Cromemco®

QUADART

INSTRUCTION MANUAL

CROMEMCO, Inc.
280 Bernardo Avenue
Mountain View, CA 94043

Part No. 023-2005 November 1981

SIO WRITE REGISTERS

(Each Channel)

WR0

D7 D6 D5 D4 D3 D2 D1 D0

D2	D1	D0	
0	0	0	Register 0
0	0	1	Register 1
0	1	0	Register 2
0	1	1	Register 3
1	0	0	Register 4
1	0	1	Register 5
1	1	0	Register 6
1	1	1	Register 7

D5	D4	D3	
0	0	0	Null Code
0	0	1	Send Abort (SDLC)
0	1	0	Reset EXSTAT Interrupts
0	1	1	Channel Reset
1	0	0	Enable Interrupt On Next Received Character
1	0	1	Reset Tx Int Pending
1	1	0	RXSPCL Reset
1	1	1	Return From Int (CH 0 & 2 Only)

D7	D6	
0	0	Null Code
0	1	Reset Rx CRC Checker
1	0	Reset Tx CRC Generator
1	1	Reset Tx Underrun/EOM Latch

WR1

D7 D6 D5 D4 D3 D2 D1 D0

Bit	
D0	EXSTAT Int Enable
D1	TBE Int Enable
D2	Status Affects Vector (CH 1 & 3 Only)

D4	D3	
0	0	RCA And RXSPCL Int Disable
0	1	RCA Int On First Character Only And On RXSPCL
1	0	RCA Int On All Characters And On RXSPCL (Parity Affects Vector)
1	1	RCA Int On All Characters And On RXSPCL (Parity Does Not Affect Vector)

Bit		
D5	X	Wait/Ready Function Not Used On Quadart; Program D7 to 0
D6	X	
D7	0	

WR2 (CH 1 & 3 ONLY)

D7 D6 D5 D4 D3 D2 D1 D0

Bit		
D0	V0	Interrupt Vector
D1	V1	
D2	V2	
D3	V3	
D4	V4	
D5	V5	
D6	V6	
D7	V7	

WR3

D7 D6 D5 D4 D3 D2 D1 D0

Bit	
D0	Rx Enable
D1	Sync Character Load Inhibit
D2	SDLC Address Search Mode
D3	Rx CRC Enable
D4	Enter Hunt Mode
D5	Auto Enables

D7	D6	
0	0	Rx 5 Bits/Character
0	1	Rx 7 Bits/Character
1	0	Rx 6 Bits/Character
1	1	Rx 8 Bits/Character

WR4

D7 D6 D5 D4 D3 D2 D1 D0

Bit	
D0	Parity Enable
D1	Parity Even/Odd

D3	D2	
0	0	Sync Modes Enable
0	1	1 Stop Bit/Character
1	0	1 1/2 Stop Bits/Character
1	1	2 Stop Bits/Character

D5	D4	
0	0	8 Bit Sync Character
0	1	16 Bit Sync Character
1	0	SDLC Mode (01111110 Flag)
1	1	Ext Sync (Not Used)

D7	D6	
0	0	X1 Clock Mode
0	1	X16 Clock Mode
1	0	X32 Clock Mode
1	1	X64 Clock Mode

WR5

D7 D6 D5 D4 D3 D2 D1 D0

Bit	
D0	Tx CRC Enable
D1	RTS
D2	SDLC/CRC-16
D3	Tx Enable
D4	Send Break

D6	D5	
0	0	Tx 5 Bits (Or Less) /Character
0	1	Tx 7 Bits/Character
1	0	Tx 6 Bits/Character
1	1	Tx 8 Bits/Character

Bit	
D7	DTR

WR6

D7 D6 D5 D4 D3 D2 D1 D0

Bit		
D0	SYNC Bit 0	*
D1	SYNC Bit 1	
D2	SYNC Bit 2	
D3	SYNC Bit 3	
D4	SYNC Bit 4	
D5	SYNC Bit 5	
D6	SYNC Bit 6	
D7	SYNC Bit 7	

* Address Field In SDLC Mode

WR7

D7 D6 D5 D4 D3 D2 D1 D0

Bit		
D0	SYNC Bit 8	*
D1	SYNC Bit 9	
D2	SYNC Bit 10	
D3	SYNC Bit 11	
D4	SYNC Bit 12	
D5	SYNC Bit 13	
D6	SYNC Bit 14	
D7	SYNC Bit 15	

* Must Be Programmed To 01111110 Flag in SDLC Mode

2006-33

Cromemco®

QUADART

INSTRUCTION MANUAL

CROMEMCO, Inc.
280 Bernardo Avenue
Mountain View, CA 94043

Part No. 023-2005

November 1981

This manual was produced on a Cromemco System Three computer utilizing a Cromemco HDD-22 Hard Disk Storage System running under the Cromemco Cromix™ Operating System. The text was edited with the Cromemco Cromix Screen Editor. The edited text was proofread using the Cromemco SpellMaster™ Program and then formatted using the Cromemco Word Processing System Formatter II. Final camera-ready copy was printed on a Cromemco 3355A printer.

QUADART TECHNICAL SPECIFICATIONS

Serial Channels:	4 independent channels
Serial Protocols:	- Asynchronous Byte - Synchronous Byte (BiSync) - Synchronous Bit (SDLC) - RS-232C Modem Control
Channel Bit Rates:	Asynchronous, 75 to 19,200 bits/sec with x16 clock; synchronous, 61 to 300,000 bits/sec; rates software selectable
Character Length:	Transmitter, 1 to 8 bits; receiver, 5 to 8 bits
Error Detection:	Parity, CRC-16 or CRC-CCITT generation and checking, receiver framing and overrun, transmitter underrun, break generation and detection
Modem Status Indicators:	CTS, DSR, DCD, DTR, RTS, RI, and CY LED indicators on each channel
Diagnostic Capability:	Channel-to-channel loopback (input/output channels software selectable)
Vectored Interrupts:	65,536 possible restart locations in Z80 IM2 mode; serial channels and timers internally prioritized; multiple Quadarts daisy chain prioritized
Interval Timers:	Four timers, each with a 0-16.384 mSec range; cascadable to 1 Sec intervals; resolution 3.25 uSec (counter mode), 4.00 uSec (timer mode)
LSI Device Types:	Two - Z80 SIO/2 (Serial I/O) One - Z80 PIO (Parallel I/O) Two - Z80 CTC (Counter/Timer Circuit)
Processor Interface:	Cromemco C-Bus to IOP board
Power (from S-100 bus):	+ 8 VDC @ 1.5 A +18 VDC @ 100 mA -18 VDC @ 100 mA
Operating Environment:	0-55 degrees Celsius

Table of Contents

ABOUT THIS MANUAL

Chapter 1 introduces the reader to the Quadart interface board. It tells what the Quadart is, how it is used, and how it should be set up. The chapter also discusses the Quadart's interrupt structure, modem control, and loopback system. This chapter contains many illustrations, diagrams, and tables which are referenced in later chapters.

Each of the four Quadart channels may operate in one of three modes -- **asynchronous, byte synchronous,** or **SDLC.** Chapters 5, 6 and 7 discuss each of these modes separately. Chapters 2, 3 and 4 give preliminary information common to all three modes. Chapter 8 discusses interrupts; Chapter 9 discusses modems. Chapter 10 describes the Quadart loopback system, which is useful for diagnostics or self tests; and Chapter 11 describes how the Quadart timers operate.

The reader of this manual should be familiar with Z80 Assembly Language, and should have read Cromemco's Input/Output Processor Instruction Manual, part number 023-2006. Reference will also be made to Zilog or Mostek Technical Manuals included in the Quadart documentation package. Also, the reader will find the following list of references helpful: Mc Namara, J. E., Technical Aspects of Data Communication, Digital Equipment Corporation, 1977; General Information -- Binary Synchronous Communications (BISYNC specifications), IBM, GA27-3004-2 File TP-09; General Information -- Synchronous Data Link Control (SDLC specifications), IBM, GA27-3093 File GENL-09.

Certain conventions will be consistently used throughout this manual:

1. Positive logic is assumed.

2. **Set** means logic 1, **reset** means logic 0 as these terms apply to bit states.

3. The three SIO modes will be called **asynchronous mode, byte synchronous mode,** and **SDLC mode.**

4. External/Status is abbreviated to **EXSTAT.**

5. Special Receive is abbreviated to **RXSPCL.**

Chapter 1

INTRODUCTION AND MANUAL PREVIEW

This chapter gives the reader a broad overview of Quadart applications. Most of the topics lightly touched upon here are detailed fully in later chapters.

QUADART FEATURES

- FOUR INDEPENDENT FULL-DUPLEX CHANNELS
- SYNCHRONOUS BYTE (BISYNC) FEATURES:
  - 5 to 8 bits/character
  - Programmable sync characters
  - Transparent text mode operation
  - Automatic sync insertion during idle
  - Hardware CRC generation and detection
  - CRC-16 or CRC-CCITT polynomials
  - 61 to 300,000 baud
- ASYNCHRONOUS FEATURES:
  - 5 to 8 bits/character
  - 1 1-1/2 or 2 stop bits
  - Break generation and detection
  - Parity overrun and framing error detection
  - Even ODO or no parity
  - 75 to 19,200 baud with X16 clock
- MODEM CONTROLS AND INDICATORS:
  - CTS, DSR, DCD and RI LINES usable for modem control or user defined input
  - DTR, RTS and CY LINES usable for modem control or user defined output
  - Led indicators for CTS, DSR, DCD, DTR, RTS RI and CY on each channel
- SYNCHRONOUS BIT (SDLC AND HDLC) FEATURES:
  - 1 to 8 bits/transmitter character
  - 5 to 8 bits/receiver character
  - Hardware address recognition
  - Automatic zero insert delete
  - 1-field residue handling
  - Automatic flag insertion between messages
  - Hardware CRC-16 generation and detection
  - 61 to 300,000 baud
- INTERRUPT SYSTEM FEATURES:
  - Channel functions and timers internally prioritized
  - Channel functions and timers generate unique Z80 interrupt mode 2 vectors
  - Multiple quadarts daisy-chain prioritized
- SOFTWARE CONTROLLED CHANNEL-TO-CHANNEL LOOPBACK FOR SELF TEST AND DIAGNOSTICS
- FOUR CRYSTAL CONTROLLED GENERAL PURPOSE TIMERS

1.1 QUADART BLOCK DIAGRAM

The Cromemco Quadart is a four channel serial I/O interface board which connects up to four modems/terminals and to a Cromemco IOP (Input/Output Processor) board. A program running on the IOP board configures the Quadart channels and performs parallel I/O with the Quadart in response to either polled status or Quadart generated interrupts. The Quadart functions primarily as a four channel USART which formats and buffers I/O data, provides channel communications and error status, and performs serial to parallel and parallel to serial data conversions, modem control and status monitoring. In addition to these basic functions, the Quadart also offloads the IOP processor by providing a variety of useful hardware features like CRC generation and checking, SDLC address recognition logic, programmable sync characters (8-bit or 16-bit), status variable interrupt vectors, and modem controlled receiver and transmitter enabling and disabling. The Quadart interfaces to the terminal/modems with RS-232C drivers and receivers at connectors located along the top edge of the board. Both Data Terminal Equipment (DTE) and Data Communication Equipment (DCE) connectors, plus LED modem status indicators, are provided for each channel.

Up to four Quadarts can be connected to a single IOP. The number of IOP/Quadart sets is limited only by the number of available IOP ports. The Quadart is connected to the IOP by an overhead independent 50-conductor expansion bus called a Cromemco- or **C-Bus.** This bus physically consists of a ribbon cable and connector assembly which mates an IOP to one or more Quadarts which it controls at connectors located along the top edge of each board. The IOP and Quadart each occupy a single S-100 bus slot of the **host system** (see Figure 1). The Quadart only draws power from the host S-100 bus, while the IOP draws power and optionally communicates with the **host processor** over the S-100 bus. A block diagram of the Quadart interface is shown in Figure 2. The four Quadart channels are referred to as **channel 0** through **channel 3** throughout the manual. The Quadart functions which apply to each Quadart channel independently of the other three are shown stacked four deep in the block diagram. Virtually all major Quadart functions, except loopback, center around five Z80 LSI devices (see Figure 3). Two Z80 SIO/2 (Serial Input/Output, option 2) devices form the heart of the Quadart interface. Each SIO controls two channels by providing serial data formatting, received data and error FIFO (buffering) control, SDLC zero insert/delete, CRC generation and checking, sync character transmission and detection, modem control, and standard software status signals (e.g., Transmitter Buffer Empty, Receiver Character Available).

Two Z80 CTCs (Counter/Timer Circuit) contain four counters apiece. Four of these counters are used as local TxC and RxC clocks primarily in asynchronous mode. The other four counters are available for general purpose use: like providing a real time clock for data logging applications and generating various time out intervals required by different serial protocols. These general purpose counter/timers will be referred to as **timer A** through **timer D.**

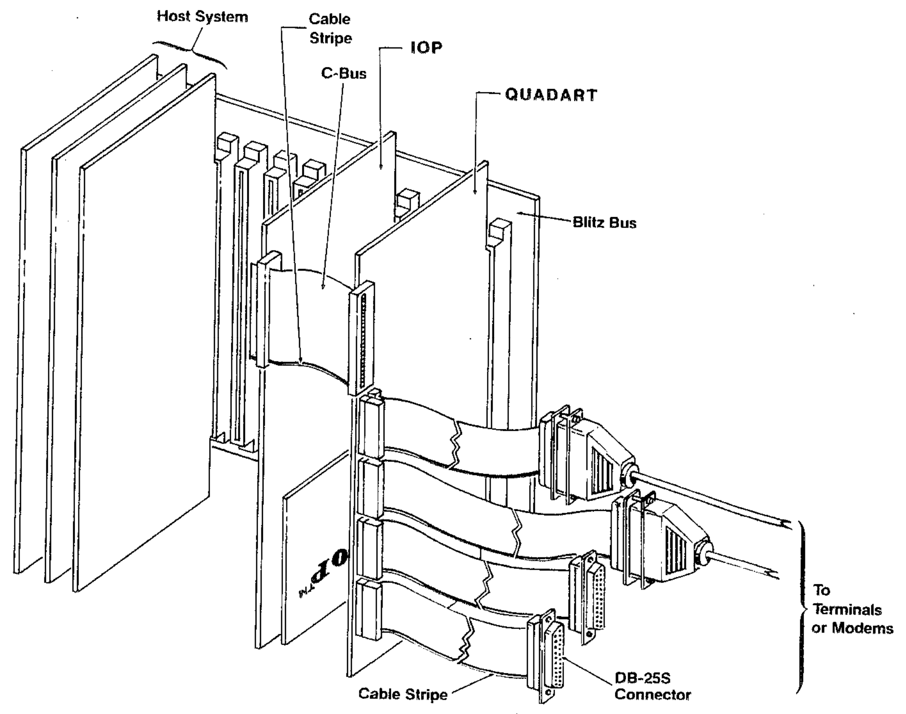


Figure 1: IOP/QUADART IN AN S-100 BUS SYSTEM

Finally, for switched line applications a single Z80 PIO (Parallel Input/Output) device provides the user with RI (Ring Indicator) and DSR (Data Set Ready) modem input lines on each channel. The PIO also supports a user defined modem output line **CY** on each channel (these lines might be used to control an auto dialer, for example), and four clock source control bits, **ExtCk0** through **ExtCk3**, which select between local and modem supplied $\overline{TxC}$ and $\overline{RxC}$ clock inputs to the SIOs.

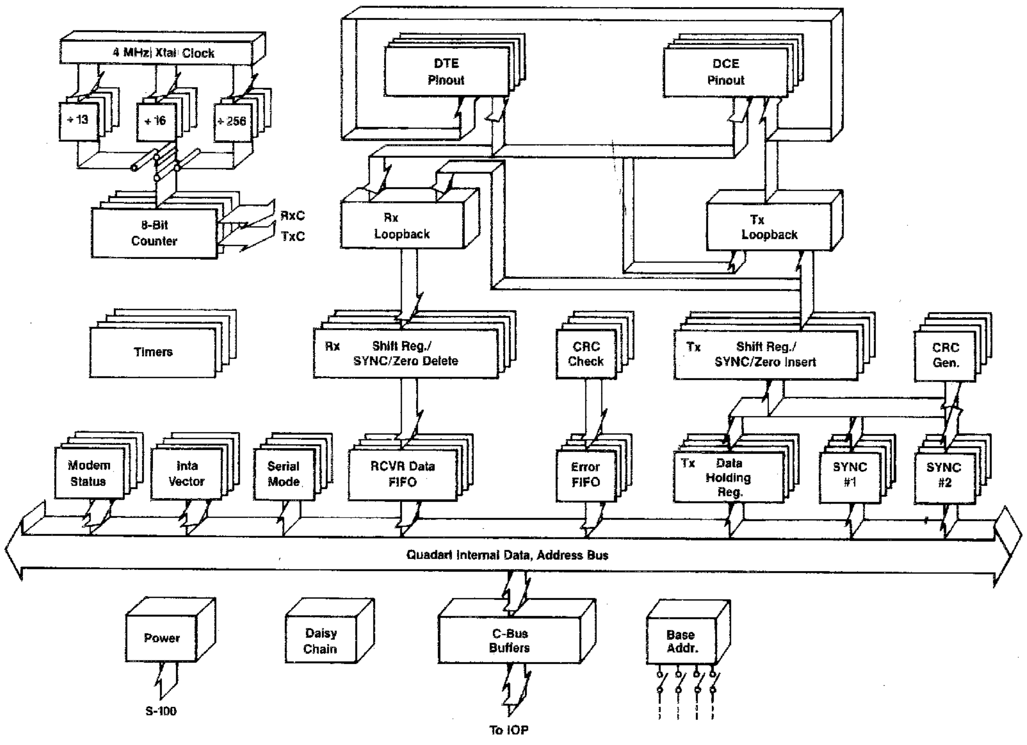


Figure 2: **QUADART BLOCK DIAGRAM**

All three Z80 LSI devices (SIO, CTC, PIO) may be programmed to generate unique interrupt vectors to the IOP; these interrupts are organized in a fixed priority scheme on the Quadart (see Figure 14), and each interrupt source may be independently enabled or disabled from generating interrupts.

The unique Quadart **loopback** system shown in the block diagram lets the interface be diagnosed and self-tested. By outputting a single byte to Quadart register CNTRL, one of eight data sources (four TxD SIO outputs and four incoming RxD modem lines) may be routed to one of eight data sinks (four outgoing $\overline{\text{TxD}}$ modem lines and four $\overline{\text{RxD}}$ SIO inputs).

1.2 SERIAL CHANNELS AND DATA FORMATS

The two SIOs control four completely independent serial I/O channels; each channel may differ in mode, bit rate, character length, parity, sync characters, interrupts armed, and so on. The channels are configured by the IOP program as part of a power up initialization procedure by writing control bytes to the SIOs. After resetting a channel, the first control byte written must define the channel mode -- **asynchronous, byte synchronous,** or **SDLC.** In the Zilog and Mostek documentation, the byte synchronous mode is termed **Monosync** if an 8-bit sync character is selected. It is termed **Bisync** (not to be confused with IBM BiSync, which is a protocol, not a mode) if a 16-bit sync character is selected. These two options will collectively be termed **byte synchronous mode** throughout this manual. The remainder of this section briefly describes each of these modes and their corresponding data formats.

Asynchronous Mode

Characters are transmitted and received as isolated **information packets** in the asynchronous mode with an undefined interval of time separating the packets. The data line idles in a marking state while no data is being transmitted. The transmitter and receiver clock rates, which define bit length intervals, need only be approximately equal since character synchronization is reestablished by the receiver on the start bit of each new character. This operating mode is commonly selected when communicating with a keyboard device where the intervals between operator keystrokes are highly variable.

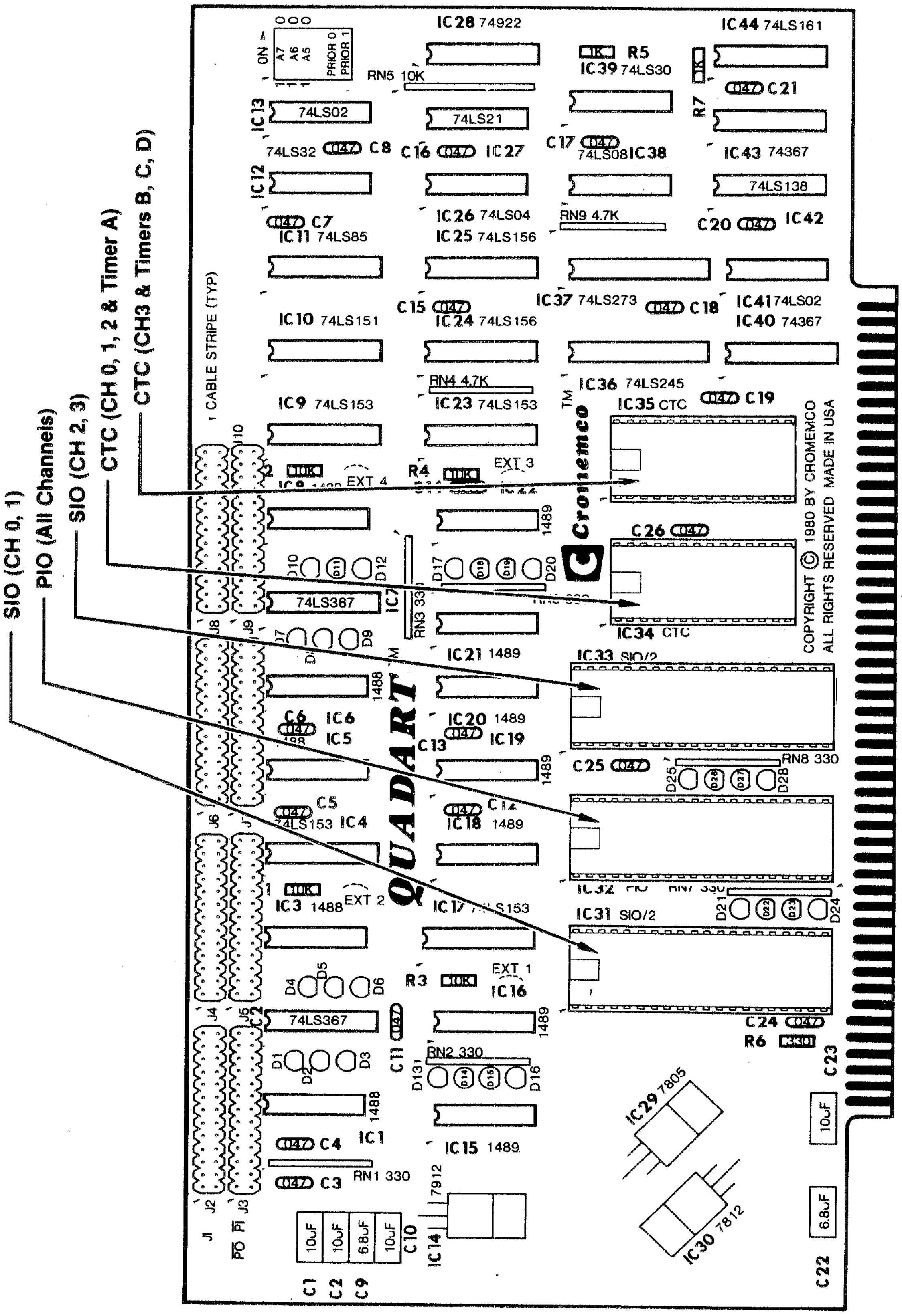

SIO (CH 0, 1)
PIO (All Channels)
SIO (CH 2, 3)
CTC (CH 0, 1, 2 & Timer A)
CTC (CH3 & Timers B, C, D)
IC35 CTC
IC34 CTC
IC33 SIO/2
IC32 PIO
IC31 SIO/2
QUADART
Cromemco
COPYRIGHT © 1980 BY CROMEMCO
ALL RIGHTS RESERVED MADE IN USA

Figure 3: QUADART LSI DEVICES

The SIO Asynchronous Mode data format is shown in Figure 4. Each character is framed by 1 spacing start bit and either 1, 1-1/2 or 2 stop bits. The data bits are transmitted and received sequentially starting with the character LSB, and the number of character bits may vary from 1 to 8 for transmitted data, and from 5 to 8 for received data, although the transmitted and received character lengths are typically made to match. In addition to the data bits, an optional even or odd parity bit may be appended to the end of the character for error detection. Thus the longest possible transmitter/receiver character is 12 bits long -- 1 start bit, 8 data bits, 1 parity bit, and 2 stop bits.

Note the signal level differences between the SIO $\overline{\text{TxD}}$ and $\overline{\text{RxD}}$ pins and those at the RS-232C interface circuitry. At the active low SIO pins, a marking level corresponds to a TTL high (greater that +2.4 volts) and a spacing level to a TTL low (less than +0.4 volts); whereas an RS-232C marking level corresponds to a voltage less than -5 volts, and a spacing level to voltage greater than +5 volts. This same definition of spacing and marking levels applies to all SIO operating modes (synchronous as well as asynchronous). Refer to Chapter 5 for further details on the asynchronous mode.

Byte Synchronous Mode

In the byte synchronous mode, a block consisting of an integral number of characters is typically transmitted and received as a unit called a **message.** The message data bits are transmitted and received in synchronization with a common clock signal. The Quadart TxC clock is typically supplied by an attached modem in the byte synchronous mode. The Quadart RxC clock is extracted from the RxD signal by the modem, and is then fed to the Quadart RxC input. Since there are no framing bits to delimit character boundaries in synchronous modes, characters are synchronized by recognizing unique **sync** character bit patterns. The Quadart receivers are synchronized by recognizing one sync character which enables the receiver to begin assembling and loading characters for reading by the IOP. The sync character length (8-bits or 16-bits), and its value, are software defined. The byte synchronous message format is shown in Figure 5.

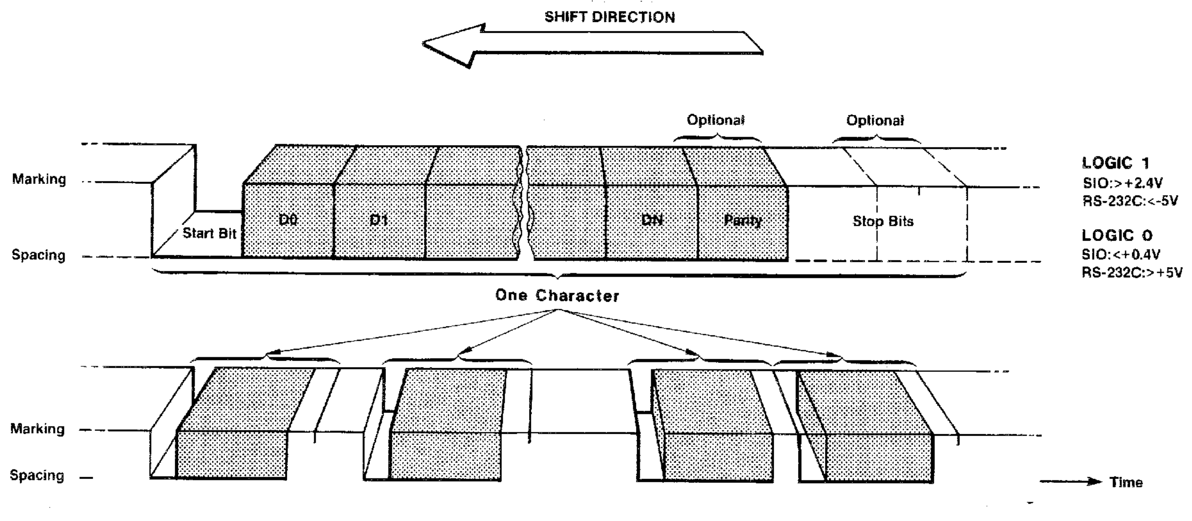


Figure 4: ASYNCHRONOUS DATA FORMAT

The byte synchronous message has three major components: one or more leading sync characters for character synchronization, a data field consisting of an integral number of characters, and an optional 16-bit CRC block check character. The characters of the data field are transmitted LSB first and an optional parity bit may be included in each character. The SIO permits transmitter character lengths of 1 to 8 bits, and receiver character lengths of 5 to 8 bits (add 1 bit to both with parity); typically both lengths are made to match. Since SIO character lengths are adjustable, the popular 6-bit Transcode, 7-bit ASCII, and 8-bit EBCDIC codes (with or without parity) may easily be implemented, as well as custom codes (see Appendices B, C and D). A CRC block check character may be appended to the data before the TxD line automatically idles by continuously sending sync characters after the transmitter underruns at the end of a message. When CRC is sent (either a CRC-CCITT or CRC-16 polynomial may be selected), the receiving channel must be forewarned by a control character imbedded in the message (e.g., an ETX or ETB in IBM BiSync protocol). Again notice the difference in levels at the SIO pins and at the RS-232C interface. Refer to Chapter 6 for further details on the byte synchronous mode.

SDLC (HDLC) Mode

In the SDLC mode a block of bits, termed a **frame**, is transmitted and received as a unit in synchronization with a common clock signal. As in the byte synchronous mode, the $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clock signals feed the SIOs and are usually supplied by an attached modem. The frame data format is shown in Figure 6. The SDLC frame has five major components: one or more opening 01111110b flags (sync characters) which enable the receiver to determine where the address field begins, an 8-bit address field, an I-field consisting of zero or more bits, an optional 16-bit CRC block check character, and a closing 01111110b flag.

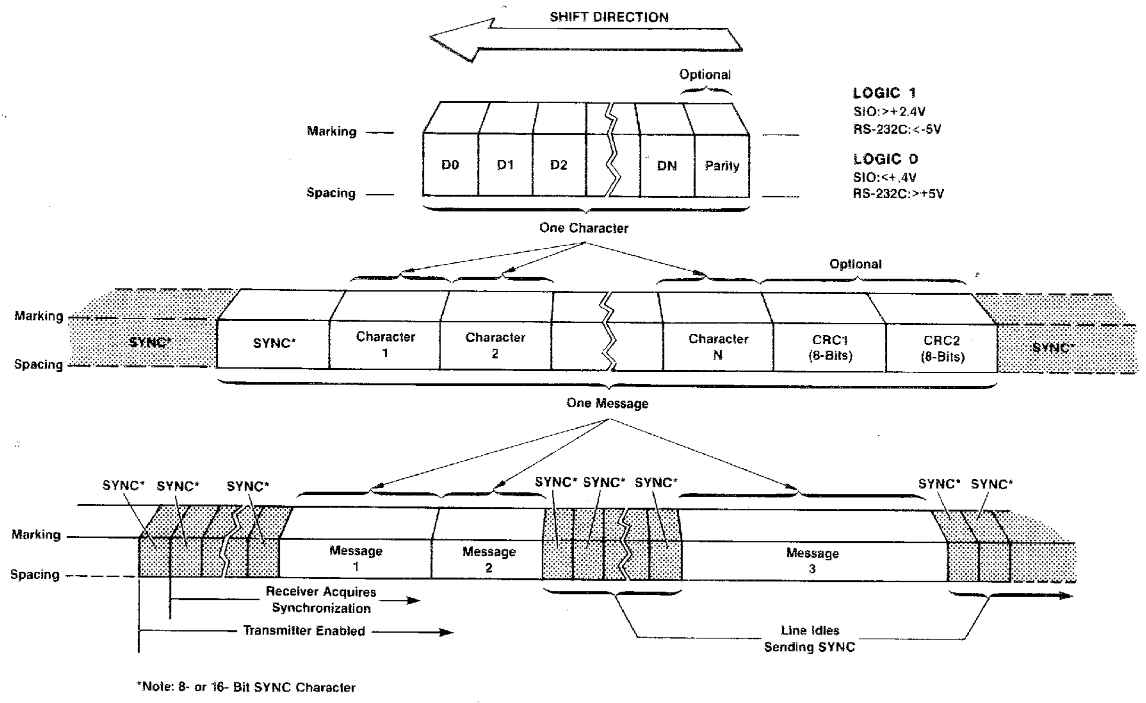


Figure 5: BYTE SYNCHRONOUS MESSAGE FORMAT

The 8-bit SDLC control field required by the SDLC protocol is sent and received as ordinary I-Field data. A channel's receiver may be programmed to only accept frames matching a software selectable address (or the 11111111b global address), and otherwise to remain in the hunt mode. The address field and I-field data are loaded into the SIO transmitter as 1- to 8-bit characters (LSB transmitted first). The transmitter character length may be changed between character transmissions to match the conveniently available data length. The receiver character length may vary from 5 to 8 bits. In contrast to the byte synchronous mode, the transmitter and receiver character lengths are not necessarily made to match since the data field is bit rather than character oriented.

The SDLC/HDLC protocol requires that the receiver recognize the 01111110b flag at any time. Thus a single binary zero must be inserted after five contiguous ones in the address byte, I-field data, and CRC character taken together as a single serial bit stream. The SIO transmitter automatically inserts all such required zeros as each character is shifted out. The SIO receiver automatically removes them as each character is shifted in, making the zero insert/delete function transparent to both ends of the link.

1.3 QUADART ADDRESSING

The Quadart is controlled by, and exchanges data with the IOP over the C-Bus. The Quadart communicates with the IOP by interrupt requests, acknowledgments and vectors placed on the C-Bus, but for the most part, commands and data are routed through 21 I/O ports on the Quadart under IOP program control. The 21 Quadart I/O ports names and addresses are listed in Table 1. Of the 21 ports, 18 are bidirectional, and all but 1 (port CNTRL) access internal SIO, CTC or PIO registers.

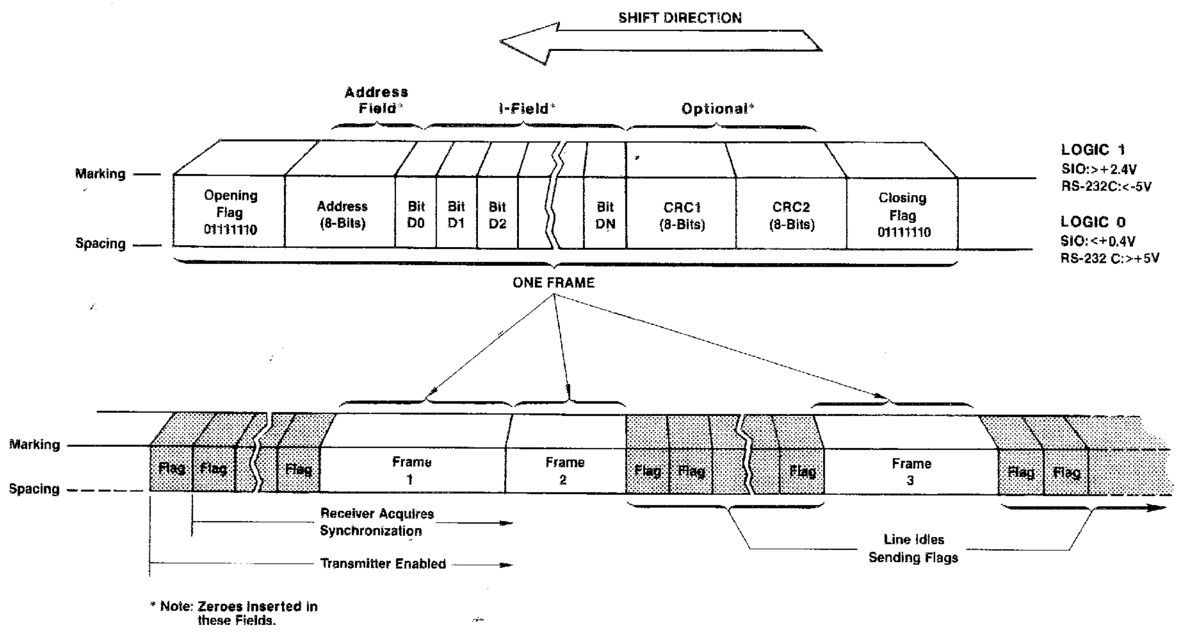
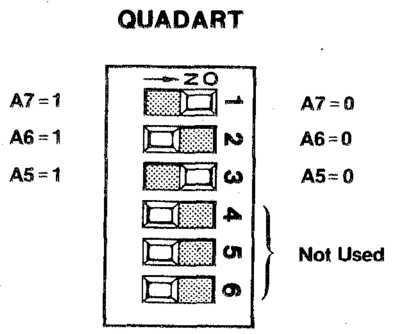


Figure 6: SDLC FRAME DATA FORMAT

Table 1: QUADART INPUT/OUTPUT PORTS

I/O PORT ADDRESS	DEVICE/ REGISTER	QUADART CHANNEL	DIRECTION
Qbase + 00h	SIO Data A	Ch 0	In/Out
Qbase + 01h	SIO Command/Status A	Ch 0	In/Out
Qbase + 02h	SIO Data B	Ch 1	In/Out
Qbase + 03h	SIO Command/Status B	Ch 1	In/Out
Qbase + 04h	SIO Data A	Ch 2	In/Out
Qbase + 05h	SIO Command/Status A	Ch 2	In/Out
Qbase + 06h	SIO Data B	Ch 3	In/Out
Qbase + 07h	SIO Command/Status B	Ch 3	In/Out
Qbase + 08h	PIO Data A	Ch 0 & 1	In/Out
Qbase + 09h	PIO Control A	Ch 0 & 1	Out
Qbase + 0Ah	PIO Data B	Ch 2 & 3	In/Out
Qbase + 0Bh	PIO Control B	Ch 2 & 3	Out
Qbase + 0Ch	CTC CS0	Ch 0	In/Out
Qbase + 0Dh	CTC CS1	Ch 1	In/Out
Qbase + 0Eh	CTC CS2	Ch 2	In/Out
Qbase + 0Fh	CTC CS3	Timer A	In/Out
Qbase + 10h	CTC CS0	Ch 3	In/Out
Qbase + 11h	CTC CS1	Timer B	In/Out
Qbase + 12h	CTC CS2	Timer C	In/Out
Qbase + 13h	CTC CS3	Timer D	In/Out
Qbase + 14h	CNTRL	All	Out

The Quadart I/O ports may be mapped into one of eight different regions of the IOP I/O space with the Quadart base address switch, located in the upper right-hand corner of the board. The base address switch selects the Quadart **base address** as illustrated below in Figure 7, and all 21 ports then maintain a fixed offset relative to the base address (Qbase). If, for example, all slide switches are thrown to the left in the figure, Qbase becomes **E0h**. The IOP program would then read data from the channel 2 SIO Data A register into the IOP's Z80A accumulator by executing an **IN A,(E4h)** instruction, and it would output a byte of data from the accumulator to port CNTRL by executing an **OUT (F4h),A** instruction. For consistency, a Qbase of 40h (the switch setting shown in the figure) will be assumed throughout the manual in all programming examples.



Switch Position			Quadart
A7	A6	A5	Base Address
0	0	0	00h
0	0	1	20h
0	1	0	40h
0	1	1	60h
1	0	0	80h
1	0	1	A0h
1	1	0	C0h
1	1	1	E0h

Figure 7: QUADART BASE ADDRESS SWITCH

1.4 QUADART PIN OUTS, CABLES AND CONNECTORS

There are 10 connectors in addition to the S-100 edge connector on the Quadart board. The pin numbers and their corresponding labels are called out in Table 2 below. The connector numbers (J1 through J10) are clearly labeled on the board component side. All **ground** references imply a connection to S-100 bus ground (0 volt) potential. The reader is also referred to the Quadart Schematic Diagram near the end of the manual.

Cromemco supplies two cable assemblies for interfacing DTE and DCE equipment to Quadart connectors J2 through J9. Each of these assemblies has a 25-pin female DB-25S connector, a 25-conductor ribbon cable, and a 26-pin female connector which mates to J2 through J9. Use Cromemco part number 519-0017 when ordering a 62 cm cable, and 519-0018 for a 110 cm cable. The cables must be installed with the **cable stripe** aligned as shown in Figure 1 to realize the Table 2 pin out at the DB-25S connector. Also refer to Figure 8 for physical J2 through J9 pin numbering.

Cromemco also supplies two C-Bus cable number assemblies. Use part number 519-0100 when ordering a two connector C-Bus cable (an IOP and one peripheral), and part number 519-0101 for a five connector cable (an IOP and up to four peripherals).

Table 2: QUADART CONNECTOR PIN OUTS

Connector J1

Pin 1	PRIORITY OUT
Pin 2	PRIORITY IN

	Connector J2 (DTE)	**Connector J4** (DTE)	**Connector J6** (DTE)	**Connector J8** (DTE)
Pin 2	RxD Ch 0	RxD Ch 1	RxD Ch 2	RxD Ch 3
Pin 3	TxD Ch 0	TxD Ch 1	TxD Ch 2	TxD Ch 3
Pin 4	CTS Ch 0	CTS Ch 1	CTS Ch 2	CTS Ch 3
Pin 5	RTS Ch 0	RTS Ch 1	RTS Ch 2	RTS Ch 3
Pin 6	DCD Ch 0	DCD Ch 1	DCD Ch 2	DCD Ch 3
Pin 7	Ground	Ground	Ground	Ground
Pin 8	DTR Ch 0	DTR Ch 1	DTR Ch 2	DTR Ch 3
Pin 20	DCD Ch 0	DCD Ch 1	DCD Ch 2	DCD Ch 3

	Connector J3 (DCE)	**Connector J5** (DCE)	**Connector J7** (DCE)	**Connector J9** (DCE)	
Pin 2	TxD Ch 0	TxD Ch 1	TxD Ch 2	TxD Ch 3	Y – Grn
Pin 3	RxD Ch 0	RxD Ch 1	RxD Ch 2	RxD Ch 3	Br – Pur
Pin 4	RTS Ch 0	RTS Ch 1	RTS Ch 2	RTS Ch 3	[illegible]
Pin 5	CTS Ch 0	CTS Ch 1	CTS Ch 2	CTS Ch 3	[illegible]
Pin 6	DSR Ch 0	DSR Ch 1	DSR Ch 2	DSR Ch 3	[illegible]
Pin 7	Ground	Ground	Ground	Ground	[illegible]
Pin 8	DCD Ch 0	DCD Ch 1	DCD Ch 2	DCD Ch 3	[illegible]
Pin 11	CY Ch 0	CY Ch 1	CY Ch 2	CY Ch 3	[illegible]
Pin 15	TxC Ch 0	TxC Ch 1	TxC Ch 2	TxC Ch 3	[illegible]
Pin 17	RxC Ch 0	RxC Ch 1	RxC Ch 2	RxC Ch 3	[illegible]
Pin 20	DTR Ch 0	DTR Ch 1	DTR Ch 2	DTR Ch 3	[illegible]
Pin 22	RI Ch 0	RI Ch 1	RI Ch 2	RI Ch 3	[illegible]
Pin 24	ExtCk Ch 0	ExtCk Ch 1	ExtCk Ch 2	ExtCk Ch 3	[illegible]

Table 2: QUADART CONNECTOR PIN OUTS (continued)

Connector J10

Pin 1	Ground	Pin 26	A10
Pin 2	RESET	Pin 27	A11
Pin 3	CLOCK	Pin 28	Ground
Pin 4	Ground	Pin 29	A12
Pin 5	WAIT	Pin 30	A13
Pin 6	D0	Pin 31	A14
Pin 7	D1	Pin 32	A15
Pin 8	D2	Pin 33	RD
Pin 9	D3	Pin 34	Ground
Pin 10	D4	Pin 35	WR
Pin 11	D5	Pin 36	M1
Pin 12	D6	Pin 37	MREQ
Pin 13	D7	Pin 38	IORQ
Pin 14	Ground	Pin 39	RFSH
Pin 15	A0	Pin 40	CPU DISCONNECT
Pin 16	A1	Pin 41	BUS AVAILABLE
Pin 17	A2	Pin 42	HALT
Pin 18	A3	Pin 43	Ground
Pin 19	A4	Pin 44	INT
Pin 20	A5	Pin 45	NMI
Pin 21	Ground	Pin 46	no connection
Pin 22	A6	Pin 47	no connection
Pin 23	A7	Pin 48	3rd PRIORITY IN/OUT
Pin 24	A8	Pin 49	no connection
Pin 25	A9	Pin 50	Ground

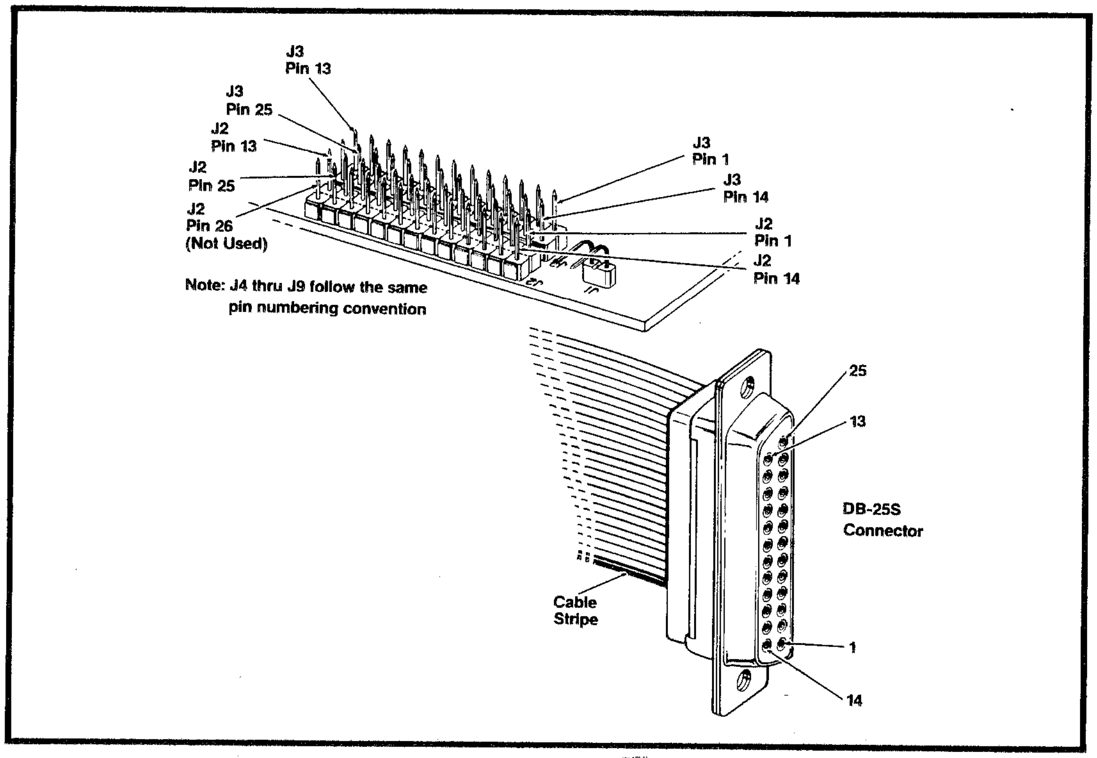


Figure 8: J2 THRU J9 PIN NUMBERS

1.5 QUADART MODEM CONTROL

The Quadart supplies a complete set of RS-232C compatible handshake lines for switched service full duplex applications. The RS-232C interface for one of the four channels is shown in Figure 9. Note that the RxC and TxC clock input lines feed multiplexers which may route these signals, or local onboard CTC bit rate clock sources, to the SIO $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ input pins. Lines DTR, DCD, CTS and RTS, are SIO controlled. DCD and CTS may be programmed to serve as auto enables to the SIO receiver and transmitter, respectively, and all four lines may be used for general purpose I/O (e.g., for controlling an auto dialer). Lines RI, CY and DSR are controlled by the PIO for switched service applications or general purpose I/O. The RxD and TxD data lines are typically connected to their corresponding SIO $\overline{\text{RxD}}$ and $\overline{\text{TxD}}$ pins, but they may optionally be rerouted by the Quadart loopback circuitry. Finally note that lines RI, DTR, CY, DCD, DSR, CTS and RTS are monitored by LED indicators (28 indicators total on the board).

The 1488 line driver is used for all outbound Quadart signals. It converts 0 VDC and +4 VDC TTL input levels to -10 VDC (marking) and +10 VDC (spacing) RS-232C output levels, respectively. The 1489 line receiver is used for all inbound signals, and it converts <-5 VDC and >+5 VDC input levels to +4 VDC and 0 VDC output levels, respectively, as shown in Figure 10.

1.6 IOP/QUADART STAND ALONE SYSTEMS

The simplest complete Quadart system consists of a small, powered S-100 bus (e.g., a Cromemco Model CC-8 eight slot card cage with a Model PS-8 power supply), a single Quadart, a single IOP, and up to four modems or terminals connected to the Quadart. This basic system is illustrated physically in Figure 11, and in block diagram form in Figure 12. A dedicated system of this type could perform a variety of useful tasks including protocol conversions, data concentration, data monitoring and line multiplexing. As more serial channels are added to the system, a single IOP would still control the system, but another Quadart (with a different base address) would be added to manage each new group of four channels.

A program running in IOP memory controls a Quadart by signals passed back and forth over the 50-conductor C-Bus which interconnects them. A typical Quadart application would employ interrupt driven I/O with the Quadart initiating data/command transfers with interrupt requests to the IOP, followed by an interrupt vector passed over the C-bus. The IOP would then respond by either inputting data or status, or outputting data or commands through one or more of the Quadart's 21 I/O ports.

Quadart data transfers are usually interrupt driven for two reasons. First, multichannel, high speed serial communication systems require fast processor response times. Second, all Quadart LSI devices may be easily programmed to supply unique Z80 Interrupt Mode 2 (IM2) vectors in response to an interrupt acknowledge from the IOP. Consequently, the typical Quadart control program in IOP ROM memory has a Quadart initialization segment, a continuously running background program, and several IM2 service routines (which respond to conditions such as TBE, RCA, modem line transitions, and so on for each channel). The 16 Kbytes of IOP RAM memory would be used for data buffering, RAM variables and the stack.

It should be noted that when using a single channel, or few low speed channels, interrupts may be avoided altogether; all the information in the interrupt vectors may be gleaned by polling.

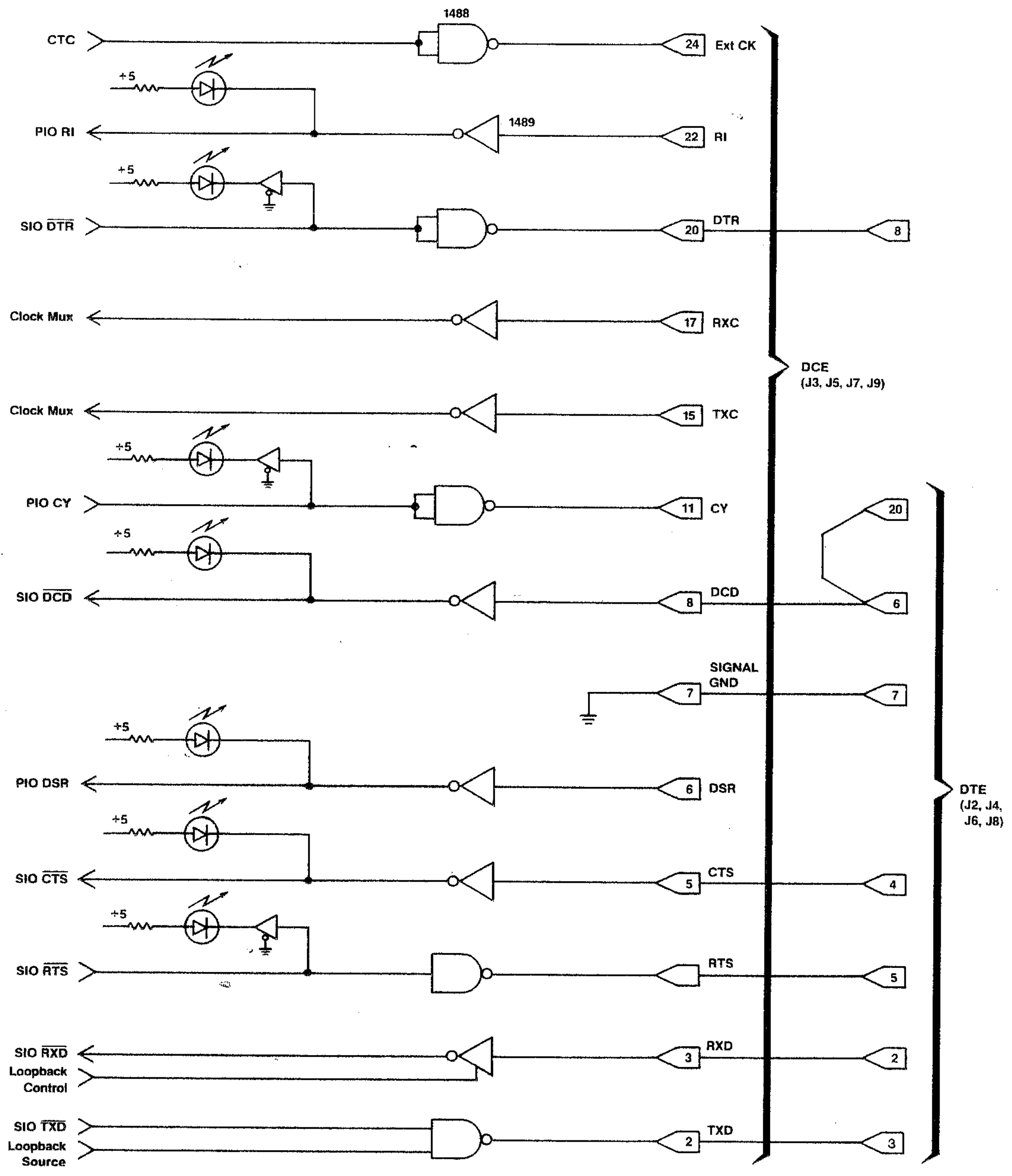


Figure 9: QUADART MODEM INTERFACE (EACH CHANNEL)

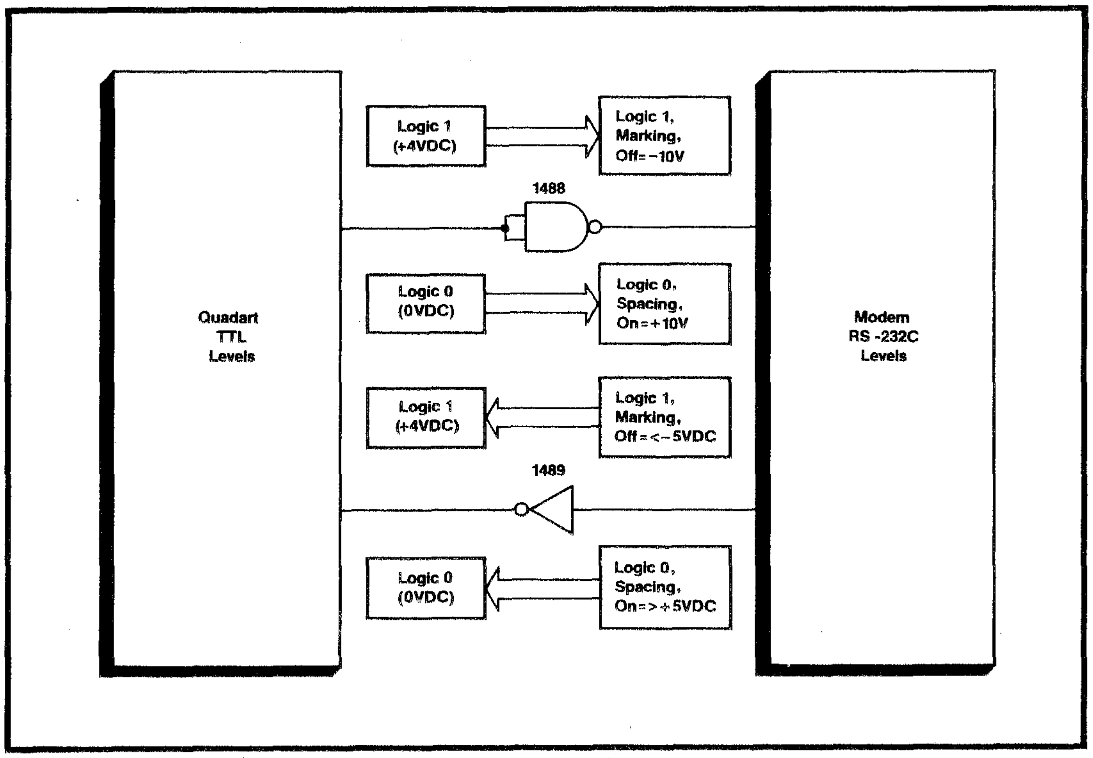


Figure 10: QUADART RS-232C LINE DRIVERS/RECEIVERS

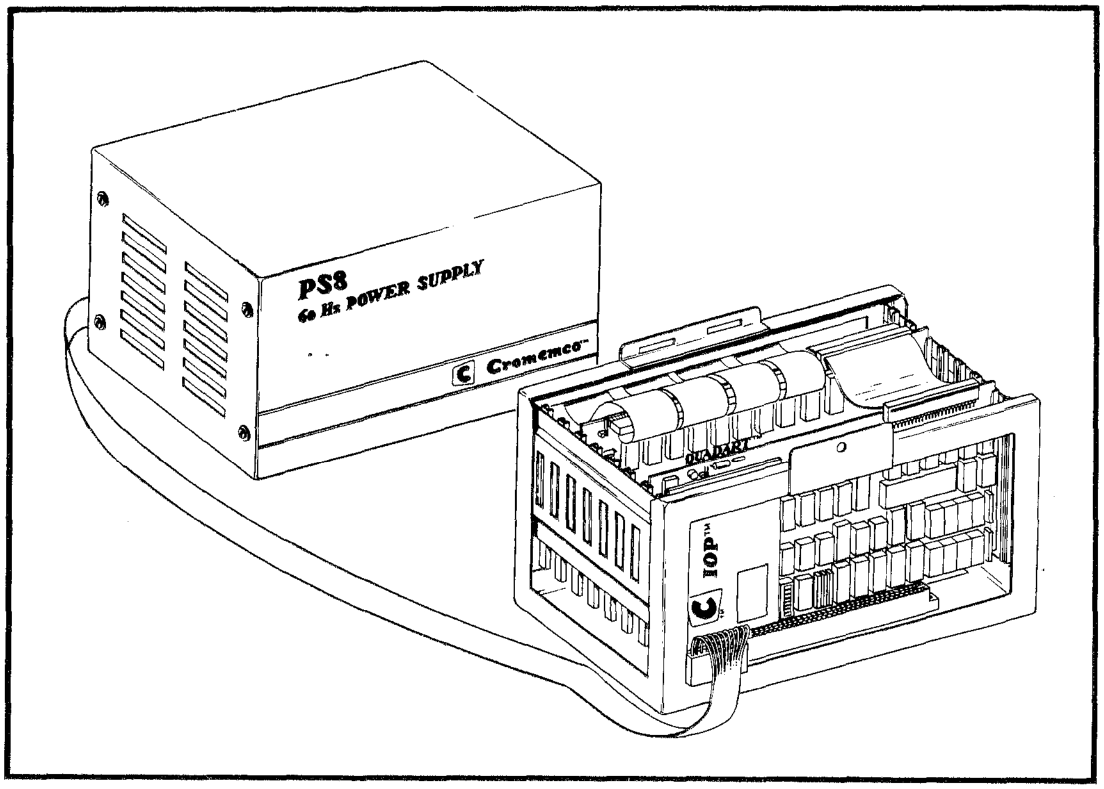


Figure 11: IOP/QUADART STAND ALONE SYSTEM

1.7 IOP/QUADART IN HOST SYSTEMS

The advantages offered by distributed processing and independent expansion buses are fully realized when the IOP and one or more Quadarts act as a serial I/O subsystem for a host system. With this architecture, serial I/O is just one of several **system tasks** with each task being managed by its own dedicated processor and bus. Cromemco's **functional multiprocessing** architecture with distributed processors **and** independent expansion buses differs significantly from systems using multiple processors which compete with one another for use of a shared bus. In these systems, the potential gains offered by several processors are lost by the fixed bandwidth of the bus. In C-Bus systems, however, the host bus is used only for passing task commands and preprocessed data, while all system raw data is handled in a parallel fashion by multiple C-Buses.

The host processor communicates with each task managing IOP through two integral IOP input/output ports. The base address of these two ports is switch selectable to accommodate several IOPs in the same host system. The I/O ports are named from the host's viewpoint; the host processor outputs bytes to the IOP through the **data port** and **command port,** and the host inputs bytes from the IOP through the **data port** and **status port.** Many of the command and status bits are user defined, permitting the programmer to tailor the bit structure to fit each task efficiently. This arrangement is shown in Figure 13.

Any IOP in a host system may be programmed to issue a maskable interrupt request to the host processor over the S-100 bus. This topic is introduced in the next section, and is discussed fully in Chapter 8.

1.8 IOP/QUADART/HOST INTERRUPT STRUCTURE

The Quadart and IOP offer the user tremendous flexibility in designing a system interrupt structure -- from very elaborate schemes to no interrupts whatsoever (a polled system). The range of interrupt configuration options can be best understood by viewing the various Quadart interrupt sources and priorities at four levels: at the Quadart **intra-device** level, the Quadart **inter-device** level, the Quadart **board** level, and the IOP/Quadart subsystem **task** level. This structure is illustrated in Figure 14.

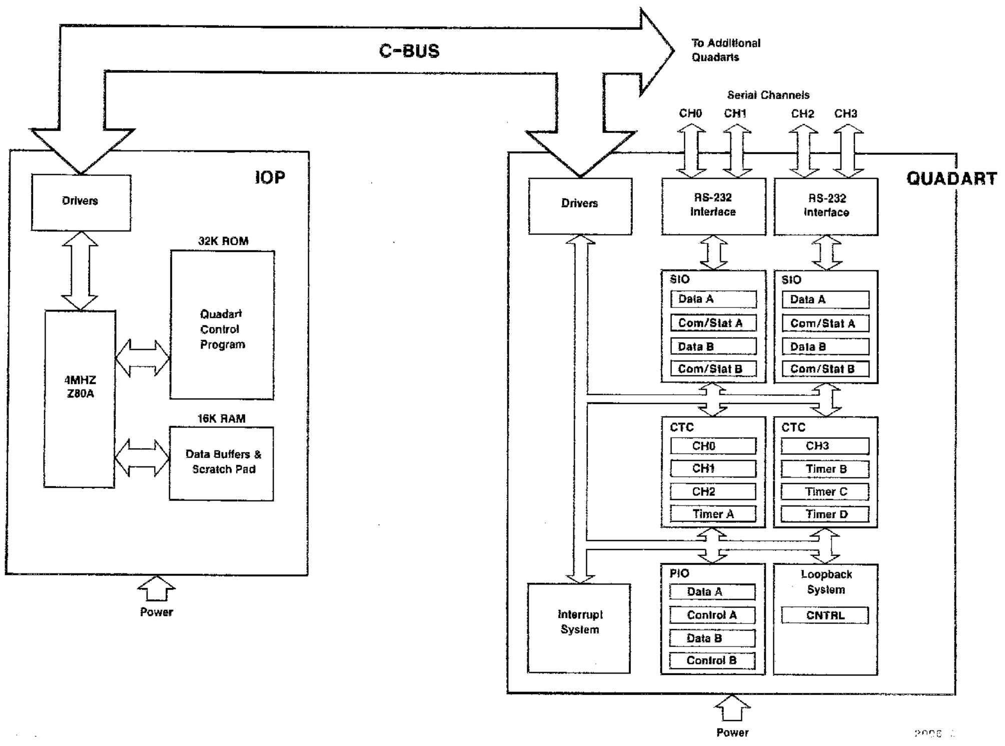


Figure 12: IOP/QUADART STAND ALONE BLOCK DIAGRAM

The Quadart interrupt structure at the intra-device level is shown to the left in the figure. This category of interrupts refers to the different interrupt sources within the same LSI device (SIO, CTC and PIO). Each SIO channel has four potential interrupt sources, and each of these sources may be programmed to supply a unique interrupt vector. The priority within each channel is fixed with Special Receive (RXSPCL) Conditions having the highest priority, and Transmitter Buffer Empty (TBE) conditions having the lowest. The PIO manages eight DSR and RI modem input lines in two groups of four. A transition of any logical combination of lines within a group may be programmed to generate an interrupt with its own vector. The priorities of the two PIO groups are fixed with channel 0 and 1 transitions having higher priority than channel 2 and 3 transitions. Each of the two CTCs manages four counter/timer circuits, and each counter may be individually programmed to generate an interrupt upon reaching count zero with its own unique vector. The priorities of the CTC channels are also fixed and ordered as shown in the figure. Note that there are a total of 26 Quadart interrupt sources and corresponding vectors (labeled V0 through V25 in the figure), and each of these sources may individually be enabled or disabled from generating an interrupt to the IOP, and supplying a vector during interrupt acknowledge.

The Quadart inter-device structure refers to the interrupt priorities among entire LSI devices. This priority scheme is established through the use of the IEI (Interrupt Enable In) and IEO (Interrupt Enable Out) pins on each LSI device. The Quadart wiring gives the highest priority to the SIO which manages channels 0 and 1, and gives the lowest priority to the CTC which manages the bit rate of channel 3 and timers B, C and D. Taking the intra- and inter-device priorities together, SIO channel 0 RXSPCL Conditions have the highest interrupt priority on the entire Quadart board, while timer D has the lowest priority. All priorities discussed so far are fixed, and all interrupt requests and interrupt vectors are sent to the C-Bus managing IOP processor, not the host.

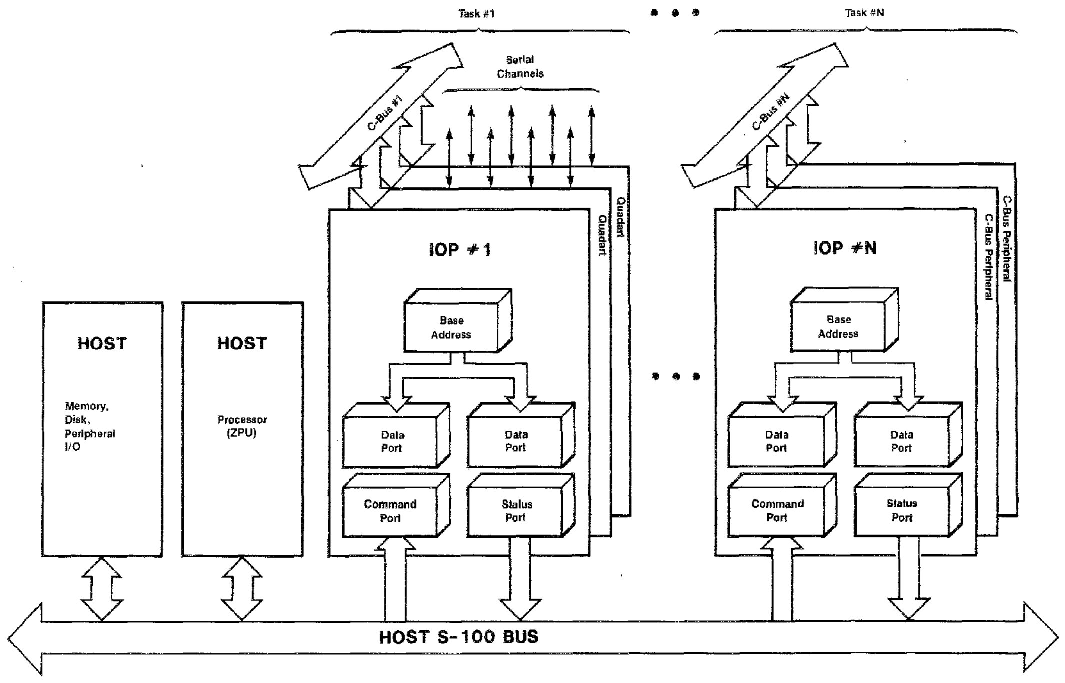


Figure 13: IOP/QUADART IN A HOST SYSTEM

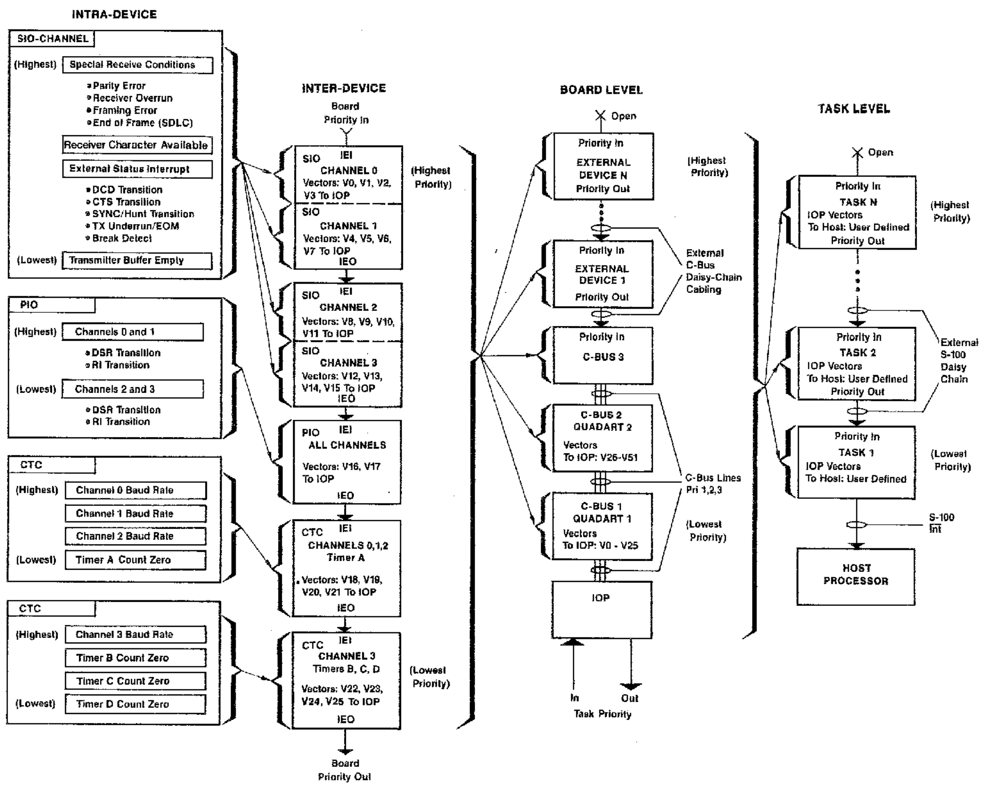

INTRA-DEVICE
SIO-CHANNEL
(Highest) Special Receive Conditions
•Parity Error
•Receiver Overrun
•Framing Error
•End of Frame (SDLC)
Receiver Character Available
External Status Interrupt
•DCD Transition
•CTS Transition
•SYNC/Hunt Transition
•TX Underrun/EOM
•Break Detect
(Lowest) Transmitter Buffer Empty
PIO
(Highest) Channels 0 and 1
•DSR Transition
•RI Transition
(Lowest) Channels 2 and 3
•DSR Transition
•RI Transition
CTC
(Highest) Channel 0 Baud Rate
Channel 1 Baud Rate
Channel 2 Baud Rate
(Lowest) Timer A Count Zero
CTC
(Highest) Channel 3 Baud Rate
Timer B Count Zero
Timer C Count Zero
(Lowest) Timer D Count Zero
INTER-DEVICE
Board Priority In
SIO IEI CHANNEL 0 Vectors: V0, V1, V2, V3 To IOP
SIO CHANNEL 1 Vectors: V4, V5, V6, V7 To IOP IEO
(Highest Priority)
SIO IEI CHANNEL 2 Vectors: V8, V9, V10, V11 To IOP
SIO CHANNEL 3 Vectors: V12, V13, V14, V15 To IOP IEO
PIO IEI ALL CHANNELS Vectors: V16, V17 To IOP IEO
CTC IEI CHANNELS 0,1,2 Timer A Vectors: V18, V19, V20, V21 To IOP IEO
CTC IEI CHANNEL 3 Timers B, C, D Vectors: V22, V23, V24, V25 To IOP IEO
(Lowest Priority)
Board Priority Out
BOARD LEVEL
Open
Priority In EXTERNAL DEVICE N Priority Out
(Highest Priority)
Priority In EXTERNAL DEVICE 1 Priority Out
External C-Bus Daisy-Chain Cabling
Priority In C-BUS 3
C-BUS 2 QUADART 2 Vectors To IOP: V26-V51
C-BUS 1 QUADART 1 Vectors To IOP: V0 - V25
C-Bus Lines Pri 1,2,3
(Lowest Priority)
IOP
In Out Task Priority
TASK LEVEL
Open
Priority In TASK N IOP Vectors To Host: User Defined Priority Out
(Highest Priority)
Priority In TASK 2 IOP Vectors To Host: User Defined Priority Out
External S-100 Daisy Chain
Priority In TASK 1 IOP Vectors To Host: User Defined
(Lowest Priority)
S-100 Int
HOST PROCESSOR

Figure 14: QUADART/IOP/HOST INTERRUPT STRUCTURE

When an IOP manages several Quadarts, or any other peripheral in the expanding Cromemco C-Bus product line, then individual subsystems may be prioritized among themselves at the board level with the C-Bus Priority In and Priority Out lines on each board. This scheme prioritizes the boards from C-Bus 1 (lowest priority), C-Bus 2, C-Bus 3, External Device 1, External Device 2, ..., to External Device N (highest priority). Quadarts are assigned board priorities C-Bus 1 through C-Bus 3 by properly wiring a DIP header plug for socket IC28 (see Section 8.3), and in this case, no further external wiring is required. As more boards are added to the C-Bus, external C-Bus daisy chain cabling must be supplied interconnecting board Priority In and Priority Out lines to assign board priorities External Device 1 through External Device N. Note that board priorities are user defined, and again the boards send interrupt requests and interrupt vectors to the IOP processor, not the host.

An IOP and the peripherals attached to its C-Bus would typically perform some function related task for a host system, such as serial I/O. The individual tasks may be prioritized among themselves at the task level by using the S-100 Bus Priority In and Priority Out lines on each IOP board. External S-100 Bus Daisy-Chain cabling interconnecting the IOPs (or many other Cromemco S-100 Bus task subsystems such as the 16FDC, PRI, and TU-ART) prioritize the tasks, giving the task at the end of the daisy chain the highest priority, and the task at the beginning of the chain the lowest priority. Each task issues interrupt requests to the host processor over S-100 bus line $\overline{\text{INT}}$, and in the case of an IOP-managed task, the IOP may supply a variable program defined interrupt vector during interrupt acknowledge from the host. This topic is discussed fully in Cromemco's Input/Output Processor Instruction Manual.

1.9 CHANNEL TO CHANNEL LOOPBACK

By outputting one byte to Quadart port **CNTRL,** the Quadart **loopback** system creates a single loopback data path. For diagnosing, self-testing and developing programs, the loopback data path may connect any SIO $\overline{\text{TxD}}$ output pin or modem RxD input line (data sources) to any SIO $\overline{\text{RxD}}$ input pin or modem TxD output line (data sinks). The loopback system is idealized in Figure 15.

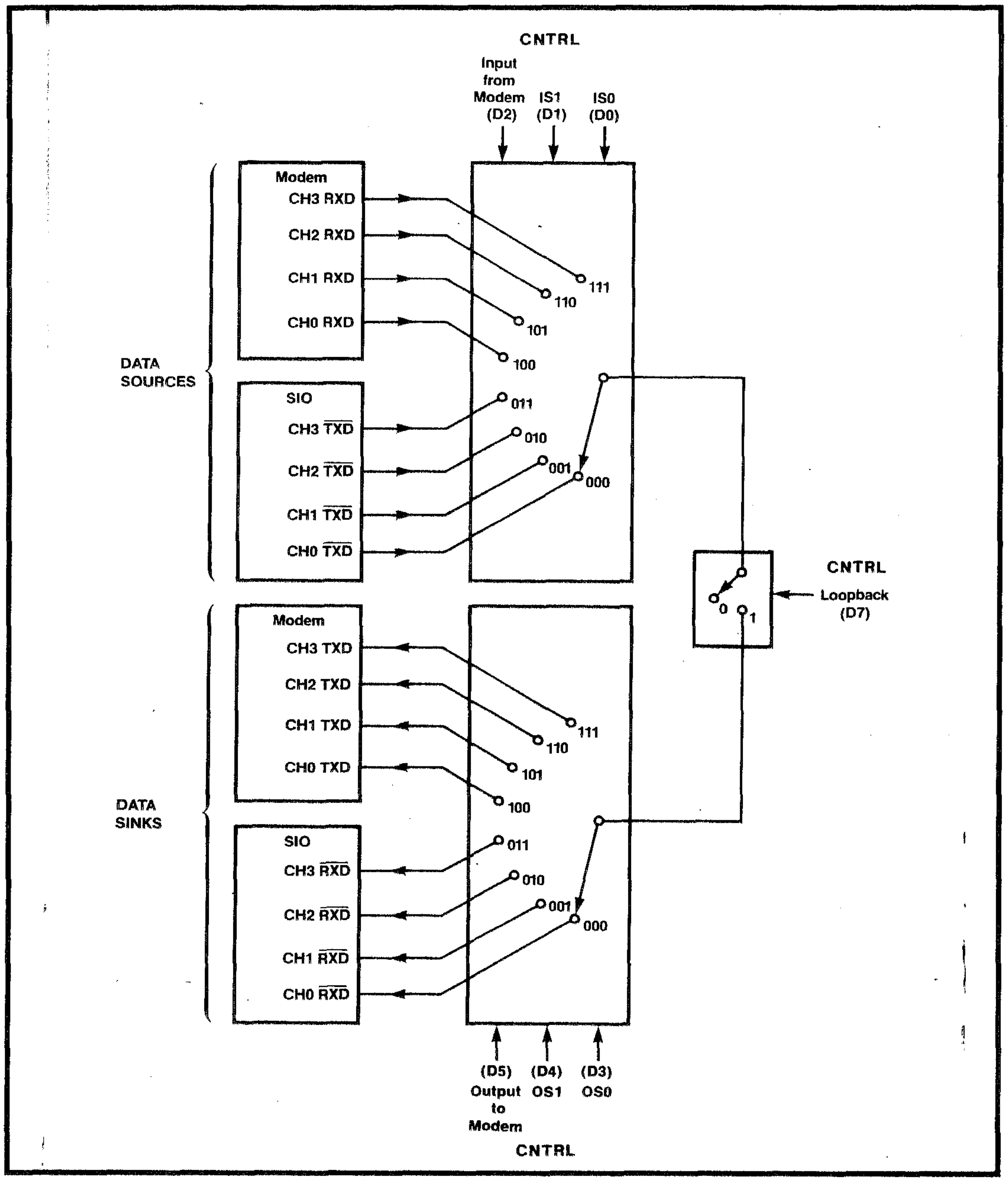


Figure 15: IDEALIZED QUADART LOOPBACK SYSTEM

The eight Quadart data sources are at the top left of the figure. Bits D2, D1 and D0 output to port CNTRL and select one of these sources by multiplexing it onto the loopback path. CNTRL bit D7 either opens or closes the loopback path, while bits D5, D4 and D3 route the loopback data to one of eight data sinks, shown at the bottom left. The figure shows the state of the loopback circuitry after a Quadart reset or a Power On Clear (POC). Chapter 10 discusses the Quadart loopback system in detail.

1.10 IOP/QUADART SET UP AND INITIAL CHECKOUT

The following initial checkout procedure exercises all four SIO channels in the asynchronous mode, all four CTC channels, and certain PIO data paths. Successfully completing the procedure should give the user confidence that the major subsystems on the IOP and the Quadart (excepting the loopback system, which is not exercised) are functioning properly.

EQUIPMENT SET UP

The following equipment is needed for this test procedure: a powered S-100 bus (two adjacent slots); a Cromemco IOP board; a Cromemco IOP Monitor ROM (version 00.08 or higher); a Cromemco Quadart board; a Cromemco C-Bus cable assembly; a console set at 75, 110, 134.5, 150, 300, 600, 1200, 2400, 4800, 9600 or 19200 bits per second with an RS-232C DTE interface; and a 25-pin DTE cable assembly from the console which mates to Quadart connector J2.

1. Set the Quadart base address to 40h (the setting shown in Figure 7). The IOP address switch setting is irrelevant in this procedure.

2. Install a Cromemco IOP Monitor ROM in IOP socket IC9. Align IC pin 1 with IOP legend arrow tip which points to socket pin 1.

3. Install C-Bus cabling between IOP socket J1 and Quadart socket J10. Align cable stripe as shown in Figure 1.

4. Connect DTE cabling between console and Quadart socket J2 (channel 0). Align cable stripe as shown in Figure 1.

5. Install both IOP and Quadart in adjacent S-100 bus slots with power off. Apply power to S-100 bus and to the console.

INITIAL CHECKOUT

1. Press the console **RETURN** key several times (ten at most). The IOP Monitor program is scanning the Quadart ports for activity, and it needs several RETURN keystrokes to determine the console bit rate.

2. The IOP Monitor program should now print the message **CROMEMCO IOP XX.YY** to the console, where **XX** is the IOP Monitor program version number, and **YY** is its release number.

3. Type **Z 4000 S80 33 <CR>** at the console, where <CR> stands for a RETURN keystroke. This is an IOP Monitor command which fills (zaps) an 80h long swath of IOP RAM memory starting at 4000h with the data byte 33h. You should see the command echoed on the console exactly as typed.

4. Type **DM 4000 <CR>** at the console. This is the IOP Monitor display memory command which causes a (default) swath of 80h bytes of IOP RAM memory to be formatted and displayed on the console. You should now see all 33h data. This completes the test of Quadart channel 0 at the console's current bit rate.

5. To further exercise CTC channel 0 (the channel 0 bit rate clock), turn power off, select another console bit rate from the above group, apply power, then test from the first step.

6. To test the other three Quadart channels, turn power off, remove the DTE connector from Quadart J2 and reconnect it to J4 (channel 1), or J6 (channel 2), or to J8 (channel 3). Apply power and test from the first step. This completes the initial IOP/Quadart check out.

If the check out procedure is not successful, check each procedure step carefully again before contacting Cromemco. Note the following possibilities: the Quadart base address is not set to 40h; the IOP Monitor ROM is installed in the wrong socket, is installed backwards, is bent under the pin, or is the wrong ROM; the DTE cable assembly stripe is aligned wrong, attached to the wrong connector (e.g., J3 instead of J2) or the cable/connectors are defective; the interface levels are wrong (e.g., 20 mA current loop instead of RS-232C), the bit rate is not allowed, or the console is defective.

Chapter 2

QUADART INITIALIZATION

This chapter's opening section discusses the general aspects of Quadart initialization. Sections 2.2 and 2.3 deal with configuring the PIO and CTCs to provide $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clocks to the SIO inputs; this material is a prerequisite to Chapters 4 through 7 on operating the SIOs in the three major modes. SIO clock control is only one function performed by the PIO and CTCs; PIO initialization and programming for modem control appear in Chapter 9, while CTC initialization and programming for a general purpose timer control are covered in Chapter 11. Chapters 5 through 7 discuss SIO initialization for each of the three major operating modes.

2.1 QUADART RESET AND POWER ON CLEAR STATE

The Quadart board is reset by an active low level on C-Bus line $\overline{\text{RESET}}$, and the $\overline{\text{RESET}}$ line is controlled from the IOP board. When the power is turned on, Power On Clear (POC) circuitry on the IOP forces line $\overline{\text{RESET}}$ to dwell at 0 volts momentarily then change to +4 volts, thereby resetting the Quadart after each POC sequence. The IOP may also force C-Bus line $\overline{\text{RESET}}$ low at any time by setting bit D7 in its Control Register, IOP output port 02h. Bit D7 is not latched, and is for test purposes only. These are the only two ways to reset the Quadart. Note in particular that an active low level on S-100 bus line $\overline{\text{RESET}}$ does **not** reset the Quadart.

A reset from either source resets both SIOs, both CTCs and the loopback system, but **not** the PIO (the device has no reset pin). A reset has the following effect on the Quadart:

1. The four SIO transmitters and receivers are disabled. The disabled transmitter TxD outputs are forced marking, resulting in -10 volt levels (approximately) at the RS-232C interface.

2. All RTS and DTR modem output lines are turned off (marking), resulting in -10 volt levels at the RS-232C interface.

3. All SIO interrupts (RXSPCL, RCA, EXSTAT and TBE interrupts from every channel) are disabled.

4. All CTC counter/timers stop counting.

5. All CTC channel interrupts are disabled.

6. No loopback data path exists. This means that all modem RxD input lines directly drive their normal SIO $\overline{RxD}$ input pins, and all SIO $\overline{TxD}$ output pins directly drive their normal TxD modem output lines.

7. The PIO state is undefined after a POC, and the PIO state is unchanged by a C-Bus clear from the IOP implying:

   a. the eight PIO output lines CY0 through CY3 and ExtCk0 through ExtCk3 are latched in random states after a POC; the eight output lines are unchanged by a C-Bus clear.

   b. the PIO interrupt configuration is random after a POC so the PIO may be issuing spurious interrupt requests to the IOP; the PIO interrupt configuration is unchanged by a C-Bus clear.

Spurious PIO interrupts after a POC should not be troublesome since IOP interrupts are always disabled after a POC (the Z80A powers up with its $\overline{INT}$ input pin masked). A POC also forces the IOP processor to start program execution at address 0000h (the physical beginning of IOP ROM), and the program starting at 0000h would typically begin with an IOP/Quadart initialization routine, following this outline:

1. Locate the stack in IOP RAM memory.

2. Configure the IOP board. See Cromemco's <u>Input/Output Processor Instruction Manual</u>.

3. Configure the Quadart PIO (mode, input/output lines and interrupts); see Section 2.2 and Chapter 9.

4. Configure the Quadart CTCs (baud rates, timers and interrupts); see Section 2.3 and Chapter 11.

5. Configure the Quadart SIOs (modes, channel parameters and interrupts); see Chapters 4, 5, 6 or 7, depending on the SIO mode.

If using interrupts:

6. Load the IOP Z80A Interrupt Register to define the IM2 indirect jump table memory page (I-Page).

7. Select Z80A Interrupt Mode 2 (IM2).

8. Enable Interrupts.

After initializing the IOP/Quadart system, the IOP program would either begin its main Quadart control routine in a polled environment, or begin running a background program waiting for Quadart interrupts in an interrupt driven environment.

2.2 PIO INITIALIZATION

The PIO provides SIO $\overline{TxC}$ and $\overline{RxC}$ clock source control, and supplies additional modem control for lines (DSR, RI and CY) which are not supported by the SIOs for switched line service. The PIO itself is a programmable, dual port (PIO A and PIO B), parallel I/O device. The PIO's 16 I/O lines may be individually programmed as either inputs or outputs, and preselected logical combinations of line transitions may optionally generate interrupts to the IOP. Once the IOP has defined the PIO mode by outputting control bytes to the PIO Control A and PIO Control B registers (ports Qbase + 09h and Qbase + 0Bh), the IOP performs parallel I/O with the PIO by outputting and inputting bytes to and from the PIO Data A and PIO Data B registers (ports Qbase + 08h and Qbase + 0Ah). Since the PIO's 16 I/O lines are hardware committed to fixed Quadart functions, the data flow direction (IN/OUT) of each line must conform to the directions defined in Figure 16.

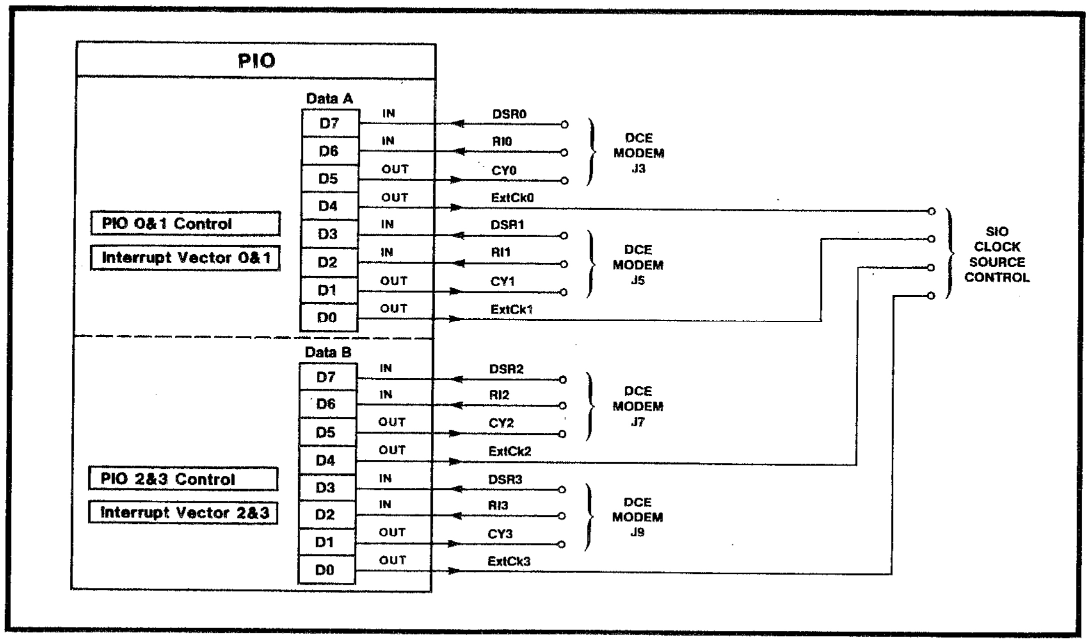


Figure 16: SIMPLIFIED QUADART PIO BLOCK DIAGRAM

The PIO is logically partitioned into two ports, PIO A and PIO B. Each port manages eight bits, and each 8-bit group services two of the Quadart's four channels. Each channel is assigned two modem input lines, **DSR** and **RI**, one user defined modem output line, **CY**, and one SIO clock source control line, **ExtCk**. To establish this correspondence, PIO A and PIO B must be programmed to the **control mode** with lines D7, D6, D3 and D2 defined as inputs, and lines D5, D4, D1 and D0 defined as outputs. The mode of each PIO port is selected by writing a mode control byte, formatted as shown in Figure 17, to PIO Control Port A and Control Port B. The PIO expects an I/O line direction definition byte to follow the mode control byte, and its format is also shown in Figure 17.

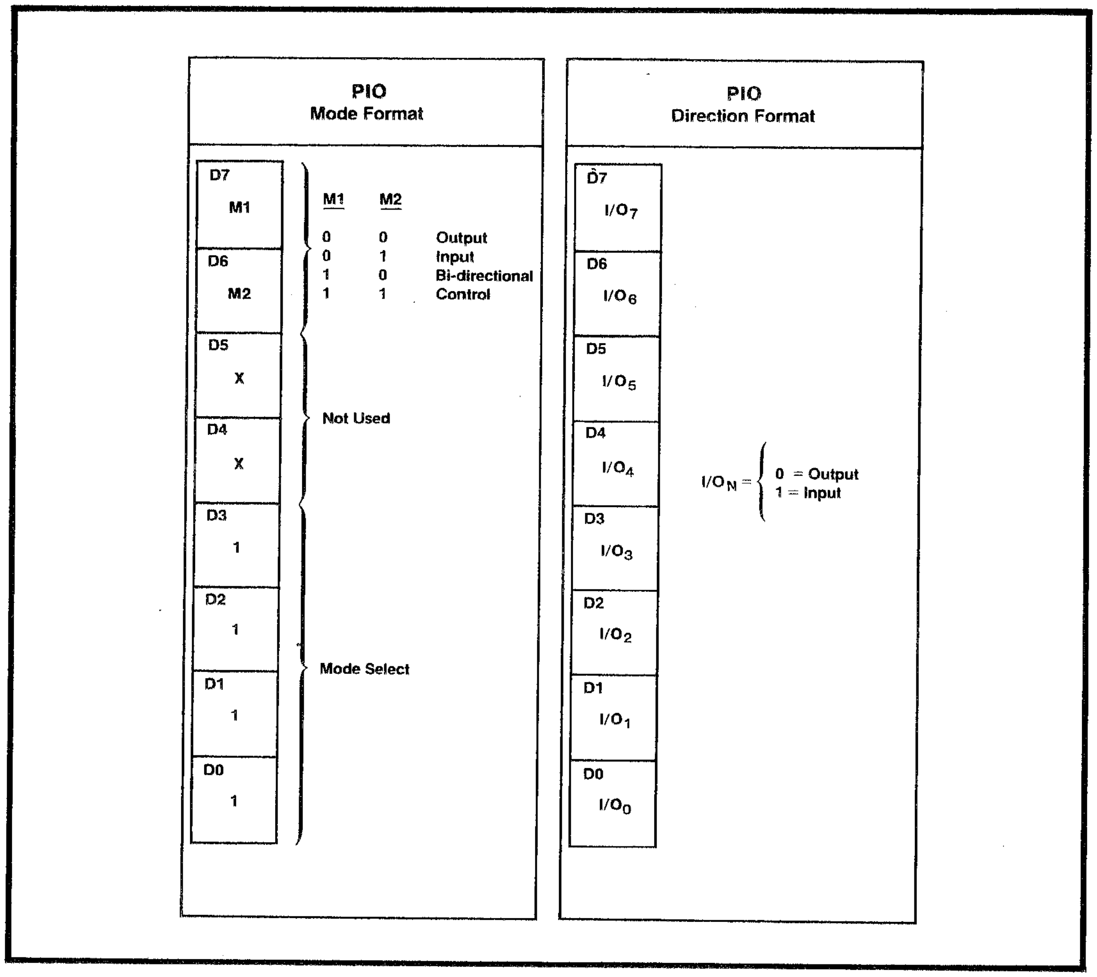


Figure 17: PIO MODE & DIRECTION CONTROL FORMATS

The PIO interrupt mode and vector for port A and port B are defined by writing control bytes to each PIO Control Port. The interrupt control byte formats are shown in Figure 18. Only one control byte is written to each port if its interrupts are disabled, but three bytes must be written if its interrupts are enabled; first, the port's interrupt vector; second, an interrupt control byte; and third, an interrupt mask byte. Refer to Section 9.2 for further details on PIO interrupt programming. The following Z80 assembly language subroutine initializes each PIO port in the control mode, defines the PIO line directions as per Figure 16, and disables interrupts from both PIO ports. A Quadart Base Address of 40h is assumed.

The IOP may commence parallel I/O with the PIO after it is initialized. The IOP outputs 8 bits to the PIO in parallel, but only PIO output lines D5, D4, D1 and D0 are affected. The output data is latched in non-inverted form. The IOP inputs eight bits from the PIO in parallel, and all eight bits are valid. Bits D7, D6, D3 and D2 reflect the real time values of the DSR and RI modem lines (logic 0 = spacing, logic 1 = marking), while bits D5, D4, D1 and D0 reflect the latched values previously outputted to the CY and ExtCk output lines.

The IOP program typically changes only one PIO output line at a time and leaves the other three in their previous states, except during PIO initialization when four lines are defined in parallel. This is easily done by reading PIO data to get the current output line states, then logically ANDing this byte with a mask containing a single logic 0 and the rest logic 1s to reset an output line, or by logically ORing this byte with a mask containing a single logic 1 and the rest logic 0s to set an output line. This modified byte is then written as PIO data to complete the output line modification. This process is illustrated in the program segment illustrating PIO output bit control.

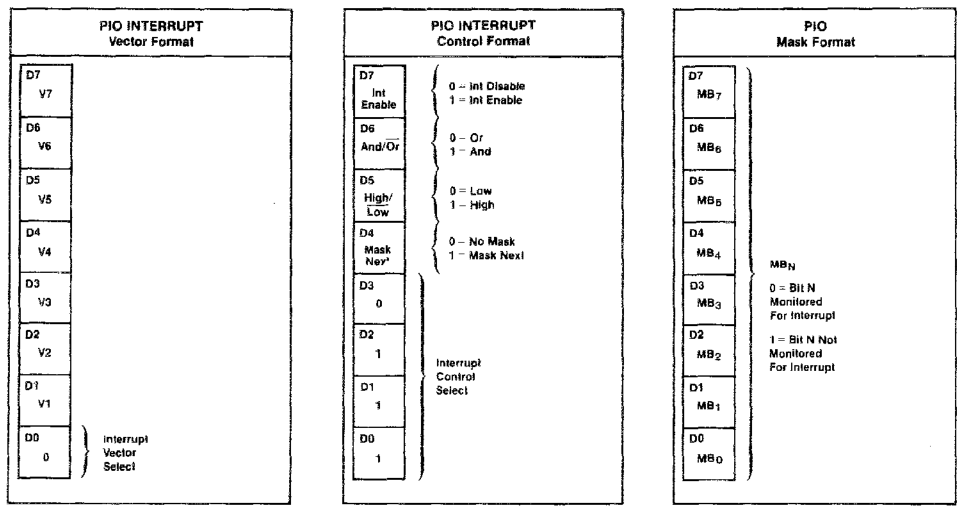


Figure 18: PIO INTERRUPT CONTROL BYTE FORMATS

```
CROMEMCO Z80 Macro Assembler version 03.07


                   ;       PIO Initialization Subroutine

                   ;  This subroutine initializes PIO port A and port B
                   ;  in the control mode, defines the sixteen I/O line
                   ;  directions, and disables both port A and port B
                   ;  interrupts.

        (0040)     QBASE    EQU     40h             ;Assumed Quadart Base Address.
        (0049)     PCOM01   EQU     QBASE+09h       ;Ch 0&1 PIO Command Port.
        (004A)     PCOM23   EQU     QBASE+0Ah       ;Ch 2&3 PIO Command Port.
        (00CF)     PMODE    EQU     11001111b       ;Control mode byte.
        (00CC)     PDIR     EQU     11001100b       ;I/O line directions control byte,
                                                    ;0 = output, 1 = input.
        (0000)     PIDSBL   EQU     00000111b       ;Disable interrupts control byte.

                            ENTRY   PINIT           ;This module is called by
                                                    ;external modules.

0000  3ECF         PINIT:   LD      A,PMODE         ;Ch 0&1 PIO to control mode,
0002  D349                  OUT     PCOM01,A        ;
0004  3ECC                  LD      A,PDIR          ;define Ch 0&1 line directions,
0006  D349                  OUT     PCOM01,A        ;
0008  3E00                  LD      A,PIDSBL        ;disable Ch 0&1 PIO interrupts.
000A  D349                  OUT     PCOM01,A        ;

000C  3ECF                  LD      A,PMODE         ;Ch 2&3 PIO to control mode,
000E  D34A                  OUT     PCOM23,A        ;
0010  3ECC                  LD      A,PDIR          ;define Ch 2&3 line directions,
0012  D34A                  OUT     PCOM23,A        ;
0014  3E00                  LD      A,PIDSBL        ;disable Ch 2&3 PIO interrupts.
0016  D34A                  OUT     PCOM23,A        ;

0018  C9                    RET
```

```
CROMEMCO Z80 Macro Assembler version 03.07


                   ;       PIO Output Bit Control Example

                   ;  This example program first initializes the Quadart
                   ;  PIO by calling routine PINIT (see above), then it
                   ;  initializes lines CY0, ExtCk0, CY1, ExtCk1 to logic 0,
                   ;  and lines CY2, ExtCk2, CY3, ExtCk3 to logic 1.  The
                   ;  routine then sets bit CY0, and resets lines ExtCk3
                   ;  from their current states while leaving all other
                   ;  PIO output lines unaffected.

        (0040)     QBASE    EQU     40h             ;Assumed Quadart Base Address.
        (0048)     PDAT01   EQU     QBASE+08h       ;Ch 0&1 PIO Data Port.
        (004A)     PDAT23   EQU     QBASE+0AH       ;Ch 2&3 PIO Data Port.

                            EXTRN   PINIT           ;External PIO module.

0000' CD0000#      PIOBIT:  CALL    PINIT           ;Initialize PIO mode, lines.

0003' 3E00                  LD      A,00000000b     ;Reset CY0, ExtCk0, CY1 and
0005' D348                  OUT     PDAT01,A        ;ExtCk1 in parallel.
0007' 3E33                  LD      A,00110011b     ;Set CY2, ExtCk2, CY3 and
0009' D34A                  OUT     PDAT23,A        ;ExtCk3 in parallel.

                   ;        :
                   ;        :                       (do something else)
                   ;        :

000B' DB48                  IN      A,PDAT01        ;Set CY0, Ch 0&1 bit D5.
000D' F620                  OR      00100000b       ;
000F' D348                  OUT     PDAT01,A        ;

0011' DB4A                  IN      A,PDAT23        ;Reset ExtCk3, Ch 2&3 bit D0.
0013' E6FE                  AND     11111110b       ;
0015' D34A                  OUT     PDAT23,A        ;

                   ;        :
                   ;        :                       (continue)
                   ;        :

0017' (0000')               END     PIOBIT
```

2.3 CTC INITIALIZATION

Eight independent programmable timer/counters are supplied by the two CTCs. Four of these are available as SIO $\overline{TxC}$ and $\overline{RxC}$ clock sources, and the other four may be used as general purpose timers. The eight CTC channels are mapped into the Quadart I/O port numbers listed in Table 1. Each channel's mode of operation, counting rate, and interrupt behavior upon reaching count zero are programmed by outputting control bytes to its port. Each channel's current count may be sampled in real time by inputting a byte from its port. A simplified representation of a single CTC device is shown in Figure 19.

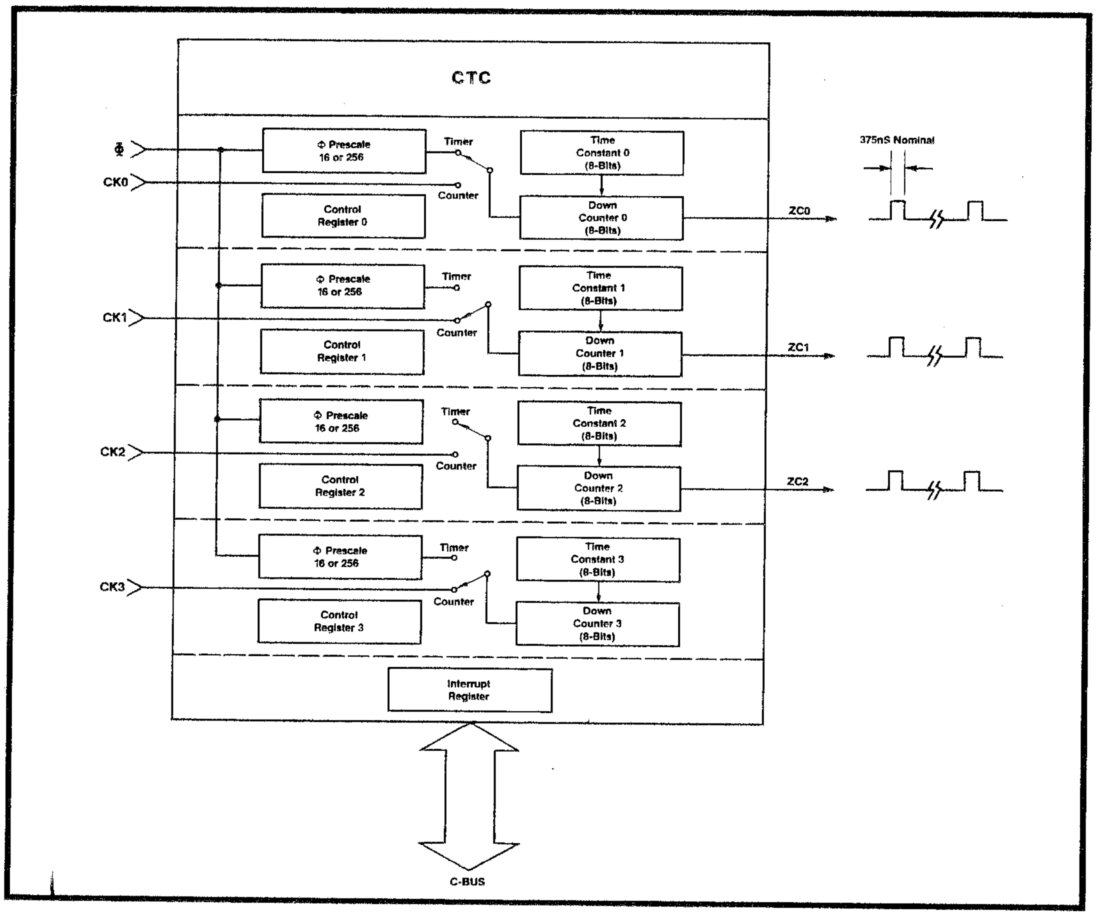


Figure 19: SIMPLIFIED CTC INTERNAL STRUCTURE

The timing element in each CTC channel is an 8-bit down counter. Three of four counters on each CTC supply a hardware zero count output pin, and four of these pins are multiplexed to the SIO $\overline{TxD}$ and $\overline{RxD}$ input pins as shown in Figure 21 below. A CTC channel is initialized by writing a **mode control** byte and a **time constant** byte to its port (see Figure 20 for formats). The mode control byte connects one of three clock frequencies to the down counter: **250 KHz** (4.000 MHz prescaled by 16); **15.625 KHz** (4.000 MHz prescaled by 256); or **307.692 KHz** (4.000 MHz externally prescaled by 13). A counter is automatically preloaded with the programmed 8-bit time constant (00h is interpreted as 256 decimal) from which it counts down until reaching count zero. Upon reaching count zero, the zero count pin pulses high for 375 nSec, the time constant is automatically reloaded, and counting continues uninterrupted.

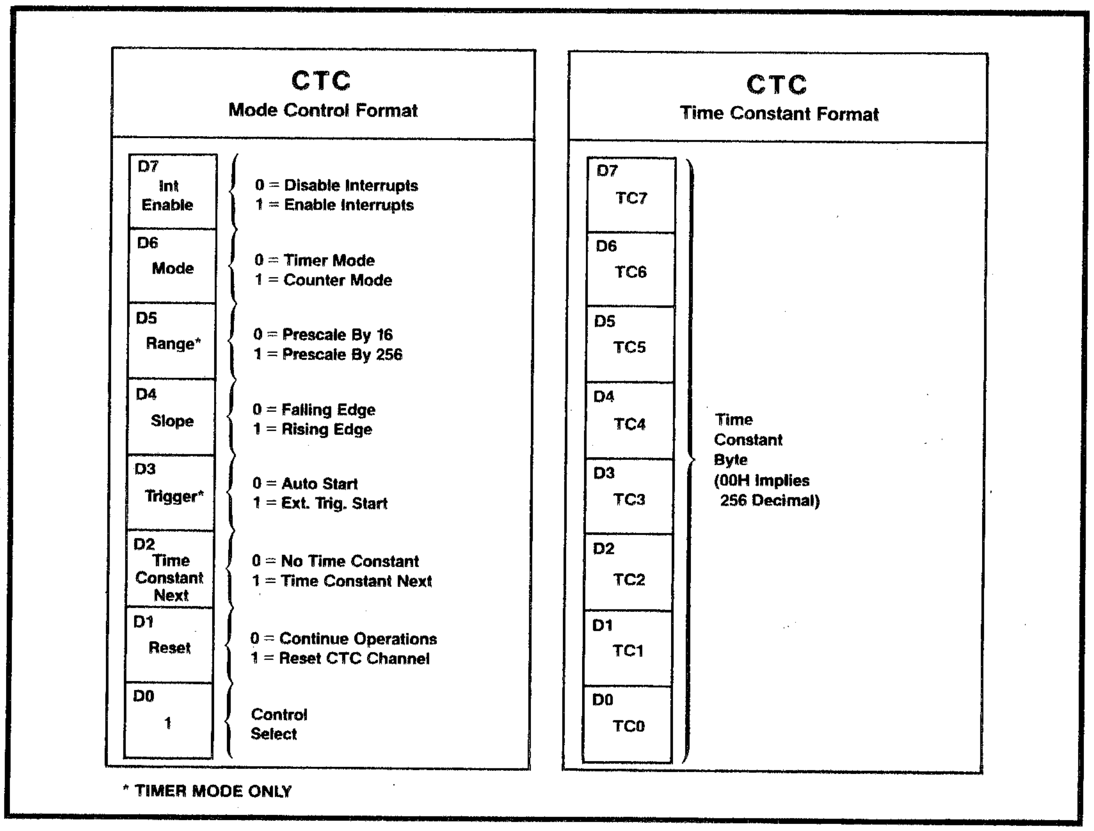


Figure 20: CTC MODE CONTROL AND TIME CONSTANT FORMATS

Bit D6 of the mode control byte programs the CTC channel to either the **timer mode** or the **counter mode.** In the timer mode, clock input O, which parallel feeds all CTC channels, is prescaled by 16 if bit D5 is reset, or 256 if bit D5 is set. On the Quadart, O is a symmetric 4.000 MHz crystal controlled clock (see Figure 21). Bit D4 causes the counter to either decrement on O's rising or falling edge. In the timer mode then, the waveform at a CTC zero count output pin is a pulse train whose course frequency may be varied by selecting between a 16 or a 256 prescaler, and whose fine frequency may be varied by choosing one of 256 possible time constants. Expressed mathematically, Ft = (4,000,000 Hz) / ([prescaler]*[time constant])

where Ft is the CTC zero count pulse train frequency in the timer mode, prescaler is either 16 or 256, and the time constant ranges from 1 to 256 (256 is programmed as 00h). Choosing a prescale factor of 16 and a time constant of 1 gives the maximum timer mode pulse train frequency (250 KHz). Choosing a prescale factor of 256 and a time constant of 256 gives the minimum frequency (61.0 Hz).

In the counter mode, CTC inputs CK0, CK1 and CK2 clock the down counters; or more precisely, transitions in CK0, CK1 and CK2 enable their respective counters to be decremented on O's next rising edge. On the Quadart, the CK0, CK1 and CK2 inputs are parallel driven by a 307,692 Hz asymmetric waveform. This signal is derived from a 4.000 MHz crystal controlled clock feeding a modulus 13 prescaler (IC44 in the Quadart schematic).

In the counter mode then, the CTC zero count output waveform is a pulse train whose frequency varies by choosing 1 of 256 possible time constants. Expressed mathematically,

Fc = (307,692 Hz) / (time constant)

where Fc is the CTC zero count output frequency in the counter mode, and the time constant ranges from 1 to 256 (programmed as 00h). Choosing a time constant of 1 gives the maximum counter mode pulse train frequency (307.692 KHz). Choosing a time constant of 256 gives the minimum frequency (1.202 KHz).

Refer to Chapter 3 for a CTC bit rate programming example, and Chapter 11 for a CTC general purpose timer programming example. Also see the <u>CTC Technical Manual</u> for further CTC programming details.

Chapter 3

CHANNEL BIT RATES

A Quadart channel's **bit rate** measures the frequency with which individual serial data bits are transmitted and received over the channel, measured in bits per second. A channel's transmitter and receiver bit rates, which may differ, are equal to the SIO $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ input frequencies divided by the channel clock multiplier. The clock multiplier (X1, X16, X32 or X64) is programmed during SIO initialization, and the same value is used for both the transmitter and receiver.

When a Quadart channel is operating in the asynchronous mode a local CTC counter typically drives the SIO $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ inputs in parallel. When the serial channel is operating in the synchronous mode, modem TxC and RxC inputs typically drive the SIO $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clock inputs. This chapter begins by discussing how to control SIO clock source selection; Sections 3.2 and 3.3 then discuss asynchronous and synchronous bit rate programming.

3.1 SIO CLOCK SOURCE CONTROL

The $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clock inputs to each SIO channel are driven by the outputs of a dual 4-input multiplexer (see Figure 21). Multiplexer control lines **ExtCk0 through ExtCk3** and **EXT0 through EXT3** connect either CTC outputs, or modem input lines TxC and RxC to the SIO clock inputs.

The IOP controls individual ExtCk lines with data bits written to the PIO (see Section 2.2). The Quadart user may either install insulated jumper straps at any board position EXT0 through EXT3, or leave them open (factory shipped condition). The PIO outputs and the user defined jumper straps together multiplex either a CTC clock or a modem TxC/RxC clock to the SIO inputs as shown in Table 3.

The four Quadart EXT jumper pads are shipped open anticipating the usual Quadart configurations (the last two table entries). The normal DCE and DTE set ups for an asynchronous channel are shown in Figure 22, and the normal DCE set up for a synchronous channel is shown in Figure 23.

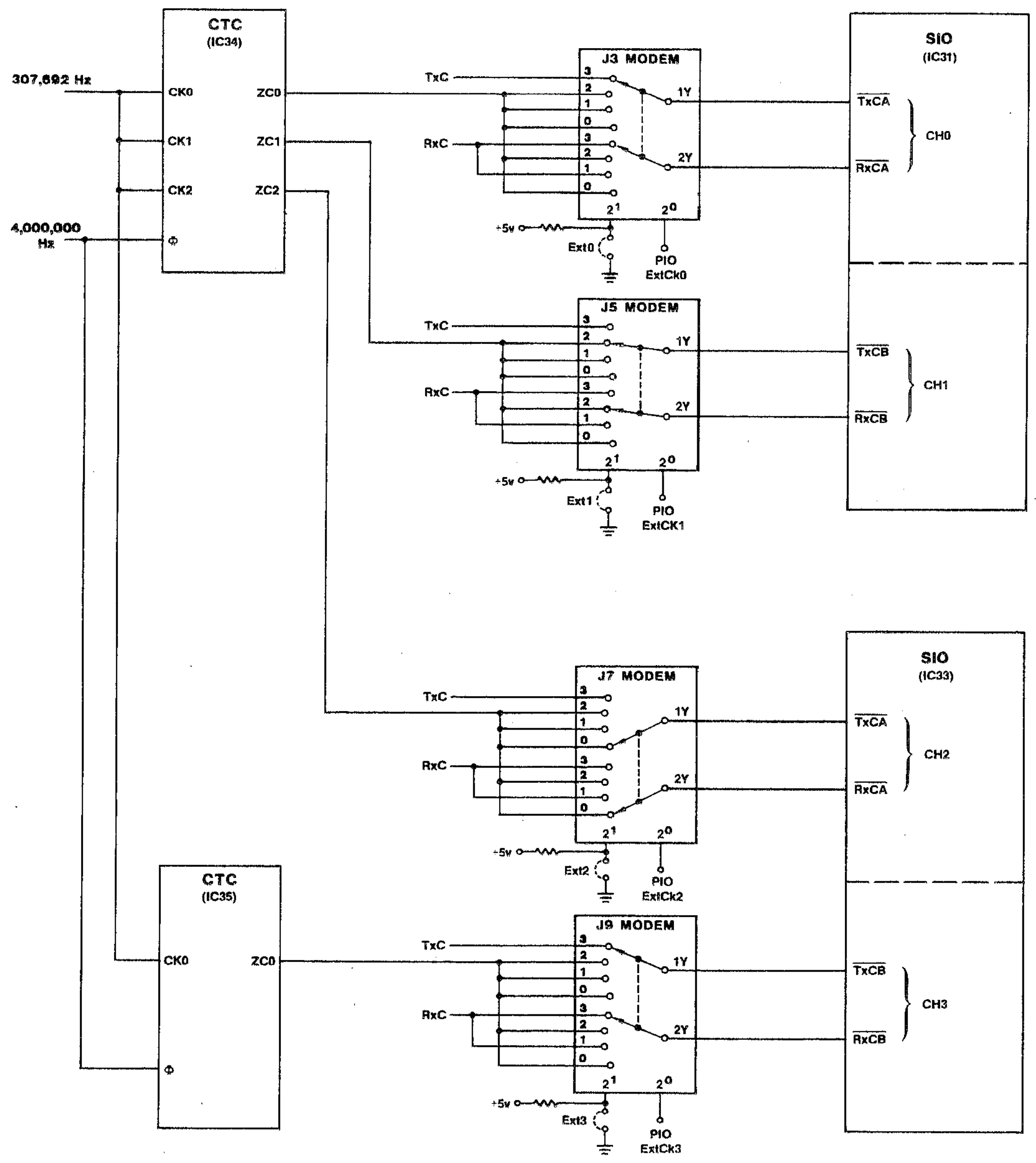


Figure 21: SIO CLOCK SOURCE CONTROL CIRCUITRY

Table 3: SIO TxC AND RxC CLOCK SOURCES

(Each Channel)

EXT Jumper	PIO Bit ExtCk	SIO $\overline{RxC}$ Source	SIO $\overline{TxC}$ Source
Installed	0	CTC	CTC
Installed	1	Modem RxC	CTC
Removed	0	CTC	CTC
Removed	1	Modem RxC	Modem TxC

An asynchronous channel typically employs two local clock sources, one at each end of the link, running at X16, X32 or X64 the bit rate (not at X1 the bit rate). These clocks must closely match in frequency (the Quadart clocks are crystal controlled), but their relative phase is not important. On the Quadart, the CTC channels serve as local clocks, and they are made to parallel drive the SIO $\overline{TxC}$ and $\overline{RxC}$ inputs by resetting PIO bit ExtCk to logic 0.

Character exchange begins when the IOP outputs a character to an SIO Data Port. This loads the character into the SIO Tx buffer which in turn feeds the channel transmitter. The asynchronous transmitter formats the character and shifts it out bit by bit, beginning with the start bit, then the LSB, and so on, ending with the stop bit(s). Each bit requires 16, 32 or 64 clock cycles to be shifted out, depending on the clock multiplier. In a DCE connection, the TxD data bits are converted to RS-232C levels and fed to a modem for modulation. At the opposite end of the DCE link, a second modem demodulates the RxD data and converts it again to RS-232C levels. In the DTE link (e.g., a Quadart channel connected to a CRT terminal), the character bits are passed directly from one station to another, unmodulated and at RS-232C levels over the interconnecting cable.

At the receiving station, the leading edge of the start bit causes the asynchronous receiver to sample each bit near its midpoint (for a X16 clock, it waits 8 clock cycles, then samples the RxD line at 16 clock cycle intervals near the data bit midpoint), and assembles the character for parallel reading. A X16 multiplier is most often used as it is easy to generate digitally, it provides good resolution for midpoint sampling, and it does not demand an excessively high clock frequency.

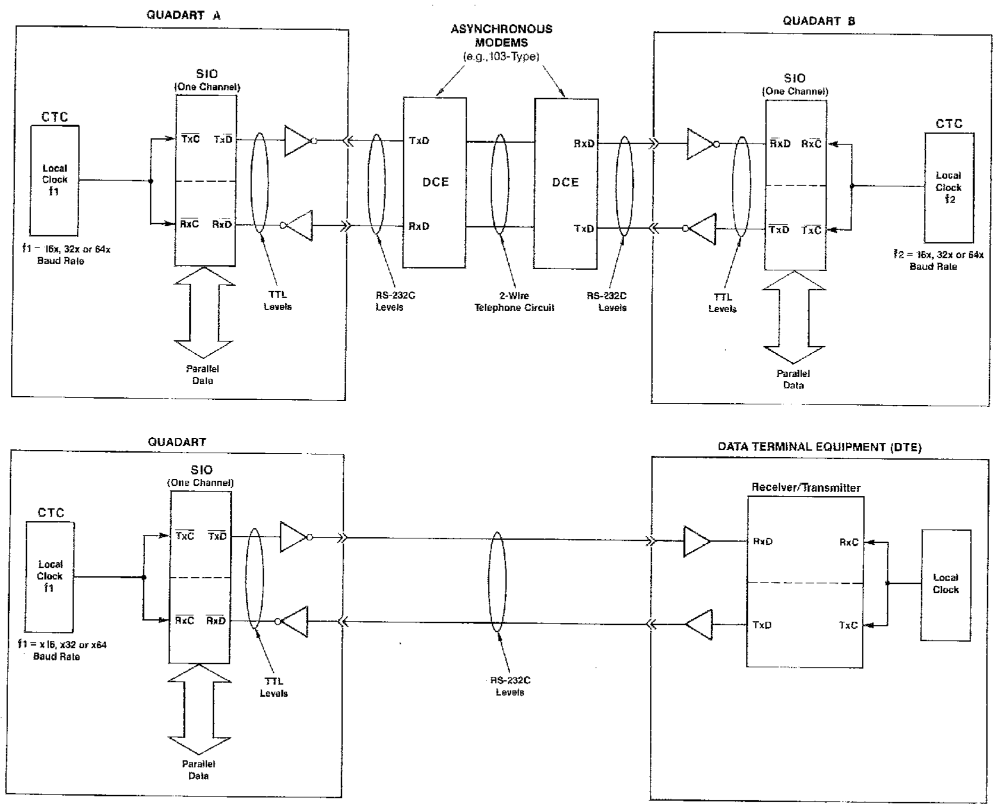


Figure 22: ASYNCHRONOUS DCE AND DTE CHANNEL SET UPS

In a Quadart DCE connection the maximum asynchronous bit rate is limited by either the telephone line bandwidth or the maximum modem baud rate (e.g., 300 baud for a Bell 103A Modem). In a DTE connection the maximum bit rate is limited by the interconnecting cable's bandwidth, by the maximum DTE equipment baud rate (e.g., 19,200 bits per second for most CRT terminals), or by the maximum CTC clock rate (307,692 Hz yielding 19,200 bits per second with a X16 multiplier).

In a synchronous communications link the Quadart is typically connected to a DCE-wired synchronous modem. Unlike the asynchronous case just described, $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ signals input to the SIO are supplied by a modem, not a local CTC (see Figure 23). This connection is established by programming PIO bit ExtCk to logic 1, the last entry of Table 3.

Each modem generates its own local TxC clock signal (some modem types may optionally slave the answering modem's TxC clock to the calling modem's TxC clock). Since the Quadart SIOs must be programmed with a X1 multiplier in the synchronous modes, the synchronous bit rate is then equal to the modem TxC frequency. For example, a Bell 201C modem supplying a 4,800 Hz TxC clock would exchange data at a 4,800 bits per second rate (the corresponding telephone line signaling rate is 1,600 baud, where each signal phase change encodes 3 binary bits).

A typical synchronous data exchange between Quadart A and Quadart B in the figure might proceed as follows: The IOP loads a character into the Quadart A transmitter. The transmitter shifts out the data bit by bit on each $\overline{\text{TxC}}$ falling edge, starting with the LSB and ending with the MSB. The originate modem modulates the TxD data for transmission over the 2-wire service. The receive modem demodulates the bits as RxD data, and also recovers the transmit clock from the data which is then termed circuit RxC. The modem RxC signal is routed to a Quadart B receiver, and this RxC signal is used to shift in one RxD data bit to the SIO receiver on each $\overline{\text{RxC}}$ rising edge. The SIO receiver then assembles the RxD bits into a character for parallel reading to complete the exchange. To recapitulate; in the asynchronous modes, a channel's bit rate is set by programming a local CTC clock frequency and an SIO multiplier of X16, X32 or X64, and PIO bit ExtCk is reset connecting a CTC to the SIO TxC and RxC inputs. In the synchronous modes, a channel's bit rate us set by the attached

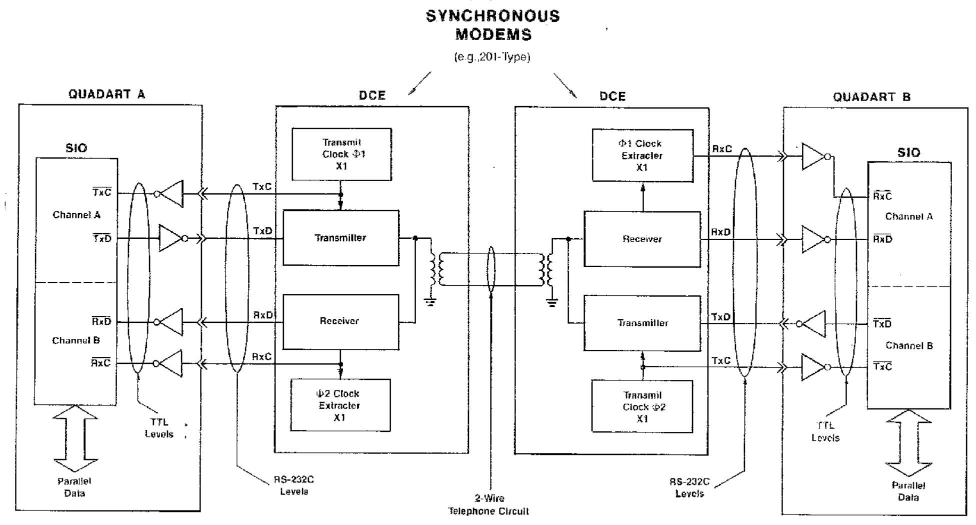


Figure 23: SYNCHRONOUS DCE CHANNEL SET UP

modem's TxC clock frequency, and no CTC programming is required. In the synchronous modes, the SIO multiplier must be X1, and PIO bit ExtCk is set connecting modem TxC and RxC lines to their corresponding SIO inputs. These are the normal configurations, and they correspond to the last two entries in Table 3 with the EXT jumper strap removed.

Installing the EXT jumper strap allows two Quadarts to be wired directly together without a modem (a **null modem**) in the synchronous modes. In this configuration (the second Table 3 entry) each channel receives its TxC clock from a local CTC. This same CTC signal is brought out to DCE connectors J3, J5, J7 and J9 as **External Clock**, pin 24, but converted to RS-232C levels (this signal is always enabled). The External Clock output signal from one channel is wired as a **modem RxC clock** input to another channel (DCE connector pin 17).

Since the CTC clock signal consists of a pulse train, with each pulse's width nominally equal to 375 nanoseconds, the External Clock cable length must be limited to a few inches in length. This means that only Quadarts occupying adjacent S-100 bus slots, or SIO channels on the same Quadart board, may be interconnected in this fashion, regardless of the channel bit rate. Note that this restriction does not apply to a direct asynchronous connection between Quadart channels since each end of an asynchronous link has its own local clock. Methods for extending the physical length between null modem Quadarts channels are treated in the Section 3.3.

3.2 ASYNCHRONOUS BIT RATE PROGRAMMING

A Quadart channel is initialized for asynchronous operation as follows:

1. Reset channel PIO bit ExtCk to connect a CTC to the SIO $\overline{TxC}$ and $\overline{RxC}$ inputs (see Section 3.1).

2. Define the channel asynchronous bit rate by outputting two bytes to the channel CTC port (a mode byte and a time constant byte).

3. Configure the SIO channel for asynchronous operation with a X16 clock multiplier (see Chapter 5).

An asynchronous channel is usually programmed with a X16 multiplier, so the CTC output frequency should be set 16 times higher than the bit rate. Table 4 tabulates CTC prescalers and time constants for several commonly used asynchronous bit rates assuming a X16 multiplier. Of course, custom bit rates may be generated with proper time constant/prescaler selections.

The following Z80 assembly language routine programs the CTC counters assuming all Quadart channels are operated in the asynchronous mode with X16 multipliers. Channel 1 is programmed for 110 bps, channel 2 for 300 bps, channel 3 for 4800 bps and channel 4 for 19200 bps, using the data from Table 3.

Table 4: COMMON BIT RATE CTC CONSTANTS

(assumes SIO X16 multiplier)

	TIMER MODE		**COUNTER MODE**
BIT RATE (bits/sec)	PRESCALER (decimal)	TIME CONSTANT (decimal)	TIME CONSTANT (decimal)
75	16	208	256
110	16	142	175
134.5	16	116	143
150	16	104	128
200	16	78	96
300	16	52	64
600	16	26	32
1,200	16	13	16
2,400	--	--	8
4,800	--	--	4
9,600	--	--	2
19,200	--	--	1

```
CROMEMCO Z80 Macro Assembler version 03.07

                 
                 ;       ASYNCHRONOUS BIT RATE SELECTION ROUTINE
                 
                 ;  This program segment initializes the four bit rate CTC
                 ;  counters assuming SIO asynchronous modes with X16 multi-
                 ;  pliers, and the channel bit rates shown below.  This
                 ;  segment calls external PIO initialization module PINIT.
                 
                 ;             Quadart       Bit Rate
                 ;             Channel      (bits/sec)
                 
                 ;                0            110
                 ;                1            300
                 ;                2           4800
                 ;                3          19200
                 
                 ;  For illustration purposes, the CTC counters driving
                 ;  Quadart channels 0 and 1 are operated in the timer
                 ;  mode, and those driving channels 2 and 3 are oper-
                 ;  ated in the counter mode.
                 
                         EXTRN   PINIT            ;PIO Initialization Routine.
                 
      (0040)     QBASE   EQU     40h              ;Assumed Quadart base address.
      (004C)     CTC0    EQU     QBASE+0Ch        ;Ch 0 CTC Port.
      (004D)     CTC1    EQU     QBASE+0Dh        ;Ch 1 CTC Port.
      (004E)     CTC2    EQU     QBASE+0Eh        ;Ch 2 CTC Port.
      (0050)     CTC3    EQU     QBASE+10h        ;Ch 3 CTC Port.
      (0048)     PDAT01  EQU     QBASE+08h        ;Ch 0&1 PIO Data Port.
      (004A)     PDAT23  EQU     QBASE+0Ah        ;Ch 2&3 PIO Data Port.
      (0007)     MODE1   EQU     00000111b        ;Mode 1:
                                                  ; - Disable PIO interrupts
                                                  ; - Timer mode
                                                  ; - Prescale by 16
                                                  ; - Falling edge trigger
                                                  ; - Auto start
                                                  ; - Time constant next
                                                  ; - Reset PIO channel
      (0067)     MODE2   EQU     01100111b        ;Mode 2:
                                                  ; - Disable PIO interrupts
                                                  ; - Counter mode
                                                  ; - Falling edge trigger
                                                  ; - Time constant next
                                                  ; - Reset PIO channel
                 
0000' CD0000#    RATES:  CALL    PINIT            ;Initialize PIO.
                 
0003' DB48               IN      A,PDAT01         ;Reset all PIO ExtCk
0005' E6EE               AND     11101110b        ;bits connecting CTC zero
0007' D348               OUT     PDAT01,A         ;count outputs to all SIO
0009' DB4A               IN      A,PDAT23         ;TxC and RxC inputs.
000B' E6EE               AND     11101110b        ;
000D' D34A               OUT     PDAT23,A         ;
                 
000F' 3E07               LD      A,MODE1          ;Ch 0 timer mode
0011' D34C               OUT     CTC0,A           ;(see above).
0013' 3E8E               LD      A,142            ;Prescale 16, time constant
0015' D34C               OUT     CTC0,A           ;142 for 110 baud.
                 
0017' 3E07               LD      A,MODE1          ;Ch 1 timer mode
0019' D34D               OUT     CTC1,A           ;(see above).
001B' 3E34               LD      A,52             ;Prescale 16, time constant
001D' D34D               OUT     CTC1,A           ;52 for 300 baud.
                 
001F' 3E67               LD      A,MODE2          ;Ch 2 counter mode
0021' D34E               OUT     CTC2,A           ;(see above).
0023' 3E04               LD      A,4              ;Time constant
0025' D34E               OUT     CTC2,A           ;4 for 4800 baud.
                 
0027' 3E67               LD      A,MODE2          ;Ch 3 counter mode
0029' D350               OUT     CTC3,A           ;(see above).
002B' 3E01               LD      A,1              ;Time constant
002D' D350               OUT     CTC3,A           ;1 for 19200 baud.
                 
002F' (0000')            END     RATES
```

3.3 SYNCHRONOUS BIT RATE PROGRAMMING

Synchronous bit rates are usually determined and supplied by the attached modem, as pointed out in Section 3.1. In such cases no CTC programming is required. A Quadart channel is then initialized for synchronous operation (byte synchronous or SDLC) as follows:

1. Set channel PIO bit ExtCk to connect modem TxC and RxC to the SIO $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ inputs (see Section 3.1).

2. Configure the SIO channel for either byte synchronous or SDLC operation with a X1 clock multiplier (see Chapters 6 and 7).

As pointed out in Section 3.1, two separated Quadarts may be wired directly together without modems provided a common clock source parallel drives them both. This arrangement is shown in Figure 24 (the figure represents either two physically separate Quadarts, or two channels on the same Quadart). Here, both Quadarts are initialized as in steps (1) and (2) above. The direct data and clock link between them operates at RS-232C levels, and the physical cable's bandwidth would typically limit the maximum bit transfer rate.

```
CROMEMCO Z80 Macro Assembler version 03.07


                          ;       SYNCHRONOUS NULL MODEM BIT RATE ROUTINE

                          ;  This routine configures Quadart A shown in figure 25 for
                          ;  synchronous null modem operation at 25 Kbits/second.  The
                          ;  channel 0 CTC clock is used to generate a 50 KHz pulse
                          ;  train in the timer mode at twice the desired data rate.
                          ;  PIO channel 0 bit ExtCk0 is set to connect the channel 0
                          ;  modem RxC and TxC (which are driven at 50 KHz) to the
                          ;  corresponding channel 0 SIO RxC and TxC inputs.

                                  EXTRN   PINIT             ;PIO initialize routine.

          (0040)          QBASE   EQU     40h               ;Assumed Quadart Base Address.
          (0048)          PDAT01  EQU     QBASE+08h         ;Ch 0&1 PIO Data Port.
          (004C)          CTC0    EQU     QBASE+0Ch         ;Ch 0 CTC Port.
          (0007)          MODE    EQU     00000111b         ;CTC Mode:
                                                            ; - Disable CTC interrupts
                                                            ; - Timer mode
                                                            ; - Prescale by 16
                                                            ; - Falling edge trigger
                                                            ; - Auto start
                                                            ; - Time constant next
                                                            ; - Reset CTC channel

0000' CD0000#             SYNBPS: CALL    PINIT             ;Initialize PIO.

0003' DB48                        IN      A,PDAT01          ;Set PIO bit ExtCk0 connecting
0005' F610                        OR      00010000b         ;modem TxC and RxC lines to
0007' D348                        OUT     PDAT01,A          ;SIO TxC and RxC inputs.

0009' 3E07                        LD      A,MODE            ;Program CTC channel 0 mode
000B' D34C                        OUT     CTC0,A            ;(see above).
000D' 3E05                        LD      A,5               ;Time constant = 5 for 50 KHz
000F' D34C                        OUT     CTC0,A            ;CTC output frequency.

0011' (0000')                     END     SYNBPS
```

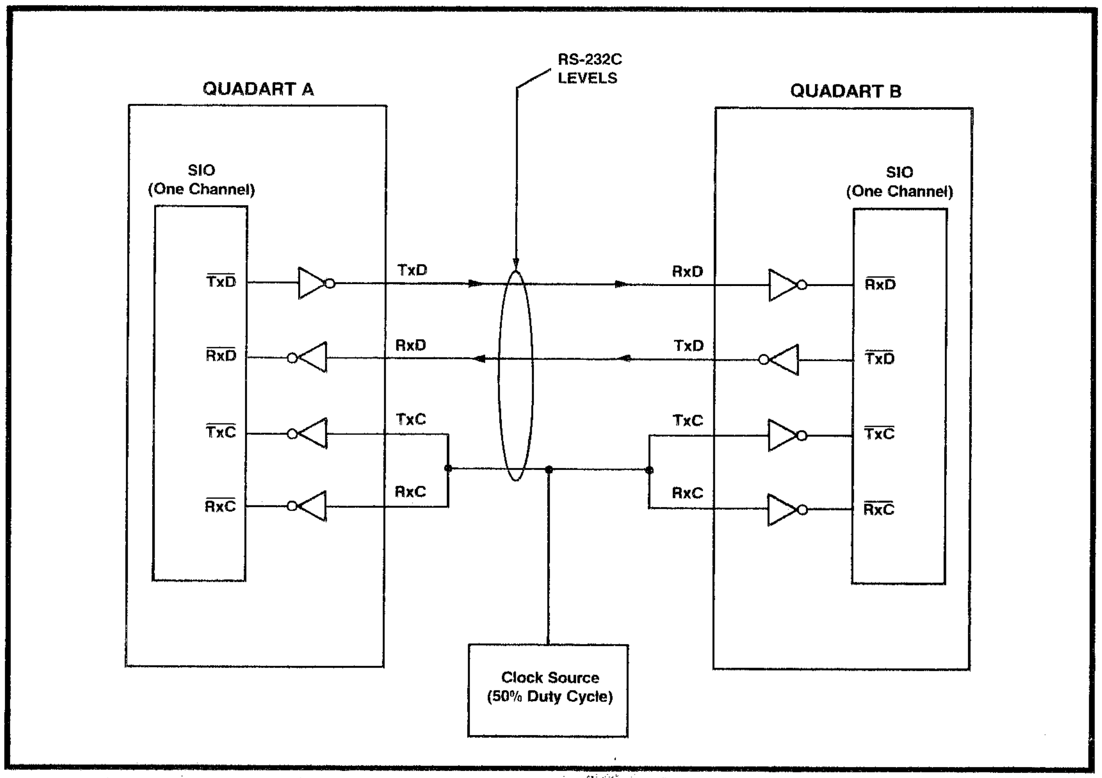


Figure 24: A DIRECT SYNCHRONOUS CONNECTION

A Quadart CTC clock brought out to DCE pin External Clock may serve as a common clock source by installing an EXT jumper strap (see Section 3.1), but the duty cycle of this waveform is so low that the interconnecting cable length is limited to a few inches. This limitation may be overcome by leaving the EXT jumper pads open (no jumper), and feeding the External Clock signal to a flip flop, thereby generating a 50% duty cycle clock signal (see Figure 25). If this is done, the CTC clock rate must be programmed to twice (X2) the synchronous bit rate, since the flip flop divides the External Clock frequency by two. Again, the synchronous channel would be initialized per steps (1) and (2) above.

The above Z80 assembly language routine programs Quadart channel 0 for null modem synchronous operation at 25 Kbits/second (this channel plays the role of Quadart A in Figure 25). The channel 0 CTC is used to generate a 50 KHz pulse train (twice the bit rate) in the timer mode at RS-232C line External Clock (J3, pin 24). In this application the EXT0 jumper pads must be left open (no jumper).

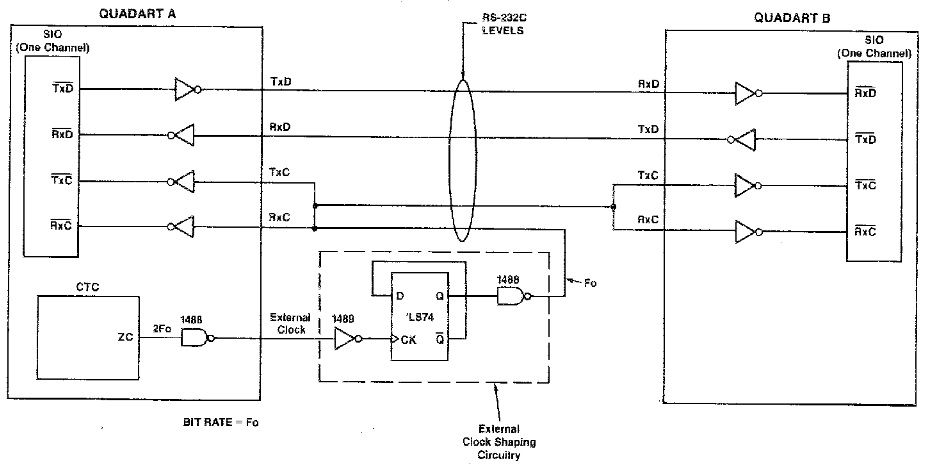


Figure 25: A DIRECT CONNECTION WITH A X2 CTC CLOCK

Chapter 4

SIO FUNCTIONAL DESCRIPTION

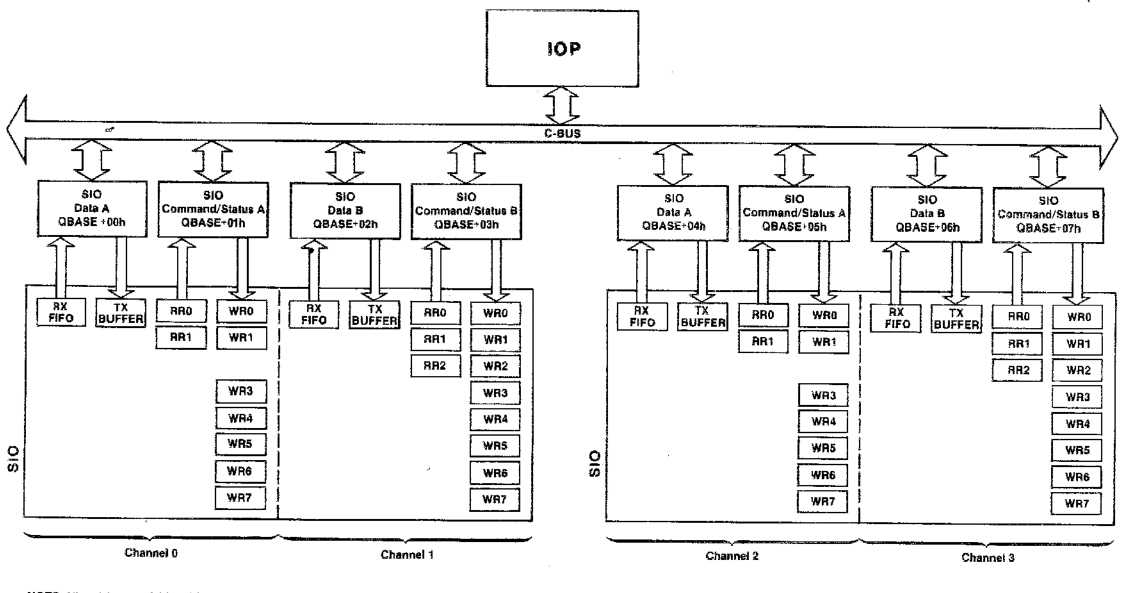
This chapter discusses SIO features which are common to the asynchronous, byte synchronous and SDLC modes. The following three chapters supplement this material by detailing SIO features which are unique to each mode.

4.1 SIO PROGRAMMING

The programming model for each SIO channel consists of 13 internal SIO registers. These 13 registers are logically grouped into 2 **data registers** (1 input and 1 output), and 11 **command/status registers** (3 input and 8 output). The IOP reads SIO channel status from 3 status registers termed RR0 through RR2 (for read register 0 through 2), and outputs SIO channel commands to 8 registers termed WR0 through WR7 (for write register 0 through 7). The IOP accesses all 13 registers of a single SIO channel through 2 dedicated Quadart I/O ports, **SIO Data** and **SIO Command/Status.** The SIO data and command/status port addresses are listed as the first 8 entries in Table 1, and the resulting arrangement is shown in Figure 26.

The Zilog and Mostek documentation logically partitions each SIO into Channel A and Channel B. On the Quadart, channels 0 and 1 correspond to SIO channels A and B (IC31 in the Quadart schematic diagram). Channels 2 and 3 correspond to SIO channels A and B (IC33). There are no WR2 and RR2 registers in channels 0 and 2 (the interrupt vector written to and read from channel 1 through these registers is shared by channels 0 and 1; likewise for channels 2 and 3).

Once an SIO channel has been configured with personality control bytes written to WR0 through WR7, the IOP transmits serial data characters by outputting them to the channel SIO Data Port, which in turn loads the characters into the channel Tx buffer. The IOP receives serial data characters by inputting them from the channel SIO Data Port, which in turn is fed data from the top of the channel Rx FIFO. All data read and written by the IOP in this manner are 8 bits wide, although some character bits may be ignored when read, or defined as filler bits when written, depending on the transmitter and receiver character lengths. Note that SIO modes are undefined after a POC, so the IOP **must not** attempt data I/O with an SIO until the channel is initialized by writing control bytes to WR0 through WR7.



NOTE: All registers are 8-bits wide.

Figure 26: SIO INTERNAL REGISTER ACCESS

WR0 through WR7, and RR0 through RR2 channel bit assignments appear on the manual overleaf for convenient reference, and a detailed description of each bit function is in Appendix A near the end of the manual. All read and write operations from RR0 through RR2 and to WR0 through WR7 **must** begin by outputting a byte to WR0 (the SIO always assumes the first byte written is directed to WR0) since the last three bits (D2, D1 and D0) of the byte output to WR0 are used to address an SIO register on the following I/O transfer. If the operation following the output to WR0 is another write, then the output data is used as a control byte which is routed to the WR0 - WR7 register targetted with WR0 byte. If the operation following the output to WR0 is a read, then the input data is a status byte supplied by the SIO from the targetted RR0 - RR2 register.

If a command is directed to WR0 itself, then the transaction is completed in one output operation. While bits D2, D1 and D0 serve as register pointers, the other five bits are used to issue various channel commands which are executed with the first WR0 write. This structure allows the programmer to issue frequently used commands to WR0 with each SIO register access. Care must be exercised when writing to and reading from the SIO command/status registers while any channel interrupts are enabled. See Section 8.2 for further details.

The SIO read/write registers are generally organized by function. WR1 and WR2 configure the channel interrupt response; WR3 configures the channel receiver; WR4 defines the channel mode; WR5 configures the channel transmitter; and WR6 and WR7 define the sync characters in the synchronous modes. RR0 provides Rx, Tx and modem status; RR1 provides channel Rx error status; and RR2 supplies a variable interrupt vector.

Not all of the SIO read/write registers are used in some channel operating modes while in others, all are used. For instance, half duplex polled asynchronous I/O with no error checking (e.g., serial output to a data logger) might only involve registers WR0 (to reset the channel and address the other registers), WR1 (to disable interrupts), WR4 (to define the mode, clock multiplier, stop bits and no parity), WR5 (to configure the transmitter) and RR0 (to poll for a TBE condition). On the other hand, interrupt driven full duplex synchronous operation would typically involve all registers, except possibly RR2.

The following example recapitulates several Quadart topics discussed thus far, and looks ahead to Chapter 5 with a simple, yet complete IOP **echo program.** Refer to Appendix A for SIO bit function descriptions.

Example: AN ASYNCHRONOUS ECHO PROGRAM

The Z80 assembly language IOP resident program below configures Quadart channel 0 for polled full duplex asynchronous serial I/O at 300 baud, no parity, and no error (framing or overrun) checking to keep the example simple. The Tx and Rx character lengths are set to 8 bits with 1 stop bit, or 10 bits/character including the start and stop bits. After channel 0 is initialized, the routine first polls RR0 until there is an Rx character available, then reads the character through the channel 0 SIO Data Port, then outputs the unmodified character back through the channel 0 SIO Data Port, completing the echo operation.

The program assumes an RS-232C, 300 baud terminal attached to Quadart channel 0 DTE connector J2. The program is located in IOP RAM memory spanning 4000h - 4034h (35h bytes). The program is short enough that the reader may wish to manually load and verify its operation using IOPMON, the IOP monitor program. This is easily done by: (1) setting the terminal to 300 baud, (2) connecting the terminal to Quadart connector J2, (3) entering IOPMON by depressing the RETURN key, (4) entering the code with the **SM 4000** (substitute memory starting at 4000h) command, (5) verifying the entries with the **DM 4000** (display 80h of memory starting with address 4000h), (6) starting program execution with the **G 4000** (go to 4000h and begin execution) command, and (7) pressing any console key and observing the character echoed back to the terminal. The channel 0 CTC frequency may easily be changed to match other terminal baud rates by altering the contents of RAM address 4010h, as shown in Table 4.

```
CROMEMCO Z80 Macro Assembler version 03.07

                ;       ECHO PROGRAM EXAMPLE

                ;  See manual text for program description.

      (0040)    QBASE   EQU     40h             ;Assumed Quadart Base Address.
      (0040)    SDATA0  EQU     QBASE+00h       ;Ch 0 SIO Data Port.
      (0041)    SCMST0  EQU     QBASE+01h       ;Ch 0 SIO Command/Status Port.
      (0048)    PDAT01  EQU     QBASE+08h       ;Ch 0&1 PIO Data Port.
      (0049)    PCOM01  EQU     QBASE+09h       ;Ch 0&1 PIO Command Port.
      (004C)    CTC0    EQU     QBASE+0CH       ;Ch 0 CTC Port.

      (4000)            ORG     4000h           ;Start of IOP ram.

                        ;=================================================
                        ; Initialize Quadart
                        ; Channel 0.
                        ;=================================================

4000  3ECF      ECHO:   LD      A,11001111b     ;PIO servicing Ch 0&1 to
4002  D349              OUT     (PCOM01),A      ;control mode 3.
4004  3ECC              LD      A,11001100b     ;Define I/O lines;
4006  D349              OUT     (PCOM01),A      ;0=output, 1=input.
4008  3E00              LD      A,00000000b     ;Reset ExtCk0 = 0 to connect
400A  D348              OUT     (PDAT01),A      ;CTC to SIO Ch 0 TxC and RxC.

400C  3E67              LD      A,01100111b     ;Ch 0 CTC to counter mode,
400E  D34C              OUT     (CTC0),A        ;interrupts disabled.
4010  3E40              LD      A,64            ;Time constant for 300 baud
4012  D34C              OUT     (CTC0),A        ;assuming X16 multiplier.

4014  211F40            LD      HL,SIOBEG       ;Point to SIO Ch 0 initial-
4017  0E41              LD      C,SCMST0        ;ization data in preparation
4019  0608              LD      B,SIOEND-SIOBEG ;for OTIR block move.
401B  EDB3              OTIR                    ;Initialize SIO Ch 0.

401D  1808              JR      RCA?            ;Jump around block move data.

                        ;=======================
                        ; SIO Block Move
                        ; Initialization Data.
                        ;=======================

401F  19        SIOBEG: DB      00011001b       ;Point to WR1, reset Ch 0.
4020  00                DB      00000000b       ; - Disable all Ch 0 interrupts

4021  04                DB      00000100b       ;Point to WR4.
4022  44                DB      01000100b       ; - X16 clock
                                                ; - asynchronous mode
                                                ; - 1 stop bit
                                                ; - no parity

4023  03                DB      00000011b       ;Point to WR3.
4024  C1                DB      11000001b       ; - 8 bit Rx characters
                                                ; - no auto enables
                                                ; - enable receiver

4025  05                DB      00000101b       ;Point to WR5.
4026  EA                DB      11101010b       ; - DTR on
                                                ; - 8 bit Tx characters
                                                ; - enable transmitter
                                                ; - no break
                                                ; - RTS on
                SIOEND:

                        ;=================================================
                        ; Main Program Echo Loop
                        ;=================================================

4027  3E00      RCA?:   LD      A,0             ;Point to RR0,
4029  D341              OUT     (SCMST0),A      ;
402B  DB41              IN      A,(SCMST0)      ;then read RR0 and test status
402D  CB47              BIT     0,A             ;bit RCA -- try again if Rx
402F  28F6              JR      Z,RCA?          ;character not available.

4031  DB40              IN      A,(SDATA0)      ;Read data from FIFO top,
4033  D340              OUT     (SDATA0),A      ;echo it back to terminal.

4035  18F0              JR      RCA?            ;Repeat indefinitely.


4037  (4000)            END     ECHO
```

Several comments are in order after presenting this first complete example program. First, notice that only one of the four Quadart channels is used. A POC disables all SIO transmitters and receivers (see Section 2.1), so only the channels used need to be initialized. Four such echo routines could be sequentially arranged and executed in IOP memory, each with a different bit rate, echoing to four different terminals. Second, this program does not do anything with the received data other than echoing it back to the terminal. The usual application would perform error checking on the received data, store it in a RAM buffer, and preprocess it in some way for a host processor. Third, this is a polled, as opposed to interrupt driven routine. When more than one Quadart channel is in use, IM2 interrupt driven routines are the rule as they are faster, shorter and inherently simpler. Finally, note how the OTIR I/O block move instruction is economically used to initialize SIO channel 0 in the program.

4.2 SIO INTERRUPTS

The SIO may interrupt the IOP in response to four general events within an SIO channel: Special Receive (RXSPCL) Conditions; Receiver Character Available (RCA); Transmitter Buffer Empty (TBE); and External/Status (EXSTAT) transitions. The SIO provides a wide variety of interrupt configuration options, allowing the programmer to selectively enable and disable events which may generate interrupts, and also to choose the type of interrupt vector supplied by the SIO during interrupt acknowledge. The complete range of options for a single channel is shown in Figure 27.

An SIO channel's interrupt configuration is defined by control bytes output to SIO registers WR1 and WR2. Register WR2 in channel 1 is shared by channels 0 and 1, while register WR2 in channel 3 is shared by channels 2 and 3. Register WR2 is loaded with the interrupt vector which the SIO places on the C-Bus in response to an interrupt acknowledge from the IOP. If WR1 control bit Status Affects Vector is set, then bits V3, V2 and V1 of the interrupt vector placed on the bus are modified to point at the highest priority enabled interrupt event currently requesting service; if control bit Status Affects Vector is reset, then the contents of WR2 are placed unmodified on the bus during interrupt acknowledge. Table 5 shows the values of V3, V2 and V1 corresponding to each enabled event, with the event priorities listed in descending order.

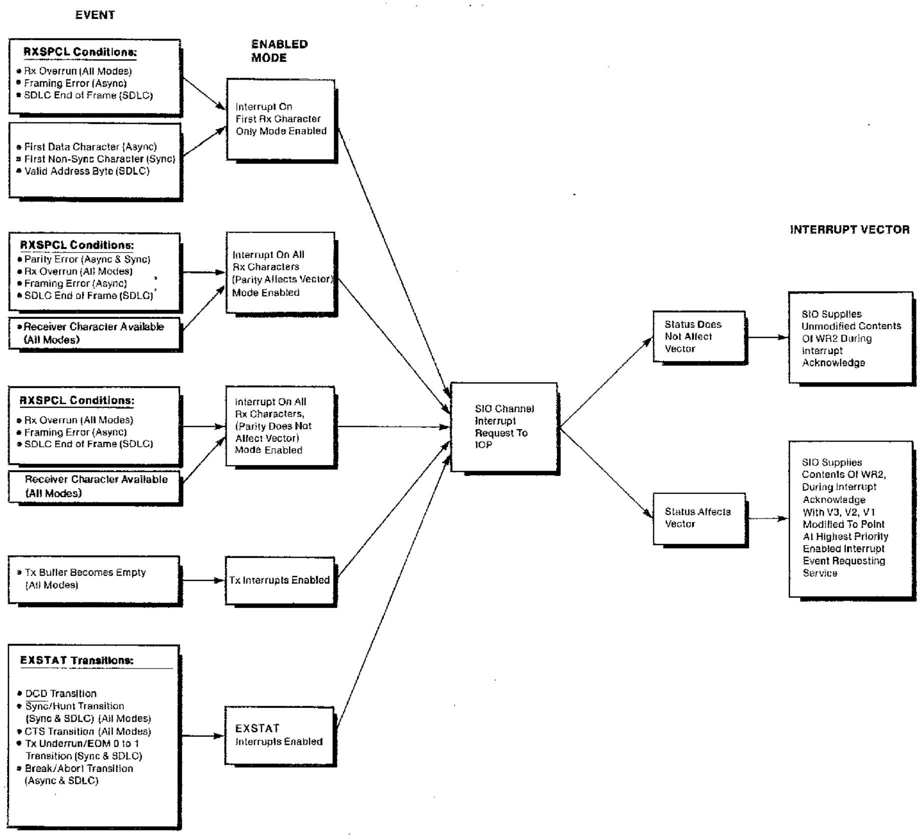

EVENT
ENABLED MODE
INTERRUPT VECTOR
RXSPCL Conditions:
• Rx Overrun (All Modes)
• Framing Error (Async)
• SDLC End of Frame (SDLC)
• First Data Character (Async)
• First Non-Sync Character (Sync)
• Valid Address Byte (SDLC)
Interrupt On First Rx Character Only Mode Enabled
RXSPCL Conditions:
• Parity Error (Async & Sync)
• Rx Overrun (All Modes)
• Framing Error (Async)
• SDLC End of Frame (SDLC)
• Receiver Character Available (All Modes)
Interrupt On All Rx Characters (Parity Affects Vector) Mode Enabled
RXSPCL Conditions:
• Rx Overrun (All Modes)
• Framing Error (Async)
• SDLC End of Frame (SDLC)
Receiver Character Available (All Modes)
Interrupt On All Rx Characters, (Parity Does Not Affect Vector) Mode Enabled
• Tx Buffer Becomes Empty (All Modes)
Tx Interrupts Enabled
EXSTAT Transitions:
• DCD Transition
• Sync/Hunt Transition (Sync & SDLC) (All Modes)
• CTS Transition (All Modes)
• Tx Underrun/EOM 0 to 1 Transition (Sync & SDLC)
• Break/Abort Transition (Async & SDLC)
EXSTAT Interrupts Enabled
SIO Channel Interrupt Request To IOP
Status Does Not Affect Vector
Status Affects Vector
SIO Supplies Unmodified Contents Of WR2 During Interrupt Acknowledge
SIO Supplies Contents Of WR2, During Interrupt Acknowledge With V3, V2, V1 Modified To Point At Highest Priority Enabled Interrupt Event Requesting Service

Figure 27: SIO INTERRUPT STRUCTURE (EACH CHANNEL)

Table 5: SIO INTERRUPT VECTORS AND PRIORITIES

V3	V2	V1	WR2 CHANNEL 1	PRIORITY
1	1	1	Ch 0 RXSPCL	Highest
1	1	0	Ch 0 RCA	
1	0	0	Ch 0 TBE	
1	0	1	Ch 0 EXSTAT	
0	1	1	Ch 1 RXSPCL	
0	1	0	Ch 1 RCA	
0	0	0	Ch 1 TBE	
0	0	1	Ch 1 EXSTAT	
V3	V2	V1	WR2 CHANNEL 3	
1	1	1	Ch 2 RXSPCL	
1	1	0	Ch 2 RCA	
1	0	0	Ch 2 TBE	
1	0	1	Ch 2 EXSTAT	
0	1	1	Ch 3 RXSPCL	
0	1	0	Ch 3 RCA	
0	0	0	Ch 3 TBE	
0	0	1	Ch 3 EXSTAT	Lowest

Whenever interrupt driven Quadarts are used, control bit Status Affects Vector should be set to take advantage of the hardware supplied variable IM2 interrupt vector.

Higher priority Quadart events issue interrupt requests to the IOP in the middle of IOP servicing to lower priority events, but not vice versa. The IOP then suspends service to the lower priority event, and begins servicing the higher priority event (thus nesting service routines) **provided** the lower priority service routine has enabled IOP interrupts with an EI instruction.

Refer to Figure 27 and note there are three modes in which RCA interrupts may be armed: (1) RCA Interrupt On First Character Only, (2) RCA Interrupts On All Characters (Parity Affects Vector), and (3) RCA Interrupts On All Characters (Parity Does Not Affect Vector). If any of these three modes is enabled, RXSPCL interrupts are also **automatically enabled**; if none are enabled (mode RCA and RXSPCL Interrupts Disabled is selected), then RXSPCL interrupts are also automatically disabled. In this case RCA status is polled from RR0, and RXSPCL conditions are polled from RR1.

The RCA Interrupt On First Character Only mode is intended primarily for DMA block transfer data moves using the SIO Wait/Ready functions. In this mode, receiver interrupts are generated only by the first receiver character, and thereafter, by framing errors (asynchronous mode), overrun errors and SDLC end of frame conditions only (not parity). In these cases, the RXSPCL bits and the received data are both held (even if the received data is read) until an RXSPCL Reset command is issued to WR0. This mode would not typically be enabled since the Quadart SIO Wait/Ready output pins are uncommitted.

One of the two RCA Interrupts On All Character modes would typically be enabled on the Quadart. Parity errors may optionally be included or excluded from generating RXSPCL interrupts when selecting between these two modes. An RCA interrupt only is generated as each character is assembled and transferred to the Rx FIFO when there are no RXSPCL conditions on the character. If there is an RXSPCL condition on the assembled character then an RXSPCL interrupt **and** an RCA interrupt request become active simultaneously. RXSPCL interrupts have higher priority than concurrent RCA interrupts, so the SIO will always supply an RXSPCL vector first if control bit Status Affects Vector is set. The service routine removes the RXSPCL interrupt request by issuing an RXSPCL Reset command to WR0. After this is done, the RCA interrupt request will become active, and it is removed in turn by reading the character from the channel SIO Data Port. The two interrupt request removal actions may take place in the RXSPCL service routine (reading the Rx data removes the RCA interrupt; if this is done before interrupts are reenabled, no RCA interrupt acknowledge from the IOP occurs since interrupt requests are not latched by the Z80A), or in separate RXSPCL and RCA service routines.

Some characters do **not** generate any interrupts when received in the RCA Interrupts On All Character modes, and they are: (1) sync characters in byte synchronous mode if control bit Sync Character Load Inhibit is set, (2) all flags in SDLC mode, (3) all bits of **wrong address** frames in SDLC mode if control bit Address Search Mode is set, and (4) characters matching WR6 in SDLC mode if control bit Inhibit Sync Character Load is set. All bits and characters in these four categories are neither assembled nor loaded into the Rx FIFO.

The channel transmitter may issue interrupt requests to the IOP provided control bit TBE Interrupts Enable is set. If control bit Status Affects Vector is also set, then TBE interrupts supply a unique vector during interrupt acknowledge from the IOP. TBE interrupts are issued immediately after the contents of the Tx buffer are transferred to the transmitter itself for shifting out, leaving an empty Tx buffer behind. The TBE service routine removes the interrupt request by either loading a new character into the Tx buffer, or, if no new character is currently available for transmission, by issuing a Reset TBE Interrupt Pending command to WR0. This command removes the interrupt request and temporarily inhibits further TBE interrupts until the TX buffer is loaded and again becomes empty. A Quadart reset or POC results in a set TBE status bit (TBE status true), but no TBE interrupts are issued to the IOP until the Tx buffer is loaded and becomes empty for the first time. TBE status is polled from RR0 if control bit TBE Interrupts Enable is reset.

Five RR0 bits taken together form the EXSTAT bit group: Break/Abort, Tx Underrun/EOM, CTS, $\overline{\text{Sync}}$/Hunt and DCD. Certain bit transitions within this group generate EXSTAT interrupt requests to the IOP if control bit EXSTAT Interrupts Enable is set. The EXSTAT bit group also has its own unique interrupt vector if control bit Status Affects Vector is set. EXSTAT interrupt requests to the IOP may only be removed by issuing a Reset EXSTAT Interrupts command to WR0. See Section 4.5 for a further discussion of the EXSTAT bit group.

Finally note that all Quadart interrupt service routines must return to the background program with an RETI (not an RET) instruction. Internal SIO, CTC and PIO circuits detect the RETI opcode on the C-Bus, and use this event to update their IEI and IEO interrupt priority lines. See Sections 8.1 and 8.2 for further details.

4.3 RECEIVED DATA FIFO AND ERROR FIFO

This section explains how SIO received data FIFO and error (RXSPCL) FIFO work together. A thorough understanding of this topic is essential for efficient SIO programming.

Serial RxD data received by an SIO channel is shifted into the channel receiver, assembled to form a character, and loaded from the receiver into the bottom of a three deep received data FIFO. As the assembled character is loaded into the Rx FIFO, four RXSPCL bits are simultaneously loaded into a four wide, three deep error FIFO (see Appendix A for a detailed description of each RXSPCL bit). The four bits at the top of the error FIFO feed the four RR1 RXSPCL status bits as shown in

Figure 30 below. The function of individual RXSPCL bits varies from mode to mode as shown in Table 6. With the exception of CRC error status, the status loaded with each character applies to the character itself.

Table 6: RXSPCL CONDITIONS VERSUS MODE

RR1 Bit	Asynchronous Mode	Byte Synchronous Mode	SDLC Mode
D7	Not Used	Not Used	End Of Frame
D6	Framing Error	CRC Error	CRC Error
D5	Overrun Error	Overrun Error	Overrun Error
D4	Parity Error	Parity Error	Not Used

Three cases may arise after each new character is assembled: (1) if the received data FIFO is empty when a character is transferred into it from the receiver, then both the character and its associated RXSPCL bits rise side by side to the FIFO tops and become available for reading and for generating interrupts, (2) if one or two characters and their associated RXSPCL bits are stacked unread in the FIFOs, then a new character and its RXSPCL bits are loaded below those unread in the FIFOs, (3) if three characters and their RXSPCL bits are stacked unread in the FIFOs, then the new character overwrites the last character received on the bottom of the data FIFO, and at the same time, RXSPCL bit Rx Overrun Error is set and this status along with three RXSPCL status bits on the new character overwrite the four old RXSPCL status bits on the bottom of the error FIFO. These three cases are illustrated below in Figure 28.

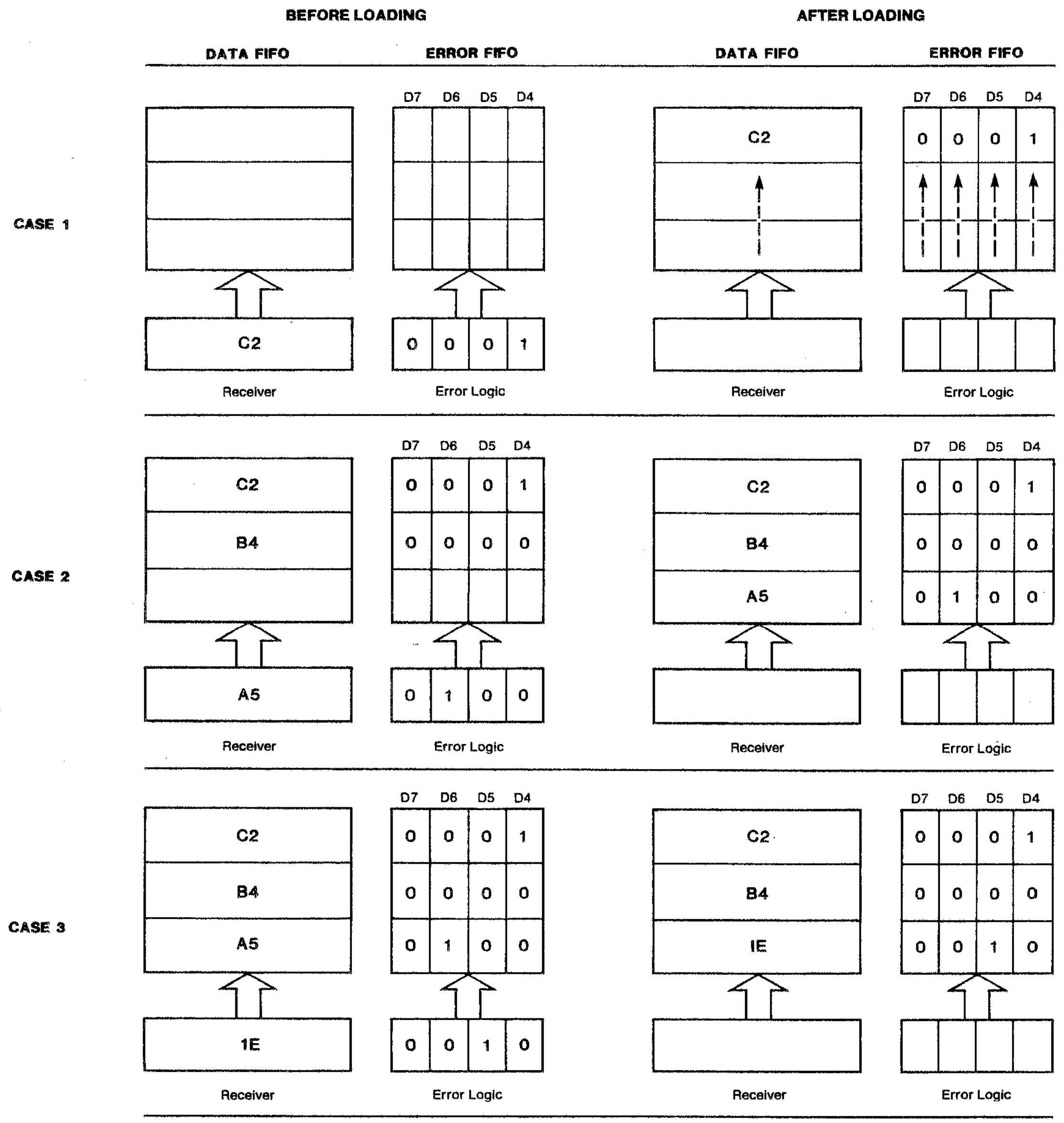
When a character rises to the top of the received data FIFO, RR0 status bit RCA is set and the character becomes available for reading from the channel SIO Data Port. As status bit RCA is set, the four RXSPCL bits simultaneously become valid and available from reading from RR1 (depending on the channel interrupt mode, these events may also cause interrupts -- see Section 4.2). Reading a character from the channel SIO Data Port resets status bit RCA, and if there are more unread characters in the Rx FIFO, all characters and RXSPCL bits move up one position in the FIFOs again setting status bit RCA. If there are no more data in the FIFOs, then reading the channel SIO Data Port and RR1 will return the previously read character and RXSPCL bits. Carefully note that **reading a character** from the data FIFO causes characters and RXSPCL bits to be removed and others to move up in both FIFOs, whereas reading RXSPCL bits from RR1 does **not** affect either FIFO's contents. Consequently, the RXSPCL bits must be read from RR1 **before** the character opposite it is read from the channel SIO Data Port. If this order is reversed, RXSPCL bits on the **next** character may be erroneously read instead. These two cases are illustrated in Figure 29.

Whenever a set Parity Error status bit, or a set Rx Overrun Error status bit reach the top of the error FIFO, the corresponding RR1 bits D4 and D5 are latched set (the receiver error detection logic does not check for parity, nor does it ever flag a parity error unless control bit Parity Enable is set, whereas Rx Overrun Errors are always flagged). These two bits remain latched set in RR1, even if characters without parity and overrun errors are read from the data FIFO, until a RXSPCL Reset command is issued to WR0 (see Figure 30). The RXSPCL Reset command not only resets RR1 status bits Parity Error and Rx Overrun Error, but it also resets RR1 status bits SDLC End Of Frame and CRC/Framing Error (reading RR1 after this command is issued always returns four logic 0 bits). Any pending RXSPCL interrupts (if Rx Interrupts are enabled) are also removed by the RXSPCL Reset command. Since the RXSPCL Reset command resets all four RXSPCL bits, RR1 must be read **before** issuing the command.

The interaction between the RXSPCL Reset command and the latching of status bits Parity Error and Rx Overrun Error in RR1 described above is idealized in Figure 30. Note that all RXSPCL bits in the error FIFO itself are unaffected by the RXSPCL Reset command.

4.4 CHANGING TX AND RX CHARACTER LENGTHS

Channel transmitter and receiver character lengths which are defined during channel initialization may be changed while the channel is operating **on the fly.** In all modes the transmitter samples the transmitter character length to be used on a character at the time the character is transferred from the Tx buffer to the transmitter itself. So changing the transmitter character length while character N is in the process of being shifted out has no affect on character N; the new transmitter character length takes affect on character N+1. In both polled and interrupt driven applications, the transmitter character length should be changed **only** when status bit TBE is set; otherwise there is uncertainty as to whether the change takes affect before or after the Tx buffer empties. For example, entering a TBE



D7=SDLC End of Frame
D6=CRC/Framing Error
D5=Rx Overrun Error
D4=Parity Error

Figure 28: SIO FIFO LOADING

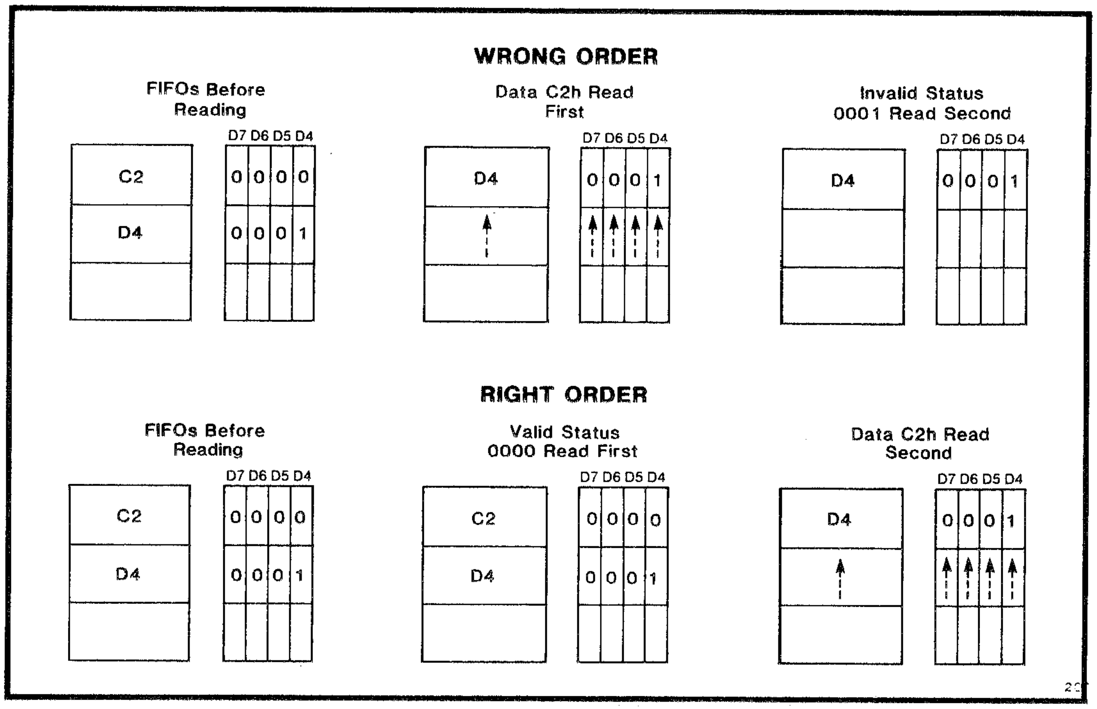


Figure 29: SIO FIFO READING

interrupt service routine implies that status bit TBE has become set as a result of character N being loaded into the transmitter. The transmitter character length may then safely be changed for character N+1, followed by character N+1 being loaded into the Tx buffer through an SIO Data port for sending with the new character length. Note that any character sent after a transmitter character length change is always properly formatted (including start bit, stop bits and parity, if applicable), and its bits are shifted out in the normal direction with no extraneous bits either added or deleted from the character.

The receiver character length may be changed on the fly in all modes. The new length applies to the character currently being assembled and all characters which follow it, provided the programmed length exceeds the number of character bits currently assembled. For example, if the receiver has assembled 3 character bits at the moment the receiver character length is changed to 6 bits, then the entire character is assembled with 6 data bits. If, on the other hand, 7 bits have been assembled at the moment the length is changed to 6 bits, then character assembly continues with the previous length until completed, and the new length then applies only to the following characters. Changing the receiver character length has no affect on characters and status already loaded into the received data and error FIFOs.

When the receiver character length is switched from 8 bits to 6 bits/character, this does **not** necessarily mean that the 6-bit character is assembled right justified with the LSB character bit at position D0 in the Rx FIFO. In fact, in the byte synchronous and SDLC modes, switching from 8 bits to 6 bits/character causes the receiver to assemble the next character by merely shifting in the following six data bits without right justification. In the asynchronous mode, however, data is always assembled right justified since character synchronization is reestablished on the start bit of each new character. The situation in the synchronous modes, and the method to right justify the read data, is best illustrated by an example.

Figure 31 shows three characters transmitted in either SDLC or byte synchronous mode. The first character, 00110101b, is sent as an 8-bit character, while the second and third characters, 101111b and 011100b, are sent as 6-bit characters. To the left in the figure, the receiver assembles the first character with an 8-bit length, and the second and third characters with 6-bit lengths (unison switching). Note that when the receiver is switched to 6 bits/character, the receiver merely shifts in the next six data bits to form the next character. This results in the character data bits appearing in positions D7 through D2 instead of the right justified D5 through D0 positions for the second, third, and all subsequently received characters.

This situation may be corrected by delaying the 6 bits/character switch one character, as shown at the right of the figure. Here, the **first** and the **second** characters are assembled with 8-bit lengths, and the switch to 6 bits/character is made on the third character (delayed switching). The second assembled character then consists of eight data bits; the six low order right justified bits of the second transmitted character (101111b), and two high order bits of the third character (00b in this example). When the switch to 6 bits/character is then made, the next incoming six bits are likewise shifted into right justified positions for reading from the FIFO, as are all subsequently 6-bit received characters. Changing the receiver character length in this fashion has no affect on the CRC check result in either the byte synchronous or SDLC modes.

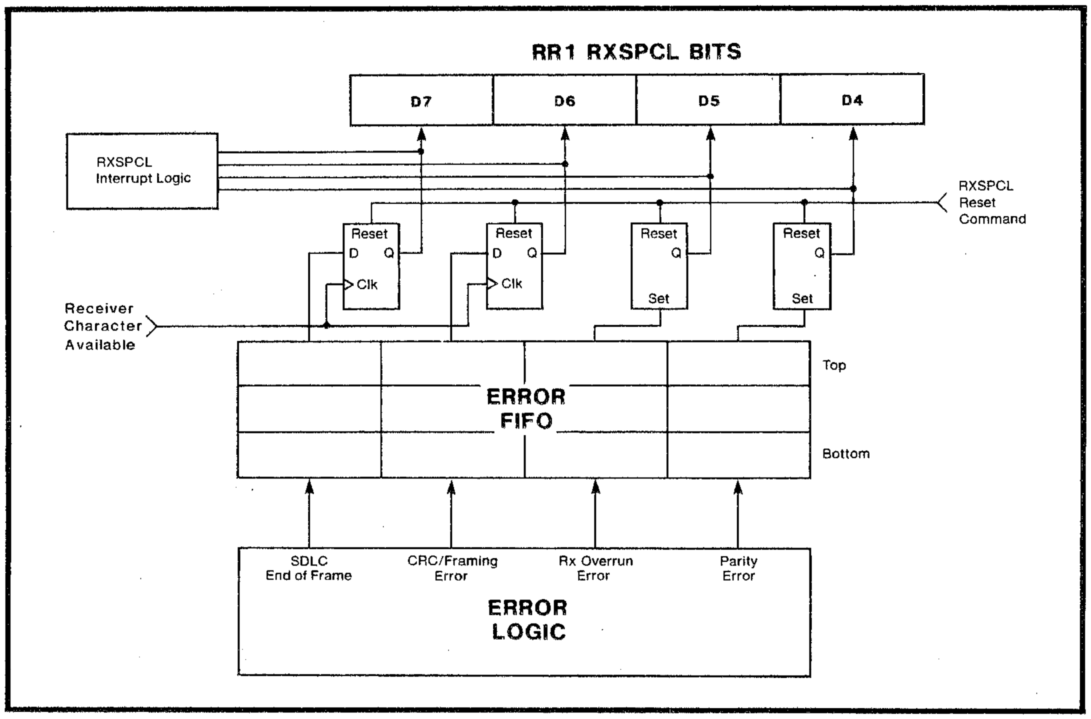


Figure 30: IDEALIZED RR1 AND ERROR FIFO BEHAVIOR

Finally note that the SDLC residue codes presented in Section 7.4 assume that the received data is read right justified; if the read data is not right justified, the residue data presented in the table of that section are invalid. As standard programming practice it is recommended that **all read data be right justified.**

To generalize: In the byte synchronous and SDLC modes, if receiver characters ... N-3, N-2, N-1 are X data bits long and characters N, N+1, N+2, ... are Y data bits long, and it is desired to receive all characters right justified, then the receiver data length should be set to X for characters ... N-3, N-2, N-1, N, and the length should be switched to Y on character N+1.

4.5 EXSTAT BIT GROUP

Five status bits taken together form the EXSTAT (External/Status) bit group, and these are: Break/Abort, Tx Underrun/EOM, CTS (Clear To Send), Sync/Hunt, and DCD (Data Carrier Detect). See Appendix

A for a detailed description of each bit function. These bits are available for reading from RR0. DCD and CTS report on channel modem line status and the remaining three report on internal SIO channel status.

Five events within the EXSTAT bit group may result in interrupt requests to the IOP if control bit EXSTAT Interrupts Enable is set: a change in modem input line DCD; a change in internal status bit Sync/Hunt; a change in modem input line CTS; a reset to set transition in internal status bit Tx Underrun/EOM; and a change in internal status bit Break/Abort. Note that a change in either direction of any of these five bits, except Tx Underrun/EOM, generates an EXSTAT interrupt. When one of these transitions occurs an EXSTAT interrupt is issued to the IOP and all five EXSTAT bits in RR0 are latched in the states which exist at the moment the interrupt occurs (the other three RR0 bits are not latched, however). This feature allows the IOP to capture DCD and CTS transitions lasting longer than 200 nSec which may change back again before being serviced. The EXSTAT service routine must remove the interrupt request by issuing a Reset EXSTAT Interrupts command to WR0 (it will not go away otherwise). This command removes the interrupt request and it frees the five EXSTAT bits to reflect current conditions (it does **not** reset them all to logic 0). Another EXSTAT interrupt is generated if one of the five EXSTAT bits changes state when freed.

If control bit EXSTAT Interrupts Enable is reset then the five EXSTAT bits are latched as before, but transitions within the bit group do not generate interrupts. In this context the Reset EXSTAT Interrupts command is used to free, or unlatch the EXSTAT bits, not to remove an EXSTAT interrupt.

The EXSTAT behavior just described then suggests the following approach to managing this bit group:

1. If control bit EXSTAT Interrupts Enable is reset, then the polling routine should issue a Reset EXSTAT Interrupts command immediately before reading RR0 to make the EXSTAT bit group reflect current conditions.

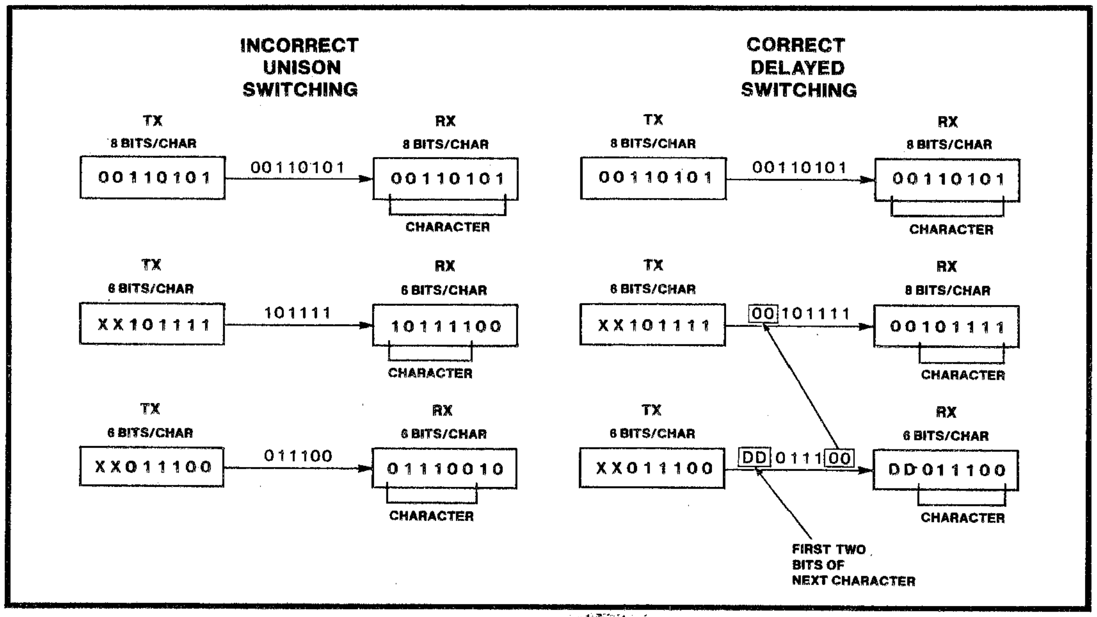


Figure 31: RIGHT JUSTIFYING READ DATA

2. If control bit EXSTAT Interrupts Enable is set, then the EXSTAT interrupt service routine should read RR0 before issuing the Reset EXSTAT Interrupts command to determine the interrupt source. The EXSTAT service routine must issue a Reset EXSTAT Interrupts command to remove the interrupt before returning to the background IOP program. The service routine must return to the background program with an RETI instruction -- see Section 8.1.

3. The Reset EXSTAT Interrupts command should be issued twice after control bit EXSTAT Interrupts Enable is set and before IOP interrupts are enabled during SIO channel initialization. This procedure unconditionally clears any spurious EXSTAT interrupts which might result from enabling this circuitry. If, for example, the SIO internal logic powers up with an EXSTAT interrupt pending and the EXSTAT bits latched, then issuing the first reset command might free certain bits to different states thereby generating another EXSTAT interrupt; this interrupt is then cleared by the second EXSTAT reset command.

Chapter 5

ASYNCHRONOUS SERIAL I/O

Any combination of Quadart channels may be programmed for half or full duplex asynchronous serial I/O operation; the remaining channels may then be programmed for synchronous operation, or left unused. The steps required to initialize an asynchronous Quadart channel are presented in Section 5.1. Sections 5.2 and 5.3 discuss asynchronous data transmission and reception, and Section 5.4 covers break generation and detection. The chapter concludes with a complete asynchronous interrupt driven programming example. The asynchronous data format is shown in Figure 32.

5.1 CHANNEL INITIALIZATION

After any POC or reset from the IOP: all Quadart interrupts (except pending PIO interrupts) are disabled, all CTC channels stop counting, all SIO transmitters and receivers are disabled, and modem output lines DTR and RTS are turned off (marking). Selected CTC and SIO channels may be forced to these states at any time by writing reset commands to these devices. The PIO controlled CY lines are in random states after a POC, and are unchanged after a reset from the IOP. From this state a Quadart channel is configured for asynchronous serial I/O by the following steps:

1. Disable IOP interrupts during Quadart initialization since spurious Quadart interrupts may occur as its interrupt circuitry is enabled. After the Quadart is properly initialized there should be no Quadart interrupts pending.

2. Initialize the PIO (see Section 2.2). Reset channel PIO bit ExtCk connecting a local CTC clock to the channel's SIO $\overline{TxC}$ and $\overline{RxC}$ inputs.

3. Initialize the channel CTC (see Section 2.3). Select mode and time constant to generate the desired asynchronous bit rate (see Section 3.2).

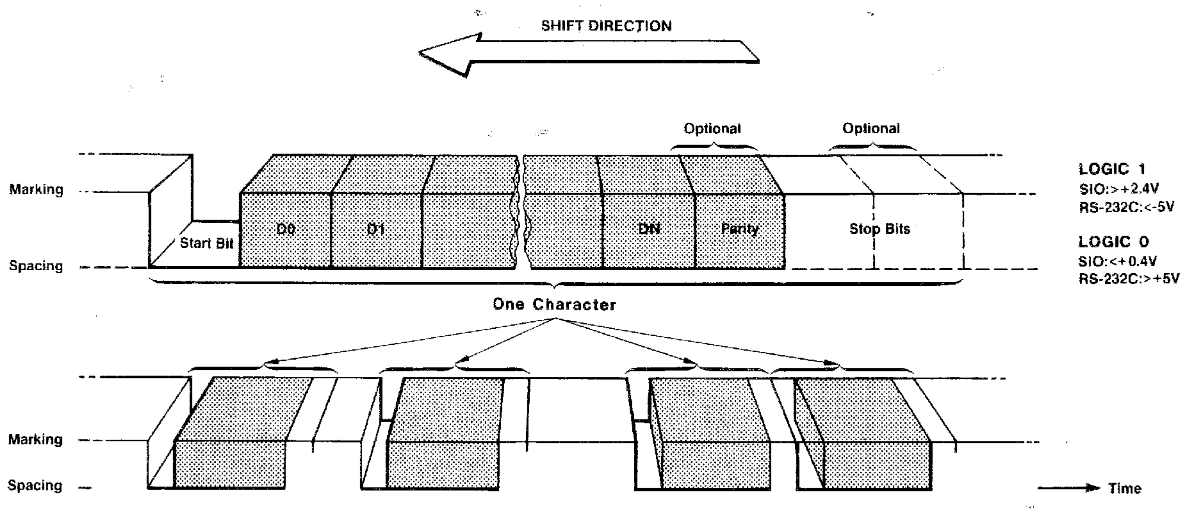


Figure 32: ASYNCHRONOUS MODE DATA FORMAT

4. Initialize the channel SIO by outputting control bytes to WR0 through WR5. See Figure 33 for asynchronous configuration options, and Appendix A for detailed bit descriptions. Register WR4, which defines the channel mode, must be programmed before registers WR1, WR3 and WR5, since these registers are used differently in each SIO mode.

5. If other Quadart channels are to be used, initialize them; if not, Quadart initialization is complete since all other channels are disabled.

At this point the asynchronous channel is ready for operation. If interrupts are used, the IOP must load the IOP Z80A I-register, select IM2 mode, and enable IOP interrupts (execute an EI instruction). The IOP sends data characters by writing bytes to the channel SIO Data Port when status bit TBE is set; this status is determined by polling RR0, or by a TBE interrupt (the first TBE interrupt does not occur until the IOP has loaded a character into the Tx buffer and the buffer then becomes empty). The IOP reads data characters by reading bytes from the channel SIO Data Port when status bit RCA is set; this status is determined by polling RR0, or by an RCA interrupt -- see Section 4.2.

The recommended order for loading SIO registers is: (1) WR4 to define the asynchronous mode, (2) WR1 to configure the channel interrupt behavior, (3) WR2 to define the base interrupt vector if any SIO channel interrupt is enabled, (4) WR3 to configure and enable the channel receiver, and (5) WR5 to configure and enable the channel transmitter. Registers WR6 and WR7 are not used in the asynchronous mode.

The complete range of SIO asynchronous mode configuration options is illustrated in Figure 33. Note that some WR1 through WR5 bits select among available options, while other **don't care** control bits are conservatively programmed reset (function disabled).

A Channel Reset command should be issued to WR0 before loading SIO registers WR1 through WR5. The Reset EXSTAT Interrupts command should be issued twice after enabling EXSTAT interrupts if control bit EXSTAT Interrupts Enable is set, as explained in Section 4.5. Configuring a channel for asynchronous operation automatically defines the EXSTAT and RXSPCL bit group functions as shown in Table 7. The Reset Channel command resets all four RXSPCL bits, and the two Reset EXSTAT Interrupts commands makes the EXSTAT bit group reflect current status.

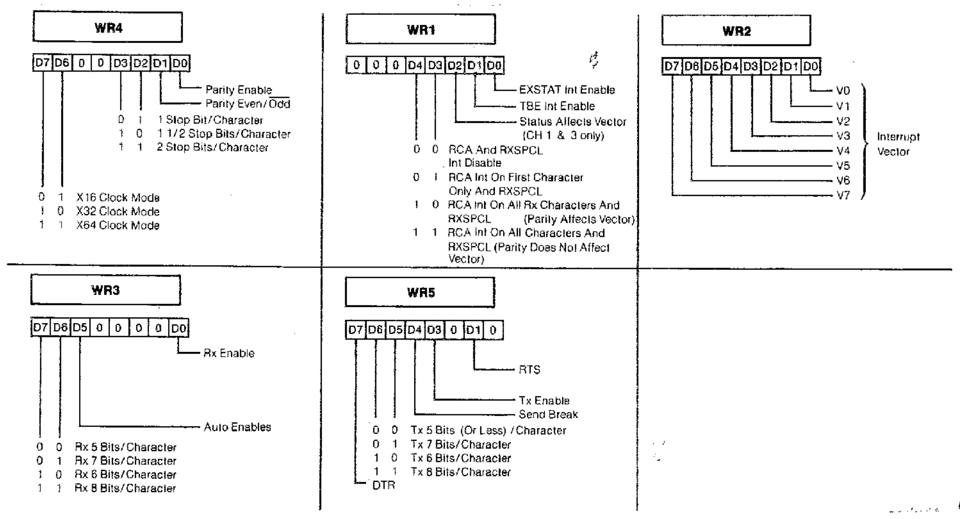


Figure 33: SIO ASYNCHRONOUS MODE OPTIONS

Table 7: ASYNCHRONOUS RR0 AND RR1 BIT FUNCTIONS

Bit	RR0	Bit	RR1
D7	Break	D7	Logic 0 (not used)
D6	Logic 1 (not used)	D6	Framing Error
D5	CTS	D5	Rx Overrun Error
D4	Logic 0 (not used)	D4	Parity Error
D3	DCD	D3	Logic 1 (not used)
D2	TBE	D2	Logic 1 (not used)
D1	Interrupt Pending	D1	Logic 1 (not used)
D0	RCA	D0	All Sent

Example

The following Z80 program initializes Quadart channel 1 for asynchronous full duplex interrupt driven operation at 300 bits per second with a X16 clock multiplier. The routine begins by configuring the PIO to connect the channel 0 CTC zero count output to the SIO channel 1 $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clock inputs. It then programs the CTC channel 1 zero count output pin to generate a 4,800 Hz pulse train.

An OTIR block move instruction then programs channel 0 for asynchronous operation, a X16 clock multiplier, 1 stop bit, and odd parity for both the receiver and the transmitter. TBE Interrupts, RCA Interrupts On All Characters (Parity Affects Vector), and EXSTAT Interrupts are enabled, and Status Affects Vector mode is selected which causes a variable interrupt vector, rather than a fixed one, to be supplied during interrupt acknowledge from the IOP. The receiver character length is set to 7 bits, Auto Enables are disabled so that channel 0 DCD and CTS modem input lines then have no affect on receiver and transmitter enabling and disabling, and the channel receiver is enabled. Finally, the transmitter character length is set to 7 bits and the transmitter is enabled. SIO initialization begins with a channel 0 reset and two Reset EXSTAT Interrupts commands are issued after EXSTAT interrupts are enabled. The program, written as subroutine INIT.1, is referenced as an external call in an asynchronous example program that appears at the end of this chapter.

```
CROMEMCO Z80 Macro Assembler version 03.07

                    ;       ASYNCHRONOUS INITIALIZATION ROUTINE

                    ;  See manual text for program description.

      (0040)        QBASE   EQU     40h             ;Assumed Quadart Base Address.
      (0043)        SCMST1  EQU     QBASE+03h       ;Ch 1 SIO Command/Status Port.
      (0048)        PDAT01  EQU     QBASE+08h       ;Ch 0&1 PIO Data Port.
      (0049)        PCOM01  EQU     QBASE+09h       ;Ch 0&1 PIO Command Port.
      (004D)        CTC1    EQU     QBASE+0Dh       ;Ch 1 CTC Port.

                            ENTRY   INIT.1

                            ;======================================================
                            ; Initialize Quadart
                            ; Channel 1
                            ;======================================================

0000' 3ECF          INIT.1: LD      A,11001111b     ;Ch 0&1 PIO to
0002' D349                  OUT     (PCOM01),A      ;control mode 3.
0004' 3ECC                  LD      A,11001100b     ;Define I/O lines;
0006' D349                  OUT     (PCOM01),A      ;0=output; 1=input.
0008' 3E07                  LD      A,00000111b     ;Disable PIO interrupts.
000A' D349                  OUT     (PCOM01),A      ;
000C' 3E00                  LD      A,00000000b     ;Reset ExtCk0 to connect
000E' D348                  OUT     (PDAT01),A      ;CTC to SIO Ch 1 TxC and RxC.

0010' 3E47                  LD      A,01000111b     ;Ch 1 CTC to counter mode,
0012' D34D                  OUT     (CTC1),A        ;interrupts disabled.
0014' 3E40                  LD      A,64            ;Time constant = 64 for 300
0016' D34D                  OUT     (CTC1),A        ;baud with X16 multiplier.

0018' 212200'               LD      HL,SIOBEG       ;Point to Ch 1 initialization
001B' 0E43                  LD      C,SCMST1        ;data for OTIR block move.
001D' 060B                  LD      B,SIOEND-SIOBEG ;
001F' EDB3                  OTIR                    ;Initialize SIO Ch 1.

0021' C9                    RET                     ;


                            ;======================
                            ; SIO Block Move Ch 1
                            ; Initialization Data
                            ;======================

0022' 18            SIOBEG: DB      00011000b       ;Reset Channel 1 Command.

0023' 04                    DB      00000100b       ;Point to WR4.
0024' 45                    DB      01000101b       ; - X16 clock
                                                    ; - asynchronous mode
                                                    ; - 1 stop bit
                                                    ; - enable odd parity

0025' 01                    DB      00000001b       ;Point to WR1.
0026' 17                    DB      00010111b       ; - RCA Int On All Chartacters
                                                    ;   (parity affects vector)
                                                    ; - status affects vector
                                                    ; - TBE interrupts enable
                                                    ; - EXSTAT interrupts enable

0027' 02                    DB      00000010b       ;Point to WR2, define Interrupt
0028' 00                    DB      00000000b       ;Vector.  Resulting IM2
                                                    ;indirect jump table assuming
                                                    ;Reg. I loaded with 44h:
                                                    ;4400h -> Ch 1 TBE
                                                    ;4402h -> Ch 1 EXSTAT
                                                    ;4404h -> Ch 1 RCA
                                                    ;4406h -> Ch 1 RXSPCL

0029' 13                    DB      00010011b       ;Point to WR3, reset EXSTAT interrupts
002A' 41                    DB      01000001b       ; - 7 bit Rx characters
                                                    ; - no auto enables
                                                    ; - enable receiver

002B' 15                    DB      00010101b       ;Point to WR5, reset EXSTAT interrupts
002C' AA                    DB      10101010b       ; - DTR on
                                                    ; - 7 bit Tx characters
                                                    ; - break off
                                                    ; - enable Tx
                                                    ; - RTS on

                    SIOEND:

002D' (0000')               END     INIT.1
```

.2 ASYNCHRONOUS TRANSMIT

After the channel transmitter is initialized and enabled, but before any characters are loaded for transmission, modem line TxD is held marking (approximately -10 VDC) and status bit TBE is set. If control bit TBE Interrupts Enable is set, the first TBE interrupt does not occur until a character is loaded into the Tx buffer and it then becomes empty. Outputting a character to the channel SIO Data Port loads it into a Tx buffer where it is held until the transmitter itself is empty (the transmitter is singly buffered). The character in the Tx buffer is loaded into the transmitter as soon as the transmitter is empty. Then, if enabled, Status bit TBE changes from reset to set generating a TBE interrupt.

If status bit TBE changes from reset to set and the IOP has no further data to send, then the pending TBE interrupt may be removed by issuing a Reset TBE Interrupt Pending command to WR0. Another TBE Interrupt does not occur until the Tx buffer is loaded and again becomes empty. As the transmitter itself becomes empty the TxD line reverts to a marking state, and status bit All Sent is set after the last stop bit clears the transmitter (status bit All Sent does not generate any type of interrupt, and it is active in the asynchronous mode only).

The transmitter automatically adds a start bit, a parity bit (if control bit Parity Enable is set), and the programmed number of stop bits to any data character loaded by the IOP. The bits shifted out are: start bit, D0 (LSB), D1, ... , DN (MSB), parity (if enabled), stop bit(s), in that order. Modem line TxD is spacing for the start bit, spacing for logic 0 data/parity bits, marking for logic 1 data/parity bits, and <u>spacing</u> for stop bits. The bits are shifted out on the $\overline{\text{TxC}}$ falling edge at either 1/16th, 1/32nd, or 1/64th the $\overline{\text{TxC}}$ input rate, depending on the programmed multiplier.

If the programmed Tx character length is 6 or 7 bits, then the character bits must be loaded into the channel SIO Data Port right justified (MSB bits are irrelevant). If the programmed Tx character length is 5 bits or less, then the unused high order bits must be formatted in a special way. The required format is detailed in Appendix A under the WR5 bit descriptions. Programming a new Tx character length does not affect a character in the process of being shifted out of the transmitter. The newly programmed character length applies to all following characters since the Tx character length is sampled as a character is loaded from the Tx buffer into the transmitter.

If control bits Tx Enable and Auto Enable are set then modem line CTS functions as a transmitter enable circuit. In this mode a single character may be loaded into the transmitter while modem line CTS is off which forces status bit TBE set. When modem line CTS later turns on (spacing), the transmitter is enabled, status bit TBE is reset, and the character is properly sent. If the transmitter is disabled either by line CTS or by resetting control bit Tx Enable, the character currently being shifted out is sent completely, but the TxD line goes to a marking state after the character is sent while the transmitter remains disabled. Any character in the Tx buffer when the transmitter is disabled remains in the Tx buffer with status bit TBE reset. Any character held in the Tx buffer is properly transmitted when the transmitter is subsequently reenabled.

5.3 ASYNCHRONOUS RECEIVE

Just after the channel receiver has been reset, initialized and enabled, but before any characters have been received, the Rx FIFO is empty (random data), status bit RCA is reset, all RXSPCL bits are reset, and the EXSTAT bit group reflects current status.

The enabled receiver begins character assembly as the first spacing start bit arrives. If this bit is still spacing at its midpoint, then the remaining character bits are sampled at their midpoints; D0 (LSB), D1, ... , DN (MSB), parity (if enabled), and stop bit(s), in that order. After the character is completely assembled it is transferred to the Rx FIFO for reading from the channel SIO Data Port, and status bit RCA is set. Eight bits are always read from this port. If the programmed Rx character length is 8 bits, the byte consists of bits D7 (MSB) on the left through D0 (LSB) on the right (the parity bit is not included in the read data). If the programmed character length is 7 or less, then the extra high order bits to the left are set when assembled, followed by the parity bit (if enabled), followed by DN (MSB) through D0 (LSB) to the right.

As each character is assembled and loaded into the Rx FIFO, three RXSPCL bits are parallel loaded into the error FIFO: Framing Error, Rx Overrun Error, and Parity Error. If control bit Parity Enable is set and the character is assembled with incorrect parity, then a set Parity Error bit is loaded; otherwise a reset Parity Error bit is loaded. If a spacing condition is sampled where the receiver expects to see a stop bit, a set Framing Error bit is loaded; otherwise a reset Framing

Error bit is loaded. If three characters are stacked unread in the received data FIFO as a fourth character is transferred from the receiver to the FIFO, then the character on the bottom of the data FIFO is overwritten and set status bit Rx Overrun Error is loaded into the bottom of the error FIFO opposite the fourth character on the bottom of the data FIFO. The fourth character's Framing and Parity status overwrite the third character's status on the bottom of the error FIFO also. Received data which does not overwrite the bottom FIFO character in this way causes a reset Rx Overrun Error bit to be loaded into the error FIFO. Refer to Section 4.3 for more information on the data and error FIFOs.

The channel receiver may be polled or interrupt driven. If the RCA and RXSPCL Interrupts Disabled mode is selected, then RCA status is polled from RR0. RCA conditions generate IOP interrupts with a unique interrupt vector if mode RCA Interrupt On First Character, or mode RCA Interrupts On All Characters (Parity Does/Parity Does Not Affect Vector) is enabled. See Section 4.2 and Appendix A for a description of these modes.

If control bit EXSTAT Interrupts Enable is set, then a change (in either direction) in any of the four active EXSTAT bits (Break, Tx Underrun, CTS and DCD) causes an EXSTAT interrupt which latches the four EXSTAT bits. The EXSTAT interrupt must be removed by a Reset EXSTAT Interrupts command issued to WR0. This also frees (unlatches) the four bits. If control bit EXSTAT Interrupts Enable is reset, then the four EXSTAT bits are still latched in RR0 and freed with the Reset EXSTAT Interrupts command, but no EXSTAT interrupts are generated. See Section 4.5 for further details on the EXSTAT bit group.

If control bits Rx Enable and Auto Enable are set, then modem line DCD functions as a receiver enable. If the receiver is disabled either by line DCD or by resetting control bit Rx Enable, then any character currently being assembled is destroyed and no RCA interrupt occurs. The IOP may be alerted that DCD has disabled the channel receiver by enabling EXSTAT interrupts and sampling DCD status in the service routine. The channel receiver operates normally as soon as the DCD line turns on (spacing). Any characters or RXSPCL status in the FIFOs may still be read while the receiver is disabled since disabling the channel receiver does not alter either FIFO.

5.4 BREAK GENERATION AND DETECTION

A break sequence is generated by setting control bit Send Break. This immediately forces the channel TxD line spacing, and the spacing condition persists until the Send Break bit is reset. The Send Break bit only affects the transmitter TxD output line; the transmitter continues to operate normally in all other respects. For instance, if Send Break is set momentarily and then reset while a character is being shifted out, then the TxD line is forced spacing only momentarily and the rest of the character is transmitted correctly. The normal procedure for sending a timed break is: (1) wait until all characters clear the channel transmitter by monitoring status bit All Sent, (2) set control bit Send Break, (3) time the break interval by loading a dummy character into the channel SIO Data Port and monitoring status bit All Sent for a break interval of one character transmit time, (4) repeat the previous step repeatedly for longer break intervals, and (5) reset control bit Send Break to terminate the break sequence.

A break sequence is detected by the channel receiver when a null character with a framing error is received which sets status bit Break. This implies that the transmitter must hold the spacing condition for at least a full character transmit time. A single extraneous null (00h) character is placed in the Rx FIFO (regardless of break duration), but status bit RCA is not set until the RxD line starts marking. If EXSTAT interrupts are enabled, then a change (in either direction) in status bit Break causes an EXSTAT interrupt, and the other two active EXSTAT bits are then latched in the states which existed at the time of the interrupt. The EXSTAT bits remain latched until a Reset EXSTAT Interrupts command is issued to WR0. This command frees the EXSTAT bits to reflect current status, and another EXTSTAT interrupt is generated if one of the three bits changes state when freed with the command. The EXSTAT interrupt service routine should follow this outline when servicing break generated interrupt: (1) the EXSTAT service routine reads RR0 and determines that a break caused the interrupt, (2) a Reset EXSTAT Interrupts command is issued to WR0 so break termination can be detected, (3) a reset to set transition in status bit Break occurs signifying the break is terminated, and another EXSTAT interrupt results, (4) the service routine reads and discards a single extraneous null character to remove the RCA interrupt generated by this character, (5) the service routine again issues a Reset EXSTAT Interrupts command so the next change in the bit group can be detected, and (6) the host system is notified that a break has occurred so appropriate action can be taken.

5.5 AN ASYNCHRONOUS PROGRAMMING EXAMPLE

This section presents a complete interrupt driven asynchronous program. The program resides in IOP RAM memory starting at address 4000h (the start of IOP RAM memory), and the interrupt service routines start at 4400h (the IOP Z80A I-page is set to 44h). In overview, the program receives serial asynchronous data over Quadart channel 1, and stores each receiver character in a RAM buffer until a Line Feed character is received. Once it is received, the program transmits the entire buffer (with the Line Feed) over Quadart channel 1. The duration between each receiver character may vary, but the entire buffer is transmitted as a block (each RAM buffer character is loaded as soon as the Tx buffer empties). The program then functions as a simple data compaction routine. The principles illustrated here can be easily extended to two or more incoming data streams and buffers, and a more complex transmit routine which concentrates the multiplexed data onto a single line.

A more detailed program description follows:

1. Program execution begins at address 4000h, the starting address of the background program. The background program initializes Quadart channel 1 (external routine INIT.1 is called -- see Section 5.1), initializes some program variables, selects I-Page = 44h, enables IOP interrupts, and then enters a **jump to itself** loop. From this point on all data movement is interrupt driven.

2. An RCA interrupt causes the service routine to first check RAM flag ABTFLG (abort flag). A set flag signifies an RXSPCL (error) interrupt has occurred; a reset flag signifies no receive error has occurred. If the flag is set the RCA service routine reads and discards all receiver characters except a Line Feed. When the ending Line Feed is received then message **Abort <CR> <LF>** is transmitted. Each character received with ABTFLG reset is stored in RAM buffer RXBUFF. In this case, the entire buffer is transmitted over channel 1 when a Line Feed is received. Transmissions of both the abort message and the RXBUFF contents are initiated in the RCA routine by positioning the RAM pointer to the beginning of the message, loading the first message character into the Tx buffer, and incrementing the pointer. This enables TBE interrupts, and the TBE service routine then transmits the message completely.

3. TBE interrupts cause the service routine to get an RXBUFF character, load it into the Tx buffer for transmission, and increment the current RAM pointer value. If the character is a Line Feed character, the routine waits until TBE is again true, and then issues a Reset TBE Interrupt Pending command to WR0. This disables TBE interrupts until the transmitter buffer is again loaded in the RCA service routine. The TBE routine then enables the channel 1 receiver and resets the RAM buffer pointer.

4. The EXSTAT interrupt service routine checks for break start/termination. Message **Abort <CR> <LF>** is transmitted over channel 1 when the break sequence terminates. This is done by loading the first abort message character into the transmitter buffer which initiates TBE interrupts. The TBE service routine then transmits the message and reinitializes the channel 1 receiver and the RAM buffer for the next message.

5. The RXSPCL interrupt service routine (which is entered in the event of a Parity, Rx Overrun, or Framing error) sets RAM flag ABTFLG. When a Line Feed is received, this flag alters the RCA service routine to send an abort message instead of the contents of the buffer.

```
CROMEMCO Z80 Macro Assembler version 03.07

                  ;         ASYNCHRONOUS PROGRAMMING EXAMPLE

                  ;  See manual text for program description.

        (0040)    QBASE     EQU     40h                ;Assumed Quadart Base Address.
        (0042)    SDATA1    EQU     QBASE+02h          ;Ch 1 SIO Data Port.
        (0043)    SCMST1    EQU     QBASE+03h          ;Ch 1 SIO Command/Status Port.
        (0048)    PDAT01    EQU     QBASE+08h          ;Ch 0&1 PIO Data Port.
        (0049)    PCOM01    EQU     QBASE+09h          ;Ch 0&1 PIO Command Port.
        (004D)    CTC1      EQU     QBASE+0Dh          ;Ch 1 CTC Port.
        (000D)    CR        EQU     0Dh                ;ASCII Carriage Return.
        (000A)    LF        EQU     0Ah                ;ASCII Line Feed.

                            EXTRN   INIT.1             ;See Section 5.1.

        (4000)              ORG      4000h              ;Start of IOP ram.

                            ;============================================
                            ;  Initialization Routine
                            ;============================================

4000    310060    ASYNC:    LD      SP,6000h           ;Locate stack.
4003    3E00                LD      A,0                ;
4005    32A944              LD      (ABTFLG),A         ;Set ram flag ABORT.
4008    32A844              LD      (BREAK),A          ;Set ram flag BREAK.

400B    CD0000#             CALL    INIT.1             ;Initialize Quadart Ch. 1.

400E    21B144              LD      HL,RXBUFF          ;HL points to Rxbuffer.
4011    3E44                LD      A,44h              ;Select I-Page.
4013    ED47                LD      I,A                ;
4015    ED5E                IM      2                  ;Select Interrupt Mode 2.
4017    FB                  EI                         ;Enable IOP interrupts.
4018    18FE                JR      $                  ;Loop -- wait for interrupts.


                            ;==============================================
                            ;  Interrupt Service Routines
                            ;==============================================

        (4400)              ORG     4400h              ;I-Page.

                            ;======================
                            ; IM2 Indirect Jump Table
                            ;======================

4400    4D44                DW      TBE1               ;Ch 1 TBE service.
4402    7844                DW      EXSTAT1            ;Ch 1 EXSTAT service.
4404    1644                DW      RCA1               ;Ch 1 RCA service.
4406    0844                DW      RXSPCL1            ;Ch 1 RXSPCL service.

                  ;         Service routines appear in priority order (highest first).

                            ;======================
                            ;  RXSPCL1 Service
                            ;======================

4408    F5        RXSPCL1:PUSH      AF                 ;Save AF on stack.

4409    3E30                LD      A,00110000b        ;Reset Error Command
440B    D343                OUT     (SCMST1),A         ;to WR0.

440D    3EFF                LD      A,-1               ;Set ram flag ABTFLG.
440F    32A944              LD      (ABTFLG),A         ;

4412    F1                  POP     AF                 ;Restore AF.
4413    FB                  EI                         ;Enable IOP interrupts.
4414    ED4D                RETI                       ;Return from interrupt.

                            ;======================
                            ; RCA1 Service
                            ;======================

4416    F5        RCA1:     PUSH    AF                 ;Save registers on stack.
4417    C5                  PUSH    BC                 ;

4418    DB42                IN      A,(SDATA1)         ;Read character.
441A    E67F                AND     01111111b          ;Strip parity MSB.
441C    47                  LD      B,A                ;Store it in reg. B.
```

```
441D  3AA944             LD      A,(ABTFLG)        ;Is ram abort flag set?
4420  3C                 INC     A                 ;
4421  200E               JR      NZ,NOTABT         ;
4423  3E0A               LD      A,LF              ;Abort flag is set.
4425  B8                 CP      B                 ;Is character = LF ?
4426  2020               JR      NZ,EXIT1          ;No => exit.
4428  21AA44             LD      HL,ABTMSG         ;Yes => send abort message.
442B  7E                 LD      A,(HL)            ;Load first abort message
442C  D342               OUT     (SDATA1),A        ;character into Tx buffer.
442E  23                 INC     HL                ;Since TBE interrupts enabled,
442F  1817               JR      EXIT1             ;transmitter will now send
                                                   ;message until LF encountered.

4431  78        NOTABT:  LD      A,B               ;Abort flag not set.
4432  77                 LD      (HL),A            ;Stash character in
4433  23                 INC     HL                ;Rxbuffer and move pointer.

4434  3E0A               LD      A,LF              ;Is character = LF ?
4436  B8                 CP      B                 ;
4437  200F               JR      NZ,EXIT1          ;No => exit.

4439  3E03               LD      A,3               ;Yes => end of message -- point to
443B  D343               OUT     (SCMST1),A        ;channel 1 WR3 and disable
443D  3E00               LD      A,0               ;receiver.
443F  D343               OUT     (SCMST1),A        ;

4441  21B144             LD      HL,RXBUFF         ;Load first Rxbuff character into
4444  7E                 LD      A,(HL)            ;transmitter buffer, then bump
4445  D342               OUT     (SDATA1),A        ;pointer.  Since TBE interrupts are
4447  23                 INC     HL                ;enabled, transmitter will now send
                                                   ;Rxbuffer contents until LF encountered.

4448  C1        EXIT1:   POP     BC                ;Restore registers.
4449  F1                 POP     AF                ;
444A  FB                 EI                        ;Enable IOP interrupts.
444B  ED4D               RETI                      ;Return from interrupt.

                         ;=======================
                         ;  TBE1 Service
                         ;=======================

444D  F5        TBE1:    PUSH    AF                ;Save registers on stack.

444E  7E                 LD      A,(HL)            ;Get a character and
444F  D342               OUT     (SDATA1),A        ;send it, then bump the
4451  23                 INC     HL                ;data pointer.

4452  FE0A               CP      LF                ;Is character = LF ?
4454  201E               JR      NZ,EXIT2          ;No => exit.

4456  3E01      SENT?:   LD      A,1               ;Yes => poll channel 1 RR1 until
4458  D343               OUT     (SCMST1),A        ;bit All Send is true,
445A  DB43               IN      A,(SCMST1)        ;
445C  CB47               BIT     0,A               ;
445E  28F6               JR      Z,SENT?           ;
4460  3E28               LD      A,00101000b       ;then issue Reset TBE Interrupts
4462  D343               OUT     (SCMST1),A        ;command to WR0.

4464  3E03               LD      A,3               ;Point to channel 1 WR3
4466  D343               OUT     (SCMST1),A        ;and reenable receiver.
4468  3E41               LD      A,01000001b       ;
446A  D343               OUT     (SCMST1),A        ;
446C  21B144             LD      HL,RXBUFF         ;Reset Rxbuffer pointer.
446F  3E00               LD      A,0               ;Reset ram ABTFLG.
4471  32A944             LD      (ABTFLG),A        ;

4474  F1        EXIT2:   POP     AF                ;Restore registers.
4475  FB                 EI                        ;Enable IOP interrupts.
4476  ED4D               RETI                      ;Return from interrupt.

                         ;=======================
                         ;  EXSTAT1 Service
                         ;=======================

4478  F5        EXSTAT1:PUSH     AF                ;Save regsiters on stack.
4479  C5                 PUSH    BC                ;

447A  3E00               LD      A,0               ;Read channel 1 RR0.
447C  D343               OUT     (SCMST1),A        ;
447E  DB43               IN      A,(SCMST1)        ;
```

```
4480 CB7F                BIT    7,A              ;Is status bit Break set?
4482 2807                JR     Z,BRK0           ;No => test for break termination.

4484 3EFF       BRK1:    LD     A,-1             ;Yes => set ram flag BREAK
4486 32A844              LD     (BREAK),A        ;and exit.
4489 1814                JR     EXIT3            ;

448B 3AA844     BRK0:    LD     A,(BREAK)        ;RR0 bit Break is reset.
448E 3C                  INC    A                ;Is ram flag BREAK set?
448F 200E                JR     NZ,EXIT3         ;No => exit.

4491 3E00                LD     A,0              ;Yes => Break termination.
4493 32A844              LD     (BREAK),A        ;Clear ram flag BREAK, then
4496 DB42                IN     A,(SDATA1)       ;read and discard null character from
4498 21AA44              LD     HL,ABTMSG        ;FIFO.  Load transmitter buffer
449B 7E                  LD     A,(HL)           ;with first abort messg character.
449C D342                OUT    (SDATA1),A       ;Since TBE interrupts are enabled,
449E 23                  INC    HL               ;the transmitter will now send the
                                                 ;abort message until LF encountered.

449F 3E10       EXIT3:   LD     A,00010000b      ;Reset EXSTAT Interrupts
44A1 D343                OUT    (SCMST1),A       ;command to WR0.

44A3 C1                  POP    BC               ;Restore registers.
44A4 F1                  POP    AF               ;
44A5 FB                  EI                      ;Enable IOP interrupts.
44A6 ED4D                RETI                    ;Return from interrupt.

                         ;========================
                         ;  Flags, Abort Message
                         ;  and Rxbuffer area.
                         ;========================

44A8 (0001)     BREAK:   DS     1                ;Ram BREAK flag.
44A9 (0001)     ABTFLG:  DS     1                ;Ram ABORT flag.
44AA 41626F72   ABTMSG:  DB     'Abort',CR,LF    ;Abort message.
44B1 (0100)     RXBUFF:  DS     100h             ;Rxbuffer area.

45B1 (4000)              END    ASYNC
```

Chapter 6

BYTE SYNCHRONOUS SERIAL I/O

Any combination of Quadart channels may be programmed for byte synchronous serial I/O operation; the remaining channels may be programmed for asynchronous or SDLC operation, or left unused. The steps required to initialize a Quadart channel for byte synchronous operation are presented in Section 6.1. Section 6.2 discusses byte synchronous transmission and CRC generation, and Section 6.3 covers reception and CRC checking. The chapter concludes with a complete byte synchronous programming example.

In the byte synchronous mode the programmer selects an 8-bit or 16-bit sync character when the SIO channel is initialized with control bits written to register WR4. These options are referred to as Monosync and Bisync in the Zilog and Mostek documentation. When using 8-bit sync characters, SIO register WR6 is loaded with the sync character used by the channel transmitter and WR7 is loaded with the sync character used by the channel receiver. When using a 16-bit sync character, the sync character value is loaded into registers WR6 (low order, sent/received first) and WR7 (high order, sent/received second). The same character is used by both the channel transmitter and receiver. The programmed sync character is repeatedly sent by the channel transmitter as message preamble and when the transmitter is idling with no data to send. After initialization the channel receiver defaults to the hunt mode: it begins assembling and loading characters into the received data FIFO only after one or more programmed sync character (either 8-bit or 16-bit) are received.

A X1 clock multiplier **must** be selected in the byte synchronous mode. Typically an attached modem generates the $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clock signals input to the SIO in byte synchronous mode. Each TxD data bit is shifted out of the channel transmitter on a $\overline{\text{TxC}}$ falling edge, and each RxD data bit is sampled on an $\overline{\text{RxC}}$ rising edge.

The byte synchronous message format is shown in Figure 34. This waveform format is generated by the channel SIO transmitter at its $\overline{\text{TxD}}$ output pin, and it is expected by the channel SIO receiver at its $\overline{\text{RxD}}$ input pin. A byte synchronous message consists of at least one sync character followed by one or more control and data characters. Individual characters consist of one to eight bits, with or without parity. The SIO transmitter permits the programmer to send the current

16-bit CRC block check value at any time, but commonly used protocols requires that CRC, or CRC followed by pad characters, appear only at the end of intermediate blocks and messages.

6.1 CHANNEL INITIALIZATION

After any POC or reset from the IOP all Quadart interrupts (except pending PIO interrupts) are disabled, all CTC channels stop counting, all SIO transmitters and receivers are disabled, and modem output lines DTR and RTS are turned off (marking). Selected CTC and SIO channels may be forced to these states at any time by writing reset commands to these devices. The PIO controlled CY lines are in random states after a POC, and are unchanged by a reset from the IOP. From this state, a Quadart channel is configured for byte synchronous I/O by the following steps:

1. Disable IOP interrupts during Quadart initialization since spurious Quadart interrupts may occur as its interrupt circuitry is enabled. After the channel is properly initialized there should be no pending Quadart interrupts.

2. Initialize the PIO (see Section 2.2). Set channel PIO bit ExtCk connecting modem supplied TxC and RxC clocks to the channel's SIO clock inputs.

3. Initialize the channel SIO by outputting control bytes to WR0 through WR7. See Figure 35 for the byte synchronous configuration options and Appendix A for detailed bit descriptions. Register WR4, which defines the channel mode, must be programmed before registers WR1, WR3 and WR5, since these registers are used differently in each SIO mode.

4. If other Quadart channels are to be used, initialize them; if not, Quadart initialization is complete since all other channels are disabled by a POC or a reset.

At this point the byte synchronous channel is ready for operation. If interrupts are used, the IOP must load the IOP Z80A I-register, select IM2 mode, and enable IOP interrupts (execute an EI instruction). The IOP sends data characters by outputting bytes to the channel SIO Data Port when status bit TBE is set; this status is determined by polling RR0, or by a TBE interrupt (the first TBE interrupt does not occur until the IOP loads a character into the Tx buffer and the buffer then becomes empty). The IOP reads data characters by inputting bytes from the channel SIO Data Port when status bit RCA is set; this status is determined by polling RR0, or by servicing a RCA interrupt -- see Section 4.2.

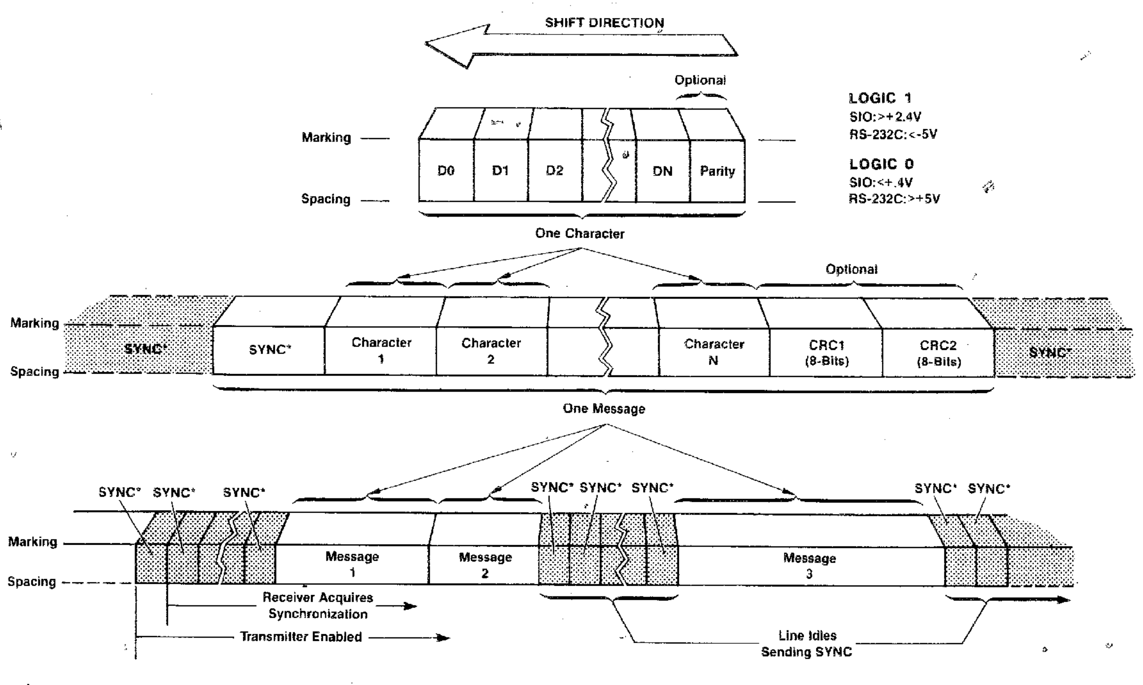
The recommended order for loading SIO registers is: (1) WR4 to define the byte synchronous mode and sync character length, (2) WR1 to configure the channel interrupt behavior, (3) WR2 to define the base interrupt vector if any SIO channel interrupt is enabled, (4) WR6 to load the transmitter sync character if using 8-bit sync characters, or to define the low order sync bits if using a 16-bit sync character, (5) WR7 to load the receiver sync character if using 8-bit sync characters, or to define the high order sync bits if using a 16-bit sync character, (6) WR3 to configure and enable the channel receiver, and (5) WR5 to configure and enable the channel transmitter. Register WR5 also defines the CRC polynomial used by both the channel transmitter and receiver. Either CRC-16 or CRC-CCITT may be selected in the byte synchronous mode, but note that the Reset Tx CRC Generator and Reset Rx CRC Checker commands preload the generator and checker with all zeroes, regardless of the polynomial used.

The complete range of byte synchronous mode SIO configuration options is illustrated in Figure 35. Some WR1 through WR7 bits select among available options, while other **don't care** bits are conservatively programmed reset (function disabled).

A Channel Reset command should be issued to WR0 before loading SIO registers WR1 through WR7. As explained in Section 4.5, the Reset EXSTAT Interrupts command should be issued twice after enabling EXSTAT interrupts if control bit EXSTAT Interrupts Enable is set. Configuring a channel for byte synchronous operation automatically defines the EXSTAT group and RXSPCL group bit functions as shown in Table 8. A Reset Channel command resets all four RXSPCL bits and the two Reset EXSTAT Interrupts commands makes the EXSTAT bit group reflect current status.

6.2 BYTE SYNCHRONOUS TRANSMIT/CRC GENERATION

After the transmitter is initialized but before it is enabled, SIO pin $\overline{\text{TxD}}$ is held marking and status bit TBE is set. If control bit Send Break bit is set at any



*Note: 8- or 16- Bit SYNC Character

Figure 34: BYTE SYNCHRONOUS MODE MESSAGE FORMAT

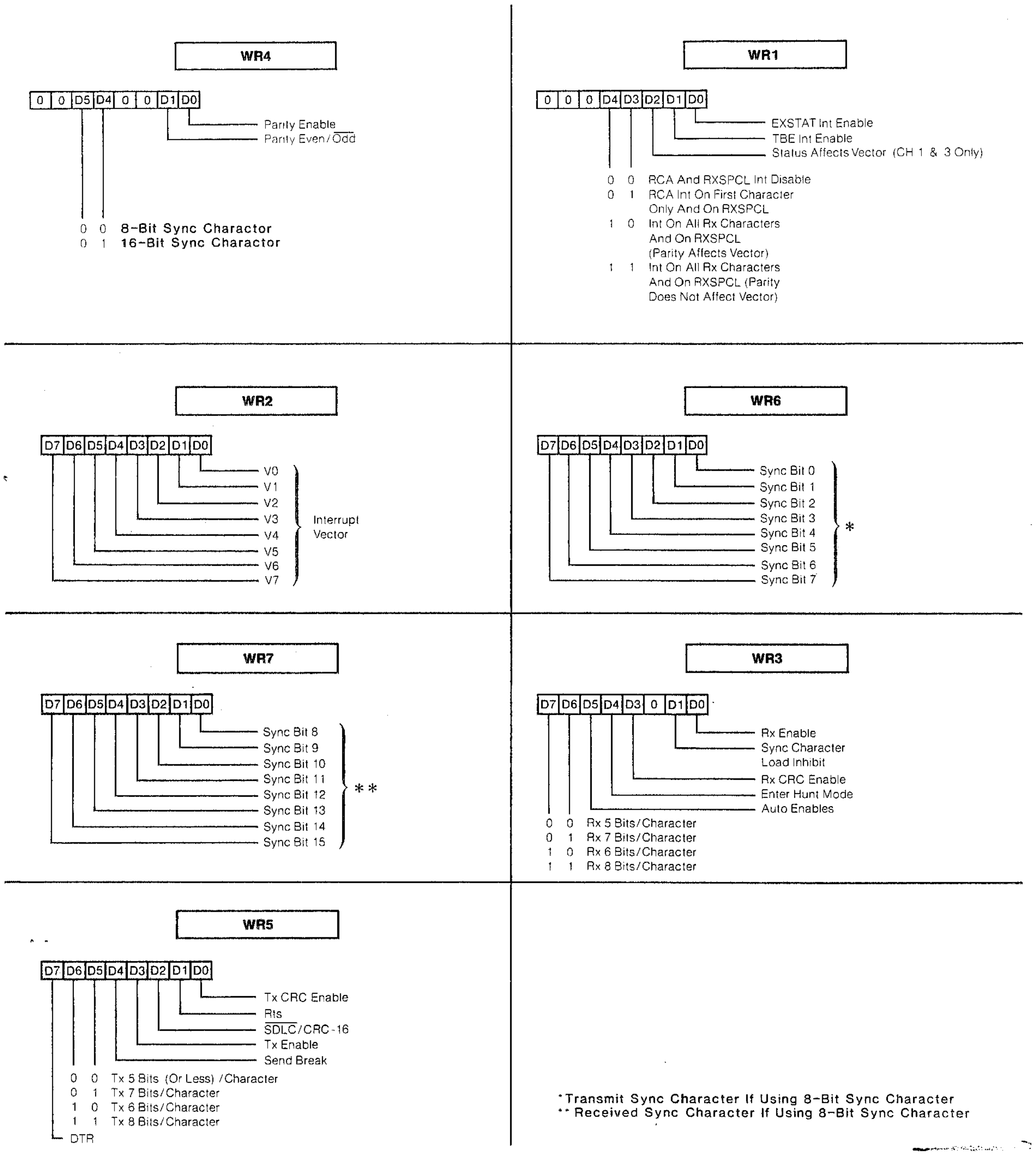


Figure 35: SIO BYTE SYNCHRONOUS MODE OPTIONS

time (with the transmitter either enabled or disabled) the TxD line is forced spacing, and it remains spacing until control bit Send Break is reset; since any bit pattern is possible while transmitting transparent binary data, the channel receiver does not recognize any type of break condition in the byte synchronous mode The channel transmitter begins sending sync characters when enabled, and it continues sending sync characters until the first character is loaded into the transmitter for shifting out.

Table 8: BYTE SYNCHRONOUS RR0 AND RR1 BIT FUNCTIONS

Bit	RR0	Bit	RR1
D7	Logic 0 (not used)	D7	Logic 0 (not used)
D6	Tx Underrun/EOM	D6	CRC Error
D5	$\overline{\text{Clear To}}$ Send	D5	Rx Overrun Error
D4	$\overline{\text{Sync}}$/Hunt	D4	Parity Error
D3	Data Carrier Detect	D3	Logic 1 (not used)
D2	Tx Buffer Empty	D2	Logic 1 (not used)
D1	Interrupt Pending	D1	Logic 1 (not used)
D0	Rx Char Available	D0	Logic 1 (not used)

Section 6.4 gives a byte synchronous programming example which also discusses initialization steps.

The IOP transmits a byte synchronous control or data character by loading it into the channel SIO Data Port when status bit TBE is set. The transmitter automatically adds a parity bit (if control bit Parity Enable is set) to any data character loaded by the IOP. The bits shifted out are: D0 (LSB), D1, ... , DN (MSB), parity (if enabled), in that order. Sync characters are shifted out Sync Bit 0 first, ..., Sync Bit N last (where N is either 7 or 15). Modem line TxD is spacing for logic 0 data/parity bits and marking for logic 1 data/parity bits. The bits are shifted out on the TxC falling edge.

If the programmed Tx character length is 6 or 7 bits, then the character bits must be loaded into the channel SIO Data Port right justified (the MSB bits are irrelevant). If the programmed Tx character length is 5 bits or less, then the unused high order bits must be formatted in a special way. The required format is detailed in Appendix A under the WR5 bit descriptions. Programming a new Tx character length does not affect any character currently being shifted out of the transmitter; a Tx character's length is fixed as the character is loaded from the Tx buffer into the transmitter.

A flowchart of interrupt driven byte synchronous message transmission is shown in Figure 36. The flowchart assumes that control bits TBE Interrupts Enable and EXSTAT Interrupts Enable are set. Many of the decisions made in the flowchart pertain directly to a few IBM BiSync Protocol requirements. The protocol requires, in part, that certain message characters be included, and others be excluded from the CRC block check character. Also when the transmitter temporarily has no more data to send, but does not want to relinquish the line (a **line turn around**), then it must idle sending sync characters; and these characters are to be excluded from the CRC calculation. When it idles the values of the sync characters must vary between transparent and nontransparent transmissions. The protocol also requires that the transmitter be able to send CRC immediately followed by more data in the case of intermediate block transmissions, or CRC followed by 0FFh pad characters in the case of message termination.

The flowchart begins after a POC by initializing the channel transmitter and enabling TBE interrupts. The transmitter automatically begins sending sync characters as soon as it is enabled. In this case, as well as when the transmitter underruns and then begins sending sync, transmitter inserted syncs are always either 8 or 16 bits long, regardless of the programmed Tx character length; and these sync characters are automatically excluded from the Tx CRC calculation. This does not apply to a sync character pattern loaded into the Tx buffer as ordinary data, however. After a pause for sync capture by the receiver (two or more sync characters should be sent before message characters), the CRC generator is reset to all zeroes. A decision is then made to either include or exclude the first message characters), the CRC generator is reset to all zeroes. A decision is then made to either include or exclude the first message character from the CRC (e.g., a leading 0FFh pad or an SOH after line turn around would be excluded while subsequent SOH characters would be included in BiSync Protocol). Control bit Tx CRC Enable is set to include a Tx character in the CRC, and is reset to exclude it, provided the bit is changed before the character is loaded into the Tx buffer. The first message character is then loaded into the empty Tx buffer, the Tx buffer quickly empties into the transmitter generating the first TBE interrupt, and the transmitter begins shifting out the first character (a TBE interrupt does not occur until this first character clears the Tx buffer).

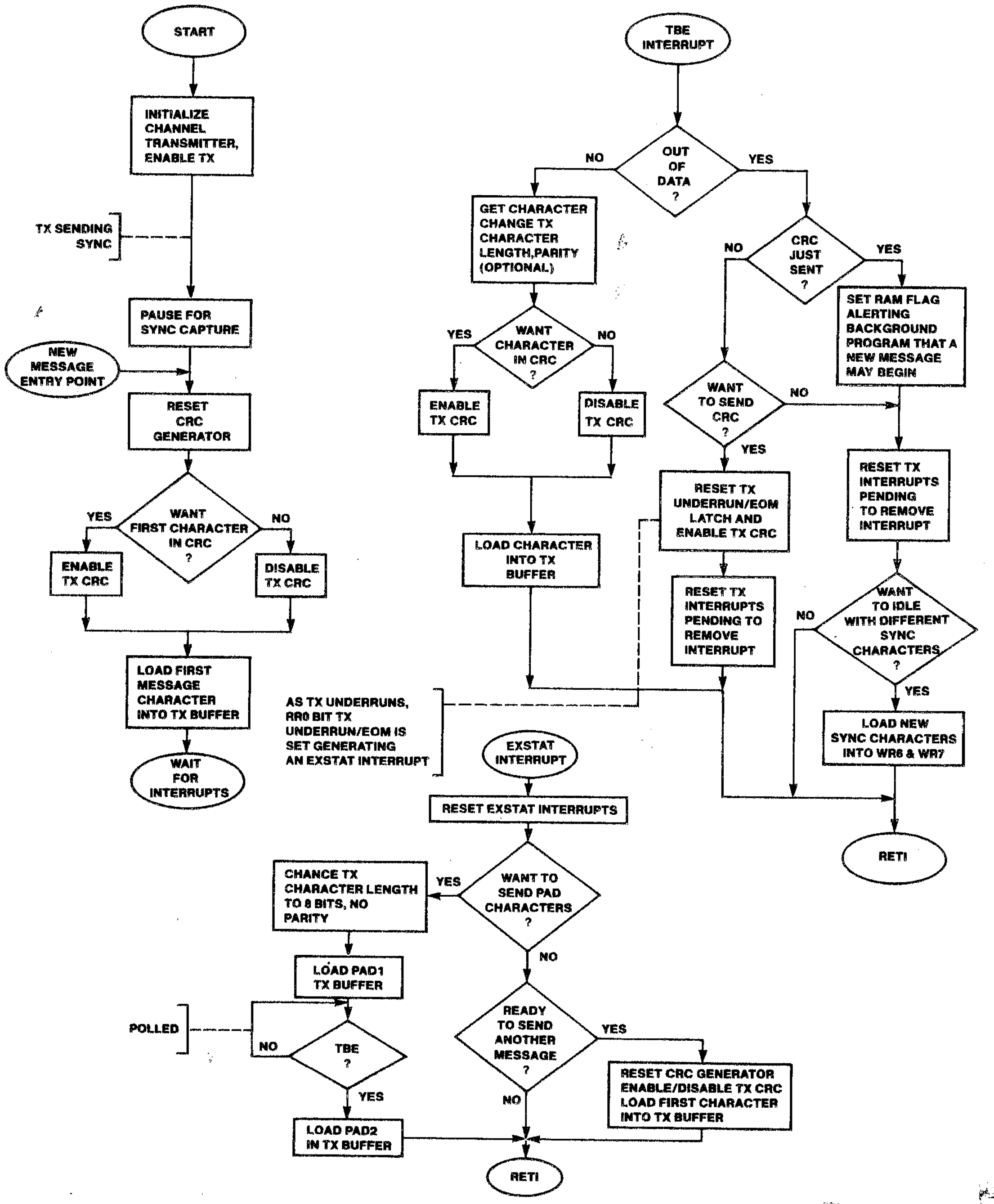

START
INITIALIZE CHANNEL TRANSMITTER, ENABLE TX
TX SENDING SYNC
PAUSE FOR SYNC CAPTURE
NEW MESSAGE ENTRY POINT
RESET CRC GENERATOR
WANT FIRST CHARACTER IN CRC ?
YES
NO
ENABLE TX CRC
DISABLE TX CRC
LOAD FIRST MESSAGE CHARACTER INTO TX BUFFER
WAIT FOR INTERRUPTS
TBE INTERRUPT
OUT OF DATA ?
NO
YES
GET CHARACTER CHANGE TX CHARACTER LENGTH,PARITY (OPTIONAL)
WANT CHARACTER IN CRC ?
YES
NO
ENABLE TX CRC
DISABLE TX CRC
LOAD CHARACTER INTO TX BUFFER
CRC JUST SENT ?
NO
YES
SET RAM FLAG ALERTING BACKGROUND PROGRAM THAT A NEW MESSAGE MAY BEGIN
WANT TO SEND CRC ?
NO
YES
RESET TX UNDERRUN/EOM LATCH AND ENABLE TX CRC
RESET TX INTERRUPTS PENDING TO REMOVE INTERRUPT
RESET TX INTERRUPTS PENDING TO REMOVE INTERRUPT
WANT TO IDLE WITH DIFFERENT SYNC CHARACTERS ?
NO
YES
LOAD NEW SYNC CHARACTERS INTO WR6 & WR7
RETI
AS TX UNDERRUNS, RR0 BIT TX UNDERRUN/EOM IS SET GENERATING AN EXSTAT INTERRUPT
EXSTAT INTERRUPT
RESET EXSTAT INTERRUPTS
WANT TO SEND PAD CHARACTERS ?
YES
NO
CHANCE TX CHARACTER LENGTH TO 8 BITS, NO PARITY
LOAD PAD1 TX BUFFER
POLLED
NO
TBE ?
YES
LOAD PAD2 IN TX BUFFER
READY TO SEND ANOTHER MESSAGE ?
YES
NO
RESET CRC GENERATOR ENABLE/DISABLE TX CRC LOAD FIRST CHARACTER INTO TX BUFFER
RETI

Figure 36: BYTE SYNCHRONOUS TRANSMIT FLOWCHART

For transmitting ordinary transparent or nontransparent data, the TBE service routine first decides whether to change the Tx character length and parity. If, for example, a leading 0FFh pad is sent with an 8-bit Tx character length and no parity (enabled parity adds an extra bit to the programmed Tx character length), then these parameters might be changed to a 7-bit Tx character length with odd parity for ASCII data transmission. A decision is then made to include or exclude the character from the CRC, followed by loading the character into the Tx buffer for transmission.

When a TBE interrupt occurs and the transmitter is out of data, then: (1) the transmitter is just temporarily out of data, so sync characters should be sent during idle, or (2) the end of message (EOM) has been reached, so CRC followed by either pad characters or sync characters should be sent, or (3) the end of an intermediate block has been reached, so CRC immediately followed by another message should be sent.

In the first case CRC has not just been sent, and should not be. A Reset TBE Interrupts Pending command is issued to WR0 which removes the TBE interrupt (status bit TBE is reset), and the transmitter then underruns with no data to send. When the transmitter underruns with the Tx Underrun/EOM latch set **or** control bit Tx Enable reset, then sync characters only are automatically sent (not a CRC character) and no further TBE interrupts occur. No EXSTAT interrupt occurs since the Tx Underrun/EOM bit is initially set when the transmitter is initialized it does not change state as a result of the underrun. The sync characters sent during idle may be changed by loading new values into WR6 and WR7 (e.g., SYN SYN might be sent during nontransparent idle, and DLE SYN during transparent idle). As Tx data again becomes available, the background routine must load a data character into the Tx buffer to reenable TBE interrupts. The CRC generator should not be reset, however.

In the second case, CRC has not just been sent, but should be followed by pads or sync. In this case a Reset Tx Underrun/EOM Latch command is issued to WR0 **and** control bit Tx CRC Enable is set which forces the transmitter to send CRC when it underruns. The state of the Tx Underrun/EOM latch is read from RR0. This bit is first set when the channel transmitter is initialized; it can only be reset by direct command and thereafter by a transmitter underrun. Note that resetting the bit by direct command does **not** generate an EXSTAT interrupt, but a reset to set bit change as the transmitter underruns **does** cause an EXSTAT interrupt (if enabled).

A Reset TBE Interrupts Pending command is then issued to WR0 to remove the TBE interrupt and the service routine returns to the background program. As the transmitter later underruns an EXSTAT interrupt occurs and CRC begins shifting out of the transmitter with status bit Tx Underrun/EOM set and status bit TBE reset. If another character **is not** loaded into the Tx buffer before CRC is completely sent, then by default the transmitter loads a sync character into the transmitter and begins sending it with status bit TBE set (this event generates a TBE interrupt). If another character **is** loaded into the Tx buffer while CRC is being sent, then this character will follow CRC, and a TBE interrupt will result when the character empties into the transmitter. Thus, the EXSTAT service routine, which responds to the Tx underrun before the sync generated TBE interrupt, decides whether to follow CRC by pads or sync. If pad characters are to follow, the EXSTAT service routine may either load them as ordinary data, or it may load WR6 and WR7 with pad characters which will be sent during idle. If the pad characters are loaded as ordinary data, a TBE interrupt occurs as the last one clears the Tx buffer. The flowchart path in this case indicates that CRC has just been sent (this status could be detected by reading a RAM flag which would be set when the Tx Underrun/EOM latch is reset), and a Reset TBE Interrupts command should be issued to WR0 to remove the TBE interrupt. The transmitter then idles sending sync. If sync characters instead of pads are to follow the CRC, then the EXSTAT service routine merely resets EXSTAT interrupts and returns. Again, a TBE interrupt occurs after the first sync character enters the transmitter, and the interrupt is cleared by a Reset TBE Interrupts command.

In the third case, CRC has not just been sent, but should be, and data from the next intermediate block should immediately follow CRC. Here again the Tx Underrun/EOM latch is reset by direct command, control bit Tx CRC Enable is set, and the TBE interrupt is removed by issuing a Reset TBE Interrupts Pending command just as in the previous case. The path taken in the EXSTAT service routine following a Tx underrun differs, however. Now, instead of loading pad characters into the Tx buffer, the EXSTAT service resets the CRC generator (optional), enables or disables the Tx CRC to include or exclude the character from the CRC, and then loads the first block character into the Tx buffer. Since this character is loaded while CRC is being sent, it immediately follows CRC. When this character empties into the transmitter, another TBE interrupt results, and normal interrupt driven data transfer resumes with a new CRC if the generator is reset, or with an ongoing CRC otherwise.

To summarize the key points from the foregoing discussion, in the byte synchronous mode:

1. Control bit Tx CRC Enable must be set or reset before a character is loaded into the Tx buffer to include or exclude it from the CRC calculation.

2. Together, the Tx Underrun/EOM latch and control bit Tx CRC Enable determine whether only sync, or CRC followed by sync is sent when the transmitter underruns. If the latch is reset **and** the control bit are set as the Tx underrun occurs, then CRC is sent; CRC is not sent otherwise.

3. The Tx Underrun/EOM latch is set when the transmitter is initialized and when it underruns; it is reset only by a Reset Tx Underrun/EOM Latch command.

4. A Tx Underrun/EOM EXSTAT interrupt only occurs when the bit changes from reset to set. This can only occur if the bit has been reset by direct command, and then set by a Tx underrun.

5. If a character is loaded into the Tx buffer as CRC is being sent, then the character immediately follows the CRC; if the CRC is completely sent and no character is in the Tx buffer, sync follows the CRC.

6. If a character follows the CRC, then a TBE interrupt occurs as the character is loaded into the transmitter; if sync follows the CRC, a TBE interrupt occurs when the first sync character is loaded into the transmitter (but not on following sync characters).

7. Sync characters automatically sent by the transmitter are automatically excluded from the CRC calculation.

The sequence of events presented in the flowchart may be easily modified for a polled environment.

Message transmission may be terminated as soon as CRC and all pad characters have cleared the transmitter. This status may be determined from status bit TBE (status bit All Sent does not function in the byte synchronous mode). The transmitter may be disabled by resetting control bit Tx Enable, or by modem line CTS turning off when control bit Auto Enables is set. If the transmitter is disabled as a character or CRC is being shifted out, then the transmitter completely sends the character or CRC before forcing line TxD marking. Any character in the Tx buffer remains there when the transmitter is disabled. If CRC is being sent when the transmitter is disabled, a full 16 bits are sent, but sync bits replace CRC bits.

6.3 BYTE SYNCHRONOUS RECEIVE/CRC CHECKING

Initializing and enabling the channel receiver forces it into the hunt mode, and it remains there until at least one character matching the programmed receiver sync character (8 or 16 bits) is received. When this character is received, status bit $\overline{\text{Sync}}$/Hunt changes from set to reset, and this transition causes an EXSTAT interrupt, if enabled. After sync capture, received data is assembled and transferred to the Rx FIFO until the receiver is either placed in the hunt mode by setting control bit Enter Hunt mode, or disabled.

The SIO receiver samples the $\overline{\text{RxD}}$ data line on the $\overline{\text{RxC}}$ rising edge. Data characters are assembled LSB first: D0 (LSB), D1, ... , DN (MSB) and parity (if enabled), in that order. After the character is completely assembled, it is transferred to the Rx data FIFO for reading from the channel SIO Data Port and status bit RCA is set. Eight bits are always read from this port, and if the programmed Rx character length is 8 bits, then the byte consists of bits D7 (MSB) on the left through D0 (LSB) on the right (the parity bit is not included in the read data). If the programmed character length is 7 or less, then the extra high order bits to the left are the low order bits of the next character, followed by the parity bit (if enabled) and DN (MSB) through D0 (LSB) to the right. Byte synchronous data is not automatically right justified if the Rx character length is changed on the fly -- see Section 4.4.

As each character is assembled and loaded into the Rx FIFO, three RXSPCL bits are parallel loaded into the error FIFO: CRC Error, Rx Overrun Error, and Parity Error. If control bit Parity Enable is set and the character is assembled with incorrect parity, a set Parity Error bit is loaded; otherwise a reset Parity Error bit is loaded. If three characters are stacked unread in the received data FIFO as a fourth character is transferred from the receiver to the FIFO, then the character on the bottom of the Rx FIFO is overwritten, and a set status bit Rx Overrun Error is loaded into the bottom of the error FIFO opposite the overwritten data character. The fourth character's CRC and Parity status overwrite the third character's status on the bottom of the error FIFO also. Received data which does not overwrite the bottom FIFO character in this way causes a reset Rx Overrun Error bit to be loaded into the error FIFO. CRC Errors are covered below. Refer to Section 4.3 for more information on the data and error FIFOs.

The SIO internal CRC circuitry requires that no more than **one** character be in the received data FIFO at any time. The reason for this is that received character N is either included or excluded from the CRC calculation depending on the state of control bit Rx CRC Enable at the time character N+1 is transferred from the receiver to the Rx FIFO; since each character must be checked individually before the next character arrives, there can't be two characters in the FIFO at once.

The operation of the CRC checking circuitry can best be understood with a simple example. Assume that a transmitter sends an **empty** message. To completely format the message, the transmitter might send STX EXT CRC1 CRC2 PAD1 PAD2. Characters STX and ETX may be encoded in any way, but assume they are in 6-bit transcode with odd parity. Then STX, EXT are each 7 bits long, the 16-bit CRC is received as two 8-bit bytes, and PAD1 and PAD2 are 8-bit 0FFh characters. The SIO receiver feeds the 16-bit CRC checker through an 8-bit delay register, as shown in Figure 37. The CRC result, or remainder, is valid when the entire 16-bit CRC character (both CRC1 and CRC2) fits evenly in the checker. Assume now that characters STX, and EXT have been received, and that when character EXT is received the Rx character length is changed from 6 to 8 bits. When both CRC1 and CRC2 have arrived the situation in the figure, part (A), results.

Here, CRC1 fits evenly in the delay register, and CRC2 has just been transferred from the receiver to the Rx FIFO. Both CRC1 and CRC2 are read from the FIFO and discarded. A set CRC Error bit moves to the top of the error FIFO where it may be read from RR1. This bit reflects the current CRC remainder, and consequently it is almost always set until the correct CRC arrives. Note that status bit CRC Error **cannot** generate an interrupt of any kind; it must be polled at the proper moment.

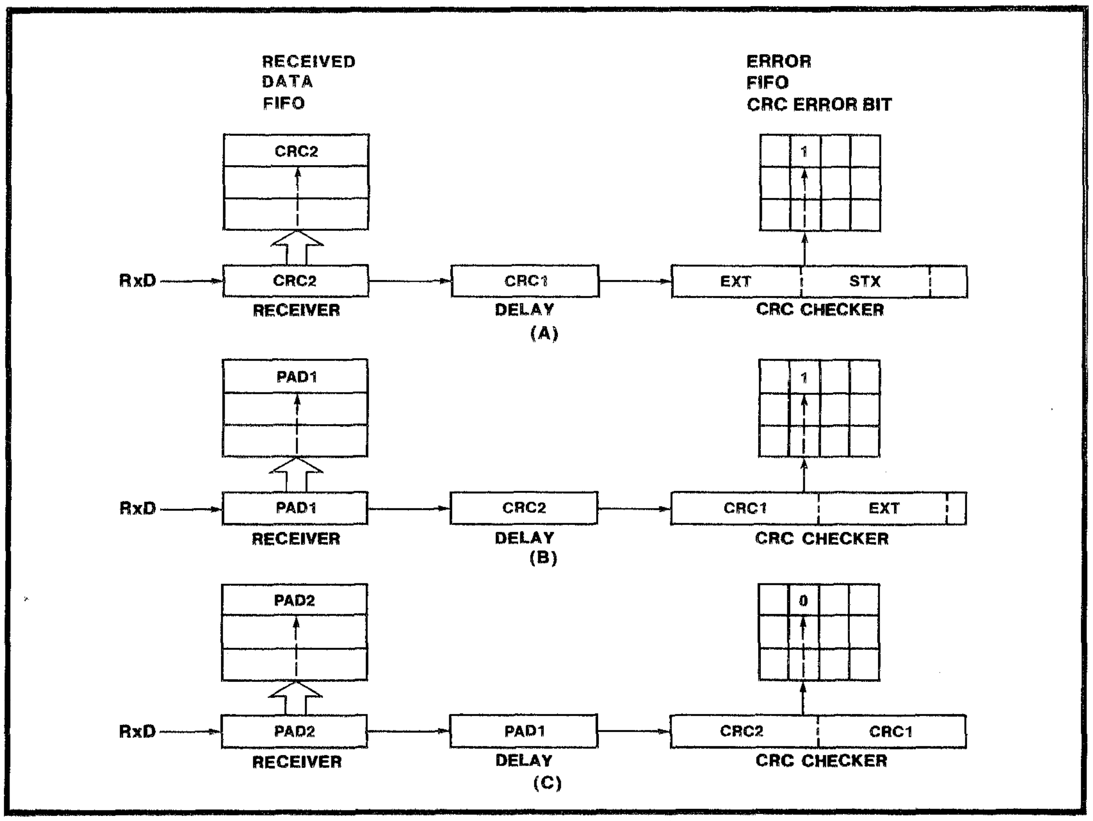


Figure 37: CRC ERROR CHECKING

PAD1 has arrived in part (B) of the figure, and since it is 8 bits long, then CRC2 evenly fits in the delay register, CRC2 is wholly in the CRC checker, and a set CRC Error bit is loaded into the error FIFO at the same time as PAD1 is loaded into the Rx FIFO. In part (C), PAD2 has arrived, PAD1 is in the delay register, and CRC1 and CRC2 fit evenly in the CRC checker. At this time status bit CRC Error is valid. The status is loaded into the error FIFO at the same time PAD2 is loaded into the Rx FIFO, so when PAD2 sets status bit RCA, then the moment has arrived to sample valid CRC Error status. Conservatively, the bit should be sampled before PAD2 is read from the FIFO to preclude a case where another character is stacked with PAD2 in the FIFO. In this case reading PAD2 first would pop both PAD2 and the valid CRC Error status bit from the FIFOs.

The proper procedure for byte synchronous CRC checking is then: (1) reset the CRC checker at the beginning of each message, (2) use the transmitter to send CRC followed by at least two 8-bit pad characters (any 8-bit characters will do, but Bisync Protocol requires at least one 0FFh pad), (3) alter the channel receiver so CRC comes next by receiving a control character like ETX, (4) change the Rx character length to 8 bits which then applies to CRC1 and the following characters, (5) read and discard CRC1, (6) read and discard CRC2, (7) read and discard PAD1, (8) PAD2 sets status bit RCA 16 bit times after CRC is completely received, (9) sample valid status bit CRC Error and take appropriate action, (10) read and discard PAD2.

The flowchart for receiving a byte synchronous interrupt driven message is shown in Figure 38. The flowchart assumes that either RCA Interrupts On All Characters (Parity Does/Parity Does Not Affect Vector) mode is enabled. After a POC the channel receiver is initialized and enabled with control bit Sync Character Load Inhibit set. This inhibits sync characters from being loaded into the Rx FIFO which means an RCA interrupt does not occur until the first nonsync message character is received. The CRC checker is then reset to all zeroes and the IOP program executes a background program while waiting for RCA interrupts. As soon as the receiver is synchronized, EXSTAT bit $\overline{\text{Sync}}$/Hunt changes from set to reset generating an EXSTAT interrupt, if enabled. The service routine should issue a Reset EXSTAT Interrupts command to WR0 to remove the EXSTAT interrupt.

Once an RCA interrupt occurs control bit Sync Character Load Inhibit is reset which forces all received characters to pass through the Rx FIFO. This must be done because sync characters which are not loaded into the FIFO are still routed to the CRC checker. If all receive characters are read from the Rx FIFO, then the IOP program may decide on a character by character basis whether to include or exclude each from the CRC calculation. If an RCA interrupt results from receiving CRC1, CRC2, or PAD1, then these characters are read and discarded to remove the RCA interrupt as described above. If the receiver character is PAD2, then status bit CRC Error is read from RR1 and appropriate action is taken. PAD2 is then read to remove the RCA interrupt request to the IOP and the remaining flowchart steps initialize the receiver for the next message. After these steps, the next RCA interrupt is generated by the first nonsync character of the following message after sync is acquired.

If an RCA interrupt results from any characters other than CRC and PADs, then the control/data character is read from the FIFO and stored in a RAM buffer if it is a

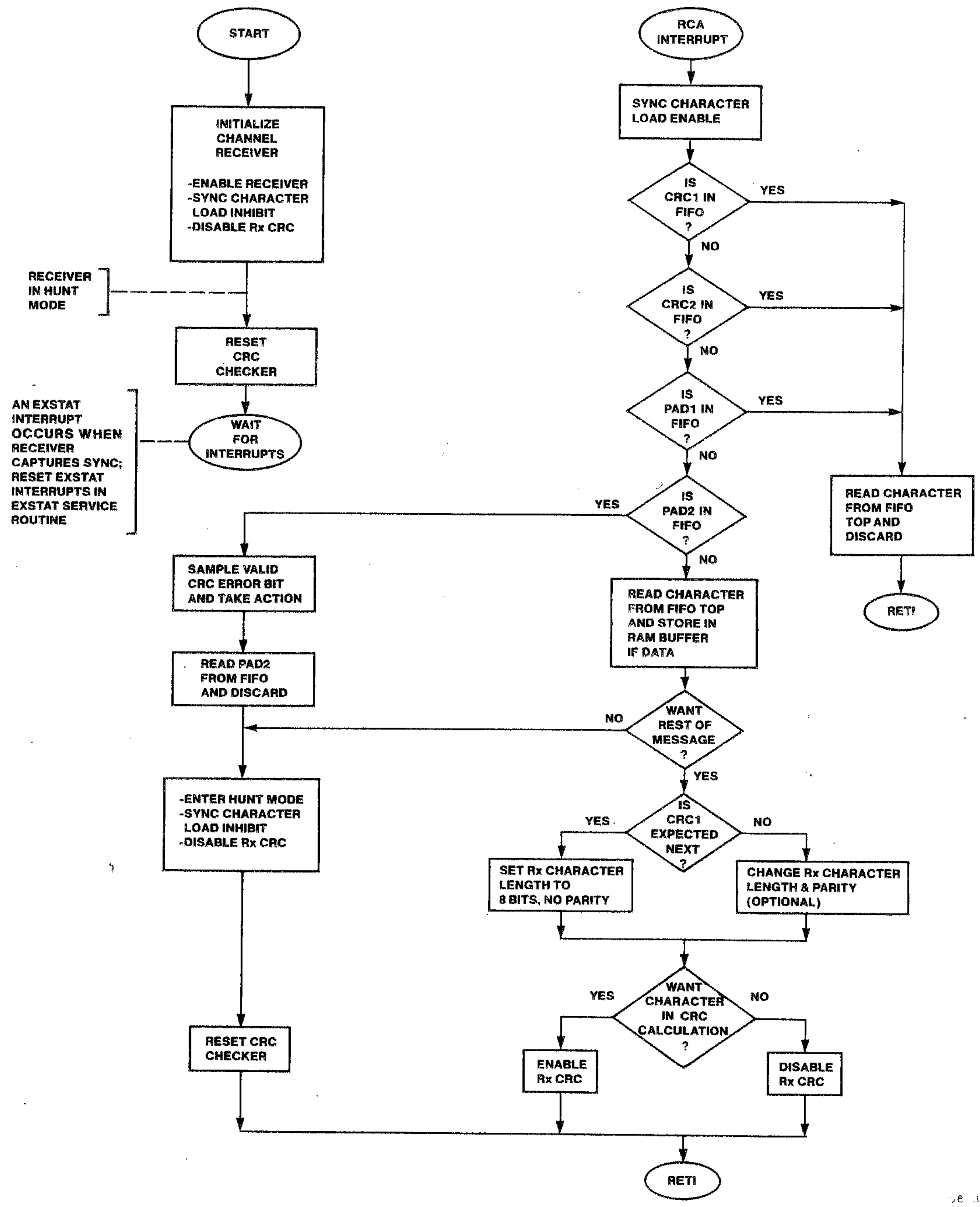

START
INITIALIZE CHANNEL RECEIVER
-ENABLE RECEIVER
-SYNC CHARACTER LOAD INHIBIT
-DISABLE Rx CRC
RECEIVER IN HUNT MODE
RESET CRC CHECKER
AN EXSTAT INTERRUPT OCCURS WHEN RECEIVER CAPTURES SYNC; RESET EXSTAT INTERRUPTS IN EXSTAT SERVICE ROUTINE
WAIT FOR INTERRUPTS
SAMPLE VALID CRC ERROR BIT AND TAKE ACTION
READ PAD2 FROM FIFO AND DISCARD
-ENTER HUNT MODE
-SYNC CHARACTER LOAD INHIBIT
-DISABLE Rx CRC
RESET CRC CHECKER
RCA INTERRUPT
SYNC CHARACTER LOAD ENABLE
IS CRC1 IN FIFO ?
YES
NO
IS CRC2 IN FIFO ?
YES
NO
IS PAD1 IN FIFO ?
YES
NO
IS PAD2 IN FIFO ?
YES
NO
READ CHARACTER FROM FIFO TOP AND DISCARD
RETI
READ CHARACTER FROM FIFO TOP AND STORE IN RAM BUFFER IF DATA
WANT REST OF MESSAGE ?
NO
YES
IS CRC1 EXPECTED NEXT ?
YES
NO
SET Rx CHARACTER LENGTH TO 8 BITS, NO PARITY
CHANGE Rx CHARACTER LENGTH & PARITY (OPTIONAL)
WANT CHARACTER IN CRC CALCULATION ?
YES
NO
ENABLE Rx CRC
DISABLE Rx CRC
RETI

Figure 38: BYTE SYNCHRONOUS RECEIVE FLOWCHART

data character. If the character value implies CRC is coming next, the Rx character length should be changed to 8 bits for the reasons discussed above. The Rx character length may be changed, but the change should be delayed one character to right justify the read data as discussed in Section 4.4. The BiSync Protocol requires that some characters be included, while other characters be excluded, from the CRC calculation (e.g., sync characters are excluded during nontransparent reception, but are included during transparent reception except when preceded by a single DLE, and so on). Setting control bit Rx CRC Enable includes the character in the CRC calculation; resetting it excludes the character from the CRC calculation, provided the bit is in its proper state before the next character is loaded into the Rx FIFO as explained above.

If the receiver is disabled by resetting control bit Rx Enable, or by an off DCD modem line when control bit Auto Enables mode is set, then the character currently being assembled in the receiver is lost, but any characters or status in the FIFOs may still be correctly read. If character synchronization is lost the channel receiver may be placed in the hunt mode at any time by setting control bit Enter Hunt Mode. This sets status bit $\overline{\text{Sync}}$/Hunt which generates an EXSTAT interrupt, if enabled. The EXSTAT interrupt should be removed by a Reset EXSTAT Interrupts command, and when the receiver is resynchronized, another EXSTAT interrupt occurs as status bit $\overline{\text{Sync}}$/Hunt changes from set to reset. This EXSTAT interrupt should be handled in the same way.

6.4 A BYTE SYNCHRONOUS PROGRAMMING EXAMPLE

In overview, the following program connects the channel 1 SIO transmitter to the channel 1 receiver using the Quadart loopback system, then sends a short byte synchronous message followed by CRC and two pad characters. The received data is stored in a RAM buffer and a CRC check is performed. A period character is sent to the host system by the IOP if the message is received with no CRC error, and an asterisk is sent to the host if a CRC error is detected. The program then loops back and repeats this process indefinitely.

Since there is no modem supplied clock, the channel 1 CTC feeds SIO $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ inputs (at 1200 bps). Channel 1 is configured in the byte synchronous mode with a 16-bit sync character. CRC-16 is used with parity disabled (both for the transmitter and the receiver). The transmitter is interrupt driven (TBE interrupts enabled, EXSTAT interrupts enabled) and the receiver is polled (all RCA and RXSPCL interrupts disabled). The program configures the IOP Z80A for IM2 interrupts with an I-Page of 44h, and a base interrupt vector of 00h is written to WR2 with control bit Status Affects Vector set. This means there are two active interrupts: TBE with indirect jump address 4400h, and EXSTAT interrupts with indirect jump address 4402h. ASCII code is used for message characters, so the 16-bit sync character is defined to be SYN SYN (3A3Ah). The transmitter and the receiver character lengths are set to 8 bits, so the MSB of all characters sent and received is logic 0.

The program is subdivided into three major segments: initialization, main background program, and interrupt foreground routines. The main program is entered with both the CRC generator and checker reset, and with the transmitter beginning to send sync after being enabled. The routine then pauses for the receiver to capture sync. An EXSTAT interrupt occurs as status bit $\overline{\text{Sync}}$/Hunt is reset when the receiver is synchronized, and the service routine removes the interrupt and returns. Control bit Tx CRC Enable is set and the first message character is then loaded into the Tx buffer. TBE interrupts become active as the Tx buffer empties for the first time.

Each TBE interrupt causes the transmitter to get and send a Tx data character, except when flag 0FFh is read at the end of the Tx data. In this event, RAM flag EOMFLG is set and a Reset TBE Interrupts Pending command is issued followed by a Reset Tx Underrun/EOM Latch command. This last command causes CRC to be sent as the transmitter underruns (control bit Tx CRC Enable is set throughout the routine). An EXSTAT interrupt is generated when the transmitter underruns, and the EXSTAT service routine loads two pad characters in the Tx buffer before CRC is completely sent which causes them to immediately follow the CRC.

Back in the main program, the receiver continually polls status bit RCA. The first time this bit is set, control bit Rx CRC Enable is set. Each Rx character is read and stored in a RAM Rx data buffer until control character ETX is received, which signifies that the CRC follows. A counter (Register C) is used to count characters from ETX onward (including ETX). The characters following ETX (CRC1, CRC2 and PAD1) are read and discarded. When (C) equals four, then PAD2 is in the data FIFO and status bit CRC Error is valid. This bit is polled and the outcome is sent to the host. PAD2 is then read and discarded, and the receiver is forced to the hunt mode. The entire process is then repeated beginning with a pause for sync capture, and so on.

```
CROMEMCO Z80 Macro Assembler version 03.07

                   ;       BYTE SYNCHRONOUS EXAMPLE ROUTINE
                   ;
                   ;  See manual text for program description.
                   ;
       (0040)      QBASE   EQU     40h             ;Assumed Quadart Base Address.
       (0042)      SDATA1  EQU     QBASE+02h       ;Ch 1 SIO Data Port.
       (0043)      SCMST1  EQU     QBASE+03h       ;Ch 1 SIO Command/Status Port.
       (0048)      PDAT01  EQU     QBASE+08h       ;Ch 0&1 PIO Data Port.
       (0049)      PCOM01  EQU     QBASE+09h       ;Ch 0&1 PIO Command Port.
       (004D)      CTC1    EQU     QBASE+0Dh       ;Ch 1 CTC Port.
       (0054)      CNTRL   EQU     QBASE+14h       ;Loopback Control Port.
       (0001)      HDATA   EQU     1               ;IOP <-> HOST Data Port.
       (0002)      FLAGS   EQU     2               ;IOP Flags Port.

       (0002)      STX     EQU     02h             ;ASCII equivalents.
       (003A)      SYN     EQU     3Ah             ;
       (0003)      ETX     EQU     03h             ;

       (4000)              ORG     4000h           ;Start of IOP ram.

                           ;==========================================
                           ; Initialization Routine
                           ;==========================================

4000  310060       BISYNC: LD      SP,6000h        ;Locate stack.

4003  3ECF                 LD      A,11001111b     ;Ch 0&1 PIO to
4005  D349                 OUT     (PCOM01),A      ;Control Mode 3.
4007  3ECC                 LD      A,11001100b     ;Define I/O lines:
4009  D349                 OUT     (PCOM01),A      ;0=output, 1=input.
400B  3E07                 LD      A,00000111b     ;Disable PIO interrupts.
400D  D349                 OUT     (PCOM01),A      ;
400F  3E00                 LD      A,00000000b     ;Reset ExtCk1 to connect
4011  D348                 OUT     (PDAT01),A      ;CTC to SIO Ch 1 TxC and RxC.

4013  3E47                 LD      A,01000111b     ;Ch 1 CTC to counter mode,
4015  D34D                 OUT     (CTC1),A        ;interrupts disabled.
4017  3E00                 LD      A,0             ;Time constant = 256 for
4019  D34D                 OUT     (CTC1),A        ;1200 bps with X1 multiplier.

401B  213140               LD      HL,SIOBEG       ;Point to Ch 1 initialization
401E  0E43                 LD      C,SCMST1        ;data for OTIR block move.
4020  060F                 LD      B,SIOEND-SIOBEG ;
4022  EDB3                 OTIR                    ;Initialize SIO Ch 1.

4024  3E89                 LD      A,10001001b     ;Loopback Ch 1 TxD to
4026  D354                 OUT     (CNTRL),A       ;Ch 1 RxD.
4028  3E44                 LD      A,44h           ;Select I-Page.
402A  ED47                 LD      I,A             ;
402C  ED5E                 IM      2               ;Select Interrupt Mode 2.
402E  FB                   EI                      ;Enable IOP interrupts.

402F  180F                 JR      AROUND          ;Jump around SIO data.

                           ;=====================
                           ; SIO Block Move Ch 1
                           ; Initialization Data
                           ;=====================

4031  18           SIOBEG: DB      00011000b       ;Reset Ch 1 command.

4032  04                   DB      00000100b       ;Point to WR4.
4033  10                   DB      00010000b       ; - X1 Clock
                                                   ; - 16 bit sync character
                                                   ; - no parity

4034  01                   DB      00000001b       ;Point to WR1.
4035  07                   DB      00000111b       ; - disable Rx interrupts
                                                   ; - status affects vector
                                                   ; - enable TBE interrupts
                                                   ; - enable EXSTAT interrupts

4036  02                   DB      00000010b       ;Point to WR2.
4037  00                   DB      00000000b       ;Base Interrupt Vector = 00h.
                                                   ;Resulting IM2 Jump Table
                                                   ;assuming I-Page = 44h:
                                                   ;4400h -> Ch 1 TBE
                                                   ;4402h -> Ch 1 EXSTAT
                                                   ;4404h -> Ch 1 RCA
                                                   ;4406h -> Ch 1 RXSPCL

4038  06                   DB      00000110b       ;Point to WR6.
```

```
4039  3A                     DB     SYN             ;Define low order sync character.

403A  07                     DB     00000111b       ;Point to WR7.
403B  3A                     DB     SYN             ;Define high order sync character.

403C  53                     DB     01010011b       ;Point to WR3, reset EXSTAT intrpts,
403D  C3                     DB     11000011b       ;and reset CRC checker.
                                                    ; - 8 bit Rx characters
                                                    ; - Rx CRC disable
                                                    ; - Sync Character Load Inhibit
                                                    ; - Rx enable

403E  95                     DB     10010101b       ;Point to WR5, reset EXSTAT intrpts,
403F  EA                     DB     11101010b       ;reset CRC generator.
                                                    ; - DTR on
                                                    ; - 8 bit Tx characters
                                                    ; - Tx enable
                                                    ; - CRC-16
                                                    ; - RTS on
                                                    ; - Tx CRC disable

                    SIOEND:

                             ;=============================================
                             ; Main Program
                             ;=============================================

4040  3E00          AROUND:  LD     A,0             ;Reset ram end of message
4042  32CA40                 LD     (EOMFLG),A      ;flag used in TBE service.

4045  210000                 LD     HL,0            ;Pause for receiver sync
4048  2B            PAUSE:   DEC    HL              ;capture by decrementing
4049  7C                     LD     A,H             ;HL thru full range.
404A  B5                     OR     L               ;This pause is needed only
404B  20FB                   JR     NZ,PAUSE        ;on first pass.

404D  F3                     DI                     ;
404E  3E85                   LD     A,10000101b     ;Point to WR5, reset
4050  D343                   OUT    (SCMST1),A      ;CRC generator
4052  3EEB                   LD     A,11101011b     ;and enable Tx CRC.
4054  D343                   OUT    (SCMST1),A      ;
4056  FB                     EI                     ;

4057  21C440                 LD     HL,TXDATA       ;Point to Tx data.  Load first
405A  7E                     LD     A,(HL)          ;character into Tx buffer.  TBE
405B  D342                   OUT    (SDATA1),A      ;interrupts will now send rest
                                                    ;of message.

405D  DD21CB40               LD     IX,RXDATA       ;Point to Rx data area.
4061  0E00                   LD     C,0             ;Reg. C counts characters read
                                                    ;near end of message.  (C)=1 for
                                                    ;ETX read, 2 for CRC1 read, 3 for
                                                    ;CRC2 read, 4 for PAD1 read.

4063  F3            RCA?:    DI                     ;
4064  3E00                   LD     A,0             ;Read RR0 and loop until
4066  D343                   OUT    (SCMST1),A      ;RCA status is true.
4068  DB43                   IN     A,(SCMST1)      ;
406A  CB47                   BIT    0,A             ;
406C  28F5                   JR     Z,RCA?          ;
406E  FB                     EI                     ;

406F  F3                     DI                     ;
4070  3E03                   LD     A,3             ;Point to WR3,
4072  D343                   OUT    (SCMST1),A      ;
4074  3EC9                   LD     A,11001001b     ;enable Rx CRC and sync
4076  D343                   OUT    (SCMST1),A      ;character load enable.
4078  FB                     EI                     ;

4079  79                     LD     A,C             ;Test the value of
407A  FE00                   CP     0               ;Reg. C ...
407C  200E                   JR     NZ,NREND        ;

407E  DB42                   IN     A,(SDATA1)      ;If (C)=0, read data from
4080  DD7700                 LD     (IX),A          ;FIFO, store in ram buffer,
4083  DD23                   INC    IX              ;bump pointer.  If character
4085  FE03                   CP     ETX             ;is ETX control, set (C)=1.
4087  20DA                   JR     NZ,RCA?         ;Go back and test RCA status
4089  0C                     INC    C               ;in either case.
408A  18D7                   JR     RCA?            ;

408C  FE04          NREND:   CP     4               ;
408E  2805                   JR     Z,CHKCRC        ;

```

```
4090  0C                       INC     C               ;If 0 < (C) < 4, then
4091  DB42                     IN      A,(SDATA1)      ;read and discard character
4093  18CE                     JR      RCA?            ;from FIFO, check RCA status.

4095  F3             CHKCRC:   DI                      ;
4096  3E01                     LD      A,1             ;If (C)=4, then read
4098  D343                     OUT     (SCMST1),A      ;CRC Error bit from RR1.
409A  DB43                     IN      A,(SCMST1)      ;
409C  FB                       EI                      ;
409D  CB77                     BIT     6,A             ;If bit is reset, send a
409F  2804                     JR      Z,OK            ;period to the host for
40A1  1E2A                     LD      E,'*'           ;OK.  If bit is set, send
40A3  1802                     JR      TOHOST          ;an asterisk to host for
40A5  1E2E           OK:       LD      E,'.'           ;CRC Error.

40A7  DB02           TOHOST:   IN      A,(FLAGS)       ;Loop until IOP flag port
40A9  CB7F                     BIT     7,A             ;bit D7 is set, then
40AB  28FA                     JR      Z,TOHOST        ;send either a period or
40AD  7B                       LD      A,E             ;an asterisk to host.
40AE  D301                     OUT     (HDATA),A       ;

40B0  DB42                     IN      A,(SDATA1)      ;Read and discard PAD2,
40B2  3E00                     LD      A,0             ;then reset ram EOMFLG.
40B4  32CA40                   LD      (EOMFLG),A      ;

40B7  F3                       DI                      ;
40B8  3E43                     LD      A,01000011b     ;Point to WR3, reset
40BA  D343                     OUT     (SCMST1),A      ;CRC checker, enter hunt
40BC  3ED3                     LD      A,11010011b     ;mode, disable Rx CRC, inhibit
40BE  D343                     OUT     (SCMST1),A      ;sync character loading.
40C0  FB                       EI                      ;

40C1  C34040                   JP      AROUND          ;Start over.


40C4  02414243       TXDATA:   DB      STX,'ABC',ETX   ;Tx Data Area.
40C9  24                       DB      '$'             ;Tx Data Delimiter.

40CA  (0001)         EOMFLG:   DS      1               ;End Of Message Flag.
                                                       ; 0 => not end of messg
                                                       ;-1 => end of messg

40CB  (0010)         RXDATA:   DS      10h             ;Rx Data Area.

                               ;==============================================
                               ; Ch 1 Interrupt Service Routines
                               ;==============================================

      (4400)                   ORG     4400h           ;I-Page.

                               ;=======================
                               ; IM2 Indirect Jump Table
                               ;=======================

4400  0844                     DW      TBE1            ;Ch 1 TBE service
4402  2C44                     DW      EXSTAT1         ;Ch 1 EXSTAT service
4404  (0002)                   DS      2               ;Ch 1 RCA service (not used)
4406  (0002)                   DS      2               ;Ch 1 RXSPCL service (not used)

                               ;=======================
                               ; TBE1 Service
                               ;=======================

4408  F5             TBE1:     PUSH    AF              ;

4409  3ACA40                   LD      A,(EOMFLG)      ;Test end of message ram
440C  3C                       INC     A               ;flag.  If -1, Reset TBE
440D  2006                     JR      NZ,NOTEOM       ;Interrupts Pending and
440F  3E28                     LD      A,00101000b     ;exit.
4411  D343                     OUT     (SCMST1),A      ;
4413  1813                     JR      EXIT1           ;

4415  23             NOTEOM:   INC     HL              ;Bump Tx data pointer.
4416  7E                       LD      A,(HL)          ;Get character.
4417  FE24                     CP      '$'             ;Out of Tx data?
4419  2804                     JR      Z,EOM           ;Jump to EOM if so.
441B  D342                     OUT     (SDATA1),A      ;Send character and exit
441D  1809                     JR      EXIT1           ;if not.

441F  3EE8           EOM:      LD      A,11101000b     ;End of message, so reset
4421  D343                     OUT     (SCMST1),A      ;Tx underrun/EOM latch and
4423  3EFF                     LD      A,-1            ;reset TBE interrupts pending.
```

```
4425  32CA40                LD     (EOMFLG),A     ;CRC will be send when Tx
                                                  ;underrun occurs.  Set ram
                                                  ;end of message flag.

4428  F1          EXIT1:    POP    AF             ;
4429  FB                    EI                    ;
442A  ED4D                  RETI                  ;

                            ;======================
                            ; EXSTAT1 Service
                            ;======================

442C  F5          EXSTAT1:PUSH     AF             ;

442D  3E10                  LD     A,00010000b    ;Reset EXSTAT interrupts.
442F  D343                  OUT    (SCMST1),A     ;

4431  3ACA40                LD     A,(EOMFLG)     ;Test ram EOMFLG.  If
4434  3C                    INC    A              ;(EOMFLG)=0 then exit.
4435  2012                  JR     NZ,EXIT2       ;

4437  3EFF                  LD     A,0FFh         ;If (EOMFLG)=-1, then send
4439  D342                  OUT    (SDATA1),A     ;first 0FFh pad character,

443B  3E00        TBE?:     LD     A,0            ;loop until TBE status
443D  D343                  OUT    (SCMST1),A     ;is true,
443F  DB43                  IN     A,(SCMST1)     ;
4441  CB57                  BIT    2,A            ;
4443  28F6                  JR     Z,TBE?         ;

4445  3EFF                  LD     A,0FFh         ;then load second 0FFh pad
4447  D342                  OUT    (SDATA1),A     ;character and exit.

4449  F1          EXIT2:    POP    AF             ;
444A  FB                    EI                    ;
444B  ED4D                  RETI                  ;

444D  (4000)                END    BISYNC
```

Chapter 7

SDLC (HDLC) SERIAL I/O

Any combination of Quadart channels may be programmed for SDLC operation; the remaining channels may then be programmed for asynchronous or byte synchronous operation, or left unused. The steps required to initialize a Quadart channel for SDLC operation are presented in Section 7.1. Section 7.2 covers SDLC transmission and CRC generation. Section 7.3 covers reception and CRC checking; and SDLC residue codes are covered in Section 7.4. The chapter concludes with a complete SDLC programming example.

The byte synchronous and SDLC modes differ in most respects. The major differences are: (1) the SDLC mode is a bit oriented protocol which means that it can easily accommodate transparent data with variable data lengths and bit patterns, (2) the channel transmitter and receiver character lengths may, and often do differ, (3) the channel transmitter automatically inserts a single logic zero in the TxD data after five contiguous ones (except in flags or an SDLC abort sequence), and these inserted zeroes are automatically removed from the RxD data by the SIO receiver, (4) the programmed sync character **must** be the 01111110b flag, (5) flags are never loaded into the Rx data FIFO, (6) the SDLC mode has address recognition hardware which may be programmed to generate RCA interrupts only after a software selectable frame address is received, (7) the CRC-CCITT polynomial **must** be selected in SDLC mode for both the transmitter and the receiver, (8) the CRC generator and checker are preloaded with all ones when reset, and a special nonzero remainder is expected in the CRC checker, (9) the Reset CRC Checker command is not used in the SDLC mode since the CRC checker is automatically reset by any closing flag, (10) CRC checking is relatively easy since SIO hardware can generate an interrupt at the end of an SDLC frame, and the CRC Error bit is valid at this time, (11) parity checking is not used in SDLC mode, (12) SDLC mode supports break (abort) generation and detection, and (13) since SDLC is a bit as opposed to a character oriented mode, a hardware **residue code** is available to separate frame ending data bits from CRC bits in the received data. As in the byte synchronous mode, a X1 clock multiplier **must** be selected, and typically an attached modem generates $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clocks input to the SIO in SDLC mode.

The SDLC frame format is shown in Figure 39. This waveform is generated by the SIO transmitter at its $\overline{\text{TxD}}$

output pin, and it is expected by the SIO receiver at its $\overline{\text{RxD}}$ input pin. Each SDLC frame consists of an opening flag, an 8-bit address field, an integral number of data bits (zero is allowed, and the maximum number is limited only by the expected channel error rate), and a 16-bit CRC-CCITT character (optional, but required in SDLC protocol), followed by a closing flag. The 8-bit SDLC control field, which immediately follows the frame address, is transmitted and received as ordinary data. The closing flag of one frame may serve as the opening flag of the following frame provided the channel receiver is not placed in the hunt mode after receiving the closing flag. If the receiver is placed in the hunt mode, then another opening flag is needed to synchronize the receiver. The SIO may correctly receive shared zero flags (e.g., a zero, six ones, a zero, six ones, and so on), but it cannot transmit them. The SIO allows the transmitter to idle sending flags when temporarily out of data, but the SDLC protocol requires that flags only appear between frames (they are used to open and close frames), not in the middle of frames.

7.1 CHANNEL INITIALIZATION

After any POC or reset from the IOP, all Quadart interrupts (excepting pending PIO interrupts) are disabled, all CTC channels stop counting, all SIO transmitters and receivers are disabled, and modem output lines DTR and RTS are turned off (marking). Selected CTC and SIO channels may be forced to these states at any time by writing channel reset commands to these devices. The PIO controlled CY lines are in random states after a POC and are unchanged by a C-Bus reset.

From this state, a Quadart channel is configured for SDLC I/O by the following steps:

1. Disable IOP interrupts during Quadart initialization since spurious Quadart interrupts may occur as its interrupt circuitry is enabled. After the channel is properly initialized there should be no pending interrupts.

2. Initialize the PIO (see Section 2.2). Set channel bit ExtCk connecting modem supplied $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clocks to the channel's SIO clock inputs.

3. Initialize the channel SIO by outputting control bytes to WR0 through WR7. See Figure 40 for the SDLC configuration options and Appendix A for detailed bit descriptions. Register WR4, which defines the channel mode, must be programmed before registers WR1, WR3 and WR5, since these registers are used differently in each SIO mode.

4. If other Quadart channels are to be used, initialize them; if not, Quadart initialization is complete since all other channels are disabled by a POC or a reset.

At this point the SDLC channel is ready for operation. If interrupts are used, load the IOP Z80A I-register, select IM2 mode, and enable IOP interrupts (execute an EI instruction). The IOP transmits data characters by writing bytes to the channel SIO Data Port when status bit TBE is set; this status is determined by polling RR0, or by a TBE interrupt (the first TBE interrupt does not occur until the IOP loads a character into the Tx buffer and the buffer then becomes empty). The IOP receives data characters by reading bytes from the channel SIO Data Port when status bit RCA is set; this status is determined by polling RR0, or by an RCA interrupt -- see Section 4.2.

The recommended order for loading SIO registers is: (1) WR4 to define the SDLC mode, (2) WR1 to configure the channel interrupt behavior, (3) WR2 to define the base interrupt vector if any SIO channel interrupt is enabled, (4) WR6 to define the 8-bit SDLC address if control bit Address Search Mode is set, (5) WR7 with the SDLC 01111110b flag, (6) WR3 to configure and enable the channel receiver, and (5) WR5 to configure and enable the channel transmitter. Register WR5 also defines the CRC polynomial used by both the channel transmitter and receiver, and CRC-CCITT must be selected.

The complete range of SDLC mode SIO configuration options is illustrated in Figure 40. Note that some WR1 through WR7 bit select among available options while other **don't care** bits are conservatively programmed reset (function disabled).

A Channel Reset command should be issued to WR0 before loading WR1 through WR7. The Reset EXSTAT Interrupts command should be issued twice after if EXSTAT interrupts are enabled as explained in Section 4.5.

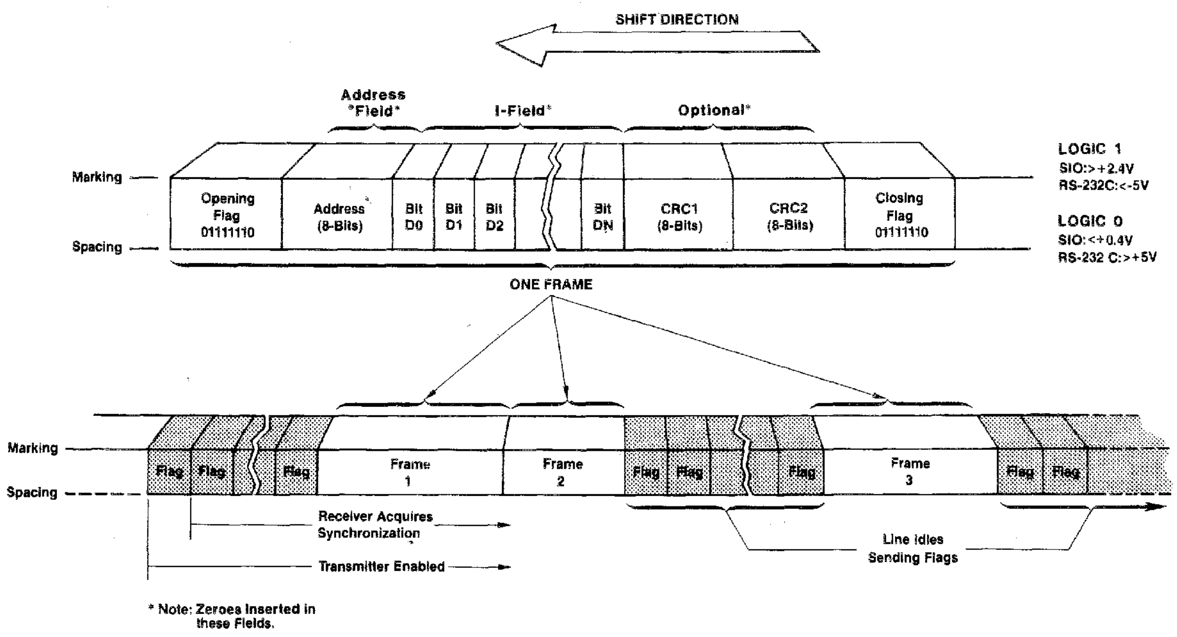


Figure 39: SDLC (HDLC) FRAME FORMAT

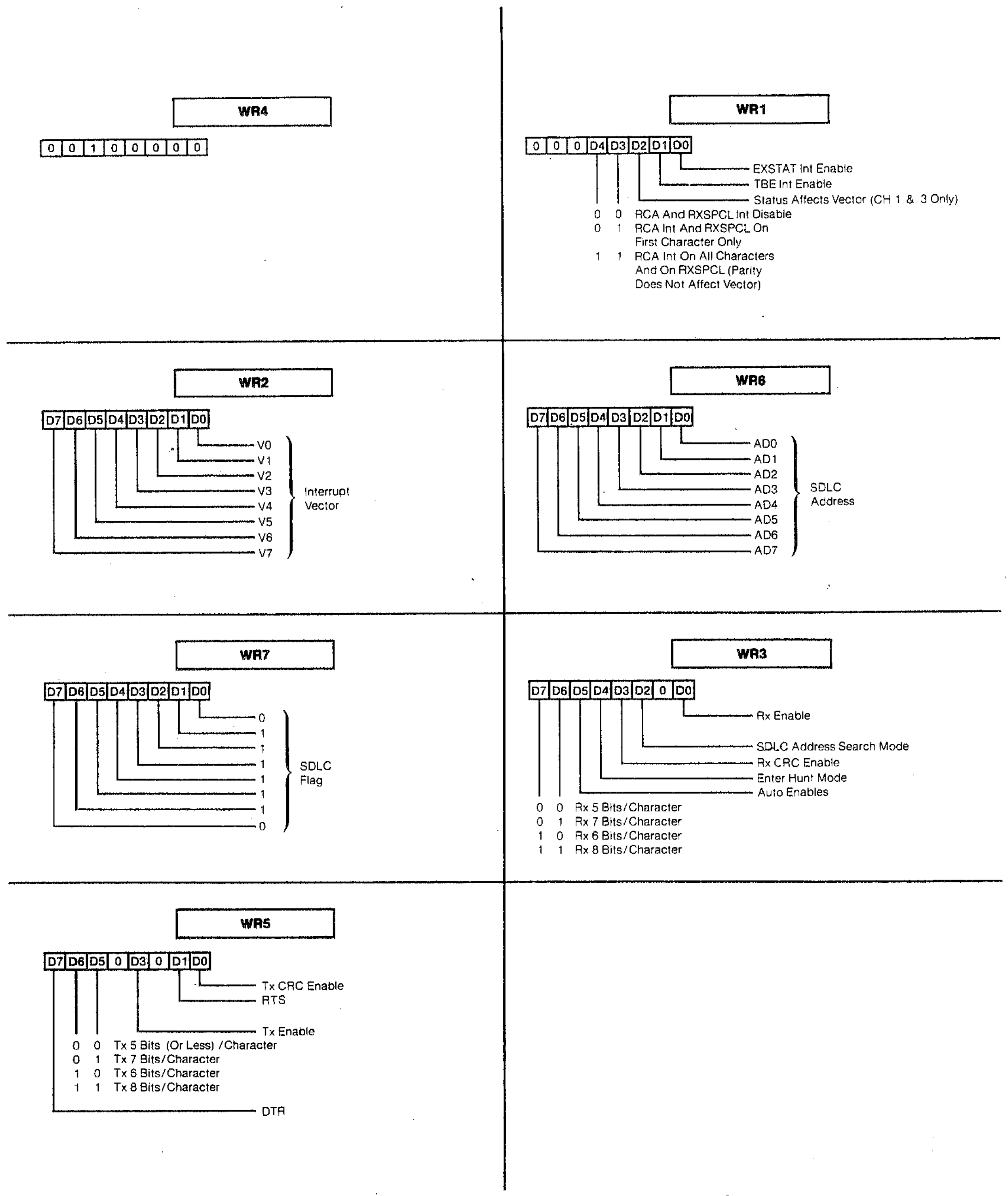


Figure 40: SIO SDLC MODE OPTIONS

Configuring a channel for SDLC operation automatically defines the EXSTAT group and RXSPCL group bit functions as shown in Table 9. A Reset Channel command resets all four RXSPCL bits and the two Reset EXSTAT Interrupts commands makes the EXSTAT bit group reflect current status.

Table 9: SDLC MODE RR0 AND RR1 BIT FUNCTIONS

Bit	RR0	Bit	RR1
D7	Abort Detect	D7	SDLC End Of Frame
D6	Tx Underrun/EOM	D6	CRC Error
D5	CTS	D5	Rx Overrun Error
D4	$\overline{\text{Sync}}$/Hunt	D4	Logic 0 (not used)
D3	DCD	D3	Residue Code 2
D2	TBE	D2	Residue Code 1
D1	Interrupt Pending	D1	Residue Code 0
D0	RCA	D0	Logic 1 (not used)

Section 7.4 gives an SDLC programming example which also discusses the SIO initialization steps.

7.2 SDLC TRANSMIT/CRC GENERATION

After the SIO transmitter is initialized but before it is enabled, line $\overline{\text{TxD}}$ is held marking. When enabled the channel transmitter begins sending flags. It continues sending flags until the frame address is loaded into the transmitter for shifting out, the transmitter is disabled, an abort sequence is sent, or until control bit Send Break is set.

Control bit Send Break is active, although it is normally reset at all times in SDLC mode. If bit Send Break is set the line TxD is unconditionally forced spacing until the bit is reset. The Send Abort command may be issued at any time the transmitter is enabled to force line TxD line to send from 8 to 13 ones (marking) in succession, followed by flags. The TxD line continues marking if the command is issued repeatedly. If the command is issued to an empty transmitter then an abort sequence is sent, status bit TBE is set before, during and after the abort sequence, and no TBE interrupt results. If the command is issued while data is being shifted out of the transmitter, then the abort sequence is sent, the Tx buffer and the transmitter are cleared, status bit TBE is set, and a TBE interrupt is generated, if it is enabled.

The IOP transmits characters by writing bytes to the channel SIO Data Port which directly feeds a Tx buffer. Parity is active in the SDLC mode, but it is not used in SDLC protocol, so control bit Parity Enable should be permanently reset. The bits shifted out are: D0 (LSB), D1, ... , DN (MSB), in that order. Modem line TxD is spacing for logic 0 data and marking for logic 1 data. Data bits are shifted out at the SIO $\overline{\text{TxC}}$ input rate on the $\overline{\text{TxC}}$ falling edge.

Character bits must be loaded into the channel SIO Data Port right justified if the programmed Tx character length is 6 or 7 bits. If the programmed Tx character length is 5 bits or less, then the unused high order bits must be formatted in a special way. The required format is detailed in Appendix A under the WR5 bit descriptions. Programming a new Tx character length does not affect any character in the process of being shifted out of the transmitter; the Tx character length is sampled as a character is loaded from the Tx buffer into the transmitter.

A flowchart of interrupt driven SDLC transmission is shown in Figure 41 assuming TBE and EXSTAT interrupts are enabled. The flowchart begins after a POC by initializing the channel and enabling the channel transmitter which then starts sending flags. After a pause allowing the receiver to be synchronized (two or more flags should be sent before the frame address), the CRC generator is reset to all ones. Control bit Tx CRC Enable is permanently set since all SDLC frame characters are included in the calculation. Since the Tx character length is sampled as a character is transferred from the Tx buffer to the transmitter itself, the Tx character length is first set to 8 bits, and the frame address is then loaded into the Tx buffer.

A Reset Tx Underrun/EOM Latch command is then issued to WR0. If the Tx Underrun/EOM latch is reset **and** control bit Tx CRC Enable is set as the transmitter underruns, then a CRC character followed by flags are automatically sent by the transmitter; only flags are sent after an underrun if either of these conditions are not satisfied. If the Reset Tx Underrun/EOM Latch command is issued when the transmitter has already underrun (empty), then CRC followed by flags are immediately

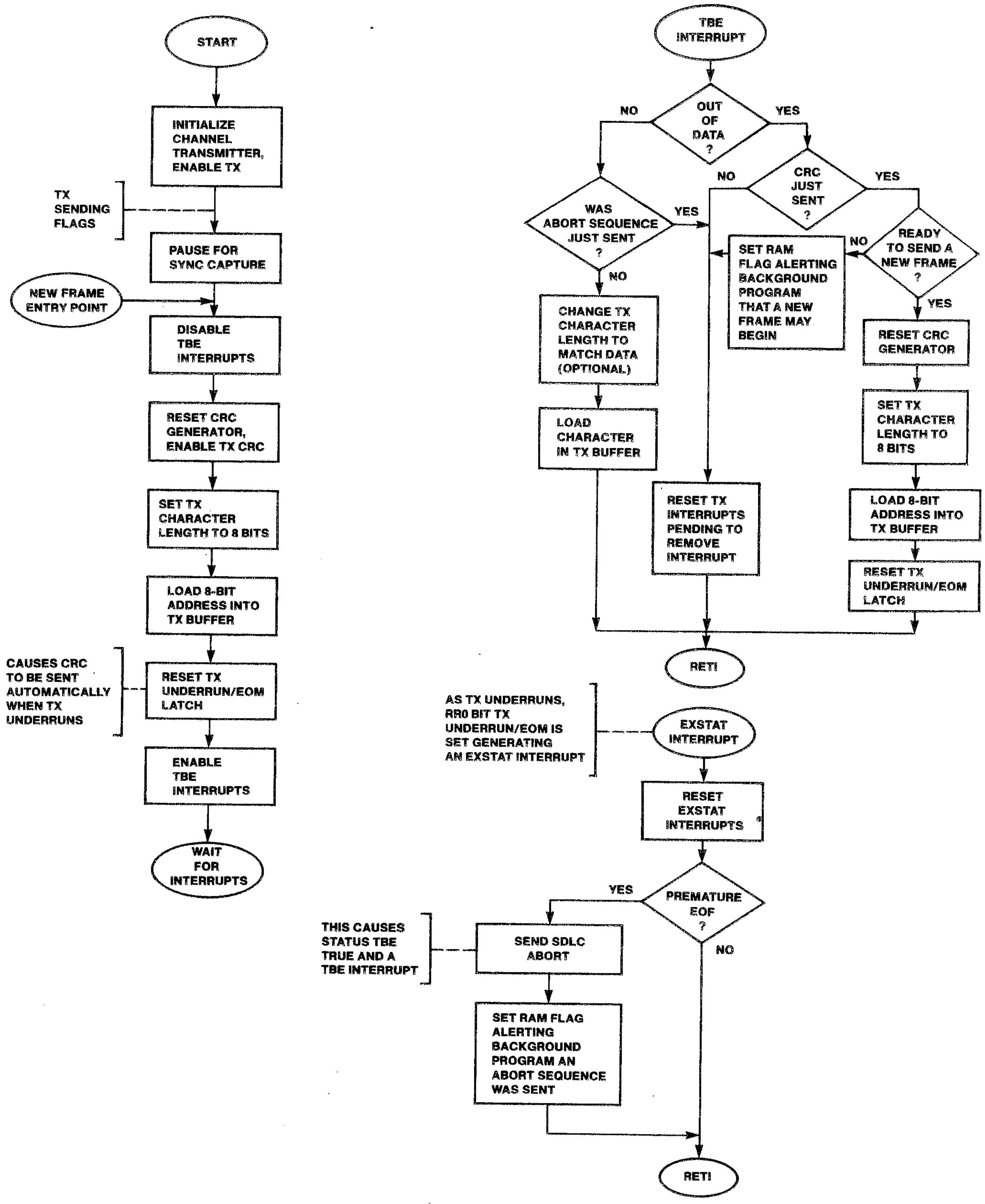

START
INITIALIZE CHANNEL TRANSMITTER, ENABLE TX
TX SENDING FLAGS
PAUSE FOR SYNC CAPTURE
NEW FRAME ENTRY POINT
DISABLE TBE INTERRUPTS
RESET CRC GENERATOR, ENABLE TX CRC
SET TX CHARACTER LENGTH TO 8 BITS
LOAD 8-BIT ADDRESS INTO TX BUFFER
CAUSES CRC TO BE SENT AUTOMATICALLY WHEN TX UNDERRUNS
RESET TX UNDERRUN/EOM LATCH
ENABLE TBE INTERRUPTS
WAIT FOR INTERRUPTS
TBE INTERRUPT
OUT OF DATA ?
NO
YES
CRC JUST SENT ?
NO
YES
WAS ABORT SEQUENCE JUST SENT ?
YES
NO
READY TO SEND A NEW FRAME ?
NO
YES
SET RAM FLAG ALERTING BACKGROUND PROGRAM THAT A NEW FRAME MAY BEGIN
CHANGE TX CHARACTER LENGTH TO MATCH DATA (OPTIONAL)
LOAD CHARACTER IN TX BUFFER
RESET TX INTERRUPTS PENDING TO REMOVE INTERRUPT
RESET CRC GENERATOR
SET TX CHARACTER LENGTH TO 8 BITS
LOAD 8-BIT ADDRESS INTO TX BUFFER
RESET TX UNDERRUN/EOM LATCH
RETI
AS TX UNDERRUNS, RR0 BIT TX UNDERRUN/EOM IS SET GENERATING AN EXSTAT INTERRUPT
EXSTAT INTERRUPT
RESET EXSTAT INTERRUPTS
PREMATURE EOF ?
YES
NO
THIS CAUSES STATUS TBE TRUE AND A TBE INTERRUPT
SEND SDLC ABORT
SET RAM FLAG ALERTING BACKGROUND PROGRAM AN ABORT SEQUENCE WAS SENT
RETI

Figure 41: SDLC TRANSMIT FLOWCHART

sent. Consequently, the underrun latch must be reset **after** at least one character is loaded in the Tx buffer to prevent an immediate CRC from being sent. The latch should also be reset as soon as possible (after loading the address field) to give the IOP the most time to react to a premature Tx underrun and send an abort sequence before any closing flags are sent.

TBE interrupts are disabled before the address field is loaded and they are not enabled until after the Tx Underrun/EOM latch is reset. This guarantees the latch is reset before any possible Tx underrun. After enabling TBE interrupts, the background program waits for the first TBE interrupt signaling the SDLC control field may be loaded and sent.

A TBE interrupt may result from any one of the following conditions: (1) the Tx buffer is empty after loading a data character into the transmitter, (2) the Tx buffer is empty as a result of issuing an SDLC Abort command which clears both the Tx buffer and the transmitter, or (3) a flag has just been loaded into the transmitter after CRC has been sent. Status bit TBE would change from reset to set as each of these conditions occur in polled environments.

The first condition occurs during ordinary data transmission, or at the planned end of an SDLC frame. If there are more data to send, the Tx character length is optionally changed and a character is loaded into the Tx buffer for transmission (the Tx character length might be changed when switching from an 8-bit control field to 5-bit I-field data characters, for example). The Tx character length should be changed before the character is loaded into the Tx buffer, but after the previous character has cleared the buffer (after a TBE interrupt). If there are no more data to send, a Reset TBE Interrupt Pending command is issued which removes the TBE interrupt and allows the transmitter to underrun without generating further TBE interrupts. CRC is automatically sent since the Tx Underrun/EOM latch is reset as the transmitter underruns, and status bit TBE is reset while CRC is shifting out.

The following situation would typically lead to the second type of TBE interrupt. With higher priority interrupts, or while interrupt are temporarily disabled, a TBE interrupt is not serviced until the transmitter has already underrun. This sets bit Tx Underrun/EOM which generates an EXSTAT interrupt, but EXSTAT servicing is held off until TBE servicing is completed since it has a lower interrupt priority. So the TBE service routine proceeds as normal to load data into the

Tx buffer, unaware of the underrun. This data is placed in the Tx buffer as the transmitter is shifting out CRC. An EXSTAT interrupt then occurs after a return from the TBE service routine, prompting the EXSTAT service routine to issue an SDLC abort sequence which overwrites the CRC being shifted out with from 18 to 13 consecutive ones. The abort command also clears the Tx buffer and transmitter which sets status bit TBE and generates a TBE interrupt belonging to the second category mentioned above. Here the TBE service routine merely issues a Reset TBE Interrupt Pending command to WR0 to remove the TBE interrupt, and the background program initiates abort recovery.

While CRC is being sent status bit TBE is reset, but after it is completely sent, a single flag is automatically loaded into the transmitter. At this time status bit TBE is set which leads to the third case mentioned above. The TBE service routine may then either issue a Reset TBE Interrupts Pending command to WR0 if there are no more data to send, or it may immediately begin sending another frame after resetting the CRC generator.

A transmitter underrun generates an EXSTAT interrupt provided EXSTAT interrupts are enabled and a Reset Tx Underrun/EOM Latch command has been issued. If the underrun is planned, then the EXSTAT service routine merely resets EXSTAT interrupts and returns. If the underrun is premature, then a Send SDLC Abort command is issued to WR0 as discussed above. The planned/premature EOF status may easily be managed with a RAM flag controlled from the TBE service routine.

An SDLC transmission is terminated as follows: (1) after the last data is sent the TBE service routine issues a Reset TBE Interrupts Pending command which removes the TBE interrupt and allows the transmitter to underrun, (2) the transmitter underruns which generates an EXSTAT interrupt and CRC begins shifting out, (3) after CRC is shifted out and a flag is loaded into the transmitter, a TBE interrupt occurs. At this point the transmitter may be disabled since the last flag is automatically sent followed by a marking TxD line.

The channel transmitter is disabled by resetting control bit Tx Enable, or by an off CTS line while control bit Auto Enables is set. If the transmitter is disabled while a data character or a flag is being shifted out, then the character is completely transmitted followed by a marking TxD line. If the transmitter is disabled as CRC is shifting out, then a full 16 bits are sent, but flag bits replace CRC bits.

7.3 SDLC RECEIVE/CRC CHECKING

Initializing and enabling the channel receiver forces it into the hunt mode, and it remains there until at least one flag is received. After one flag is received status bit $\overline{\text{Sync}}$/Hunt changes from set to a reset, and this transition generates an EXSTAT interrupt, if enabled. After sync capture, received data are assembled and transferred to the Rx FIFO until the receiver is placed in the hunt mode by setting control bit Enter Hunt Mode, or the receiver is disabled. Flags are never transferred to the Rx FIFO in SDLC mode. The channel receiver automatically removes any inserted zeroes from the address field, the I-Field, and the CRC character.

If control bit SDLC Address Search Mode is set, the receiver begins loading characters into the Rx FIFO only after an address matching that programmed into WR6, or after the 11111111b global address is received (the eight bit address must immediately follow a flag for address recognition to occur). If control bit Sync Character Load Inhibit is set, then received characters matching the programmed value of WR6 are inhibited from being loaded into the Rx FIFO. This bit should be permanently reset in SDLC mode to force all received characters (except flags) to be loaded into the Rx FIFO. The SIO address recognition circuitry functions properly regardless of the programmed Rx character length, although it is most convenient to set the Rx character length to 8 bits until both the 8-bit address and control fields have been received. If control bit Address Search Mode is reset, then the receiver begins assembling and loading data into the Rx FIFO beginning with the first nonflag which follows one or more flags.

Once received data transfer begins, all address field bits, control field bits and I-Field bits are routed to the received data FIFO for reading provided control bit Sync Character Load Inhibit is reset. An incomplete version of the CRC character is also loaded into the Rx FIFO (the receiver stops assembling the CRC character two bits before it is completely in the receiver in SDLC mode). The receiver assembles the address field right justified with the address LSB in bit position D0. When the Rx character length is subsequently changed to value N, the receiver merely shifts in the next N bits to form a character which may result in data which is **not** right justified, depending upon the time at which the change is made. See Section 4.4 for a discussion of this topic. After each character is completely assembled it is transferred to the Rx FIFO for reading from the channel SIO Data Port, and status bit RCA is set. Eight bits are always read from the SIO Data Port, and if the programmed Rx character length is 8 bits, the byte consists of bits D7 (MSB) on the left through D0 (LSB) on the right if the data has been right justified. If the Rx character length is 7 bits or less and the data is right justified, then the high order bit or bits (D7, D6 and D5) of the read data consists of the leading (LSB) bits of the following character.

As each character is assembled and loaded into the Rx FIFO, three RXSPCL bits are parallel loaded into the error FIFO: SDLC End Of Frame, CRC Error, and Rx Overrun Error. A set SDLC End Of Frame (EOF) bit is loaded into the error FIFO when a closing flag is detected by a sync register in front of the channel receiver. At the moment set EOF status is loaded into the error FIFO: (1) an incomplete version of the trailing CRC bits is simultaneously loaded into the Rx FIFO, (2) the CRC calculation is complete and the result is loaded as status bit CRC Error, and (3) the three SDLC residue code bits are valid and available for reading from RR1. If a reset EOF status bit is loaded, then status bit CRC Error and the three bit residue code are invalid.

If three characters are stacked unread in the receive data FIFO as a fourth character is transferred from the receiver to the FIFO, then the character on the bottom of the Rx FIFO is overwritten, and set status bit Rx Overrun Error is loaded into the bottom of the error FIFO opposite the overwritten character. The fourth character's SDLC EOF and CRC Error status overwrite the third character's status on the bottom of the error FIFO also. Received data which does not overwrite the bottom FIFO character in this way causes a reset Rx Overrun Error bit to be loaded into the error FIFO. Refer to Section 4.3 for more information on the data and error FIFOs.

In contrast to the byte synchronous mode, the receiving channel is not forewarned that an EOF is imminent by control characters imbedded in the I-field. Thus the receiver cannot differentiate between CRC bits and data bits at the time they are received, so CRC bits are typically stored as ordinary data in a RAM buffer until an EOF occurs. When an EOF does occur the SIO provides both valid CRC Error status and a 3-bit **residue code** which is read from RR1. The residue code defines the boundary between data bits and CRC bits in the previously received data, allowing the IOP program to go back to the RAM buffer data and separate the two. Section 7.4 describes how to use the residue bits when the received data has been right justified in the manner described in Section 4.4.

Any SDLC frame may be prematurely aborted when the receiver detects an abort sequence consisting of seven or more consecutive ones in the RxD data. This condition sets status bit Break/Abort (it does not set status bit RCA, however), and when the continuous ones terminate, bit Break/Abort is reset. If EXSTAT interrupts are enabled, then the reset to set transition in status bit Break/Abort generates an EXSTAT interrupt. The interrupt should be removed and the Break/Abort bit freed (unlatched) by issuing a Reset EXSTAT Interrupts command in the service routine. When status bit Break/Abort later changes from set to reset when the abort sequence terminates, another EXSTAT interrupt occurs and again the Reset EXSTAT command should be issued. The background program should be altered and the frame aborted so it can flush the RAM data buffer and initiate recovery. Note that status bit Break/Abort should also be set when the RxD line begins marking due to a disabled transmitter; but in this case status bit Abort is not later reset. This allows the receiving station to differentiate between an abort sequence and a disabled transmitter.

The flowchart for receiving a SDLC interrupt driven message is shown in Figure 42. The flowchart assumes that mode Rx Interrupt On All Characters (Parity Does Not Affect Vector) and EXSTAT Interrupts are enabled. The channel receiver is initialized with an 8-bit Rx character length anticipating the 8-bit address field, and since all frame characters are included in the CRC calculation, control bit Rx CRC Enable is permanently set. Enabling the receiver places it in the hunt mode.

The first RCA interrupt is generated when the address field is loaded into the Rx FIFO. If control bit Address Search Mode is reset, then the frame may either be accepted, or it may be rejected by setting control bit Enter Hunt Mode which forces the receiver into the hunt mode. If control bit Address Search Mode is set, then the first RCA interrupt is generated by an address matching WR6, or global 11111111b address in the data FIFO. In all cases the address should be read and discarded.

Once the address field, control field and first I-Field character are received, the Rx character length may be changed before the next RCA interrupt. Thereafter, data and CRC characters are stored in a RAM buffer until an RXSPCL interrupt occurs with status bit End Of Frame set. The RXSPCL service routine then samples status bit CRC Error, and if the bit is reset (no CRC Error), then the residue code bits are used to separate data from CRC bits in the RAM buffer (see following section).

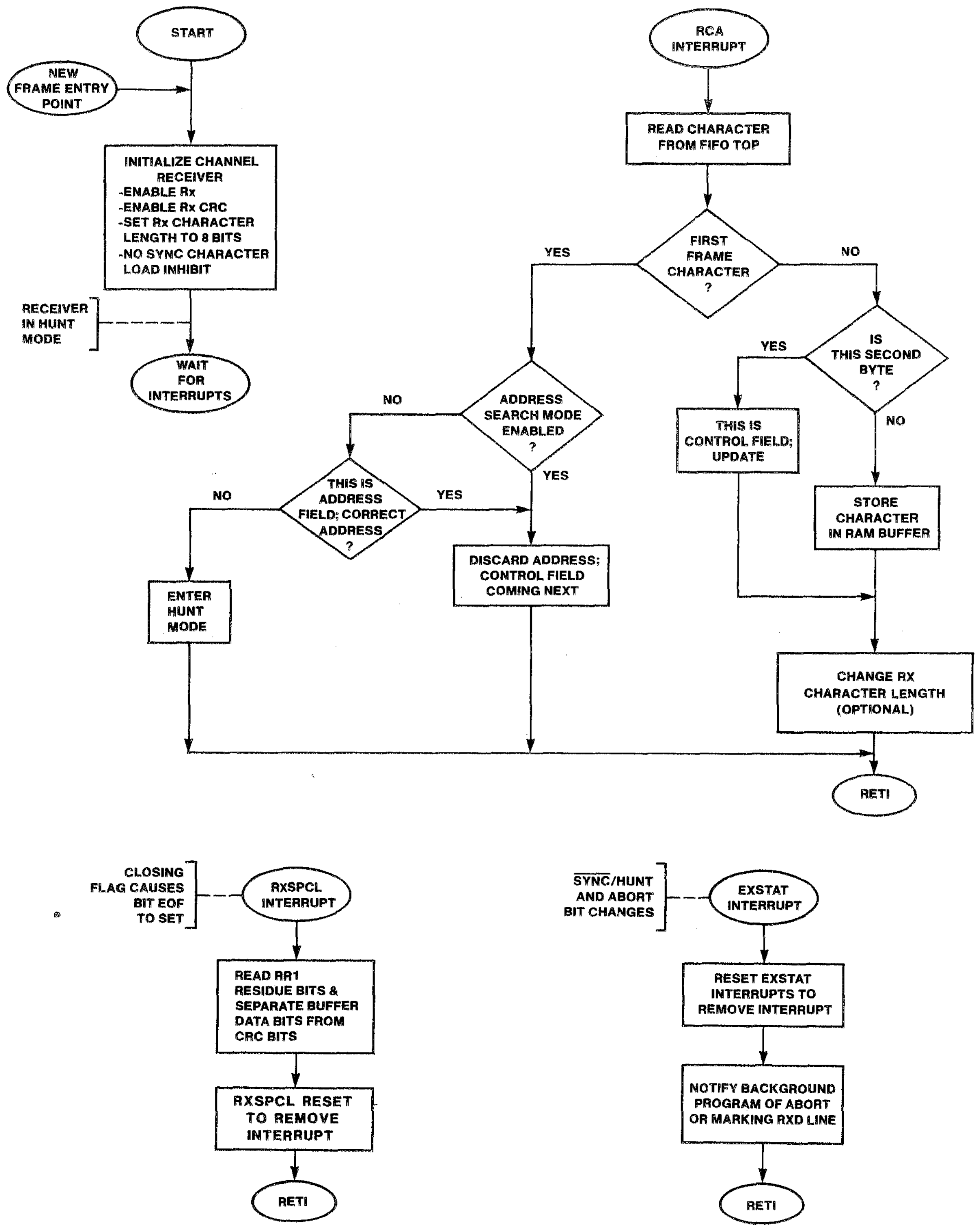

START
NEW FRAME ENTRY POINT
INITIALIZE CHANNEL RECEIVER
-ENABLE Rx
-ENABLE Rx CRC
-SET Rx CHARACTER LENGTH TO 8 BITS
-NO SYNC CHARACTER LOAD INHIBIT
RECEIVER IN HUNT MODE
WAIT FOR INTERRUPTS
RCA INTERRUPT
READ CHARACTER FROM FIFO TOP
FIRST FRAME CHARACTER ?
YES
NO
ADDRESS SEARCH MODE ENABLED ?
NO
YES
THIS IS ADDRESS FIELD; CORRECT ADDRESS ?
NO
YES
ENTER HUNT MODE
DISCARD ADDRESS; CONTROL FIELD COMING NEXT
IS THIS SECOND BYTE ?
YES
NO
THIS IS CONTROL FIELD; UPDATE
STORE CHARACTER IN RAM BUFFER
CHANGE RX CHARACTER LENGTH (OPTIONAL)
RETI
CLOSING FLAG CAUSES BIT EOF TO SET
RXSPCL INTERRUPT
READ RR1 RESIDUE BITS & SEPARATE BUFFER DATA BITS FROM CRC BITS
RXSPCL RESET TO REMOVE INTERRUPT
RETI
SYNC/HUNT AND ABORT BIT CHANGES
EXSTAT INTERRUPT
RESET EXSTAT INTERRUPTS TO REMOVE INTERRUPT
NOTIFY BACKGROUND PROGRAM OF ABORT OR MARKING RXD LINE
RETI

Figure 42: SDLC RECEIVE FLOWCHART

The EXSTAT service routine is entered when status bit $\overline{\text{Sync}}$/Hunt changes state (reset when an opening flag is received, and set when the receiver is forced to the hunt mode by setting control bit Enter Hunt Mode), and also when status bit Break/Abort changes state. In these cases, a Reset EXSTAT Interrupts command should be issued to remove the interrupt, and to free the EXSTAT bits to reflect current status.

The channel receiver is disabled either by resetting control bit Rx Enable, or by an off DCD modem line if control bit Auto Enables is set. When the receiver is disabled, the character currently being assembled is lost, but any characters and status already in the FIFOs may be correctly read. If synchronization is lost, or if the frame is not wanted, the receiver may be placed in the hunt mode at any time by setting control bit Enter Hunt Mode. This sets status bit $\overline{\text{Sync}}$/Hunt which, if it is enabled, generates an EXSTAT interrupt. The interrupt should be removed by a Reset EXSTAT Interrupts command. When the receiver later detects an opening flag another EXSTAT interrupt occurs as status bit $\overline{\text{Sync}}$/Hunt is reset. This interrupt should be handled in the same way.

7.4 SDLC RESIDUE CODES

Since SDLC is a bit oriented protocol, the Tx character length is typically chosen to match the conveniently available data length (from 1 to 8 bits). The Rx character length may, or may not be equal to the Tx character length depending on the application; data may be gathered and loaded into the transmitter as 4-bit nibbles, and read from the receiver as 8-bit bytes, for example. As the transmitter underruns, a 16-bit CRC character is appended to the outgoing bit stream, followed by at least one flag. The CRC character bits are read as ordinary data by the receiver since there is no way to differentiate between data and CRC until an after-the-fact flag has arrived. When the closing flag does arrive, the receiver knows that the previous 16 bits were CRC, and those before it were data. From this fact, and from the current Rx character length, a 3-bit residue code is calculated and made available at RR1 when status bit End Of Frame is set. This code allows the IOP to go back to the previously stored CRC and data bits and separate the two (all data and CRC bits have been loaded into the Rx FIFO for reading before status bit End Of Frame is set).

The residue codes and the corresponding data/CRC boundaries are listed in Table 10 for each Rx character length assuming the Rx data have been assembled **right justified** (see Section 4.4). The table also assumes that the following I-Field bit stream precedes the CRC character (inserted zeroes are not shown), where each capital letter represents one data bit, and bit Z is received last.

```
              <-- Shift Direction <--

    ...  KLMNOPQRSTUVWXYZ <16-BIT CRC> <FLAG> ...
```

The trailing data bits shown could be transmitted by loading an SIO transmitter set to 8 bits/character with byte <RQPONMLK> followed by byte <ZYXWVUTS>, since least significant bits are shifted out of the transmitter first.

The last four characters read from the received data FIFO before an End Of Frame are listed in the table, and they are headed LAST-3, LAST-2, LAST-1 and LAST, where LAST is the byte read with status bit End Of Frame set. Note that eight bits are always read from the FIFO, but depending on the selected Rx character length, from zero to three higher order bits should be ignored. These bits, along with the CRC bits, are indicated with a dash (-) character in the table. The residue code bits are tabulated in the same order that they are read from RR1; Residue Code 2, Residue Code 1, Residue Code 0.

Table 10: SDLC RESIDUE CODES

LAST - 3	LAST - 2	LAST - 1	LAST	Residue Code
76543210	76543210	76543210	76543210	
8-BIT RX CHARACTER LENGTH:				
RQPONMLK	ZYXWVUTS	--------	--------	011
QPONMLKJ	YXWVUTSR	-------Z	--------	111
PONMLKJI	XWVUTSRQ	------ZY	--------	000
WVUTSRQP	-----ZYX	--------	--------	100
VUTSRQPO	----ZYXW	--------	--------	010
UTSRQPON	---ZYXWV	--------	--------	110
TSRQPONM	--ZYXWVU	--------	--------	001
SRQPONML	-ZYXWVUT	--------	--------	101
7-BIT RX CHARACTER LENGTH:				
-SRQPONM	-ZYXWVUT	--------	--------	011
-RQPONML	-YXWVUTS	-------Z	--------	000
-XWVUTSR	------ZY	--------	--------	100
-WVUTSRQ	-----ZYX	--------	--------	010
-VUTSRQP	----ZYXW	--------	--------	110
-UTSRQPO	---ZYXWV	--------	--------	001
-TSRQPON	--ZYXWVU	--------	--------	101
6-BIT RX CHARACTER LENGTH:				
--TSRQPO	--ZYXWVU	--------	--------	000
--YXWVUT	-------Z	--------	--------	100
--XWVUTS	------ZY	--------	--------	010
--WVUTSR	-----ZYX	--------	--------	110
--VUTSRQ	----ZYXW	--------	--------	001
--UTSRQP	---ZYXWV	--------	--------	101
5-BIT RX CHARACTER LENGTH:				
---ZYXWV	--------	--------	--------	100
---YXWVU	-------Z	--------	--------	010
---XWVUT	------ZY	--------	--------	110
---WVUTS	-----ZYX	--------	--------	001
---VUTSR	----ZYXW	--------	--------	000

Example

(A) Assume the Rx character length is set to 6 bits, and the following data have been read from the Rx FIFO when an SDLC End Of Frame occurs and residue code 100 is read: ... 33h, C8h, ABh, 28h, 1Dh (received last).

The table indicates that 28h and 1Dh should be discarded, and the LSB of byte ABh is the last I-Field data bit. When the two high order bits of the previous bytes are masked off, these bytes then represent ending I-Field data ... 110011b, 001000b, 1b.

(B) Assume the Rx character length is set to 5 bits, and the following data have been read from the Rx FIFO when an SDLC End Of Frame occurs and residue code 000 is read; ... 33h, 91h, 2Ch, B6h, D2h (received last). The table indicates that B6h and D2h should be discarded, and the four low order bits of 2Ch are the last I-Field data bits. When the three high order bits of the previous bytes are masked off, these bytes then represent ending I-Field data ... 10011b, 10001b, 1100b.

7.5 AN SDLC PROGRAMMING EXAMPLE

In overview, the following SDLC example program connects the Quadart channel 0 transmitter to the channel 3 receiver using the loopback system. The interrupt driven transmit channel then sends a short SDLC frame. The address field is transmitted as a single 8-bit character. The Tx character length is then switched to 5 bits or less/character, and the I-Field data is then formatted and transmitted as several 3-bit (single digit octal) characters. The polled receive channel is initialized to 8 bits/character for the address field and the first I-Field character, then is switched to 6 bits/character (two digit octal) thereafter. This delayed Tx character length switching results in right justified Rx data as explained in Section 4.4. No control field is used in this example. The receive channel stores each sequential character in a RAM buffer (including the address field), then checks CRC error status and SDLC residue bits when status End Of Frame is detected. If there is no CRC error and the expected SDLC residue bits are observed, a period character is sent to the host; if either a CRC error or incorrect residue bits are detected, an asterisk character is sent to the host. The receiver is then placed in the hunt mode, and the process is repeated indefinitely.

The program is subdivided into three major components: initialization, the main background routine, and a TBE interrupt foreground routine. In the initialization segment, the channel 0 CTC is made to feed the channel 0 SIO $\overline{\text{TxC}}$ input, and the channel 3 CTC is made to feed the channel 3 SIO $\overline{\text{RxC}}$ input (both at 9600 bps) since there are no modem supplied clocks. Channel 0 is initialized with its transmitter enabled and its receiver disabled. The transmit character length is set to 8 bits/character for sending the address field. As soon as the transmitter is enabled, it begins sending flags. Channel 3 is initialized with its transmitter disabled and its receiver enabled. The received character length is set to 8 bits/character anticipating the eight bit address field, and the receiver is put in the Address Search Mode. Z80A interrupt mode IM2 is then selected, an I-Page = 44h is chosen, and IOP interrupts are enabled. No interrupts immediately result since all interrupts are disabled in the initialization routine.

The main background program begins by polling the channel 3 receiver until it is synchronized (at least one flag received). The address field (33h) is then loaded into the Tx buffer and sent to the receiver. The Tx Underrun/EOM latch is then reset which causes CRC followed by a flag to be sent when the transmitter underruns at the end of the frame. The transmitter is then configured for interrupt driven data transmission by enabling channel 0 TBE interrupts. From this point until the transmitter underruns, all data is sent in response to channel 0 TBE interrupts.

The background program repeatedly polls the channel 3 receiver for RCA true status. The first two characters are received as 8-bit characters (the 8-bit address field and 6 I-field bits which are shifted in right justified), and all subsequent characters are received as 6-bit characters. Status EOF is checked before reading a character from the Rx FIFO. If status EOF is not true, then the character is stored sequentially with those previously received starting at RAM address RXBUFF. When status EOF is true, status bit CRC Error is checked, followed by the three SDLC residue bits. Since data is transmitted as 3-bit characters and received as 6-bit characters, then the expected residue bits are 000 if the number of 3-bit I-Field characters is even, and 110 if the number is odd (the sample Tx data consists of seven octal values, so residue 110 should be sampled in the program). If status bit CRC Error is reset and one of the two expected residue codes is sampled, then a period character is sent to the host; if either condition is not met, an asterisk is sent to the host. After reporting the frame transfer outcome to the host, the channel 3 receiver is placed in the hunt mode and its character length is switched back to 8 bits/character for the next pass. The background program then loops back and the process is repeated.

The TBE interrupt foreground routine is entered for the first time after the frame address has been loaded into the Tx buffer in the background routine, and the buffer becomes empty. At this point the transmitter character length is switched from 8 bits/character to 5 bits or less/character. The switch has an affect only on the first service routine entry; thereafter it is a null operation since the character length is already 5 or less bits/character. For normal I-Field data transmission the interrupt routine gets a 3-bit (right justified) octal value from the RAM Tx data area, formats it for 3 data bits/character transmission and loads it into the Tx buffer for sending. The Tx data is delimited by two $ characters (2424h). The TBE service routine issues a Reset TBE Interrupts command in response to the first delimiter and this allows the transmitter to underrun and automatically send CRC. When a closing flag is loaded into the transmitter another TBE interrupt results, the second delimiter is sampled, and a Reset TBE Interrupts command is again issued. At this point, the next TBE interrupt does not occur until the address field byte is loaded into the Tx buffer in the background program at the start of a new pass.

```
CROMEMCO Z80 Macro Assembler version 03.07

                  ;        SDLC PROGRAMMING EXAMPLE
                  ;
                  ; See manual text for program description.
                  ;

         (0040)   QBASE    EQU     40H             ;Quadart Base Address.
         (0040)   SDATA0   EQU     QBASE+00h       ;Ch 0 SIO Data Port.
         (0041)   SCMST0   EQU     QBASE+01h       ;Ch 0 SIO Command/Status Port.
         (0042)   SDATA1   EQU     QBASE+02h       ;Ch 1 SIO Data Port.
         (0043)   SCMST1   EQU     QBASE+03h       ;Ch 1 SIO Command/Status Port.
         (0046)   SDATA3   EQU     QBASE+06h       ;Ch 3 SIO Data Port.
         (0047)   SCMST3   EQU     QBASE+07h       ;Ch 3 SIO Command/Status Port.
         (0048)   PDAT01   EQU     QBASE+08h       ;Ch 0&1 PIO Data Port.
         (0049)   PCOM01   EQU     QBASE+09h       ;Ch 0&1 PIO Command Port.
         (004A)   PDAT23   EQU     QBASE+0Ah       ;Ch 2&3 PIO Data Port.
         (004B)   PCOM23   EQU     QBASE+0Bh       ;Ch 2&3 PIO Command Port.
         (004C)   CTC0     EQU     QBASE+0Ch       ;Ch 0 CTC Port.
         (0050)   CTC3     EQU     QBASE+10h       ;Ch 3 CTC Port.
         (0054)   CNTRL    EQU     QBASE+14h       ;Loopback Control Port.
         (0001)   HDATA    EQU     1               ;IOP <-> Host Data Port.
         (0002)   FLAGS    EQU     2               ;IOP Flags Port.

         (4000)            ORG     4000h           ;Start of IOP ram.

                           ;==============================================
                           ;  Initialization Routine
                           ;==============================================

4000  310060      SDLC:    LD      SP,6000h        ;Locate stack.

4003  3E47                 LD      A,01000111b     ;Ch 0 CTC to counter mode,
4005  D34C                 OUT     (CTC0),A        ;interrupts disabled.
4007  3E20                 LD      A,32            ;Time constant = 32 for
4009  D34C                 OUT     (CTC0),A        ;9600 bps with X1 multiplier.

400B  3E47                 LD      A,01000111b     ;Ch 3 CTC to counter mode,
400D  D350                 OUT     (CTC3),A        ;interrupts disabled.
400F  3E20                 LD      A,32            ;Time constant = 32 for
4011  D350                 OUT     (CTC3),A        ;9600 bps with X1 multiplier.

4013  0E49                 LD      C,PCOM01        ;Initialize Ch 0 PIO
4015  0603                 LD      B,PIOEND-PIOBEG ; (see block move
4017  215A40               LD      HL,PIOBEG       ;  data below).
401A  EDB3                 OTIR                    ;

401C  0E4B                 LD      C,PCOM23        ;Initialize Ch 3 PIO
401E  0603                 LD      B,PIOEND-PIOBEG ; (see block move
4020  215A40               LD      HL,PIOBEG       ;  data below).
4023  EDB3                 OTIR                    ;

4025  3E00                 LD      A,00000000b     ;Reset ExtCk 0 and 3 to logic 0
4027  D348                 OUT     (PDAT01),A      ;to connect CTC zero count out-
4029  D34A                 OUT     (PDAT23),A      ;puts to SIO TxC and RxC inputs.

402B  0E41                 LD      C,SCMST0        ;Initialize SIO Ch 0.
402D  060B                 LD      B,S0END-S0BEG   ; (See block move
402F  215D40               LD      HL,S0BEG        ;  data below).
4032  EDB3                 OTIR                    ;

4034  3E01                 LD      A,1             ;Point to Ch 1 WR1.
4036  D343                 OUT     (SCMST1),A      ; - disable all Ch 1 interrupts
4038  3E04                 LD      A,00000100b     ; - status affects vector
403A  D343                 OUT     (SCMST1),A      ;   (shared w/Ch 0)

403C  3E02                 LD      A,2             ;Point to Ch 1 WR2 (shared w/Ch 0).
403E  D343                 OUT     (SCMST1),A      ;Base Interrupt Vector = 00h.
4040  3E00                 LD      A,00000000b     ;Resulting IM2 Jump Table
4042  D343                 OUT     (SCMST1),A      ;assuming I-Page = 44h:
                                                   ;4400h -> Ch 0 TBE
                                                   ;4402h -> Ch 0 EXSTAT (not used)
                                                   ;4404h -> Ch 0 RCA (not used)
                                                   ;4406h -> Ch 0 RXSPCL (not used)

4044  0E47                 LD      C,SCMST3        ;Initialize SIO Ch 3.
4046  060D                 LD      B,S3END-S3BEG   ; (See block move
4048  216840               LD      HL,S3BEG        ;  data below).
404B  EDB3                 OTIR                    ;

404D  3E98                 LD      A,10011000b     ;Loopback Ch 0 TxD
404F  D354                 OUT     (CNTRL),A       ;to Ch 3 RxD.

4051  3E44                 LD      A,44h           ;Select I-Page.
4053  ED47                 LD      I,A             ;
```

```
4055  ED5E              IM      2               ;Select Interrupt Mode 2.
4057  FB                EI                      ;Enable IOP Interrupts.

4058  181B              JR      AROUND          ;Jump around block move
                                                ;initialization data.

                  ;======================
                  ;  PIO and SIO Block Move
                  ;  Initialization Data
                  ;======================

405A  CF        PIOBEG: DB      11001111b       ;Control mode.
405B  CC                DB      11001100b       ;Define PIO line directions.
405C  07                DB      00000111b       ;Disable PIO interrupts.
                PIOEND:

405D  18        S0BEG:  DB      00011000b       ;Reset Ch 0 command.

405E  04                DB      00000100b       ;Point to WR4.
405F  20                DB      00100000b       ; - X1 clock
                                                ; - SDLC mode
                                                ; - no parity

4060  01                DB      00000001b       ;Point to WR1.
4061  00                DB      00000000b       ; - disable all Ch 0 interrupts

4062  47                DB      01000111b       ;Point to WR7 and reset
4063  7E                DB      01111110b       ;CRC generator.
                                                ; - 01111110 flag

4064  15                DB      00010101b       ;Point to WR5 and reset
4065  69                DB      01101001b       ;EXSTAT interrupts.
                                                ; - 8 bit Tx characters.
                                                ; - break off
                                                ; - enable Tx
                                                ; - SDLC CRC
                                                ; - enable Tx CRC

4066  13                DB      00010011b       ;Point to WR3 and reset
4067  00                DB      00000000b       ;EXSTAT interrupts.
                                                ; - disable Rx
                S0END:

4068  18        S3BEG:  DB      00011000b       ;Reset Ch 3 command.

4069  04                DB      00000100b       ;Point to WR4.
406A  20                DB      00100000b       ; - X1 clock
                                                ; - SDLC mode
                                                ; - no parity

406B  01                DB      00000001b       ;Point to WR1.
406C  00                DB      00000000b       ; - disable all Ch 3 interrupts

406D  46                DB      01000110b       ;Point to WR6 and reset
406E  33                DB      00110011b       ;CRC checker.
                                                ; - Address Field = 33h

406F  07                DB      00000111b       ;Point to WR7.
4070  7E                DB      01111110b       ; - 01111110 flag

4071  15                DB      00010101b       ;Point to WR5 and reset
4072  00                DB      00000000b       ;EXSTAT interrupts.
                                                ; - SDLC CRC
                                                ; - disable Tx

4073  13                DB      00010011b       ;Point to WR3 and reset
4074  CD                DB      11001101b       ;EXSTAT interrupts.
                                                ; - 8 bit Rx characters
                                                ; - no auto enables
                                                ; - enable Rx CRC
                                                ; - enable address search mode
                                                ; - no inhibit sync load
                                                ; - Rx enable
                S3END:

                  ;==================================================
                  ;  Main Program
                  ;==================================================

4075  3E10      AROUND: LD      A,00010000b     ;Wait for Ch 3 receiver
4077  D347              OUT     (SCMST3),A      ;to aquire sync on first
4079  DB47              IN      A,(SCMST3)      ;pass only.
407B  CB67              BIT     4,A             ;
407D  20F6              JR      NZ,AROUND       ;

```

```
407F  0E02              AGAIN:   LD     C,2               ;Received character counter.
4081  DD213944                   LD     IX,RXBUFF         ;Initialize ram Tx and
4085  FD213044                   LD     IY,TXBUFF         ;Rx buffer pointers.

4089  F3                         DI                       ;
408A  3E81                       LD     A,10000001b       ;Point to Ch 0 WR1 and
408C  D341                       OUT    (SCMST0),A        ;reset Tx CRC generator.
408E  3E00                       LD     A,0               ;Disable Ch 0 TBE
4090  D341                       OUT    (SCMST0),A        ;interrupts.

4092  3E05                       LD     A,5               ;Point to Ch 0 WR5.
4094  D341                       OUT    (SCMST0),A        ;
4096  3E69                       LD     A,01101001b       ;Set Tx character length to
4098  D341                       OUT    (SCMST0),A        ;8 bits for address field.

409A  3E33                       LD     A,33h             ;Load and transmit address
409C  D340                       OUT    (SDATA0),A        ;field.

409E  3EC1                       LD     A,11000001b       ;Point to Ch 0 WR1 and reset
40A0  D341                       OUT    (SCMST0),A        ;Tx Underrun/EOM latch.
40A2  3E02                       LD     A,00000010b       ;Enable Ch 0 TBE
40A4  D341                       OUT    (SCMST0),A        ;interrupts.
40A6  FB                         EI                       ;

40A7  F3                RCA?:    DI                       ;
40A8  3E00                       LD     A,0               ;Poll until Ch 3 RCA status
40AA  D347                       OUT    (SCMST3),A        ;is true.
40AC  DB47                       IN     A,(SCMST3)        ;
40AE  CB47                       BIT    0,A               ;
40B0  28F5                       JR     Z,RCA?            ;
40B2  FB                         EI                       ;

40B3  F3                         DI                       ;
40B4  0D                         DEC    C                 ;Is first I-Field character
40B5  2008                       JR     NZ,NOTNOW         ;now in Rx FIFO?

40B7  3E03                       LD     A,3               ;If first I-Field character
40B9  D347                       OUT    (SCMST3),A        ;is now in Rx FIFO, change
40BB  3E89                       LD     A,10001001b       ;Rx character length to
40BD  D347                       OUT    (SCMST3),A        ;6 bits/character.

40BF  3E01              NOTNOW:  LD     A,1               ;Check for an EOF
40C1  D347                       OUT    (SCMST3),A        ;condition before
40C3  DB47                       IN     A,(SCMST3)        ;reading Rx FIFO.
40C5  FB                         EI                       ;
40C6  CB7F                       BIT    7,A               ;
40C8  2009                       JR     NZ,EOF            ;

40CA  DB46                       IN     A,(SDATA3)        ;If not EOF, read character
40CC  DD7700                     LD     (IX),A            ;and store it in ram buffer.
40CF  DD23                       INC    IX                ;
40D1  18D4                       JR     RCA?              ;

40D3  F5                EOF:     PUSH   AF                ;An EOF has been detected,
40D4  DB46                       IN     A,(SDATA3)        ;so read and discard last
40D6  F1                         POP    AF                ;CRC bits from Rx FIFO.
40D7  CB77                       BIT    6,A               ;First check for a
40D9  200A                       JR     NZ,ERROR          ;CRC error,
40DB  E60E                       AND    00001110b         ;then for either residue
40DD  FE00                       CP     00000000b         ;code 000 ...
40DF  2808                       JR     Z,OKAY            ;
40E1  FE0C                       CP     00001100b         ;or residue code 110.
40E3  2804                       JR     Z,OKAY            ;

40E5  1E2A              ERROR:   LD     E,'*'             ;Report outcome to host.
40E7  1802                       JR     TOHOST            ;If (no CRC error) and (residue
40E9  1E2E              OKAY:    LD     E,'.'             ;000 or 110), then success
40EB  DB02              TOHOST:  IN     A,(FLAGS)         ; => send a period to the host.
40ED  CB7F                       BIT    7,A               ;If (CRC error) or not (residue
40EF  28FA                       JR     Z,TOHOST          ;000 or 110), then error
40F1  7B                         LD     A,E               ; => send an asterisk to the
40F2  D301                       OUT    (HDATA),A         ;host.

40F4  F3                         DI                       ;
40F5  3E03                       LD     A,3               ;Point to Ch 3 WR3.
40F7  D347                       OUT    (SCMST3),A        ;Enter hunt mode and set
40F9  3EDD                       LD     A,11011101b       ;Rx character length to
40FB  D347                       OUT    (SCMST3),A        ;8 bits/char for address field.
40FD  FB                         EI                       ;

40FE  C37F40                     JP     AGAIN             ;Repeat.

                                 ;==============================================
                                 ;  Ch 0 Interrupt Service Routines
                                 ;==============================================

      (4400)              ORG     4400h           ;I-Page.

                          ;=======================
                          ; IM2 Indirect Jump Table
                          ;=======================

4400  (0008)              DS      8               ;No Ch 1 service.

4408  1044                DW      TBE0            ;Ch 0 TBE service
440A  (0002)              DS      2               ;Ch 0 EXSTAT service (not used)
440C  (0002)              DS      2               ;Ch 0 RCA service (not used)
440E  (0002)              DS      2               ;Ch 0 RXSPCL service (not used)

                          ;=======================
                          ; TBE0 Service
                          ;=======================

4410  F5          TBE0:   PUSH    AF              ;Save AF on stack.

4411  3E05                LD      A,5             ;Point to WR5.
4413  D341                OUT     (SCMST0),A      ;Change Tx character
4415  3E09                LD      A,00001001b     ;length to 5 bits or
4417  D341                OUT     (SCMST0),A      ;less.

4419  FD7E00              LD      A,(IY)          ;Get a Tx character from ram
441C  FD23                INC     IY              ;buffer and move pointer.  If
441E  FE24                CP      '$'             ;character is '$' delimiter,
4420  2806                JR      Z,WRAPUP        ;then don't send it.

4422  F6C0                OR      11000000b       ;If character is not '$'
4424  D340                OUT     (SDATA0),A      ;delimiter, send it to
4426  1804                JR      EXIT            ;Ch 3 receiver.

4428  3E28        WRAPUP: LD      A,00101000b     ;Reset TBE Interrupts
442A  D341                OUT     (SCMST0),A      ;Pending command.

442C  F1          EXIT:   POP     AF              ;Restore AF, enable IOP
442D  FB                  EI                      ;interrupts and return.
442E  ED4D                RETI                    ;

                          ;=======================
                          ; Rx and Tx ram buffers
                          ;=======================

4430  07000601    TXBUFF: DB      7q,0q,6q,1q,5q  ;Transmit data =
4435  02042424            DB      2q,4q,'$$'      ;seven octal values and
                                                  ;termination characters.

4439  (0020)      RXBUFF: DS      20h             ;Rx ram buffer area.

4459  (4000)              END     SDLC
```

Chapter 8

QUADART INTERRUPTS

This chapter summarizes and cross references interrupt related topics discussed in this manual. It also describes how to configure the Quadart in different interrupt structures at the C-Bus board level and at the S-100 bus task level. Rereading Section 1.8, with special emphasis on Figure 14, would be an excellent introduction to this chapter.

8.1 INTRA- AND INTER-DEVICE INTERRUPT SUMMARY

Quadart interrupts are generated and prioritized at four hierarchical levels: the intra-device, the inter-device, the board, and the task level. The interrupt priorities at the intra- and inter-device levels are fixed on the Quadart. The Quadart board itself may be prioritized at different C-Bus levels, and the task which the Quadart supports may be prioritized among other tasks at the S-100 bus level (see Sections 8.3 and 8.4).

The fixed interrupt priority scheme among all Quadart interrupt sources at the intra- and inter-device level is shown in Table 11. The interrupt source priorities descend from left to right, then top to bottom in the table. Interrupt priorities within a device are defined by its fixed internal circuitry, and interrupt priorities among the five Quadart LSI devices are defined by the IEI (Interrupt Enable In) and IEO (Interrupt Enable Out) interconnections between them.

Table 11: QUADART INTERRUPT SOURCES AND PRIORITIES

SIO (IC31):	Ch 0 RXSPCL, RCA, EXSTAT, TBE Ch 1 RXSPCL, RCA, EXSTAT, TBE
SIO (IC33):	Ch 2 RXSPCL, RCA, EXSTAT, TBE Ch 3 RXSPCL, RCA, EXSTAT, TBE
PIO (IC32):	Ch 0 & 1 DSR/RI Ch 2 & 3 DSR/RI
CTC (IC34):	Ch 0 Clock, Ch 1 Clock, Ch 2 Clock, Timer A
CTC (IC35):	Ch 3 Clock, Timer B, Timer C, Timer D

Each of the 26 table entries may be programmed to supply a unique interrupt vector during interrupt acknowledge from the IOP. One base vector is defined for each SIO channel pair; either the base vector itself, or a base vector with bits V3, V2 and V1 modified to point to one of eight interrupt sources may be selected as an interrupt response. The PIO provides for two independent interrupt vectors, one for Ch 0 & 1, and the other for Ch 2 & 3. One base vector is defined for each set of four CTC counter/timers; bits V2 and V1 are then modified to point to the counter/timer requesting service. In most applications the interrupt vectors would be assigned disjoint values for quick and efficient interrupt servicing.

Reference the following manual sections for detailed descriptions of each Table 11 interrupt source:

RXSPCL	Parity Error, Rx Overrun Error, CRC/Framing Error, SDLC End Of Frame bit group -- enabled/disabled automatically with RCA interrupts. See Sections 4.2, 4.3, 5.3, 6.3 and 7.3.
RCA	Receive Character Available. See Sections 4.2, 4.3, 5.3, 6.3 and 7.3.
EXSTAT	DCD, $\overline{\text{Sync}}$/Hunt, CTS, Tx Underrun/EOM, Break/Abort bit group. See Sections 4.2, 4.5 and 9.1.
TBE	Transmitter Buffer Empty. See Sections 4.2, 5.2, 6.2 and 7.2.
PIO DSR/RI	Data Set Ready and Ring Indicator PIO input circuits. PIO output lines ExtCk and CY may also be programmed to generate interrupts, but this would never be done in practice. See Sections 2.2 and 9.2.
CTC Clocks	The CTC clocks are used as local Quadart $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ sources primarily in the asynchronous mode, and would not be used to generate interrupts. See Section 2.3.
CTC Timers	The CTC timers are used primarily to generate time out intervals required by different serial protocols and DCE devices. See Sections 11.1 and 11.2.

The Quadart IEO and IEI lines are interconnected as shown in Figure 43 (the actual connection incorporates look ahead logic to shorten the propagation time between devices, but it is equivalent to that shown). The IEO and IEI lines work like this:

1. When an interrupt condition becomes active in any device in the chain, the device forces its $\overline{\text{INT}}$ output line active low and its IEO line low provided: (a) its IEI pin is inactive high, meaning no higher priority devices are currently under service or requesting service, and (b) Z80 status M1 is false; if M1 is true, $\overline{\text{INT}}$ and IEO are forced low after M1 goes false. If either of these two conditions is not satisfied, the device must hold off or remove its $\overline{\text{INT}}$ request. If IEI on any device goes active low, then it must force its IEO line low to pass the inhibit signal down the interrupt chain. Note that there may be only one device in the chain which is holding $\overline{\text{INT}}$ low while waiting for service (the highest active priority device in the chain).

2. An interrupting device then waits for an interrupt acknowledge (INTA) from the Z80 directed to it. A Z80 INTA is signaled by C-Bus status lines $\overline{\text{M1}}$ and $\overline{\text{IORQ}}$ being low simultaneously. A device knows an INTA is directed to it when it is waiting for service **and** its IEI line is high when INTA occurs (no higher priority device is requesting service). The device then uses this event to gate its interrupt vector onto the C-Bus data lines, and to remove its interrupt request (float its $\overline{\text{INT}}$ output line). The interrupt vector placed on the bus points to the highest priority intra-device source which is currently requesting service.

3. The interrupt vector read from the C-Bus data lines is used by the IOP Z80 along with the contents of its I-Register to form an indirect jump address (which must be an even number). The two bytes starting at the indirect jump address are then loaded into the Z80 PC register, and service routine execution begins with Z80 interrupts disabled (they are disabled automatically by an INTA). In order to reenable Z80 interrupts, the service routine must execute an EI instruction before returning to the background program. As soon as the EI instruction is executed, higher priority devices may interrupt any lower priority routine in progress thus suspending service to a lower priority device and nesting the service routines; this may happen repeatedly with several levels of nesting. Any service routine may inhibit all higher priority interrupts by not executing an

EI instruction until just before returning to the background program.

4. A device under service must release its IEO line to lower priority devices when all of its services routine are completed. This is done by internal circuitry in each device which detects RETI (Return From Interrupt) instruction opcode (EDh followed by 4Dh) on the C-Bus data lines while its IEI line is high. If there are higher priority devices awaiting service which have not received an INTA directed to them, then these devices momentarily force their IEO lines inactive high allowing a lower priority device to release its IEO line when an RETI occurs. If two or more service routines to the same device are nested, then an RETI instruction must be decoded for each one before the device IEO line goes high.

Figure 43 shows how the IEI and IEO lines behave when two service routines become nested. In (1), no interrupt requests are active. In (2), the PIO issues an interrupt request to the IOP by forcing its $\overline{\text{INT}}$ and IEO lines low and lower priority devices pass the inhibit signal down the chain. In (3), an SIO issues an interrupt request to the IOP by forcing its $\overline{\text{INT}}$ and IEO lines low. Again the low IEO is passed down the chain. Assuming the PIO interrupt service routine has executed an EI instruction, the IOP suspends PIO servicing and begins servicing the SIO (if an EI instruction is not executed, the same line states exist, but PIO servicing continues and the SIO waits for service). In (4), the SIO service routine executes an RETI instruction which releases line IEO, and the Z80 resumes service to the PIO. In (5), the PIO service routine executes an RETI instruction which releases PIO line IEO.

8.2 THREE IMPORTANT INTERRUPT NOTES

Three potential interrupt problems are easily avoided by following the procedures outlined below:

1. All Quadart service routines **must** terminate with an RETI instruction, **not** an RET instruction. If an RET instruction is used instead, then IEO lines which go active low on the Quadart never go back inactive high.

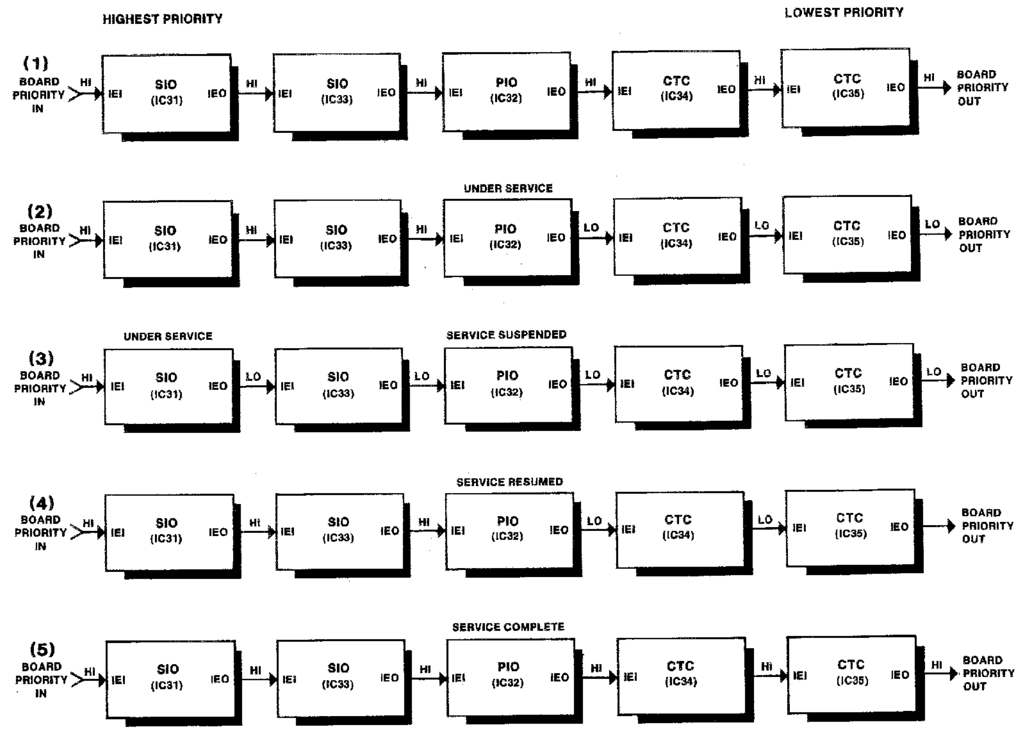


Figure 43: IEI AND IEO LINE BEHAVIOR

2. Caution must be exercised during writes to SIO registers WR1 through WR7, and reads from RR0 through RR2 with interrupts enabled. For example, assume a write to channel 1 register WR5 is in progress with interrupts enabled; a WR0 write has taken place with bits D2, D1 and D0 pointing to WR5, but before the command byte to WR5 is written, an interrupt occurs and servicing begins. In the service routine, a write to channel 1 WR1 is attempted; the first **pointer byte** supposedly written to WR0 is actually interpreted by the SIO as the second half of the incomplete WR5 write from the background program. The **WR1 command byte** in the service routine is then erroneously used as a pointer byte for a new and random SIO register access. When the service routine terminates and the background program **completes** its write to WR5, it writes the command byte intended for WR5 to the random register targetted in the service routine. Note that if the service routine in the example had written to another channel (0, 2 or 3), or if the background program had disabled interrupts before the WR5 write and enabled interrupts afterwards, then both the background and the service routine writes would have been performed correctly. The general approach to avoiding this potential problem is then:

   A. If all interrupt sources within an SIO channel are disabled (e.g., channel 0 TBE, RXSPCL, RCA and EXSTAT interrupts disabled -- a polled channel), then no action is required.

   B. If any interrupt source within an SIO channel is enabled, then IOP interrupts should be disabled while writing to registers WR1 through WR7, or reading from RR0 through RR2 within the channel. Commands written to WR0 may be written with IOP interrupts enabled since the command is completed in a single write operation. IOP interrupts are automatically disabled following an INTA from the IOP (every interrupt service routine starts out with IOP interrupts disabled), or IOP interrupts may be disabled by executing a DI instruction before commencing SIO register I/O. After the register I/O is complete, IOP interrupts should be reenabled by executing an EI instruction.

For example:

```
DI                         ;Disable IOP interrupts.
LD      A,5                ;Point to WR5 for next write.
OUT     (SIOCOM),A         ;
LD      A,CMD              ;Write command byte to WR5.
OUT     (SIOCOM),A         ;
EI                         ;Reenable IOP interrupts.
```

3. A command which disables PIO interrupts should not be written while IOP interrupts are enabled. The reason for this is that a PIO interrupt condition may occur (RI or DSR changes state) as the PIO interrupt disable control word arrives. The PIO then pulls C-Bus line $\overline{\text{INT}}$ active low before the PIO interrupt disable command has taken affect. When it does take affect shortly after, the PIO does not respond to the INTA from the IOP (it does not supply an interrupt vector). This potential problem is avoided by disabling IOP interrupts with a DI instruction before writing the disable PIO interrupts control word (03h) to the PIO, followed by an EI instruction.

For example:

```
DI                         ;Disable IOP interrupts.
LD      A,03h              ;Write disable PIO.
OUT     (PIOCOM),A         ;interrupts command.
EI                         ;Reenable IOP interrupts.
```

8.3 C-BUS BOARD LEVEL INTERRUPTS

The entire Quadart board may be prioritized with other boards connected to the same physical C-Bus cable by using the PRIORITY IN and PRIORITY OUT lines found on each Cromemco C-Bus board (see Figure 14). These lines are active low and behave at the board level exactly as the IEI and IEO lines function at the inter-device level (see Section 8.1). Three integral C-Bus lines (PRI 1, PRI 2 and PRI 3) are dedicated to interconnecting the PRIORITY IN/OUT lines of up to three C-Bus boards without requiring external cabling. These three boards are defined to have priorities CBUS 2 (lowest priority), CBUS 3 and CBUS 4 (highest priority). Additional boards defined as External Device 1, External Device 2, and so on, may be connected to a C-Bus cable with CBUS 2, CBUS 3 and CBUS 4. Daisy chain wiring external to the C-Bus must be provided to prioritize them. The resulting interrupt priority arrangement is shown in Figure 44. Note that the External Device at the end of the daisy chain has the highest board priority, and CBUS 2 has the lowest board priority.

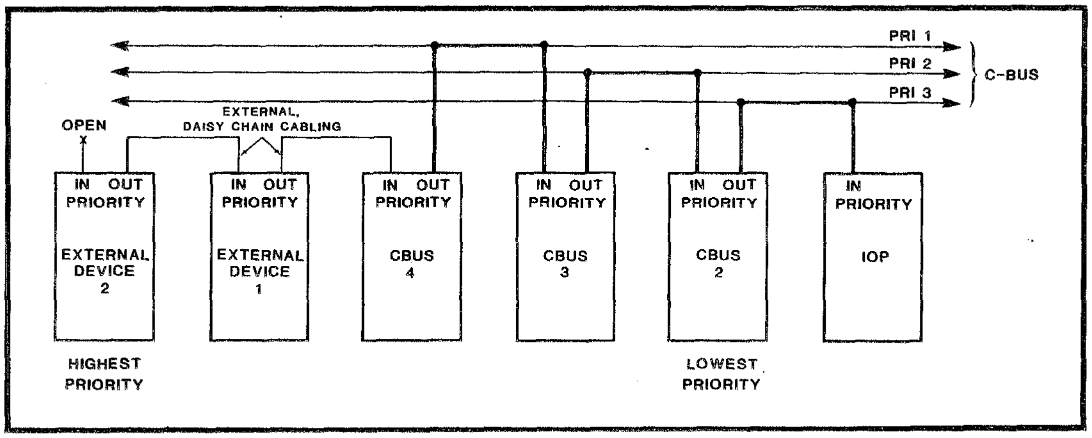


Figure 44: C-BUS INTERRUPT WIRING

A Quadart board is assigned priority CBUS 2, CBUS 3 or CBUS 4 by installing a 16-pin DIP header plug in Quadart socket IC28. The plug must be wired in one of the three configurations shown in Figure 45. The plug then interconnects PRI 1, PRI 2 and PRI 3 as shown above in Figure 44. When the Quadart is used as an External Device, external cabling must be attached to Quadart connector J1 as shown in Figure 44, and in this case, socket IC28 should be left empty.

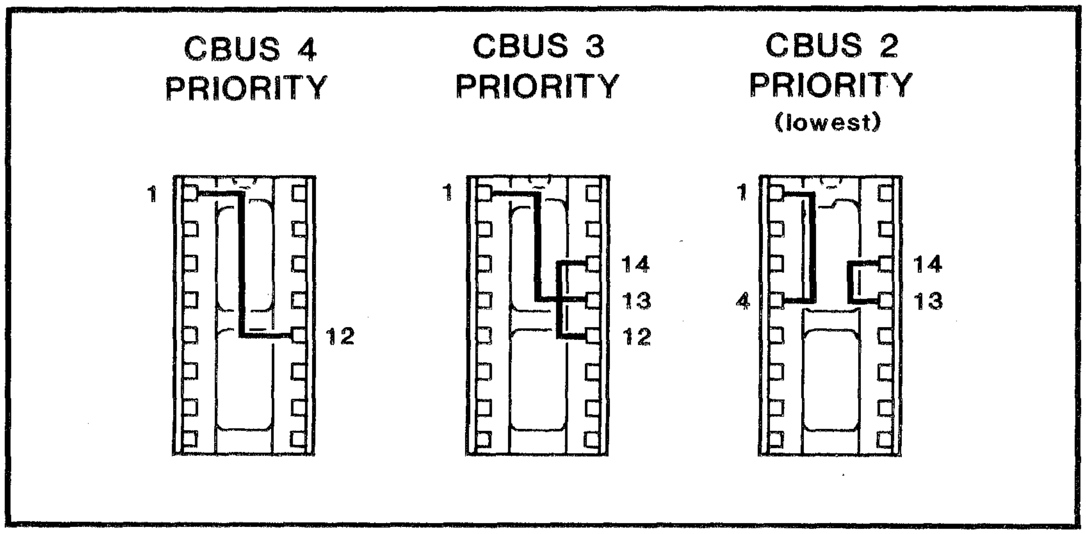


Figure 45: IC28 PLUG WIRING

8.4 S-100 BUS TASK LEVEL INTERRUPTS

An IOP together with a C-Bus cable and the boards attached to it form a S-100 bus task, and each such task appears as four parallel I/O ports to the host processor. Tasks may interrupt the host processor using the S-100 bus $\overline{\text{INT}}$ line, and place interrupt vectors on the S-100 data lines in response to an INTA from the host processor. An IOP and the boards it controls may be task prioritized among other tasks as well as a variety of Cromemco S-100 products such as the PRI Printer Interface, the TU-ART Serial Interface, the 4FDC and 16FDC Floppy Disk Controllers, and the WDI Hard Disk Interface. This is done by using the PRIORITY IN and PRIORITY OUT lines provided by each board; these lines are active low, but they behave somewhat differently than the IEI/IEO and board level PRIORITY IN/OUT lines.

The S-100 PRIORITY IN/PRIORITY OUT lines prioritize only **concurrently pending** interrupt requests to the host processor. That is, if two or more task interrupt requests are simultaneously pending, then the S-100 PRIORITY IN/OUT circuitry causes the host processor to service the highest priority request. Lower priority tasks continue to assert S-100 line $\overline{\text{INT}}$ active low while this service routine begins, however. If the service routine enables interrupts with an EI instruction, then the highest priority of the currently pending interrupts is then serviced, regardless of its priority relative to the interrupted service routine. Consequently, task interrupt priorities must be managed by the S-100 PRIORITY IN/OUT lines in concert with host executive software.

The IOP PRIORITY IN/OUT lines are brought out for interfacing at IOP connector J2. A representative interrupt structure using the S-100 PRIORITY IN/OUT daisy chain is shown in Figure 46.

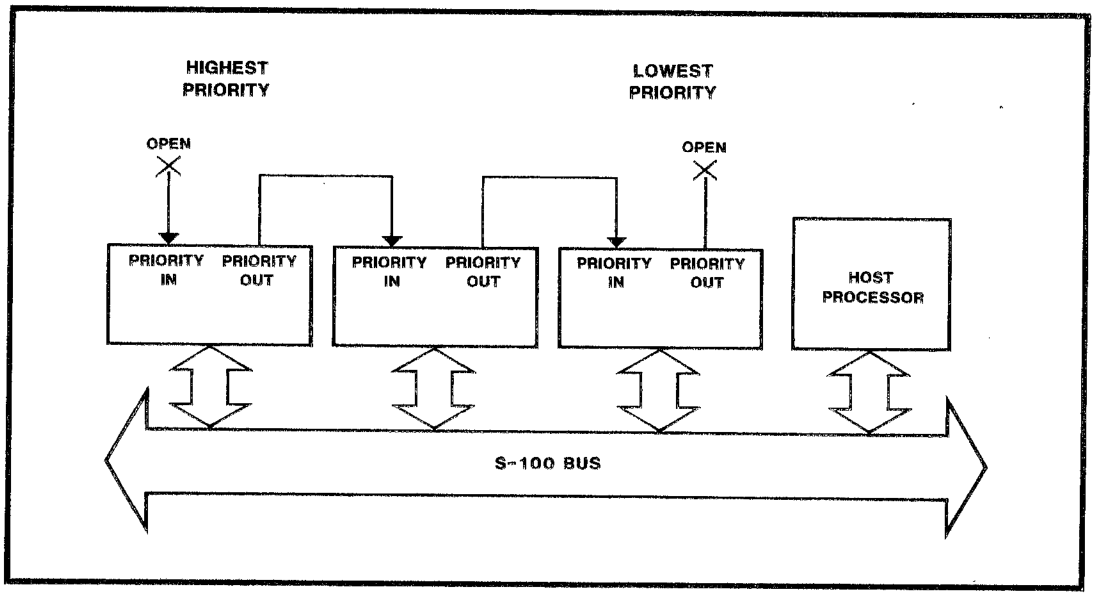


Figure 46: S-100 BUS DAISY CHAIN STRUCTURE

Chapter 9

MODEMS

The Quadart supplies a complete set of RS-232C compatible handshake lines. The RS-232C interface for one of four identical channels is shown in Figure 9. Both DTE and DCE connectors are provided for each Quadart channel. Since the DTE and DCE connectors for each channel are parallel driven, only one of the two should be attached to external equipment at any time. Modems should always use the DTE connector.

This chapter focuses on seven interface circuits and how they are controlled: RTS (Request To Send); CTS (Clear To Send); DSR (Data Set Ready); DCD (Data Carrier Detect); CY (User Defined Output To Modem); DTR (Data Terminal Ready); and RI (Ring Indicator). Each circuit is monitored by an LED indicator which is lit when the circuit is on (spacing), and extinguished when the circuit is off (marking). The Quadart LED physical layout is illustrated in Figure 47.

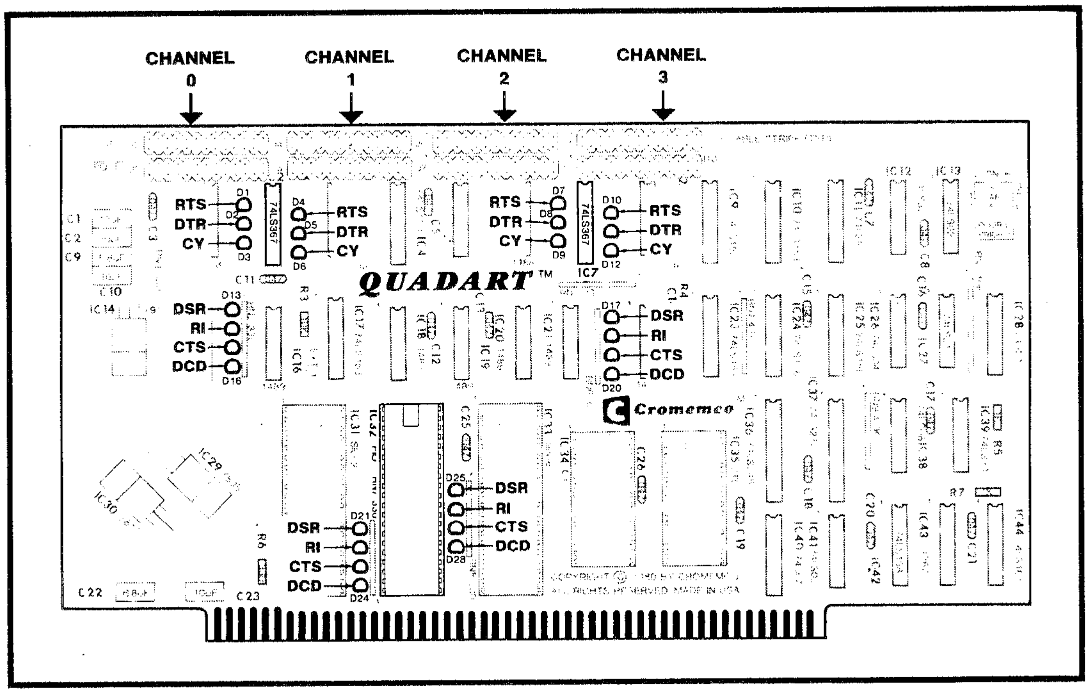


Figure 47: QUADART MODEM STATUS INDICATORS

Of the seven circuits, four are inputs to the Quadart (CTS, DSR, DCD and RI), and three are outputs from the Quadart (RTS, CY and DTR). Five of the seven circuits may be programmed to generate IOP interrupts (all but RTS and DTR), all seven may be employed for user defined I/O over the RS-232C interface, and selected circuits may be programmed to carry out predefined SIO modem control functions.

The following manual sections describe the other RS-232C interface circuits:

Circuit	**Manual Sections**
ExtCk	1.4, 1.5, 2.2, 3.1, 3.2, 3.3
RxC, TxC	1.4, 1.5, 3.1, 3.2, 3.3
RxD, TxD	1.2, 1.4, 1.5, 1.9, 3.1, chapter 10

9.1 CIRCUIT DESCRIPTIONS

This section gives a detailed description of the seven RS-232C control circuits. The input circuit group (CTS, DSR, DCD and RI) is described first.

CTS Clear To Send. Read from SIO register RR0 bit D5; logic 0 = off (marking), logic 1 = on (spacing). The modem turns this circuit on to signify it is ready to accept data for transmission. Circuit CTS may be used for user defined I/O, or it may be programmed to function as an SIO channel transmitter enable line by setting control bits Auto Enables and Tx Enable (then CTS on enables the transmitter, CTS off disables it). In the asynchronous mode, a character may be loaded into the Tx buffer while the transmitter is disabled by a CTS off circuit; status bit TBE is then reset. When circuit CTS subsequently turns on, the preloaded character is properly sent and status bit TBE is set. In all modes, when the channel transmitter is disabled by CTS turning off, any data character in the process of being shifted out is completely sent before the transmitter $\overline{\text{TxD}}$ line goes marking. In the synchronous modes, if CRC is being sent when the transmitter is disabled by circuit CTS, a full 16 bits is sent, but sync (or flag) bits replace CRC bits as soon as the transmitter is disabled. A change (in either direction) in status bit CTS latches all five EXSTAT bits, and it also generates an EXSTAT interrupt if control bit EXSTAT Interrupts Enable is set. A Reset EXSTAT Interrupts command must then be issued to WR0 to remove the interrupt and/or to unlatch the five EXSTAT bits.

DSR Data Set Ready. Read from PIO -- DSR0 from Data A bit D7, DSR1 from Data A bit D3, DSR2 from Data B bit D7 and DSR3 from Data B bit D3 (see Figure 16); logic 1 = off (marking), logic 0 = on (spacing). The modem turns this circuit on to signify that it is connected to the line and ready to exchange further control and data signals with the Quadart; an off DSR circuit indicates the modem is not ready to operate (e.g., when it is not powered, or when it is in the test mode). Circuit DSR has no predefined function on the Quadart, so its use is completely defined by the IOP software. Any AND/OR combination of PIO controlled circuits (including DSR) may be programmed to generate interrupts to the IOP. See Section 9.2 for PIO interrupt programming details.

DCD Data Carrier Detect. Read from SIO register RR0 bit D3; logic 0 = off (marking), logic 1 = on (spacing). The modem turns this circuit on to signify that the received signal meets the modem's suitability requirements (e.g., power and frequency), and it turns DCD off to signify that the received signal is unsuitable for demodulation. Circuit DCD may be used for user defined I/O, or it may be programmed to function as an SIO channel receiver enable line by setting control bits Rx Enable and Auto Enable (then DCD on enables the channel receiver, and DCD off disables it). If the channel receiver is disabled by circuit DCD while a character is in the process of being assembled, then this character is lost (and status bit RCA is not set), but any characters already in the Rx FIFO may be properly read. A change (either direction) in status bit DCD latches all five EXSTAT bits, and it also generates an EXSTAT interrupt if control bit EXSTAT Interrupts Enable is set. A Reset EXSTAT Interrupts command must then be issued to WR0 to remove the interrupt and/or to unlatch the five EXSTAT bits.

RI Ring Indicator. Read from PIO -- RI0 from Data A bit D6, RI1 from Data A bit D2, RI2 from Data B bit D6 and RI3 from Data B bit D2 (see Figure 16); logic 1 = off (marking), logic 0 = on (spacing). The modem turns this circuit on to signify that a calling signal is being received; an off RI circuit indicates that no calling signal is being received in switched network use. Like DSR, circuit RI has no predefined function on the Quadart so its use is completely defined by the IOP software. Any AND/OR combination of PIO controlled circuits (including DSR) may be programmed to generate interrupts to the IOP. See Section 9.2 for PIO interrupt programming details.

The following three RS-232C circuits are Quadart outputs and are used for modem control.

RTS Request To Send. Written to SIO register WR5 bit D1; logic 0 = off (marking), logic 1 = on (spacing). This circuit is provided to turn the modem transmitter on and off in half duplex applications. This circuit must be turned on to request the modem to enter the data transmit mode, then the modem should indicate its readiness to transmit data by turning circuit CTS on. RTS must be turned off to force the modem to the non-transmit mode after all data has been transferred to the modem over circuit TxD. After turning RTS off, RTS must not be turned back on again until after the modem has turned CTS off (the RTS and CTS circuits must interlock). In byte synchronous and SDLC modes, modem line RTS directly follows control bit RTS. In the asynchronous mode, setting control bit RTS turns modem line RTS on, but the modem line will not turn off until: (1) control bit RTS is reset, and (2) the transmitter and Tx buffer are empty. This behavior makes it difficult to use modem line RTS as a general purpose output line in the asynchronous mode if the channel transmitter is used. Control bit RTS does not generate interrupts.

CY General Purpose RS-232C Output. Written to PIO -- CY0 to Data A bit D5, CY1 to Data A bit D1, CY2 to Data B bit D5 and CY3 to Data B bit D1 (see Figure 16); logic 1 = off (marking), logic 0 = on (spacing). This line is provided for user defined output to the RS-232C interface, e.g., automatic dialer control or modem speed control. This circuit may be programmed to generate a PIO interrupt although it would almost always be inhibited from doing so.

DTR Data Terminal Ready. Written to SIO register D7; logic 0 = off (marking), logic 1 = on (spacing). The Quadart turns this circuit on to signify that it is ready to transmit and receive data. When the Quadart/modem station is configured for automatic call, answering the Quadart turns DTR on to connect the modem to the line in response to an on RI circuit, and it turns DTR off to terminate the call and remove the modem from the line. Circuit DTR should not be turned off until all transmitter data clears the transmitter itself. Since SIO transmit channels are buffered, then either a timeout interval in the synchronous mode, or a set condition of status bit All Sent in the asynchronous mode should be observed before resetting DTR. Circuit DTR may also be used for user defined output to the RS-232C interface; it does not generate interrupts.

9.2 PIO INTERRUPT PROGRAMMING

All four RS-232C input circuits (CTS, DSR, DCD and RI) may be programmed to generate interrupts to the IOP. Circuits CTS and DCD are managed by the SIOs as members of the EXSTAT bit group. Their interrupt behavior has been discussed above in Sections 4.2 and 4.5. Circuits DSR and RI are managed by the Quadart PIO. This section describes how to program the PIO to generate interrupts from these two RS-232C circuits.

The PIO interrupt behavior is defined by six bytes written to the PIO -- three to register PIO Control A, and three to register PIO Control B. The bytes written to PIO Control A define the interrupt behavior of Quadart channels 0 and 1, while those written to PIO Control B define interrupt behavior of channels 2 and 3. The command byte formats are shown above in Figure 18. The PIO interrupt response should be initialized by following the flowchart shown in Figure 48.

The flowchart must be followed twice, once for each PIO port. After a PIO port is initialized, port interrupts may be reconfigured by writing a new Interrupt Control Word and optionally a new Interrupt Mask to PIO Control A/B. PIO interrupts may be enabled or disabled at any time without affecting the PIO interrupt control word by writing a special PIO command formatted as shown in the

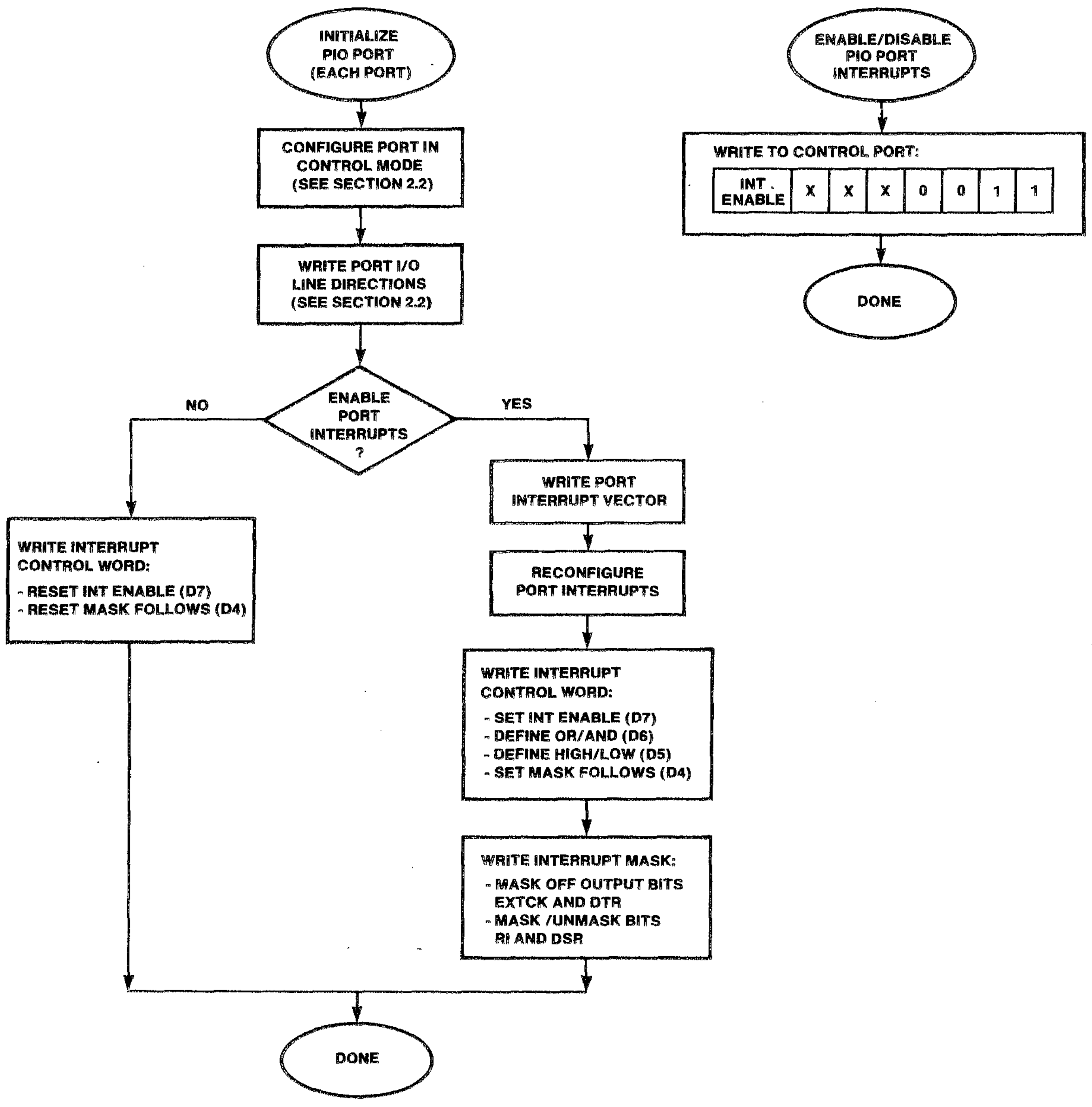

INITIALIZE PIO PORT (EACH PORT)
CONFIGURE PORT IN CONTROL MODE (SEE SECTION 2.2)
WRITE PORT I/O LINE DIRECTIONS (SEE SECTION 2.2)
ENABLE PORT INTERRUPTS ?
NO
YES
WRITE INTERRUPT CONTROL WORD:
- RESET INT ENABLE (D7)
- RESET MASK FOLLOWS (D4)
WRITE PORT INTERRUPT VECTOR
RECONFIGURE PORT INTERRUPTS
WRITE INTERRUPT CONTROL WORD:
- SET INT ENABLE (D7)
- DEFINE OR/AND (D6)
- DEFINE HIGH/LOW (D5)
- SET MASK FOLLOWS (D4)
WRITE INTERRUPT MASK:
- MASK OFF OUTPUT BITS EXTCK AND DTR
- MASK /UNMASK BITS RI AND DSR
DONE
ENABLE/DISABLE PIO PORT INTERRUPTS
WRITE TO CONTROL PORT:
INT. ENABLE
X
X
X
0
0
1
1
DONE

Figure 48: PIO INTERRUPT PROGRAMMING

figure. If an interrupt occurs while PIO interrupts are disabled, it is internally latched and passed on to the IOP when PIO interrupts are later reenabled. IOP interrupts should be disabled while writing a control byte which disables PIO interrupts (see Section 8.2).

Bits And/Or and High/Low together with the Interrupt Mask define a logic expression which generates port interrupts. The Interrupt Mask bits either include or exclude each port I/O line from the interrupt generating expression; a reset mask bit includes an I/O line, and a set mask bit excludes an I/O line from the expression (see Figure 18). Port output lines ExtCk and DTR as well as input lines may be included in the logic expression, although in normal applications only input lines DSR and RI would be included in the interrupt generating expression.

Once the lines to be included in the expression are defined, control bit High/Low defines the logic sense, and control bit And/Or defines the logic operation performed on the selected operands. All operands are combined to feed the PIO interrupt logic as shown in Figure 49.

The PIO interrupt logic operates as follows:

1. A PIO interrupt may be issued to the IOP only if the generating logic expression (node Y in the figure) changes from false to true. If more than one line is unmasked in the expression, then individual line transitions may or may not generate interrupts, depending upon the states of the other lines and the selected logic function.

2. A PIO interrupt is latched, prioritized and asserted when it is the highest active interrupt requesting service from the IOP.

3. A PIO interrupt request is removed (PIO pin $\overline{\text{INT}}$ goes high) by an IOP interrupt acknowledge (lines $\overline{\text{IORQ}}$ and $\overline{\text{M1}}$ active low simultaneously, IEI inactive high) to the PIO. At this time the PIO port places its interrupt vector on the C-Bus. Note that an INTA from the IOP is targetted to one device only, the highest priority device requesting service.

4. Internal PIO circuitry decodes any RETI opcode from the C-Bus data lines and uses this event to update its IEO output line (all IOP interrupt service routines must return to the background program with an RETI instruction). When PIO servicing is complete, the RETI instruction from its service routine causes the PIO to force its IEO line inactive high thereby enabling service to lower priority interrupts in the chain.

5. If PIO interrupt N occurs, then PIO interrupt N+1 is issued only when the generating logic equation Y goes false after INTA to interrupt N and is then true, or becomes true, after the RETI instruction ending interrupt N service is executed.

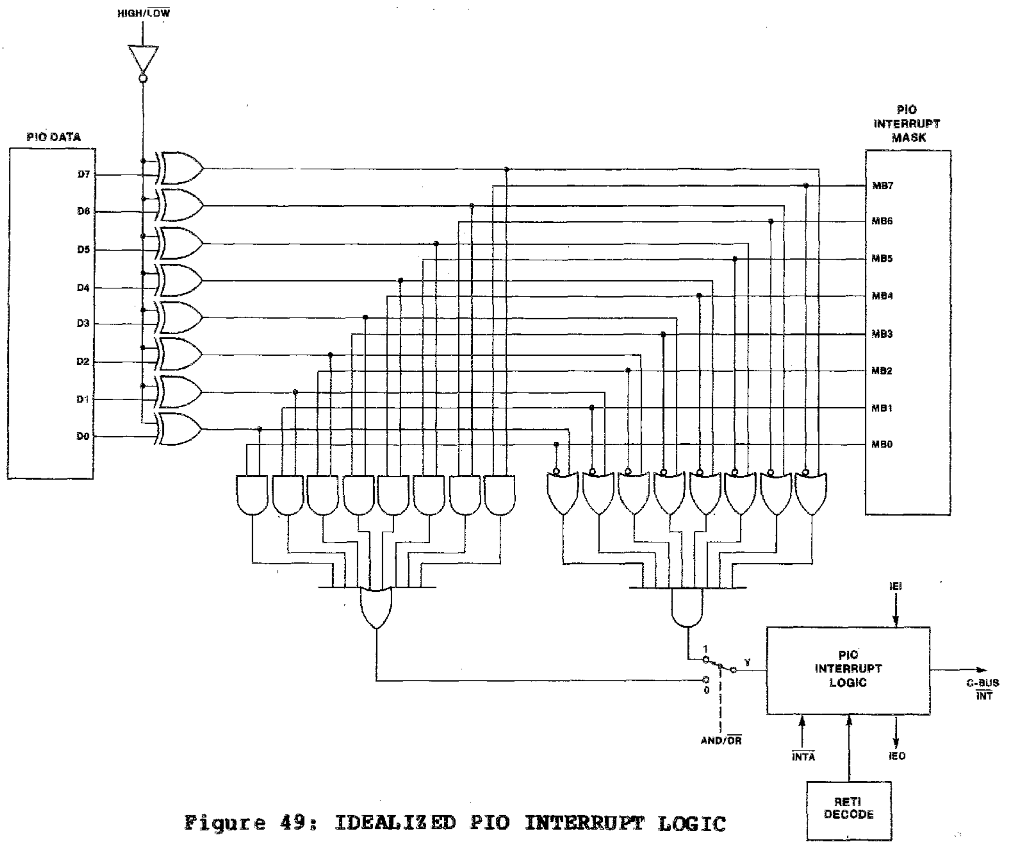


Figure 49: IDEALIZED PIO INTERRUPT LOGIC

Example

(A) Assume that in a certain application two Ring Indicator circuits are to be monitored as shown in Figure 50, one from channel 0 and one from channel 1. If it is desired to generate a PIO interrupt when either of the two RI circuits turn on (logic 0), then the generating function Y = $\overline{\text{RI0}}$ + $\overline{\text{RI1}}$ would be chosen and programmed with the interrupt control word and mask shown. If, for example, RI1 generates a PIO interrupt by turning on, then the service routine might mask off RI1 so that a change in RI0 could be detected (Y = $\overline{\text{RI0}}$ + 0), and later unmask it when the call is completed.

(B) Assume that in another application circuits DSR0 and RI0 are monitored as shown in Figure 50. If it is desired to generate a PIO interrupt only when both DSR0 and RI0 are on (both logic 0), then the generating function Y = $\overline{\text{DSR0}}$ * $\overline{\text{RI0}}$ would be chosen and programmed with the interrupt control word and mask shown. The service routine might then mask off RI0 to detect a loss of circuit DSR0 during the call (Y = $\overline{\text{DSR0}}$ * 1), and later unmask RI0 when the call is completed.

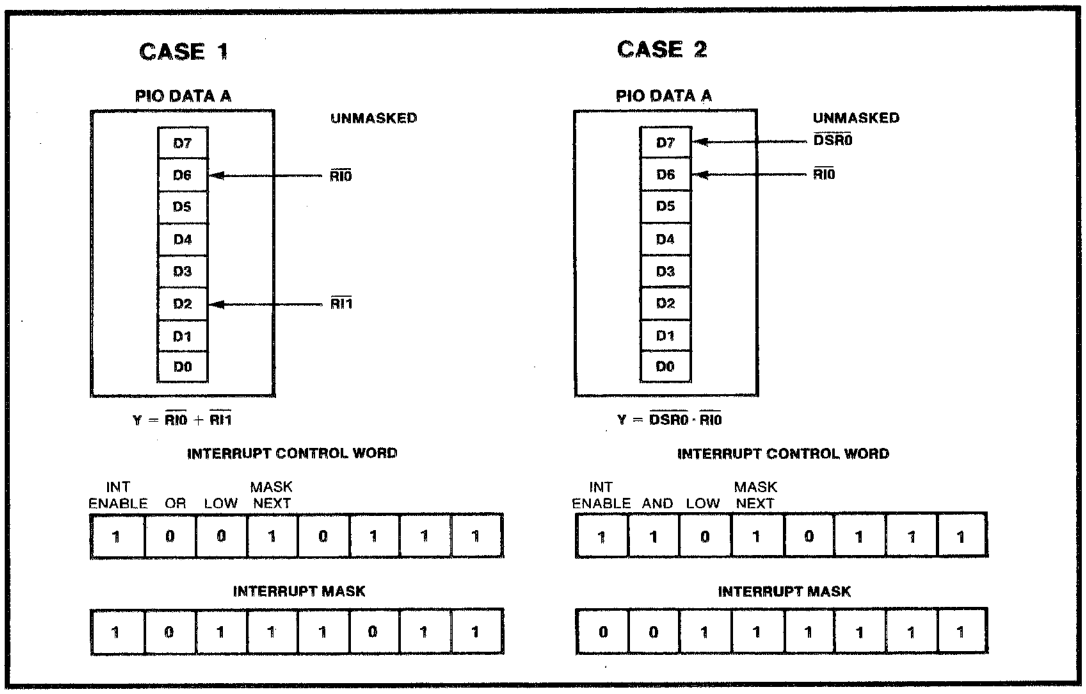


Figure 50: PIO INTERRUPT APPLICATION EXAMPLES

9.3 USER DEFINED MODEM LINE USE

Circuits CTS, DSR, DCD and RI may be used for general purpose RS-232C input, and RTS, CY and DTR may be used for general purpose RS-232C output as mentioned above. This section outlines how these circuits might be used to control an 801-type Automatic Calling Unit (ACU), or dialer. One of many possible connections between the Quadart and the ACU are shown in Figure 51.

In this application, two of four Quadart channels are used to control the ACU leaving the other two channels available for serial I/O. All active ACU control lines can be connected to the Quadart with a 12 conductor cable. ACU signal digit lines NB8, NB4, NB2 and NB1 are driven by SIO bits RTS and DTR. This allows two bits at a time to be defined with a single write to SIO register WR5. ACU bits COS (Call Origination Status), DLO (Data Line Occupied), ACR (Abandon Call -- Retry) and PND (Present Next Digit) are accessed with one read from register PIO Data B. ACU control bits CRQ (Call Request) and DPR (Digit Present) are likewise controlled with one write to register PIO Data B. Finally, ACU status line PWI (Power Indicator) is fed to circuit CTS which is read at SIO register RR0 bit D5. This is done to isolate it easily from the other four ACU status bits for generating a unique interrupt; and to mask it easily at bit position D5 to form a memory status image with the other four status bits which are read from the PIO at bit positions D7, D6, D3 and D2.

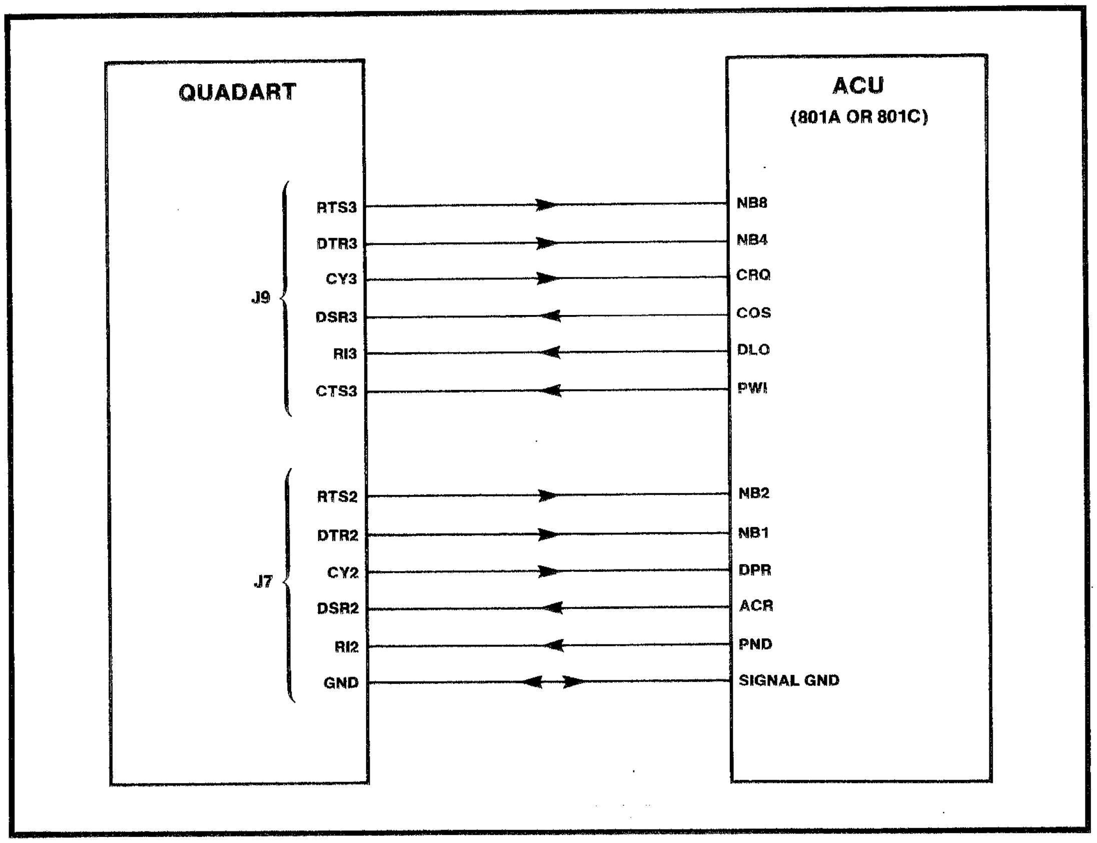


Figure 51: A QUADART/ACU CONNECTION

Chapter 10

LOOPBACK

During normal Quadart operation, four channels of incoming modem RxD serial data drive four SIO $\overline{\text{RxD}}$ input pins, and four SIO $\overline{\text{TxD}}$ output pins drive four channels of outgoing modem TxD serial data. The Quadart loopback system enables the IOP to create other data paths, connecting one of eight possible serial data sources (four modem RxD inputs and four SIO $\overline{\text{TxD}}$ outputs) to one of eight serial data sinks (four modem TxD outputs and four SIO $\overline{\text{RxD}}$ inputs).

The loopback system is initially disabled after a POC or a C-Bus reset which means the Quadart system defaults to the normal data paths described above. A loopback data path is created when the IOP writes a control byte to Quadart port CNTRL with bit Loopback (D7) set; any path created in this way is destroyed by writing another byte with bit Loopback reset. A simplified picture of the loopback system appears above in Figure 15, and a more detailed version appears in Figure 52. The control byte written to port CNTRL is formatted as shown in Figure 53.

Figure 52 shows seven CNTRL bits driving multiplexers and data selectors in the loopback system (CNTRL bit D6 is not used). Port CNTRL itself is a 74LS273 octal latch (IC37) whose outputs are reset by a POC or C-Bus reset, and the resulting loopback system state is that shown in the figure.

When control bit Loopback is set, CNTRL bits Output To Modem (D5), OS1 (D4) and OS0 (D3) select one of eight serial data sinks; bits OS1 and OS0 (Output Select 1 and 0) function as channel select lines, and bit Output To Modem chooses between modem TxD outputs and SIO $\overline{\text{RxD}}$ inputs. Likewise, CNTRL bits Input From Modem (D2), IS1 (D1) and IS0 (D0) select one of eight data sources; again, IS1 and IS0 (Input Select 0 and 1) function as channel select lines, and control bit Input From Modem chooses between modem RxD inputs and SIO $\overline{\text{TxD}}$ outputs. There are 64 possible data source/sink combinations, but 4 of these are not permitted (see below).

Refer to Figure 52 and note that when bit Loopback is set there are four general categories of data sources and sinks.

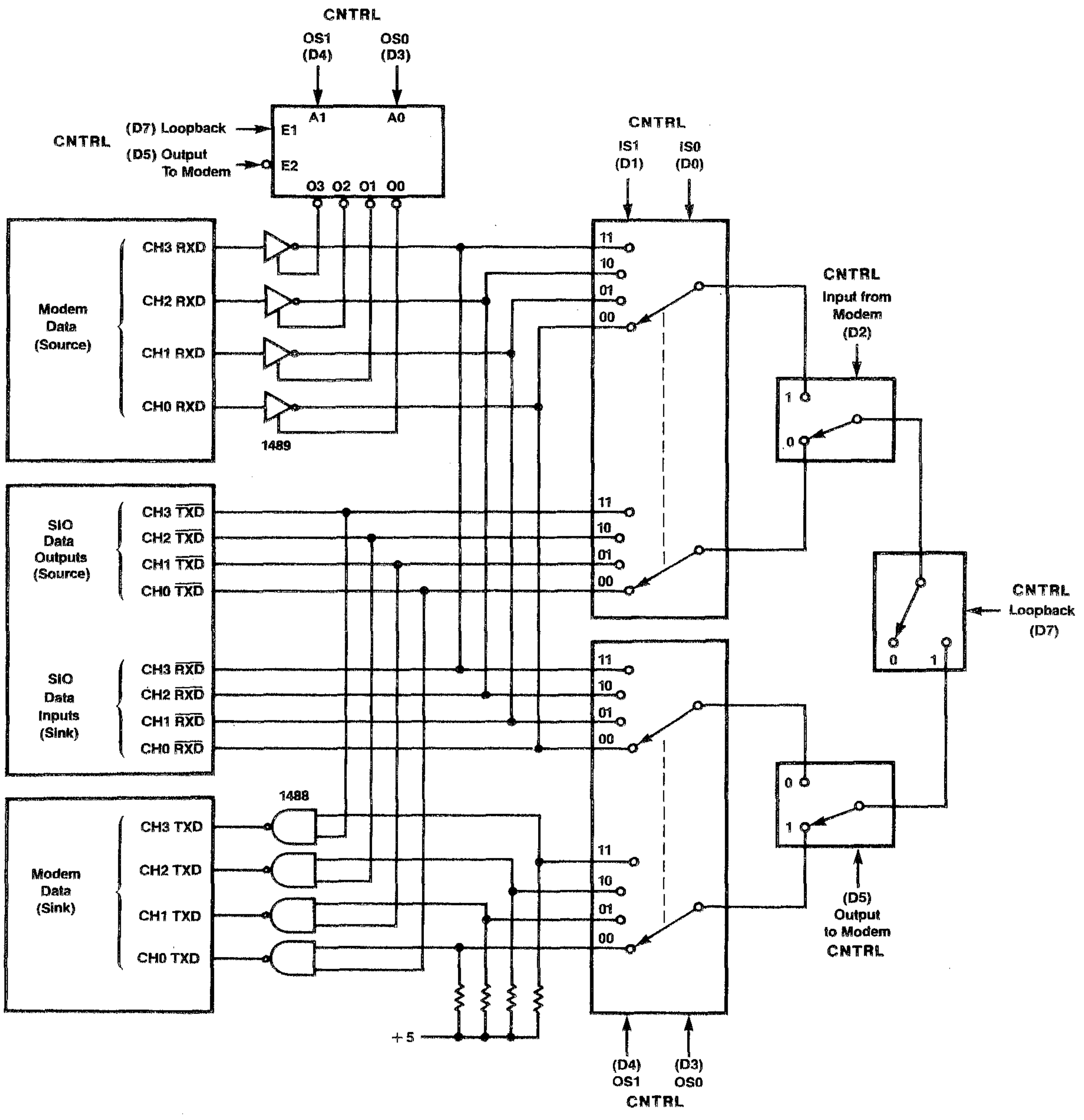


Figure 52: QUADART LOOPBACK SUBSYSTEM

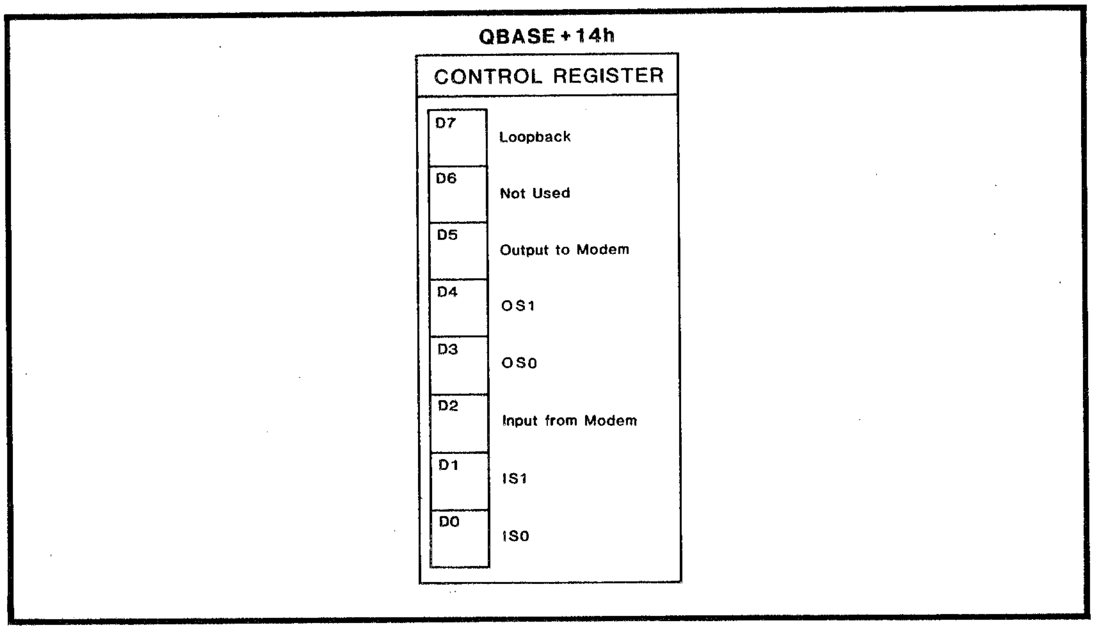


Figure 53: CNTRL PORT BIT ASSIGNMENTS

SIO $\overline{\text{TxD}}$ Source

In this case control bit Input From Modem is reset. The selected SIO $\overline{\text{TxD}}$ output drives **both** the loopback data sink and its normal modem TxD output channel (by driving a 1488 NAND driver input) in parallel.

Modem RxD Source

In this case control bit Input From Modem is set. The selected modem RxD input channel drives **both** the loopback data sink and its normal SIO $\overline{\text{RxD}}$ input in parallel.

SIO $\overline{\text{RxD}}$ Sink

In this case control bit Output To Modem is reset. Here the data selector shown in the upper left of Figure 52 is enabled which in turn disables one 1489 receiver to avoid conflict between the loopback data source and the normal modem RxD input at the selected SIO $\overline{\text{RxD}}$ input pin. Thus the selected loopback data source **replaces** the normal modem RxD data input to the SIO. If and only if one of the four **normal** data paths is selected with the loopback system (e.g., routing channel 2 modem RxD data to channel 2 SIO $\overline{\text{RxD}}$), then the SIO $\overline{\text{RxD}}$ input pin is left floating (the 1489 receiver which is disabled to avoid conflict also disables data from reaching the SIO $\overline{\text{RxD}}$ input), and marking data only is read from the SIO receiver. This is an inconsequential restriction since these four normal data paths exist when the loopback system is disabled.

Modem TxD Sink

In this case control bit Output To Modem is set. The four modem TxD channels are driven by 1488 NAND drivers. In normal operation, one input on each of these drivers is pulled up to +5 VDC and the other input is fed by an SIO $\overline{\text{TxD}}$ output pin. With the loopback system enabled, the loopback data source drives one of the pulled-up 1488 NAND input lines with an open collector data selector output. In this case the other NAND input from the SIO $\overline{\text{TxD}}$ output must be forced to logic 1 (marking), otherwise mixing of the loopback data source and the normal SIO $\overline{\text{TxD}}$ output data to the TxD modem line will occur. This is easily done by disabling the channel SIO transmitter which forces its $\overline{\text{TxD}}$ output line to a marking state.

Table 12 below tabulates in matrix form the control byte sent to port CNTRL by the IOP versus the resulting loopback data path. The table assumes CNTRL bit D6 is reset. When the loopback system is disabled by writing a byte to port CNTRL with control bit Loopback reset, then the states of the other seven bits are irrelevant.

Table 12: CNTRL BYTE FOR EACH LOOPBACK PATH

		SOURCES							
		SIO $\overline{\text{TxD}}$				Modem RxD			
SINKS		Ch0	Ch1	Ch2	Ch3	Ch0	Ch1	Ch2	Ch3
SIO $\overline{\text{RxD}}$	Ch0	80h	81h	82h	83h	---	85h	86h	87h
	Ch1	88h	89h	8Ah	8Bh	8Ch	---	8Eh	8Fh
	Ch2	90h	91h	92h	93h	94h	95h	---	97h
	Ch3	98h	99h	9Ah	9Bh	9Ch	9Dh	9Eh	---
Modem TxD	Ch0	(A0h	A1h	A2h	A3h	A4h	A5h	A6h	A7h)
	Ch1	(A8h	A9h	AAh	ABh	ACh	ADh	AEh	AFh)
	Ch2	(B0h	B1h	B2h	B3h	B4h	B5h	B6h	B7h)
	Ch3	(B8h	B9h	BAh	BBh	BCh	BDh	BEh	BFh)

Note: Dashed table entries indicate the loopback path is not permitted; if the path is created, only marking data will be read.

Table entries in parentheses indicate that if the loopback sink Modem Channel N TxD is selected, then the SIO Channel N $\overline{\text{TxD}}$ must be forced marking.

Chapter 11

TIMERS

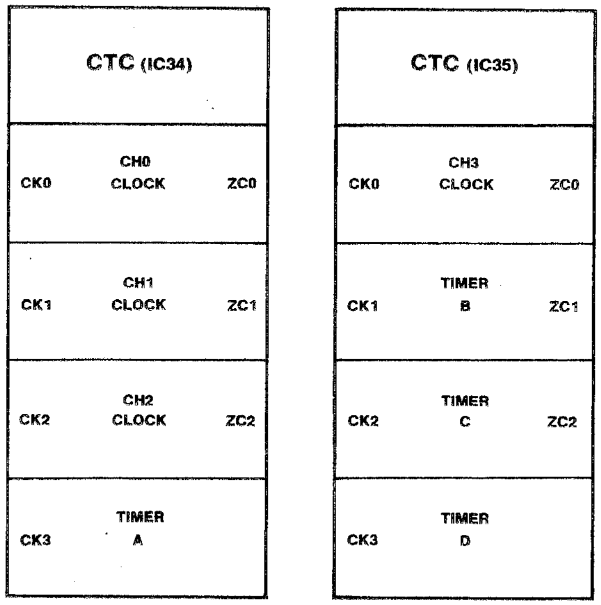
The Quadart provides four general purpose timers. The timers may be used singly for timing relatively short events: for example, sampling or changing a user defined input/output line once every 1.00 mSec in response to a timer interrupt. And timers may be cascaded for longer intervals: for example, cascading two timers to generate an interrupt every 1.00 Sec for automatically inserting sync-idle characters in a BiSync protocol message, or maintaining a 24 hour clock in a data logging application.

The first section describes Quadart timer operation, and the second section presents a timer programming example. Section 2.3 covers CTC initialization. Sections 3.2 and 3.3 discuss Quadart local $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clock programming (the local clocks and timers are merely different channels on the CTCs); and the CTC Technical Manual included in the Quadart documentation package gives further CTC programming details.

11.1 TIMER PROGRAMMING

Two Quadart CTCs contain eight counter/timers; four of these are used for local $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ clock sources, and the other four are used as general purpose timers (timer A through timer D). An idealized CTC device is shown in Figure 19, and Figure 54 shows how the two CTCs are partitioned to perform the local clock and timer functions (also see Figure 21).

Each timer is initialized by writing mode control and time constant bytes to its assigned I/O port (see Figure 54) which starts the timer counting down at the software selected rate. A timer may be interrogated in real time by reading its current count from its assigned port, or it may be programmed to generate an interrupt upon reaching count zero, or both. In either case the time constant (1 to 256 decimal) is automatically reloaded upon reaching count zero, and the count down sequence repeats indefinitely. The timer count down sequence may be stopped at any time by outputting a byte to the timer port with CTC control bit Reset Channel set. Three out of four CTC counter/timers drive a zero count output pin which is pulsed high for 375 nSec (nominal) upon reaching count zero; the fourth timer has no such output pin.



TIMER	PORT ADDRESS
A	Qbase+0Fh
B	Qbase+11h
C	Qbase+12h
D	Qbase+13h

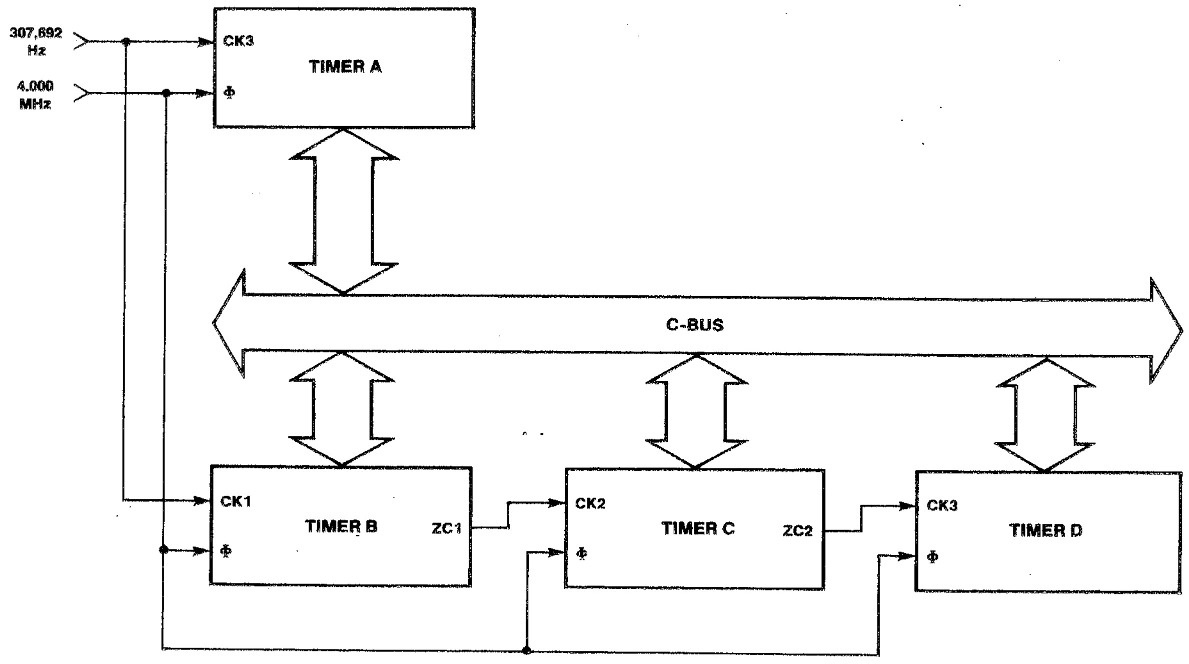


Figure 54: QUADART TIMERS

CTC initialization (interrupts disabled) has been discussed above in Section 2.3. Each timer is initialized to operate in either the timer mode or counter mode. In the **timer mode** a timer is always driven by a crystal controlled 4.000 MHz square wave. The 4.000 MHz clock may be prescaled by either 16 or 256, and the prescaled clock then feeds the timer at either 250 KHz or 15.625 KHz. In the **counter mode,** timer A and timer B are clocked by the same 307,692 Hz non-symmetric waveform (the 4.000 MHz clock externally prescaled by 13), timer C is clocked by the timer B zero count output, and timer D is clocked by the timer C zero count output (see Figure 54). Timers B, C and D are cascaded in this way to generate extended timing intervals. Note that any combination of timer/counter mode operation is allowed, but timer C must be operated in the counter mode to cascade it with timer B, and timer D must also be operated in the counter mode to cascade it with timer C.

Example

(A) Assume timer A is to generate 1.000 mSec timing intervals. Timer A could then be operated in the timer mode with 4.000 MHz, a prescale factor = 16, and a time constant = 250 decimal;

T = (1/4.000 MHz)*16*250 = (4 uSec)*250 = 1.000 mSec

(B) Assume a 1.000 Sec timing interval is needed to drive a software maintained 24 hour clock in a data logging application. Timers C and D could then be cascaded to generate this interval as follows: timer C in the timer mode, prescale factor = 256 and a time constant = 125; timer D in the counter mode, time constant = 125. This causes the timer C zero count output to pulse high once every (1/4.000 MHz)*256*125 = 8.000 mSec. The resulting pulse train clocks timer D which cycles through count zero every 125*8.000 mSec, or 1.000 Sec.

(C) The maximum extended timing interval results from cascading timers B, C and D. Timer B would be operated in the timer mode, prescale factor = 256 and a time constant = 256. Timers C and D would be operated in the counter mode with time constants equal to 256. The zero count output of timer B would have a period of (1/4.000 MHz)*256*256 = 16.384 mSec, the zero count output of timer C would have a period of 256*16.384 mSec = 4.194 Sec, and timer D would cycle through count zero every

256*4.194 Sec, or 1074 Sec (17.90 minutes).

Each timer may individually be enabled/disabled from generating IOP interrupts by setting/resetting control bit Int Enable in the mode control byte written during timer initialization (see Figure 20). Quadart timers would typically be operated in an interrupt driven mode. If timer A interrupts are enabled, then an interrupt vector must be written to port Qbase+0Ch (this vector is shared by the channels 0, 1 and 2 clocks, along with timer A). If timer B, C or D interrupts are enabled, then an interrupt vector must be written to port Qbase+10h (this vector is shared by the channel 3 clock, and timers B, C and D). The formats for these two vectors are shown in Figure 55. The values written are actually **base vectors**; a CTC automatically modifies base vector bits D2 and D1 to point to the highest priority CTC interrupt requesting service before placing the vector on the C-Bus during interrupt acknowledge from the IOP. The CTC interrupt vectors should be disjoint with those supplied by the SIOs, PIO, and the other CTC for fast and efficient interrupt servicing.

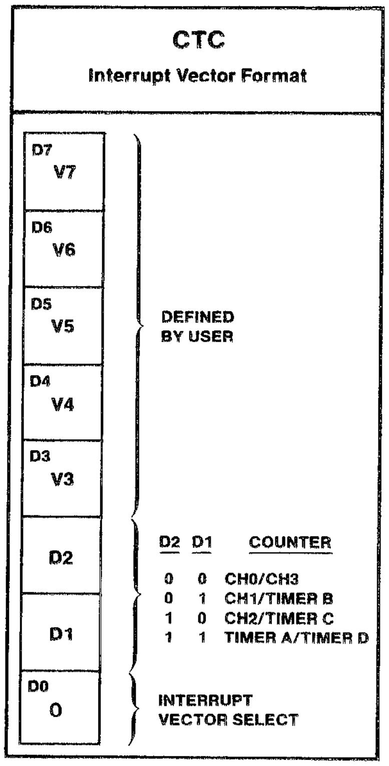


Figure 55: CTC INTERRUPT VECTOR FORMAT

The four Quadart timers are prioritized among the other Quadart interrupt sources as shown in Figure 14 and Table 11; among the timers, timer A has the highest priority and timer D the lowest (it has the lowest priority on the entire Quadart board). When programming the CTCs for interrupt driven operation, all service routines must return to the background program with an RETI (not a RET) instruction. Internal CTC circuitry detects the RETI instruction code on the C-Bus, and uses this event to update its IEO output line.

11.2 A TIMER PROGRAMMING EXAMPLE

The following program maintains a 24 hour interrupt driven clock using Quadart timers C and D. Timer C is configured in the timer mode with interrupts disabled to clock timer D once every 8.000 mSec, and timer D is configured in the counter mode cascading it with timer C. Timer D's time constant is set to 125 which results in a timer D vectored interrupt once every 1.000 Sec. The timer D service routine updates three RAM variables; binary seconds, binary minutes and binary hours (the three values are assumed initialized to the proper time). The three time bytes would be available for reading by any coresident routines in IOP memory.

```
CROMEMCO Z80 Macro Assembler version 03.07


                 ;        TIMER 24 HOUR CLOCK PROGRAMMING EXAMPLE
                 ;
                 ;   See manual text for program description.
                 ;

      (0040)     QBASE    EQU     40h             ;Assumed Quadart base address.
      (0050)     VLOAD    EQU     QBASE+10h       ;Interrupt vector port.
      (0052)     TIMERC   EQU     QBASE+12h       ;Timer C port.
      (0053)     TIMERD   EQU     QBASE+13h       ;Timer D port.

      (4000)              ORG     4000h           ;Start of IOP ram.

                 ;================================================
                 ;  Timer C and timer D initialization.
                 ;================================================

4000  310060     CLOCK:   LD      SP,6000h        ;Locate stack.

4003  3E44                LD      A,44h           ;Define I-Page = 44h.
4005  ED47                LD      I,A             ;
4007  ED5E                IM      2               ;Enable Z80 interrupt mode 2.

4009  3E00                LD      A,0             ;Load 00h CTC interrupt vector.
400B  D350                OUT     (VLOAD),A       ;Resulting IM2 jump table
                                                  ;assuming I-Page = 44h:
                                                  ;4400h -> Ch 3 clock (not used)
                                                  ;4402h -> timer B (not used)
                                                  ;4404h -> timer C (not used)
                                                  ;4406h -> timer D

400D  3E27                LD      A,00100111b     ;Initialize timer C:
400F  D352                OUT     (TIMERC),A      ; - disable timer C interrupts
                                                  ; - timer mode
                                                  ; - prescale factor = 256
                                                  ; - falling clock trigger
                                                  ; - time constant follows
                                                  ; - reset timer C

4011  3E7D                LD      A,125           ;Timer C time constant =
4013  D352                OUT     (TIMERC),A      ;125 for 8 mSec period.

4015  3EC7                LD      A,11000111b     ;Initialize timer D:
4017  D353                OUT     (TIMERD),A      ; - enable timer D interrupts
                                                  ; - counter mode (cascade
                                                  ;   with timer C)
                                                  ; - falling clock trigger
                                                  ; - time constant follows
                                                  ; - reset timer D

4019  3E7D                LD      A,125           ;Timer D time constant =
401B  D353                OUT     (TIMERD),A      ;125 for 1 Sec period.

401D  FB                  EI                      ;Enable IOP interrupts.

                 ;
                 ;        (Background Program)
                 ;

                          ;============================================
                          ; CTC Interrupt Service
                          ;============================================

      (4400)              ORG     4400h           ;I-Page.

                          ;========================
                          ; IM2 Indirect Jump Table
                          ;========================

4400  (0002)              DS      2               ;Ch 3 clock service (not used)
4402  (0002)              DS      2               ;Timer B service (not used)
4404  (0002)              DS      2               ;Timer C service (not used)
4406  0844                DW      SRVTD           ;Timer D service

4408  F5         SRVTD:   PUSH    AF              ;Save registers on stack.
4409  E5                  PUSH    HL              ;

440A  212644              LD      HL,TIME-1       ;Point to ram seconds, minutes and
                                                  ;hours values.

440D  3E3C                LD      A,60            ;Update seconds.
440F  CD2044              CALL    UPDATE          ;
4412  3E3C                LD      A,60            ;Update minutes.
4414  CD2044              CALL    UPDATE          ;
```

```
4417  3E18                LD     A,24            ;Update hours.
4419  CD2044              CALL   UPDATE          ;

441C  E1                  POP    HL              ;Restore registers.
441D  F1                  POP    AF              ;
441E  ED4D                RETI                   ;

                          ;=======================
                          ; Update subroutine
                          ;=======================

4420  23         UPDATE:  INC    HL              ;Move pointer to next time unit.

4421  34                  INC    (HL)            ;Increment time unit.
4422  96                  SUB    (HL)            ;If new value less than its
4423  2001                JR     NZ,THRU         ;roll over value, leave alone;
4425  77                  LD     (HL),A          ;if new value is at roll over
4426  C9         THRU:    RET                    ;value then zero time unit.

                          ;=======================
                          ; Ram time values
                          ;=======================

4427  00         TIME:    DB     00h             ;Seconds.
4428  00                  DB     00h             ;Minutes.
4429  00                  DB     00h             ;Hours.

442A  (4000)              END    CLOCK
```

Appendix A

SIO READ/WRITE REGISTER BIT FUNCTIONS

WRITE REGISTER 0

D7	D6	Function
0	0	Null Command
0	1	Reset Rx CRC Checker
1	0	Reset Tx CRC Generator
1	1	Reset Tx Underrun/EOM Latch

Null Command

No operation.

Reset Rx CRC Checker

This command has no effect in the asynchronous mode; in the byte synchronous mode, it presets the channel CRC checker to all zeroes; in the SDLC mode, it presets the CRC checker to all ones.

Reset Tx CRC Generator

This command has no effect in the asynchronous mode; in the byte synchronous mode, it presets the channel CRC generator to all zeroes; in the SDLC mode, it presets the CRC generator to all ones.

Reset Tx Underrun/EOM Latch

The Tx Underrun/EOM latch status is read from RR0 bit D6. The latch is set following a Quadart POC, and also as the channel transmitter underruns (both the Tx buffer and the transmitter empty); the latch may only be reset by issuing the Reset Tx Underrun/EOM command to WR0. If the latch has been reset by this command any time prior to a Tx underrun **while** control bit Tx CRC Enable is set, then the current CRC character value is appended to the data before the transmission idles sending sync characters (issuing this command to an already empty transmitter also causes the current CRC to be sent). If the latch is set at the time of the underrun **or** if control bit Tx CRC Enable is reset, then only sync characters, and not CRC, follows the data.

D5	D4	D3	Function
0	0	0	Null Command
0	0	1	Send SDLC Abort
0	1	0	Reset EXSTAT Interrupts
0	1	1	Channel Reset
1	0	0	Enable Interrupt on Next Rx Character
1	0	1	Reset TBE Interrupt Pending
1	1	0	RXSPCL Reset
1	1	1	Return From Interrupt (Ch 0 & 2 only)

Null Command

No operation.

Send SDLC Abort

This command only has effect in the SDLC mode. Issuing this command causes eight sequential ones to be immediately transmitted. Since the SDLC protocol allows for five contiguous ones, there may be from eight to thirteen ones transmitted as a result. All data held in the Tx buffer, and any character being shifted out of the transmitter is lost. Also this command immediately sets status bit TBE. If the command is issued while the transmitter is empty, then an abort sequence is sent, but no TBE interrupt can result since the Tx buffer is already empty. If the command is issued while the Tx buffer is full, an abort sequence is sent and a TBE interrupt is generated, if enabled.

Reset EXSTAT Interrupts

A change in any of the five EXSTAT bits (DCD, $\overline{\text{Sync}}$/Hunt, CTC, Tx Underrun/EOM and Break/Abort) causes all five EXSTAT bits to be latched in RR0, and if control bit Enable EXSTAT Interrupts is set, an EXSTAT interrupt to occur. A change in either direction (set or reset) of any bit causes this to occur, with the exception of the Tx Underrun/EOM bit which can only generate an EXSTAT interrupt when it changes from reset to set, not when it is reset by direct command. Note that the other three RR0 bits are not latched, nor are they affected by this command. Issuing a Reset EXSTAT Interrupts command removes any current EXSTAT interrupt (it will not go away otherwise) and frees the five EXSTAT bits to reflect current conditions. If freeing the five EXSTAT bits causes one or more to change state, another EXSTAT interrupt results.

Channel Reset

Issuing this command disables the channel transmitter ($\overline{TxD}$ held marking), modem output lines (RTS and DTR turned off), receiver, and all channel interrupts. All channel registers WR0 through WR7 must then be rewritten.

Enable Interrupt on Next Rx Character

This command only has effect in the RCA Interrupt On First Character mode; if this mode is selected, then the SIO interrupts the IOP only after the first data character is received (the interrupt is removed by reading the data character from the Rx FIFO). This command also rearms the SIO to interrupt the IOP on the next RCA condition.

Reset TBE Interrupt Pending

No affect if control bit TBE Interrupts Enable is reset; if control bit TBE Interrupts Enable is set and a TBE interrupt occurs, then issuing this command removes the TBE interrupt and inhibits further TBE interrupts until either the Tx buffer is loaded with another character and again becomes empty, or until a sync character is automatically loaded into the transmitter after the CRC is sent (synchronous modes only).

RXSPCL Reset

Issuing this command in all modes resets all four RXSPCL status bits in RR1 and removes any current RXSPCL interrupt. This command does not affect any RXSPCL bits stacked in the error FIFO. RXSPCL interrupts from Parity or Rx Overrun Errors must be removed by this command since these errors latch their corresponding bits set in RR1.

Return From Interrupt

Not used; the Quadart implements this function in hardware.

D2	D1	D0	Function
0	0	0	Point to Register 0
0	0	1	Point to Register 1
0	1	0	Point to Register 2
0	1	1	Point to Register 3
1	0	0	Point to Register 4
1	0	1	Point to Register 5
1	1	0	Point to Register 6
1	1	1	Point to Register 7

These three bits target WR0 through WR7 for the following output write, or RR0 through RR2 for the following input read operation.

WRITE REGISTER 1

D7	D6	D5	Function
0	X	X	Disable Wait/Ready

Disable Wait/Ready

The SIO wait/ready features are not implemented on the Quadart, thus bit D7 must be reset. The programmed states of D6 and D5 are then irrelevant.

D4	D3	Function
0	0	Disable RCA and RXSPCL Interrupts
0	1	RCA Int on First Character Only
1	0	RCA Int on All Chars (parity affects vector)
1	1	RCA Int on All Chars (parity does not affect vector)

Disable RCA and RXSPCL Interrupts

In this mode, no RCA interrupts and no RXSCPL interrupts (parity error, