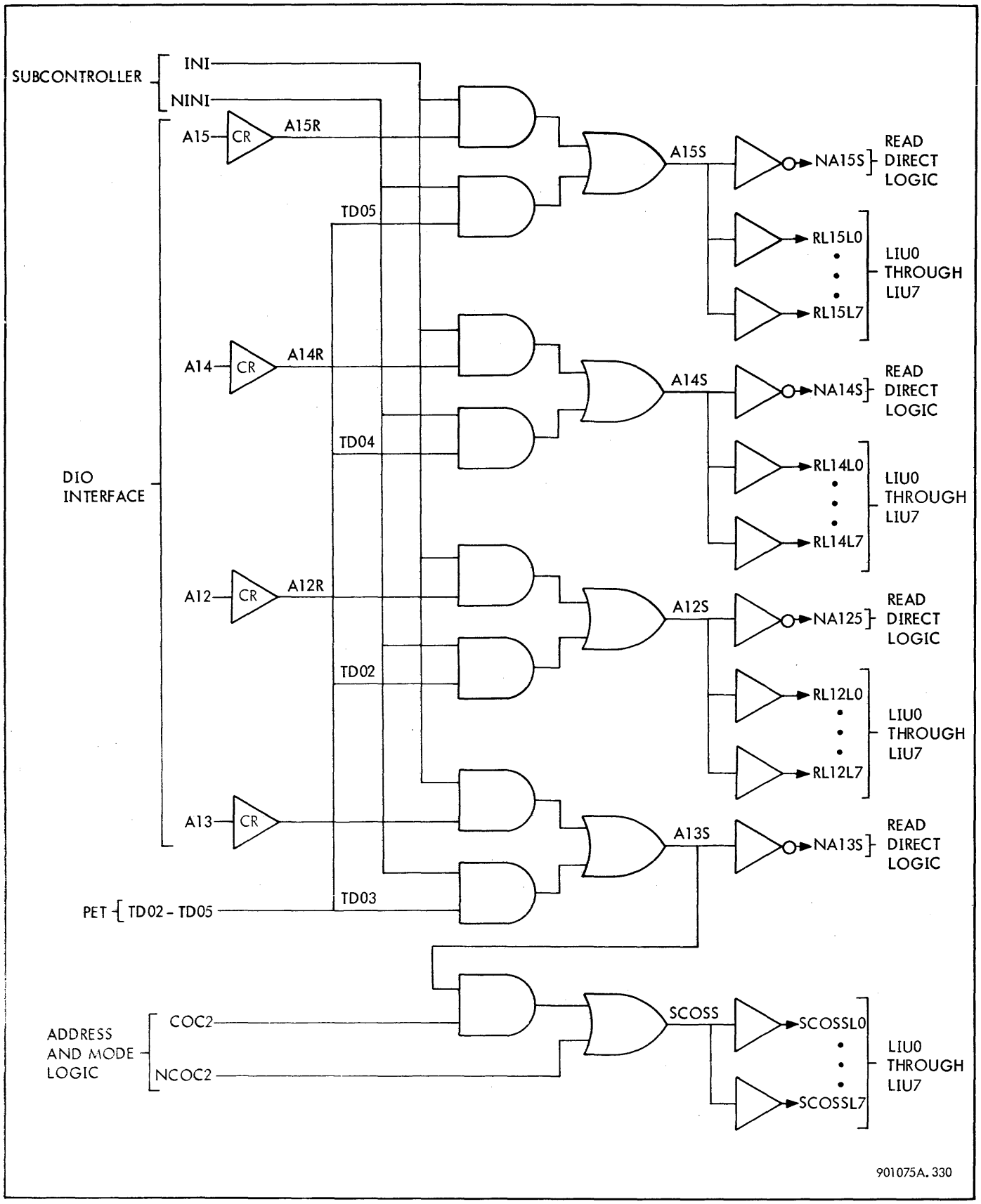


Figure 3-30. Function Logic, Logic Diagram

3-52 Read Direct and Scope Sync Logic. This logic is shown in figure 3-31. The purpose of the read direct logic is to generate a signal (ERDD) that gates send scanner signals SS1 through SS6 (LIU and line number) on to the DIO interface lines during a read direct instruction. (See paragraph 3-48.) Signal ERDD also drives line DB25 through a cable driver.

The CPU drives line RWD high during a write direct instruction and holds it low during a read direct instruction. During a read direct instruction encoding the output response function, RWDR (RWD) is inverted and is gated with NA12S through NA15S to generate signal READDIR. Signals NA12S through NA15S are true during the output response function.

READDIR = NRWDR NA12S NA13S NA14S NA15S

Signal READDIR is gated with COC1 (properly addressed read/write instruction) and with FS1 (function strobe received) to generate ERDD.

Signal RWDR is used by the status logic to enable the generation of condition codes during a write direct instruction. The CPU may also drive lines DB24 and DB25 during a write direct instruction to generate a signal for synchronizing an oscilloscope. The duration of the sync pulse is limited by the function strobe because signal COCFS1 is also used in its generation.

3-53 External Interrupt Logic (Send). The external interrupt logic that applies to a send operation is shown in figure 3-32. A timing diagram for this logic is shown in figure 3-26. The purpose of this logic is to generate an interrupt signal (SINT) on an external priority interrupt line every time that the send module makes a service request.

Signal NSSR, received from the send scanner logic, is normally high when the scanner is running. Signal NSSR goes low when a send module makes a service request. This raises SSR, which is gated with NCOC2 to provide a set signal that sets flip-flop FA. Signal FA controls the setting of flip-flop FB which sets one-half clock time after FA. Flip-flop FA is clocked by signal CLOCK, and flip-flop FB is clocked by signal NCLOCKA which falls one-half clock time after signal CLOCK.

S/FA = SSR NCOC2
R/FA = .
C/FA = CLOCK
S/FB = ULSC11A
ULSC11A = NCOC2 FA SSR
R/FB = .
C/FB = NCLOCKA

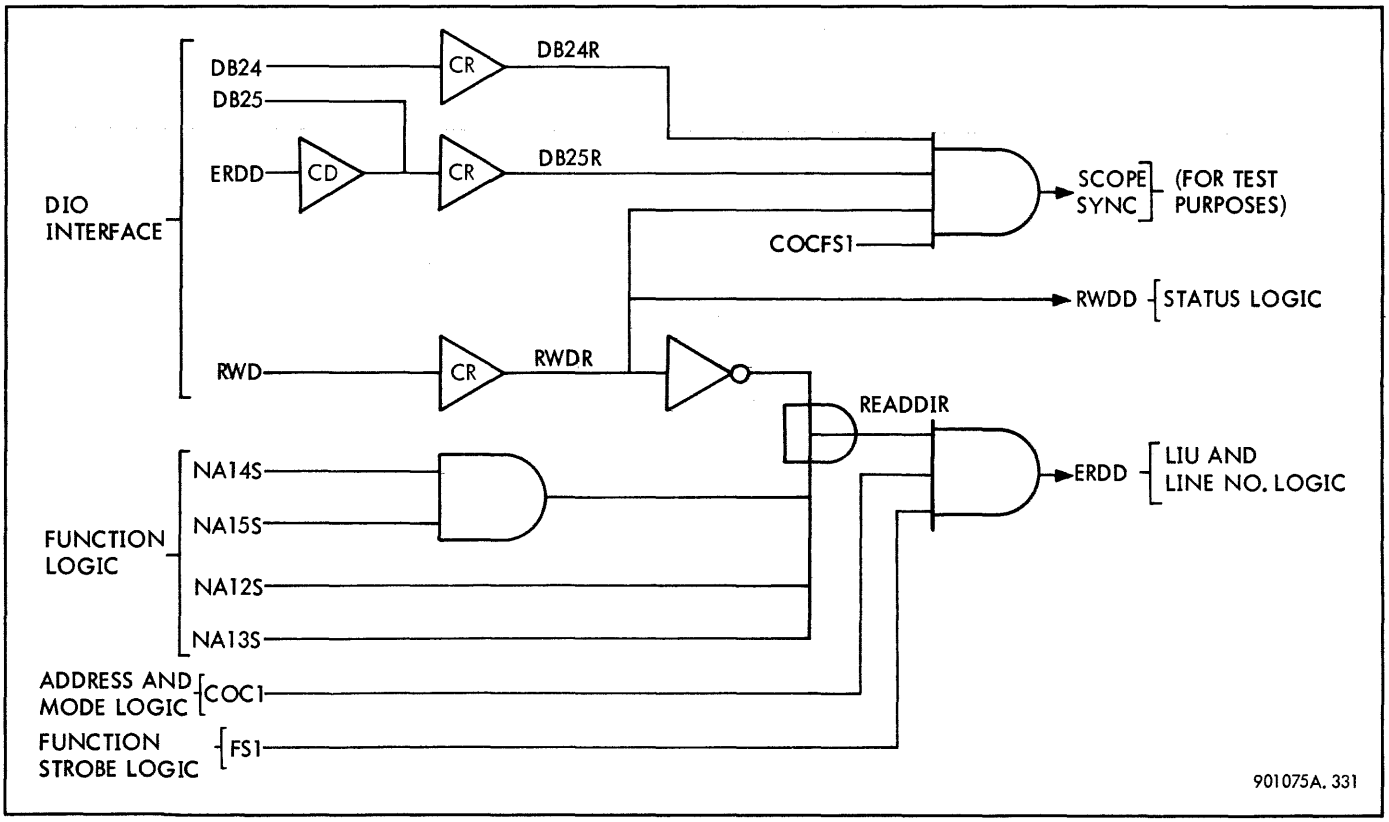


Figure 3-31. Read Direct and Scope Sync Logic, Logic Diagram

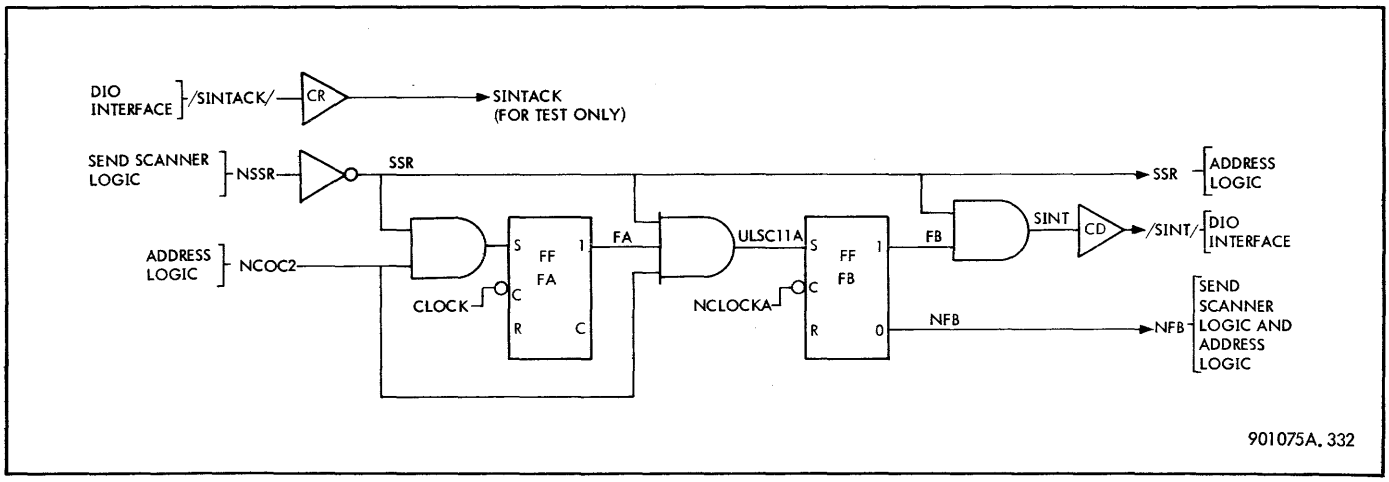


Figure 3-32. Send Interrupt Logic, Logic Diagram

The send interrupt signal, **SINT**, is generated and is sent to the CPU when signal FB goes true.

/SINT/ = SINT = SSR FB

During the time that flip-flop FB is set, signal NFB is used by the send scanner logic to inhibit the send scanner from running again until after the read direct or the write direct instruction (resulting from the interrupt) has been issued by the CPU.

Signals NFB and SSR are also used by the address logic to generate signal COC1 when operating offline using the PET.

In response to interrupt signal SINT, the CPU issues signal send interrupt acknowledge SINTACK, which is used only for maintenance.

3-54 Status Logic. The status information consists of condition code bits CC3 and CC4, which the COC generates in response to all write direct instructions. The condition code lines are held false during a read direct instruction. (For Sigma 2, the CC3 and CC4 bits are designated overflow and carry, respectively.) The logic that generates the condition code information is shown in figure 3-33. See tables 2-3 and 2-4 for interpretation of the condition code bits during receive and send operations.

The source of the status information is the circuits contained on the send, receive, and interface modules addressed by the write direct instruction. When the instruction is received, the addressed send-receive position generates signals NSTM1 and NSTM2. These signals are received by the status logic as NSTM1G0 through NSTM1G7 (from LIU0 through LIU7, respectively) and by NSTM2G0 through NSTM2G7 (from LIU0 through LIU7, respectively). Each of these groups is gated to generate signals NST1 and NST2.

NST1 = NST1G0 NST1G1 NST1G2 NST1G3
NST1G4 NST1G5 NST1G6 NST1G7

NST2 = NST2G0 NST2G1 NST2G2 NST2G3
NST2G4 NST2G5 NST2G6 NST2G7

Since only the addressed send-receive position controls the status lines, the status encoded in signals NST1 and NST2 applies only to the addressed position.

Signals NST1 and NST2 are inverted to generate signals ST1 and ST2. Signal ST1 is gated with RWDR and COCFS1 to generate signal CC3D. Signal CC3D drives line CC3 to the CPU through a cable driver. Similarly, signal ST2 is gated with RWDR and COCFS1 to generate signal CC4D. Signal CC4D drives line CC4 to the CPU through a cable driver.

/CC4/ = CC4D = ST2 RWDR COCFS1

/CC3/ = CC3D = ST1 RWDR COCFS1

Signal RWDR, received from the read direct logic, is true only during a write direct instruction. Thus, the condition code lines are always false during a read direct instruction. Signal COCFS1 is true after the function strobe is received from the CPU; therefore, the condition code lines are not driven until that time.

3-55. Reset Logic. The reset logic is shown in figure 3-34. When operating online, the reset signal (RSTR) is received from the MIOP. The reset signal can also be generated by the PET when operating offline. Signal NRESET goes false when the reset signal is generated by either source. This signal is applied to an inverter that generates signal RESET.

RESET = RSTR INI + TSSD NINI

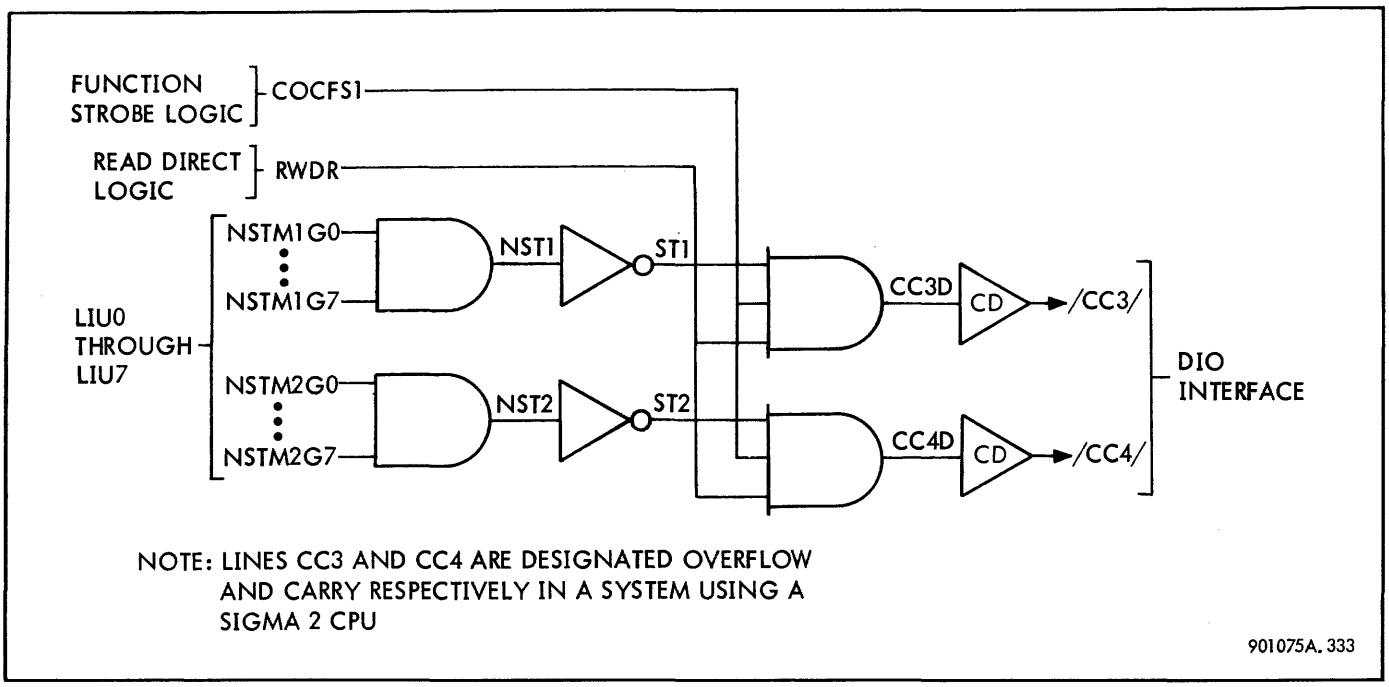


Figure 3-33. Status (Condition Code) Logic, Logic Diagram

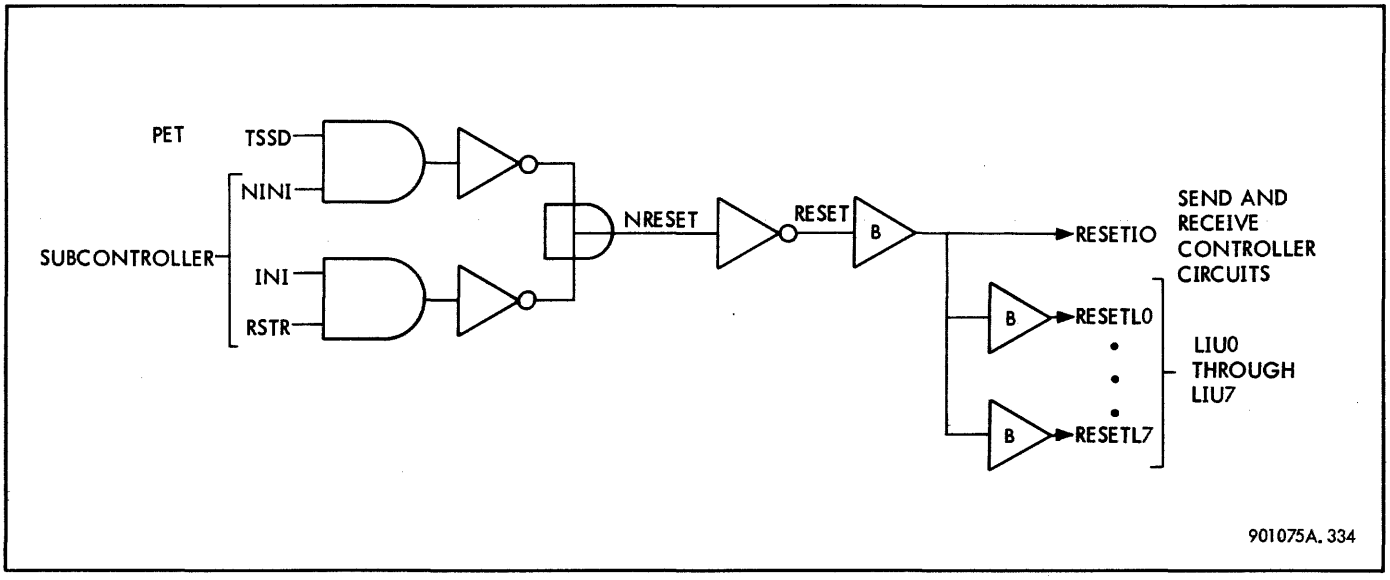


Figure 3-34. Reset Logic, Logic Diagram

Signal RESET is applied to a buffer amplifier which generates signal RESETIO. This signal is used by both the send and the receive circuits in the controller. Signal RESETIO is also applied to eight buffer amplifiers that distribute the reset signal (designated RESETL0 through RESETL7) to the eight LIU's.

3-56 Line Interface Unit

The registers and the logic element groups that comprise the Line Interface Unit (LIU) are segregated into four functional groups: the LIU control logic, the send module, the receive module, and the interface module.

Each LIU employs a maximum of eight send-receive positions. Each position consists of one send, one receive, and one interface module. The LIU control logic, contained on two LT46 modules, controls all eight send-receive positions.

Since the COC controller serves up to eight LIU's, the signals generated by the controller that are uniquely applied to a specific LIU incorporate the LIU number as part of the signal designator. For example, the signals generated by the receive scanner decoding logic (RSEL0 through RSEL7) that represent the LIU number are applied to their respective LIU's; however, the signal designation at each of the LIU's is simply RSEL. The LIU number is dropped. The opposite is true of signals generated by the LIU's. For example, the service call signal from any LIU is simply NRSRM. Its designation at the controller, however, reflects the number of the LIU from which it is received; NRSRM0 is received from LIU0, NRSM1 is received from LIU1, and so forth.

3-57 LIU CONTROL LOGIC. The LIU control logic consists of the three functional blocks and the miscellaneous gating shown in figure 3-3.

3-58 Send Module Selection Logic. The send module selection logic (figure 3-35) decodes signals COSA1 through COSA3 generated by the send scanner decoding logic in the controller. For a particular configuration of the state of the three input signals, the decoding logic generates one of the select transmitter signals (STE0 through STE7). These signals select the send module and the interface module in the send-receive position specified by the send scanner decoding logic. The source of the address may have been the send scanner (if a read/write instruction is not being executed) or it may have been encoded in the read/write instruction.

The complements of the three signals received from the send scanner decoding logic are formed by inverters in the selection logic.

STE0 = NCOSA1 NCOSA2 NCOSA3 SCOSS

STE1 = COSA1 NCOSA2 NCOSA3 SCOSS

STE2 = NCOSA1 COSA2 NCOSA3 SCOSS

⋮ ⋮ ⋮ ⋮ ⋮

STE7 = COSA1 COSA2 COSA3 SCOSS

The select transmitter signals are generated only if signal SCOSS is true. During a read/write instruction, SCOSS is true if the function pertains to a send operation, and SCOSS is false if it pertains to a receive operation. Signal SCOSS also is true during the time that a read/write instruction is not being executed by the COC. (See paragraph 3-51.) Signal NSCOSS (SCOSS through an inverter) is used as an enable signal by the receive module selection logic because it is true when the function applies to a receive operation.

3-59 Receive Module Selection Logic. The receive module selection logic (figure 3-36) functions in a similar manner to the send module selection logic; that is, it decodes signals COSA1 through COSA3 generated by the send scanner decoding logic in the controller. This logic decodes the three input signals to generate eight select receiver signals (SRE0 through SRE7). These signals select the interface module in the send-receive position specified by the send scanner decoding logic during a read/write direct instruction.

Signal NSCOSS, received from the send module selection logic, is true if the function pertains to a receive operation. Signal NSCOSS enables decoding of COSA1 through COSA3 to generate the select receiver signals. The complements of COSA1 through COSA3 are formed by inverters in the selection logic.

SRE0 = NCOSA1 NCOSA2 NCOSA3 NSCOSS

SRE1 = COSA1 NCOSA2 NCOSA3 NSCOSS

SRE2 = NCOSA1 COSA2 NCOSA3 NSCOSS

⋮ ⋮ ⋮ ⋮ ⋮

SRE7 = COSA1 COSA2 COSA3 NSCOSS

3-60 Function Decoding Logic. The function decoding logic (figure 3-37) decodes the function encoded in the read/write direct instruction. The function, encoded in bits 28 through 31 of the effective address field of the instruction, is placed on DIO interface lines A12 through A15. The function logic in the controller generates the following signals from A12 through A15: RL15 (A15), RL14 (A14), RL12 (A12), and SCOSS (A13). Signal A13 (SCOSS) is true when the function applies to a transmit operation, and A13 is false when the function applies to a receive operation. Signal A13 is used to enable decoding of the appropriate select transmitter or select receiver signals (STE or SRE). The other three signals (RL12, RL14, and RL15) are decoded to specify the function that is to be performed. The functions that are encoded in a write direct instruction are described in paragraph 2-22. The functions that are encoded in a read direct instruction are described in paragraph 2-32. The output response function (the only one encoded in the read direct instruction) is essentially decoded by the read direct logic in the controller. (All four function code bits are false.) The output response function is, therefore, not decoded by the function decoding logic in the LIU.

