pdp11

DU11 single line programmable synchronous interface maintenance manual

digital

EK-DU11-MM-001

DU11 single line programmable synchronous interface maintenance manual

digital equipment corporation • maynard, massachusetts

Preliminary Edition, October 1973
1st Edition, January 1974
2nd Printing, March 1974
3rd Printing, December 1974

The material in this manual is for informational purposes and is subject to change without notice.

Digital Equipment Corporation assumes no responsibility for any errors which may appear in this manual.

Printed in U.S.A.

4/79-14

CONTENTS

CONTENTS (Cont)

Page

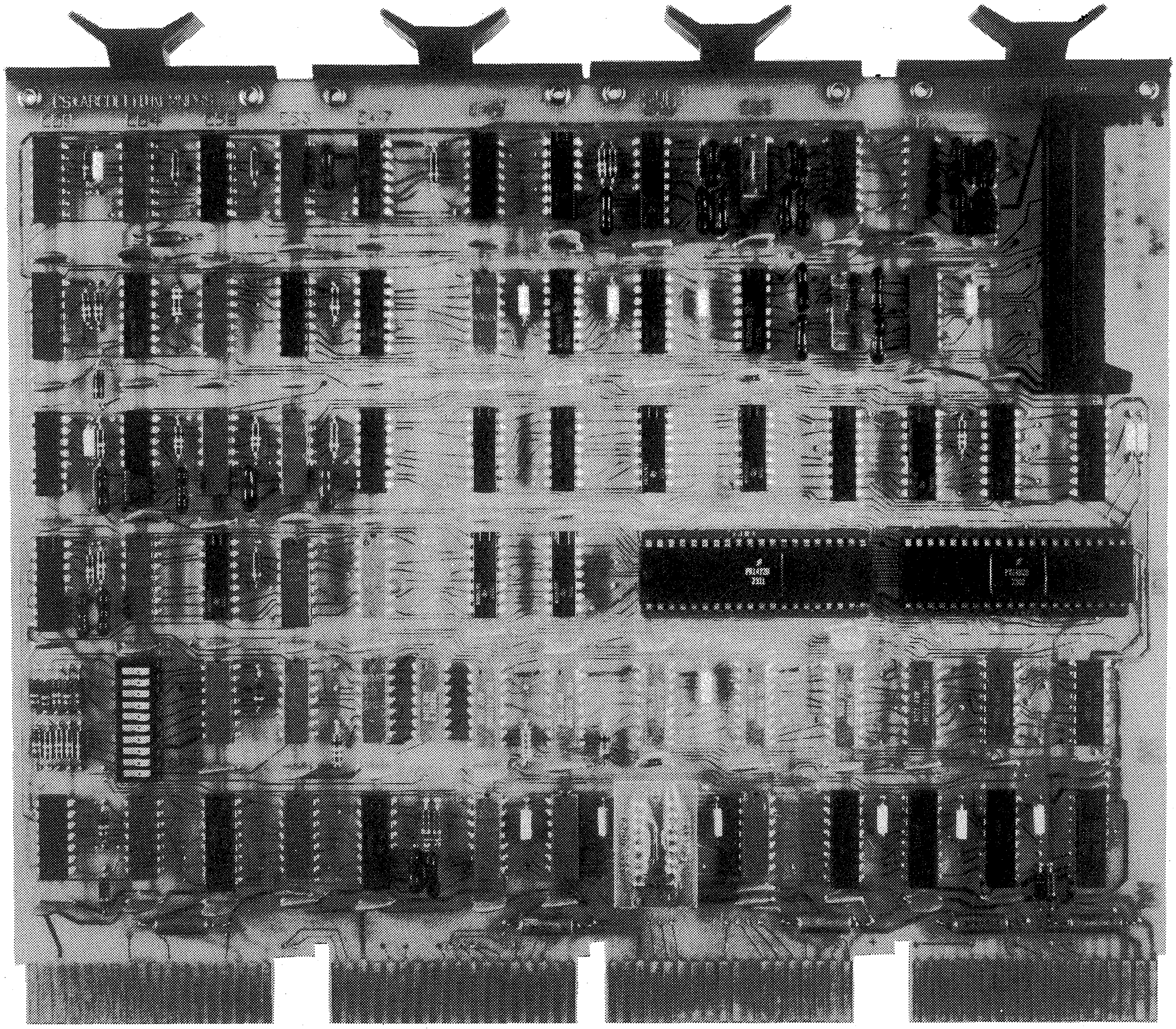
CONTENTS (Cont)

ILLUSTRATIONS

ILLUSTRATIONS (Cont)

TABLES

DU11 SINGLE LINE PROGRAMMABLE SYNCHRONOUS INTERFACE MAINTENANCE MANUAL



6731-1

DU11 Programmable Synchronous Interface

CHAPTER 1
INTRODUCTION

1.1 SCOPE

This manual provides a complete description of the DU11 Line Interface including installation, theory of operation, programming, and maintenance. The level of discussion assumes that the reader is familiar with basic digital computer theory.

This chapter contains introductory information. It includes a description of data communication techniques and systems, a general description of the DU11, a physical description of the DU11, DU11 specifications, and an explanation of engineering drawing conventions.

1.2 DATA COMMUNICATION TECHNIQUES AND SYSTEMS

1.2.1 Data Communication Techniques

There are several techniques used for the transfer of data communication signals. Each has its particular advantages and disadvantages.

1.2.1.1 Pulse Coding – Standard data communication messages are sent in some form of pulse code. There are several varieties of pulsed codes used in the transferral of data in digital form. Binary signals, by their very nature, are natural elements for digital data codes. Such codes are said to be in "binary format."

A formatted binary code can represent different symbols only by allowing sufficient binary elements for each symbol. If we think of one binary digit (or "bit") representing each symbol, we have only two choices: one symbol represented by the "on" state, the other represented by the "off" state. With such an arrangement, we could let the "on" or one state represent "no" and the "off" or zero state represent "yes." While it would be difficult with such an arrangement, we could convey messages of a very limited nature from a remote station (such as the answer to "Is the temperature at your station over 70° F?").

If, instead of using one binary digit for our character, we use two, we have more characters to choose from. Our choice for a one-bit code was limited to two: 0 or 1. Our choice for a two-bit code is four: 00, 01, 010, or 11. If we choose a three-bit code, our choice is eight: 000, 001, 010, 011, 100, 101, 110, and 111. It can be shown that for a code with a character makeup of n bits, the number of characters available will be 2^n. In communications parlance, instead of calling these codes one-bit codes, two-bit codes, etc., they are called one-level codes, two-level codes, etc. Although any arbitrary meaning can be assigned to a code character, it is more practical for the majority of operations to let the characters represent numbers, punctuation marks, spaces, and letters of the alphabet. In addition to these, some special codes use characters for other meanings.

1.2.1.2 Pulse Code Transmission – In order to transmit code characters, it is necessary to arrange their elements in a way that will allow their reception without uncertainty. There are several techniques by which this may be done; these techniques fall into two broad categories: serial data transmission and parallel data transmission.

Because the DU11 is a serial communication interface, only serial data transmission techniques will be discussed.

There are two basic techniques of serial data transmission: asynchronous and synchronous. These two techniques as well as a third, isochronous, will be discussed in the following paragraphs.

1.2.1.2.1 Asynchronous Serial Transmission – This technique enables data to be transferred as it becomes available. This is possible by framing each data character with a begin signal (START bit) and an end signal (STOP bit), so that the equipment receiving the data (the interface receiver) knows when a data character is being presented on the communication line and when the line is inactive.

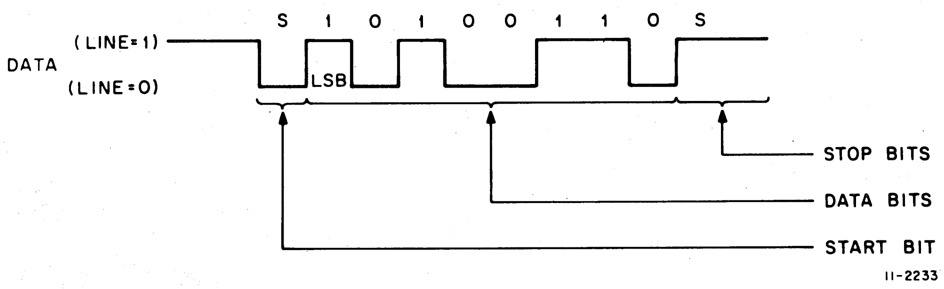


Figure 1-1 Asynchronous Technique Format

Hence, each character consists of three parts: a START bit, the data bits, and a STOP bit (Figure 1-1). A START bit is a line state (usually a zero) that lasts for 1 bit time. The data bits represent the actual binary character being transferred. In many applications the characters are 8 bits long with the least significant bit being sent out and received first. A STOP bit is a line state (usually a one) that lasts for 1, 1.42, or 2 bit times; it indicates that character transmission is complete. The STOP bit enables the interface receiver to check synchronization after each character transmission. If the STOP bit is not received properly, i.e., it is not presented on the line immediately after the last data bit, the character received is considered erroneous and re-transmission is necessary.

Clocking for the interface transmitter and interface receiver during asynchronous transmission is provided by two different sources that are asynchronous to one another. The transmitter clock is enabled when data is available for transmission and clocks the character onto the line. The receiver clock is enabled when a START bit is detected on the line and samples the data bits as they are presented on the line. The receiver is also equipped with a counter that counts the character bits received. When a complete character and a STOP bit are received (the receiver must know the number of bits per character), the receiver clock is disabled until the next START bit is detected.

The asynchronous serial data transmission technique has the following advantages:

a. Can be generated easily by electromechanical equipment (e.g., Teletype® keyboard).

b. Can be used easily to drive mechanical equipment (e.g., Teletype printer).

c. Characters can be sent asynchronously (as they become available) because each character has its own synchronizing information.

The disadvantages of the asynchronous serial data transmission technique are:

a. Separate timing required for both transmitter and receiver.

b. Distortion sensitive because the receiver depends on incoming signal sequences to become synchronized. Any distortion in these sequences will affect the reliability with which the character is assembled.

c. Speed limited because a reasonable amount of margin between characters must be built in to accommodate distortion.

d. Inefficient because at least 10 bit times are required to send 8 bits of data. If a 2 bit time STOP bit is used, it takes 11 bit times to transfer 8 bits of data.

1.2.1.2.2 Synchronous Serial Transmission – This technique does not use START and STOP bits to accomplish synchronization. Instead, the entire block of data (message) is preceded on the line by a synchronizing code. When the interface receiver recognizes this code (henceforth referred to as sync characters), it locks in and, using a counter, assembles the data characters which follow. Hence, as in the asynchronous technique, the receiver must know the number of bits per character.

This technique requires that the clocking for the interface transmitter and interface receiver be provided by a common clock source. The clock signal is provided to the transmitter and receiver on lines separate from the data line. At the transmitter, the clock signal serves to clock the data onto the line. At the receiver, the clock signal gates the data in. Figure 1-2 illustrates the timing for a synchronous communication system using modems.

®Teletype is a registered trademark of Teletype Corporation.

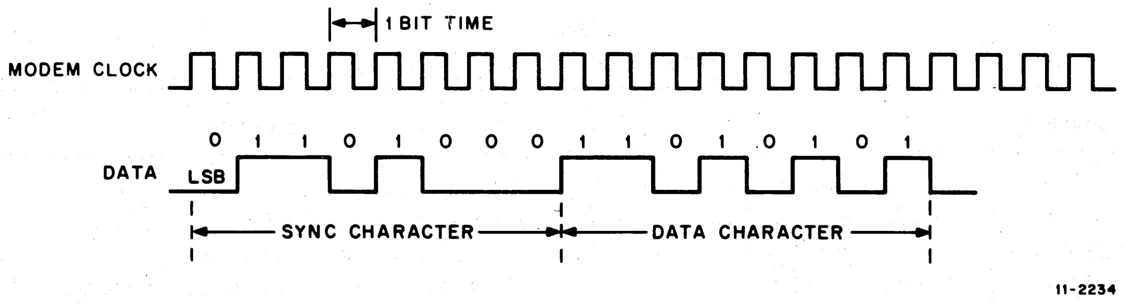


Figure 1-2 Synchronous Format

As shown in Figure 1-2, the modem provides the clock, the transmitter presents the data to the line on the positive going edge of the clock and the receiver samples the data on the negative going edge. If the transmitter pauses at any time and fails to inhibit the clock, the receiver will continue to sample the line, synchronization is lost and the remainder of the message will be erroneous.

The advantages of the synchronous serial data transmission technique are:

a. Modem timing sources can be used for both transmitter and receiver.

b. Interface receiver does not require clock-synchronizing logic as the asynchronous technique does.

c. Highly efficient because there are no bit times wasted with the use of START and STOP bits. All bits on the line are data, with the exception of the sync characters at the beginning of the bit stream.

d. Low distortion sensitivity because the timing is provided along with the data.

e. Higher speeds are achievable because of the low distortion sensitivity.

The disadvantages of the synchronous serial data transmission technique are:

a. Characters must be sent synchronously, not asynchronously (asynchronous transmission is desirable for most real time and mechanical applications).

b. One bit time added to or missing from the data-bit stream can cause the entire message to be faulty.

c. The common-carrier equipment required to accommodate this mode of operation is more expensive than the equipment required for asynchronous modes of operation.

d. Mechanical equipment cannot transmit or receive this format directly.

1.2.1.2.3 Isochronous Serial Transmission – This technique is essentially the transmission of asynchronous data over a synchronous modem. Character synchronization is achieved via START and STOP bits; a common timing source is used for both the transmitter and receiver.

The isochronous technique does have advantages over the asynchronous technique. Clocking for isochronous operations emanates from the modems and is synchronous to the data; hence, the receiver does not require clock-synchronizing logic and distortion sensitivity is low making higher speeds possible.

1.2.2 Data Communication Systems

1.2.2.1 Synchronous Systems – Synchronous modulator-demodulators (modems) have permitted a higher rate of data transmission than asynchronous modems over a voice grade facility. The nature of these transmission techniques has also resulted in higher efficiency by eliminating the need for synchronizing information with every character.

The logic design of interfaces to a synchronous modem is considerably easier than the design of an asynchronous interface because there is no need for bit synchronization and sampling hardware. Most synchronous modems supply all the timing necessary to receive each bit as it is made available from the modem. The difficulty in designing a synchronous modem interface is to design the capability of communicating in the message formats used in synchronous communications.

Table 1-1
Representative Message Codes

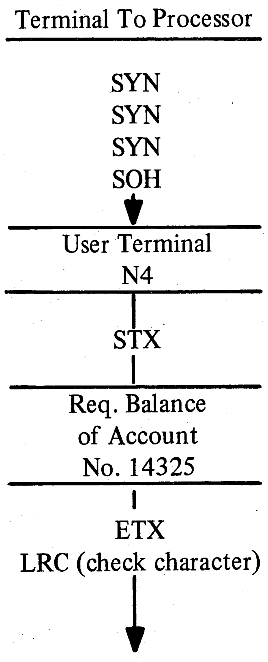
Character	Meaning	Function
SYN	Synchronizing signal	Establish character framing
SOH	Start of heading signal	Precedes block message heading characters
STX	Start of text signal	Precedes block of text characters
ETX	End of text signal	Terminates a block of characters started with STX
ACK	Acknowledge signal	*Affirmative acknowledgment of message received
NAK	Negative acknowledge signal	*Negative acknowledgment of message received

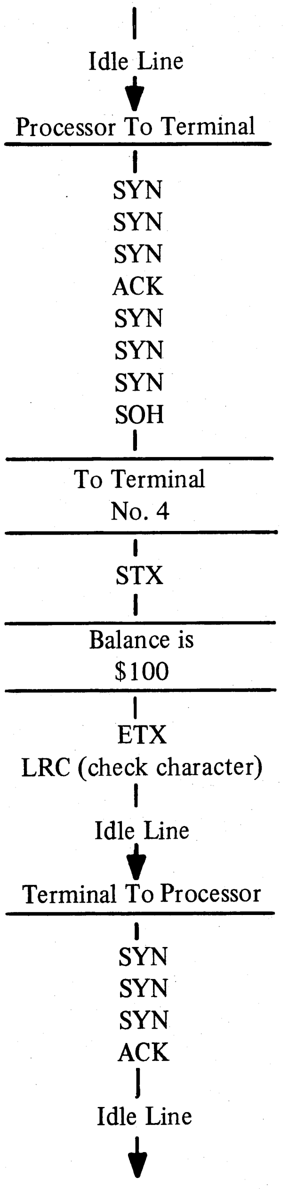
*ACK and NAK are sent by the station that received the message to the station that originated the message.

It is not the purpose of this manual to discuss the format for synchronous communication in detail. However, a brief description of these formats is outlined below to facilitate the reader's understanding of synchronous interface design.

Because the synchronous transmission technique provides only bit recovery timing, there must be a way to establish character framing and message framing. This is accomplished by using codes (usually ASCII) that are assigned for synchronous message formatting purposes. Representative message codes are listed in Table 1-1.

A typical message that might be sent between two devices (a terminal and a processor) follows.





1.2.2.2 Computer Application – Electronic computers are often connected into communication systems to help transmit and process digital data. By using computer systems to concentrate data from many low-speed terminals over one voice grade facility, significant improvements can be made in the efficiency of a data communication system. Since most long-range communication systems are connected through common carrier facilities, a communication system using a computer should be interfaced to the correct type facility. There are two basic types of common carrier facilities to which computers must be interfaced: asynchronous serial and synchronous serial. We have already pointed out the advantages and disadvantages of these two types of facilities. Based on these advantages and disadvantages, Table 1-2 shows typical speeds and applications of these two techniques.

As shown in Table 1-2, there are three basic communication applications to be solved by the computer communications engineer:

Low speed terminal equipment, such as Teletypes.
Medium speed terminal equipment.
Intercomputer communications.

1.3 GENERAL DESCRIPTION

The DU11 interface is a single line, program controlled, double-buffered communication interface. It provides serial to parallel and parallel to serial data conversion, EIA* to TTL (transistor-transistor logic) and TTL to EIA voltage level conversion and modem control for full or half duplex communication systems.

The DU11 is compatible with all PDP-11 family computers and is available in two models. Model DU11-DA is the basic version and is completely contained on the M7822 module. The basic version is compatible with the Bell 201 synchronous modem or equivalent. Model DU11-EA is simply the basic version adapted to current mode operation. The DU11-EA version consists of the basic M7822 module plus the DF11-G current mode converter. The DU11-EA is compatible with the Bell 303, wide band, synchronous modem or equivalent. A typical communication system using the DU11 is shown in Figure 1-3.

Interface operation is completely program controlled. The mode of operation (synchronous or isochronous), character length (5, 6, 7, or 8 bits plus parity if selected), parity enable and sense (odd or even), sync character configuration, and duplex mode (full or half) are all selected via the program.

1.4 PHYSICAL DESCRIPTION

The DU11 interface is completely contained on a single M7822 Quad Integrated Circuit module (Figure 1-4). This module can be mounted easily in the PDP-11 processor small peripheral controller slot (exceptions noted in Chapter 2) or in one of four slots in a DD11-A or DD11-B peripheral mounting panel.

All DU11 operating power is provided by the mounting panel in which it is installed. The power is taken from the mounting box power supply. For proper operation, the module requires +5 V @ 2.2 A, −15 V @ 0.17 A, and +15 V @ 0.07 A.

The mounting panel also connects the DU11 to the Unibus. All Unibus input/output signals enter and leave the module via the mounting panel pins. Refer to Chapter 2 for Unibus to mounting panel connection.

Table 1-2
Computer Communications Applications

Speed	Asynchronous	Synchronous
Low 0 to 300 baud	Electromechanical terminals such as keyboard printers and Teletypes.	Operations tend to be asynchronous at these speeds.
Medium 300 to 3000 baud	Unbuffered terminals such as paper tape readers and punches, card readers and line printers.	Buffered terminals such as displays, buffered card readers, and line printer configurations.
High 5000 baud and up	Not frequently used.	Intercomputer communications.

*EIA – A standardized set of signal characteristics (time duration, voltage, and current) specified by the Electronic Industries Association.

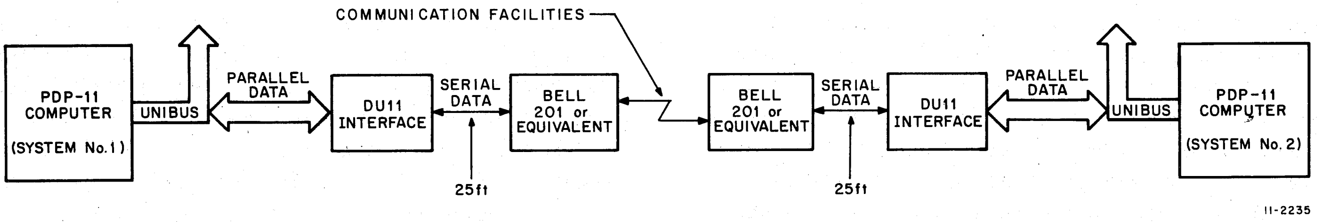


Figure 1-3 Typical Communication System Using the DU11 Interface

Major DU11 components are also labeled on Figure 1-4: the rocker switches which are used to select the interface Unibus address, the priority plug which determines the bus request (BR) priority level of the interface (BR5 plug normally installed at factory), the SAR (receiver) and SAT (transmitter) chips, and jumpers W2, W4–W6, and W9–W16 (a complete description of the jumpers is provided in Paragraph 2.1.4).

1.5 SPECIFICATIONS

Environmental, electrical, and performance specifications for the DU11 are contained in the following paragraphs.

1.5.1 Environmental

Ambient temperature
10° to 50° C (50° to 122° F)

Relative humidity
20% to 95% (without condensation)

1.5.2 Electrical

DC voltage requirements
+5 V @ 2.2 A
- 15 V @ 0.17 A
+15 V @ 0.07 A

Electrical Characteristics
Electrical characteristics of this interface meet EIA standard RS-232C and PDP-11 Unibus Interface specifications.

1.5.3 Performance

The following paragraphs discuss the baud rate limitations of the DU11 and related program response time.

1.5.3.1 Baud Rates for Synchronous Communications – EIA/CCITT* baud rate (10K baud maximum) is limited by modem and data set interface level converters. Current mode operation (100K baud maximum) is possible only with the DU11-EA. Current mode speed is limited by DU11 logic.

Even though the DU11 can receive and transmit information at such a high rate, it may, in most cases, be impractical. Since the service of the data buffers relies solely on the program, little time if any would be left for other events. This problem would be compounded if the interface were operating in full duplex mode.

1.5.3.2 Baud Rates for Isochronous Communications – EIA/CCITT baud rate (10K baud maximum) is limited by data set interface level converters. Current mode operation baud rate (100K baud maximum) is limited by DU11 logic.

1.6 ENGINEERING DRAWINGS

A complete set of engineering drawings entitled DU11 Line Interface, Engineering Drawings is provided with each interface. The general logic symbols used on these drawings are described in the *DEC Logic Handbook,* 1972. Specific symbols and conventions are also included in the PDP-11 system manuals. The following paragraphs describe the signal nomenclature conventions used in the drawing set.

1.6.1 Basic Signal Names

Signal names in the DU11 print set are in the following basic form:

SOURCE SIGNAL NAME POLARITY

SOURCE indicates the drawing number of the print from which the signal originates. The drawing number of a print is located in the lower right-hand corner of the print title block (D1, D2, D3, D4, D5, and D6).

*CCITT – The Consultive Committee International Telegraph and Telephone is an advisory committee established under the United Nations to recommend worldwide standards.

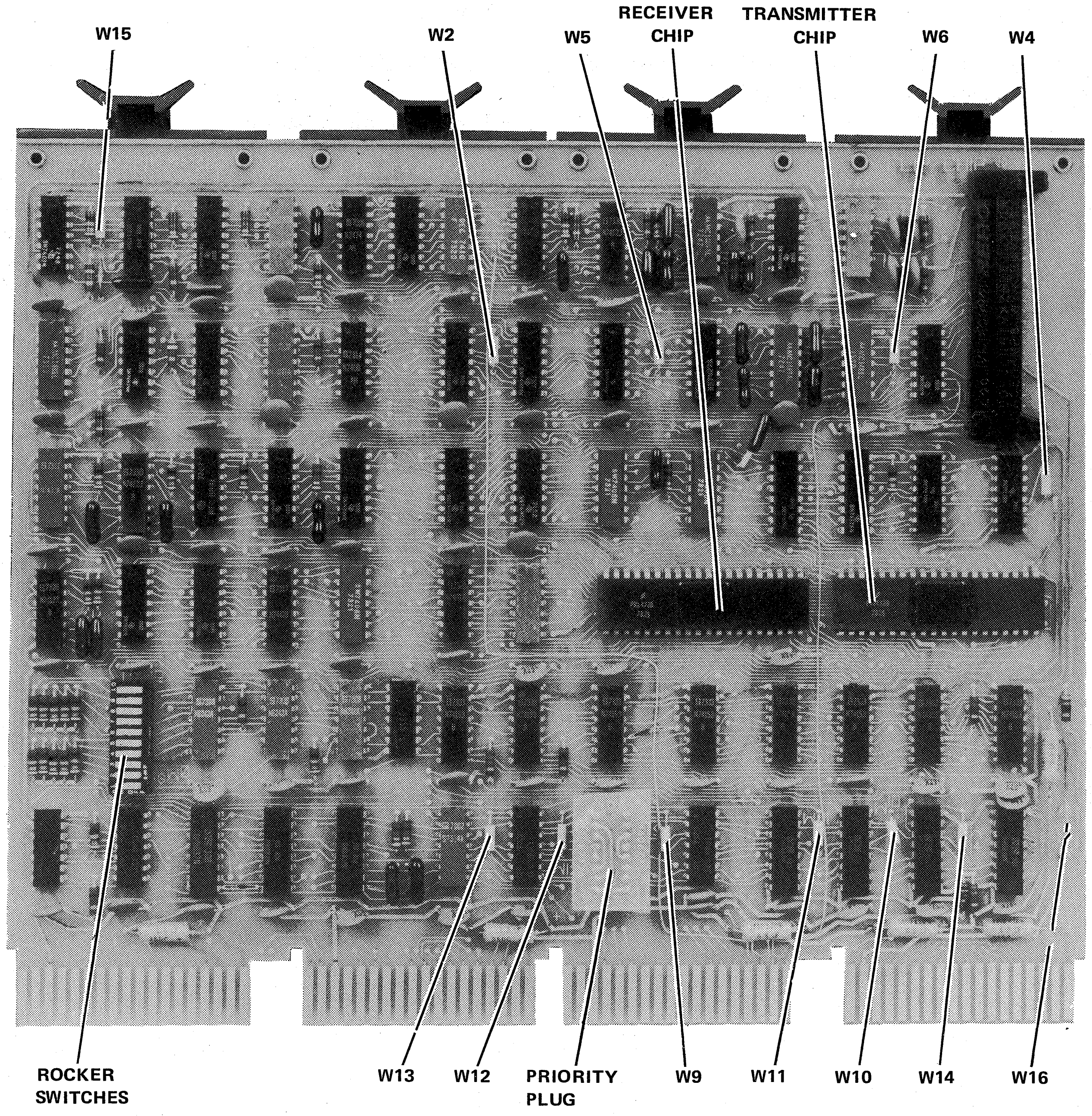


Figure 1-4 DU11 Major Components

SIGNAL NAME is the proper name of the signal. The names used on the print set are also used in this manual for correlation between the two.

POLARITY is either H or L to indicate the voltage level of the signal: H means +3 V; L means ground.

For example, the signal

D5- TX DONE H

originates on sheet 5 of the engineering drawings and is read, "When TX DONE is true, this signal is at +3 V."

Unibus signal lines do not carry a SOURCE indicator. These signal names represent a bidirectional wire-ORed bus; as a result, multiple sources for a particular bus signal exist. Each Unibus signal name is prefixed with the word BUS.

Interface signals fed to or received from the Bell 201 modem via the Berg connector on the M7822 module are preceded by the jack and pin number in parentheses:

(J1-DD) EIA DATA TERM RDY (This signal is shown on engineering drawing D6.)

Interface signals fed to or received from the Bell 303 modem via the mounting panel backpanel wiring and DF11-G level converter are preceded by the M7822 module pin number:

AF1 D6-DTR (1) H

1.6.2 Flip-Flop Signal Names

Flip-flop signal names add an extra dimension. Although flip-flops have only two outputs, four signal names are possible (Figure 1-5). The two real outputs are RX DONE (1) H on pin 5 and RX DONE (0) H on pin 6. The two additional outputs are simply the real outputs reidentified. RX DONE (1) L is electrically the same as RX DONE (0) H, and RX DONE (0) L is electrically the same as RX DONE (1) H.

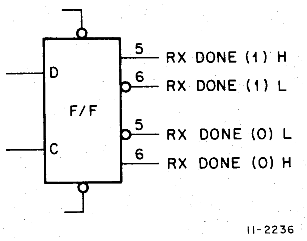


Figure 1-5 Flip-Flop Signal Names

CHAPTER 2
INSTALLATION

2.1 INSTALLATION

2.1.1 Mounting the DU11 in the Computer

There are two DU11 installation configurations:

a. The standard configuration, in which the DU11-DA interfaces with the Bell 201 synchronous modem or equivalent.

b. The current mode configuration, in which the DU11-EA interfaces with the Bell wide band 303 modem or equivalent.

2.1.1.1 Standard Configuration – In this configuration, the DU11-DA can be mounted in the small peripheral controller slot in the PDP-11/05, 10, 35, 40, 45, and 50 processors or in any one of four slots in the DD11-A or DD11-B peripheral mounting panels (Figure 2-1). The DD11-A mounting panel (Figure 2-2) is used in the PDP-11/15 and 20 computers, while the DD11-B (Figure 2-3) is used in the PDP-11/05, 10, 35, 40, 45, and 50 computers.

NOTE

The DU11-DA cannot be mounted in the small peripheral controller slot in the PDP-11/15 and 20 processors.

DD11-A and DD11-B mounting requirements are somewhat different. When using the DD11-B mounting panel, the DU11 is simply installed in the mounting panel; however, when using the DD11-A, jumper W16 (engineering drawing D1) and module G8000 must also be installed. Jumper W16 bypasses a voltage dropping resistor and the G8000 module converts the full-wave rectified +8 V/rms mounting panel input signal to a positive dc voltage which is used to drive the EIA level converters. To install the G8000 module, proceed as follows:

1. Install the G8000 module in slot A02 of the DD11-A.
2. Connect a wire between A03V2 and A02V2.
3. Connect a wire between A02N2 and CXXU1, where XX is to slot location of the M7822 module.

NOTE

Jumper W16 must not be installed if the DU11 is being installed in the DD11-B mounting panel.

2.1.1.2 Current Mode Configuration – In this configuration, the DU11-EA must be installed in the DD11-B mounting panel as shown in Figures 2-4 and 2-5.

NOTE

The DD11-A mounting panel cannot be used in the current mode configuration because it will not accommodate the DF11-G level converter.

2.1.2 Installing the Modem Cable Harness

A different cable is required to connect the DU11 to the Bell 201 modem than to the Bell 303 modem. The BC05C-25 cable harness (Figure 2-6) is used for the DU11-DA configuration; the BC01W-25 cable harness (Figure 2-7) is used for the DU11-EA configuration.

To install the cables, refer to Figure 2-8 and proceed as follows:

1. Position the Berg connector such that the connector name and pin number markings are visible and mate it fully and squarely with the Berg connector on the DU11 module (DU11-DA configuration) or the DF11-G connector module (DU11-EA configuration).

2. Align the Cinch connector (DU11-DA configuration) or the Burndy connector (DU11-EA configuration) to the receptacle located on the rear of the modem.

3. Mate the connector and tighten the two hold-down screws using a screwdriver.

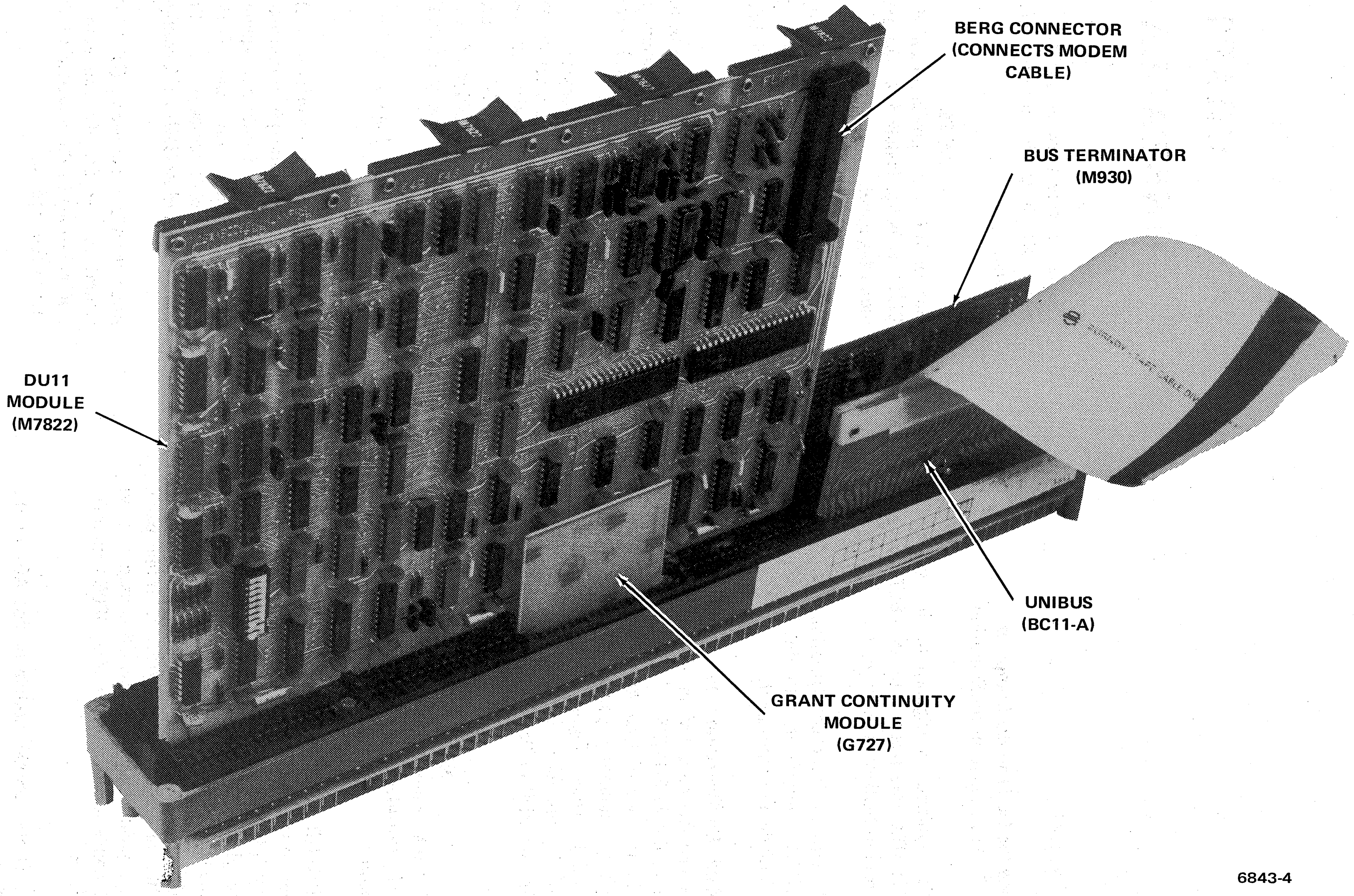
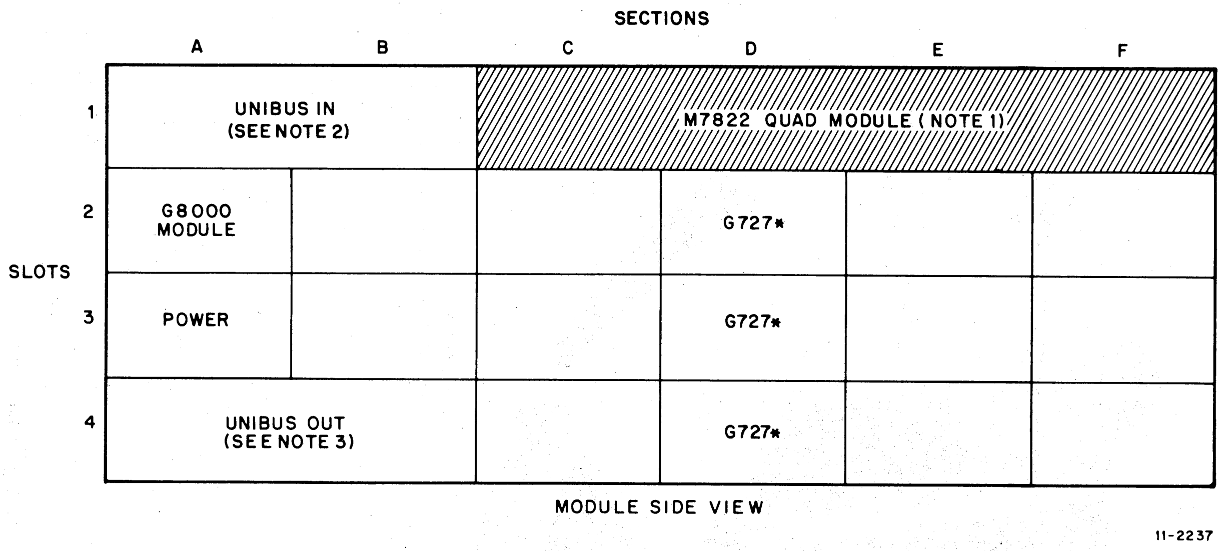


Figure 2-1 Standard Configuration (DU11-DA) Using DD11-B Mounting Panel

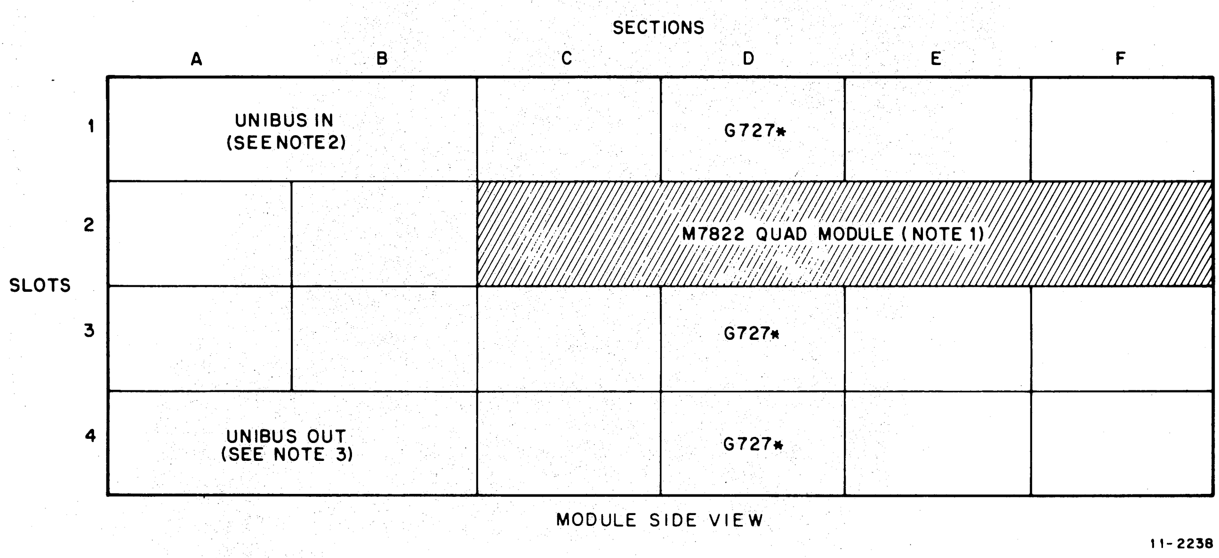


*GRANT Continuity Module (G727) must be installed in each slot that does not receive an interface logic module.

NOTES:

1. Can be mounted in slots 1, 2, 3, or 4
2. Can be M920 or BC11-A
3. Can be M920, BC11-A or M930

Figure 2-2 DU11-DA (M7822 Module) Mounted in DD11-A



*Grant Continuity Module (G727) must be installed in each slot that does not receive an interface logic module.

NOTES:

1. Can be mounted in slots 1, 2, 3, or 4
2. Can be M920 or BC11-A
3. Can be M920, BC11-A or M930

Figure 2-3 DU11-DA (M7822 Module) Mounted in DD11-B

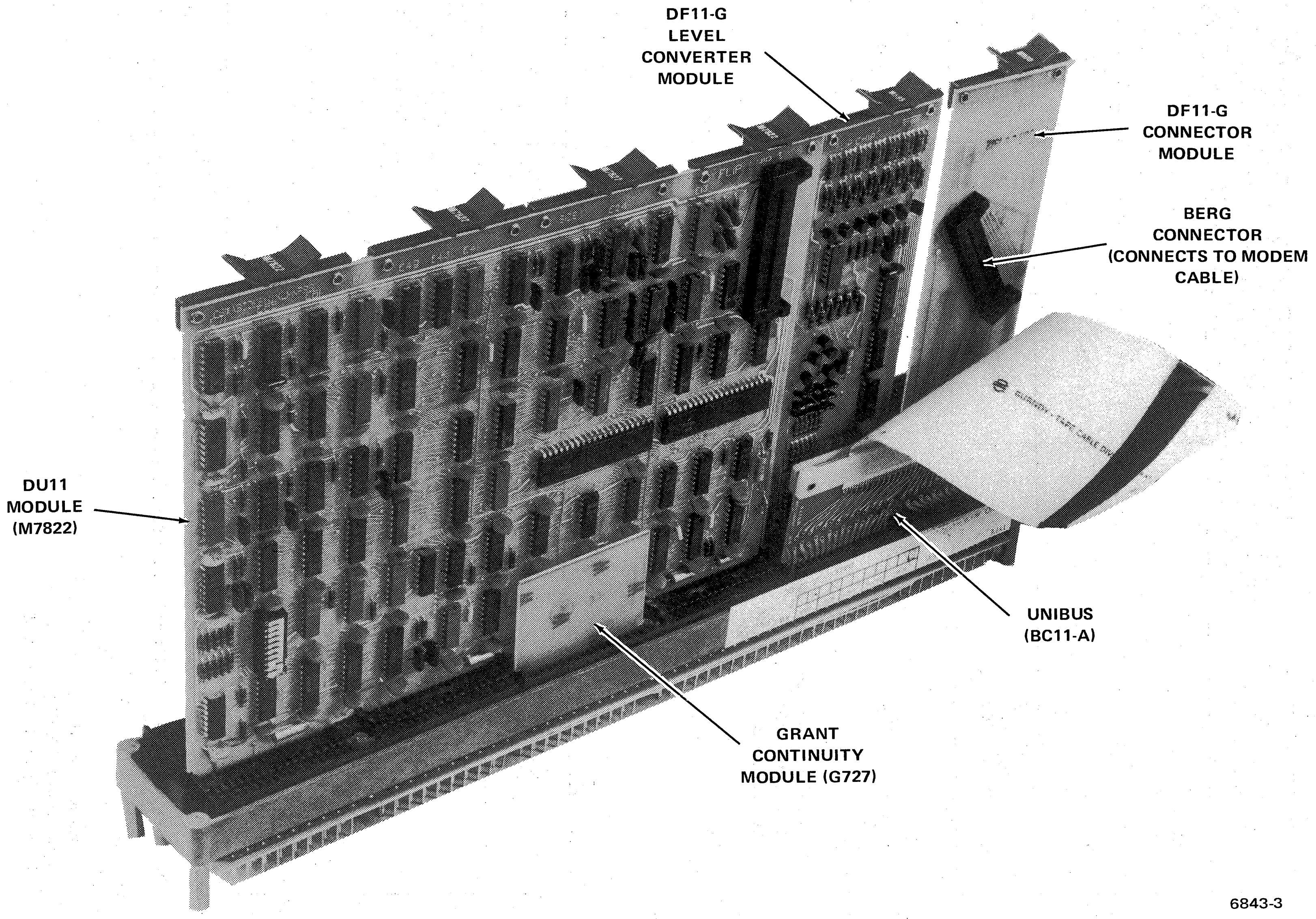
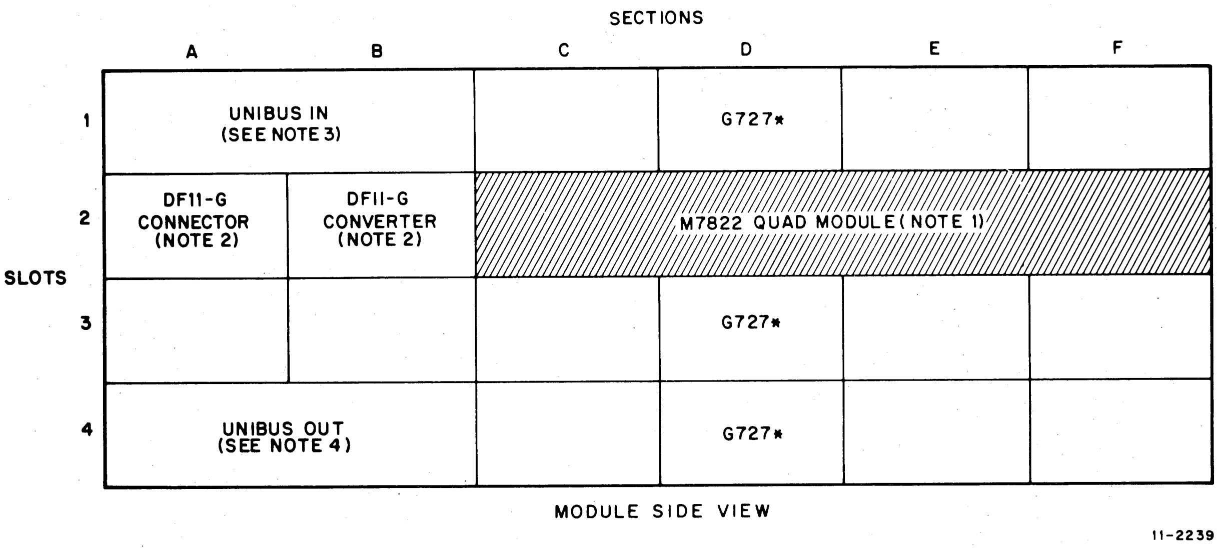


Figure 2-4 Current Mode Configuration (DU11-EA) Using DD11-B Mounting Panel



*Grant Continuity Module (G727) must be installed in each slot that does not receive an interface logic module.

NOTES:

1. Can only be mounted in slots 2 or 3
2. The DF11-G connector and converter must be mounted in the same slot as the M7822 module.
3. Can be M920 or BC11-A
4. Can be M920, BC11-A or M930

Figure 2-5 DU11-EA (M7822 Module and DF11-G Converter) Mounted in DD11-B

2.1.3 Unibus and Interrupt Vector Address Assignments

The Unibus and interrupt vector addresses must be determined prior to operating the DU11. The Unibus address is switch selectable; the interrupt vector addresses are jumper selectable (Figure 1-4 for physical location).

The Unibus address (also referred to as the device address) is controlled by ten rocker switches located in the address selection and mode control logic. The position of these switches determines the required address state (0 or 1) of bus address bits 12–03. If a rocker switch is set to ON, the switch contacts are closed and an address state of 0 is required on the related address bit to address the DU11. Hence, electrically the DU11 can have any device address within the range of 760000 to 777777; however, Digital Equipment Corporation software requires that the device address fall within the floating address range of 760010 to 763776. Refer to Appendix B for a complete discussion of DU11 address assignments.

NOTE

If a device address is selected which falls outside the floating address range, the software must be modified accordingly.

The interrupt vector addresses are also floating and are established at the factory in accordance with the vector addressing scheme described in Appendix B. If it is necessary to change the vector address, simply change jumpers W9–W14 as required. Jumpers are cut to obtain a logical zero. Jumpers W9–W14 are located in the interrupt control logic (engineering drawing D4). These jumpers control vector address bits 08–03; hence, vector addresses can be generated within the range of 000 to 774; however, software requires that the vector address fall within the floating address range of 300 to 777.

NOTE

If a vector address is selected which falls outside the floating address range, the software must be modified accordingly.

2.1.4 Jumper Assignments

Jumpers are used at various points in the DU11 circuitry to increase flexibility and to meet the floating vector address requirement described in Appendix B. For a complete description of the DU11 jumpers, refer to Table 2-1.

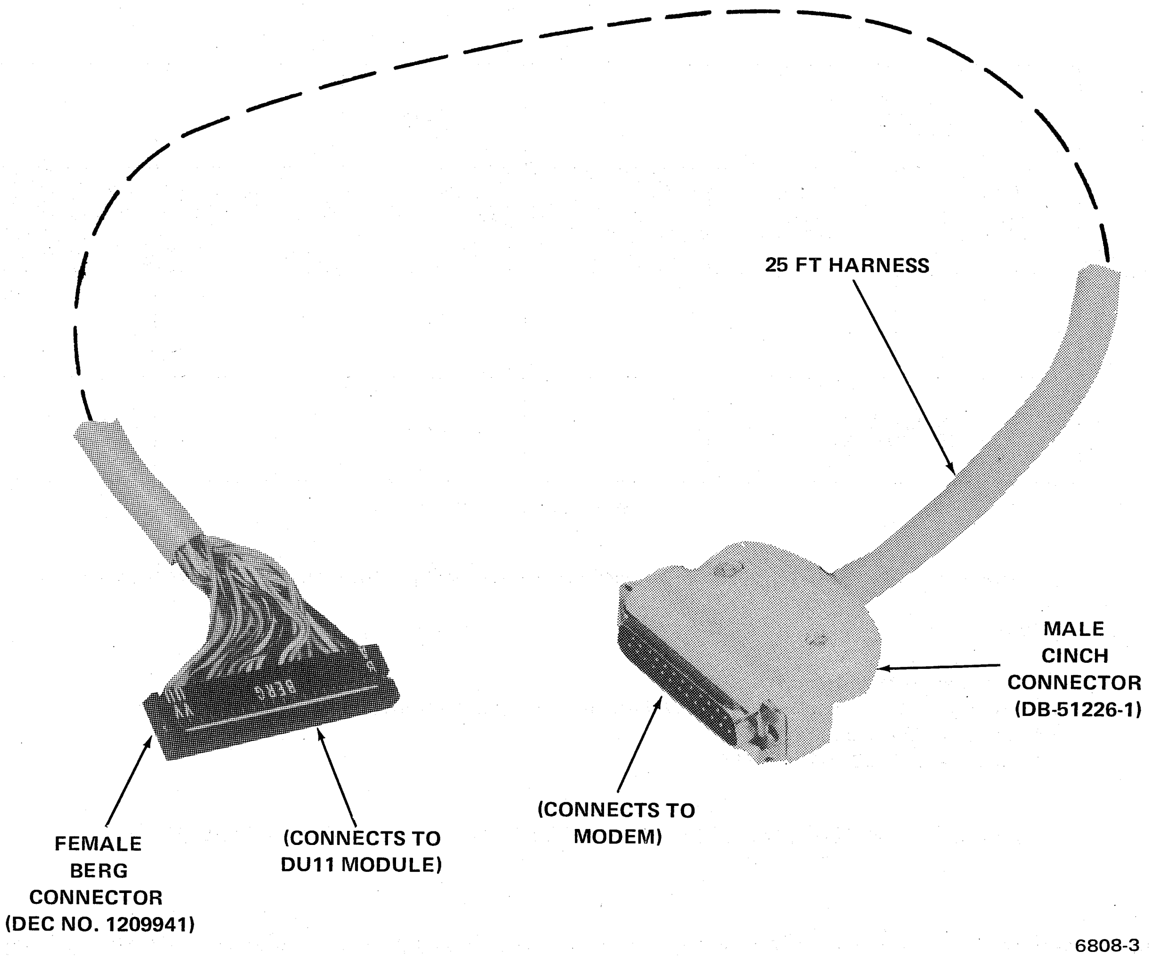


Figure 2-6 BC05C-25 Cable Harness Used to Connect DU11-DA to Bell 201 Modem

2.1.5 Priority Assignment

The priority level is determined by the priority plug located on the DU11 module. The DU11 normally has a priority level of BR5. However, the priority may be changed by simply replacing the BR5 plug with a plug wired for a different priority level.

NOTE
If the priority level is changed, the software must be modified accordingly.

2.2 INITIAL TESTING

The DU11 must be tested prior to placing the unit into operation. For initial test procedures, refer to the engineering specification, A-SP-DU11-0-4, which is provided with each DU11 delivered.

NOTE
Before running diagnostics on interface model DU11-DA, disconnect the modem cable (BC05C-25) from the rear of the modem and install the modem test connector as shown in Figure 2-9.

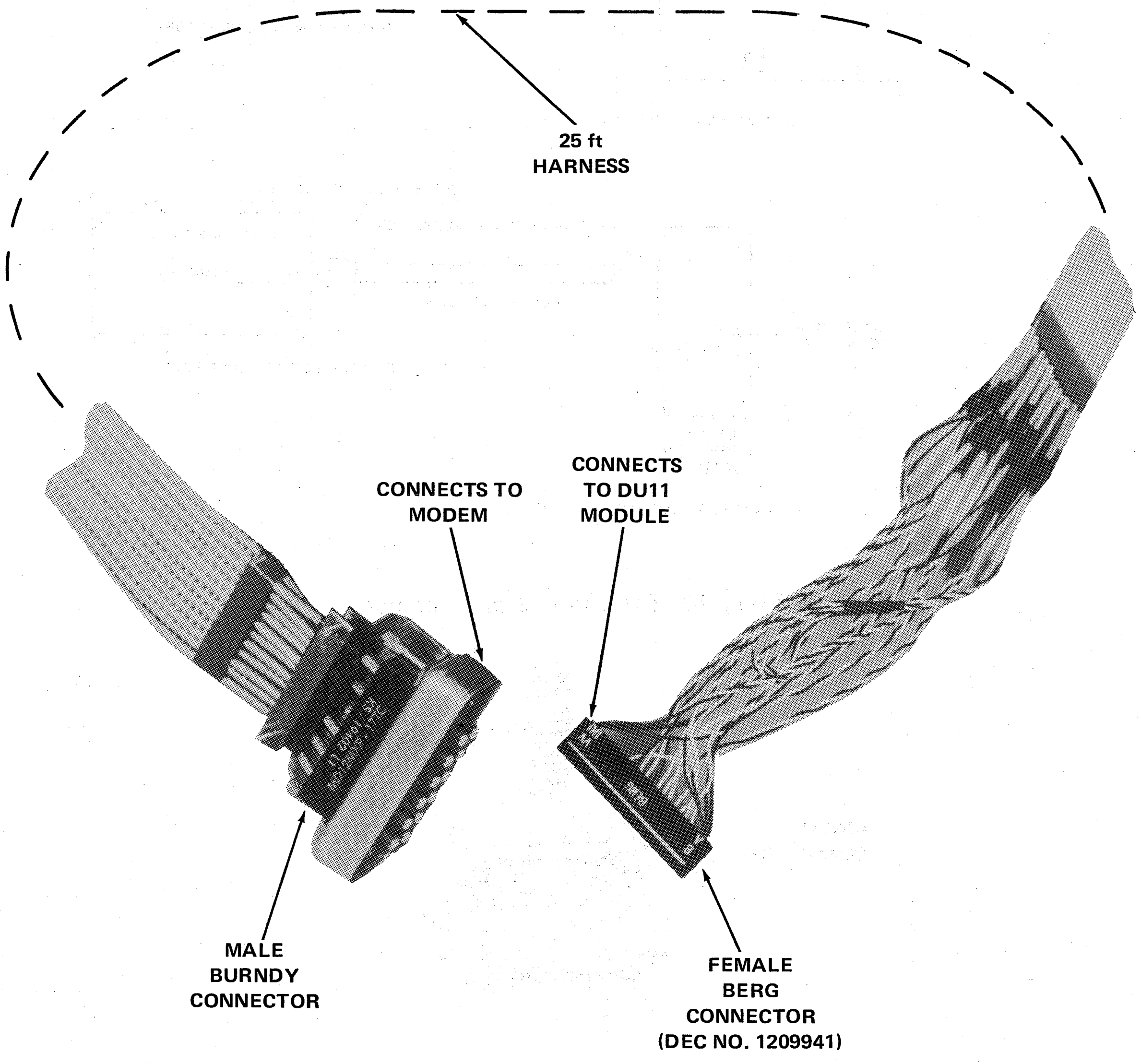


Figure 2-7 BC01W-25 Cable Harness Used to Connect DU11-EA to Bell 303 Modem

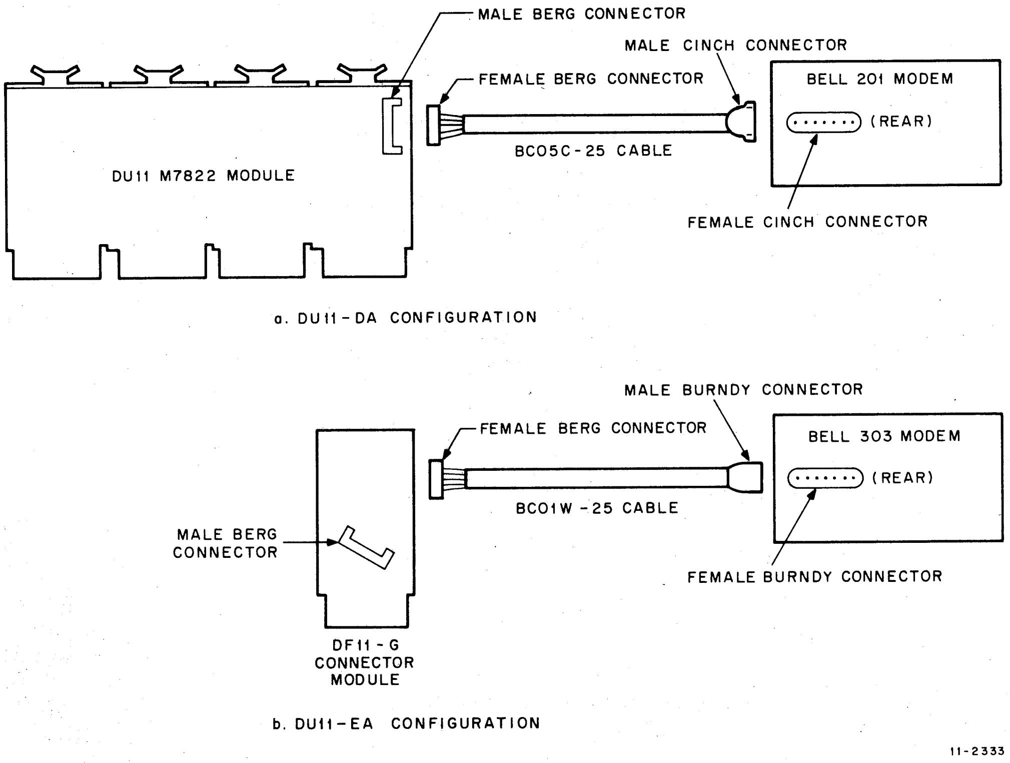


Figure 2-8 DU11 to Modem Connection

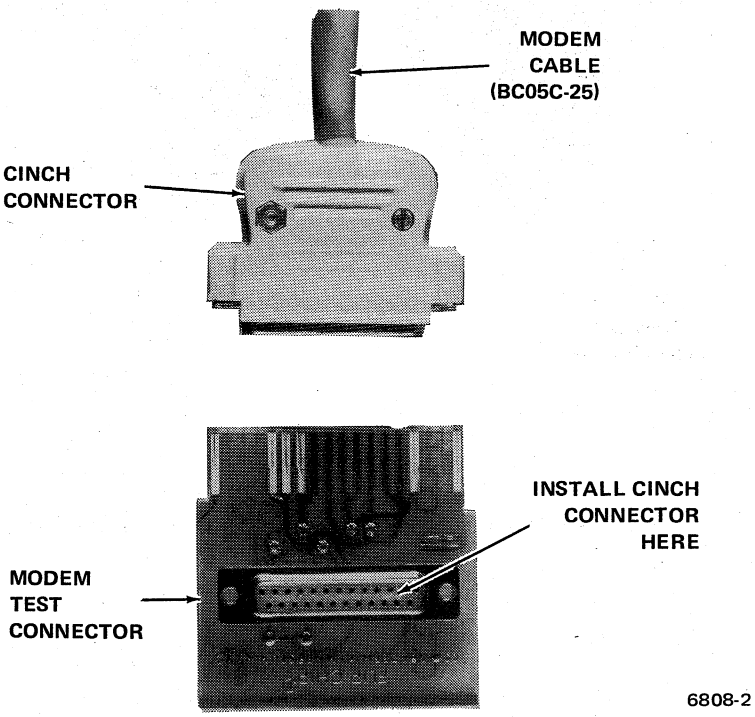


Figure 2-9 Modem Test Connection Installation

Table 2-1
Jumper Assignments

Jumper No. and Location	Normal Configuration	Function
W2/D5	Removed	This jumper may be installed to enable the receiver to synchronize internally upon receiving just one sync character, thereby negating the normal requirement of receiving two contiguous sync characters to achieve synchronization in the internal synchronous mode.
W4/D6	Installed	This jumper may be removed to disable CLR OPT L (Clear Option), thereby preventing clearing of bits 03, 02, and 01 in the RXCSR (see Chapter 3 for bit descriptions and Chapter 4 for signal functions).
W5, W6/D6	Installed	These jumpers may be removed to disconnect the secondary data channel between the modem and the DU11. Removed at customers request.
W9–W14/D4	Floating	These jumpers control the receiver and transmitter interrupt vector address (Paragraph 2.1.3) bits: **Jumper** **Address Bit** W9 BUS D03 W10 BUS D04 W11 BUS D05 W12 BUS D06 W13 BUS D07 W14 BUS D08
W15/D4	Installed	This jumper may be removed to inhibit the BUS NPR L input to the interrupt control logic. (Removed only if PDP-11/20 processor is used without KH option.)
W16/D1	Removed	This jumper must be installed if the DU11 is mounted in a DD11-A peripheral mounting panel (Paragraph 2.1.1.1).

CHAPTER 3
DEVICE REGISTERS AND INTERRUPT REQUESTS

3.1 SCOPE

This chapter provides a complete description of the DU11 device registers and the interrupt requests employed to service those registers.

3.2 DEVICE REGISTERS

All software control of the DU11 is performed by means of five device registers. These registers have been assigned bus addresses and can be read or loaded (with the exceptions noted) using any PDP-11 instruction referring to their addresses. Address assignments can be changed via the rocker switches to correspond to any address within the floating address range of 160010 to 163776.

3.2.1 Register Address Assignments

The five device registers and associated DU11 addresses are listed in Table 3-1.

3.2.2 Register Title and Bit Assignments

Each of the five device registers plays a specific role in controlling and monitoring DU11 operation. Register titles, bit titles, and read/write capability labeling are intended to facilitate the programmer's understanding of the purpose of each register relative to interface operation and to simplify software preparation.

3.2.2.1 Title Assignments – Register titles and functions are listed below:

a. RXCSR – programmed and monitored (read/write) to control the RCVR (receiver) portion of the interface; to communicate interface status, requests, and supervisory data to the modem; and to monitor status and supervisory data inputs from the modem.

b. RXDBUF – monitored (read only) to detect interface RCVR status flags and RCVR parallel data outputs.

c. PARCSR – programmed (write only) to establish the overall operating parameters of the DU11, i.e., the mode of operation (synchronous or isochronous), word length (5, 6, 7, or 8 bits plus parity), parity (enabled or disabled), parity sense (odd or even), and sync character configuration.

Table 3-1
DU11 Register Address Assignments

Register	Mnemonic	Address	Program Capability
Receiver Status Register	RXCSR	16XXX0	Read/Write
Receiver Data Buffer	RXDBUF	16XXX2	Read Only
Parameter Status Register	PARCSR	16XXX2	Write Only
Transmitter Status Register	TXCSR	16XXX4	Read/Write
Transmitter Data Buffer	TXDBUF	16XXX6	Write Only

XXX = Selected in accordance with floating device address scheme described in Appendix B.

d. TXCSR – programmed and monitored (read/write) to control the XMTR (transmitter) portion of the interface, to control the resetting and initialization of the interface, and to control and monitor the maintenance mode operation of the interface.

e. TXDBUF – programmed (write only) to provide parallel data to the interface XMTR for serial transmission to the modem.

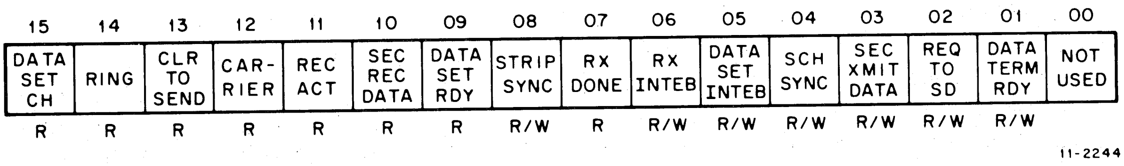
3.2.2.2 Bit Assignments – The bit names indicate the function of the bit. The bits that are defined as "not used" or "write-only" are always read as 0. In the same respect, attempts to program the "not used" bits or "read-only" bits have no effect on the bit.

The following figures and tables describe register content. Figures 3-1 through 3-5 illustrate the register formats. Tables 3-2 through 3-6 list bit descriptions.

The mnemonic INIT is used frequently in the following tables and refers to the initialization signal generated by the processor. The processor will issue an INIT signal for any one of the following conditions:

a. A programmed RESET instruction is processed.
b. The processor START switch is pressed.
c. The power fail sequence occurs.

During a power fail sequence, INIT is asserted when power is going down and again when power is coming up.



15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DATA SET CH	RING	CLR TO SEND	CAR-RIER	REC ACT	SEC REC DATA	DATA SET RDY	STRIP SYNC	RX DONE	RX INTEB	DATA SET INTEB	SCH SYNC	SEC XMIT DATA	REQ TO SD	DATA TERM RDY	NOT USED
R	R	R	R	R	R	R	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	

Figure 3-1 Receiver Status Register (RXCSR)

Table 3-2
Receiver Status Register Bit Description

Bit	Name	Description
15	DAT SET CH (Data Set Change)	When set, this bit indicates a modem status change. This bit is set by a transition of any of the following lines: • Ring • Clear To Send • Carrier • Secondary Received Data • Data Set Ready If bit 05 of this register is set, the setting of this bit will cause a RCVR interrupt. Read-only bit; cleared by INIT, Master Reset, and the DTI SEL 0 (RXCSR read strobe).
14	RING (Ring)	This bit reflects the state of the modem Ring line. When set, this bit indicates that a Ring signal is being received from the modem. Read-only bit.

Table 3-2 (Cont)
Receiver Status Register Bit Description

Bit	Name	Description
13	CLR TO SD (Clear to Send)	This bit reflects the state of the Clear to Send line from the modem. When set, this bit indicates that the modem is on and ready to accept data from the interface for transmission. Read-only bit.
12	CARRIER (Carrier)	This bit reflects the state of the modem carrier. When set, this bit indicates the Carrier is up. Read-only bit.
11	REC ACT (Receiver Active)	When the internal synchronous mode is selected, this bit is set when the proper number of contiguous sync characters (either 1 or 2, normally set for 2) have been received. If external synchronous or isochronous mode is selected, this bit follows the state of the Search Sync bit (bit 04 of this register). See Paragraph 5.3 for RCVR synchronization information. Read-only; cleared by INIT, Master Reset, and SCH SYNC (1) H (Search Sync) making 1 to 0 transition.
10	SEC RCV DAT (Secondary Receive Data)	This bit reflects the state of the Secondary Receive Data line from the modem. This bit provides a receive channel for supervisory data from the modem to the processor. Read-only bit.
09	DAT SET RDY (Data Set Ready)	This bit reflects the state of the Data Set Ready line from the modem. When set, this bit indicates that the modem is powered up and ready to transmit and receive data. Read-only bit.
08	STRIP SYNC (Strip Sync)	This bit determines whether sync characters received from the modem are to be presented to the program for reading. When this bit is set, receive characters that match the contents of the Sync register do not cause a RCVR interrupt provided no errors are detected, i.e., bit 15 of the RXDBUF is clear. Read/write bit; cleared by INIT and Master Reset.
07	RX DONE (Receiver Done)	This bit is set when synchronization has been achieved and a character has been loaded into the RXDBUF, provided the STRIP SYNC bit is not set. If the STRIP SYNC bit is set and the received character is a sync character without errors, i.e., bit 15 of the RXDBUF is clear, this bit will not be set. When set, this bit will cause a RCVR interrupt request provided bit 06 of this register is set. Read-only bit; cleared by INIT, Master Reset, and the DTI SEL 2 (RXDBUF read strobe).

Table 3-2 (Cont)
Receiver Status Register Bit Description

Bit	Name	Description
06	RX INTEB (Receiver Interrupt Enable)	When set, allows a RCVR interrupt request to be generated when the RX DONE bit is set. Read/write bit; cleared by INIT and Master Reset.
05	DAT SET INTEB (Data Set Interrupt Enable)	When set, allows a RCVR interrupt request to be generated when the DAT SET CH bit is set. Read/write bit; cleared by INIT or Master Reset.
04	SCH SYNC (Search Sync)	When set in the internal synchronous mode, enables the RCVR synchronization logic and causes the RCVR to start comparing incoming data bits to the contents of the Sync register in an attempt to recognize a sync character. When set in the isochronous mode, enables the RX DONE flag generation logic. When set in the external synchronous mode, enables the RX DONE flag generation logic and causes the RCVR to start framing incoming characters. Read/write bit; cleared by INIT and Master Reset.
03	SEC XMIT (Secondary Transmit Data)	This bit reflects the state of the Secondary Transmit Data line to the modem. This bit provides a transmit channel for supervisory data from the modem to the processor. Read/write bit; optionally cleared by INIT or Master Reset.
02	REQ TO SD (Request to Send)	When set, this bit causes the Request to Send line to the modem to be asserted. The Request to Send line is a control lead to the modem. This line must be asserted before the interface can transmit data to the modem. Read/write bit; optionally cleared by INIT and Master Reset.
01	DATA TERM RDY (Data Terminal Ready)	When set, this bit indicates the interface is powered up, programmed, and ready to receive data from the modem. Setting this bit causes the Data Terminal Ready line to the modem to be asserted. The Data Terminal Ready line is a control lead for the modem communication channel. When asserted, it permits the interface to be connected to the channel. Read/write bit; optionally cleared by INIT and Master Reset.

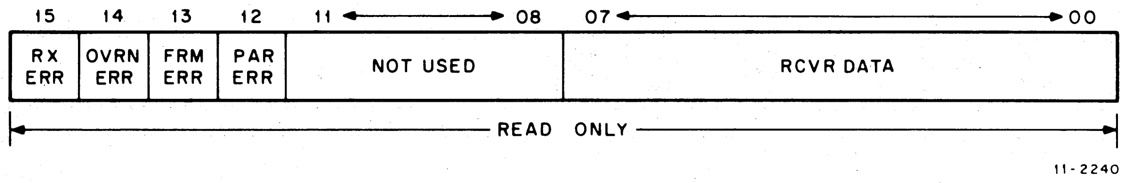


Figure 3-2 Receiver Data Buffer (RXDBUF)

Table 3-3
Receiver Data Buffer Bit Description

Bit	Name	Description
15	RX ERR (Receiver Error)	This bit is set whenever one of the three receiver error bits is set (logical OR of bits 14, 13, and 12). Read-only bit; cleared only when bits 14, 13, and 12 are cleared.
14	OVRN ERR (Overrun Error)	When set, this bit indicates that the processor has failed to service the RX DONE flag within the time required to load another character into the RXDBUF, i.e., (1/baud rate) X (bits per character) seconds. Hence, the previous character was over-written (lost). Read-only bit; cleared by INIT, Master Reset, and DTI SEL 2 (RXDBUF read strobe).
13	FRM ERR (Framing Error)	When set, indicates that character received was not followed by a valid STOP bit. This error only occurs in the isochronous mode of operation. Read-only bit; cleared by INIT, Master Reset, and DTI SEL 2.
12	PAR ERR (Parity Error)	When set, indicates that the parity of the received character does not agree with the parity programmed (odd or even). If parity is not programmed, this bit is always cleared. Read-only bit; cleared by INIT, Master Reset, and DTI SEL 2.
07–00	RCVR DATA (Receiver Data)	This register holds the received character for transfer to the program. The buffer is right justified for 5, 6, 7, or 8 bits. If parity is received it is also loaded into the buffer at the next vacant higher order bit position. Therefore, if a 5-bit character plus parity is framed by the RCVR, the parity bit would be loaded into bit position 05 in the RXDBUF and presented to the program for reading. If an 8-bit character plus parity is framed, the parity bit would not be presented to the program for reading. Read-only buffer; cannot be cleared, INIT or Master Reset sets the buffer to all ones.

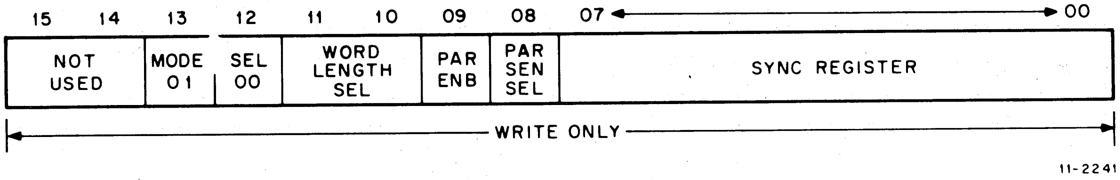


Figure 3-3 Parameter Status Register (PARCSR)

Table 3-4
Parameter Status Register Bit Description

<table>
<tr><th>Bit</th><th>Name</th><th>Description</th></tr>
<tr><td>13 and 12</td><td>MODE SEL
(Mode Select)</td><td>These bits control the mode of operation. Modes are selected as follows:
<table>
<tr><th>Mode</th><th>Bit 13</th><th>Bit 12</th></tr>
<tr><td>Internal Synchronous</td><td>1</td><td>1</td></tr>
<tr><td>External Synchronous</td><td>1</td><td>0</td></tr>
<tr><td>Isochronous</td><td>0</td><td>0</td></tr>
</table>
Any other mode select bit combinations will produce errors in the interface.
Write-only bits.</td></tr>
<tr><td>11 and 10</td><td>WORD LEN SEL
(Word Length Select)</td><td>These bits control the length of characters received and transmitted by interface. Word length (not including parity) is selected as follows:
<table>
<tr><th>Bits per Character</th><th>Bit 11</th><th>Bit 10</th></tr>
<tr><td>5</td><td>0</td><td>0</td></tr>
<tr><td>6</td><td>0</td><td>1</td></tr>
<tr><td>7</td><td>1</td><td>0</td></tr>
<tr><td>8</td><td>1</td><td>1</td></tr>
</table>
Write-only bits.</td></tr>
<tr><td>09</td><td>PAR ENB
(Parity Enable)</td><td>If this bit is set, parity will be generated by the XMTR and checked by the RCVR. If character length is less than eight bits, the parity bit is loaded into the RXDBUF for reading by the program. If bad parity is detected at the RCVR, the parity error flag is set (bit 12 of the RXDBUF).
Write-only bit.</td></tr>
</table>

Table 3-4 (Cont)
Parameter Status Register Bit Description

Bit	Name	Description
08	PAR SEN SEL (Parity Sense Select)	When the Parity Enable bit (bit 09 of this register) is set, the sense of the parity (odd or even) is controlled by this bit. When this bit is set, even parity is generated by the XMTR and checked for by the RCVR (the program does not have to provide a parity bit to the XMTR). When this bit is cleared, odd parity is generated and checked. Write-only bit.
07–00	Sync Register	This register contains the sync character. The sync character is used by the RCVR to detect received sync characters and thereby achieve synchronization. The sync character is used as a fill character by the XMTR when operating in the synchronous mode. Fill characters are transmitted when the program fails to provide characters to the XMTR fast enough to maintain continuous transmission, i.e., (1/baud rate) X (bits per character) seconds − 1/2 (bit time).

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DNA	MAINT DATA	SS CLK	MS 01	MS 00	RX INP	NOT USED	MST RST	TX DONE	TX INTEB	DNA INTEB	SEND	HALF DUP	NOT USED		BREAK
R	R/W	R/W	R/W	R/W	R		W	R	R/W	R/W	R/W	R/W			R/W

11-2242

Figure 3-4 Transmitter Status Register (TXCSR)

Table 3-5
Transmitter Status Register Bit Description

Bit	Name	Description
15	DNA (Data Not Available)	This bit is set by the XMTR when a fill character is transmitted. This applies only to the synchronous mode of operation and is caused by late program response to a TX DONE interrupt request. The processor response to TX DONE must be within (1/baud rate) X (bits per character) seconds − 1/2 (bit time). If not, the fill character is transmitted. If bit 05 of this register is set, setting this bit causes an XMTR interrupt request. Read-only bit; cleared by INIT, Master Reset, and DTI SEL 4 (TXCSR read strobe).

Table 3-5 (Cont)
Transmitter Status Register Bit Description

<table>
<tr><th>Bit</th><th>Name</th><th>Description</th></tr>
<tr><td>14</td><td>MAINT DATA
(Maintenance Data)</td><td>This bit is used in the internal loop and external loop maintenance modes by the diagnostic program to simulate an input to the RCVR. Refer to Chapter 6 for more detailed information on the use of this bit.

Read/write bit; cleared by INIT and Master Reset.</td></tr>
<tr><td>13</td><td>SS CLK
(Single Step Maintenance Clock)</td><td>This bit is used in the internal loop and external loop maintenance modes by the diagnostic program to simulate the XMTR and RCVR clocks. Refer to Chapter 6 for more detailed information on the use of this bit.

Read/write bit; cleared by INIT or Master Reset.</td></tr>
<tr><td>12 and 11</td><td>MS01/MS00
(Maintenance Mode Select 01 & 00)</td><td>These bits are used to select the normal mode of operation or one of three maintenance modes. Modes are selected as follows:
<table>
<tr><th>Mode</th><th>Bit 12</th><th>Bit 11</th></tr>
<tr><td>Normal</td><td>0</td><td>0</td></tr>
<tr><td>Internal Maintenance Loop</td><td>0</td><td>1</td></tr>
<tr><td>External Maintenance Loop</td><td>1</td><td>0</td></tr>
<tr><td>System Test</td><td>1</td><td>1</td></tr>
</table>
Refer to Chapter 6 for more detailed information on the maintenance modes and their use.

Read/write bits; cleared by INIT and Master Reset.</td></tr>
<tr><td>10</td><td>RX INP
(Receiver Input)</td><td>This bit monitors the RCVR input in the internal loop and external loop maintenance modes.

Read-only.</td></tr>
<tr><td>08</td><td>MSTRST
(Master Reset)</td><td>This bit is used to generate a CLR (Clear) pulse, which initializes the registers and the XMTR and RCVR and inhibits the BUS SSYN L (Slave Sync) signal. Refer to Chapter 4 for more detailed information on the effects of the CLR pulse.

Write-only.</td></tr>
<tr><td>07</td><td>TX DONE
(Transmitter Done)</td><td>This bit is set by INIT and Master Reset and when the first bit of the character contained in the XMTR register is placed on the XMTR output line. If bit 06 of this register is set when this bit is set, an XMTR interrupt request is generated.

Read-only bit; cleared by LD TXDBUF (TXDBUF load strobe).</td></tr>
</table>

Table 3-5 (Cont)
Transmitter Status Register Bit Description

Bit	Name	Description
06	TX INTEB (Transmitter Interrupt Enable)	When set, this bit allows an XMTR interrupt request to be generated by the TX DONE bit. Read/write bit; cleared by INIT and Master Reset.
05	DNA INTEB (Data Not Available Interrupt Enable)	When set, this bit allows a XMTR interrupt request to be generated by the DNA bit. Read/write bit; cleared by INIT and Master Reset.
04	SEND (Send)	When set, this bit enables the XMTR and transmission will start when a character is loaded into the TXDBUF. This bit must remain set until the entire message is transmitted. If not, transmission of the character currently in the XMTR register is completed and the XMTR will enter the idle state. Read/write bit; cleared by INIT and Master Reset.
03	HALF DUP (Half Duplex)	When this bit is set, operation will be in the half duplex mode. In this mode the RCVR is disabled whenever bit 04 of this register is set. Read/write bit; cleared by INIT and Master Reset.
00	BREAK (Break)	When this bit is set, the serial XMTR output is held in the space (constant LOW) condition; otherwise, operation is normal. This bit is used by the diagnostic program in the internal loop or external loop maintenance modes to inhibit the XMTR output while inputting data to the RCVR via bit 14 of this register. Read/write bit; cleared by INIT and Master Reset.

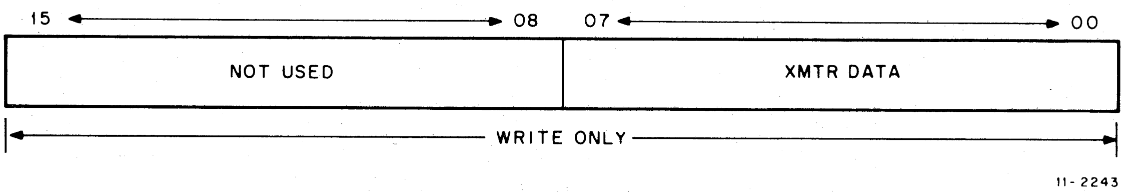


Figure 3-5 Transmitter Data Buffer (TXDBUF)

Table 3-6
Transmitter Data Buffer Bit Description

Bit	Name	Description
07–00	XMTR DATA (Transmitter Data)	This register is loaded by the program with the character to be transmitted. Character length is from 5 to 8 bits. The character is right-hand justified. If a parity bit is programmed, it is generated by the interface. Write-only bits; an INIT or Master Reset places all ones in this register.

3.3 INTERRUPT REQUESTS

The DU11 uses BR interrupts to gain control of the bus and cause a program interrupt, thereby causing the processor to branch to a subroutine.

The interface uses two interrupt vectors: one for the RCVR section and one for the XMTR section. If simultaneous RCVR and XMTR interrupt requests occur, the RCVR has priority.

Both the XMTR and RCVR sections of the interrupt control logic handle interrupt requests from two sources. A XMTR interrupt request is generated by the setting of the TX DONE bit or the DNA bit provided the TX INTEB and the DNA INTEB bits are set. A RCVR interrupt request is generated by setting the RX DONE bit or the DAT SET CH bit, provided the RX INTEB and DAT SET INTEB bits are set.

The standard DU11 interrupt priority level is BR5. However, the priority level can be changed by replacing the priority plug.

The DU11 interrupt vector addresses are floating. Floating vector addresses are used for all options and are assigned according to the scheme described in Appendix B. The vector addresses can be changed via jumpers W9–W14 in the interrupt control logic.

NOTE

If the DU11 priority plug or an interrupt vector address is changed, all DEC programs or other software referring to the standard priority level or interrupt vector addresses must also be changed.

CHAPTER 4
THEORY OF OPERATION

4.1 SCOPE

This chapter provides a detailed description of the DU11 interface. The description is provided in two parts: a functional description and a detailed logic description. Discussions make frequent reference to the DU11 Engineering Drawing Set provided with each DU11.

4.2 FUNCTIONAL DESCRIPTION

The DU11 is a single line, program controlled, double buffer communication interface. The purpose of the DU11 interface is to establish a data communication line between a PDP-11 computer (a parallel input/output device) and a Bell 201 or 301 modem (a serial input/output device). The interface is capable of:

a. handling synchronous and isochronous communication data

b. operating in half duplex or full duplex mode

c. handling variable length characters (5, 6, 7, or 8 bits plus parity)

d. generating a parity bit (odd or even) which is transmitted with the data character to the modem

e. verifying received character parity (odd or even)

f. inhibiting the XMTR (transmitter) data output for maintenance purposes

g. controlling the modem.

In line with these capabilities, the interface performs the following operations:

a. Converts parallel data inputs (from the computer) to serial data outputs (to the modem).

b. Converts serial data inputs (from the modem) to parallel outputs (to the computer).

c. Inhibits RCVR (receiver) operation when transmitting in the half duplex mode.

d. Establishes synchronization prior to allowing received data to be transferred to the computer.

e. Generates interrupt requests to the program in response to any one of the following conditions:

- synchronized data received from the modem
- XMTR ready to accept another character for transmission
- modem status change
- Sync (fill) character being transmitted to the modem.

f. Provides control signals to the modem and monitors modem status lines.

4.2.1 Initialization and Programming

Before the DU11 interface can begin to handle data, it must be initialized and programmed.

Initializing the interface prepares it to be programmed. All registers are initialized, all flip-flops are cleared, and the RCVR and XMTR are forced to the idle state. When the RCVR is in the idle state, the RXDBUF (Receiver Data Buffer) is set to all 1s, the RCVR Sync register is cleared, and the RCVR timing and control logic and output flags are cleared. When the XMTR is in the idle state, the XMTR output is a constant MARK (HIGH), the XMTR timing and control logic is cleared, and the XMTR output flags are set.

Programming the DU11 establishes its operating parameters. General operating parameters are controlled by the PARCSR (Parameter Status Register), while specific operating parameters pertaining directly to the RCVR, XMTR, and maintenance circuitry are controlled by the RXCSR (Receiver Status Register) and the TXCSR (Transmitter Status Register). The PARCSR is programmed to select the mode of operation (isochronous, internal synchronous, or external synchronous), word length (5, 6, 7, or 8 bits plus parity), parity (enable or disable), parity sense (odd or even), and sync character configuration. The RXCSR is programmed to enable or disable the RCVR data handling logic, strip sync logic and interrupt logic, and to communicate interface status, requests and supervisory data to the modem. The TXCSR is programmed to select the maintenance mode (normal, internal loop, external loop, and system test); drive the maintenance clock; provide a maintenance data input; reset and initialize the overall interface; enable or disable the XMTR data handling logic, data output logic, and interrupt logic; and select the interface duplex mode (half or full).

Once these three registers are programmed, DU11 operation can begin. Assuming the RCVR and XMTR data handling and interrupt logic is enabled, the XMTR will begin outputting serial characters when a character is loaded into the TXDBUF (Transmitter Data Buffer) and the RCVR will begin accepting serial characters when they appear on the line.

4.2.2 Handshaking Sequence

Handshaking sequences serve to establish the data communication channel. This becomes necessary when the interface is connected to a modem. If the interface is connected to something other than a modem, e.g., a limited distance adapter, handshaking may not be required; the program simply initiates data communication by loading a character into the TXDBUF.

If a handshaking sequence is necessary, the program must be written accordingly. The following paragraphs explain a typical handshaking sequence.

The handshaking sequence is initiated when the modem at one data communication station places a call to a modem at another data communication station. A call may be placed simply by pressing the modem RING button if only two stations are linked in the communication system or dialing-up station no. 1 on the modem at station no. 2, if more than two stations are linked.

Once initiated, the handshaking sequence will be executed as programmed. Figure 4-1 illustrates a DU11 to modem interface; Figure 4-2 illustrates the handshaking sequence. When modem no. 2 places a call to modem no. 1:

a. Modem no. 1 asserts RING to DU11 no. 1.

b. DU11 no. 1 asserts DATA TERM RDY (Data Terminal Ready).

c. Modem no. 1 sends CARRIER to modem no. 2.

d. Modem no. 2 asserts CARRIER to DU11 no. 2 and sends CARRIER back to modem no. 1.

e. Modem no. 1 then asserts CARRIER to DU11 no. 1 and the SCH SYNC (Search Sync) bit is set at DU11 no. 2 to enable the RCVR.

f. DU11 no. 1 asserts REQ TO SD (Request to Send) to modem no. 1.

g. Modem no. 1 asserts CLR TO SD (Clear to Send) to DU11 no. 1.

h. The SEND and TX INTEB (XMTR Interrupt Enable) bits are set at DU11 no. 1 to enable the XMTR.

Note that the modem DATA SET RDY (Data Set Ready) output must be asserted before any data communication can actually take place. The DATA SET RDY line indicates that the modem is powered up and conditions are go.

Supervisory data may also be transmitted simultaneously with the normal communication data. The SEC XMIT (Secondary Transmit) and SEC REC (Secondary Receive) lines provide a supervisory data communication channel.

All data communications can be terminated by simply clearing the Data Terminal Ready bit in the RXCSR.

4.2.3 Basic Operation

The basic operation flow diagram (Figure 4-3) provides a general description of communication systems using the DU11 interface. Only major steps are shown in an attempt to give the reader a general understanding of overall interface operation.

The DU11 interface may be used in two basic configurations: the long distance system which uses modems for data transmission over telephone lines, and the local system which uses a limited distance adapter (LDA) to link-up the interfaces.

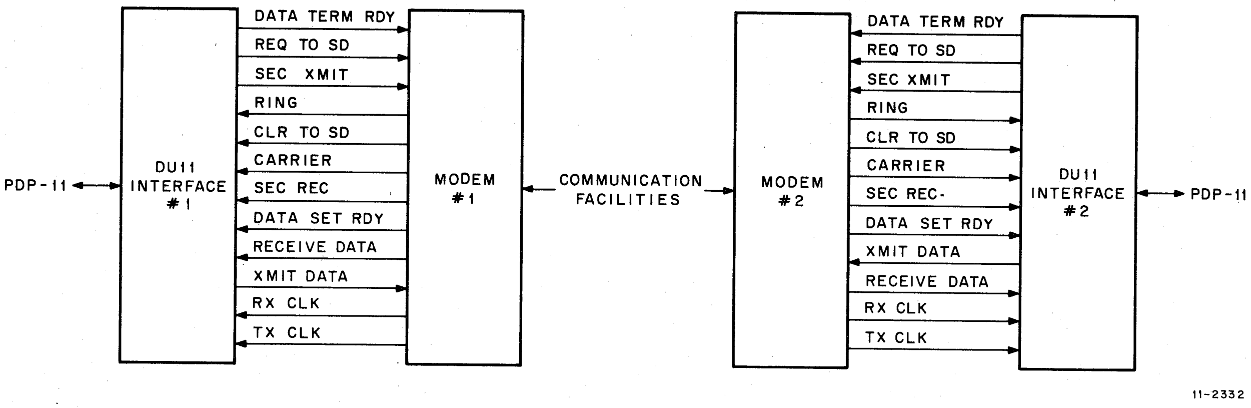


Figure 4-1 DU11 to Modem Interface Diagram

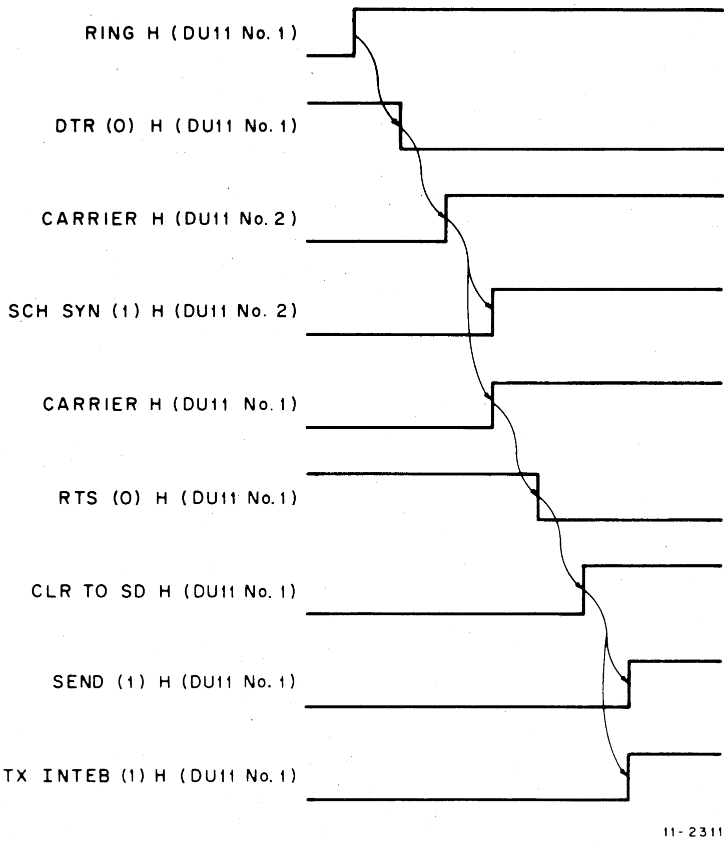


Figure 4-2 Handshaking Sequence Timing Diagram

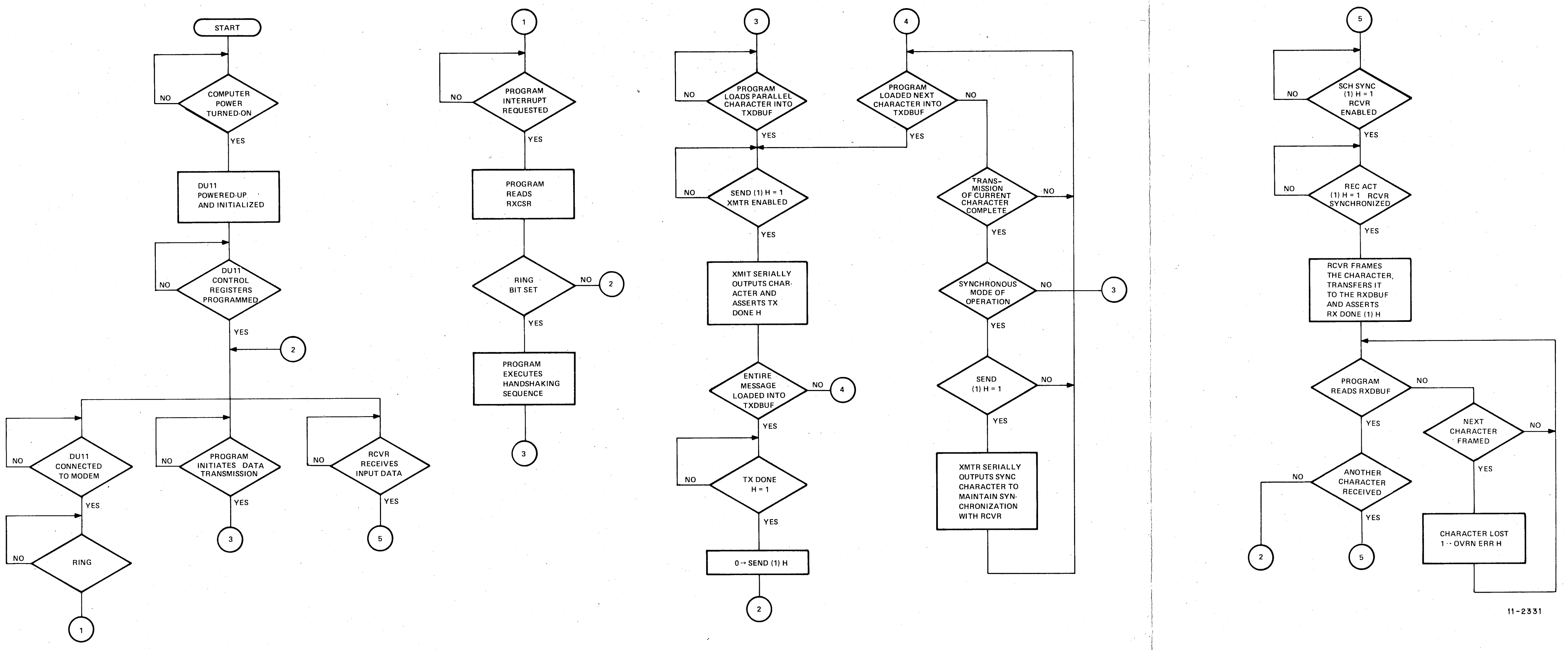

START
COMPUTER POWER TURNED-ON
NO
YES
DU11 POWERED-UP AND INITIALIZED
DU11 CONTROL REGISTERS PROGRAMMED
NO
YES
2
DU11 CONNECTED TO MODEM
NO
YES
RING
NO
1
PROGRAM INITIATES DATA TRANSMISSION
NO
YES
3
RCVR RECEIVES INPUT DATA
NO
YES
5
1
PROGRAM INTERRUPT REQUESTED
NO
YES
PROGRAM READS RXCSR
RING BIT SET
NO
2
YES
PROGRAM EXECUTES HANDSHAKING SEQUENCE
3
3
PROGRAM LOADS PARALLEL CHARACTER INTO TXDBUF
NO
YES
SEND (1) H = 1 XMTR ENABLED
NO
YES
XMIT SERIALLY OUTPUTS CHARACTER AND ASSERTS TX DONE H
ENTIRE MESSAGE LOADED INTO TXDBUF
NO
4
YES
TX DONE H = 1
NO
YES
0 → SEND (1) H
2
4
PROGRAM LOADED NEXT CHARACTER INTO TXDBUF
NO
YES
TRANSMISSION OF CURRENT CHARACTER COMPLETE
NO
YES
SYNCHRONOUS MODE OF OPERATION
NO
3
YES
SEND (1) H = 1
NO
YES
XMTR SERIALLY OUTPUTS SYNC CHARACTER TO MAINTAIN SYNCHRONIZATION WITH RCVR
5
SCH SYNC (1) H = 1 RCVR ENABLED
NO
YES
REC ACT (1) H = 1 RCVR SYNCHRONIZED
NO
YES
RCVR FRAMES THE CHARACTER, TRANSFERS IT TO THE RXDBUF AND ASSERTS RX DONE (1) H
PROGRAM READS RXDBUF
NO
YES
NEXT CHARACTER FRAMED
NO
YES
ANOTHER CHARACTER RECEIVED
NO
YES
2
5
CHARACTER LOST 1 → OVRN ERR H
11-2331

Figure 4-3 Basic Operation Flow Diagram

Referring to Figure 4-3, note that the interface is automatically powered up and initialized when computer power is turned on. The program also initializes the interface via the MSTRST (Master Reset) bit in the TXCSR and normally does so just prior to programming or reprogramming the control registers.

Once the interface is programmed, data communication can commence. Data communication can be initiated by the program or by the modem operator. The program initiates communication simply by loading a character into the TXDBUF. The XMTR then outputs that character to its modem, which transmits it to the modem at a remote station. The modem operator initiates data communication by dialing a remote station on the modem. The modem at the remote station detects the call and responds by asserting RING to its own interface. The interface then generates a RCVR interrupt, which causes the processor to branch to an interrupt service subroutine. The service subroutine directs the processor to execute the handshaking sequence and load a character into the TXDBUF. If the XMTR is enabled [SEND (1) H asserted], the XMTR begins to output the character to the modem and asserts TX DONE H to request the next character of the message. The modem transmits the character to the modem that placed the call. If more characters are to be transmitted, the program responds to the TX DONE H flag and loads the next character into the TXDBUF. When transmission of the current character is complete, transmission of the next character begins. When all characters comprising the message have been loaded into the TXDBUF and TX DONE H is asserted, the program clears SEND (1) H.

As soon as transmission of the current character is complete, the XMTR output marks (Constant HIGH). Note that TX DONE H will not change state after SEND (1) H is cleared.

If, during message transmission, the program fails to load the TXDBUF before current character transmission is complete, one of two operations will result, depending on the mode of operation. If the XMTR is programmed for the synchronous mode, a sync character will be output to maintain synchronization with the RCVR (synchronous mode requires continuous transmission). If the isochronous mode is programmed, the XMTR will simply pause until the next character is loaded into the TXDBUF and then resume operation.

Data reception is initiated when input data is provided to the RCVR, provided SCH SYNC (1) H is asserted. Before the RCVR can begin framing characters for reading by the program, however, REC ACT (1) H must assert indicating that the RCVR at the receiving station is synchronized with the XMTR at the sending station. Once REC ACT (1) H asserts, the RCVR frames the very next character, transfers the character to the RXDBUF, and asserts RX DONE (1) H. This assertion causes a RCVR interrupt request to be generated. The processor branches to an interrupt service subroutine and ultimately reads the RXDBUF. If the program fails to read the RXDBUF before the next character is framed and transferred to the RXDBUF, the character previously framed is lost and OVRN ERR H (overrun error) is asserted.

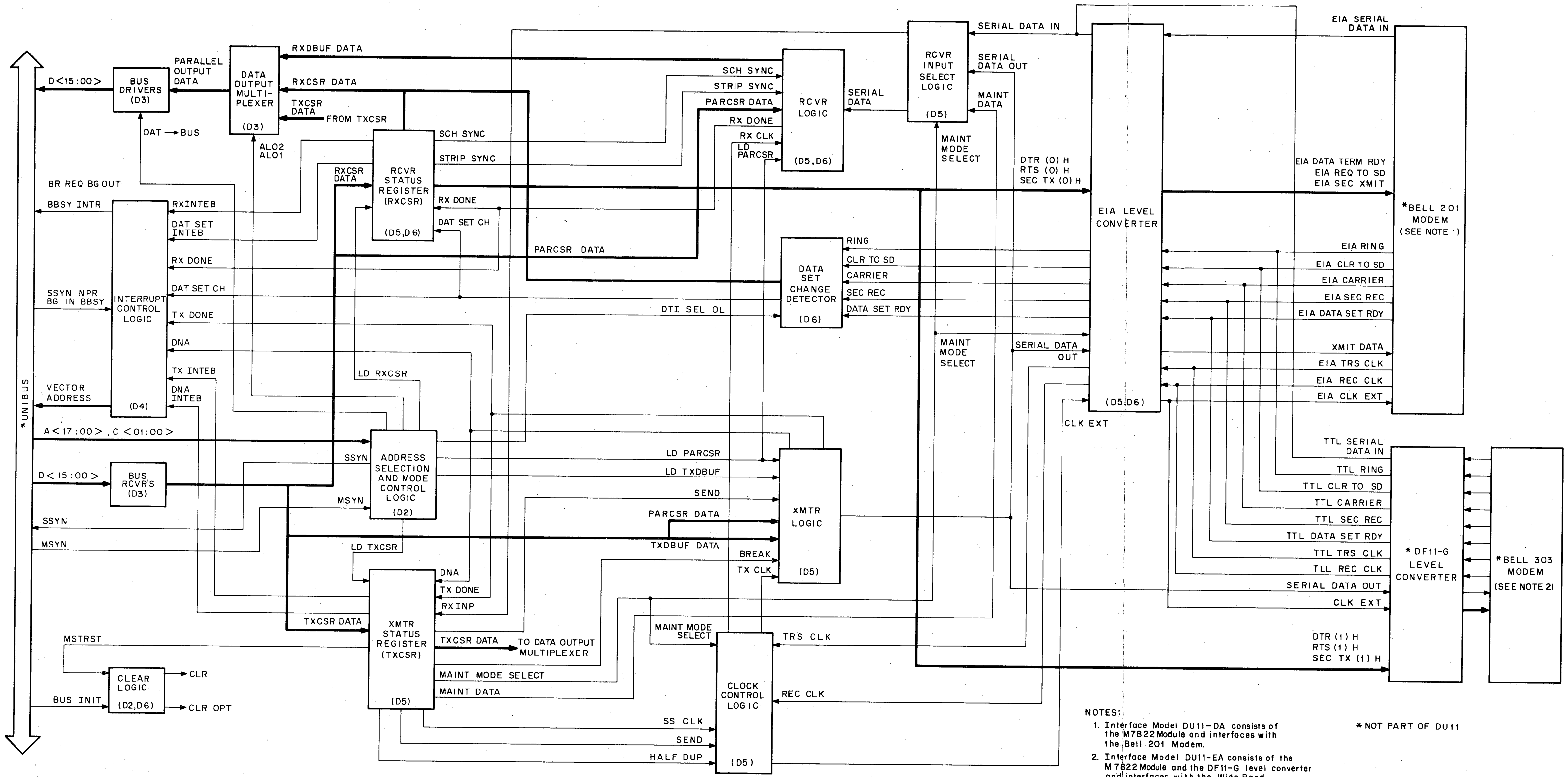
4.2.4 Data Flow Analysis

As previously stated, the DU11 is a program controlled communication interface capable of providing a full duplex communication channel between a PDP-11 computer and a Bell 201 or 303 modem. Because the interface is program controlled, there are obviously two types of data flowing into and out of the unit: control data and communication data.

4.2.4.1 Data Source and Destination – Data is input to the interface from two sources. The program inputs control data and communication data to the interface via the Unibus. The modem inputs communication data to the interface via the modem cable. Communication data input to the interface by the program for transmission can be output immediately to the modem. However, communication data received by the interface from the modem cannot be presented to the program for reading until the interface RCVR is synchronized with the incoming data.

4.2.4.2 Data Flow – Control data is loaded into the DU11 control registers (RXCSR, TXCSR, and PARCSR) and from there controls the operation of the interface. Communication data to be transmitted is loaded into the TXDBUF; communication data received is loaded into the RXDBUF (Figure 4-4).

To program (load) the control registers, the program places the A bits (register address bits), the C bits (bus operation control bits), and the D bits (control data bits) on the Unibus. The A and C bits are applied to the address selection and mode control logic. The D bits are applied to the bus receivers and from there routed to the RXCSR, TXCSR, RCVR logic, and XMTR logic. The address selection and mode control logic decodes the A and C bits and generates the required strobe to load the control register addressed. If, for example, the PARCSR register is addressed, the LD PARCSR (Load PARCSR) strobe is generated and identical PARCSR registers in the RCVR logic and XMTR logic are loaded. Once the control registers are programmed, DU11 operation can begin.



NOTES:

1. Interface Model DU11–DA consists of the M7822 Module and interfaces with the Bell 201 Modem.
2. Interface Model DU11–EA consists of the M7822 Module and the DF11–G level converter and interfaces with the Wide Band Bell 303 Modem.

* NOT PART OF DU11

11-2321

Figure 4-4 DU11 Functional Block Diagram

The program can now initiate data transmission by loading a character into the TXDBUF in the XMTR logic. To load the TXDBUF, the program places the proper A and C bits and the character to be transmitted on the Unibus. The address selection and mode control logic decodes the A and C bits and generates the LD TXDBUF (Load TXDBUF) strobe to load the character into the TXDBUF. The XMTR then outputs the character serially to the modem. The character is transmitted by the modem at the transmitting station to the modem at the receiving station. Assuming the RCVR is synchronized, the RCVR frames the serial character and raises the RX DONE output, which causes a RCVR interrupt request and the interrupt vector address to be sent to the processor. The processor responds by branching to a interrupt service subroutine and reading the RXCSR to determine the nature of the interrupt. Upon determining that an RX DONE interrupt has been requested, the program places the RXDBUF A bits and DATI bus operation C bits on the Unibus. The address selection and mode control logic decodes the A and C bits and enables the RCVR output to the data output multiplexer and the bus drivers, thereby placing the RXDBUF contents on the Unibus.

4.2.5 Functional Block Diagram Description

Functionally, the DU11 can be divided into twelve logic sections, each section performing a specific function in accomplishing the overall task of data handling. The following paragraphs describe the specific function of each logic section (Figure 4-4).

4.2.5.1 Clear Logic – The clear logic initializes the interface logic. The CLR (clear) and CLR OPT (Clear Option) pulses combine to initialize all DU11 registers, flip-flops, and the XMTR and RCVR logic. BUS SSYN L is also inhibited as long as CLR is asserted. These pulses are generated by BUS INIT or MSTRST (Master Reset). BUS INIT is received by the interface whenever the computer START switch is pressed, the processor executes a RESET instruction, or the power fail sequence occurs. MSTRST is programmed controlled and is generated by setting the MSTRST bit in the TXCSR. The MSTRST bit is normally set to clear the DU11 prior to programming.

While the clear logic normally provides both the CLR and CLR OPT outputs, CLR OPT may be disabled by removing jumper W4 (engineering drawing D6). With CLR OPT disabled, the interface may be cleared without clearing the modem control lines, hence the handshaking sequence does not have to be repeated each time the interface is cleared.

4.2.5.2 Address Selection and Mode Control Logic – The address selection and mode control logic decodes the A and C bits placed on the Unibus by the program and generates the gating and strobe signals necessary to perform the operation decoded. Upon receiving the A and C bits, the logic decodes the bits and generates strobes to load data from the Unibus into the RXCSR, TXCSR, PARCSR, or TXDBUF. The A and C bits are also decoded to select the data output multiplexer output via the DAT → BUS (data to bus) and AL02 and AL01 (address lines) control signals.

The BUS SSYN (Bus Slave Sync) line to the Unibus is raised to signify completion of transfer. The MSYN (Master Sync) input is asserted by the program whenever it addresses a Unibus device.

4.2.5.3 Receiver Status Register (RXCSR) – The RXCSR stores control data for the RCVR section of the DU11 and also monitors modem control lines and RCVR interrupt requests. The RXCSR is a 16-bit read/write register that can be programmed by word or byte (high or low).

4.2.5.4 Transmitter Status Register (TXCSR) – The TXCSR stores control data for the XMTR section and maintenance data for the DU11 and monitors the XMTR interrupt requests. The TXCSR also monitors the RX INP (RCVR Input) line in the internal and external loop maintenance modes. The TXCSR is a 16-bit read/write register that can be programmed by word or byte (high or low).

4.2.5.5 Clock Control Logic – The clock control logic decodes the maintenance mode select bits and selects the XMTR and RCVR clock inputs. There are three possible clock sources: the modem, the programmable SS CLK (Single Step Clock), and the system test clock that is contained within the clock control logic. If the normal operating mode is decoded, the modem clocks are selected. If the internal loop or external loop maintenance mode is decoded, the SS CLK is selected. If the system test mode is decoded, the system test clock is selected. Maintenance modes are discussed in more detail in Chapter 6.

When the DU11 is transmitting in the half duplex mode, the SEND and HALF DUP (Half Duplex) inputs are asserted inhibiting the RX CLK, thereby disabling the RCVR. This circuit is necessary because a half duplex modem, by definition, transmits to the RCVR at the sending station as well as to the RCVR at the receiving station.

The CLK EXT (Clock External) output is used in the external loop maintenance mode. The modem test connector must be installed in this mode. The CLK EXT output is looped back by the modem test connector and used to drive the XMTR and RCVR. See Chapter 6 for a detailed description of the external loop mode operation.

4.2.5.6 RCVR Input Select Logic – The RCVR input select logic decodes the maintenance mode select bits and selects the RCVR data input accordingly. There are three possible data sources: the modem (SERIAL DATA IN), the XMTR (SERIAL DATA OUT), and the program (MAINT DATA). The modem data input is selected in the normal operating mode while the XMTR data is selected in the system test mode. In the internal loop and external loop maintenance modes, either the XMTR data or the MAINT DATA may be selected. Note that in the external loop maintenance mode the modem test connector must be installed. The modem test connector serves to loop the XMTR output back to the RCVR input select logic via the SERIAL DATA IN line.

NOTE

Whenever the program activates the MAINT DATA input in either the internal loop or external loop maintenance mode, the BREAK bit in the TXCSR must be set to inhibit the XMTR data output. This removes the possibility of two simultaneous data inputs to the RCVR.

In addition, the RCVR input select logic provides the RX INP monitor signal to the TXCSR. The RX INP line enables program monitoring of the RCVR serial data input in the internal loop and external loop maintenance modes.

4.2.5.7 RCVR Logic – The RCVR logic constitutes the receiver section of the DU11 and includes a double buffered programmable RCVR, synchronization logic, and RX DONE (Receiver Done) generation logic. The RCVR logic detects the serial received character, accomplishes synchronization, frames the received character, raises the RX DONE flag, and holds the framed character (for program reading) until the next character is framed.

Once the RCVR logic is enabled, it operates as programmed. The SCH SYNC input enables the RCVR logic. The contents of the PARCSR determine:

a. mode of operation (internal synchronous, external synchronous, or isochronous)

b. length of character to be framed (5, 6, 7, or 8 bits plus parity)

c. parity (enabled or disabled)

d. parity sense (odd or even)

e. sync character configuration.

The method of achieving synchronization is the principle difference between the modes of operation:

a. In the internal synchronous mode, two contiguous sync characters must be recognized by the RCVR logic to achieve synchronization. Once synchronization is achieved, the RCVR starts framing on the very next character bit. The received characters must arrive at the RCVR in a continuous serial bit stream or synchronization will be lost.

b. The external synchronous mode is designed for use with communication equipment which accomplishes synchronization external to the DU11 interface. The external synchronization logic prohibits RCVR operation by inhibiting the assertion of SCH SYNC until synchronization with the XMTR has been achieved. When external synchronization is achieved, SCH SYNC asserts, forcing the RCVR logic to the synchronized state. The RCVR then starts framing immediately, beginning with the very next character bit.

c. In the isochronous mode, each received character is preceded by a START bit and succeeded by a STOP bit, which serves to synchronize the RCVR. In this mode, the receiver simply does not start framing until it recognizes a START bit. It then frames the character following the START bit and looks for a STOP bit. If a STOP bit is not detected, the character received is considered invalid, flagged as such, and held for reading by the program. Hence, in the isochronous mode, characters need not be preceded by sync characters and need not arrive contiguously at the RCVR.

The STRIP SYNC (Strip Synchronization Character) input determines whether received sync characters are to be permitted to set the RX DONE flag. If STRIP SYNC is asserted, all sync characters are discarded provided no errors are detected.

4.2.5.8 XMTR Logic – The XMTR logic constitutes the XMTR section of the DU11 and includes a double buffered programmable XMTR, data not available logic, TX DONE (Transmitter Done) generation logic, and break logic.

Once the XMTR is enabled, it operates as programmed. The SEND input enables the XMTR logic. The contents of the PARCSR determine:

a. mode operation (synchronous or isochronous)

b. length of character to be transmitted (5, 6, 7, or 8 bits plus parity)

c. parity (enabled or disabled)

d. parity sense (odd or even)

e. sync character configuration (used as fill character in synchronous mode).

There are distinct differences between the two modes of operation:

a. In the synchronous mode, the XMTR receives a parallel transmit character from the program, generates parity if programmed, serially outputs the character plus parity to the modem, and raises the TX DONE flag to request the next character. If the program fails to provide the next character before transmission of the current character is complete, the XMTR outputs fill characters to maintain continuous transmission until another data character is provided. Whenever a fill character is transmitted the DNA (Data Not Available) flag is raised to notify the program of fill character transmission.

b. In the isochronous mode, the XMTR receives a parallel transmit character from the program, generates parity if programmed, outputs a START bit, serially outputs the character plus parity, outputs a STOP bit, and raises the TX DONE flag to request the next character. However, in the isochronous mode, if the program fails to provide the next character before transmission of the current character is complete, the XMTR simply pauses until the next character is provided. Hence, the DNA flag is never used in the isochronous mode.

The BREAK input inhibits the XMTR output. Whenever the TXCSR BREAK bit is set, the BREAK input to the XMTR logic asserts and inhibits the XMTR output. This input enables the program to inhibit the XMTR output, while inputting data directly to the RCVR via the RCVR input select logic in the internal and external loop maintenance modes.

4.2.5.9 Data Set Change Detector – The data set change detector monitors the modem status inputs and detects any status change. The status lines are routed to the data output multiplexer as part of the RXCSR. When a status change on any one of the lines occurs, the DAT SET CH (Data Set Change) flag is raised to cause a interrupt request. The DAT SET CH flag is cleared by DTI SEL 0 L (RXCSR Read Strobe), which is generated when the interrupt request is serviced.

4.2.5.10 Data Output Multiplexer – The data output multiplexer controls the DU11 data output to the bus driver. There are three possible data outputs: the RXCSR, the TXCSR, and the framed character and error flags from the RCVR logic (RXDBUF). The data output is selected by the address bit inputs (AL02 and AL01).

4.2.5.11 Bus Drivers – The bus drivers apply the data output to the Unibus. The drivers are enabled by the DAT → BUS (data to bus) input from the address selection and mode control logic.

4.2.5.12 Interrupt Control Logic – The interrupt control logic enables the DU11 to gain control of the Unibus (become bus master) and causes a program interrupt to a interrupt address vector. There are four different reasons for the DU11 to request an interrupt.

a. TX DONE – XMTR is ready to accept another character for transmission.

b. DNA – XMTR is transmitting a fill character and wishes to notify the program.

c. RX DONE – RCVR has framed a character and wishes the program to read it.

d. DAT SET CH – modem status has changed and wishes to notify the program.

All interrupts are enabled or disabled via the program. The TX DONE and DNA interrupts are referred to as XMTR interrupts and are enabled by setting the TX INTEB and DNA INTEB bits, respectively, in the TXCSR. The RX

DONE and DAT SET CH interrupts are referred to as RCVR interrupts and are enabled by setting the RX INTEB and DAT SET INTEB bits, respectively, in the RXCSR.

When an interrupt request is generated, the interrupt control logic is enabled, the bus is requested and granted, the interrupt is identified and acknowledged, and the processor branches to the subroutine identified by the vector address and services the interrupt.

The BG OUT (Bus Grant Out) line propagates the BG IN to the Unibus whenever the DU11 is not requesting the bus or presently has control of the bus. This, in effect, accomplishes the daisy chaining of BG IN to all devices on the same BR level.

The NPR (Non-Processor Request) input improves NPR latency. When this input is asserted, the BG OUT output is inhibited, thereby inhibiting the granting of the bus via the BG IN input to any device on the same BR level as the DU11 and electrically further from the processor.

4.2.5.13 EIA Level Converters – The EIA level converters merely convert the logic level signals to the operating voltage levels of the Bell 201 modem. All logic signals ranging from 2.4 to 3.5 V are converted to +6 V. All ground (0 V) logic signals are converted to -6 V. The EIA level converter threshold voltage is 1.5 V. Any inputs less than 1.5 V will not cause the converter to switch.

NOTE

The functional block diagram includes the Bell 201 and the Bell 303 interface connections merely to provide the reader with a complete picture of DU11 interface connections.

4.3 DETAILED DESCRIPTION

The following paragraphs provide a detailed logic level description of the functional blocks listed below:

a. Clear Logic
b. Address Selection and Mode Control Logic
c. Data Output Multiplexer Logic
d. RCVR Input Select Logic
e. Clock Control Logic
f. RCVR Logic
g. XMTR Logic
h. Data Set Change Detector
i. Interrupt Control Logic

4.3.1 Clear Logic

The clear logic generates the CLR outputs when the BUS INIT signal is detected or bit 08 of the TXCSR is set (Figure 4-5 and engineering drawings D2 and D6). When BUS INIT L is asserted, the CLR logic outputs are asserted and remain asserted until BUS INIT is cleared. When bit 08 is set, the 6 μs one-shot is triggered on the leading edge of LD TXCSR HB H (Load TXCSR High Byte). The CLR outputs are then asserted for the duration of the one-shot (6 μs).

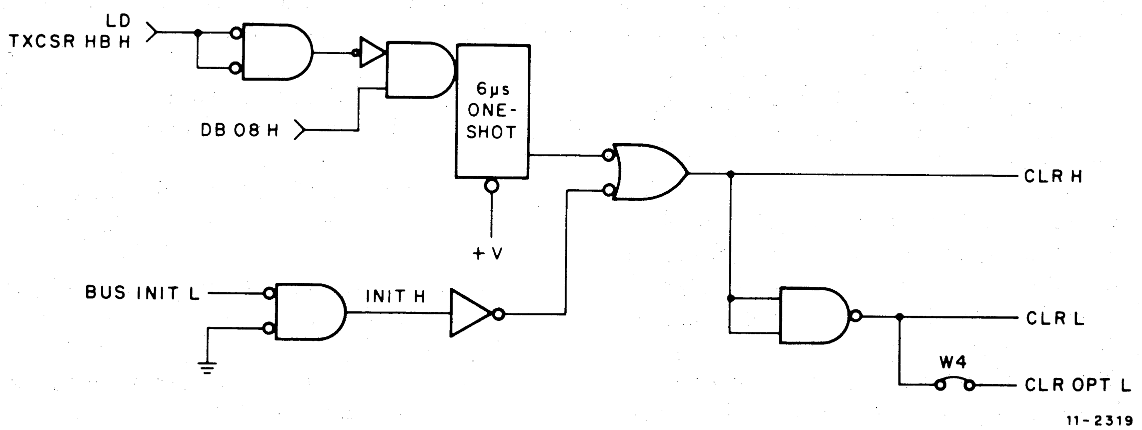


Figure 4-5 Clear Logic

While the clear logic normally provides three outputs, the CLR OPT output may be disabled. The CLR OPT output resets RXCSR bits 1, 2, and 3 and thereby clears the control lines to the modem. By removing jumper W4, CLR OPT is disabled and the DU11 may be cleared without clearing the control lines; hence, with jumper W4 removed, the handshaking sequence does not have to be repeated each time the DU11 is cleared.

NOTE

When CLR H is asserted, BUS SSYN L is inhibited for the duration of the pulse.

4.3.2 Address Selection and Mode Control Logic

The address selection and mode control logic decodes the A and C bits placed on the bus by the program and generates the gating and strobe signals necessary to perform the operation decoded (engineering drawing D2).

The DU11 address selection and mode control logic features rocker position address selection switches. These switches (SW0–SW9) control the input to one pin of each of the ten comparators. The inputs to the other comparator pins are controlled by bus address receiver bits 03–12. If the bus address bit input matches the switch input, the comparator is satisfied and the output pin goes HIGH (+3 V). For example, if rocker switch SW9 is set to the ON position (closed), bus A12 L must be cleared (+3 V) to satisfy the comparator.

To address the DU11, the program must generate a bus address that matches all the rocker position switch inputs. When such an address is placed on the bus, all the comparators are satisfied and COMP H (Compare) asserts. Whenever the program generates a device address, DEV ADDR H asserts because all device addresses are relegated to the uppermost 4K of address space and MSYN (Master SYNC) accompanies all addresses. Hence, ADRS SEL L is asserted enabling the BUS SSYN circuit, enabling the decoder and conditioning the AND gate.

Thus, the DU11 is selected and the remaining A bits (A00–A02) and C bits, (C00, C01) must be decoded to generate the gating and strobe signals. The A bits select the particular register. During DATOB operations, the A bits also indicate the register byte (high or low) to be loaded. Refer to Table 4-1 for bit configurations required to select each register. The C bits select the mode of operation. Refer to Table 4-2 for C bit configuration.

The following paragraphs explain the operation of the remaining logic with respect to DATI, DATOB, and DATO bus operations.

Table 4-1
Bus Address Register Select Bit Configurations

Register	Bus Address Bits		
	02	01	00*
RXCSR	0	0	0
RXDBUF	0	1	0
PARCSR	0	1	0
TXCSR	1	0	0
TXDBUF	1	1	0

*This bit is set for DATOB operations on the high byte of the RXCSR or TXCSR.

Table 4-2
Bus Operation Control Bit Configurations

Mode	Control Bits	
	C01	C00
DATI	0	0
DATIP	0	1
DATO	1	0
DATOB	1	1

Note: DATI causes data to be transferred out of the interface. DATO causes data to be transferred into the interface.

4.3.2.1 Typical DATI Logic Operation – When the processor places the necessary address and control bits on the Unibus to perform a DATI (Read) operation on the RXCSR, the logic operation is as follows (engineering drawing D2 and Figure 4-6):

a. The processor places the proper A and C bits on the Unibus and waits 150 ns (minimum) to allow for signal skew (75 ns) and logic delay (75 ns) before asserting BUS MSYN L (Master Sync).

b. If the interface still has BUS SSYN L asserted from a previous bus cycle, the processor waits until BUS SSYN L is cleared and then asserts BUS MSYN.

c. The assertion of BUS MSYN L causes ADRS SEL L to assert.

d. ADRS SEL L asserts DAT → BUS L and combines with CL01 H, AL02 H, and AL01 H which are cleared to enable the DTI SEL 0 L decoder output.

e. Referring to engineering drawing D3, the assertion of DAT → BUS L enables the bus drivers. AL02 H and AL01 H cleared select the RXCSR output from the data output multiplexer.

f. Referring back to engineering drawing D2, 250 ns after the assertion of ADRS SEL L, BUS SSYN L asserts provided CLR H is cleared (the assertion of CLR H inhibits BUS SSYN L).

g. When the processor detects BUS SSYN L, it waits 75 ns to allow for skew and internal gating, then strobes the data from the interface and clears BUS MSYN L.

h. The processor then waits another 75 ns to ensure that no errors occur and clears the A and C lines.

i. When the interface detects the clearing of BUS MSYN L, it immediately clears DAT → BUS L inhibiting the bus drivers and clears BUS SSYN L 250 ns later. The Unibus is now free for other use.

4.3.2.2 Typical DATOB Logic Operation – When the program places the necessary A, C, and D bits on the Unibus to perform a DATOB (Byte Load) operation on the high byte of the TXCSR the logic operation is as follows (engineering drawing D2 and Figure 4-7):

a. The program places the proper A, C, and D bits on the Unibus and waits 150 ns (minimum) to allow for signal skew (75 ns) and logic delay (75 ns) and asserts BUS MSYN L, provided BUS SSYN L from the previous bus cycle is cleared.

b. The assertion of BUS MSYN L causes ADRS SEL L to assert.

c. ADRS SEL L combines with CL01 H and AL02 H asserted and AL01 H cleared to enable the DTO SEL 4 L decoder output.

d. DTO SEL 4 L asserted causes LD TXCSR HB H to assert (INH LB H asserted inhibits LD TXCSR LB H).

e. LD TXCSR HB H asserted loads the high byte of the TXCSR (engineering drawing D5).

f. 250 ns after the assertion of ADRS SEL L, BUS SSYN L asserts provided CLR H is cleared.

g. When the processor detects BUS SSYN L, it waits 75 ns and clears BUS MSYN L.

h. The processor then waits another 75 ns and clears the A, C, and D lines.

i. When the interface detects the clearing of BUS MSYN L, it immediately clears LD TXCSR HB H and 250 ns later clears BUS SSYN L.

4.3.2.3 Typical DATO Logic Operation – When the processor places the necessary A, C, and D bits on the Unibus to perform a DATO (load) operation on the TXDBUF, the logic operation is as follows (engineering drawing D2 and Figure 4-8):

a. The processor places the proper A, C, and D bits on the Unibus, waits 150 ns, and asserts BUS MSYN L provided BUS SSYN L from the previous bus cycle is cleared.

b. The assertion of BUS MSYN L causes ADRS SEL L to assert.

c. ADRS SEL L combines with CL01 H, AL02 H, and AL01 H asserted to enable the DTO SEL 6 L decoder output.

d. DTO SEL 6 L asserted triggers the 350 ns one-shot causing LD TXDBUF (1) H to assert.

e. LD TXDBUF (1) H asserted loads the TXDBUF and, after a 150 ns delay, inhibits BUS SSYN L.

f. 350 ns after the one-shot is triggered, it resets and, after a 150 ns delay, causes BUS SSYN L to assert provided CLR H is cleared.

g. When the processor detects BUS SSYN L, it waits 75 ns and clears BUS MSYN L.

h. The processor then waits another 75 ns and clears the A, C, and D lines.

i. When the interface detects the clearing of BUS MSYN L, it immediately clears DTO SEL 6 L and 250 ns later clears BUS SSYN L.

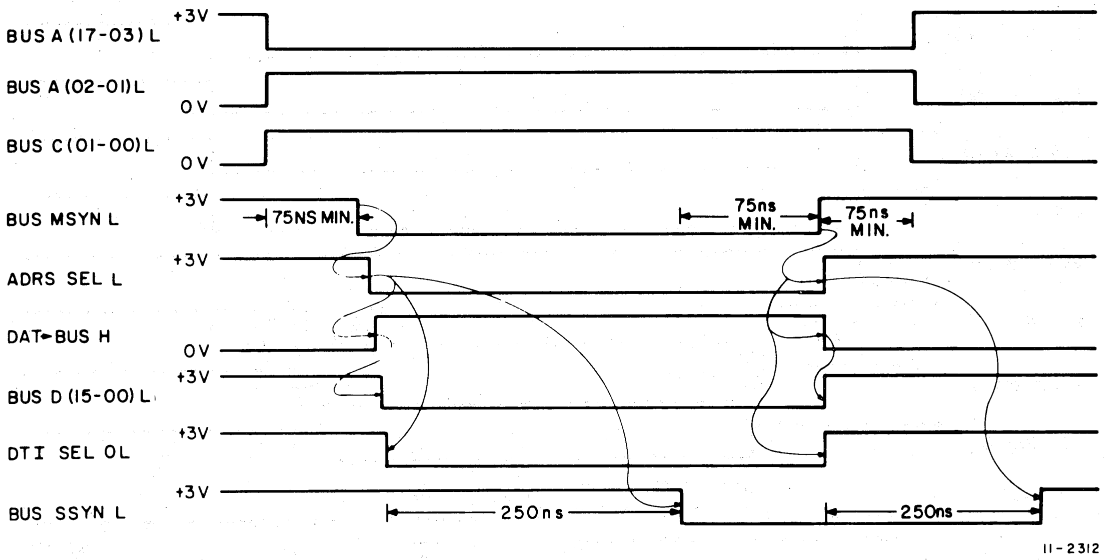


Figure 4-6 DATI Timing Diagram

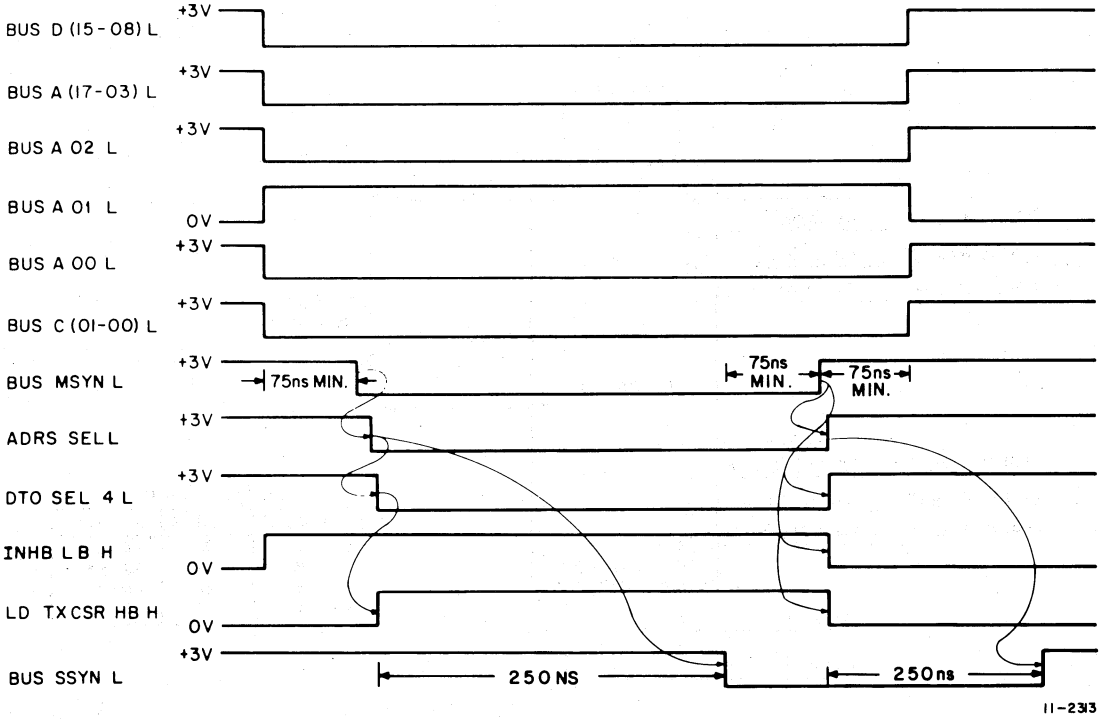


Figure 4-7 DATOB Timing Diagram

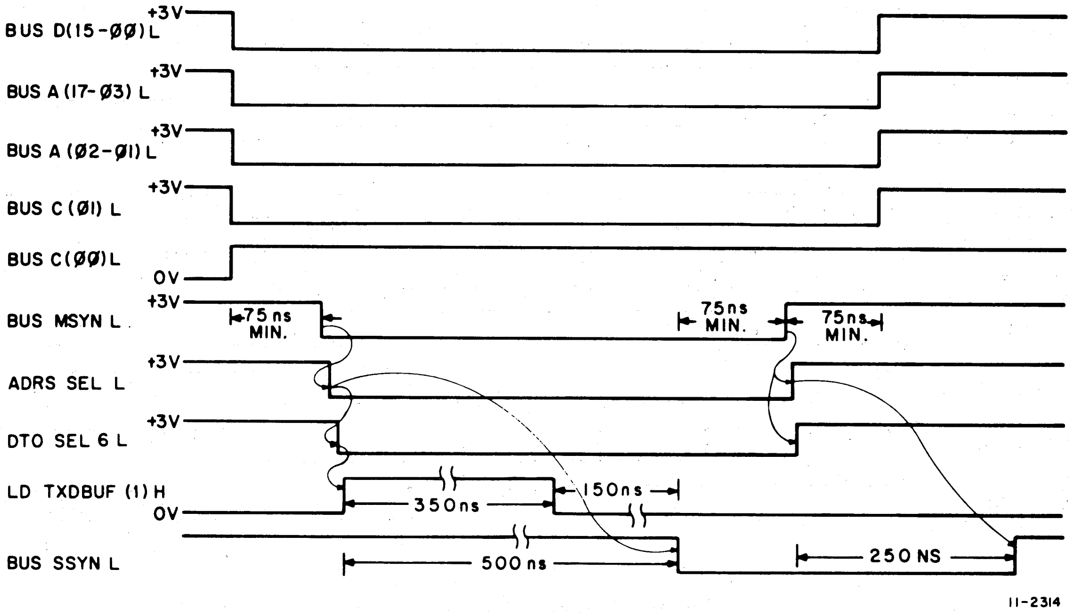
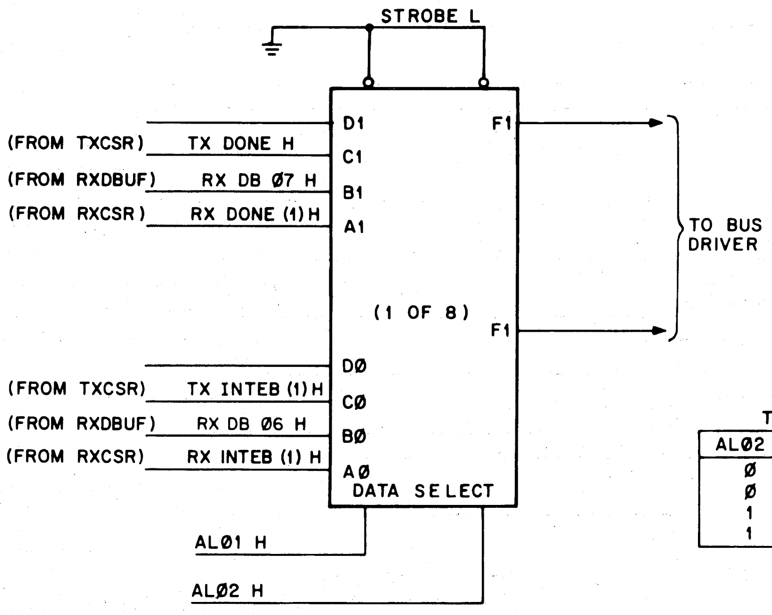


Figure 4-8 DATO Timing Diagram



TRUTH TABLE

ALØ2 H	ALØ1 H	OUTPUT
Ø	Ø	AØ, A1
Ø	1	BØ, B1
1	Ø	CØ, C1
1	1	DØ, D1

11-2315

Figure 4-9 Data Output Multiplexer Logic

4.3.3 Data Output Multiplexer Logic

The data output multiplexer logic decodes address bits A02 and A01 and selects one of three register data inputs to be output to the bus drivers. Figure 4-9 shows one of a total of eight multiplexers used in the DU11. Refer to engineering drawing D3 for a complete logic picture.

4.3.4 RCVR Input Select Logic

The RCVR input select logic decodes the maintenance mode select bits and selects the RCVR data input accordingly.

There are three possible data sources (Figure 4-10 and engineering drawing D5): the modem (SERIAL DATA IN), the program [MAINT DATA (1) H], and the XMTR (SERIAL DATA OUT H). Refer to Table 4-3 for maintenance mode select bit configurations and corresponding data sources.

The maintenance mode select bits enable the proper data source inputs. In the normal operating modes, MS 00 (0) H is asserted and MS 01 (1) H is cleared enabling the SERIAL DATA IN input and disabling the RX INP (Receiver Input) output. In the internal loop maintenance mode, MS 00 (0) H and MS 01 (1) H are cleared, thus asserting EN BIT WID H which enables the MAINT DATA (1) H input and the RX INP output. Also in internal loop maintenance mode, MS 00 (1) H is asserted enabling the SERIAL DATA OUT H input. In the external loop maintenance mode, MS 00 (0) H and MS 01 (1) H are asserted, enabling the MAINT DATA (1) H and SERIAL DATA IN inputs and the RX INP output. Note that the modem test connector is installed in the external loop maintenance mode; hence, the SERIAL DATA IN input originates from the XMTR (Figure 4-11). In the system test mode, MS 00 (1) H is asserted enabling SERIAL DATA OUT H.

NOTE

In the internal and external maintenance modes two data sources (Program and XMTR) can be simultaneously input to the RCVR; hence, care must be taken when programming these modes to assure that only one of the sources is active. The break logic is provided for just that purpose. When the BREAK bit is set the XMTR output (SERIAL DATA OUT H) is inhibited. Refer to Paragraph 4.3.7 for a description of the break logic.

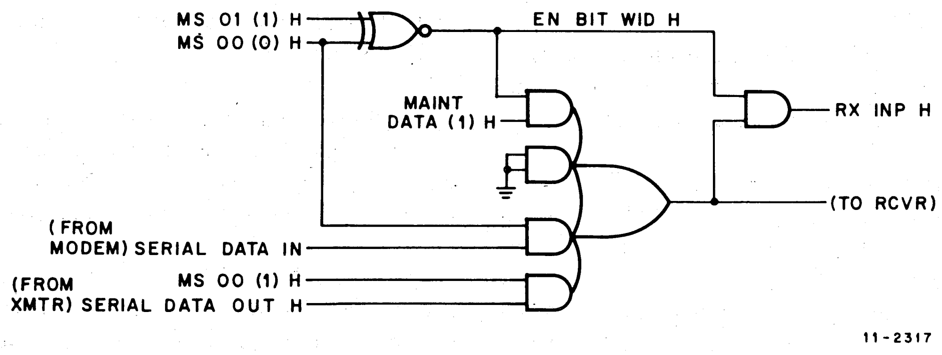


Figure 4-10 RCVR Input Select Logic

Table 4-3
Maintenance Mode to Data Source Relationship

Maintenance Mode	Maintenance Mode Select Bits		Data Source
	MS 01	MS 00	
Normal Operation	0	0	modem
Internal Loop	0	1	Program or XMTR
External Loop	1	0	Program or XMTR
System Test	1	1	XMTR

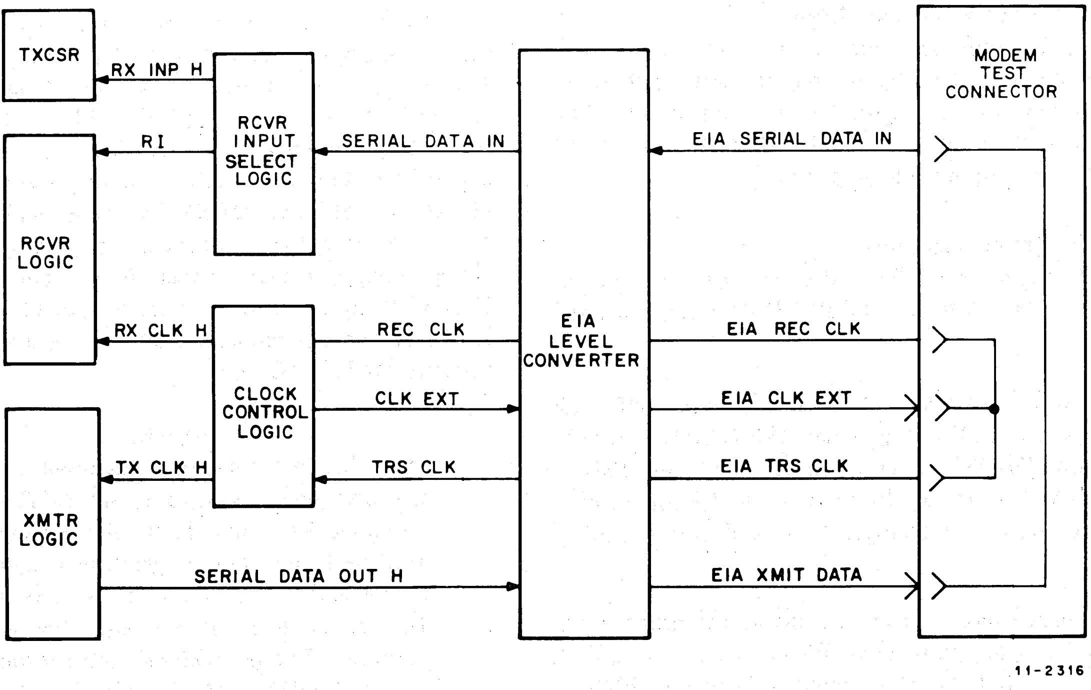


Figure 4-11 External Loop Maintenance Mode Interconnection Diagram

4.3.5 Clock Control Logic

The clock control logic decodes the maintenance mode select bits and selects the XMTR and RCVR clock inputs accordingly (Figure 4-12 and engineering drawing D5).

Different clocks are used for different modes of operation. In the normal operating mode, MS 00 (0) H is asserted enabling the modem clock inputs (TRS CLK and REC CLK) that are routed to the XMTR and RCVR. In the internal loop maintenance mode, MS 00 (1) H is asserted enabling the SS CLK (1) H input to drive the XMTR and RCVR. In the external loop maintenance mode, MS 01 (1) H and MS 00 (0) H are asserted, enabling SS CLK (0) H and the TRS CLK and REC CLK inputs. SS CLK (0) H drives CLK EXT, which is routed to the modem test connector, looped back, and applied to the TRS CLK and REC CLK inputs (Figure 411). In the system test mode, MS 01 (1) H and MS 00 (1) H are asserted enabling the internal system test clock to drive the XMTR and RCVR. In this mode, the modem clock inputs (TRS CLK and REC CLK) are inhibited.

When the DU11 is transmitting in the half duplex mode, the REC CLK input is disabled. When the HALF DUP (1) H (Half Duplex) and SEND (1) H inputs are asserted, REC CLK is inhibited.

The SS CLK inputs are provided for maintenance purposes. This program controlled clock is used in the internal and external loop maintenance modes and can be operated very slowly via the program to facilitate troubleshooting.

The SYS TST CLK H output is also used for maintenance purposes. This output is used to provide an asynchronous clocking source for the system test mode.

4.3.6 RCVR Logic

The RCVR detects the serial received character, accomplishes synchronization, frames the received character, detects errors, raises the RX DONE flag, and holds the framed character (for program reading) until the next character is framed (Figure 4-13 and engineering drawing D5).

Before RCVR operation can begin, the RCVR logic must be initialized and the PARCSR, TXCSR and RXCSR registers programmed.

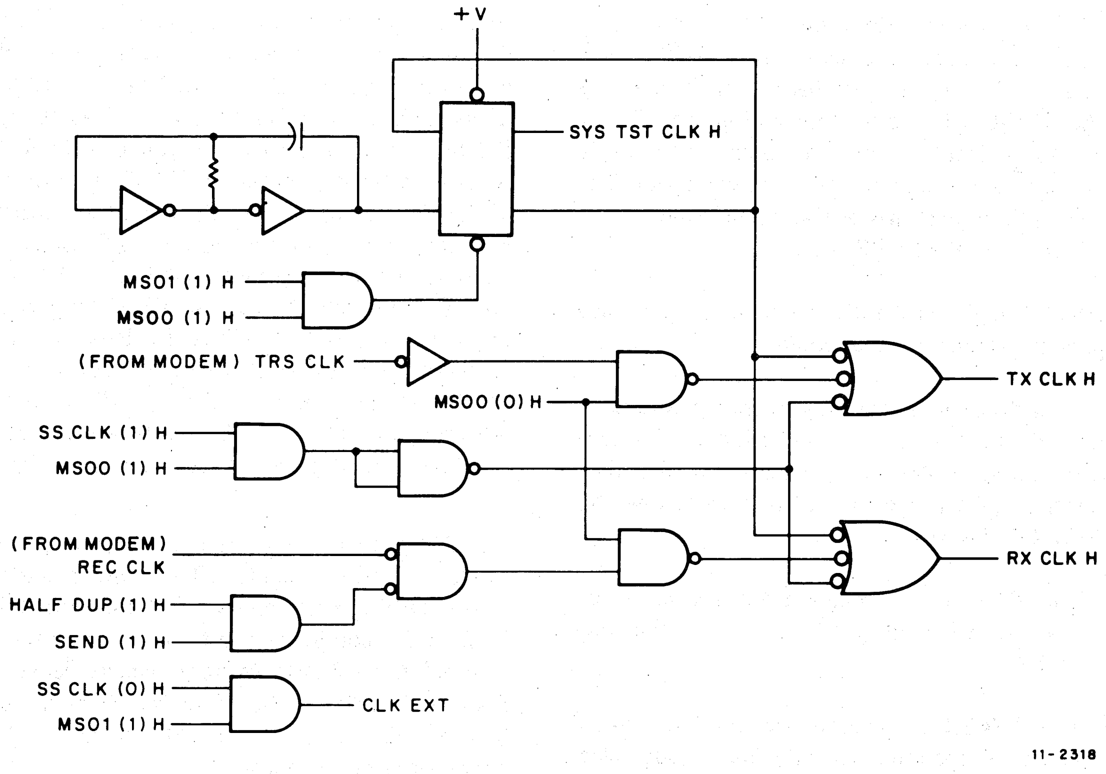


Figure 4-12 Clock Control Logic

4.3.6.1 RCVR Logic Initialization and Programming – The RCVR logic is initialized by BUS INIT or by setting the MSTRST bit, which causes the assertion of the CLR inputs. The CLR L input resets all RCVR flip-flops and the CLR H input forces the RCVR to the idle state. In the idle state, the RXDBUF is set to all 1's and the sync register and all timing and control logic and output flags are cleared.

The RCVR is then programmed by loading the PARCSR, TXCSR, and RXCSR. To program the PARCSR the PARCSR load strobe is asserted, PARCSR bits 13–08 are loaded into the control register, PARCSR bits 07–00 are loaded into the sync register, and the SYNC INTR and SYNC MODE flip-flops are set or reset in accordance with the state of PARCSR bits 13 and 12. Bits 13 and 12 select the RCVR mode of operation. Table 4-4 lists the bit configuration for each mode of operation.

Programming the TXCSR establishes the source of the RCVR serial data input. In the normal operating modes, the data source is the modem. In the maintenance modes, data inputs from the XMTR and the program may be selected as discussed in Paragraph 4.3.4.

Table 4-4
PARCSR Mode Select Bit Configurations

Mode	Mode Select Bits	
	13	12
Isochronous	0	0
External Synchronous	1	0
Internal Synchronous	1	1

Programming the RXCSR directly controls the operation of the RCVR logic. The SCH SYNC bit must be set or RCVR operation is inhibited. The assertion of the STRIP SYNC bit enables RCVR logic to strip (discard) all received sync characters provided no errors are detected. Note that the RX INTEB bit must also be set or RX DONE flags will be ignored by the interrupt control logic.

4.3.6.2 RCVR Logic Operation – The following paragraphs provide a detailed description of RCVR logic operation in each of the three operating modes. Figure 4-13 illustrates the RCVR logic and the timing diagrams (Figures 4-14 through 4-17) illustrate logic operation in response to different character input configurations in each mode. The characters on each timing diagram are numbered in the order of their arrival at the RCVR input.

Before the RCVR logic can be discussed, the programmed operating parameters must be established.

In the following discussions, the only operating parameter that changes is the mode of operation which naturally must correspond to the particular mode being discussed. Otherwise, the control registers are programmed as follows for the entire RCVR discussion:

a. The PARCSR is programmed for an 8 bit character with odd parity and a sync character of 026_8.

b. The TXCSR is programmed for the normal mode of operation (maintenance mode select bits cleared).

c. The RXCSR is programmed to enable the RCVR (SCH SYNC bit set), strip sync characters (STRIP SYNC bit set), and enable RX DONE interrupts (RX INTEB bit set).

4.3.6.2.1 Internal Synchronous Mode – In the internal synchronous mode, characters must arrive at the RCVR in a continuous serial bit stream; character synchronization is accomplished within the RCVR logic. To accomplish synchronization, the RCVR must recognize two contiguous sync characters and, upon doing so, assert REC ACT (1) H, thereby enabling the RX DONE flag to be asserted for all succeeding received characters.

As previously stated, the program prepares the RCVR logic for operation by initializing the logic and programming the control registers. In the internal synchronous mode, bits 13 and 12 of the PARCSR are set. Therefore, when the PARCSR load strobe occurs, the SYNC INTR flip-flop and the SYNC mode flip-flop are set (Figure 4-14).

RCVR operation is enabled when SCH SYNC (1) H asserts. The assertion of SCH SYNC (1) H causes SYNC (1) L to assert when RX CLK H goes low which, in turn, asserts the SS (Search Sync) input to the RCVR; thus, the RCVR is enabled.

After one bit time delay the RCVR begins shifting in the first bit of the first character (bit S) in search of a sync character. As each bit is shifted into the RCVR register a duplicate bit is also shifted into the RXDBUF, so that the RXDBUF is always an exact duplicate of RCVR register. After each bit is received, the RCVR compares the content of the RCVR register to that of SYNC register. When a complete sync character has been shifted into the RCVR register, the two registers match and the RCVR asserts MDET H (Match Detect). MDET H triggers the MATCH one-shot, asserting MATCH L. Because no errors are detected (RX ERR H cleared), RST CHAR L (Reset Character) also asserts. RST CHAR L has no effect, however, as DATA RDY L (Data Ready) is cleared. MATCH L resets after 400 ns and sets the 1st IN flip-flop.

The RCVR then starts framing on the very next bit (bit D of the second character). When framing starts, the RCVR ceases shifting each received bit into the RXDBUF and inhibits the comparator until an entire character has been received in the RCVR register (character length specified by PARCSR). At the center of the last character bit (the parity bit in this case), the RCVR performs the following operations simultaneously:

a. checks parity

b. compares framed character to sync character

c. parallel transfers the framed character into the RXDBUF.

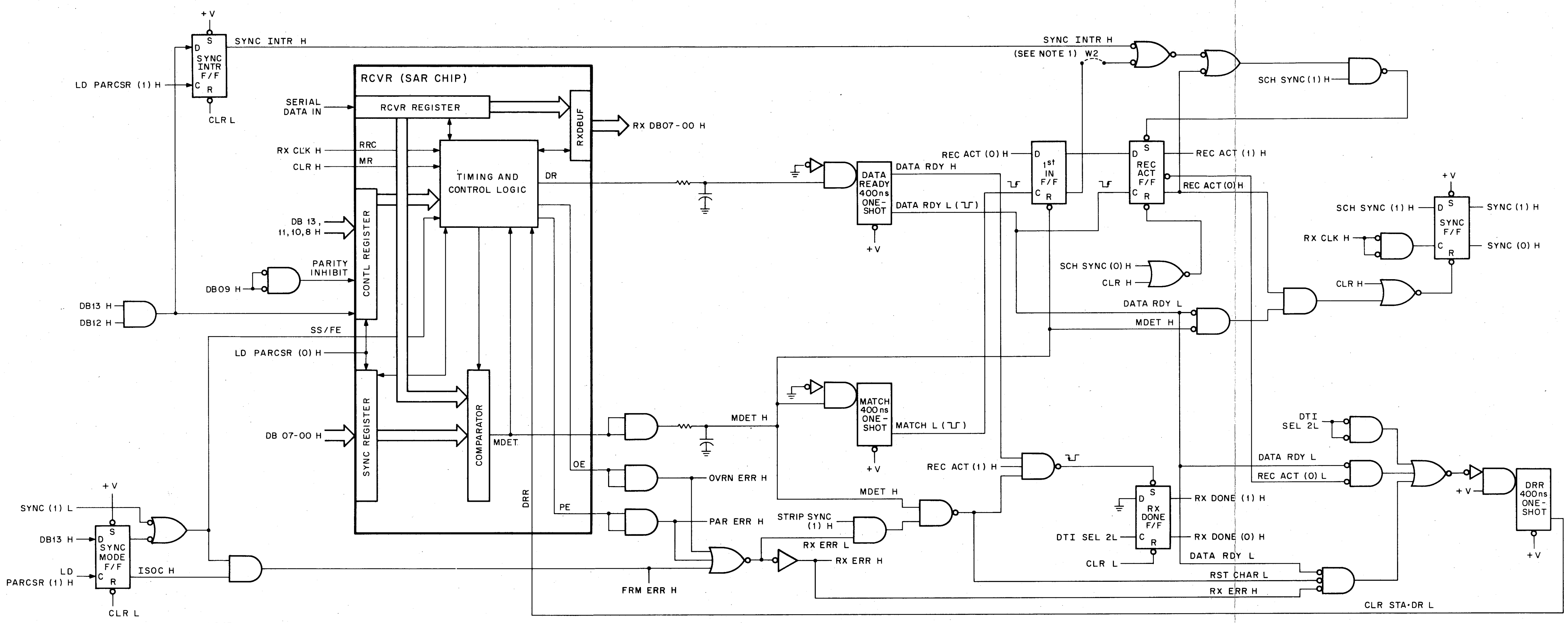
The RCVR then asserts PE (parity error) if parity is incorrect and keeps MDET H asserted if the second character is a sync character. However, in this case the second character is a data character, with good parity; hence, 200 ns later the MDET H flag is cleared, DR (Data Ready) is asserted, and the following ensues:

a. RST CHAR L clears

b. the 1st IN flip-flop resets

c. DATA RDY L asserts and is ANDed with MDET H cleared to reset the SYNC flip-flop and ANDed with REC ACT (0) L asserted to trigger the DRR (Data Ready Reset) one-shot.

d. Clearing the SYNC flip-flop clears SYNC (1) L, which re-initializes the RCVR logic and inhibits RCVR operation.



NOTE:
1 Jumper W2 is normally removed.
See Table 2-1 for jumper discriptions.

11-2320

Figure 4-13 RCVR Logic

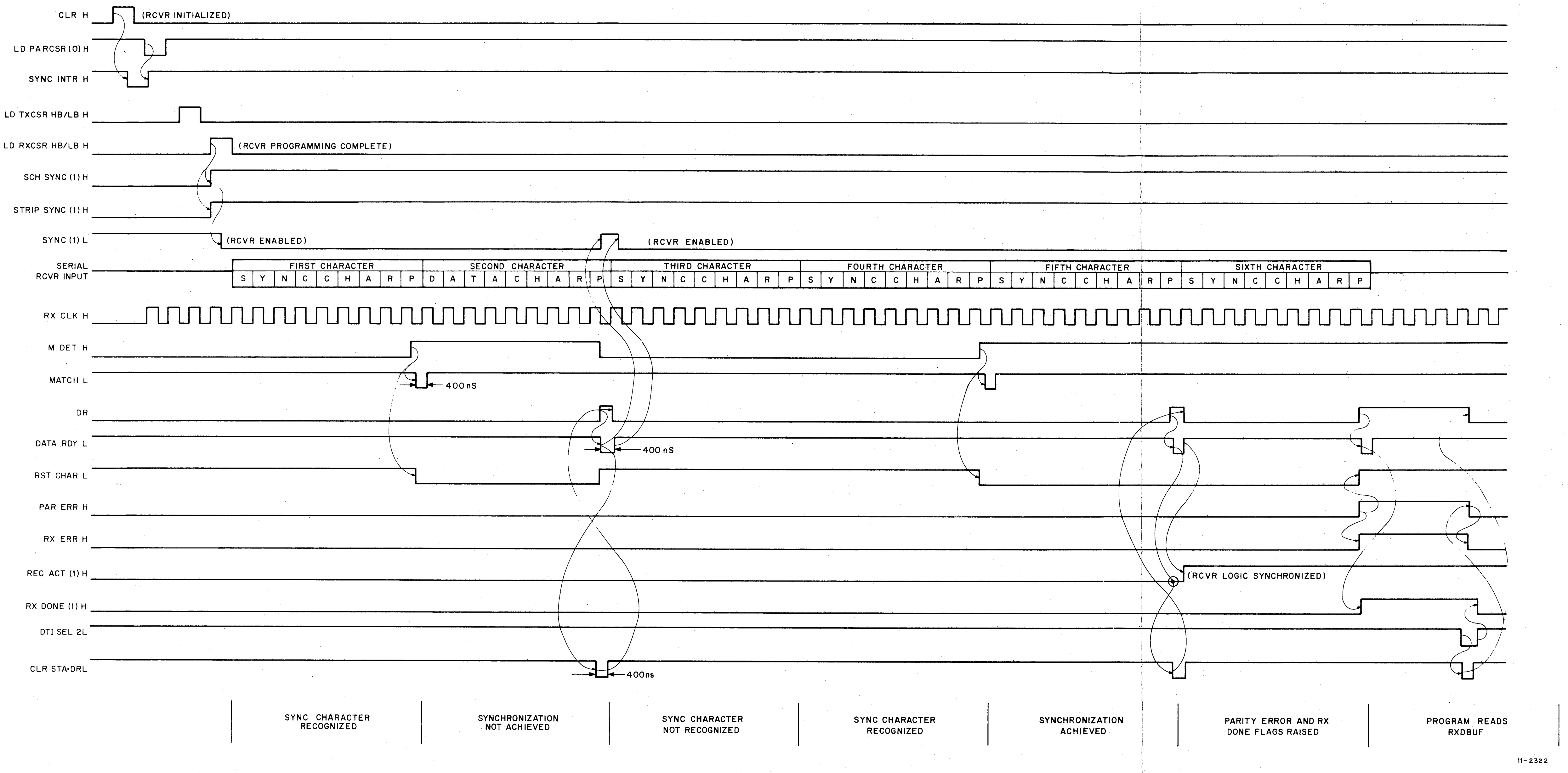


Figure 4-14 RCVR Internal Synchronous Mode Timing Diagram (Example I)

e. Triggering the DRR one-shot asserts CLR STA • DR L (Clear Status and Data Received) which clears the DR flag.

f. 400 ns after it is triggered, the DR one-shot clears DATA RDY L, thus allowing SYNC (1) L to assert again when RX CLK H goes LOW.

g. The assertion of SYNC (1) L enables RCVR operation.

After a one bit time delay, the RCVR starts shifting in the bits comprising the third character. Note that RCVR operation is inhibited when the first bit (bit S) of the third character arrives at the RCVR and is not enabled until RX CLK H goes LOW at the center of the bit S; hence, bit S is lost. Also note that the RCVR is again searching for a sync character; each time a bit is shifted into the RCVR register it is duplicated in the RXDBUF and the RCVR register is compared to the Sync register. Hence, the third character will not cause a match because the entire sync character is not received.

The RCVR simply continues shifting bits in until a match occurs. The match will occur at the center of the parity bit of the fourth character. At that time, MDET H is asserted and RST CHAR L and MATCH L assert; MATCH L resets 400 ns later and sets the 1st IN flip-flop. The RCVR starts framing on the very next bit received (bit S of the fifth character).

At the center of the parity bit of the fifth character, the RCVR simultaneously checks parity, does a sync character comparison, and transfers the framed character into the RXDBUF. The fifth character is a sync character with good parity; hence, the DR flag asserts, DATA RDY L asserts and is ANDed with REC ACT (0) L causing CLR STA • DR L to assert. If CLR STA • DR L should fail to assert at this time, the next character received would cause an overrun error. CLR STA • DR L clears the DR flag. Approximately 400 ns later, DATA RDY L clears and REC ACT (1) H asserts on the positive-going edge of DATA RDY L; thus, the RCVR logic is synchronized. Note that REC ACT (1) H conditions the AND gate connected to the set input of the RX DONE flip-flop.

The RCVR continues framing with the sixth character; at the center of the parity bit for the sixth character, the RCVR simultaneously checks parity, does a sync comparison, and transfers the framed character into the RXDBUF. The sixth character is a sync character with bad parity. Hence, the RCVR asserts the PE (parity error) output and 200 ns later asserts the DR flag while MDET H remains asserted. PE causes the assertion of PAR ERR H (parity error) and RX ERR L. With RX ERR L asserted, RST CHAR L is cleared and the AND gate connected to the RX DONE flip-flop is conditioned. When the DR flag causes the assertion of DATA RDY H, the AND gate is satisfied and RX DONE (1) H is asserted. RX DONE (1) H causes a RCVR interrupt request and the program responds by reading the RXDBUF, which causes DTI SEL 2 L to assert. The assertion of DTI SEL 2 L triggers the DRR one-shot asserting CLR STA • DR L, which clears the DR and error flags. The trailing edge of DTI SEL 2 L clears RX DONE (1) H. Note that the sixth character (a sync character) was not stripped in this case due to a parity error.

To go a bit further and explain other situations that may develop when operating in the internal synchronous mode, an additional timing diagram is provided (Figure 4-15).

As Figure 4-15 illustrates, the first two characters received by the RCVR are sync characters with no errors. At the center of the parity bit of the first character, MDET H asserts causing MATCH L and RST CHAR L to assert. 400 ns later, MATCH L clears setting the 1st IN flip-flop. At the center of the parity bit of the second character, the DR flag asserts causing DATA RDY L to assert. DATA RDY L is ANDed with REC ACT (0) L true and triggers the DRR one-shot.

The DRR one-shot asserts CLR STA • DR L, which clears the DR flag. 400 ns later, DATA RDY L resets and sets the REC ACT flip-flop; thus the RCVR is synchronized.

The third character received is also a sync character with no errors. At the center of the parity bit, the DR flag is asserted and MDET H remains asserted. DATA RDY L then asserts and is ANDed with RST CHAR L true and RX ERR H false to trigger the DRR one-shot. The DRR one-shot asserts CLR STA • DR L and the DR flag is cleared. Thus the sync character is stripped, i.e., not presented to the program.

The fourth character received is a data character with a parity error. At the center of the parity bit, the PE flag is asserted causing PAR ERR H and RX ERR H to assert. MDET H is cleared causing RST CHAR L to clear and, 200 ns later, the DR flag is asserted and DATA RDY H asserts, causing RX DONE (1) H to assert. RX DONE (1) H causes a RCVR interrupt request and the program responds by reading the RXDBUF, which causes DTI SEL 2 L to assert. CLR STA • DR L then asserts clearing the DR and PE flags. The trailing edge of DTI SEL 2 L clears RX DONE (1) H.

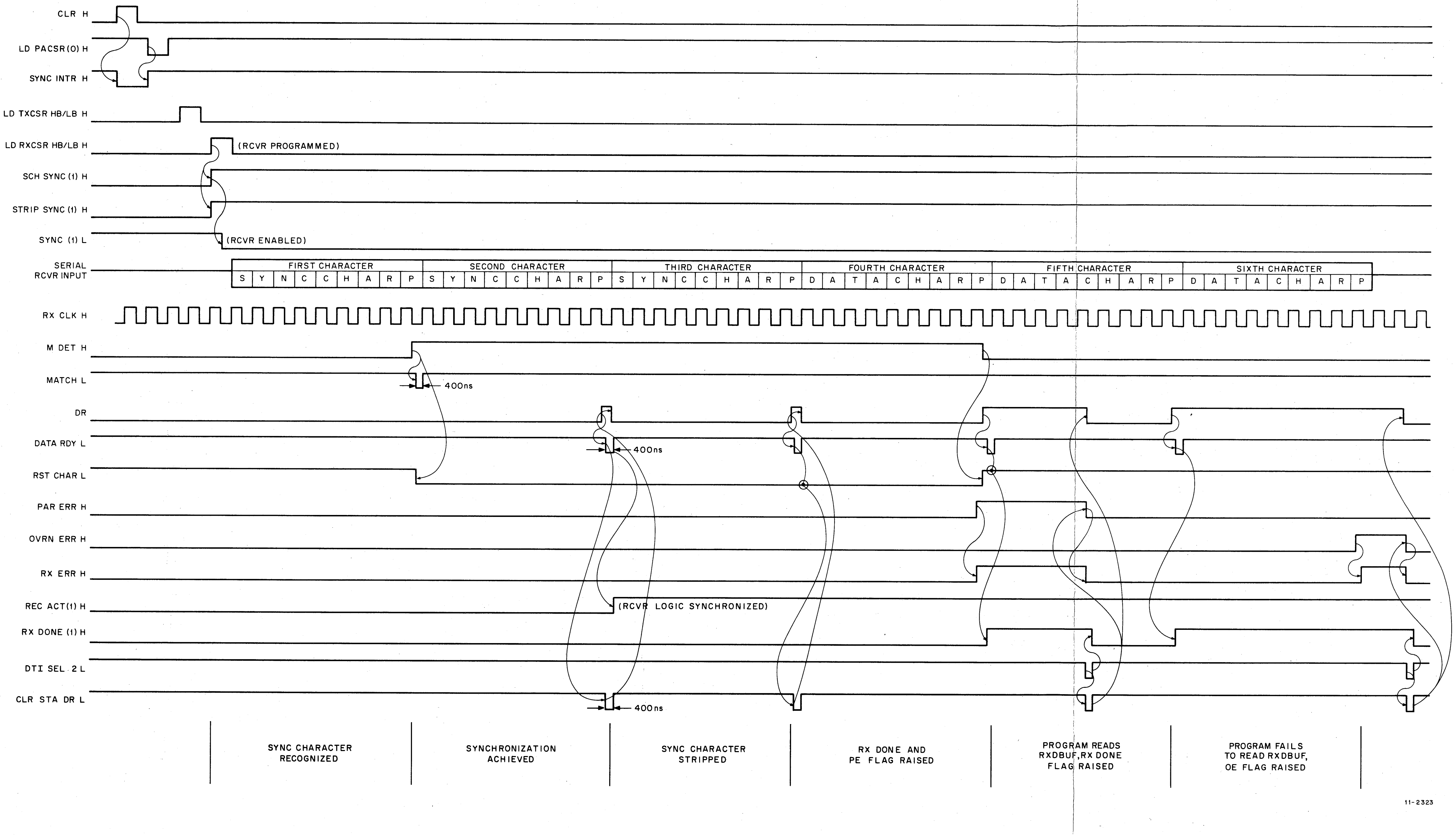


Figure 4-15 RCVR Internal Synchronous Mode Timing Diagram (Example II)

The fifth character received is also a data character but with no parity error. The DR flag causes DATA RDY H to assert which, in turn, asserts RX DONE (1) H. The RCVR interrupt request is generated; however, the program fails to read the RXDBUF before the next character (the sixth character) is framed and transferred into the RXDBUF. Hence, the fifth character is lost (overwritten) and the OE (overrun error) flag is asserted at the center of the parity bit for the sixth character causing the assertion of OVRN ERR H and RX ERR H.

The program then responds to the RCVR interrupt request caused by the fifth character and reads the RXDBUF, DTI SEL 2 L asserts, CLR STA • DR L clears the DR and OE flags, and the trailing edge of DTI SEL 2 L clears RX DONE (1) H.

4.3.6.2.2 External Synchronous Mode – In the external synchronous mode the RCVR logic sets to the synchronized condition when the SCH SYNC bit is set (Figure 4-16).

As previously stated, the program prepares the RCVR logic for operation by initializing the RCVR logic and programming the control registers. In the external synchronous mode, bits 13 and 12 of the PARSCR are set to 1 and 0, respectively. Therefore, when the PARCSR load strobe occurs, the SYNC INTR flip-flop is cleared and the SYNC mode flip-flop is set.

RCVR operation is enabled when SCH SYNC (1) H asserts. The assertion of SCH SYNC (1) H in this mode causes REC ACT (1) H to assert immediately and SYNC (1) L to assert when the RX CLK H input goes low. REC ACT (1) H conditions the RX DONE flip-flop AND gate while SYNC (1) L places a HIGH on the SS input to the RCVR. In this mode a HIGH on the SS line causes the RCVR to start framing on the very next character bit received. Hence, the RCVR begins framing on bit S of the first character and at the center of the parity bit of the first character (which, in this case, is a sync character with good parity), asserts MDET H and the DR flag. MATCH L, DATA RDY H, and RST CHAR L assert, causing CLR STA • DR L to assert, thus clearing the DR flag. Note that the RX DONE flip-flop AND gate is inhibited by RST CHAR L, hence the sync character is stripped.

The RCVR continues framing as the second character (a data character with good parity) is received. At the center of the parity bit, MDET H is cleared and the DR flag is asserted. RST CHAR L clears and DATA RDY H asserts, causing an RX DONE (1) H to assert generating a RCVR interrupt request. The program responds to the interrupt request by reading the RXDBUF which causes DTI SEL 2 L to assert. CLR STA ◦ DR L then asserts clearing the DR flag. The trailing edge of DTI SEL 2 L clears RX DONE (1) H.

4.3.6.2.3 Isochronous Mode – In the isochronous mode, each character presented to the RCVR is preceded by a START bit and succeeded by a STOP bit. Hence, the RCVR synchronizes on each received character; therefore, characters need not be presented to the RCVR contiguously (Figure 4-17).

When the PARCSR is programmed for the isochronous mode, bits 13 and 12 are cleared. Therefore, when the PARCSR load strobe occurs, the SYNC INTR flip-flop and the SYNC mode flip-flop are cleared.

RCVR operation is enabled when SCH SYNC (1) H asserts. The assertion of SCH SYNC (1) H in this mode causes REC ACT (1) H to assert immediately and SYNC (1) L to assert when the REC CLK H input goes low. REC ACT (1) H conditions the RX DONE flip-flop AND gate. However, SYNC (1) L has no effect since the OR gate connected to the SS/FE line is already receiving a low from the SYNC mode flip-flop. This is done to ensure that the RCVR FE (Framing Error) flag has absolute control over the SS/FE line. Normally in the isochronous mode the FE flag output is held LOW by the SAR chip; when a framing error is detected, the SAR chip forces FE HIGH asserting FRM ERR H and RX ERR H.

In the isochronous mode, RCVR operation is initiated when a START bit is detected on the input. Any mark to space (HIGH to LOW) transition causes the RCVR to sample the input line at the theoretical center of the START bit. If a low level input is detected the RCVR starts framing by sampling the center of succeeding data bits and shifting the data into the RCVR register. When an entire character is framed (as defined by the PARCSR) the RCVR tests the input line for a valid STOP bit, checks parity, compares the framed character to the sync character, and parallel transfers the contents of the RCVR register (minus the START and STOP bits) into the RXDBUF.

In this case the first character is a data character with good parity and a valid STOP bit; hence, the DR flag asserts, DATA RDY H asserts, and RX DONE (1) H asserts causing a RCVR interrupt request. The program responds by reading the RXDBUF, DTI SEL 2 L asserts triggering the DRR one-shot which asserts CLR STA • DR L. The assertion of CLR STA • DR L clears the DR flag. The trailing edge of DTI SEL 2 L clears RX DONE (1) H.

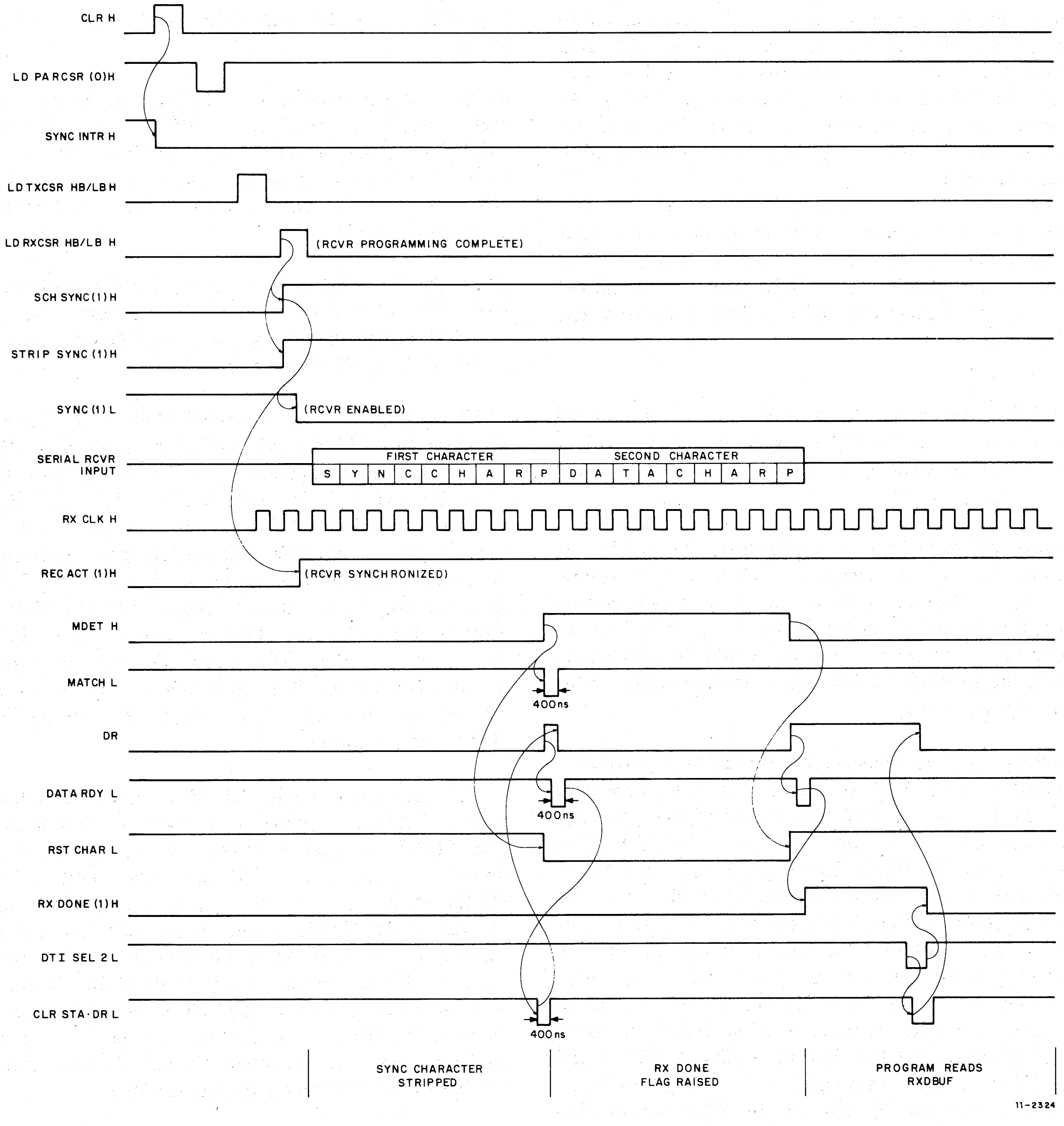


Figure 4-16 RCVR External Synchronous Timing Diagram

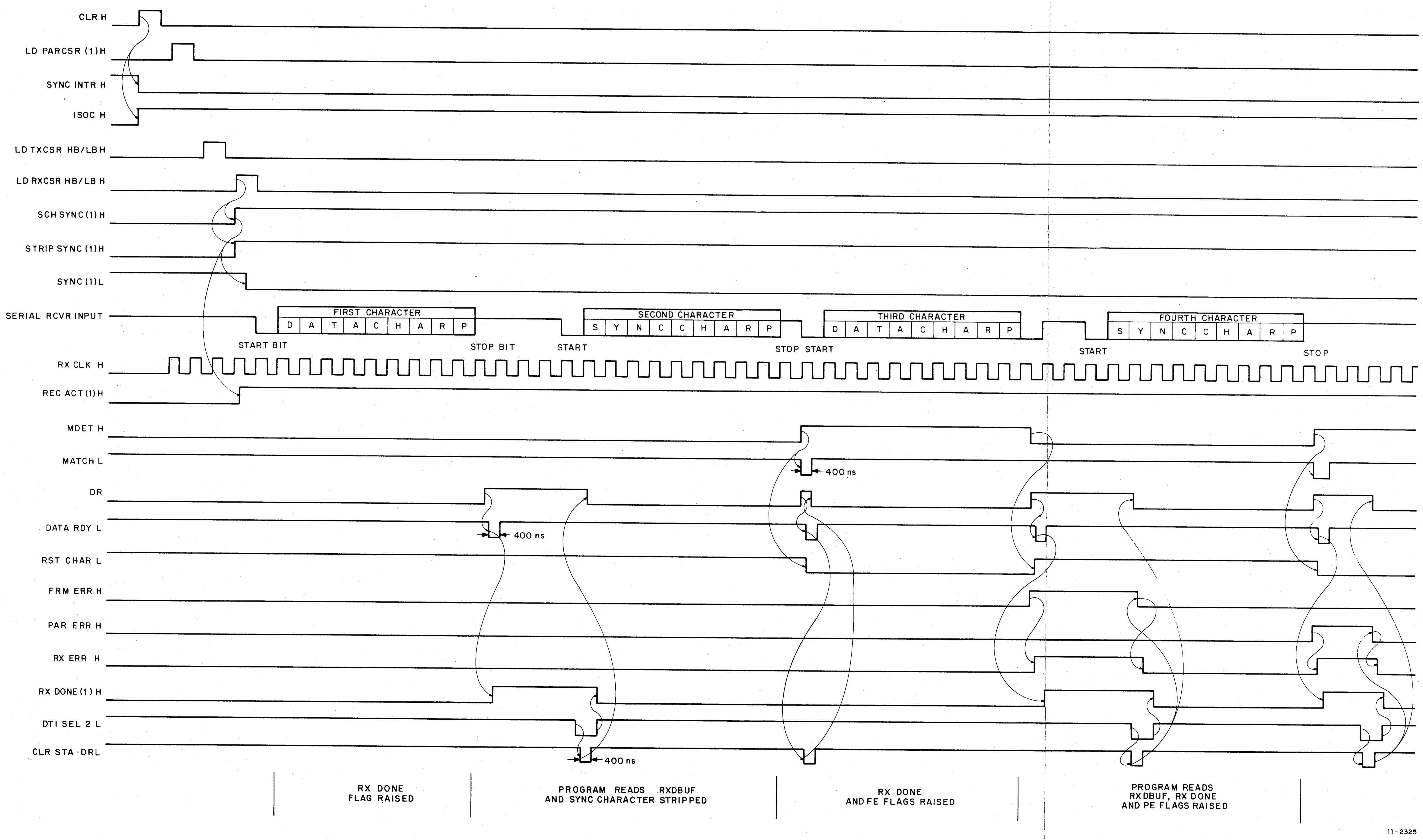


Figure 4-17 RCVR Isochronous Mode Timing Diagram

The RCVR then waits for the next START bit. When it arrives, the RCVR begins framing the second character. The second character is a sync character with good parity and a valid STOP bit. Hence at the center of the STOP bit, MDET H and the DR flag assert. MDET H asserts RST CHAR L and triggers the MATCH one-shot. The DR flag triggers the DATA READY one-shot and DATA RDY L is ANDed with RST CHAR L true and RX ERR H false to trigger to DRR one-shot. CLR STA • DR L asserts clearing the DR flag; thus the sync character is stripped.

The RCVR then receives the next START bit and begins framing the third character, which is a data character with good parity but an invalid STOP bit. Hence, at the center of the theoretical STOP bit, the FE flag is asserted causing FRM ERR H and RX ERR H to assert. 200 ns later the DR flag asserts and MDET H clears. DATA RDY H then asserts, RST CHAR L clears and RX DONE (1) H asserts causing a RCVR interrupt request. The program responds by reading the RXDBUF, DTI SEL 2 L asserts, and CLR STA • DR L asserts and clears the DR and error flags. The trailing edge of DTI SEL 2 L clears RX DONE (1) H.

The RCVR then receives the START bit for the fourth character and starts framing. The fourth character is a sync character with bad parity and a valid STOP bit. Therefore, at the center of the STOP bit the PE flag is asserted causing PAR ERR H and RX ERR H to assert; 200 ns later, MDET H and the DR flag are asserted. DATA RDY H asserts causing RX DONE (1) H to assert generating a RCVR interrupt request. The program responds, DTI SEL 2 L asserts, CLR STA • DR L asserts, DR clears, DTI SEL 2 L clears, and RX DONE (1) H clears; thus the SYNC character was not stripped due to a parity error.

4.3.7 XMTR Logic

The XMTR accepts parallel characters from the program, raises the TX DONE flag to request the next character, and serially outputs the current character to the modem. Before XMTR operation can begin, the XMTR logic must be initialized and the PARCSR and TXCSR registers programmed (Figure 4-18 and engineering drawing D5).

4.3.7.1 XMTR Logic Initialization and Programming – The XMTR logic is initialized by the CLR H input. CLR H asserted forces the XMTR to the idle state. In the idle state the timing and control logic is reset, the XMTR register output continues to MARK (Constant High), the TX DONE H flag is asserted, and the EOC L (End of Character) flag is asserted.

The XMTR is then programmed by loading the PARCSR and TXCSR. To program the PARCSR, the PARCSR load strobe is asserted, PARCSR bits 13 and 11–08 are loaded into the control register, and PARCSR bits 07–00 are loaded into the fill register. Bit 13 selects the XMTR mode of operation. If bit 13 is cleared, the XMTR operates in the isochronous mode; if bit 13 is set, the XMTR operates in the synchronous mode.

Programming the TXCSR directly controls the operation of XMTR logic and enables or disables the XMTR data output. The SEND bit must be set or XMTR operation is inhibited. If the BREAK bit is set, the XMTR data output is disabled. If the normal operating mode or the external loop maintenance mode is selected via the maintenance mode select bits, the EIA XMIT DATA output is enabled (Figure 4-18). Note that the TX INTEB and DNA INTEB bits must be set or the TX DONE and DNA flags will be ignored by the interrupt control logic.

4.3.7.2 XMTR Logic Operation – The following paragraphs provide a detailed description of the XMTR logic in each of the two operating modes. Figure 4-18 illustrates the XMTR logic. The timing diagrams (Figures 4-19 and 4-20) illustrate logic operation in response to different program inputs. The character outputs from the XMTR are shown on the timing diagrams and are numbered in the order of their transmission.

Before the XMTR logic can be discussed, the programmed operating parameters must be established. In the following discussions, the only operating parameter that changes is the mode of operation. Otherwise, the control registers are programmed as follows for the entire XMTR discussion unless stipulated differently:

a. The PARCSR is programmed for an 8 bit character with odd parity and a sync character of 026_8.

b. The TXCSR is programmed for the normal mode of operation (maintenance mode select bits cleared), to enable the XMTR (SEND bit set), to enable the XMTR output (BREAK bit cleared), to disable TX DONE interrupts (TX INTEB bit cleared), to enable DNA interrupts (DNA INTEB bit set), and to operate in the full duplex mode (HALF DUP bit cleared).

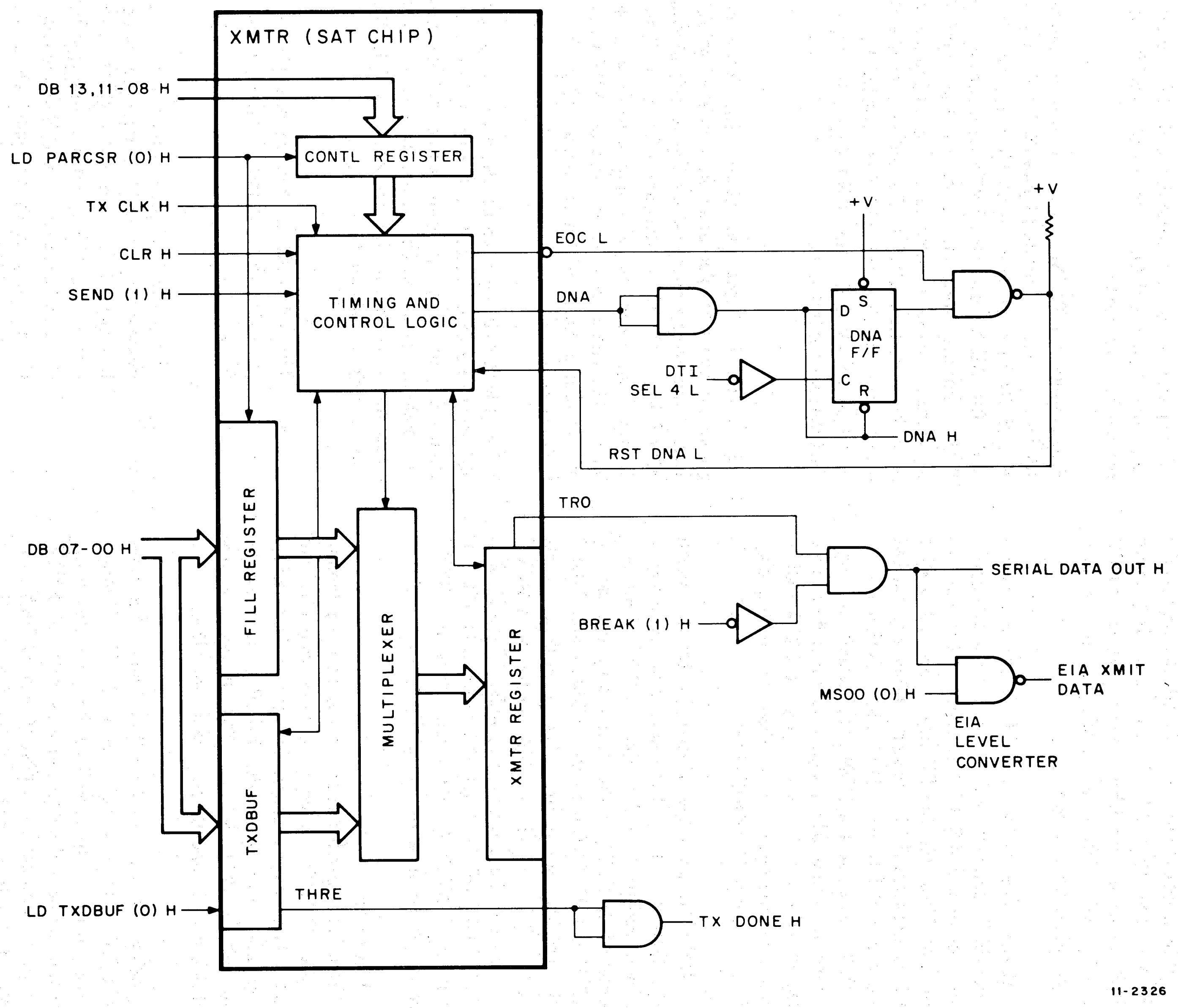


Figure 4-18 XMTR Logic

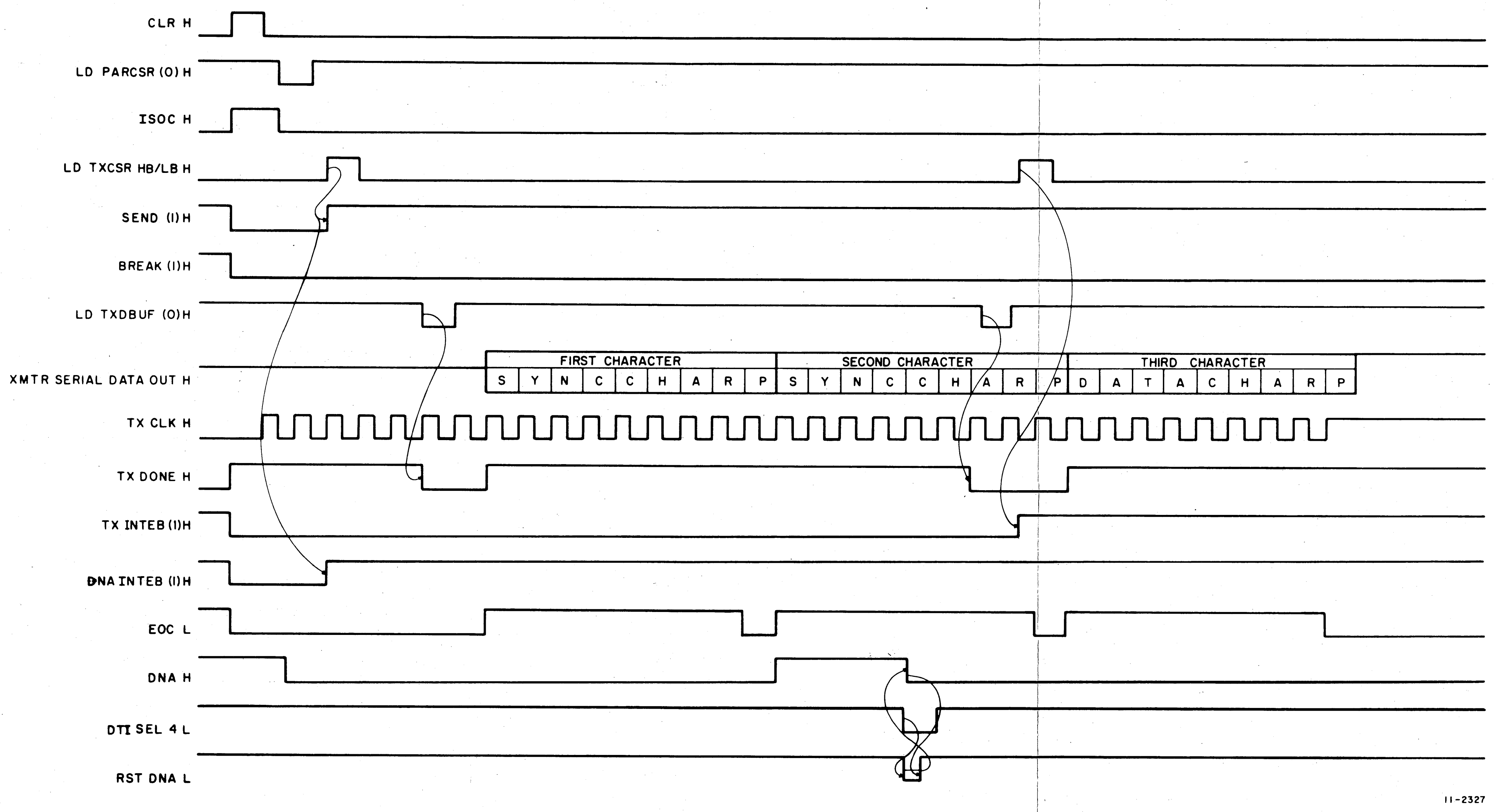


Figure 4-19 XMTR Synchronous Mode Timing Diagram

4.3.7.2.1 Synchronous Operation – In the synchronous mode, once the XMTR is enabled and outputs the first character bit stream, it will continue to output characters contiguously using fill characters whenever the program fails to load the TXDBUF.

As previously stated, the program prepares the XMTR for operation by initializing the XMTR logic and programming the PARCSR and TXCSR (Figure 4-19). In the synchronous mode, bit 13 of the PARCSR is set.

The program then initiates character transmission by loading a sync character into the TXDBUF causing TX DONE H to clear. When the TXDBUF load strobe resets, the XMTR synchronizes on the next HIGH to LOW transition of TX CLK H, delays until the next LOW to HIGH transition of TX CLK H, and then transfers the contents of the TXDBUF plus a parity bit if programmed into the XMTR register. The XMTR then places the first bit of the first character (bit S) on the TRO (Transmitter Output) line, asserts TX DONE H, and clears EOC L.

The XMTR then serially shifts the first character onto the TRO line. When the parity bit is placed on the TRO line, the XMTR asserts EOC L. At the center of the parity bit, the TXDBUF is checked to determine whether the program has loaded in another character. In this case the program did not service the TXDBUF because the TX INTEB bit was cleared; hence, the XMTR interrupt request was not generated. To maintain continuous transmission, the XMTR transfers the contents of the fill register (a sync character) plus parity into the XMTR register, places the first bit on the TRO line, clears EOC L, and asserts DNA H causing a XMTR interrupt request. The program responds by reading the TXCSR causing DTI SEL 4 L to assert. DTI SEL 4 L sets the DNA flip-flop asserting RST DNA L. RST DNA L clears DNA H, which resets the DNA flip-flop.

The program then loads a data character into the TXDBUF and sets the TX INTEB bit in the TXCSR. When the TXDBUF is loaded, the TX DONE H is cleared.

The XMTR then completes transmission of the second character. When the parity bit is placed on the TRO line, EOC L is asserted and, at the center of the parity bit, the TXDBUF is checked for the next character. At the end of the parity bit, the next character (the third character) plus parity is transferred into the XMTR register, the first bit is placed on the TRO line, TX DONE H is asserted, and EOC L is cleared. This TX DONE H will generate a XMTR interrupt request because the TX INTEB bit is set.

4.3.7.2.2 Isochronous Operation – In the isochronous mode, the XMTR adds START and STOP bits to each character transmitted. If the program fails to load the TXDBUF before transmission of the current character is complete, the XMTR simply pauses until the TXDBUF is loaded. As a result, fill characters are never transmitted and the DNA output is never asserted.

Once the XMTR is initialized, the PARCSR and TXCSR registers are programmed. In the isochronous mode, bit 13 of the PARSCR is cleared. Note that the TX INTEB bit is set in the following discussion (Figure 4-20).

The program then initiates character transmission by loading a character into the TXDBUF, thereby clearing TX DONE H. After the TXDBUF load strobe clears, the XMTR synchronizes on the next HIGH to LOW transition of TX CLK H and delays until the next LOW to HIGH transition of TX CLK H. When TX CLK H asserts, the XMTR transfers the contents of the TXDBUF plus START, parity (if programmed), and STOP bits into the XMTR register, places the START bit on the TRO line, and asserts TX DONE H. When transmission of the START bit is complete, the XMTR places the first character bit (bit D) on the TRO line and clears EOC L. While the XMTR serially shifts the first character bits onto the TRO line, TX DONE H causes a XMTR interrupt request. The program responds by reading the TXCSR and loading the second character into the TXDBUF, thus clearing TX DONE H.

When the XMTR completes transmission of the parity bit, the STOP bit is placed on the TRO line and EOC L is asserted. The next character has been loaded into the TXDBUF and the TXDBUF load strobe has cleared; therefore, when the XMTR completes transmission of the STOP bit, the contents of the TXDBUF plus the START, parity, and STOP bits are transferred into the XMTR register, and the START bit is placed on the TRO line. When transmission of the START bit is complete, the XMTR places the first bit of the second character on the TRO line and clears EOC L.

While the second character is being transmitted, TX DONE H causes another XMTR interrupt request and the program responds by reading the TXCSR and loading the third character into the TXDBUF and TX DONE H is cleared.

Also during transmission of the second character, the program clears the SEND bit in the TXCSR thus disabling the XMTR. Hence, after the XMTR places the STOP bit for the second character on the TRO line and asserts EOC L, it pauses until it is enabled again. While the XMTR pauses, its TRO line continues to Mark (HIGH).

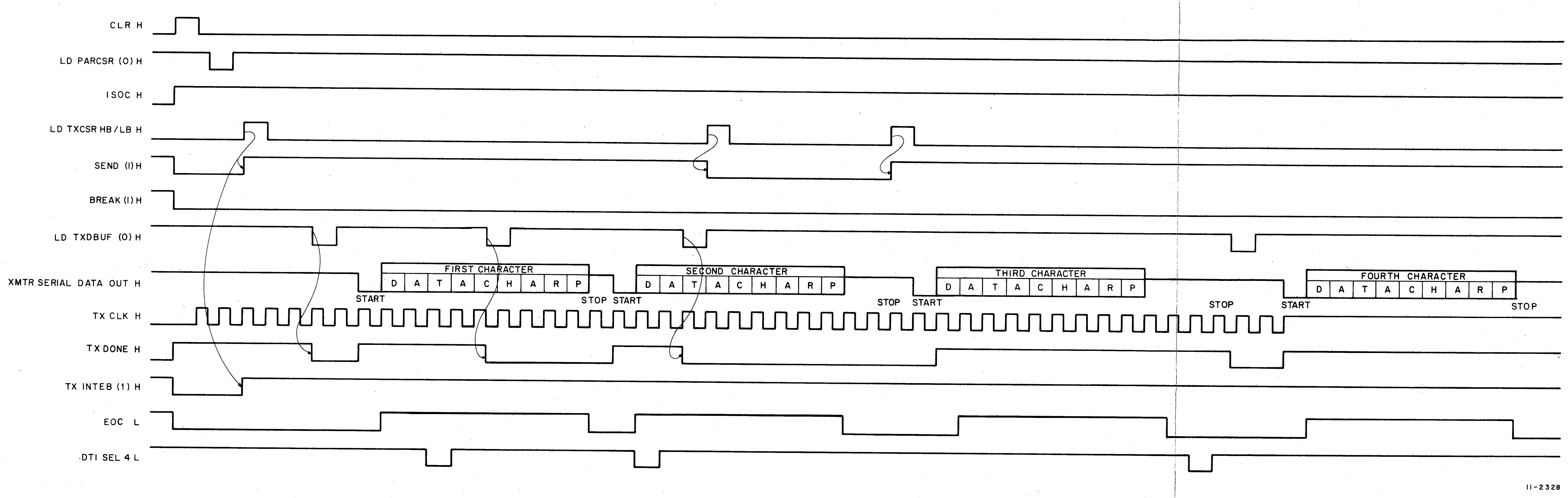


Figure 4-20 XMTR Isochronous Mode Timing Diagram

When the program enables the XMTR by setting the SEND bit, the XMTR synchronizes on the HIGH to LOW transition of TX CLK H, delays until TX CLK H asserts, transfers the contents of the TXDBUF plus the START, parity and STOP bits into the XMTR register, places the START bit on the TRO line, and asserts TX DONE H.

If the program fails to service the TXDBUF before transmission of the third character is complete, the XMTR simply pauses until the TXDBUF is serviced.

4.3.8 Data Set Change Detector Logic

The data set change detector logic monitors the modem control lines, routes the lines to the data output multiplexer as part of the RXCSR, and asserts DATSET CH (1) H whenever there is a change in any one of the control lines (engineering drawing D6).

All five control lines are electrically identical and consist of a receiver, a 600 ns delay line and an Exclusive-OR gate. Any level change at the receiver input is applied immediately to one Exclusive-OR input and delayed a minimum of 600 ns from the other input. This produces a 600 ns pulse at the output of the OR gate. Hence, the Exclusive-OR gate is satisfied causing the DATA SET CH flip-flop to set asserting DAT SET CH (1) H. When the program reads the RXCSR, DTI SEL 0 L is asserted resetting the DATA SET CH flip-flop on its trailing edge.

4.3.9 Interrupt Control Logic

The interrupt control logic enables the DU11 to gain control of the Unibus (become bus master) and cause the processor to branch to an interrupt vector that contains the new PC and PS words (engineering drawing D4).

Before RCVR and XMTR interrupts can be generated, the RXCSR and TXCSR must be programmed to enable the interrupt control logic. In the operations discussed in the following paragraphs, the RX INTEB and TX INTEB bits are set in the RXCSR and TXCSR, respectively; hence, the interrupt control logic can be activated by a RX DONE or TX DONE flag. If RX DONE and TX DONE flags occur simultaneously, the RX DONE flag takes priority.

In the sequence of operations illustrated by the interrupt control logic timing diagram (Figure 4-21), it is assumed that the Unibus is clear and that all inputs are cleared with the exception of RX INTEB (1) H and TX INTEB (1) H.

The first event is the assertion of RX DONE (1) H, which causes the assertion of REQ A H, INTR REQ H, and BR REQ L. BR REQ L is placed on the Unibus to the processor. If BUS SACK L is cleared, the processor asserts BG IN H. BG IN H causes BUS SACK L to assert clearing BR REQ L and blocking the BG OUT H signal to devices connected to the same BR level but electrically further from the processor. The processor receives the SACK signal and clears BG IN H. Assuming BUS BBSY L and BUS SSYN L are cleared from any previous bus cycles, BUS SACK L clears immediately, the V2 flip-flop sets, BUS BBSY L, BUS INTR L and the interrupt vector address assert, and the CLR INTR H AND gate is conditioned. The processor receives BUS INTR L, waits 75 ns to allow for Unibus skewing, and asserts BUS SSYN L. BUS SSYN L asserts CLR INTR H and CLR INTR L. CLR INTR H sets the REQ A flip-flop, which combines with the V2 flip-flop to inhibit further RCVR interrupt requests; CLR INTR L clears INTR REQ H, which clears the busy flip-flop thereby clearing BUS BBSY L, BUS INTR L, the vector address, and CLR INTR L.

The interrupt control logic will now begin processing the TX DONE flag, which was raised a short time after the RX DONE flag. The TX DONE flag asserted REQ B H; therefore, INT REQ H asserts again as soon as CLR INTR L is cleared. BR REQ L then asserts and the processor responds by asserting BG IN H.

Meanwhile, the processor begins executing the RCVR interrupt service subroutine and reads the RXCSR causing the assertion of DTI SEL 0 L. DTI SEL 0 L clears REQ A H after a 20 ns delay, thereby resetting the REQ A flip-flop and enabling subsequent RCVR interrupt flags to be processed.

The assertion of BG IN H, on the other hand, initiates basically the same sequence of operation as just described for the RX DONE flag with one exception, the V2 flip-flop is always reset when XMTR interrupt flags are processed, thereby asserting bit 02 (BUS D02 L) of the interrupt vector address.

In reference to the BG OUT H output of the interrupt control logic, note that BSY (1) L asserted or INTR REQ H cleared asserts EN GR H, provided BUS NPR L is cleared. If BG IN H is asserted while EN GR H is asserted, BG OUT H is asserted thus propagating the grant to the next device on the Unibus. The assertion of BUS NPR L absolutely inhibits the propagation of the grant.

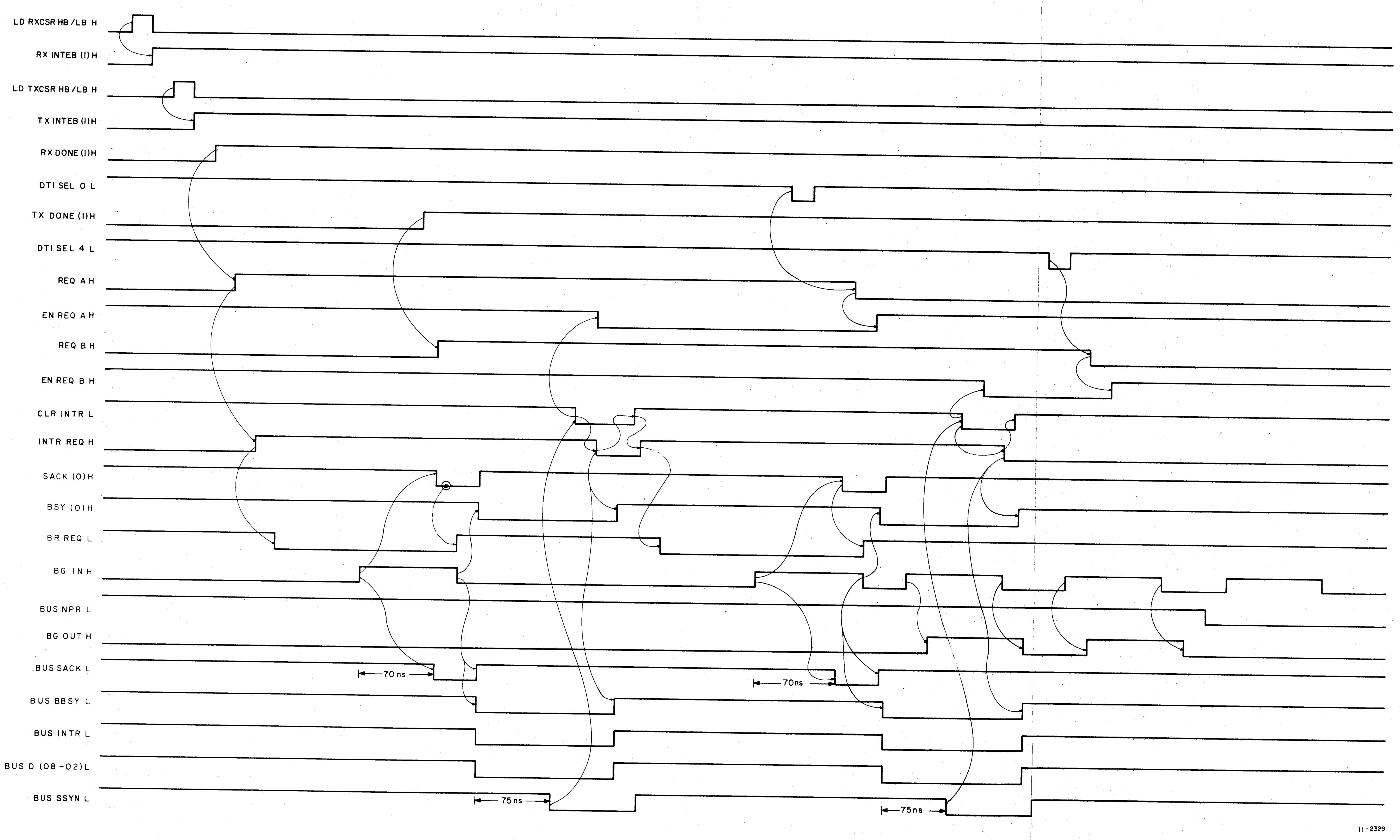


Figure 4-21 Interrupt Control Logic Timing Diagram

CHAPTER 5
PROGRAMMING REQUIREMENTS AND RECOMMENDATIONS

5.1 INTRODUCTION

To program the DU11 in the most efficient manner, the programmer must understand fully the control signal and timing requirements of the device. The following paragraphs discuss DU11 operation from a programming point of view and describe recommended programming methods. It is beyond the scope of this manual to provide detailed programming information. For more detailed information on programming in general, refer to the *Paper-Tape Software Programming Handbook,* DEC-11-XPTSA-A-D, and the individual program listings.

5.2 PROGRAMMING THE TRANSMITTER IN THE SYNCHRONOUS MODE

5.2.1 Loading the PARCSR

Once the transmitter is initialized via the BUS INIT pulse or MSTRST, the PARCSR register must be programmed (loaded) to select the mode of operation (synchronous in this case), character length, and parity. At this point the Sync register will contain all ones. Before any necessary handshaking is done with the modem, the program must load the Sync register with the desired character. When the Sync register is loaded, the character will be used for both XMTR and RCVR operation.

5.2.2 Enabling the Transmitter

Once handshaking is complete, the program can assert the SEND bit in the TXCSR. When SEND is asserted, the XMTR is enabled but will not start transmitting data until the first character is loaded into the TXDBUF. If SEND is cleared during transmission, the character currently being transmitted will be completed, the transmit line will go to a mark hold state, the internal XMTR logic will enter the idle state, and synchronization with the RCVR will be lost. When SEND is cleared, there is no guarantee that the TX DONE bit will be asserted when current character transmission is complete.

5.2.3 Detecting the Last Character of the Message

When it is necessary to know when the entire message has been transmitted, the DU11 may be programmed as follows:

a. Just prior to loading the last character of the message into the TXDBUF, clear the TX DONE INTEB bit and set the DNA INTEB bit.

b. When the last character is loaded into the register, the TX DONE bit will set but will not cause an interrupt request.

c. After the last character is transmitted, the transmitter will transmit the sync character and assert DNA, which causes an interrupt request.

d. The DNA interrupt is notification to the program that the entire message has been transmitted.

5.2.4 Transmitting Initial Sync Characters to Establish Synchronization

The transmission of initial sync characters can be accomplished in one of two ways:

a. The program may arrange its data buffer such that the required number of sync characters precede any messages. In this case, the Sync register may or may not contain the sync character. If the Sync register is not loaded, it will contain all ones subsequent to a BUS INIT or MSTRST.

   Assuming that any necessary handshaking has been completed and that SEND has been asserted, the program can commence transmission by loading a sync character into the TXDBUF. When the first data bit is transferred to the communication line, the TX DONE bit will be asserted. If the TX INTEB bit is set, an interrupt request will be generated, and the program must load another sync character into the TXDBUF.

If the sync character was not initially loaded into the Sync register, then synchronization cannot be guaranteed unless the program response time to the TX DONE bit is less than (1/baud rate X bits per char) seconds - 1/2 (bit time). This can be verified by the absence of the DNA bit in the TXCSR.

b. The program can also enable transmission of initial sync characters from the Sync register. Assuming any necessary handshaking is complete and SEND is asserted, the program loads the Sync register with a sync character, sets the DNA INTEB bit, and clears the TX INTEB bit. The program then loads a sync character into the TXDBUF and transmission begins. The TX DONE interrupt is inhibited so the contents of the Sync register are transferred to the XMTR register upon the completion of transmission of the first sync character. The second sync character is then transmitted and a DNA interrupt is generated notifying the program. The program then allows the transmission of sync characters to continue by simply monitoring the DNA until the desired number have been transmitted. Note that DNA is reset each time the program reads the TXCSR and set again when the first bit of the next sync character is placed on the communication line.

NOTE

It is suggested that a minimum of five sync characters be transmitted. In systems that are prone to error because of lost synchronization, as many as twelve sync characters may be desirable.

When the desired number of sync characters have been transmitted, the program sets the TX INTEB bit, thereby enabling the TX DONE interrupt, and responds to the interrupt by loading a message character into the TXDBUF.

5.2.5 Transmitting Sync Characters to Maintain Synchronization

After synchronization has been achieved, it can be maintained by the program by inserting sync characters into the message or by ignoring the TX DONE bit, thereby allowing sync characters from the Sync register to be transmitted.

If the latter method is chosen, it can be programmed in one of two ways. The first way would be to set the DNA INTEB bit and clear the TX INTEB bit. The program would then ignore the TX DONE bit and the XMTR would transmit a sync character and assert DNA. The program would monitor the DNA bit and, when the desired number of sync characters are transmitted, set the TX INTEB bit thereby enabling the TX DONE interrupt. The program would then respond to the TX DONE interrupt by loading a message character into the TXDBUF, thereby terminating the transmission of sync characters. The second way would be to clear the TX INTEB and DNA INTEB bits for a given period of time during message transmission thereby allowing sync characters from the sync register to be transmitted.

NOTE

The SEND bit in the TXCSR must remain set for the duration of the message; any on to off transition will cause the XMTR to enter an idle state after completion of current character transmission.

5.3 PROGRAMMING THE RCVR IN THE INTERNAL SYNCHRONOUS MODE

Once the program has completed any necessary handshaking, the receiver logic can be enabled. The program enables the receiver logic by setting the SCH SYNC (Search Sync) bit in the RXCSR. Assuming a sync character has been loaded into the Sync register (this must be done in the internal synchronous mode), the receiver begins to compare incoming character bits with the character held in the Sync register.

NOTE

For the receiver to become synchronized with XMTR either one or two contiguous sync characters must be received. The number of sync characters required is jumper selectable. The standard configuration requires two sync characters.

NOTE

Though the DU11 may be jumpered to synchronize on two contiguous sync characters, there is a situation which, if it develops, will prevent RCVR synchronization on only two contiguous sync characters. If, while the DU11 is searching for synchronization, it recognizes a sync character that is not followed contiguously by a second sync character, the RCVR internal logic resets, thereby inhibiting the RCVR bit detection logic for two bit times. Should the first bits of a proper sync character sequence occur during that two bit time period, the RCVR will fail to achieve synchronization.

When two contiguous sync characters are received, the REC ACT (Receiver Active) bit is set and any characters received after that will cause RX DONE interrupt requests, provided the RX INTEB bit is set and the STRIP SYNC bit is cleared.

NOTE

The SCH SYNC bit must remain set for the duration of the message. If not, the character being received at the time of the on to off transition will be lost along with synchronization.

If the programmer wishes the RCVR to discard all sync characters after synchronization is achieved, the STRIP SYNC bit in the RXCSR must be set. The STRIP SYNC bit inhibits the RX DONE interrupt whenever a sync character is received with no errors; however, the sync character is still held in the RXDBUF until the next character is received.

If the program fails to read the RXDBUF in response to a RX DONE interrupt, overrun errors will occur. When the RXDBUF is not serviced in the time required to receive the next character, i.e., (1/baud rate X bits per character) seconds, the character presently being held in the RXDBUF is overwritten by the next received character and the OVRN ERR (overrun error) bit is set in the RXCSR.

NOTE

The information in the following paragraph must be strictly adhered to or RCVR synchronization problems will be encountered.

If the DU11 is configured to achieve synchronization on two contiguous sync characters then receiver operation may be terminated (after the entire message is received) by simply clearing the SCH SYNC bit in the RXCSR. However, if only one character is required to achieve synchronization, receiver termination is a little more complex. If the SCH SYNC bit is cleared while a sync character is present in the RXDBUF, false synchronization will occur when the receiver is enabled (SCH SYNC bit set) to receive the next message. The program must ensure that this does not happen by transmitting a pad character, i.e., a non-sync character, immediately after the transmission of the terminating control character.

5.4 PROGRAMMING THE RCVR IN THE EXTERNAL SYNCHRONOUS MODE

The external synchronous mode enables the RCVR logic to set to the synchronize state immediately upon the assertion of the SCH SYNC (1) H input. This mode is designed for use with communication equipment capable of accomplishing synchronization external to the DU11. When the program sets the SCH SYNC bit, the REC ACT bit sets and the RCVR starts framing characters on the very next bit received. When the selected number of bits are received, the received character is transferred into the RXDBUF and the RX DONE bit is set causing an interrupt request. All other features and parameters of the internal synchronous mode apply to this mode also.

5.5 PROGRAMMING THE XMTR IN THE ISOCHRONOUS MODE

5.5.1 Loading the PARCSR

Once the XMTR is initialized via BUS INIT or MSTRST, the PARCSR must be programmed to select the mode of operation (isochronous in this case), character length, and parity. It is not necessary to load the sync register in this mode as sync characters are not required to achieve synchronization and the transmitter is not required to transmit continuously.

5.5.2 Enabling the XMTR

When the required handshaking is complete, the program sets the SEND and TX INTEB bits and loads a character into the TXDBUF. The XMTR adds the START and STOP bits and transmits the character to the modem. As soon as the first character bit is placed on the communication line by the XMTR, the TX DONE bit is asserted and remains asserted until the XMTR services the TXDBUF or clears the SEND bit.

5.6 PROGRAMMING THE RCVR IN THE ISOCHRONOUS MODE

RCVR operation is initiated by the assertion of SCH SYNC. When the program sets the SCH SYNC bit, the REC ACT bit sets and the RCVR starts framing characters upon receipt of the START bit from the XMTR. When the selected number of character bits are received, the RCVR tests the line for a valid STOP bit, transfers the received character into the RXDBUF, and sets the RX DONE bit. If the STOP bit is not detected, the FRM ERR (Framing Error) bit is also set. If the program fails to service the RXDBUF before the next character is framed, the OVRRN ERR bit is set.

CHAPTER 6
MAINTENANCE

6.1 SCOPE

This chapter lists required test equipment and provides a complete description of DU11 preventive and corrective maintenance procedures.

6.2 MAINTENANCE PHILOSOPHY

Basically, DU11 maintenance consists of preventive and corrective maintenance procedures, diagnostic programs, and a maintenance log. The preventive maintenance procedures are performed regularly in an attempt to detect any deterioration due to aging and any damage caused by improper handling of the module. The corrective maintenance procedures are performed to isolate and repair faults in module circuitry only after it has been determined that the module is faulty. The maintenance log is used to record all maintenance activities for future reference and analysis; hopefully, the log will facilitate future maintenance action and aid in detecting any component failure pattern that may develop.

6.3 PREVENTIVE MAINTENANCE

Preventive maintenance consists of tasks performed at periodic intervals to ensure proper equipment operation and minimum unscheduled downtime. These tasks consist of running diagnostics, visual inspection, operational checks, and replacement of marginal components.

The preventive maintenance schedule depends on the environmental and operating conditions that exist at the installation site. Under normal conditions, recommended preventive maintenance consists of inspection and cleaning every 600 hours of operation or every 4 months, whichever occurs first. However, relatively extreme conditions of temperature, humidity, dust, and/or abnormally heavy work loads demand more frequent maintenance. In any case, the diagnostic programs should be run once per week as part of the normal preventive maintenance schedule.

6.4 TEST EQUIPMENT REQUIRED

Maintenance procedures for the DU11 require the standard test equipment and diagnostic programs listed in Table 6-1, in addition to standard hand tools, cleaners, test cables, and probes.

6.5 CORRECTIVE MAINTENANCE

The corrective maintenance procedures are designed to aid the maintenance technician in isolating and repairing faults within the DU11 module. Hence, the technician must be otherwise equipped to determine that the DU11 is, in fact, at fault.

Table 6-1
Test Equipment Required

Equipment	Manufacturer	Designation
Multimeter	Triplett or Simpson	Model 630-NA or 260
Oscilloscope	Tektronix	Type 453
X10 Probes (2)	Tektronix	P6008
Module Extender	DIGITAL	W984 (QUAD Height)
Modem Test Connector	DIGITAL	H315A
Diagnostics (Maindecs)	DIGITAL	MAINDEC-11-DZDUA

The diagnostic programs comprise the basic tool used by the technician to isolate faults. The diagnostics exercise the DU11 in three distinct maintenance modes and provide printouts indicating the results. The printouts point the technician to a particular logic area such as the XMTR or RCVR logic. The technician uses standard test equipment (scope and probe) to further isolate the fault to a specific circuit component.

6.5.1 Maintenance Modes

The three maintenance modes are:

a. System Test
b. Internal Loop
c. External Loop

6.5.1.1 System Test Mode – This mode is selected by the system test diagnostic program. The system test diagnostic exercises all devices and interfaces connected to the Unibus and is run with the DU11 connected to the modem. The following operations are performed to exercise the DU11 (Figure 6-1):

a. The system test clock located within the DU11 is enabled [MS 01 (1) H and MS 00 (1) H asserted] and provides clock pulses to the RCVR and XMTR.

b. The XMTR and RCVR are enabled [SEND (1) H and SCH SYNC (1) H asserted].

c. A character is loaded into the TXDBUF.

d. The XMTR output (SERIAL DATA OUT H) is routed to the RCVR via the RCVR input select logic.

e. The program monitors the RX DONE and the RXDBUF for errors.

6.5.1.2 Internal Loop Mode – This mode is designed to isolate the faults within the DU11 to one of the following logic areas:

a. XMTR
b. RCVR
c. DNA
d. TX DONE
e. RX DONE
f. Synchronization

The internal loop maintenance mode enables the program to exercise the interface logic without physically disconnecting the interface from the modem. This is made possible by maintenance mode logic that electrically inhibits the clock and data channels [MS 00 (0) H cleared] between the interface and the modem. Note, however, that the modem control lines are not inhibited; therefore, care must be taken when this mode is programmed not to activate any of the modem control lines.

When the internal loop maintenance mode is programmed, the maintenance mode select bits (MS 01, MS 00) establish the required operating conditions as follows (Figure 6-1 and engineering drawings D5 and D6):

a. MS 00 (0) H is cleared thus inhibiting the modem clock inputs (TRS CLK and REC CLK), the modem input to the RCVR input select logic (SERIAL DATA IN), and the XMTR output to the modem (SERIAL DATA OUT H).

b. MS 01 (1) H and MS 00 (0) H are cleared thus enabling the program input to the RCVR via the RCVR input select logic [MAINT DATA (1) H] and the RX INP H input to the TXCSR.

NOTE

When programming the internal loop maintenance mode, ensure that SERIAL DATA OUT H is disabled (BREAK bit set) whenever the MAINT DATA (1) H input to the RCVR input select logic is active.

c. MS 00 (1) H is asserted thus enabling the programmable single step clock input [SS CLK (1) H] to drive the RCVR and XMTR and the XMTR output to the RCVR via the RCVR input select logic (SERIAL DATA OUT H).

Once the operating conditions are established, the program performs the following operations:

a. The single step clock is programmed (by alternately setting and clearing the SS CLK bit) to provide clock pulses to the RCVR and XMTR.

b. The RCVR and XMTR are enabled.

c. A character is loaded into the TXDBUF.

d. The XMTR then serially outputs the character to the RCVR via the SERIAL DATA OUT H line.

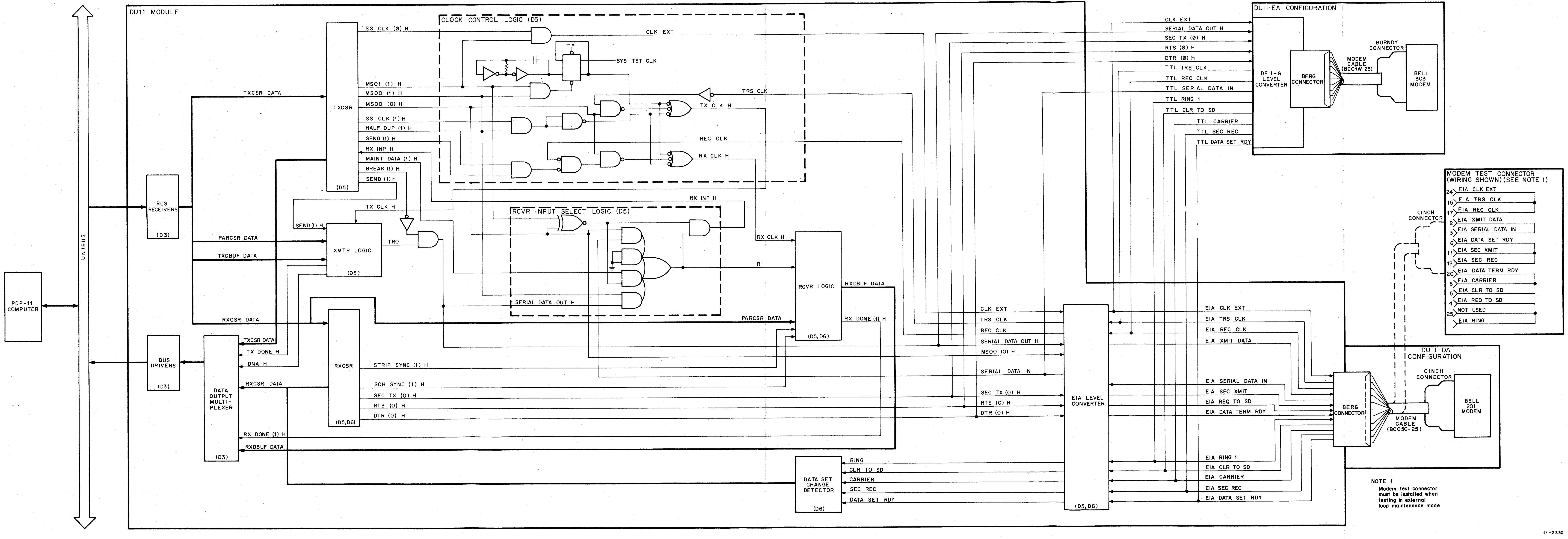


Figure 6-1 DU11 Maintenance Diagram

The program then monitors the RX INP bit, the RX DONE bit, and the RXDBUF for errors. To further isolate any errors that may be detected, the program sets the BREAK bit and inputs data directly to the RCVR via the MAINT DATA (1) H line.

6.5.1.3 External Loop Mode – This mode is designed to isolate faults occurring in the cabling connecting the DU11 to the modem, as well as faults within the DU11 level converters and data set change detector logic. Before this mode can be executed the interface must be physically disconnected from the modem and the modem test connector (H315A) installed at the modem end of the BC05C modem cable. Figure 6-1 illustrates the proper installation of the modem test connector.

When the external loop maintenance mode is programmed, the maintenance mode select bits establish the required operating conditions as follows (Figure 6-1 and engineering drawings D5 and D6):

a. MS 00 (0) H is asserted thus enabling the TRS CLK and REC CLK inputs to the clock control logic, the SERIAL DATA IN input to the RCVR input select logic and the XMTR output to the modem test connector (SERIAL DATA OUT H).

b. MS 01 (1) H and MS 00 (0) H are asserted thus enabling the program input to the RCVR via the RCVR input select logic [MAINT DATA (1) H] and the RX INP H input to the TXCSR.

NOTE

When programming the external loop maintenance mode, care must be taken to ensure that SERIAL DATA OUT H is disabled (Break bit set) whenever the MAINT DATA (1) H input to the RCVR input select logic is active.

c. MS 01 (1) H is asserted thus enabling the programmable single step clock input [SS CLK (0) H] to drive the RCVR and XMTR via CLK EXT output to the modem test connector.

Once the operating conditions are established, the program performs the following operations to check out the modem cabling.

a. The single step clock is programmed to activate the EIA CLK EXT output which is looped back by the modem test connector and applied to the TRS CLK and REC CLK input lines.

b. The RCVR and XMTR are enabled.

c. A character is loaded into the TXDBUF.

d. The XMTR then serially outputs the character to the modem test connector, which loops it back to the interface where it is applied to the RCVR via the SERIAL DATA IN line.

The program then monitors the RX INP bit, the RX DONE bit, and the RXDBUF for errors. To further isolate any errors that may be detected, the program sets the BREAK bit and inputs data directly to the RCVR via the MAINT DATA (1) H line.

To check out the modem control lines, the program individually sets and clears the modem control bits (bits 01, 02, and 03 in the RXCSR) and monitors the modem control lines and the DAT SET CH bit for errors.

APPENDIX A
REPRESENTATIVE MODEM FACILITIES AVAILABLE

Manufacturer	Model	Speed (Maximum)	Half or Full Duplex	Sync or Async	Type of Line	Comments
Bell System	103A	300 baud	Full Duplex	Async	DDD	
Bell System	103E	300 baud	Full Duplex	Async	DDD	Similar to 103A
Bell System	103F	300 baud	Full Duplex	Async	Private	
Bell System	113A	300 baud	Full Duplex	Async	DDD	Originate Only
Bell System	113B	300 baud	Full Duplex	Async	DDD	Answer Only
Bell System	201A	2000 baud	Either	Sync	DDD	Full Duplex on 2 calls
Bell System	201B	2400 baud	Either	Sync	Private	
Bell System	202B	1800 baud	Either	Async	DDD	
Bell System	202C	1200 baud	Either	Async	DDD	Full Duplex on 2 calls
Bell System	202D	1800 baud	Either	Async	Private	
Bell System	205B	600 baud 1200 baud 2400 baud	Full Duplex	Sync	Private	
Bell System	202E Series	1200 baud	Trans Only	Async	DDD Private	
Bell System	301B	40,800 baud	Either	Sync	Private Wide Band	
Bell System	303B, C, D, E	19,000 to 230,400 baud	Either	Sync	Private Wide Band	
Bell System	811B	110 baud	Either	Async	TWX Network	
Western Union	118-1A	180			Telegraph	
Western Union	1601-A	600			Voice	
Western Union	2121-A	1200			Voice Broad Band	
Western Union	2241-A	2400	Either	Either	Broad Band	
Western Union	100	200	Either	Async	Voice	

Manufacturer	Model	Speed (Maximum)	Half or Full Duplex	Sync or Async	Type of Line	Comments
Western Union	100	2400	Either	Async	Voice	
Western Union	300	18,000 40,000	Either	Sync	Broad Band	
Rixon	FM-12	1200	Either	Either	Voice Bell 4A	
Rixon	Sebit 48	4,800	Either	Sync	Voice Bell 4C	
General Electric	TDM 220	2,400	Either	Either	Private Bell 4B	

APPENDIX B
ADDRESS ASSIGNMENTS

B.1 FLOATING VECTORS

There is a floating vector convention used for communications (and other) devices that interface with the PDP-11 computer. These vector addresses are assigned, in order, starting at 300 and proceeding upwards to 777. Table B-1 shows the assigned sequence. It can be seen that the first vector address, 300, is assigned to the first DC11 in the system. If another DC11 is used, it would then be assigned vector address 310, etc. When the vector addresses have been assigned for all the DC11s (up to a maximum of 32), addresses are then assigned consecutively to each unit of the next highest ranked device (KL11 or DP11 or DM11, etc.), then to the other devices in accordance with the priority ranking (Table B-1).

If any of these devices are not included in a system, the vector address assignments move up to fill the vacancies. If a device is added to an existing system, its vector address must be inserted in the normal position and all other addresses must be moved accordingly. If this procedure is not followed, DEC software cannot test the system.

NOTE

The floating vectors range from addresses 300 to 777, but addresses 500 through 534 are reserved for special bus testers. In addition, address 1000 is used for the DS11 Synchronous Serial Line Multiplexer. Refer to Appendix A of the *PDP-11 Peripherals Handbook*, 1973-1974, for a complete discussion of Unibus addresses.

B.2 FLOATING DEVICE ADDRESS

There is a floating address convention for communication (and other) devices interfacing with PDP-11 computers. These addresses are assigned in order starting at 760010 and proceeding upward to 763776. Refer to Table B-2 for floating address sequence.

Table B-1
Priority Ranking for Floating Vectors
(starting at 300 and proceeding upwards)

Rank	Device	Vector Size (in octal)	Max. No.
1	DC11	10	32
2	KL11, DL11-A, DL11-C	10	16
3	DP11	10	32
4	DM11-A	10	16
5	DN11	4	16
6	DM11-BB	4	16
7	DR11-A	10*	32
8	DR11-C	10*	32
9	PA611 Reader	4*	16
10	PA611 Punch	4*	16
11	DT11	10*	8
12	DX11	10*	4
13	DL11-C, DL11-D, DL11-E	10	31
14	DJ11	10	16
15	DH11	10	16
16	GT40	10	1
17	LPS11	30*	1
18	DQ11	10	16
19	KW11-W	10	1
20	DU11	10	16

*The first vector for the first device of this type must always be on a $(10)_8$ boundary.

Table B-2
Floating Address Sequence

Rank	Device	Address Boundary Starting at 760010
1	DJ11	10 X (N) + 2
2	DH11	20 X (N) + 2 (go to next 20, 40, 60, or 100 boundary)
3	DQ11	10 X (N) + 2 (go to next 10, 20, 30, 40, 50, 60, 70, or 100 boundary)
4	DU11	10 X (N) + 2 (go to next 10, 20, 30, 40, 50, 60, 70, or 100 boundary)

N = number of each device

Table B-3
Floating Device Address Assignments

Device	First Address	Number of Register Addresses
DJ11 (GAP)	760010	–
DH11 #0	760020	8
DH11 #1	760040	8
DH11 (GAP)	760060	–
DQ11 #0	760070	4
DQ11 #1	760100	4
DQ11 (GAP)	760110	–
DU11 #0	760120	4
DU11 #1	760130	4

If, for example, a communication system is to contain two DH11s, two DQ11s, two DU11s and no DJ11s, the floating addresses would be assigned as shown in Table B-3.

If a DU11 in a system is not preceded by other devices in the floating vector area, it must have a starting address of 1600 for zero.

APPENDIX C
IC SCHEMATICS

The DU11 interface employs six types of integrated circuit (IC) chips in its design. A detailed schematic of each type, including a packaging diagram with pin number designations, and a truth table, is given in this appendix.

The following IC schematics are included in this appendix:

7442	4-Line-to-10-Line Decoder
74123	Monostable Multivibrator
74153	Dual 4-Line-to-1-Line Data Selector
74174	Hex D-Type Flip-Flop with Clear
74175	Quad D-Type Flip-Flop with Clear
74H74	D-Type Edge-Triggered Flip-Flop

7442 4-LINE-to-10-LINE DECODERS

In the DU11, the 7442 is used as a 3-wire binary to octal decoder as input D is used as a strobe. When D goes LOW, inputs A, B, C are decoded forcing one of eight outputs (0–7) low (see the truth table and Figure C-1).

Truth Table

BCD Input				Octal Output							
D	C	B	A	0	1	2	3	4	5	6	7
0	0	0	0	0	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	0
1	X	X	X	1	1	1	1	1	1	1	1

X = Irrelevant

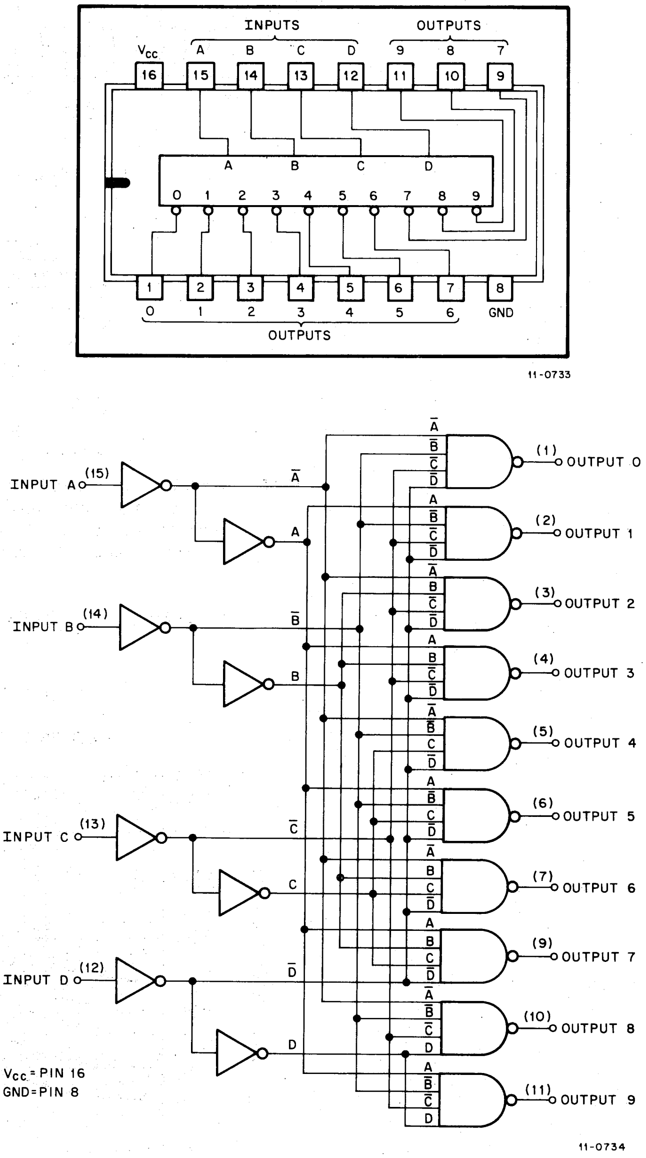
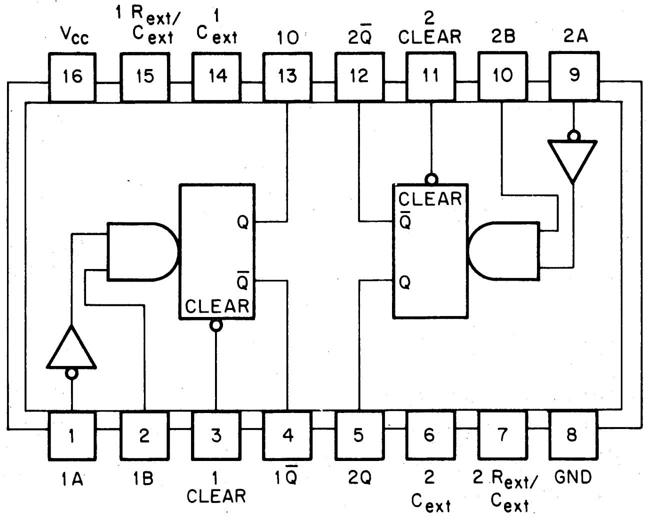


Figure C-1 7442 Package and Logic Diagrams

74123 MONOSTABLE MULTIVIBRATOR



FUNCTIONAL LOGIC / PIN LOCATOR

TRUTH TABLE

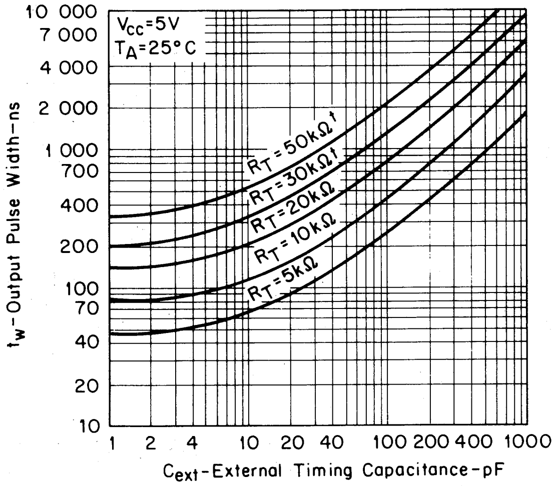
INPUTS		OUTPUTS	
A	B	Q	$\bar{Q}$
H	X	L	H
X	L	L	H
L	↑	⎍	⊔
↓	H	⎍	⊔

NOTE: H = high level (steady state), L = low level (steady state),
↑ = transition from low to high level, ↓ = transition from
high to low level, ⎍ = one high-level pulse, ⊔ = one
low-level pulse, X = irrelevant (any input, including transitions).

8E-0516

Figure C-2 DEC 74123 IC Illustrations

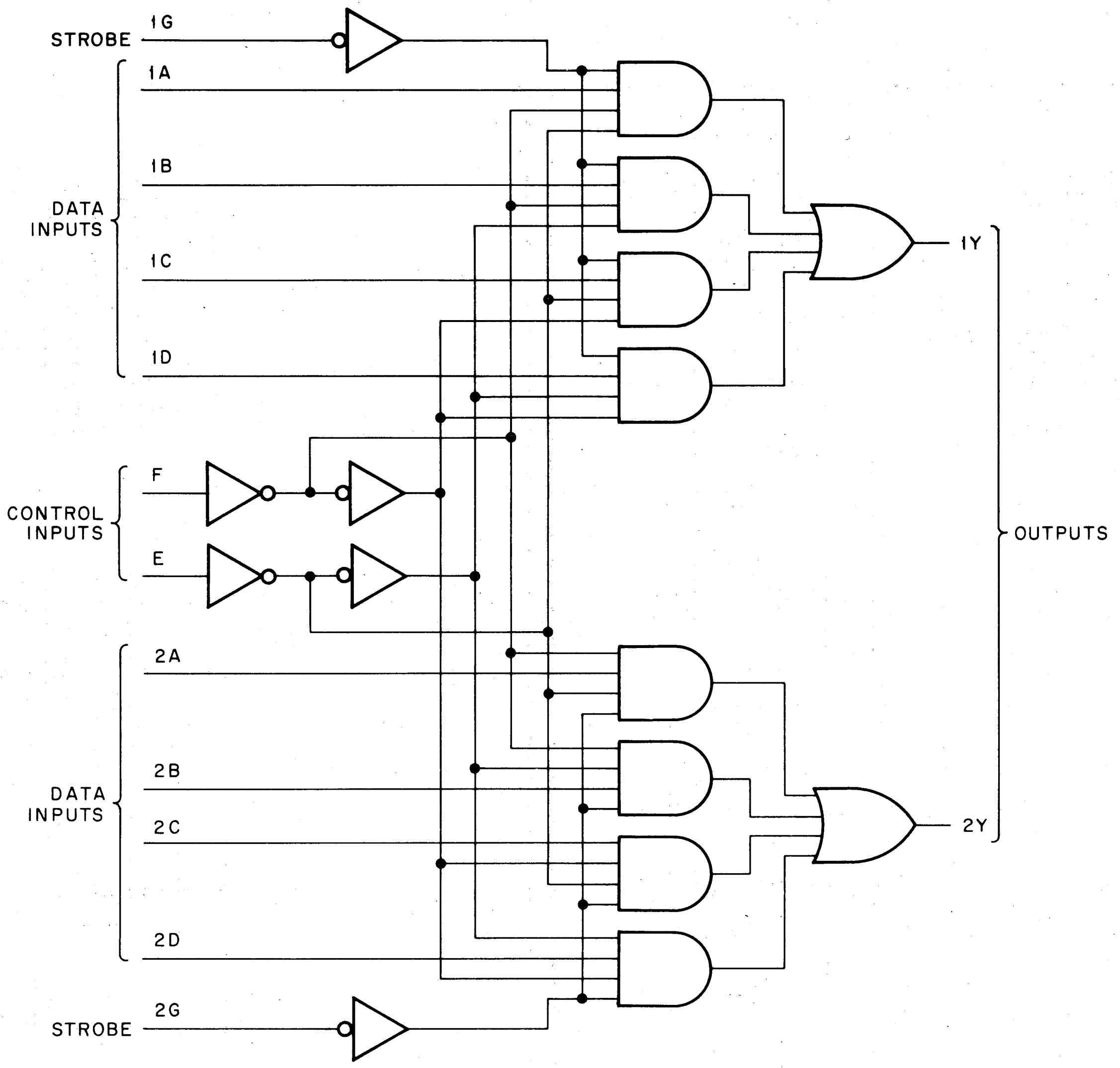
OUTPUT PULSE WIDTH
vs
EXTERNAL TIMING CAPACITANCE



8E-0524

Figure C-3 DEC 74123 IC Output Pulse Width vs External Timing Capacitance

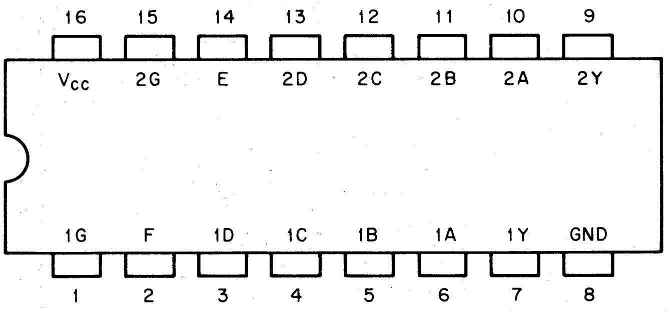
74153 4-LINE-TO-1-LINE MULTIPLEXER



LOGIC DIAGRAM

CONTROL INPUT		STROBE	OUTPUT
E	F	G	Y
LOW	LOW	LOW	A
HIGH	LOW	LOW	B
LOW	HIGH	LOW	C
HIGH	HIGH	LOW	D
DON'T CARE		HIGH	LOW

TRUTH TABLE (EACH HALF)

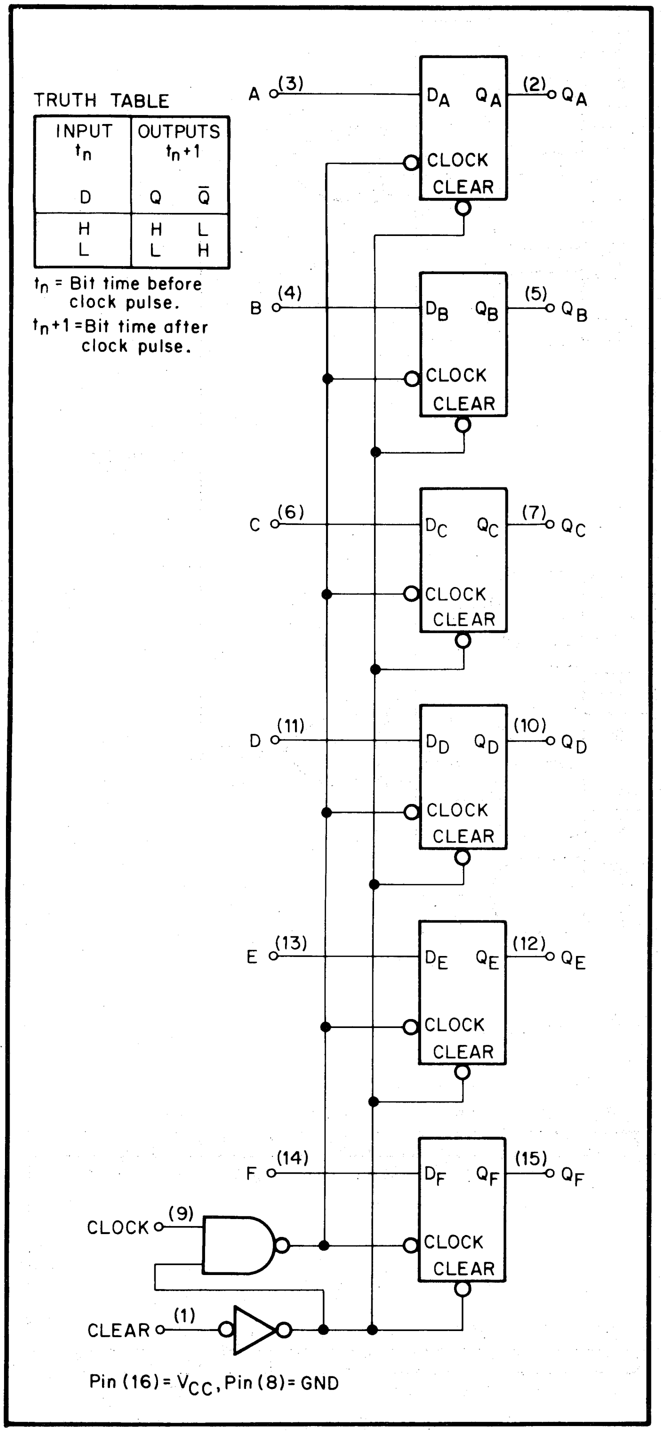


PIN LOCATOR
(TOP VIEW OF IC)

8E-0138

Figure C-4 74153 Package and Logic Diagram

74174 HEX/74175 QUAD D-TYPE FLIP-FLOPS WITH CLEAR



TRUTH TABLE

INPUT t_n	OUTPUTS t_n+1	
D	Q	$\bar{Q}$
H	H	L
L	L	H

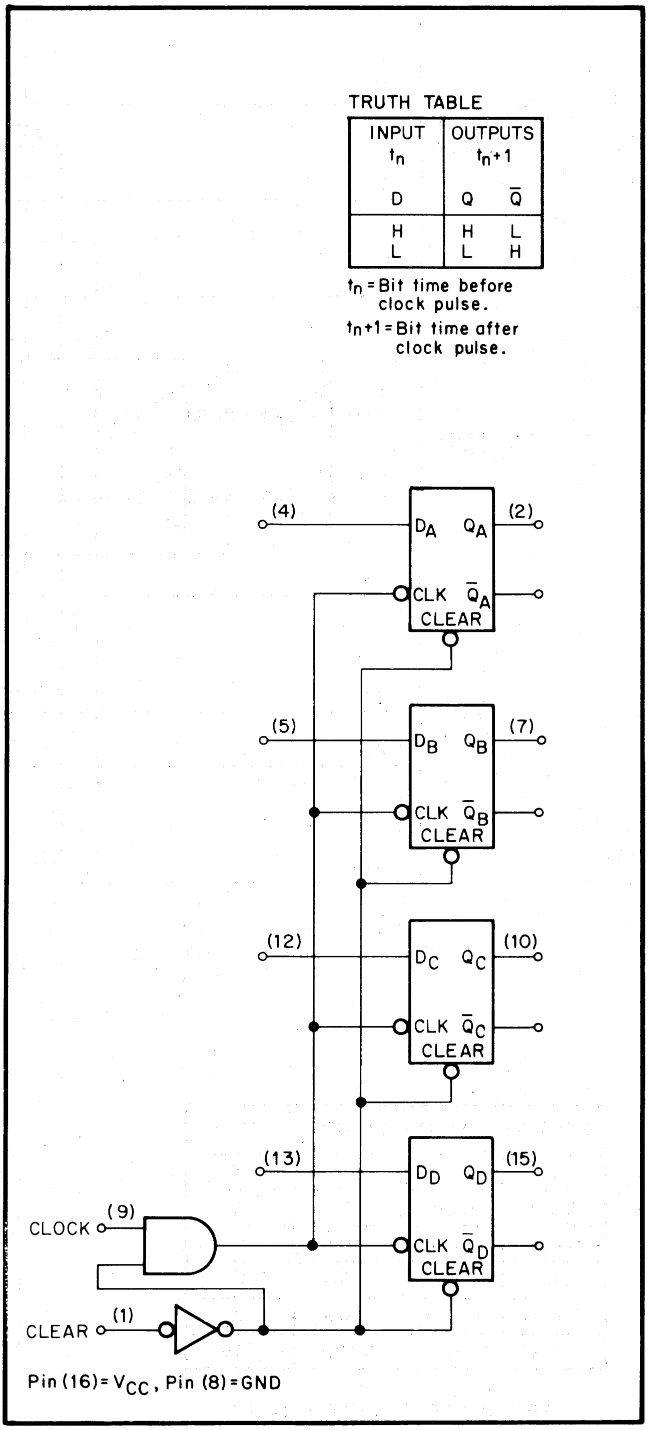
t_n = Bit time before clock pulse.

t_n+1 = Bit time after clock pulse.

Pin (16) = V_{CC}, Pin (8) = GND

11-1112

Figure C-5 74174 Logic Diagram



TRUTH TABLE

INPUT t_n	OUTPUTS t_n+1	
D	Q	$\bar{Q}$
H	H	L
L	L	H

t_n = Bit time before clock pulse.

t_n+1 = Bit time after clock pulse.

Pin (16) = V_{CC}, Pin (8) = GND

11 1113

Figure C-6 74175 Logic Diagram

74H74 D-TYPE EDGE-TRIGGERED FLIP-FLOPS

The 74H74 consists of two D-type edge-triggered flip-flops. Each flip-flop has individual clear and preset inputs and complementary Q and $\overline{Q}$ outputs. Information at input D is transferred to the Q output on the positive-going edge of the clock pulse.

TRUTH TABLE (Each Flip-Flop)

t_n	t_{n+1}	
INPUT D	OUTPUT Q	OUTPUT $\overline{Q}$
L	L	H
H	H	L

H = high level, L = low level

NOTES: A. t_n = bit time before clock pulse.
B. t_{n+1} = bit time after clock pulse.

Signal/Pin Designation

Signal Name	Circuit #1	Circuit #2
D	2	12
CLOCK	3	11
CLEAR	1	13
PRESET	4	10
Q	5	9
$\overline{Q}$	6	8

Functional Block Diagram (each flip-flop)

functional block diagram (each flip-flop)

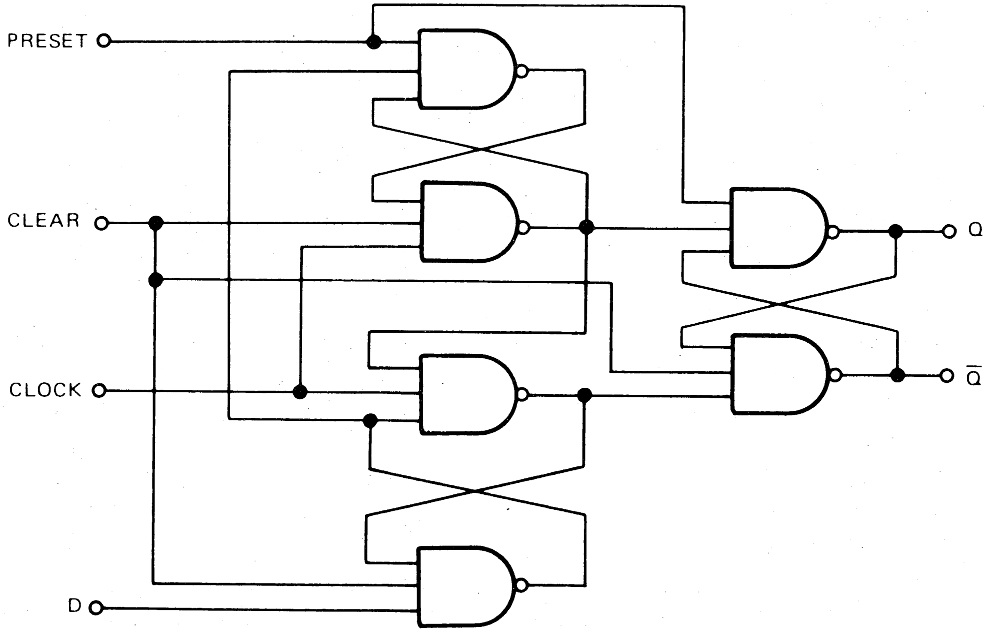


Figure C-7 74H74 Logic Diagram

digital

digital equipment corporation

Printed in U.S.A.