The functions are decoded only if an enable signal (COCL gated with LIUL) is true. These signals are true if the COC and the LIU are being addressed by the instruction.

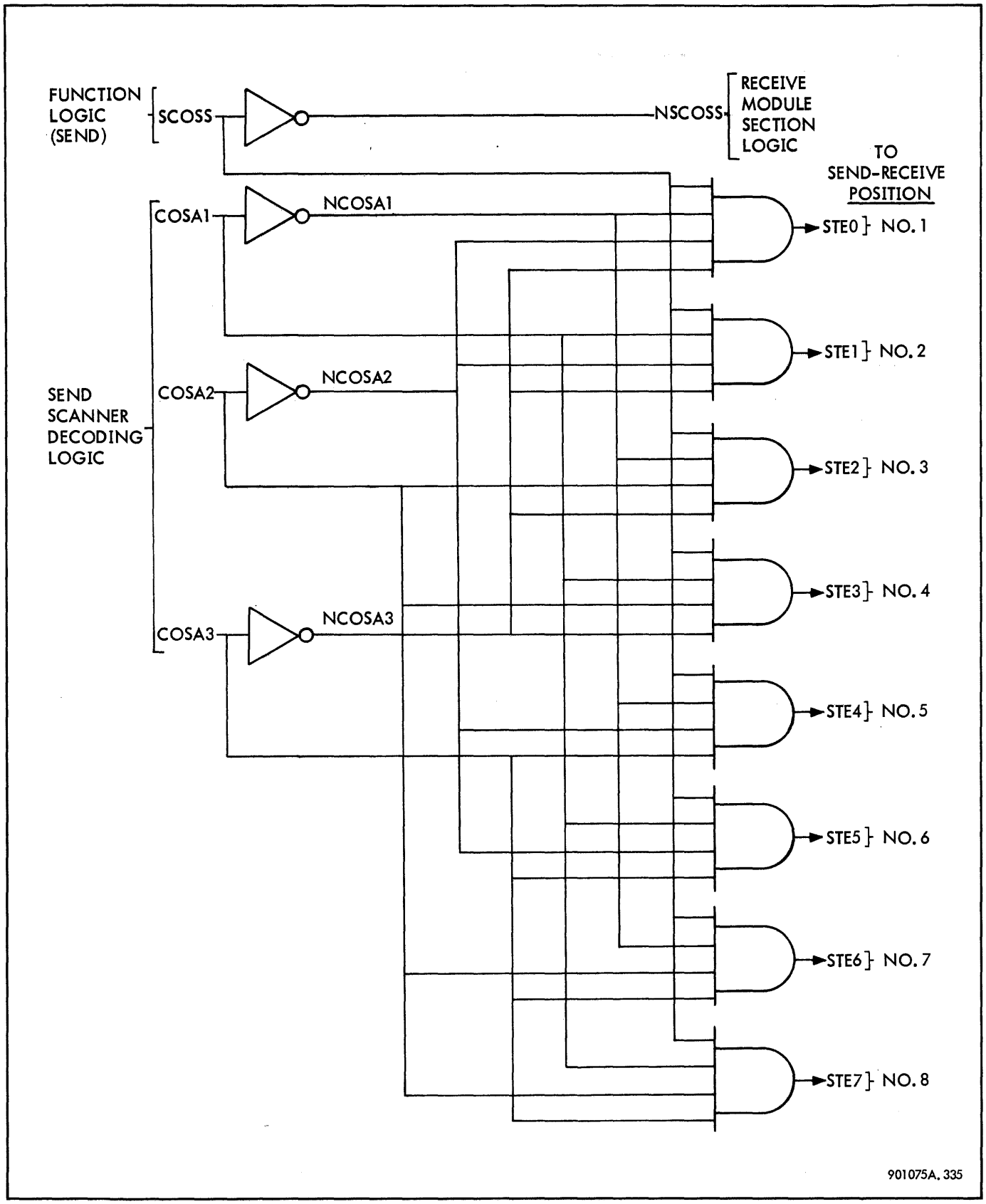


Figure 3-35. Send Module Selection Logic, Logic Diagram

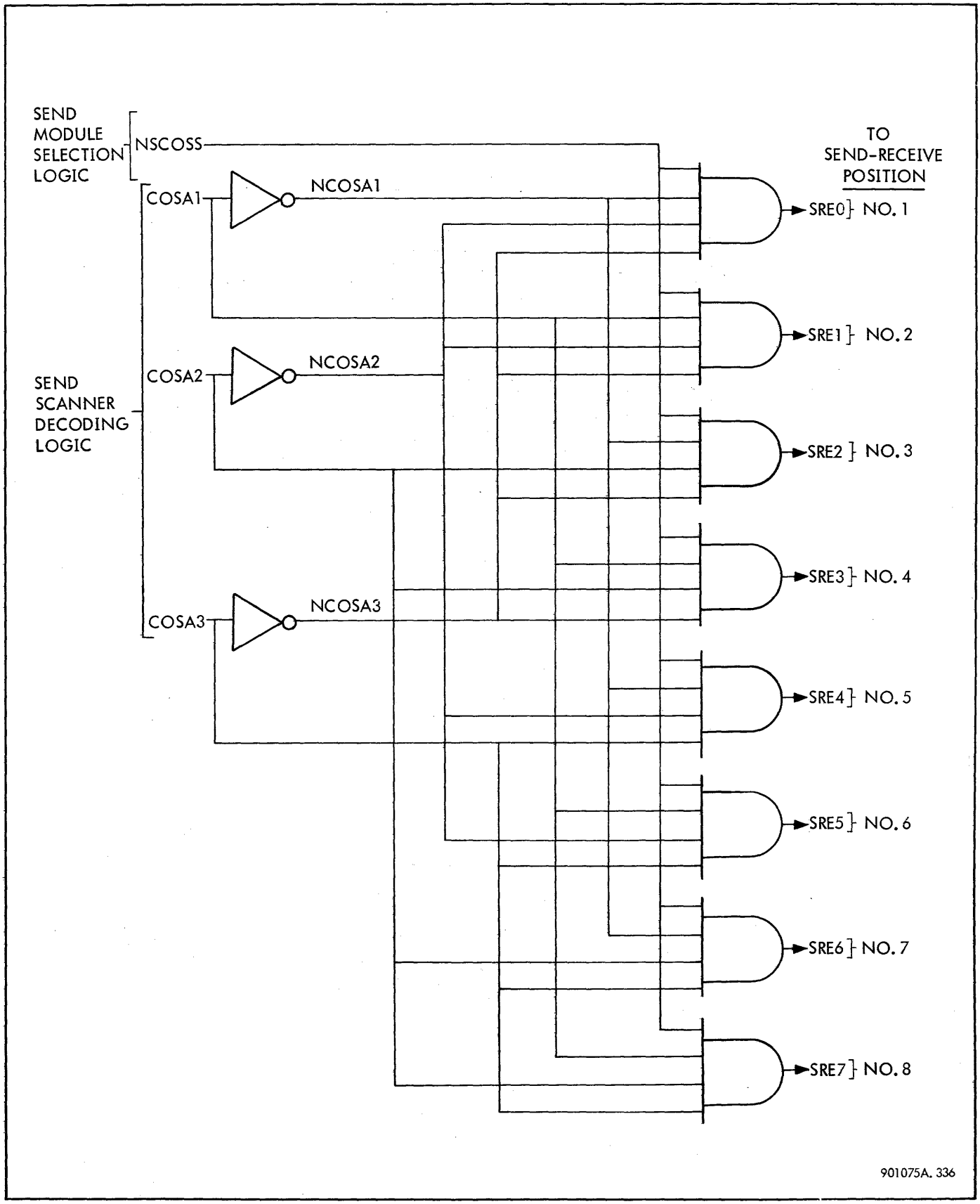


Figure 3-36. Receive Module Selection Logic, Logic Diagram

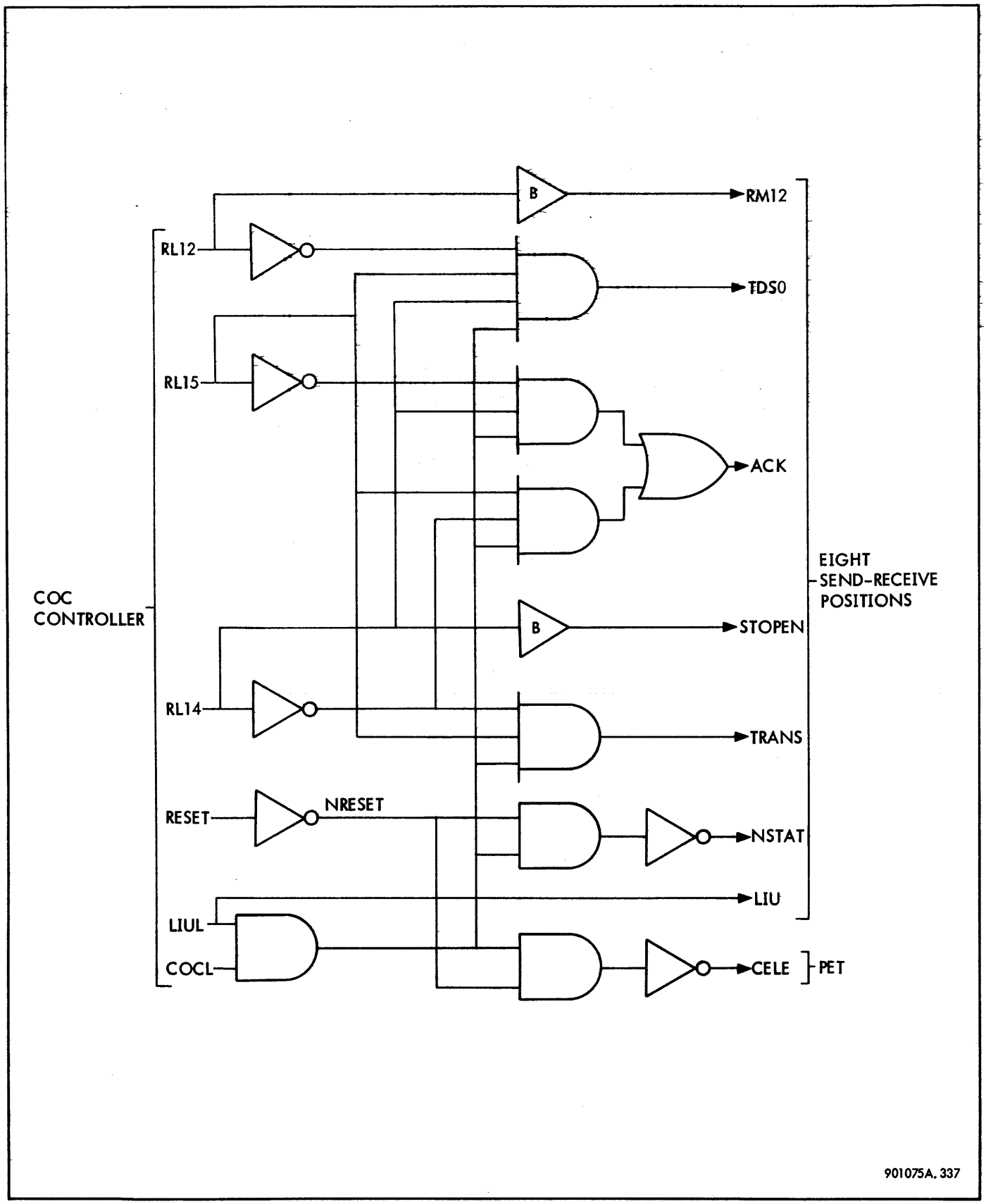


Figure 3-37. Function Decoding Logic, Logic Diagram

The function decoding logic generates the four signals described in the following steps (this logic is mechanized twice in the LIU for loading purposes):

a. Acknowledge (ACK). The acknowledge signal is generated by two configurations of signals, RL14 and RL15, as part of the following functions:

RL14	RL15	
0	1	(Turn receiver on, transmit data, and transmit long space)
1	0	(Turn receiver off and stop transmit)

ACK = COCL LIUL RL14 NRL15
+ COCL LIUL NRL14 RL15

The purpose of the acknowledge signal is to reset FSLP, to set FSTOP on the send module, and to reset the receiver on flip-flop (FRON) which is located on the associated interface module. Flip-flop FRON is reset to prevent the receive module from requesting service.

b. Stop enable (STOPEN). The stop enable signal is not actually decoded; it is the output of a buffer amplifier controlled by signal RL14. Signal STOPEN is used in the stop transmit function.

STOPEN = RL14

During the stop transmit function, signal STOPEN is sent to the send module where it is gated with signal ACK to form a term that sets flip-flop stop (FSTOP) and resets flip-flop long space (FLSP). The output of the send module immediately switches to a mark state (all ones) when flip-flop FSTOP is set.

c. Turn data set off (TDSO). Signal TDSO is used with a receiver function (turn receive data set off) or with a transmit function (turn transmit data set off) depending on the state of SCOSS (A13).

TDSO = COCL LIUL RL14 RL15 NRL12

Signal TDSO is used to set flip-flop FSOFF (if the function is turn transmit data set off) or to set flip-flop FROFF (if the function is turn receive data set off). These flip-flops are located on the interface module associated with the data set to be controlled.

d. Transmit (TRANS). During transmit operations, signal TRANS is used with the transmit data and the transmit long space functions. During receive operations, signal TRANS is used with the turn receiver on function. Signal TRANS is true when RL14 is false; RL15 is true; and the enable signal is true.

TRANS = NRL14 NRL15 COCL LIUL

During a transmit operation, whether the function applies to data or to long space depends on the state of signal RM12. Signal RM12 (RL12 through a buffer amplifier) is not used in the formation of signal TRANS. Instead, it is applied directly to the send module where it is used with signal TRANS to set flip-flop long space FLSP if the function is transmit long space. In both cases (transmit data or transmit long space), signal TRANS sets flip-flop FRUN and resets flip-flop FSTOP. Both of these flip-flops are located on the send module.

During receive operations, signal TRANS is used to set flip-flop FRON located on the interface module if the function is turn receiver on. Flip-flop FRON must be set before the receive module can request service; that is, FRON must be set when it has a character to input and when the send scanner has selected its address.

The nine possible functions are listed below with the signals that apply to each.

RL12	RL13 SCOSS	RL14	RL15	Function
0	0	0	0	Sense receive status
0	0	0	1	Turn receiver on
0	0	1	0	Turn receiver off
0	0	1	1	Turn receiver data set off
0	1	0	0	Sense transmitter status
0	1	0	1	Transmit data
1	1	0	1	Transmit long space
1	1	1	0	Stop transmit
0	1	1	1	Turn transmit data set off

3-61 INTERFACE MODULE. Three types of interface modules may be used in the LIU (see paragraph 1-2). The EIA (NT21) and the MIL-STD-188D (NT32) interface modules are logically identical. The dc (NT20) module contains fewer circuits than the other two because it is not required to control data sets. The interface module logic for an NT21 is shown in figure 3-38.

The interface modules contain circuits that assemble status information, control the send and receive modules, and provide for matching the logic levels of the line with those of the COC. These functions are described in paragraphs 3-62 through 3-64.

3-62 Cable Receivers and Drivers. The cable receivers and drivers on the NT21 and the NT32 modules provide

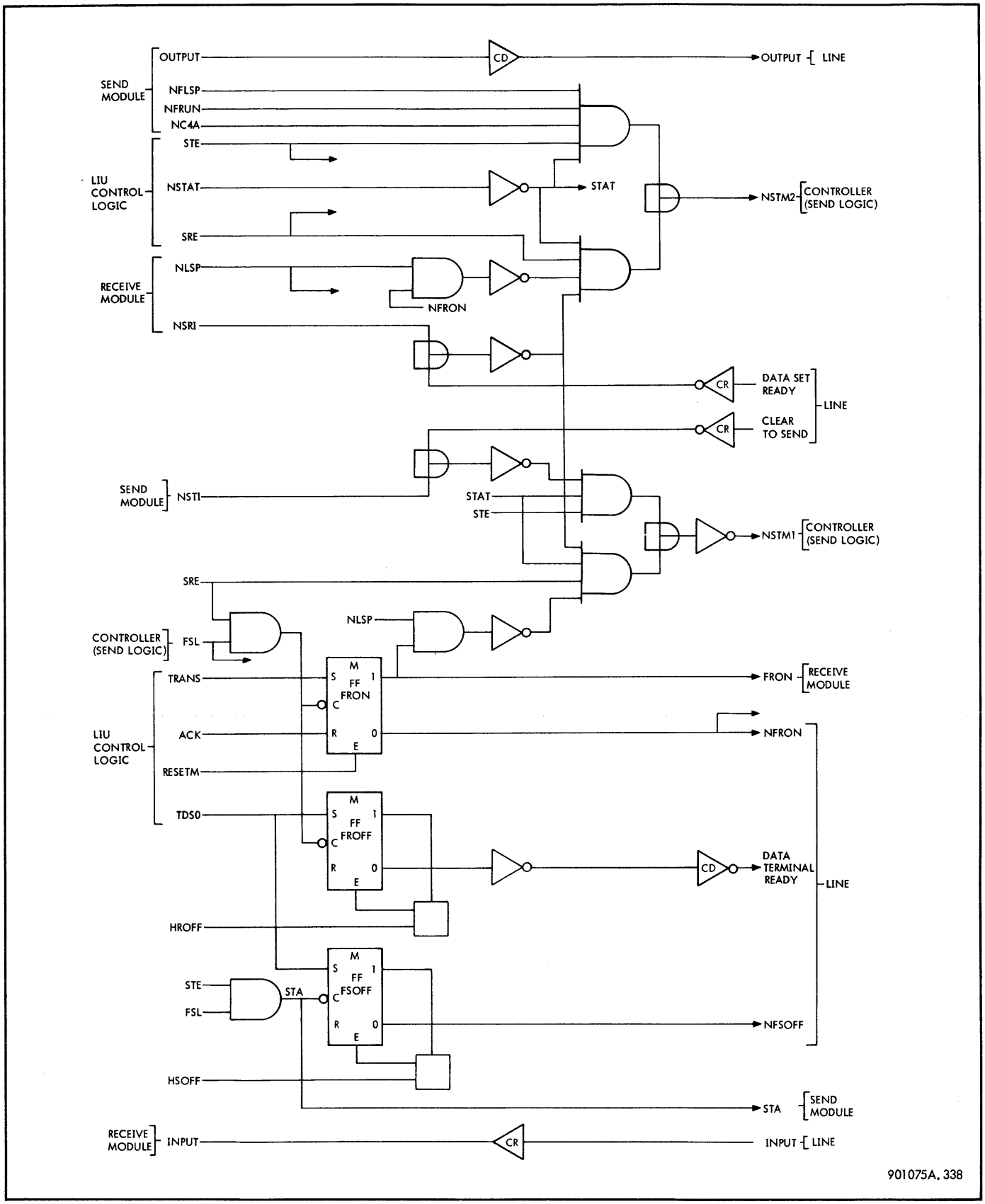


Figure 3-38. Interface Module (NT21), Logic Diagram

logic-level matching between the COC and the line. Using the NT21 module, the voltage value on the line for a marking condition (true state) is -5 volts; for a spacing condition (false state), it is +5 volts. With the NT32, the marking condition is +6 volts, and the spacing condition is -6 volts. The line is normally in a marking condition when no signals are present. The voltage value for the true and the false logic states in the COC are +4 volts (mark) and 0 volts (space), respectively. When operating with the NT20 module, a KT12 relay module (part of Relay Interface Unit Model 7620) is used between the line and the interface module. (See paragraph 3-65.)

3-63 Interface Module Status Logic. Every time that a transmit or receive function is activated, information concerning the status of the transmit or the receive circuits is sent back to the CPU via the condition code lines. The status information is assembled on the interface module for the send or the receive module being addressed. The status logic generates signals NSTM1 (CC3) and NSTM2 (CC4). These signals are applied to the status logic that function as part of the send logic in the controller. See tables 2-3 and 2-4 for the meaning of the status (condition code) signals.

The status logic is activated when signal NSTAT, received from the LIU control logic, goes false. Signal NSTAT in its quiescent state is normally true; however, it is inverted on the interface module. One of the signals, STE or SRE (transmit or receive operation), must also be true to enable the generation of status information. Signal STE or SRE is true for only the addressed transmitter or receiver (one of eight per LIU). When the enabling signals are true, the status logic samples signals from the line, from the send and receive modules, and from those generated on the interface module to generate status signals NSTM1 and NSTM2.

3-64 Interface Module Control Logic. The NT21 and the NT32 interface modules contain three flip-flops: FRON, FROFF, and FSOFF. The purpose of flip-flop FRON is to enable and to disable the service request circuits in the receiver. If signal FRON is true and if the receiver has assembled an incoming character, the receiver makes a service request when it is addressed by the receive scanner. Signal FRON in the false state inhibits service requests by the receiver even if it has assembled a character and has been addressed by the scanner. Flip-flop FRON is set by signal TRANS and is reset by signal ACK at the fall of the clock signal. The clock signal generated by signal FSL is gated with signal SRE. Signal FSL is received from the function strobe logic in the controller associated with the DIO interface. Signal SRE is the decoded receiver address signal that is received from the LIU control logic.

S/FRON = TRANS

R/FRON = ACK

C/FRON = FSL SRE

See function decoding logic, paragraph 3-60, for a description of signals TRANS and ACK. Flip-flop FRON serves the same purpose on the NT20 module as it does on the NT21 and the NT32 modules.

The purpose of the FROFF and the FSOFF flip-flops on the NT21 and the NT32 modules is to cause the data terminal ready line to the associated data set to go false for at least 50 milliseconds. This causes the data set to disconnect from the communications channel. Signal TDSO is generated when either of two functions, turn receiver data set off or turn transmit data set off, are encoded in the instruction. This signal is applied to the set input of both flip-flops. The clock signal that controls flip-flop FROFF, however, is generated only during a receive function, and the clock signal that controls flip-flop FSOFF is generated only during a transmit function.

S/FROFF = TDSO

C/FROFF = FS SRE

S/FSOFF = TDSO

C/FSOFF = FS STA

STA = STE FS

Each flip-flop controls a time delay circuit that is activated when the flip-flop sets. After a delay of at least 50 milliseconds, the time delay circuit resets the flip-flop. During the time that the flip-flop is set, the data terminal ready line is held false, causing the data set to disconnect.

3-65 RELAY INTERFACE UNIT MODEL 7620. Whenever the dc interface module (NT20) is used, a telegraphic relay module (KT12) must also be used. These relay modules are installed in the Relay Interface Unit Model 7620 which is a separate unit. The relay interface unit accommodates up to eight KT12 modules and also contains a monitor unit assembly. Each KT12 module is connected to its associated NT20 module by a cable. If the requirement for the number of NT20 modules in a system exceeds eight, additional relay interface units (one for each eight KT12 modules) are required. The relay interface unit may be installed in the same cabinet as the COC, or it may stand alone.

If the relay interface unit is used, the communication lines require a source of dc power other than that supplied by the COC power supply circuits. This dc power may be supplied either by the customer or by Dc Power Supply Model 7623.

The dc power supply basically consists of a PX14 power supply modified, as required, to adapt it to the

communications system. Electrically, the power supply operates the same as the PX14. Some of the modifications consist of an additional cable to connect the power supply to the primary power source and of wiring from the power supply output to the communications line and to the relay interface unit. Connection of the power supply, relay interface unit (including the KT12 module) and the interface module (NT20) is shown in figure 3-39.

The power supply connects to the line and to the relay interface unit terminal board. A separate set of terminals are provided for each channel. The letters (M, C, S, P, and N) on the terminal board stand for mark, common, space, positive, and negative, respectively. Internal wiring protected by circuit breakers connects the terminal boards to the receptacles (slots) into which the KT12 modules are plugged. There are eight redundant circuits (terminal boards, receptacles, and sets of circuit breakers) in the relay interface unit (only one of the circuits is shown in figure 3-39).

The power supply provides power for up to 16 simplex, half-duplex, or full-duplex circuits over distances of up to five miles. If line current is supplied by a Sigma computer, one Model 7623 power supply is necessary for unipolar operation; two power supplies are required for bipolar operation.

3-66 Monitor Unit. The monitor unit assembly in the relay interface unit is capable of monitoring both current and voltage on each individual line. A 24-position selector switch selects the desired line (mark, space or receive) of the desired channel (one of eight). A meter on the front panel has two dc voltage scales. One of two pushbuttons on the front panel is pressed to read the lower scale, otherwise the upper scale is read. When a voltage reading is desired, the other pushbutton is pressed which places the meter across the line. When this switch is not pressed, the meter is connected in series with the line to obtain a current measurement.

3-67 Telegraphic Relay Module (KT12). The KT12 module electronics consists primarily of a send and a receive relay, each controlled by a bridge rectifier and by a current regulator circuit. Up to eight of these modules may be plugged into one relay interface unit (see figure 3-39). The KT12 module interfaces with the line and with the power supply by way of the receptacle in which it is inserted. The KT12 module interfaces with its associated current interface module (NT20) by means of a cable with receptacles that attach to the outer edge of the modules.

The send relay coil is controlled by the output circuits on the NT20 module, while the receive relay coil is controlled by remote location. Both relay coils are polarized and may be used in a bipolar system. The bridge rectifier circuits ensure that the correct polarity voltage is applied to the relay coils. This system permits detection of an open line condition (malfunction) that otherwise might be construed as a long space transmission.

The line current is controlled by a regulator circuit (one circuit for send and one for receive) on the KT12 module. The regulators permit either 20 or 60 mA operation, depending on which end of the cable is attached to the KT12 module. Jumper wires, connected to plug P1 of the cable, short circuit resistors in the regulator circuit when P1 is connected to the KT12 module. This manner of connecting the cable provides 60 mA operation. In 20 mA operation, the cable is reversed so that plug P2 which contains no short circuit jumpers, is attached to the KT12 module. Attachment of the cable as shown by solid lines in figure 3-39 provides 60 mA operation; dashed lines show the reversed connection of the cable that provides 20 mA operation. A potentiometer in the regulator circuit provides a means of adjusting the current, which may be monitored on the monitor unit.

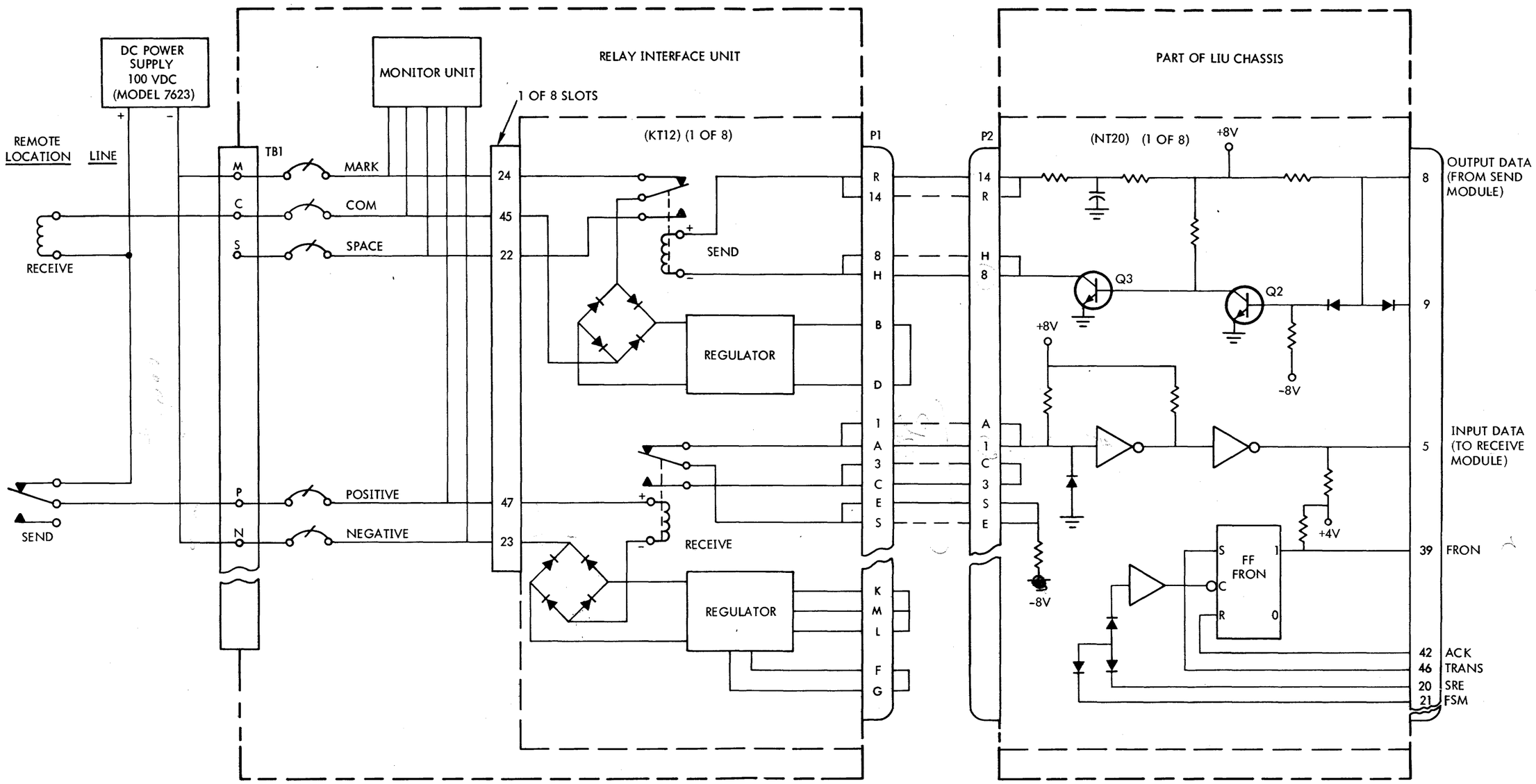
3-68 RECEIVE MODULE. Four types of receive modules are available for use with the LIU. The type used depends on the transmission format desired. (See paragraph 1-2.) The primary difference between the four types is the number of flip-flops used in the character register. The number of flip-flops corresponds to the level of the transmission code used.

The purposes of the receive module are to accept serial information from the line via the interface module, to make a request for service when a complete character has been assembled, and to parallel shift the character to the MIOP via the controller. Timing for this operation is derived from the Model 7612 timing modules. The clock signal used by the controller circuits (supplied by the MIOP) is used by the LIU only for clearing the registers, and not for shifting purposes.

All receive modules (activated or not activated) are continuously maintaining synchronization with their input lines. When a receive module is activated, it may assemble a new character, may complete an assembly in progress when activation occurred, or may contain a previously assembled character.

All four types of receive modules primarily consist of a character register, a clock counter, and a control logic. On all four types of receive modules, the clock counter consists of flip-flops FA, FB, and FC. On the LT53 and the LT54 modules (both are eight-level code), the character register consists of flip-flops R1 through R8; on the LT52 module (seven-level code), the character register consists of R1 through R7; and on the LT51 (five-level code), the character register consists of R1 through R5. On all four types of receive modules, the flip-flops that serve as the control logic are RS1, RS2, FRLS, and FDUMP. On the LT51 module only, FCASE also serves as the control logic.

Since operation of the four receive modules is somewhat similar, only the LT51 module is shown in figure 3-40. Figure 3-41 is a timing diagram showing the receive signals for an input character with the bit pattern shown in the diagram.



NOTES:

1. KT12 MODULE, NT20 MODULE, AND INTERCONNECTING CABLE ARE DC INTERFACE ASSEMBLY 149610
2. REGULATORS ON KT12 MODULE ARE 20 MA OR 60 MA, DEPENDING ON WHICH END OF CABLE (WITH OR WITHOUT JUMPERS) IS CONNECTED TO KT12 MODULE. CABLE IS SHOWN FOR 60 MA OPERATION

Figure 3-39. Relay Interface Unit Connections to the Line, Power Supply, and Interface Module (NT20), Simplified Schematic Diagram

901075A.339

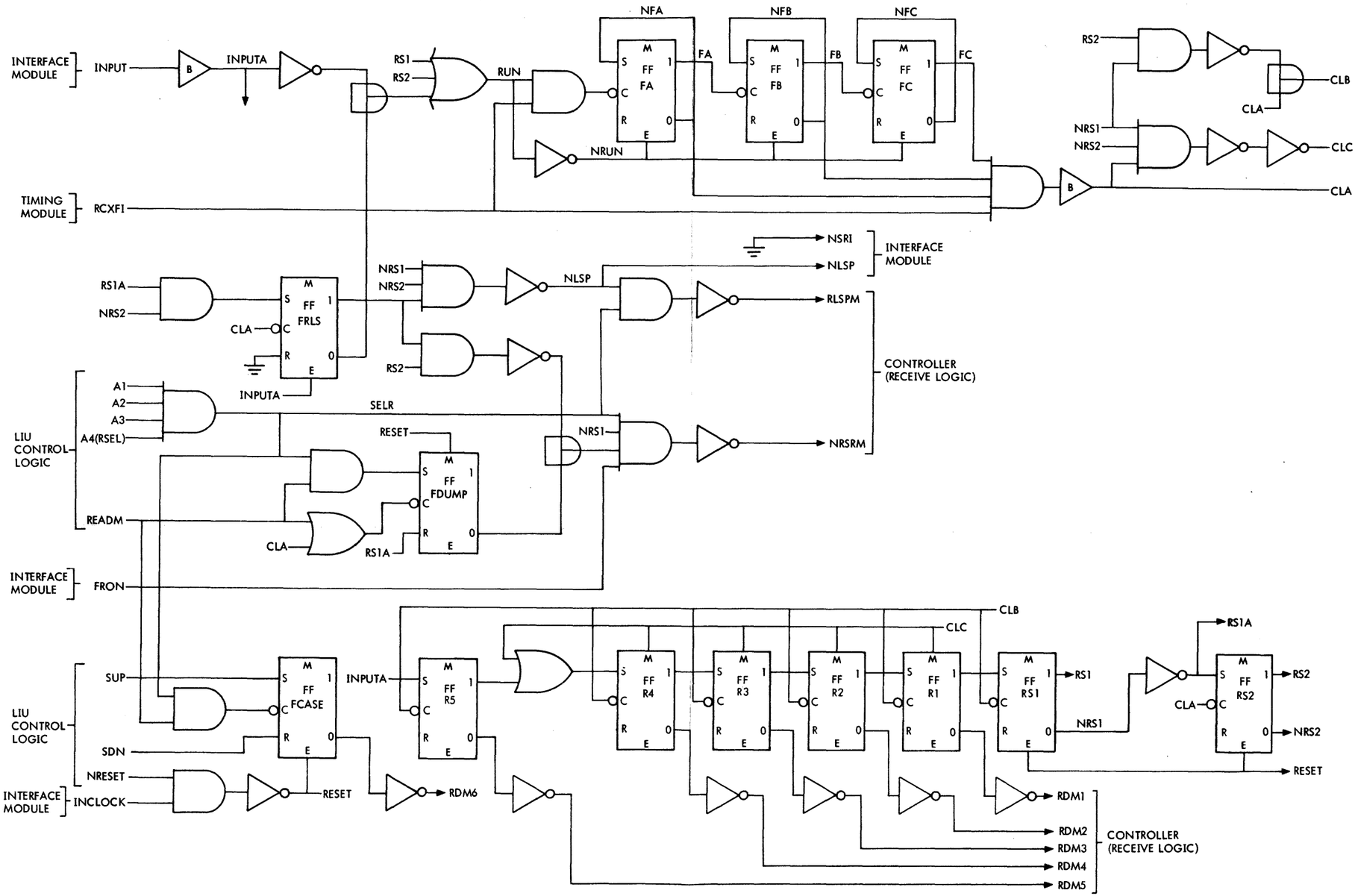


Figure 3-40. Receive Module (LT51), Logic Diagram

901075A.340

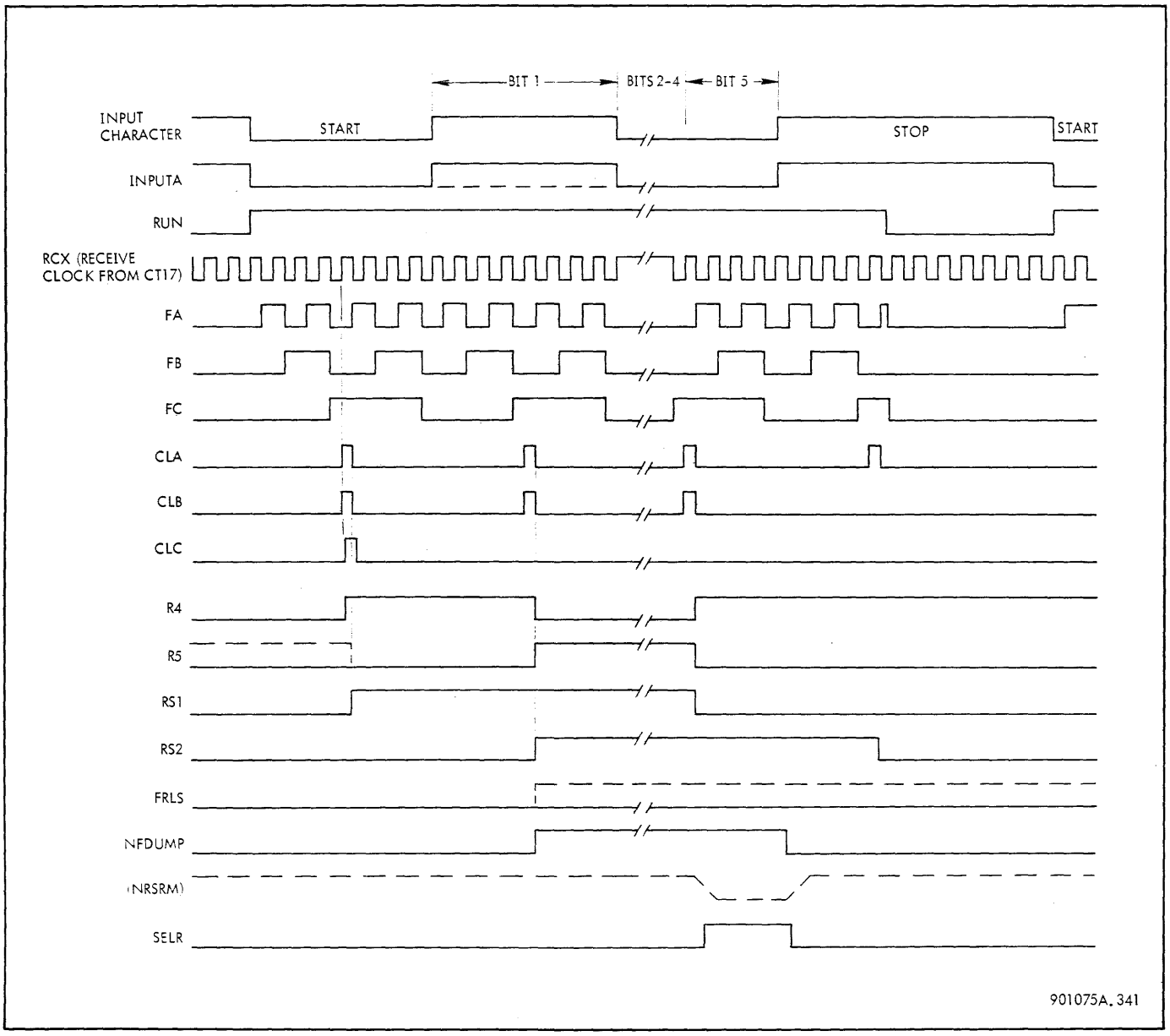


Figure 3-41. Receive Module (LT51) Input, Timing Diagram

Since the line is normally in a marking condition, signal INPUTA is normally true. When the start bit of the character is received, INPUTA goes false which causes RUN to go true. When RUN goes true, the receive clock RCXFI (format I) steps the counter. The counter generates the internal timing for shifting the incoming data into the character register. Three clock signals are generated by the counter. These are CLA, CLB, and CLC. Clock signal CLA is generated once for every incoming bit time and appears at approximately the mid-point of the bit period.

$$CLA = NFA\ NFB\ FC\ RCXFI$$

Clock signal CLB coincides with CLA except when the stop bit occurs.

$$CLB = CLA\ (RS1 + NRS2)$$

Note

The symbol △ indicates that the rise and fall times of the clock are somewhat delayed with respect to CLA.

Clock signal CLC is generated during the start bit only.

CLC = △ (CLA NRS1 NRS2)

When the first CLB clock occurs, the start bit is shifted into flip-flop R5 of the character register. Clock CLC, which occurs at the same time but is delayed slightly, sets flip-flops R1 through R4. Flip-flop RS1, which is clocked by CLB, is also set since its set input is a one before the fall of CLB. The setting of RS1 inhibits further generation of CLC clocks. RS1 also maintains the RUN signal so that the counter continues to count even though the data input line may be changing.

At the second CLB clock, the first data bit is shifted into flip-flop R5 while the start bit is shifted into R4. At the same time, the second CLA clock sets flip-flop RS2.

S/RS1 = R1

C/RS1 = CLB

R/RS1 = .

S/RS2 = RS1A

C/RS2 = CLA

R/RS2 = .

Flip-flops RS1 and RS2 are used to indicate that a complete character has been received; that is, the start bit has shifted through R5 through R1 of the character register. Signal RS2 also serves as an additional enable term for RUN so that the counter continues to generate the internal clocks (CLA, CLB, and CLC) until the full character has been received.

RUN = NINPUTA + RS1 + RS2

When the second CLA clock occurs, an attempt is made to set the receive long space flip-flop, FRLS.

S/FRLS = RS1A NRS2

C/FRLS = CLA

Long space is defined as a complete character made up of all zeros (no stop bit). If the first data bit is a zero, flip-flop FRLS is set (see the dashed line in figure 3-41). If FRLS is set, it remains set until the first one bit of the character or the stop bit occurs. At this time, INPUTA, applied to the erase input, resets FRLS.

E/FRLS = INPUTA

If FRLS is set and is not reset again until the stop bit time occurs, a long space has been detected. For this example, however, the first character bit is a one; therefore, FRLS remains in the reset state during receipt of this character.

As incoming data enters the character register, the start bit is shifted through the register until it reaches RS1. When RS1 resets, the complete character has been received with the data in bits R1 through R5. At this time, the logic generating the CLB clock is disabled, and the receive module is ready to request service.

The dump flip-flop, DUMP, is reset at the second CLA clock time at the beginning of the input character.

R/DUMP = RS1A

C/FDUMP = CLA +...

The purpose of the dump flip-flop is to terminate the request for service after the character has been sent to the MIOP. The service request is terminated when flip-flop FDUMP is set.

S/FDUMP = READM SELR

C/FDUMP = READM +...

Signal READM is generated by the byte two logic in the receive section of the controller when the receiver address is being sent to the MIOP. The receiver address, specified by the receiver scanner, is transferred to the MIOP following the transfer of the character. This transfer concludes the two-byte transmission. At the completion of the address transfer, READM goes false, which causes FDUMP to set.

The select receiver signal (SELR) goes true when the receiver scanner selects the address for the receiver module and goes false when the service request is concluded.

The receive module requests service by dropping signal NRSRM.

NRSRM = (Inverter) [FRON NRS1 SELR NFDUMP (NRFLS + NRS2)]

Signal NRSRM is high in its quiescent state. When a service request occurs, NRSRM goes false. The interpretation of the equation states that the service request occurs when:

a. The receiver on flip-flop (FRON) on the interface module is set.

b. A complete character has been shifted into the character register with the start bit, which is false, in RS1 (NRS1).

c. The receiver scanner has selected the receiver address (SELR).

d. The service request is terminated at the fall of READM (NFDUMP).

e. Long space has or has not been detected (NFRLS + NRS2). If a long space has not been detected in the above equation, flip-flop FRLS is reset by a one or by a stop bit before the fall of RS2. Therefore, signal NFRLS enables the service request (NRSRM) indicating no long space. Also, the not long space signal (NLSP) is true at this time.

NLSP = (Inverter) FRLS NRS1 NRS2

The receive long space signal is therefore false.

RLSPM = (Inverter) NSLP SELR

If a long space has been detected (flip-flop FRLS set by the start bit and not reset by a one or a stop bit), the service request signal NRSRM is disabled until the fall of RS2. In this case, the rise of NRS2 enables NRSRM, indicating that a long space has been detected. Signal NLSP is now false while RLSPM is true.

When RS2 resets, signal RUN goes false and disables the counter. Signal NRUN, applied to the erase input of the counter flip-flops, resets the counter in preparation for the next character.

E/FA = NRUN

E/FB = NRUN

E/RC = NRUN

The case flip-flop (FCASE) is used to indicate whether the incoming characters are letters or figures. The codes for letters and figures are shown below.

	R5	R4	R3	R2	R1
Letters	1	1	1	1	1
Figures	1	1	0	1	1

When a letters character is received, the shift down signal (SDN) is generated and is used as the reset term for flip-flop FCASE. When a figures character is received, the shift up signal (SUP) is generated and is used as the set term for flip-flop FCASE. Signals SDN and SUP are generated by the shift logic, which is part of the LIU control logic. After the letters or the figures code has been sent to the MIOP, flip-flop FCASE is set to the appropriate state.

S/FCASE = SUP

SUP = RDM1 RDM2 NRDM3 RDM4 RDM5

R/FCASE = SDN

SDN = RDM1 RDM2 RDM3 RDM4 RDM5

C/FCASE = SELR READM

Signal SELR indicates that the receiver address has been selected by the receiver scanner, and that signal READM is true during the transfer of the second byte to the MIOP. The fall of READM clocks flip-flop FCASE. The state of FCASE remains unchanged (1 = figures, 0 = letters) until the next time that signal SUP or SDN is generated.

3-69 SEND MODULE. There are four types of send modules available for use with the LIU. The type used depends on the transmission format desired. (See table 1-2.) The primary difference between the four types is the number of flip-flops used in the character register. The number of flip-flops corresponds to the level of the transmission code used.

The send module receives information in parallel from the CPU via the DIO interface lines and from the character transmitter logic in the send section of the controller. The information that the send module receives makes up the data portion of the character to be transmitted to the communications lines via the interface module. Both the start and the stop bits are added to the character when it is serially shifted out to the line. After a complete character has been shifted out, the send module initiates a service request to obtain the next character for transmission. Timing for this operation is derived from the timing modules Model 7612.

The four types of send modules consist primarily of a character register, a clock counter, and control logic. The character register consists of flip-flops R1 through R8 on the LT49 and LT50 modules, flip-flops R1 through R7 on the LT48 module, and flip-flops R1 through R5 on the LT47 module. On all four types of send modules except the LT47, the clock counter consists of flip-flops C1 through C5. The LT47 module has an additional clock counter flip-flop, C0, which is used to generate the one-half unit in the stop bit of the 5-level/7.5 unit code. The control logic consists of flip-flops FRUN, FSTOP, and FLSP on all four modules.

The operation of all four types of send modules is somewhat similar; therefore, only the LT47 module (used with the 5-level/7.5 unit code) is described in figure 3-42. Figure 3-43 is a timing diagram showing the send signals during an output operation.

When the transmit data function is executed with the write direct instruction, the five information bits of the character are parallel shifted into the character register flip-flops. These flip-flops were reset at the end of the transmission cycle for the previous character.

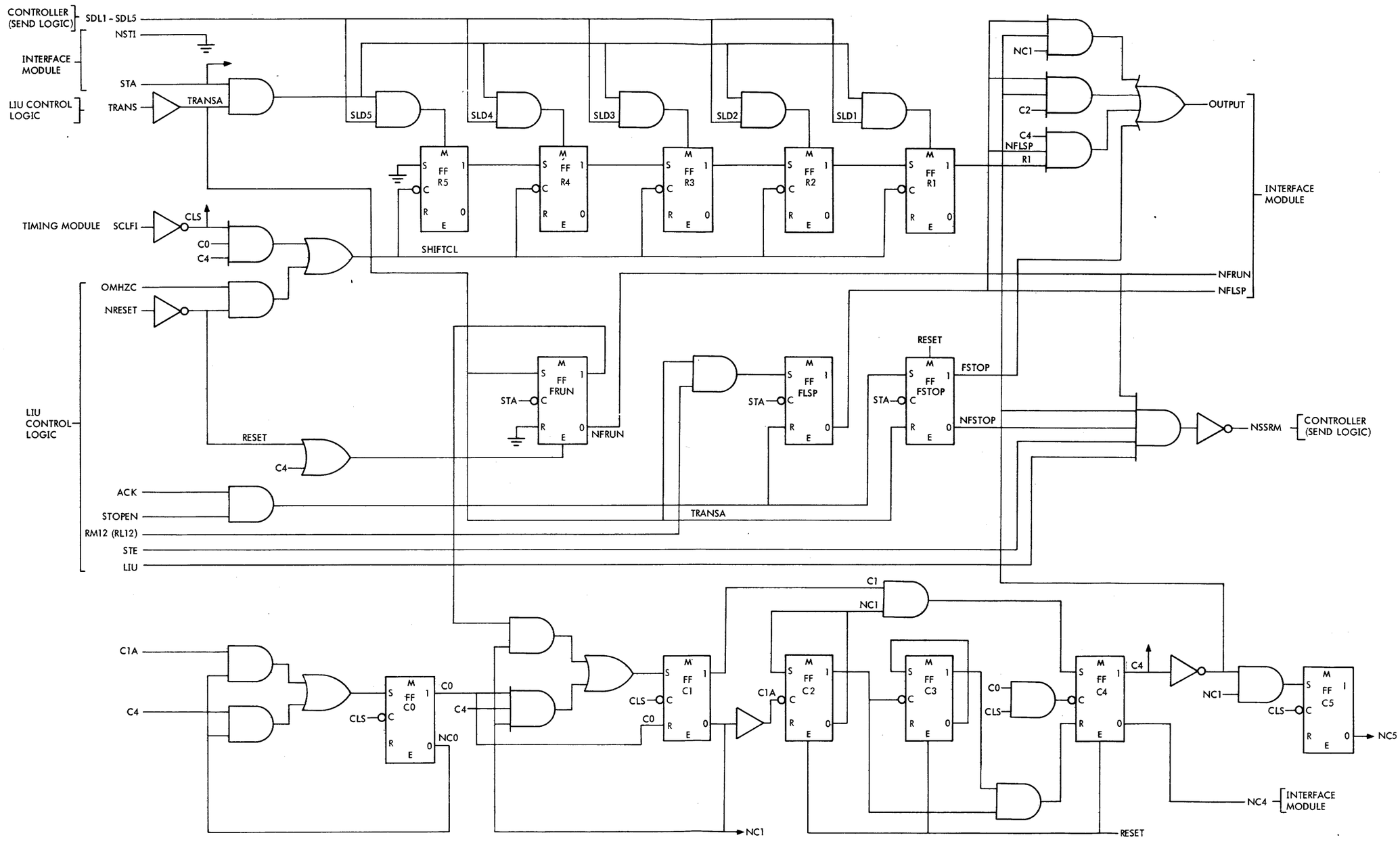


Figure 3-42. Send Module (LT47), Logic Diagram

901075A.342

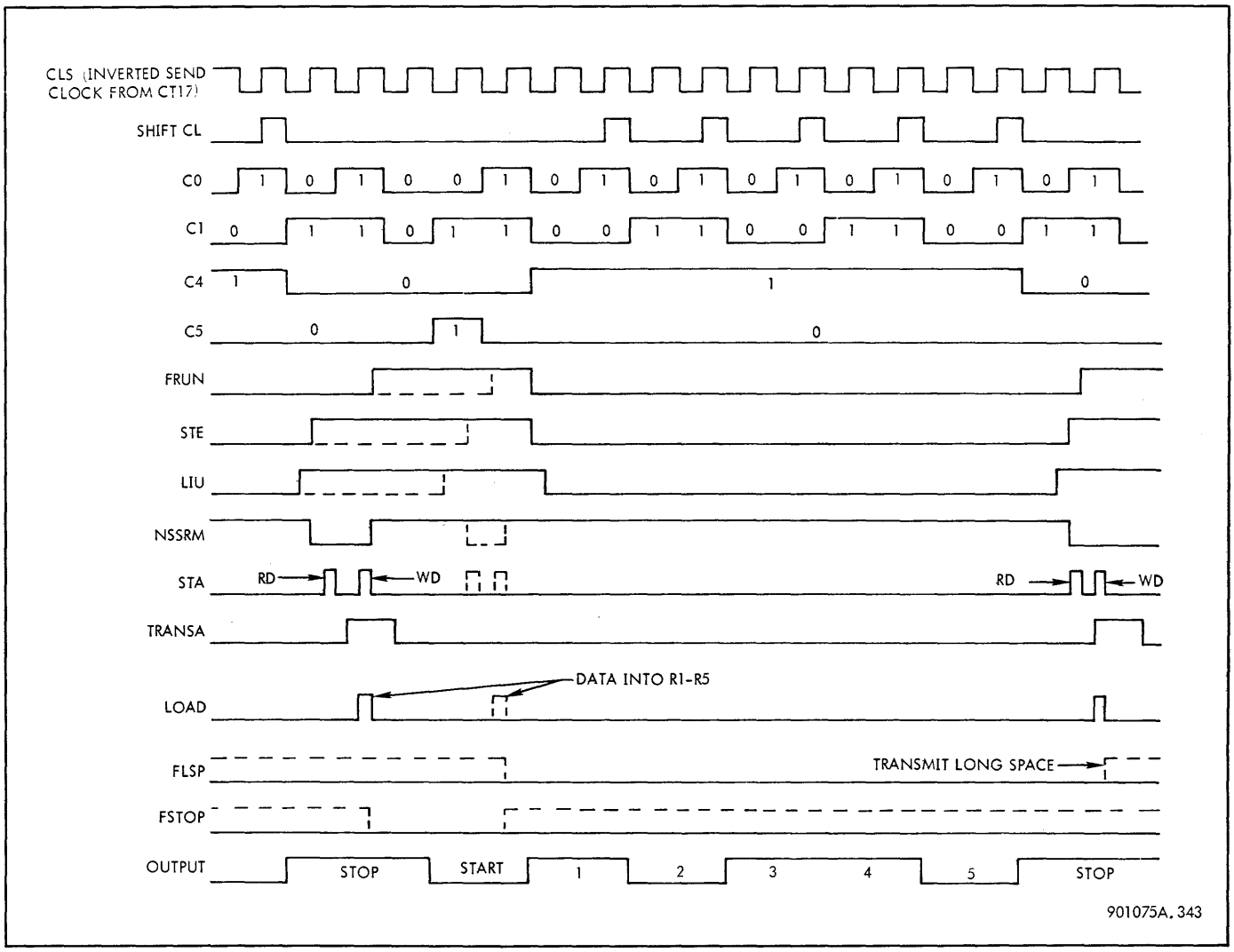


Figure 3-43. Send Module (LT47), Output Timing Diagram

The information present on lines SLD1 through SLD5 is applied to the mark inputs of flip-flops R1 through R5 and is strobed into the register when signal LOAD goes true.

M/R1 = SLD1 LOAD

.
.
.

M/R5 = SLD5 LOAD

Signal LOAD is generated on the send module when signals TRANSA and STA go true. Signal TRANSA is true when the transmit function has been decoded from the function code bits of the write direct instruction. Signal STE is gated with FSL on the interface module to generate STA. Signal STE is true when the operation applies to a send module, and FSL is the function strobe acknowledge signal.

LOAD = TRANS STA

TRANS = TRANSA

STA = STE FSL

The function strobe acknowledge signal is generated by the function strobe logic in the send section of the controller shortly after it receives the function strobe, and the signal is held true for one-half CLOCKA time. During this time, the information is set into the character register.

When signal FSL drops, signal STA also drops. At this time the run flip-flop (FRUN) is set, and the stop flip-

flop (FSTOP) is reset, because both of these flip-flops are controlled by signal TRANSA.

S/FRUN = TRANSA

C/FRUN = STA

R/FSTOP = TRANSA

C/FSTOP = STA

With the setting of flip-flop FRUN and the resetting of flip-flop FSTOP, the send module starts shifting the information serially from its output terminal under control of clock signal SHIFTCL.

SHIFTCL = C0 C4 CLS

CLS = SCLFI

SCLFI = TRANSMIT CLOCK (Format I)

Signals C0 and C4 are generated by the clock counter and are used to generate internal timing for the send module. A truth table for the counter follows.

C0	C1	C2	C3	C4	C5	
0	1	0	0	0	1	Start
1	1	0	0	0	0	
0	0	1	0	1	0	1 (Data Bits)
1	0	1	0	1	0	
0	1	1	0	1	0	2 (Data Bits)
1	1	1	0	1	0	
0	0	0	1	1	0	3 (Data Bits)
1	0	0	1	1	0	
0	1	0	1	1	0	4 (Data Bits)
1	1	0	1	1	0	
0	0	1	1	1	0	5 (Data Bits)
1	0	1	1	1	0	
0	1	1	1	0	0	Stop (1.5 Unit)
1	1	1	1	0	0	
0	0	0	0	0	0	
0	0	0	0	0	1	Long Space

When a complete character has been transmitted to the line, all of the counter flip-flops are in the zero state, which represents the transmission of the last half unit of the stop bit. The first half unit of the start bit is generated when flip-flops C1 and C5 are set.

S/C1 = NC1 FRUN + . . .

C/C1 = CLS

S/C5 = NC1 NC4

C/C5 = CLS

With succeeding clocks, CLS, the counter steps through the states shown in the truth table until it again reaches the last half unit of the stop bit. The counter equations follow.

S/C0 = NC0 C4 + NC0 C1A

C1A = (Inverter) NC1

R/C0 = .

C/C0 = CLS

S/C1 = NC1 FRUN + NC1 C0 C4

R/C1 = C0

C/C1 = CLS

S/C2 = NC2

R/C2 = .

C/C2 = C1A

S/C3 = NC3

R/C3 = .

C/C3 = C2

S/C4 = C1 NC2

R/C4 = C2 C3

C/C4 = C0 CLS

S/C5 = NC1 NC4A

NC4A = (Inverter) C4

R/C5 = .

C/C5 = CLS

The output terminal on the send module is labeled OUTPUT. The equation for OUTPUT follows.

OUTPUT = NFLSP NC1 NC4A + NFLSP C2 NC4A + NFLSP C4 R1 + FSTOP

Signal NFLSP is the reset output of the long space flip-flop and is true when a long space is not being transmitted. Flip-flop FSTOP is false at this time, because the send module is in the run mode (flip-flop FRUN set).

Signal OUTPUT is false with the counter flip-flops C0 through C5 in the first two count positions to generate the START bit. When the output is false, it represents either a start bit or a zero bit of the character. When the counter steps to the first part of the data-bit 1 position, C4 going true causes one term of the OUTPUT equation (NFLSP C4 R1) to go true. At this time, the contents of R1, which is the first data bit (least significant bit) is transmitted to the line. Flip-flop FRUN, which was set by the fall of STA, is now reset by C4 by way of the ERASE input.

E/RUN = C4 + . . .

The reset of FRUN disables the send service request line NSSRM. Signal C4 activates the shift clock signal SHIFTCL.

SHIFTCL = C0 C4 CLS

Since signal C0 is included in the SHIFTCL equation, one shift clock is generated for every two steps of the counter. A total of five shift clocks is required to shift the five data bits from the character register to the line. Flip-flop R5 is reset (zero) as the data in R5 is transferred to R4. Flip-flop R5 is reset because its set input is grounded, and its reset is floating (true). The zero is shifted through to R1, which leaves all five flip-flops (R1 through R5) in the zero state in preparation for the input of a new character.

At the time that the fifth SHIFTL occurs, flip-flop C4 is reset. This causes the term (NFLSP C4 R1) in the OUTPUT equation to go false and the term (NFLSP C2 NC4A) in the same equation to go true. The output now goes true regardless of its previous status. (A true output represents either a STOP bit or a one bit of the data.)

When C4 resets, the gate that generates the send service request is armed in preparation for the send scanner addressing. The send service request logic equation follows.

NSSRM = (Inverter) (NFRUN NC4A LIUL STE NSTOP) (C5 + NFLSP)

Signal NSSRM is normally high until a service request is made, at which time it drops. The conditions, therefore, which enable the send service request are:

a. The FRUN flip-flop is reset, which indicates that the previous service request has been processed.

b. Flip-flop C4 is reset, which indicates that either the START or the STOP bit is being transmitted.

c. Signal LIUL is true, which indicates that the LIU in which this send module is located has been addressed.

d. Signal STE is true, which indicates that the send module number has been raised.

e. Signal NSTOP is true, which indicates that the send module is not turned off.

f. Signal NFLSP is true when flip-flop C4 resets, which indicates that the character being transmitted is data and is not a long space.

g. Signal NFLSP is false when flip-flop C4 resets, which indicates that the character being transmitted is a long space, and that a service request is generated when flip-flop C5 sets.

In response to the send service request, the CPU issues a read direct instruction with the output response function. This is done so that the address of the send module generating the request can be read into the CPU. The read direct instruction is followed by a write direct instruction in which the transmit function to be executed is encoded. In both cases, signal STA is raised. At the fall of signal STA during the write direct instruction, flip-flop FRUN is set, and flip-flop ESTOP is reset. (Flip-flop FSTOP may or may not have set previously.) The service request is concluded when flip-flop FRUN is set. Dropping the service request signal enables the send scanner to start running again in search of other send modules that are waiting to request service.

The set gate to flip-flop C1 is again enabled, with flip-flop FRUN in the set state and with the counter flip-flops (C0 through C5) in the reset state. This permits the count sequence to start over.

If a long space character is to be transmitted, the long space flip-flop, FLSP, is set with the transmit long space function of a write direct instruction.

S/FLSP = TRANSA RM12

C/FLSP = STA

With flip-flop FLSP set and flip-flop FSTOP reset, signal OUTPUT is false. This represents a space condition.

The term of the logic equation NSSRM, that is designated C5 + NFLSP, is raised when flip-flop C5 is set. Since this occurs at the conclusion of the STOP bit, the service request signal is not raised until one full character of space has been transmitted. To send a second long space character following transmission of the previous one, the CPU responds to the service request by executing a second write direct instruction encoding the transmit long space function.

If it is desired to start transmitting data characters again or to end the transmission completely, a stop transmission function is executed by the CPU. This function resets the long space flip-flop and sets the stop flip-flop.

R/FLSP = ACK STOPEN

C/FLSP = STA

S/FSTOP = ACK STOPEN

C/FSTOP = STA

When flip-flop FLSP is reset and when flip-flop FSTOP is set, signal OUTPUT is true and represents a mark condition. Thus, when the transmission of a character is ended, the line goes into a marking (all ones) condition. The transmit data function can now be executed, if desired.

Three signals are sent to the interface module to generate the condition code. (The code signifies that the send module is not in the process of transmitting a character to the line.) These signals are NFRUN, NFLSP, and NC4FF which is the reset output of flip-flop C4. Each send module has a master reset signal designated NRESET, which is inverted to produce RESET on the send module. Signal RESET is used to initialize the logic when a system reset occurs. Signal NRESET is high in its quiescent state. When a reset occurs, that is, when signal RESET is true, the following events occur:

a. Flip-flop FRUN, if set, is reset (E/FRUN = RESET +...).

b. Flip-flop FLSP, if set, is reset (E/FLSP = RESET +...).

c. Flip-flop FSTOP, if reset, is set (M/FSTOP = RESET).

d. Flip-flops C2, C3, and C4, if set, are reset (E/C2 = RESET, E/C3 = RESET, and E/C4 = RESET).

e. Flip-flops R1 through R5 are reset with a high speed shift clock (OMHZC) that is applied to the clock input of the flip-flops.

3-70 Timing Logic

Bit timing for the send and the receive modules is supplied by timing generator module CT17. A description of the five format timing groups is given in paragraph 1-2.

The frequency range of the timing modules (see table 1-4) is determined by the tuning range of the basic oscillator on the CT17 module (1 to 2.5 mHz) which limits the range of frequencies to a ratio of 1 to 2.5 and the number of countdown stages on the CT17, as determined by the wiring on the connector in which the CT17 module is inserted. Frequencies outside the specified range cannot be obtained without changing the wiring or changing the CT17 clock module.

All format I, II, and IV send and receive modules derive timing from the CT17 modules in slots 21B, 20B, and 22B respectively. The CT17 module in slot 23B provides timing for all format III send and receive modules located in LIU1 through LIU7. It also provides timing for the format III send and receive modules in LIU0 if the high speed option (slot 24B) is not being used. In this case, signal NFIIIFAST is true.

Signal NFIIIFAST is grounded when a CT17 module is inserted in slot 24B. If a CT17 module is installed in slot 24B, it provides the timing for all format III send and receive modules in LIU0 (only); otherwise, the CT17 module in slot 23B provides the timing. Figure 3-44 is a logic diagram of the timing generator.

The CT17 module contains a sine wave oscillator with an adjustable tuning range of 1 mHz to 2.5 mHz. The tuning element is a variable capacitor. To obtain the frequencies specified in table 1-4, however, a crystal is used to control the oscillator, and the capacitor is adjusted for maximum output at the feedback point (pin 29). The output of the oscillator (pin 12) is a logic level square wave. The designator of the square wave at this point depends on the format it is used with, that is, the slot in which the module has been inserted.

There are three binary ripple counters on the module: a seven stage, a five stage and a three stage. Each flip-flop of each counter has a separate output terminal connected to a separate pin on the module. The output terminals in figure 3-44 are labeled divide by 2, 4, 8, and so forth. The output at 2 is a square wave that is half the frequency of the input, and the output at 4 is a square wave that is half the frequency of that at 2, and so forth. The jumpers on each of the five connectors (slots) interconnect the various output terminals of one counter to the input terminal of the following counter. The jumper arrangement on each of the five connectors is different. In this manner, the required counter stages are connected by jumpers to achieve the desired frequency for that slot. Figure 3-44 shows the required jumper connections for each format as dashed lines. For example, the output of the oscillator (pin 12) is jumpered to the input of the first counter (pin 46) for all formats. The output of that counter, labeled divide by 128, is jumpered to the input (pin 30) on the next counter for all formats except format III fast. Since format III fast is the highest speed, it requires fewer countdown stages than the other formats and is, therefore, connected to the input gating for the resync logic, bypassing the second counter. Format II and format III slow are the next highest speeds. They are counted down by the first three flip-flops of the second (five flip-flop) counter and are connected to the resync logic by way of the divide by 8 output terminal. Format I is the slowest, and format IV is the next to the slowest. These are connected to the resync logic by way of jumpers connected to the divide by 32 and the divide by 16

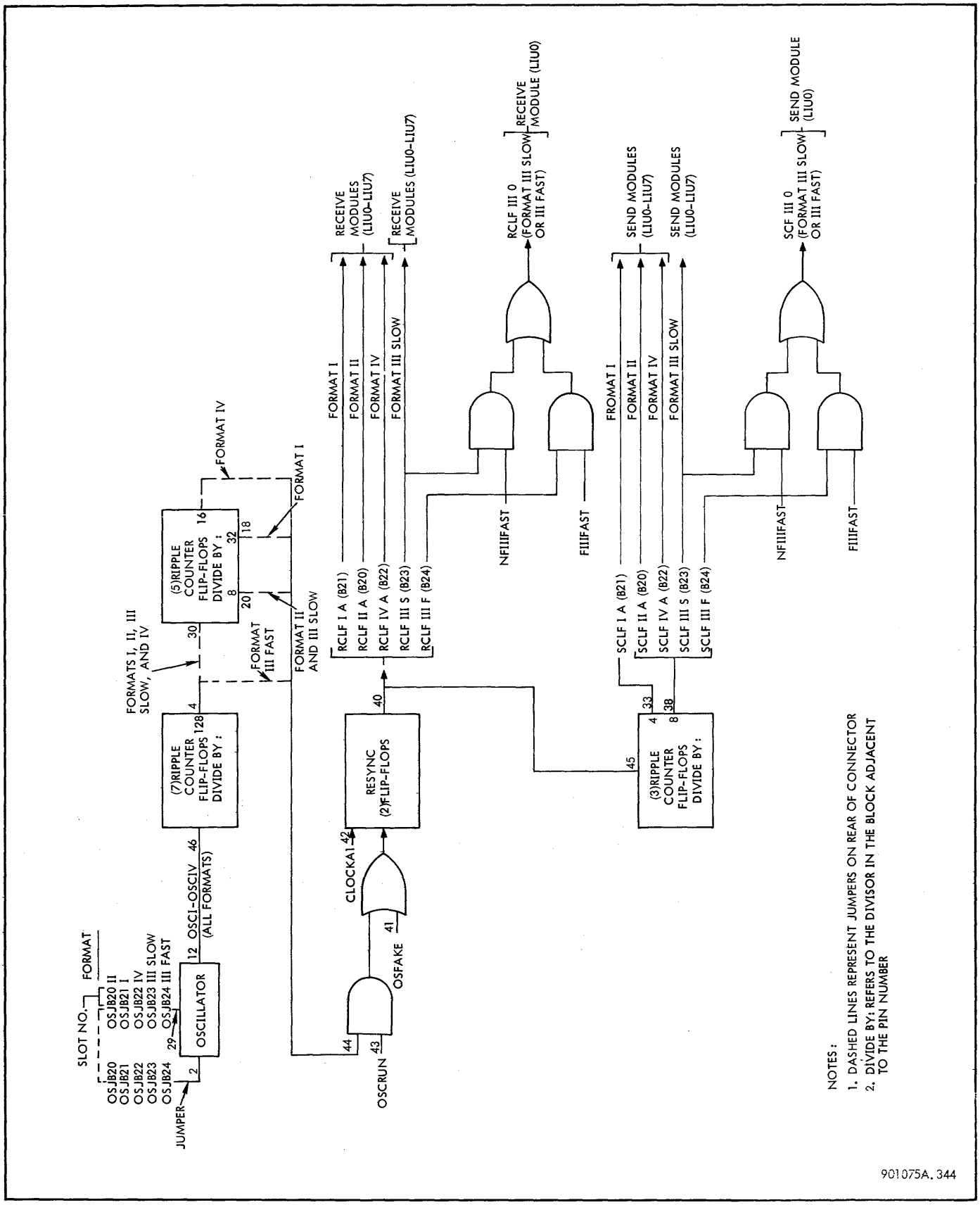


Figure 3-44. Timing Generator, Logic Diagram

outputs respectively. The purpose of the resync logic is to synchronize the format timing signals generated on the CT17 modules with the clock received from the MIOP (CL1). The clock signal used to clock the resync logic flip-flops is CLOCKA1.

CLOCKA1 = CLOCK

CLOCK = CL1R INI

Clock signal CL1R is CL1 through a cable receiver in the subcontroller, and signal INI is true when the controller is connected to the MIOP for service.

During offline operations, the resync logic is enabled by signal OSFAKE, derived from the PET. Signal OSFAKE is used for testing only. When operating online, the resync logic is enabled by the applicable format signal gated with signal OSRUN. Signal OSRUN is true when signal INI is true.

The timing signals for the receive modules are taken from the output of the resync logic (pin 40) of the applicable connector. The timing signals for the send modules, however, are counted down by a third counter consisting of three flip-flops. The difference between the send and the receive timing signals is compensated by logic on the send and the receive modules. Each format timing signal controls all of the send or receive modules, as applicable, in the LIU specified on the diagram (figure 3-44).

3-71 GLOSSARY

The glossary consists of a description of signals that interface with the COC, a description of the signals and logic elements that are internal to the COC (including the subcontroller and LIU), and a definition of some of the terms associated with communications equipment.

3-72 INTERFACE SIGNAL DESCRIPTION

A description of the signals present at the MIOP, the DIO, and the data set (line) interfaces is given in tables 3-1, 3-2, and 3-3, respectively. The DIO interface signal description (table 3-2) also includes a description of the external interrupt signals.

3-73 COC INTERNAL SIGNAL DESCRIPTION

A description of the signals internal to the controller, including the subcontroller, are presented in table 3-4. A description of the signals internal to the LIU is presented in table 3-5.

3-74 GLOSSARY OF COMMUNICATIONS TERMS

A partial list of communications terms applicable to the COC is presented in table 3-6.

Table 3-1. COC/MIOP Interface Signals

Signal	Source	Description
AIO	MIOP	Acknowledge I/O interrupt function indicator
ASC	MIOP	Acknowledge service call function indicator
AVI	DC	Available input priority signal routed sequentially through each controller from highest to lowest priority. This signal, when received by a controller, indicates that all higher priority controllers have not accepted the function indicator
AVO	DC	Available output priority signal which is generated when a function is not accepted. AVO becomes AVI into the next lower priority controller
CL1	MIOP	1-MHz clock (square wave)
DA0-DA7	MIOP/DC	Bidirectional data lines between MIOP and DC
DAP	MIOP/DC	Bidirectional ODD parity line between MIOP and DC
DOR	DC	Data/order request. If DOR is true, information being transferred is an order. If DOR is false, information being transferred is data. Part of condition code during an instruction
ED	MIOP/DC	Bidirectional end data line. Indicates last data or order byte is being transmitted

(Continued)

Table 3-1. COC/MIOP Interface Signals (Cont.)

Signal	Source	Description
ES	MIOP	End service line. Indicates last byte of information during service cycle is being transmitted
FR0-FR7	DC	Function response lines. Status information is applied to these lines during SIO, HIO, TIO, and TDV functions. The device number is applied during AIO and ASC functions
FS	MIOP	Function strobe. Indicates that the function indicator lines are stable
FSL	DC	Function strobe acknowledge. Indicates that the function response lines, condition code lines, and so forth may be strobed by the MIOP
HIO	MIOP	Halt I/O function indicator
IC	DC	Interrupt call. The controller drives this line to initiate an interrupt call
IOR	DC	Input/output request. The device controller drives this line high when information is being transferred from the MIOP to the DC. The DC holds it low when information is being transferred from the DC to the MIOP. Part of condition code during instructions
RS	DC	Request strobe. Indicates that associated signal lines are stable
RSA	MIOP	Request strobe acknowledge. Indicates that RS may be dropped
RST	MIOP	I/O reset line. Causes controller to initialize
SC	DC	Service call line driven by DC when it wants service
SIO	MIOP	Start I/O function indicator
TDV	MIOP	Test device function indicator
TIO	MIOP	Test I/O function indicator

Table 3-2. COC/DIO Interface Signals

Signal	Source	Description
A00-A15	CPU	Address lines. Encoded on these lines during a RD/WD instruction are the mode, COC address (for the DIO interface), and the function. These signals are derived from the effective address field of the RD/WD instruction
CC3, CC4	COC	Condition code lines for Sigma 5 or 7 CPU. These lines are the overflow and the carry for the Sigma 2 CPU. The COC places the status information on these two lines in response to an instruction
DB16-DB32	COC/CPU	Data lines. Lines DB16-DB23 are unidirectional, transmitting the character from the CPU to the COC during a WD instruction. Lines DB24 and DB25 are used during testing only; DB24 is unidirectional and DB25 is bidirectional. Lines DB26-DB32 transmit the line number and are bidirectional

(Continued)

Table 3-2. COC/DIO Interface Signals (Cont.)

Signal	Source	Description
RFS	CPU	Function strobe. Generated by the CPU to allow the addressed device to read the associated lines during an instruction
RFSA	COC	Function strobe acknowledge. Generated by the COC in response to a function strobe
RSTR	CPU	Reset I/O. Raised by the CPU in response to manual setting of applicable control panel switches or during a power on/power off sequence
RWD	CPU	Read direct/write direct selection line. The CPU controls this line to enable the COC to distinguish between a write direct and a read direct instruction. The selection line is driven high for a write direct and is held low for a read direct instruction
RINT	COC	Receive interrupt. Generated by the COC when it has a character assembled and is ready to input via the MIOP
RINTACK	CPU	Receive interrupt acknowledge. Generated by the CPU in response to signal RINT
SINT	COC	Send interrupt. Generated by the COC when it has transmitted a character and is ready to transmit the next one
SINTACK	CPU	Send interrupt acknowledge. Generated by the CPU in response to signal SINT. Used only for test purposes

Table 3-3. COC/Data Set Interface Signals

Signal	Source	Description
INPUT	Data set	Input character or long space from the line
OUTPUT	COC	Output character or long space to the line
Clear to send (CTS)	Data set	The clear to send signal (CTS) is generated by the data set to indicate that it is prepared to receive data from the COC in response to a request to send from the COC
Data set ready	Data set	The data set ready signal is generated by the data set when it is ready to operate This signal is in response to data terminal ready from the COC
Data terminal ready	COC	The data terminal ready signal is generated by the COC and is normally held true. The data set disconnects when it goes false for a duration of approximately 50 ms
Request to send (RTS)	COC	Request by COC for clear to send signal from the data set. It is normally true in full duplex, and is switched on prior to transmission in half duplex

(Continued)

Table 3-4. COC Controller and Subcontroller Signals

Signal	Description
A00R-A11R	DIO address line receiver outputs. Designates mode number (always 3) and COC address
A12R-A15R	DIO address line receiver outputs. Designates function DIO (read or write direct). Requires COC to perform
AIOC	Acknowledge I/O interrupt. The period during which the controller must place status on the data lines
AIOR	Receiver output. Acknowledge I/O interrupt from MIOP
ASCB	Acknowledge service call
ASCM	Acknowledge service call priority determinant
ASCR	Acknowledge service call from MIOP
AVIR	Priority signal available input
AVOD	Priority signal available output
B2	Byte two. Sets as second byte is transferred to MIOP
BSYC	Busy signal
CB2	Clock for byte two flip-flop
CC3D	Driver input, condition code 3
CC4D	Driver input, condition code 4
CL1R	1-mHz clock from MIOP
CLOCK	Buffered CL1 clock
CLOCKA	Buffered CLOCK
CLOCKA1	Buffered CLOCK
COC	Character Oriented Communications
COC1	DIO controller address successfully compared with switches and proper mode (mode 3) received
COC2	Buffered COC1
COCFS1	COC1 gated with FS1 (DIO function strobe)
COCL0-COCL7	Buffered COC1 fed to LIU0 through LIU7
COC0	PET DIO address coincidence signal
COSA1	LSB (bit 1) of output channel line number addressing

(Continued)

Table 3-4. COC Controller and Subcontroller Signals (Cont.)

Signal	Description
COSA1L0-COSA1L7	Buffered COSA1 fed to LIU0 through LIU7, respectively
COSA2	Bit 2 of output channel line number addressing
COSA2L0-COSA2L7	Buffered COSA2 fed to LIU0 through LIU7, respectively
COSA3	Bit 3 output channel line number addressing
COSA3L0-COSA3L7	Buffered COSA3 fed to LIU0 through LIU7, respectively
CLS	Low priority service call
CSL1	NFSC delayed
DA0C-DA7C	Input to data line buffers DA0D through DA7D, respectively
DA0D-DA7D	Input to data line 0 through 7 line drivers
DA0R-DA7R	Output of data line receivers from MIOP
DB16R-DB23R	Output of data line receivers 16 through 23 (DIO). Designates output character to be transmitted
DB24R-DB25R	Output of DIO data lines 24 and 25. Used only for testing
DB25D-DB31D	Input to DIO data lines 25 through 31 cable drivers. Line number from send scanner
DB26R-DB31R	Output of DIO data lines receivers 26 through 31. Designates line number to be selected
DB26S-DB31S	Buffered DB26R through DB31R. Also gated with INI
DCA	Subcontroller address successfully compared with address switches
DORD	Data/order to MIOP during service cycle; condition code (CC1) during instruction
ECD	PET control signal. Gates TC4 through TC10 to SDB lines
EDD	End data to MIOP (cable driver input)
EDR	End data from MIOP (cable receiver output)
EDTA	PET control signal. Gates received data to SDB lines
ERDD	Gating signal that gates the send scanner address on DIO data lines via DB25D through DB31D in response to a read direct instruction
ERIT	Not order in, not order out, and not terminal order
ESRFSC	End data and service connect
EWDD	COCFS1 and INI. Gates DIO data from receivers DB16R through DB23R on to SDB lines

(Continued)

Table 3-4. COC Controller and Subcontroller Signals (Cont.)

Signal	Description
FA	Flip-flop used in conjunction with flip-flop FB to control the timing of the send interrupt (SINT)
FB	Flip-flop used in conjunction with flip-flop FA to control the timing of the send interrupt (SINT)
FCOC	Flip-flop that is set by COC1 and CLOCK. Remembers COC1 coincidence
FIIIFAST	Term designating that high speed feature is installed
FR0D-FR7D	Function response line cable driver inputs
FS1	Derivation of DIO function strobe
FSC	Service connect flip-flop
FSCX	Service connect flip-flop for PET use
FSCZ	FSC or FSCX
FSD	Function strobe delayed
FSDC	Buffered FSD with small additional delay
FSDD	Buffered FSDC with small additional delay
FSL	COCFS1 timed by FX, FY, and FZ
FSL0-FSL7	Buffered FSL applied to LIU0 through LIU7, respectively
FLSD	Function strobe acknowledge cable driver input
FSR	Function strobe receiver output from MIOP
FSRC	FSR and not FSC
FSSS63	Flip-flop used with high speed option to simulate last service to line 63
FSW	PET function strobe (DIO)
FX	Flip-flop used in conjunction with FY, FZ, and CLOCK to time FSL
FY	Flip-flop. See FX
FYS	Set term for flip-flop FY
FZ	Flip-flop. See FX
HIOR	Halt I/O function indicator cable receiver output from MIOP
HPID	High priority interrupt cable driver input

(Continued)

Table 3-4. COC Controller and Subcontroller Signals (Cont.)

Signal	Description
HPIL	High priority interrupt latch
HPIR	High priority interrupt cable receiver output from MIOP
HPSD	High priority service cable driver input
HPSL	High priority service latch
HPSR	High priority service cable receiver output
ICD	Interrupt call cable driver input
IIOPD	PET control signal. Gates data line outputs to PET
INC	Inhibits new service calls
INI	Grounds MIOP/controller interface lines when controller is not connected to the MIOP (offline)
IORD	Input/output request cable driver input. Used in conjunction with DOR to define the type of service cycle. Also used for condition code (CC2) during I/O instructions
IRSC	PET control signal. Gates receive scanner outputs to PET
ISDB	PET control signal. Gates SDB lines to PET
ISSC	PET control signal. Gates send scanner outputs to PET
LIH	High priority interrupt latch
LIL	Low priority interrupt latch
LIU	Line interface unit
LIUL0-LIUL7	LIU address selection signal
LSH	High priority service latch
LSL	Low priority service latch
OIN	Order in flip-flop
OINS	Set term for OIN flip-flop
OMHZCL0-OMHZCL7	Buffered CLOCKA applied to LIU0 through LIU7, respectively
OOUT	Order out flip-flop
OOUTCL	Clock for OOUT flip-flop
OSCI	Output of CT17 crystal oscillator (format I)
OSCI12	OSC17 counted down (format I)

(Continued)

Table 3-4. COC Controller and Subcontroller Signals (Cont.)

Signal	Description
OSC17	OSCI counted down (format I)
OSCII	Output of crystal oscillator (format II)
OSCII10	OSCII7 counted down (format II)
OSCII7	OSCII counted down (format II)
OSCIIIF	Output of crystal oscillator (format III fast)
OSCIIIF7	OSCIIF counted down (format III fast)
OSCIIIS	Output of crystal oscillator (format III slow)
OSCIIIS10	OSCIIIS7 counted down (format III slow)
OSCIIIS7	OSCIIIS counted down (format III slow)
OSCIV	Output of crystal oscillator (format IV)
OSCIV11	OSCIV7 counted down (format IV)
OSCIV7	OSCIV counted down (format IV)
OSCFAKE	PET clock. Replaces OSCI through OSCIV
OSCRUN	Gates CT17 oscillator off when OSCFAKE is to be used (PET)
OSCSTEP	PET single step clock. Feeds OSCFAKE if FSW is false
OSJB20-OSJB24	Jumper on CT17 card to make oscillator crystal controlled
PETCNTR	PET control signal
RCLFI0-RCLFI7	Buffered RCLFIA. Applied to LIU0 through LIU7, respectively
RCLFIA	Receiver clock (format I)
RCLFII0-RCLFII7	Buffered RCLFIIA. Applied to LIU0 through LIU7, respectively
RCLFIIA	Receiver clock (format II)
RCLFIII0-RCLFIII7	Buffered RCLFIIIF or RCLFIIIS. Applied to LIU0 through LIU7, respectively
RESET I0-RESET I9	Series of buffers controlled by RESET
RESET L0-RESET L7	Buffered RESET I9. Applied to LIU0 through LIU7, respectively
RFSAD	DIO function strobe acknowledge cable driver input
RFSR	DIO function strobe acknowledge receiver output
RINT	Receive interrupt (external) to CPU

(Continued)

Table 3-4. COC Controller and Subcontroller Signals (Cont.)

Signal	Description
RINTACK	Receive interrupt acknowledge from CPU
RINTBLK	Flip-flop. Receive interrupt block. NRINTBLK enables RINT
RINTBLKS	Set term for RINTBLK flip-flop
RINTL	Receive interrupt latch
RL12L0-RL12L7	Buffered DIO address line A12S applied to LIU0 through LIU7, respectively. Determines function to be performed
RL14L0-RL14L7	Buffered A14S applied to LIU0 through LIU7, respectively
RL15L0-RL15L7	Buffered A15S applied to LIU0 through LIU7, respectively
RLSP	Received long space pulse. Buffered RLSP1 through RLSP7
RLSP1-RLSP7	RLSP from LIU0 through LIU7, respectively
RS1-RS6	Receive scanner flip-flop outputs
RSAL1L0-RSAL1L7	Buffered RS1 applied to LIU0 through LIU7, respectively (Receiver addressing)
RSAL2L0-RSAL2L7	Buffered RS2 applied to LIU0 through LIU7, respectively (Receiver addressing)
RSAL3L0-RSAL3L7	Buffered RS3 applied to LIU0 through LIU7, respectively (Receiver addressing)
RSAR	Request strobe acknowledge cable receiver output from MIOP
RSARC	RSAR or FSCL
RSARCX	Flip-flop. PET equivalent to RSARC
RSDA	Request strobe
RSDB	Buffered RSDA
RSDC	Buffered RSDA
RSDD	Request strobe cable driver input (buffered RSDC)
RSEL0-RSEL7	Most significant three bits of receive scanner. Used to select LIU in which desired receiver is located. Used in conjunction with RSAL1 through RSAL3 to address receiver module. Applied to LIU0 through LIU7, respectively
RSR	Receive service request
NRSRM0-NRSRM7	Receive service request inverted from LIU0 through LIU7, respectively
RSRUN	Receive scanner enable signal
RSS63	Receive scanner set to 63. Used in conjunction with high speed option to give preferred service to channels 0 through 7

(Continued)

Table 3-4. COC Controller and Subcontroller Signals (Cont.)

Signal	Description
RSTR	I/O reset. Received from MIOP
RWDR	Read/write direct cable receiver output (from DIO)
SCD	Low priority service call cable driver input
SCLFI0-SCLFI7	Buffered SCLFIA. Applied to LIU0 through LIU7, respectively
SCLFIA	Send clock (format I)
SCLFII0-SCLFII7	Buffered SCLFIIA. Applied to LIU0 through LIU7, respectively
SCLFIIA	Send clock (format II)
SCLFIII0-SCLFIII7	Buffered SCLFIIIF or SCLFIIIS. Applied to LIU0 through LIU7, respectively
SCLFIIIF	Send clock (format III fast)
SCLFIIIS	Send clock (format III slow)
SCLFIV0-SCLFIV7	Buffered SCLFIVA. Applied to LIU0 through LIU7, respectively
SCOCE	Device controller DIO address successfully compared with address switches and proper mode received
SCON	Service call on. Used when operating with PET
SCOSSL0-SCOSSL7	Buffered SCOSS. Applied to LIU0 through LIU7, respectively
SCOSS	Buffered DIO address line A13S if COC2 is true, if not, SCOSS equals NCOC2
SCOSS4-SCOSS6	Most significant three bits of send module addressing. Selects LIU that contains the desired send module. These signals are decoded to generate LIUL0 through LIUL7
SDB1-SDB8	Send data bits 1 through 8. Applied by DIO data lines DB16 through DB23 or by PET
SDL1L0-SDL1L7	Buffered SDB1 applied to LIU0 through LIU7, respectively
SDL2L0-SDL2L7	Buffered SDB2 applied to LIU0 through LIU7, respectively
SDL3L0-SDL3L7	Buffered SDB3 applied to LIU0 through LIU7, respectively
SDL4L0-SDL4L7	Buffered SDB4 applied to LIU0 through LIU7, respectively
SDL5L0-SDL5L7	Buffered SDB5 applied to LIU0 through LIU7, respectively
SDL6L0-SDL6L7	Buffered SDB6 applied to LIU0 through LIU7, respectively
SDL7L0-SDL7L7	Buffered SDB7 applied to LIU0 through LIU7, respectively
SDL8L0-SDL8L7	Buffered SDB8 applied to LIU0 through LIU7, respectively
SERV	Service flip-flop. Set by RSR

(Continued)

Table 3-4. COC Controller and Subcontroller Signals (Cont.)

Signal	Description
SERVFAST	Fast service flip-flop. Set by SERV and FIIIFAST. Used in conjunction with high speed option
SINT	Send interrupt (external) to the CPU
SIOR	Start I/O cable receiver output (from MIOP)
SS1-SS6	Send scanner outputs
SSR	Send service request
SSRA	High speed feature send service request to provide preferred service to channels 0 through 7
SSRAGO	SSRA enable signal
SSRASTOP	SSRA disable signal
SSRAP	Mark set for FSSS63 flip-flop. Equals SSR and SSRA and COC1 and NFZ
NSSRL0-NSSRL7	Send service request inverted from LIU0 through LIU7, respectively
SSRUN	Send scanner enable signal
SSS63	Send scanner set to 63. Used in conjunction with high speed option to give preferred service to channels 0 through 7
ST1	Status bit 1. Drives CC3 (carry) cable driver
ST2	Status bit 2. Drives CC4 (overflow) cable driver
NSTARESET	Reset term composed of NSTART and NSTARTS
START	Flip-flop denoting busy condition
STARTS	Set term for the START flip-flop
NSTM1G0-NSTM1G7	Status bit 1 inverted from LIU0 through LIU7, respectively
NSTM2G0-NSTM2G7	Status bit 2 inverted from LIU0 through LIU7, respectively
NSTOK	Controls condition of IORD in response to an SIO
SUSR	SSR and FCOC. Used in derivation of PET DIO address coincidence signal COC0
SWA0-SWA7	Address switches that are compared with DA0 through DA7, respectively
SYNC	Test signal that equals COCFS1, DB24, DB25, and RWDR
TC00-TC10	PET test signals
TD00-TD11	PET test signals
TDATA	Gates received data on to DA-lines

(Continued)

Table 3-4. COC Controller and Subcontroller Signals (Cont.)

Signal	Description
TDATA4	Gates received data bit 4 to DA4
TDATA5	Gates received data bit 5 to DA5
TDVR	Test device function indicator cable receiver output (MIOP)
TESTLATCH	Flip-flop used for test purposes
TI00-TI15	PET lamp drivers
TIOR	Test I/O function indicator cable receiver output (MIOP)
TO	Flip-flop, denotes terminal order
TPT1	Single step clock signal from PET
TSCAN	Gates line number on to DA-lines. (Line No. specified by scanner)
TSCAN4	Gates line number bit 4 to DA4
TSCAN5	Gates line number bit 5 to DA3
TSD1	Signal from switch on PET. Single steps RSARCX
TSD2	Signal RFSR from PET
TSH	(TIOR or SIOR or HIOR) DCA
TSSC	Signal from PET. Provides ground for TSSD
TSSD	Reset signal from PET
TSTAT	Drives DA3 and DA4 to indicate unusual end condition
TTD1-TTD2	PET drive signals
TTSH	TDVR, TIOR, SIOR or HIOR
UEND	Unusual end flip-flop
WANTI	Want interrupt flip-flop. (Interrupt request to MIOP via ICD)
WANTS	Want service flip-flop. (Service request to MIOP via SCD)

(Continued)

Table 3-5. LIU Signals

Signal	Description
ACK11, ACK12	Derivation of RL14 and RL15. It is true for the following functions: Turn receiver off, Stop transmit, Turn receiver on, Transmit data, and Transmit long space. ACK11 and ACK12 are identical signals derived in parallel for loading purposes
NC4FF0-NC4FF7	False output of counter flip-flop C4 in send module. Generated by send module in send-receive positions 0 through 7, respectively
CCRP	PET control signal
CELE	Control signal to PET
COCL	COC, DIO controller address successfully compared with address switches, and proper mode (mode 3) received
COSA1	LSB (bit 1) of output channel line number addressing
COSA2	Bit 2 of output channel line number addressing
COSA3	Bit 3 of output channel line number addressing
NELSP0-NELSP7	False output of long space flip-flop from send modules 0 through 7, respectively
FRON0-FRON7	Output of receiver on flip-flop from interface modules 0 through 7, respectively
FRUN0-FRUN7	False output of run flip-flop from send modules 0 through 7, respectively
FSL	Derivation of DIO function strobe
FSM	Buffered FSL
GOCC	PET control signal
INLOCK0-INLOCK7	Pull up voltage to enable reset. Supplied by interface modules 0 through 7 to their respective receivers
INPUT0-INPUT7	Received data from interface modules 0 through 7 to their respective receivers
LIU	Line interface unit
LIUL	Buffered LIU address selection signal
MTLRA0-MTLRA7	Jumper in CLOCKA signal path for receivers 0 through 7, respectively
MTLRB0-MTLRB7	Jumper in CLOCKB signal path for receivers 0 through 7, respectively
MTLRC0-MTLRC7	Jumper in CLOCKC signal path for receivers 0 through 7, respectively
MTLS0-MTLS7	Jumper in C4 flip-flop signal path for send modules 0 through 7, respectively
NLSP0-NLSP7	Not long space pulse from receivers 0 through 7 to their respective interface modules
OMHZCL	1-mHz clock from COC or PET

(Continued)

Table 3-5. LIU Signals (Cont.)

Signal	Description
OMHZCM	Buffered OMHZCL
OUTLOCK0-OUTLOCK7	Pull-up voltage to enable reset. Supplied by interface modules 0 through 7 to their respective send modules
OUTPUT0-OUTPUT7	Send data from send modules 0 through 7 to their respective interface modules
PCR	Pet control signal
RCLFI	Receive clock (format I)
RCLFII	Receive clock (format II)
RCLFIII	Receive clock (format III)
RCLFIV	Receive clock (format IV)
RCMFI	Buffered RCLFI
RCMFII	Buffered RCLFII
RCMFIII	Buffered RCLFIII
RCMFIV	Buffered RCLFIV
RDM1	Received data (bit 1) from receive module
RDM2	Received data (bit 2) from receive module
RDM3	Received data (bit 3) from receive module
RDM4	Received data (bit 4) from receive module
RDM5	Received data (bit 5) from receive module
RDM6	Received data (bit 6) from receive module
RDM7	Received data (bit 7) from receive module
RDM8	Received data (bit 8) from receive module
READL	Denotes receive data has been transferred to MIOP. Resets receive service request
READM	Buffered READL
NRESET11	Inverted RESETM
NRESET12	Inverted RESETIM or RESETM
RESETIM	Buffered RESETL
RESETL	Reset term from COC or PET

(Continued)

Table 3-5. LIU Signals (Cont.)

Signal	Description
RESETM	Terminated RESETL
RL12, RL14, and RL15	Buffered DIO address lines A12, A14, and A15, respectively. Determines function to be performed
RLSPM	Received long space pulse from receive module
RM12	Buffered RL12
RSAL1	Bit 1 of receiver address selection signal
RSAL2	Bit 2 of receiver address selection signal
RSAL3	Bit 3 of receiver address selection signal
NRSAM1	Inverted RSAL1
NRSAM2	Inverted RSAL2
NRSAM3	Inverted RSAL3
RSEL	LIU selection signal. Used with RSAL1 through RSAL3 and NRSAM1 through NRSAM3 to select the desired receiver
NRSRM	Inverted receiver service request from receive module
SCLFI	Send clock (format I)
SCLFII	Send clock (format II)
SCLFIII	Send clock (format III)
SCLFIV	Send clock (format IV)
SCOSS	Buffered DIO address line A13. Determines if function is to be performed by send or by receive module
SDL1-SDL8	Send data lines 1 through 8. Data to be transmitted from COC controller to send module
SRE0-SRE7	Receive modules 0 through 7 selection signal (by DIO write direct instruction)
NSRI0-NSRI7	Ground provided by receive modules 0 through 7 to their respective interface modules indicating receiver is installed
NSSRL	Buffered NSSRM
NSSRM	Inverted send module service request from send module
STA0-STA7	FSL gated with STE. Generated on interface modules and sent to their respective send modules
NSTAT	Inverted status enable signal. Equals COCL and LIUL and NRESET12

(Continued)

Table 3-5. LIU Signals (Cont.)

Signal	Description
NSTI0-NSTI7	Ground provided by send modules 0 through 7 to their respective interface modules indicating that the send module is installed
NSTM1	Inverted status bit 1 from interface module (CC3, carry)
NSTM2	Inverted status bit 2 from interface module (CC4, overflow)
STOPEN11	Stop enable term to send module (buffered RL14)
STOPEN12	Duplicates STOPEN11 for loading purposes
SUP	Shift up. Sets flip-flop FCASE. (SUP = RDM1 RDM2 NRDM3 RDM4 RDM5)
TDSCO11	Turn data set off. Function signal to interface module. (TDSO11 = RL14 RL15 NRL12)
TDSO12	Duplicates TDSO11 for loading purposes
TRANS11	Transmit function signal. (TRANS11 = RL15 NRL14)
TRANS12	Duplicates TRANS11 for loading purposes

Table 3-6. Glossary of Communications Terms

Term	Definition
Asynchronous	Pertaining to a lack of time coincidence in related sets of events; that is, the speed of operation of a subsystem is not related to the frequency of the system to which it is connected. Self-clocking in those event times is based on elapsed duration from some earlier event, not on an independent clocking signal
Baud	The number of code elements per second (bits per second)
Bit, start	One space unit which directly precedes data and denotes the first bit of a character
Bit, stop	One mark unit which directly follows data and denotes the last bit of a character
Carrier	Basic frequency of a signal on which an information signal is superimposed; one which contains no intrinsic intelligence until modulated by the information signal
Carrier, common	Company furnishing communication service to the public, and whose rates and service offerings are regulated by a government agency
Channel, four-wire	Two-way communications circuit using two paths, arranged so that communication in one direction is by one path and in the other direction by the other path
Channel, two-wire	Circuit using one line or channel for transmission in both directions, one direction at a time but reversible (half-duplex), one direction only (simplex), or both directions simultaneously (full-duplex) as in a four-wire circuit
Character	A group of bits that represents an alphanumeric or a special symbol

(Continued)

Table 3-6. Glossary of Communications Terms (Cont.)

Term	Definition
Clock	Timing source used to provide the basic sequence of pulses for the operation of a synchronous device
DATA-PHONE	A registered trademark of American Telephone and Telegraph, designating a family of devices used for data communications over the direct distance dialing telephone network
Data set	An interface unit that converts telephone line signals to digital signals and vice-versa, suitable for interfacing with a business machine
Duplex operation	In general communication, the operation of transmitting and receiving apparatus at one location in conjunction with associated transmitting and receiving apparatus at another location, with the processes of transmitting and receiving being done simultaneously
End-of-transmission	The receipt or the transmission of the last character at the end of a data message indicating the completion of a message
Format group	As applicable to the COC, each group pertains to a specific code and level and to the specific range of speeds applicable to that code and level
Full-duplex	Simultaneous transmission in both directions (see channel, four-wire)
Half-duplex	Capable of transmission in only one direction at a time but reversible
Line, private	Voice-grade channel leased from the telephone company for the exclusive use of a subscriber
Long space	The presence of a logical zero state for more than one full character time; that is, the start bit and data bits (zeros) extend through stop bit time
Mark	Representation of a logical one state
MODEM	An acronym meaning modulator-demodulator. Commonly used to describe the piece of equipment that functions at the terminals of a communication system between the digital or the audio frequency intelligence and the communication path over which it is to be sent. Another name for data set
Multiplex	Simulatenous transmission of several functions over one path
Simplex	Transmission in only one direction, irreversible
Space	Representation of a logical zero state
String	A contiguous set of characters, words, or other elements of information
Synchronous	Mode of operation in a device in which all events are controlled by equally spaced pulses from a clock
Terminal	A point in a communications network at which data can either enter or leave
Unit, terminal	Equipment on a communication channel that may be used for either input or output

SECTION IV
MAINTENANCE AND PARTS LIST

4-1 GENERAL

This section contains preventive maintenance procedures, adjustment procedures, and parts lists for the COC. The preventive maintenance procedures involve running the applicable diagnostic program to aid in the detection and the isolation of malfunctions of the COC. The adjustment procedures consist of adjustment of the timing signal. The parts lists are comprised of tables that specify the module complement and the cabling requirements of the COC.

4-2 PREVENTIVE MAINTENANCE

All documents listed on the Assembly of Maintenance Documents (147875 and 147876) should be available at the site. These documents should be complete and should accurately reflect the change level of the equipment. Field change record stickers should be applied to the equipment and should reflect the change level of the equipment.

4-3 EXTERNAL VISUAL INSPECTION

External surfaces of the equipment must be kept clean and dust-free. Doors and panels must close completely and be in reasonable alignment. The tops of cabinets must remain clear to allow free intake and exhaust of air.

4-4 INTERNAL VISUAL INSPECTION

The interiors of equipment must be free of wire cuttings, dust, spare parts, and other foreign matter. No clip leads or push-on jumpers should be in use during normal operation and all cables must be neatly dressed by clamps or by routing.

All chassis and frames must be properly bolted down, and all hardware must be in place. Air filters should be checked for cleanliness and should be replaced periodically.

4-5 PERFORMANCE TESTING

To maintain the COC properly the checkout procedures detailed in the following paragraphs should be performed every 90 days. When a malfunction occurs, the checkout procedures (error printout) provide a means of isolating the malfunction to a small area of the COC. Further checks may be made with an oscilloscope and/or other test instruments to isolate the malfunction to a specific replaceable module.

Follow the steps listed in this paragraph to check out the COC using the diagnostic program. If the COC is used with a Sigma 5 or 7, use diagnostic program number 704016; if the COC is used with a Sigma 2, use diagnostic program number 704014. The applicable diagnostic program manual is required, Sigma 2 Character Oriented Communications Controller Test, publication number 901168 or Sigma 5 and 7 Character Oriented Communications Controller Test, publication number 901156.

a. Determine the following (for use in step h).

(1) COC controller address related to DIO interface by checking setting of switches on switch module LT26 in slot 19B

(2) COC subcontroller address related to MIOP interface by checking setting of switches on switch module LT26 in slot 24C

(3) External interrupt location for the receive interrupt

(4) External interrupt location for the send interrupt

b. Place all installed send-receive channels back-to-back by connecting jumpers as shown in figure 4-1.

c. Turn power on (see applicable technical manual).

d. Place COC online by setting the toggle switch located on the LT25 module in slot 23C to the ON position (up).

Note

To provide a self-loading program, the object program is preceded by the applicable loader program. Perform step e or f as applicable.

e. Load Sigma 5 and 7 Relocatable Diagnostic Program Loader (XDS publication number 900972) when the COC is connected to a Sigma 5 or 7.

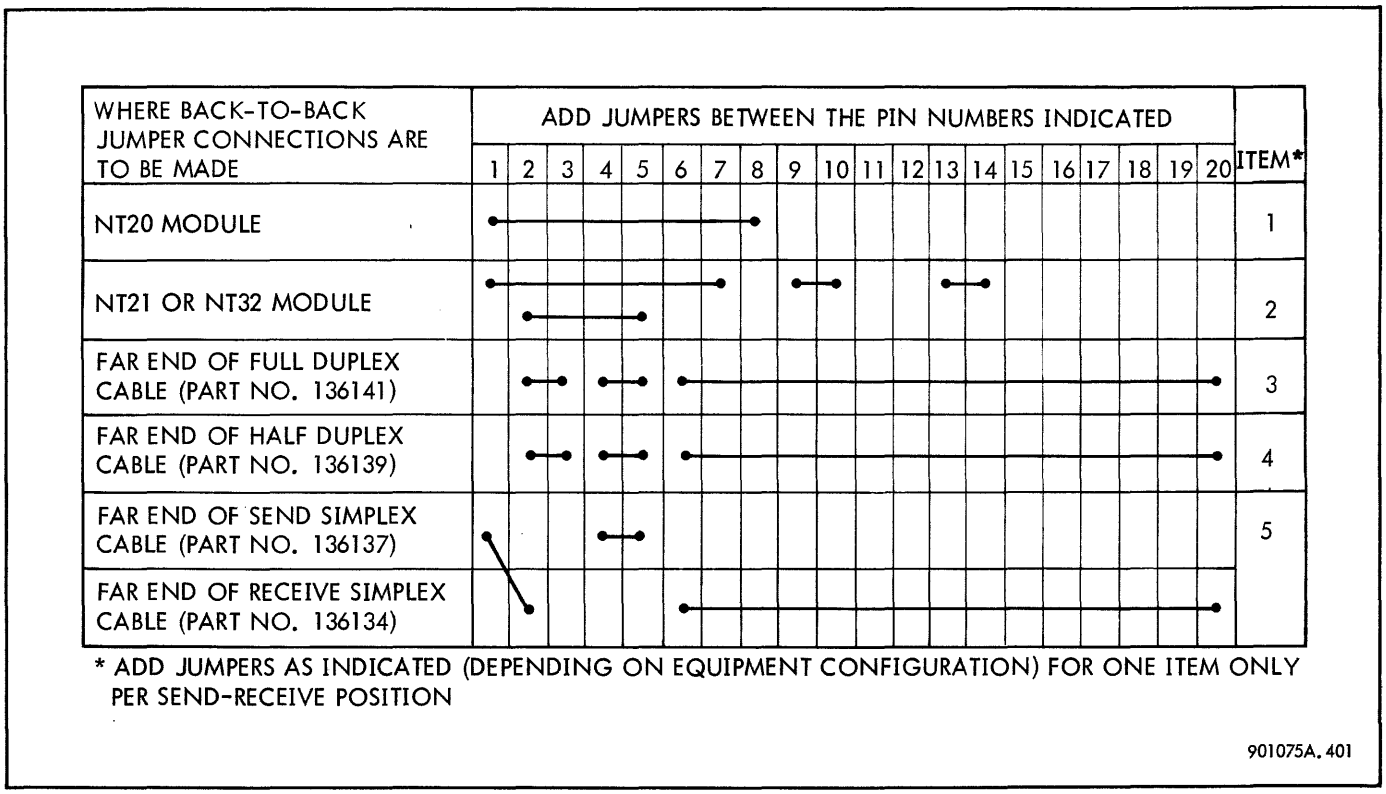


Figure 4-1. Back-to-Back Jumper Connections for Checkout

f. Load Sigma 2 Relocatable Diagnostic Program Loader (XDS publication number 901128) when the COC is connected to a Sigma 2.

g. Load diagnostic program (1 or 2 as applicable).

(1) Program number 704016 for Sigma 5 or 7. (See XDS publication number 900972 for procedures.)

(2) Program number 704014 for Sigma 2. (See XDS publication number 901128 for procedures.)

Note

A successful load is indicated by the printout: COMMUNICATIONS EQUIPMENT DIAGNOSTIC PROGRAM NO. 7040XX.

h. Declare controller environment. See Sigma 5 and 7 Character Oriented Communications Controller Test, Diagnostic Program Manual, publication number 901156, page 3-1 or Sigma 2 Character Oriented Communications Controller Test, Diagnostic Program Manual, publication number 901168, page 3-1, as applicable.

i. Declare SMD directive to cause all messages to be output on the keyboard printer.

j. Declare TEST, 1 per page 3-1 of applicable program manual (XDS publication number 901156 for Sigma 5 or 7; XDS publication number 901168 for Sigma 2).

k. Declare PAT, 3 per applicable program manual (page 3-8 of XDS publication number 901156 for Sigma 5 or 7; page 3-9 of XDS publication number 901168 for Sigma 2).

l. Declare Active Transmit Line (DAXL, d1, d2) for each installed send module (see figure 4-2) per page 3-1 of applicable program manual.

m. Declare Active Receive Line (DARL, d1, d2, d3) for each installed receive module (see figure 4-2) per page 3-2 of applicable program manual.

n. Declare ST1, 10 per page 3-3 of applicable program manual.

o. If the keyboard printer did not type out BEGIN TURN-AROUND TESTING, perform step p; if the keyboard

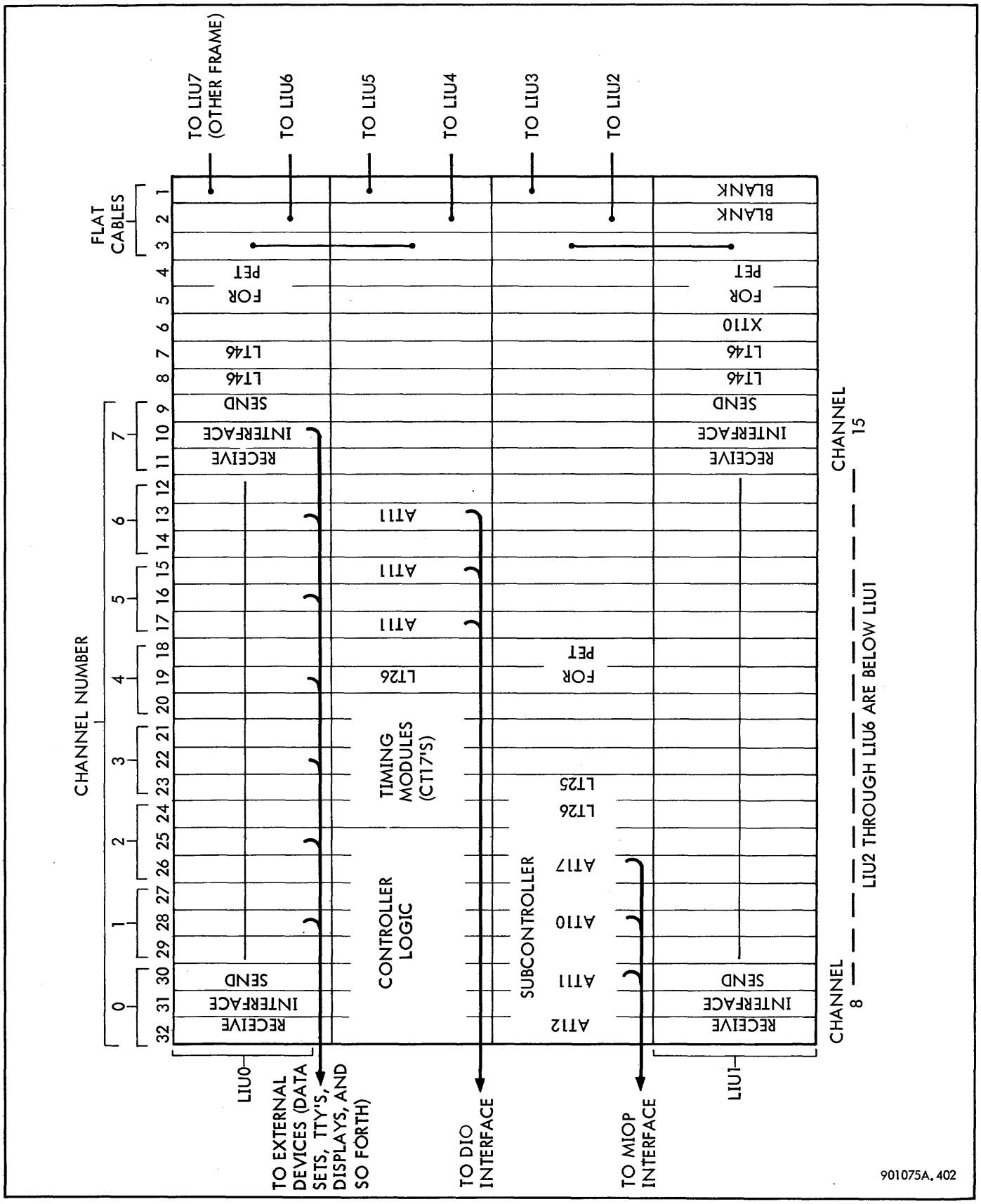


Figure 4-2. COC and LIU Cable Connections and Channel Locations

printer did print out the above message, skip step p and advance to step q.

p. Troubleshoot and repair according to the error printout. After the malfunction has been corrected, continue testing starting at step n.

q. While test 1 is running, perform normal vibration tests and voltage margin tests.

Note

If error printouts occur during the 10 minutes of the program, troubleshoot and repair according to each error printout. Return to step n. Continue testing (from step n) until the test runs for 10 minutes without error printouts. After successful testing for 10 minutes, return control to the keyboard. Perform step r.

r. Declare TIRP per applicable program manual. (See XDS publication number 901156 page 3-8 or XDS publication number 901168 page 3-4.)

s. Examine the TIRP printout. Proceed to step t if there is one line of information for each receive line declared active. If there is not information for each line declared active, troubleshoot and repair all channels for which there is no information printout. After all indicated malfunctions have been corrected, resume (repeat) testing starting with step n.

t. Examine printout. Proceed to step u if TE, %E, and CSE are equal to zero. If they are not equal to zero, troubleshoot and repair per printout and resume testing starting with step n.

u. Examine printout. If the RATE specified by the printout equals the word per minute rate for each respective line (±2), the test has been successfully completed. If so, proceed to step v; if not, troubleshoot and repair per printout and resume testing starting with step n.

v. Remove all back-to-back jumpers, and restore all output cabling.

4-6 ADJUSTMENT PROCEDURES

The only adjustment required on the COC is the variable capacitor in the tank circuit of the oscillator on each CT17 timing generator module. The frequency of the oscillator is determined by the crystal; however, no oscillations occur unless the tank circuit is adjusted to the crystal frequency by setting the variable tuning capacitor.

Make adjustments on each CT17 module as follows:

a. With power applied to the COC, check the oscillator output at module position XB38 (for formats II, III, and IV), and module position XB33 (for format I) with an oscilloscope.

b. Using XDS production tool 7104-5, adjust the piston tuning capacitor, C10 (accessible from the front of the CT17 module when it is installed), for minimum capacity (fully counter clockwise). The level may be true or false.

c. Adjust the tuning capacitor clockwise until the output waveform is observed on the oscilloscope.

d. Rock the screwdriver and select a setting that provides stable operation at high, normal, and low margins.

e. To check for the proper frequency, take the reciprocal of the baud rate (see table 1-4).

4-7 GROUP ASSEMBLY PARTS LIST

The Group Assembly Parts List is a breakdown of all systems, assemblies, and subassemblies which can be disassembled, reassembled, or replaced and which are contained in the end article. The Group Assembly Parts List consists of columnar listings of parts related to illustrations. Parts are listed in order of disassembly sequence, except in cases where sequence of disassembly cannot be maintained. Attaching parts are listed below the related assembly or subassemblies. Items which are purchased in bulk form (for example, wire and insulating materials) are not listed.

Each parts list table is arranged in seven columns as follows:

a. The figure number of the part listed and the index number corresponding to the illustration reference

b. The XDS manufacturer's part number for the part

c. The vendor's part number for the part (if available)

d. A brief description of the part

e. The manufacturer's code for the part

f. The quantity of the part used per assembly

g. Usable on code column indicating that when a letter is used in the code column, the use of the coded part is restricted to the model identified by the code letter. (Where no letter symbol appears in this column, the part is used on all models of this configuration.)

How to use the Illustrated Parts Breakdown

To obtain information about a part, the following steps should be taken:

a. Refer to the applicable assembly breakdown

b. Compare the part with the illustration until part is located

c. Note the index number

d. Locate the index number in the corresponding Group Assembly Parts List

e. Find the part number and name of part opposite the index number listed

4-8 NUMERICAL INDEX

This index is a listing of the items contained in the Group Assembly Parts List. The numerical order of the index (table 4-10 is determined by the XDS part number.

ILLUSTRATED PARTS BREAKDOWN

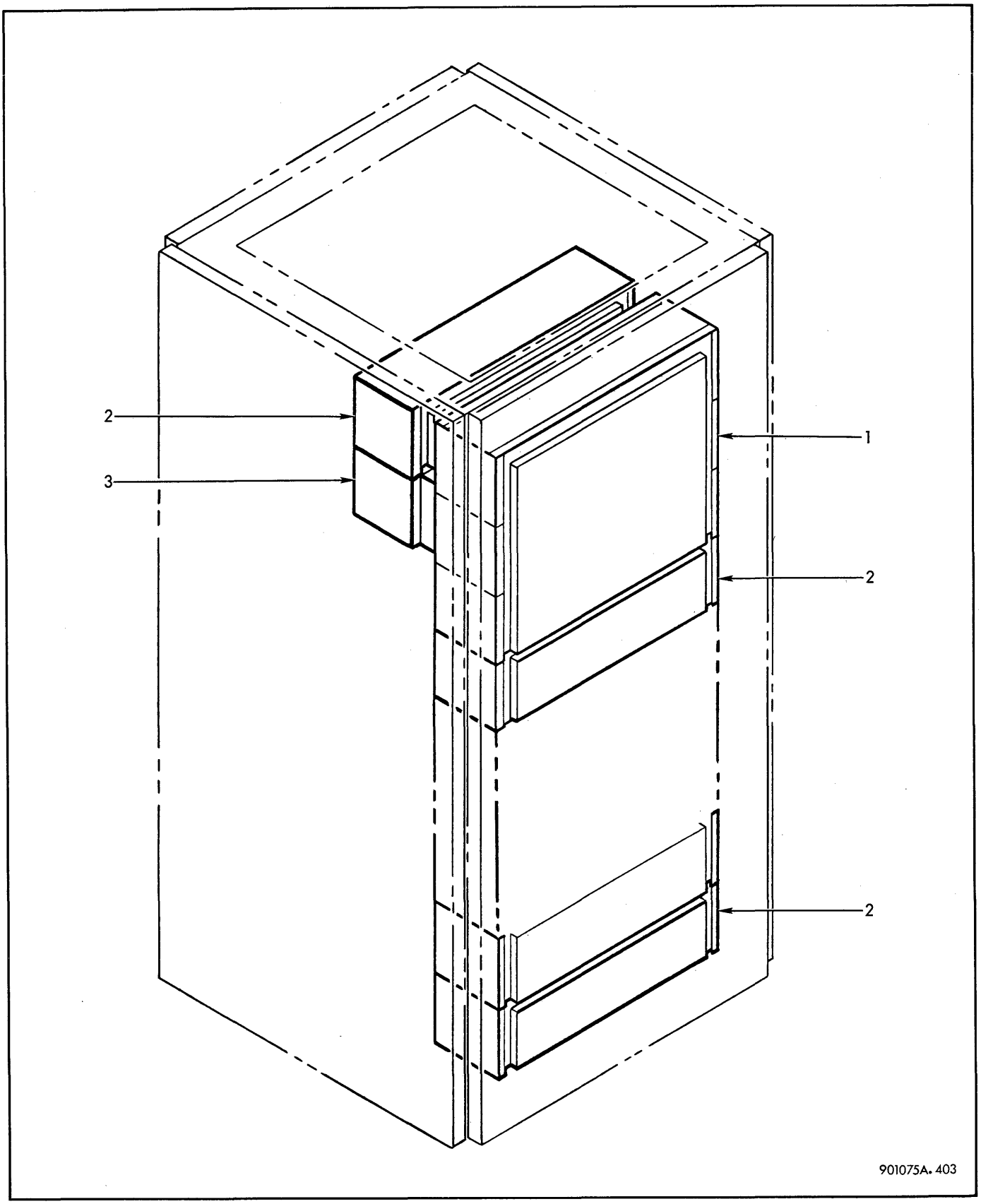


Figure 4-3. COC Equipment

Table 4-1. COC Equipment

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-3-			COC Equipment			
-1	133905		7611 Communications Controller		1	
-2	133985		. Line Interface Unit Assy		1	
-3	136614		. . Relay Interface Unit Assy		1	

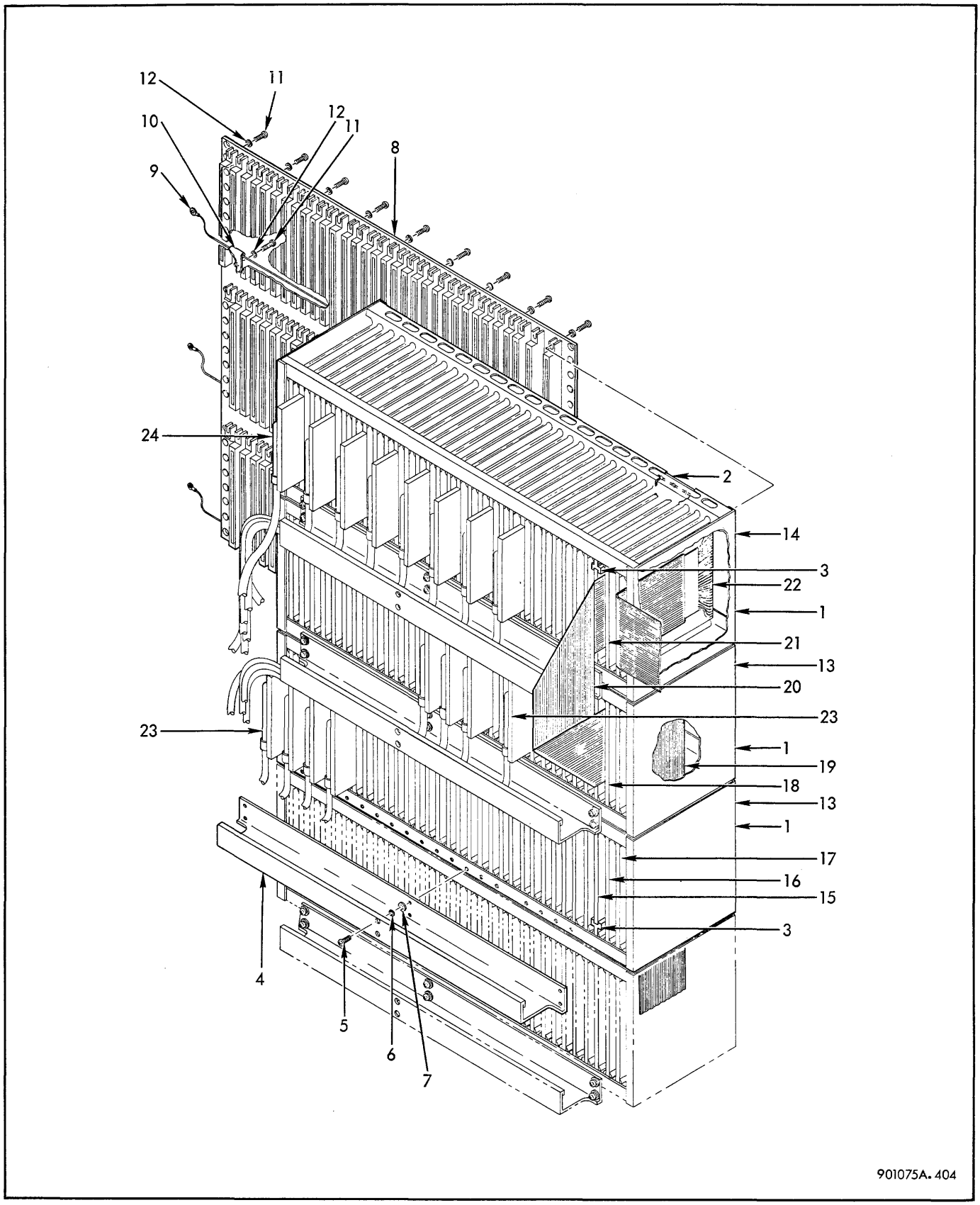


Figure 4-4. COC Controller, Exploded View

Table 4-2. COC Replaceable Parts and Interconnecting Cables

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-4-	133905		7611 Communications Controller			
-1	116231		. Chassis 32 Module		3	
-2	129567-001		. Nut, Strip Speed		6	
-3	149850		. Retainer		2	
-4	116522		. Channel Cable Routing		3	
			(Attaching Parts)			
-5	100012-203		. Screw, Pan Head		15	
-6	100018-200		. Washer, Flat		15	
-7	100024-200		. Washer, Lock Int Tooth		15	
			- - - * - - -			
-8	134038		. Wired Board Assy, 15-1/2"		1	
-9	131891-004		. Cable Assy, Busbar Pickup		3	
-10	100657-001		. Clamp, Cable		3	
			(Attaching Parts)			
-11	114538-214		. Screw, Sheet Metal Pan Head		54	
-12	100018-300		. Washer, Flat		54	
			- - - * - - -			
-13	136202		. Module Assy (Controller) (See Fig. 4-5 for Module Locations)		1	
-14	136201		. Module Assy (LIU) (See Fig. 4-7 for LIU #1 thru LIU #7) (See Fig. 4-4 Module Loaations LIU #0 thru LIU #7)		1	7611
-15	133212-171		. Ribbon Cable Assy (3C to LIU #1)		2	
-16	133212-701		. Ribbon Cable Assy (2C to LIU #2)		1	
-17	133212-122		. Ribbon Cable Assy (1C to LIU #3)		1	
-18	133212-232		. Ribbon Cable Assy (2B to LIU #4)		1	
-19	133212-292		. Ribbon Cable Assy (1B to LIU #5)		1	
-20	133204-171		. Ribbon Cable Assy (LIU #0-3A to Communication Controller, 3B)		1	

(Continued)

Table 4-2. COC Replaceable Parts and Interconnecting Cables (Cont.)

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-4-(Cont.)						
-21	133212-392		. Ribbon Cable Assy (LIU #0-2A to LIU #6)		1	
-22	133204-852		. Ribbon Cable Assy (LIU #0-1A to LIU #7)		1	
-23	127314		. Interconnecting Cable Assy (To IOP-DIO-Interr.)		8	
-23	139241L		. Interconnecting Cable Assy (To Interr., Optional)		A/R	
-23	127315		. Terminator, Connector (Optional)		8	
-24	136134		. Cable Assy, Receive Simplex Basic		A/R	
-24	135603		. Cable Assy, Receive EIA Basic		A/R	
-24	136137		. Cable Assy, Send Simplex Basic		A/R	
-24	136139		. Cable Assy, Half Duplex Basic		A/R	
-24	136141		. Cable Assy, Full Duplex Basic		A/R	
-24	136135		. Capacitor Connector Assy EIA		A/R	
-24	136136		. Capacitor Connector Assy MIL		A/R	
-24	136314		. Cable Assy, Relay Interface Basic		A/R	

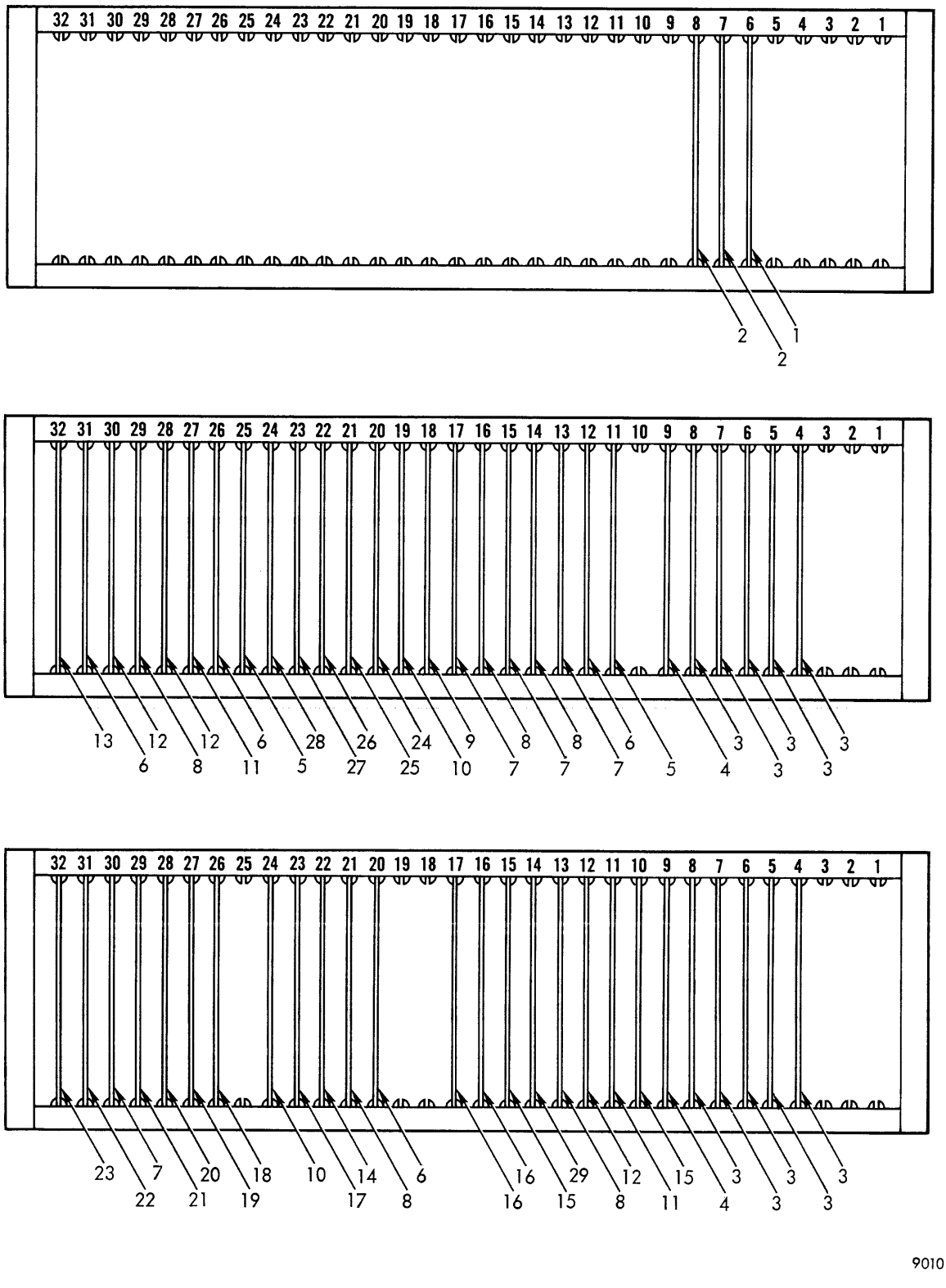


Figure 4-5. COC Module Locations

Table 4-3. COC Plug-In Modules

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-5-	136202		. Module Assy (Controller)			
-1	116257		. . Module Assy, XT10 Terminator Module		1	
-2	133987		. . Module Assy, LT46 Timing Buffer		2	
-3	134030		. . Module Assy, LT57 Collector/Driver		10	
-4	133931		. . Module Assy, LT55 Interrupt/Decoder		1	
-5	117000		. . Module Assy, IT13 Invert. Matrix		2	
-6	116994		. . Module Assy, IT11 NAND Gate		4	
-7	123019		. . Module Assy, AT11 Cable Driv/Rec		4	
-8	116056		. . Module Assy, BT10 AND/OR Gate		5	
-9	126613		. . Module Assy, BT18 BAND Gate		1	
-10	126982		. . Module Assy, LT26 Switch Comparator		2	
-11	116029		. . Module Assy, BT11 BAND Gate		2	
-12	116380		. . Module Assy, FT10 Basic Flip-Flop		3	
-13	126372		. . Module Assy, IT18 NAND Gate		1	
-14	116407		. . Module Assy, BT13 Buffered Matrix		2	
-15	117028		. . Module Assy, FT12 Gated Flip-Flop		2	
-16	117368		. . Module Assy, BT14 Gated Buffer		2	
-17	126712		. . Module Assy, LT25 Logic Element		1	
-18	126714		. . Module Assy, AT17 Cable Driv/Rec		1	
-19	126710		. . Module Assy, LT24 Logic Element		1	
-20	123018		. . Module Assy, AT10 Cable Receiver		1	
-21	133392		. . Module Assy, LT41 Logic Element		1	
-22	133657		. . Module Assy, LT43 Logic Element		1	
-23	124629		. . Module Assy, AT12 Cable Driver		1	

(Continued)

Table 4-3. COC Plug-In Modules (Cont.)

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-5-(Cont.)			(Optional)			
-24	135351		. . Module Assy, CT17 Timing Generator		1	7612
-24	128131-010		. . . Crystal (Format I, 60 wpm)		A/R	7612-01
-24	128131-011		. . . Crystal (Format I, 66-2/3 wpm)		A/R	7612-02
-24	128131-012		. . . Crystal (Format I, 75 wpm)		A/R	7612-03
-24	128131-018		. . . Crystal (Format I, 100 wpm)		A/R	7612-04
-25	135351		. . Module Assy, CT17 Timing Generator		1	7612
-25	128131-013		. . . Crystal (Format II, 148 wpm)		1	7612-06
-26	135351		. . Module Assy, CT17 Timing Generator		1	7612
-26	128131-015		. . . Crystal (Format III Slow, 150 wpm)		1	7612-07
-27	135351		. . Module Assy, CT17 Timing Generator		1	7612
-27	128131-014		. . . Crystal (Format IV, 100 wpm)		1	7612-05
			(High Speed Optional)			
-28	135351		. . Module Assy, CT17 Timing Generator		1	7614
-28	128131-015		. . . Crystal (Format III Fast, 1200 wpm)		A/R	7614
-28	128131-012		. . . Crystal (Format III Fast, 1800 wpm)		A/R	7614
-29	116986		. . . Module Assy, IT10 NAND-NOR Gate (Used with High Speed Optional CT17)		1	7614

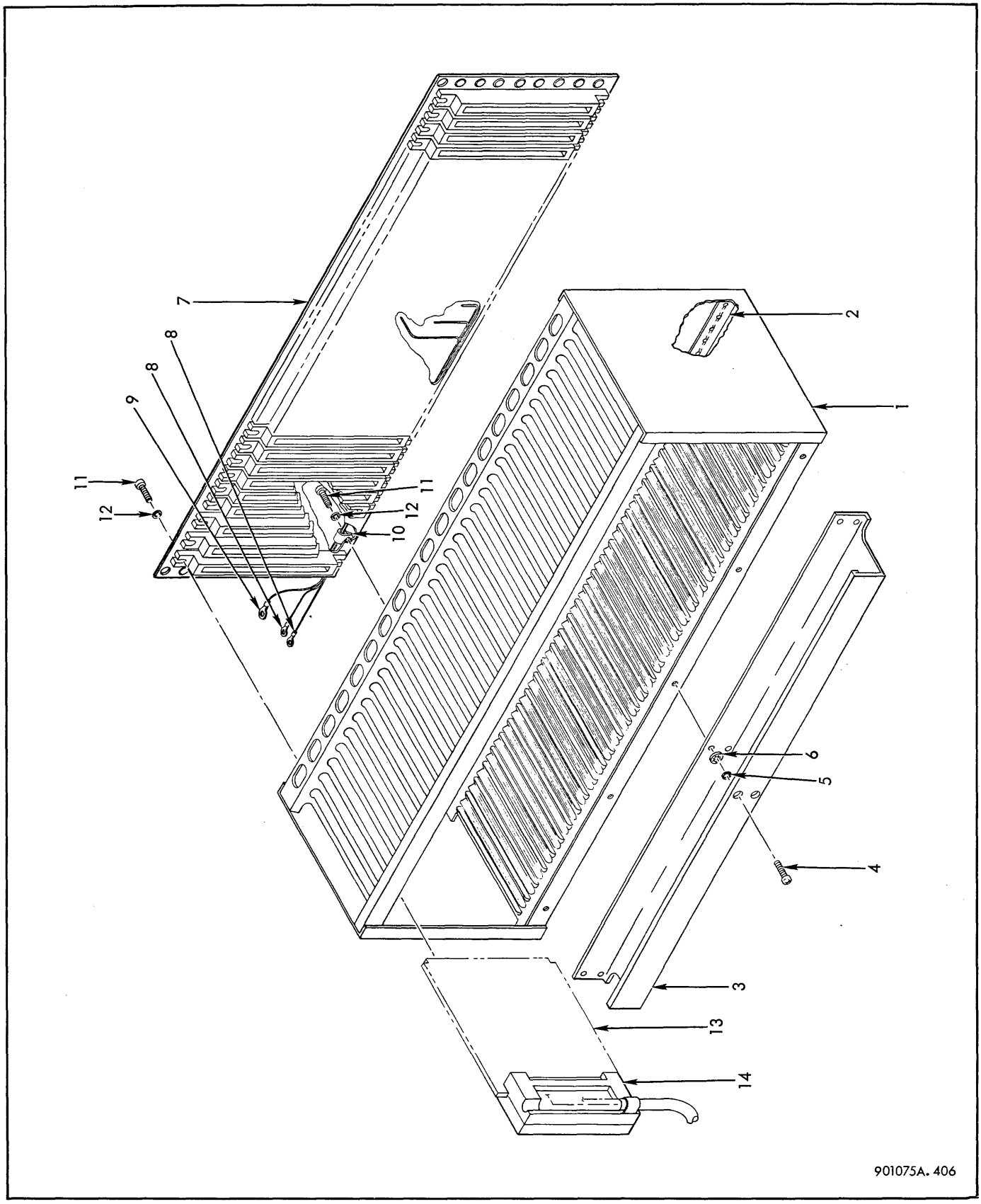


Figure 4-6. Line Interface Unit (Typical), Exploded View

Table 4-4. LIU Replaceable Parts and Interconnecting Cables

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-6-	133985		. Line Interface Unit Assy		1-7	7613
-1	116231		. . Chassis, 32 Module		1	
-2	129567		. . Nut, Strip Speed		2	
-3	116522		. . Channel, Cable Routing		1	
			(Attaching Parts)			
-4	100012-204		. . Screw, Pan Head		3	
-5	100018-200		. . Washer, Flat		3	
-6	100024-200		. . Washer, Lock Int Tooth		3	
			- - - * - - -			
-7	133986		. . Wired Board Assy, 5-1/4"		1-7	7613
-8	131891-004		. . Cable Assy, Busbar Pickup		2	
-9	131891-006		. . Cable Assy, Busbar Pickup		1	
-10	100657-001		. . Clamp, Cable Nylon		1	
			(Attaching Parts)			
-11	114538-214		. . Screw, Sheet Metal		18	
-12	100018-300		. . Washer, Flat		18	
			- - - * - - -			
-13	136201		. . Module Assy, (LIU #0 thru LIU #7) (See Fig. 4-7 for Module Locations)		1-8	
-14			. . Line Interface Connections, Cable Types, Optional, LIU #0 thru LIU #7			
	136141L		. . Cable Assy, Full Duplex Basic (Full Duplex EIA)			Optional
	136135		. . . Capacitor Connector Assy EIA		A/R	
	136141L		. . Cable Assy, Full Duplex Basic (Full Duplex MIL)			Optional
	136136		. . . Capacitor Connector Assy, MIL		A/R	
	136139L		. . Cable Assy, Half Duplex Basic (Half Duplex EIA)			Optional

(Continued)

Table 4-4. LIU Replaceable Parts and Interconnecting Cables (Cont.)

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-6-(Cont.)						
-14	136135		. . . Capacitor Connector Assy, EIA		A/R	
	136139L		. . Cable Assy, Half Duplex Basic (Half Duplex MIL)			Optional
	136136		. . . Capacitor Connector Assy, MIL		A/R	
	136137L		. . Cable Assy, Send Simplex Basic (Send Simplex EIA)			Optional
	136135		. . . Capacitor Connector Assy, EIA		A/R	
	136137L		. . Cable Assy, Send Simplex Basic (Send Simplex MIL)			Optional
	136136		. . . Capacitor Connector Assy, MIL		A/R	
	135603L		. . Cable Assy, Receive EIA Basic (Receive Simplex EIA)			Optional
	136136		. . . Capacitor Connector Assy, MIL		A/R	
	136134L		. . Cable Assy, Receive Simplex Basic (Receive Simplex MIL)			Optional
	136136		. . . Capacitor Connector Assy, MIL		A/R	
	135603L		. . Cable Assy, Receive EIA Basic (Dual Simplex EIA)			Optional
	136137L		. . . Cable Assy, Send Simplex Basic		A/R	
	136134L		. . Cable Assy, Receive Simplex Basic (Dual Simplex MIL)			Optional
	136137L		. . . Cable Assy, Send Simplex Basic		A/R	
	137314L		. . Cable Assy, Relay Interface Basic (Relay)			Optional
	136136		. . . Capacitor Connector Assy, MIL		A/R	

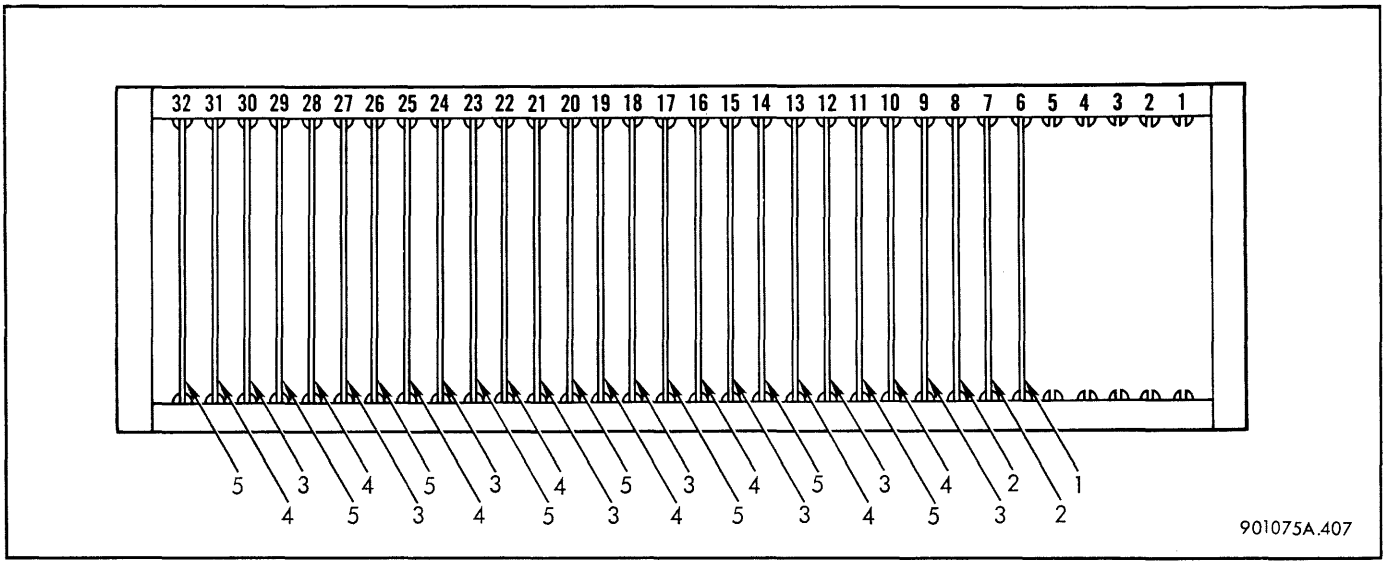


Figure 4-7. LIU Module Locations

Table 4-5. LIU Plug-In Modules

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-7-	136201		. . Module Assembly (LIU #0 thru #7)		1-8	
-1	116257		. . . Module Assy, XT10 Terminator Module		1-8	
-2	133987		. . . Module Assy, LT46 Timing Buffer		2-16	
-3	133906		. . . Module Assy, LT47 Send 5-7.5		1-8	7615-01
-3	133907		. . . Module Assy, LT48 Send 7-9		1-8	7615-02
-3	133908		. . . Module Assy, LT49 Send 8-10		1-8	7615-03
-3	133909		. . . Module Assy, LT50 Send 8-11		1-8	7615-04
-4	133914		. . . Module Assy, NT20 Current Interface		1-8	7620
-4	133915		. . . Module Assy, NT21 EIA Interface		1-8	7621
-4	133916		. . . Module Assy, NT22 MIL Interface		1-8	7613
-4	147636		. . . Module Assy, NT32 MIL Interface		1-8	7622
-5	133910		. . . Module Assy, LT51 Receive 5-7.5		1-8	7616-01
-5	133911		. . . Module Assy, LT52 Receive 7-9		1-8	7616-02
-5	133912		. . . Module Assy, LT53 Receive 8-10		1-8	7616-03
-5	133913		. . . Module Assy, LT54 Receive 8-11		1-8	7616-04

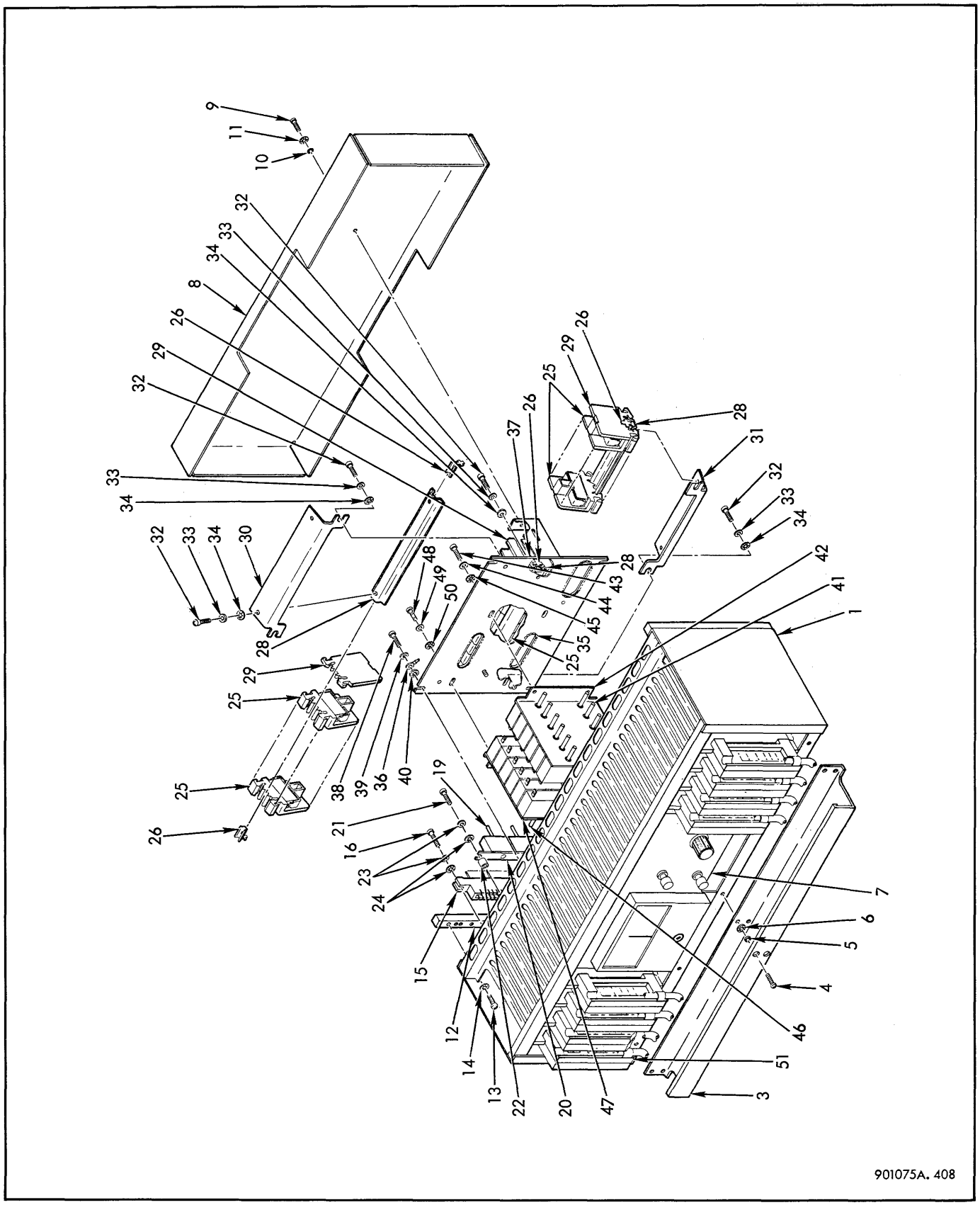


Figure 4-8. Relay Interface Unit, Exploded View

Table 4-6. Relay Interface Units Replaceable Parts

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-8-	136614		. . Relay, Interface Unit Assy		1	7620
-1	137729		. . . Chassis, Modified		1	
-2	129567		. . . Nut, Strip Speed		1	
-3	116522		. . . Channel, Cable Routing		1	
			(Attaching Parts)			
-4	100012-203		. . . Screw, Pan Head Phillips		3	
-5	100018-200		. . . Washer, Flat		3	
-6	100024-200		. . . Washer, Lock Int Tooth		3	
			- - - * - - -			
-7	139377		. . . Monitor Unit Assy **(See Fig. 4-9)**		REF	
-8	136620		. . . Cover, Rear		1	
			(Attaching Parts)			
-9	100012-304		. . . Screw, Pan Head Phillips		2	
-10	100018-300		. . . Washer, Flat		2	
-11	100024-300		. . . Washer, Lock Int Tooth		2	
			- - - * - - -			
-12	137429		. . . Block Insulating Busbar		2	
			(Attaching Parts)			
-13	100012-304		. . . Screw, Pan Head Phillips		4	
-14	100024-300		. . . Washer, Lock Int Tooth		4	
			- - - * - - -			
-15	117874		. . . Connector, Soldertail J1 thru J10		10	
			(Attaching Parts)			
-16	114538-212		. . . Screw, Sheet Metal		20	
-17	100018-300		. . . Washer, Flat		20	
-18	100024-300		. . . Washer, Lock Int Tooth		20	
			- - - * - - -			

(Continued)

Table 4-6. Relay Interface Units Replaceable Parts (Cont.)

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-8-(Cont.)						
-19	136298-002		. . . Circuit Breaker (CB-1 thru CB16)		16	
-20	136819		. . . Plate, Circuit Breaker Mounting External		8	
			(Attaching Parts)			
-21	114538-212		. . . Screw, Sheet Metal		16	
-22	107132-005		. . . Spacer, Round		16	
-23	100018-300		. . . Washer, Flat		16	
-24	100024-300		. . . Washer, Lock Int Tooth		16	
			- - - * - - -			
-25	109432-009		. . . Block, Terminal (For 22-14 AWG)		40	
-26	109432-005		. . . Retaining Clip		6	
-27	109432-006		. . . Mounting, Channel (10 Pos.) 3.52 Long		1	
-28	109432-006		. . . Mounting, Channel (15 Pos.) 5.24 Long		2	
-29	109432-010		. . . Plate, End		3	
-30	137430-001		. . . Bracket, Terminal Block Mounting		1	
-31	137430-002		. . . Bracket, Terminal Block Mounting		1	
			(Attaching Parts)			
-32	100012-304		. . . Screw, Pan Head Phillips		8	
-33	100018-300		. . . Washer, Flat		8	
-34	100024-300		. . . Washer, Lock Int Tooth		8	
			- - - * - - -			
-35	100840-002		. . . Grommet, Nylon		3	
-36	132570-001		. . . Terminal, Insulated Ring Tongue		2	

(Continued)

Table 4-6. Relay Interface Units Replaceable Parts (Cont.)

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-8-(Cont.)						
-37	136619		. . . Plate, Terminal Block Mtg		1	
			(Attaching Parts)			
-38	100012-304		. . . Screw, Pan Head Phillips		4	
-39	100018-300		. . . Washer, Flat		4	
-40	100024-300		. . . Washer, Lock Int Tooth		4	
			- - - * - - -			
-41	136298-002		. . . Circuit Breaker, (CB-17 thru CB-28)		12	
-42	136818-001		. . . Bracket, Circuit Breaker Mtg		1	
			(Attaching Parts)			
-43	100012-304		. . . Screw, Pan Head Phillips		2	
-44	100018-300		. . . Washer, Flat		2	
-45	100024-300		. . . Washer, Lock Int Tooth		2	
			- - - * - - -			
-46	136298-002		. . . Circuit Breaker, (CB-29 thru CB-40)		12	
-47	136818-002		. . . Bracket, Circuit Breaker Mtg		1	
			(Attaching Parts)			
-48	100012-304		. . . Screw, Pan Head Phillips		4	
-49	100018-300		. . . Washer, Flat		4	
-50	100024-300		. . . Washer, Lock Int Tooth		4	
			- - - * - - -			
-51	149610		. . . DC Interface Assy		8	
-51	137198		. . . . Module Assy, Telegraphic Relay (KT12)		8	

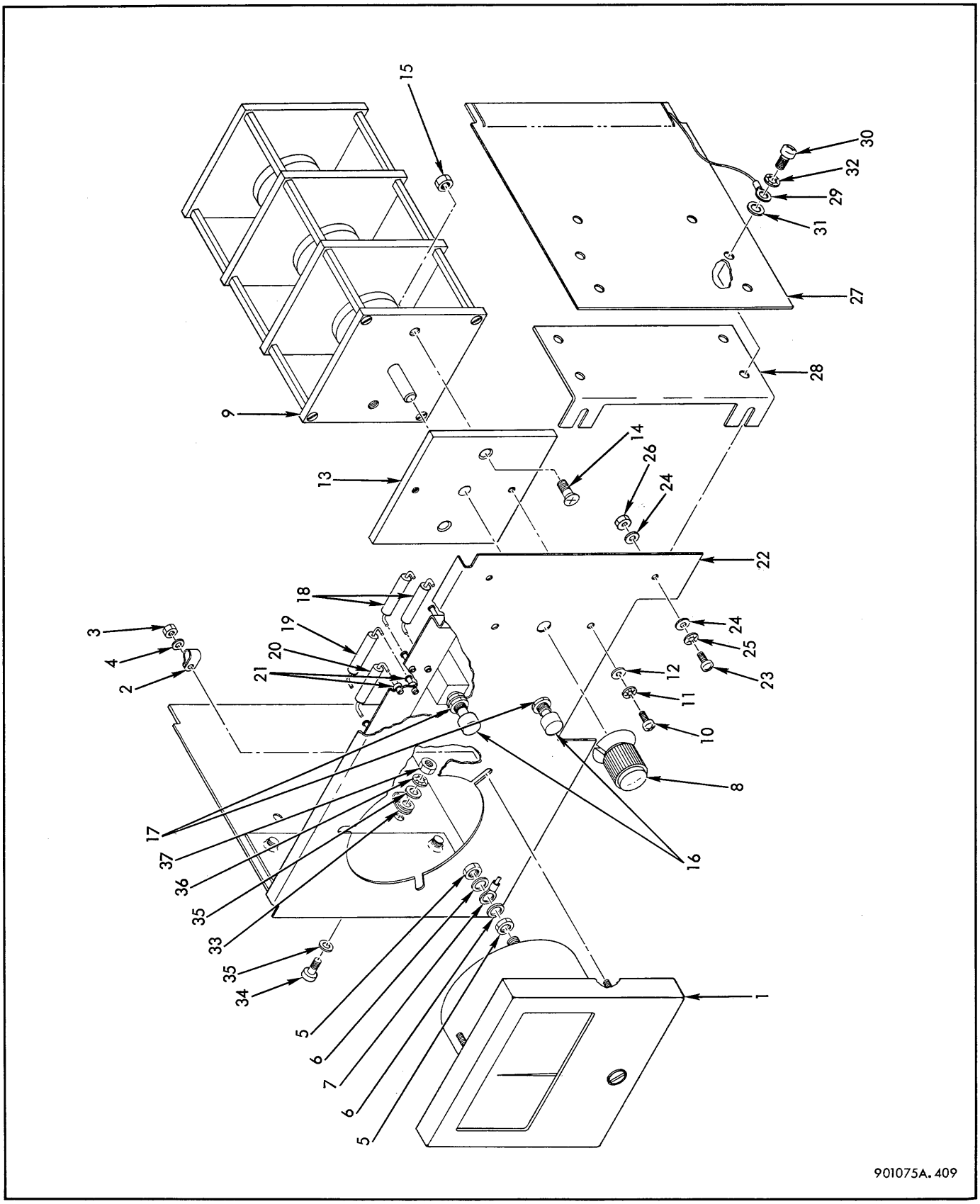


Figure 4-9. Monitor Unit, Exploded View

Table 4-7. Monitor Unit, Replaceable Parts

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-9-	139377		. . . Monitor Unit Assy		1	7620
-1	128174*		. . . . Meter, DC Voltage		1	
-2	100657-003		. . . . Clamp, Cable Nylon		1	
			(Attaching Parts)			
-3			. . . . Nut*		3	
-4	100018-300		. . . . Washer, Flat		3	
			- - - * - - -			
-5			. . . . Nut*		4	
-6			. . . . Washer*		4	
-7	100343-009		. . . . Terminal, Insulated Ring Tongue		2	
-8	146590		. . . . Knob, Control		1	
-9	123493		. . . . Switch, Rotary 3 Pole 24 Position (S3)		1	
			(Attaching Parts)			
-10	100012-306		. . . . Screw, Pan Head		2	
-11	100024-300		. . . . Washer, Lock Int Tooth		2	
-12	100018-300		. . . . Washer, Flat		2	
			- - - * - - -			
-13	137720		. . . . Spacer, Switch		1	
			(Attaching Parts)			
-14	100039-408		. . . . Screw, Flat Head Phillips		2	
-15	100008-400		. . . . Nut, Hex Machine		2	
			- - - * - - -			
-16	127003-001[t]		. . . . Switch, Pushbutton SPDT (S1, S2)		2	
-17			. . . . Nut, Hex[t]		2	

*Hardware supplied with DC Voltage Meter
[t]Supplied as part of an Assy

(Continued)

Table 4-7. Monitor Unit, Replaceable Parts (Cont.)

Fig. & Index No.	XDS Part Number	Vendor Part Number	Description 1 2 3 4 5 6 7	Mfg. Code	Units Per Assy	Usable on Code
4-9-(Cont.)						
-18	123362-384		. . . . Resistor, 1/8W 1% (R1, R2)		2	
-19	123362-266		. . . . Resistor, 1/8W 1% (R3)		1	
-20	123362-133		. . . . Resistor, 1/8W 1% (R4)		1	
-21	134118		. . . . Terminal, Feed Through		8	
-22	136615		. . . . Panel Front		1	
			(Attaching Parts)			
-23	100012-305		. . . . Screw, Pan Head		2	
-24	100018-300		. . . . Washer, Flat		4	
-25	100024-300		. . . . Washer, Lock Int Tooth		2	
-26	100008-300		. . . . Nut, Hex Machine		2	
			- - - * - - -			
-27	136616		. . . . Board, Front Panel Mounting		2	
-28	136617		. . . . Bracket, Front Cover Mounting		1	
-29	100343-003		. . . . Terminal, Insulated Ring Tongue		2	
			(Attaching Parts)			
-30	100012-304		. . . . Screw, Pan Head		8	
-31	100018-300		. . . . Washer, Flat		8	
-32	100024-300		. . . . Washer, Lock Int Tooth		8	
			- - - * - - -			
-33	100657-003		. . . . Clamp, Cable Nylon		1	
			(Attaching Parts)			
-34	100012-305		. . . . Screw, Pan Head		1	
-35	100018-300		. . . . Washer, Flat		2	
-36	100024-300		. . . . Washer, Lock Int Tooth		1	
-37	100008-300		. . . . Nut, Hex Machine		1	
			- - - * - - -			

Table 4-8. COC Replaceable Parts, Numerical Listing

Fig. & Index No.	XDS Part Number	Description
4-9-24	100008-300	Nut, Hex Mach
4-9-26	100008-300	Nut, Hex Mach
4-9-31	100008-300	Nut, Hex Mach
4-9-37	100008-300	Nut, Hex Mach
4-9-15	100008-400	Nut, Hex Mach
4-4-5	100012-203	Screw, Pan Hd
4-8-4	100012-203	Screw, Pan Hd
4-6-4	100012-204	Screw, Pan Hd
4-8-9	100012-304	Screw, Pan Hd Phil
4-8-13	100012-304	Screw, Pan Hd Phil
4-8-32	100012-304	Screw, Pan Hd Phil
4-8-38	100012-304	Screw, Pan Hd Phil
4-8-43	100012-304	Screw, Pan Hd Phil
4-8-48	100012-304	Screw, Pan Hd Phil
4-9-30	100012-304	Screw, Pan Hd Phil
4-9-23	100012-305	Screw, Pan Hd
4-9-34	100012-305	Screw, Pan Hd
4-9-10	100012-306	Screw, Pan Hd
4-4-6	100018-200	Washer, Flat
4-6-5	100018-200	Washer, Flat
4-8-4	100018-200	Washer, Flat
4-4-12	100018-300	Washer, Flat
4-6-11	100018-300	Washer, Flat
4-8-10	100018-300	Washer, Flat
4-8-17	100018-300	Washer, Flat
4-8-23	100018-300	Washer, Flat
4-8-33	100018-300	Washer, Flat
4-8-39	100018-300	Washer, Flat
4-8-44	100018-300	Washer, Flat
4-8-49	100018-300	Washer, Flat
4-9-4	100018-300	Washer, Flat
4-9-12	100018-300	Washer, Flat
4-9-35	100018-300	Washer, Flat
4-4-7	100024-200	Washer, Lock Int Tooth
4-6-6	100024-200	Washer, Lock Int Tooth
4-8-6	100024-200	Washer, Lock Int Tooth
4-8-11	100024-300	Washer, Lock Int Tooth
4-8-14	100024-300	Washer, Lock Int Tooth
4-8-18	100024-300	Washer, Lock Int Tooth
4-8-34	100024-300	Washer, Lock Int Tooth
4-8-40	100024-300	Washer, Lock Int Tooth
4-8-45	100024-300	Washer, Lock Int Tooth
4-8-50	100024-300	Washer, Lock Int Tooth
4-9-11	100024-300	Washer, Lock Int Tooth
4-9-25	100024-300	Washer, Lock Int Tooth
4-9-32	100024-300	Washer, Lock Int Tooth
4-9-36	100024-300	Washer, Lock Int Tooth
4-9-14	100039-408	Screw, Flat Hd Phil
4-9-29	100343-003	Terminal, Ins Ring Tongue
4-9-33	100343-003	Terminal, Ins Ring Tongue
4-9-7	100343-009	Terminal, Ins Ring Tongue
4-4-10	100657-001	Clamp, Cable Nylon
4-6-9	100657-001	Clamp, Cable Nylon
4-9-2	100657-003	Clamp, Cable Nylon

(Continued)

Table 4-8. COC Replaceable Parts, Numerical Listing (Cont.)

Fig. & Index No.	XDS Part Number	Description
4-8-35	100840-002	Grommet, Nylon
4-8-22	107132-005	Spacer, Round
4-8-26	109432-005	Retaining Clip
4-8-27	109432-006	Mtg Channel (10 Pos 3.52 Lg)
4-8-28	109432-006	Mtg Channel (15 Pos 5.24 Lg)
4-8-25	109432-009	Block, Terminal (For 22-14 AWG)
4-8-29	109432-010	End Plate
4-8-16	114538-212	Screw, Sheet Metal
4-8-21	114538-212	Screw, Sheet Metal
4-4-11	114538-214	Screw,Sheet Metal Pan Hd
4-6-10	114538-214	Screw,Sheet Metal Pan Hd
4-5-11	116029	Module Assy, BT11 BAND Gate
4-5-8	116056	Module Assy, BT10 AND/OR Gate
4-4-1	116231	Chassis, 32 Module
4-6-1	116231	Chassis, 32 Module
4-5-1	116257	Module Assy, XT10 Terminator Module
4-7-1	116257	Module Assy, XT10 Terminator Module
4-5-11	116380	Module Assy, FT10 Basic Flip-Flop
4-5-14	116407	Module Assy, BT13 Buffered Matrix
4-4-3	116522	Channel, Cable Routing
4-6-3	116522	Channel, Cable Routing
4-8-3	116522	Channel, Cable Routing
4-5-29	116986	Module Assy, IT10 NAND-NOR Gate (High Speed Optional)
4-5-6	116994	Module Assy, IT11 NAND Gate
4-5-5	117000	Module Assy, IT13 Inverter Matrix
4-5-15	117028	Module Assy, FT12 Gated Flip-Flop
4-5-16	117368	Module Assy, BT14 Gated Buffer
4-8-15	117874	Connector, Soldertail (J1 thru J10)
4-5-20	123018	Module Assy, AT10 Cable Receiver
4-5-7	123019	Module Assy, AT11 Cable Driver/Receiver
4-9-20	123362-133	Resistor, 1/8 W 1% (R4)
4-9-19	123362-266	Resistor, 1/8 W 1% (R3)
4-9-18	123362-384	Resistor, 1/8 W 1% (R1, R2)
4-9-9	123493	Switch, Rotary 3 Pole 24 Position (S3)
4-5-23	124629	Module Assy, AT12 Cable Driver
4-5-13	126372	Module Assy, IT18 NAND Gate
4-5-9	126613	Module Assy, BT18 BAND Gate
4-5-19	126710	Module Assy, LT24 Logic Element
4-5-17	126712	Module Assy, LT25 Logic Element
4-5-18	126714	Module Assy, AT13 Cable Driver/Receiver

(Continued)

Table 4-8. COC Replaceable Parts, Numerical Listing (Cont.)

Fig. & Index No.	XDS Part Number	Description
4-5-10	126982	Module Assy, LT26 Switch Comparator
4-9-16	127003-001	Switch, Pushbutton SPDT (S1, S2)
4-4-23	127314	Interconnecting Cable Assy, (To IOP-DIO-Interrupt)
4-4-23	127315	Terminator, Connector (Optional)
4-5-24	128131-010	Crystal (Format I, 60 wpm)
4-5-24	128131-011	Crystal (Format I, 66 wpm)
4-5-24	128131-012	Crystal (Format I, 75 wpm)
4-5-28	128131-012	Crystal (Format III Fast, 1800 wpm)
4-5-25	128131-013	Crystal (Format II, 148 wpm)
4-5-27	128131-014	Crystal (Format IV, 100 wpm)
4-6-26	128131-015	Crystal (Format III Slow, 150 wpm)
4-5-28	128131-015	Crystal (Format III Fast, 1200 wpm)
4-5-24	128131-018	Crystal (Format I, 100 wpm)
4-6-2	129567	Nut, Strip Speed
4-8-2	129567	Nut, Strip Speed
4-4-1	129567-001	Nut, Strip Speed
4-9-1	128174	Meter, Dc Voltage
4-4-9	131891-004	Cable Assy, Busbar Pickup
4-6-7	131891-004	Cable Assy, Busbar Pickup
4-6-8	131891-006	Cable Assy, Busbar Pickup
4-8-36	132570-001	Terminal, Ins Ring Tongue
4-4-20	133204-171	Ribbon Cable Assy, (LIU #0-3A to CC-3B)
4-4-22	133204-852	Ribbon Cable Assy, (LIU #0-1A to LIU #7)
4-4-17	133212-122	Ribbon Cable Assy, (IC to LIU #3)
4-4-15	133212-171	Ribbon Cable Assy, (3C to LIU #1)
4-4-18	133212-232	Ribbon Cable Assy, (2B to LIU #4)
4-4-19	133212-292	Ribbon Cable Assy, (1B to LIU #5)
4-4-21	133212-392	Ribbon Cable Assy, (LIU #0-2A to LIU #6)
4-4-16	133212-701	Ribbon Cable Assy, (2C to LIU #2)
4-5-21	133392	Module Assy, LT41 Logic Element
4-5-22	133657	Module Assy, LT43 Logic Element
4-4-	133905	7611 Communications Controller
4-3-1	133905	7611 Communications Controller
4-7-3	133906	Module Assy, LT47 Send 5-7.5
4-7-3	133907	Module Assy, LT48 Send 7-9
4-7-3	133908	Module Assy, LT49 Send 8-10
4-7-3	133909	Module Assy, LT50 Send 8-11
4-7-5	133910	Module Assy, LT51 Receive 5-7.5

(Continued)

Table 4-8. COC Replaceable Parts, Numerical Listing (Cont.)

Fig. & Index No.	XDS Part Number	Description
4-7-5	133911	Module Assy, LT52 Receive 7-9
4-7-5	133912	Module Assy, LT53 Receive 8-10
4-7-4	133914	Module Assy, NT20 Current Interface
4-7-4	133915	Module Assy, NT21 EIA Interface
4-7-4	133916	Module Assy, NT22 MIL Interface
4-5-4	133931	Module Assy, LT55 Interrupt-Decoder
4-6-	133985	Line Interface Unit Assy
4-6-6	133986	Wired Board Assy, 5-1/4"
4-5-2	133987	Module Assy, LT46 Timing Buffer
4-7-2	133987	Module Assy, LT46 Timing Buffer
4-5-3	134030	Module Assy, LT57 Collector/Driver
4-4-8	134038	Wired Board Assy, 15-1/2"
4-9-21	134118	Terminal, Feed Through
4-5-24	135351	Module Assy, CT17 Timing Generator
4-5-25	135351	Module Assy, CT17 Timing Generator
4-5-26	135351	Module Assy, CT17 Timing Generator
4-5-27	135351	Module Assy, CT17 Timing Generator
4-5-28	135351	Module Assy, CT17 Timing Generator (High Speed Optional)
4-4-24	135603	Cable Assy, Receive EIA Basic
4-5-13	135603L	Cable Assy, EIA Basic
4-4-24	136134	Cable Assy, Receive Simplex Basic
4-6-13	136134	Cable Assy, Receive Simplex Basic
4-6-13	136135	Capacitor Connector Assy, EIA
4-4-24	136135	Capacitor Connector Assy, EIA
4-6-13	136136	Capacitor Connector Assy, MIL
4-4-24	136137	Cable Assy, Send Simplex Basic
4-6-13	136137L	Cable Assy, Send Simplex Basic
4-6-13	136139	Cable Assy, Half Duplex Basic
4-4-24	136139	Cable Assy, Half Duplex Basic
4-4-25	136141	Cable Assy, Full Duplex Basic
4-6-13	136141L	Cable Assy, Full Duplex Basic
4-4-14	136201	Module Assy, (LIU #0 thru LIU #7)
4-7-	136201	Module Assy, (LIU #0 thru LIU #7)
4-4-13	136202	Module Assy, (Controller)
4-5-	136202	Module Assy, (Controller)
4-8-19	136298-002	Circuit Breaker (CB-1 thru CB-16)
4-8-41	136298-002	Circuit Breaker (CB-17 thru CB-28)
4-8-46	136298-002	Circuit Breaker (CB-29 thru CB-40)

(Continued)

Table 4-8. COC Replaceable Parts, Numerical Listing (Cont.)

Fig. & Index No.	XDS Part Number	Description
4-4-24	136314	Cable Assy, Relay Interface Basic
4-6-13	136314L	Cable Assy, Relay Interface Basic
4-8-	136614	Relay Interface Unit Assy
4-9-22	136615	Panel, Front
4-9-27	136616	Board, Front Panel Mtg
4-9-28	136617	Bracket, Front Cover Mtg
4-8-37	136619	Plate, Terminal Block Mtg
4-8-8	136620	Cover, Rear
4-8-42	136818-001	Bracket, Circuit Breaker Mtg
4-8-47	136818-002	Bracket, Circuit Breaker Mtg
4-8-20	136819	Plate, Circuit Breaker Mtg External
4-8-52	137198	Module Assy, KT12 Telegraphic Relay
4-8-12	137429	Block, Ins Busbar
4-8-30	137430-001	Bracket, Terminal Block Mtg
5-8-31	137430-002	Bracket, Terminal Block Mtg
4-4-16	137481-122	Ribbon Cable Assy, (1C to LIU #3)
4-4-14	137481-171	Ribbon Cable Assy, (3C to LIU #1)
4-4-17	137481-232	Ribbon Cable Assy, (2B to LIU #4)
4-4-18	137481-292	Ribbon Cable Assy, (1B - OIU #5)
4-4-20	137481-392	Ribbon Cable Assy, (LIU #0-2A to LIU #6)
4-4-15	137481-701	Ribbon Cable Assy, (2C to LIU #2)
4-9-13	137720	Spacer, Switch
4-8-1	137729	Chassis, Modified
4-4-23	139214L	Interconnecting Cable Assy, (To Interr Optional)
4-9-	139377	Monitor Unit Assy
4-9-8	146590	Knob, Control
4-7-4	147636	Module Assy, NT32 MIL Interface
4-8-51	149610	DC Interface Assy
4-4-3	149850	Retainer

READER SURVEY

XDS
Xerox Data Systems

Use this postpaid form to communicate any constructive comments you may have regarding this publication. Please be specific.

PUBLICATION

NUMBER	DATE OF PUBLICATION	TITLE	REV.

HOW DID YOU USE THIS PUBLICATION

☐ LEARNING ☐ INSTALLING ☐ MAINTAINING ☐ OPERATING ☐ SALES

☐ OTHER _____

COMMENTS

YOUR NAME AND RETURN ADDRESS

THANK YOU FOR YOUR INTEREST!

STAPLE

STAPLE

FOLD

FIRST CLASS
PERMIT NO. 229
EL SEGUNDO, CALIF.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

Xerox Data Systems
701 South Aviation Blvd.
El Segundo, California 90245

ATTN: MARKETING PUBLICATIONS

FOLD

CUT ALONG LINE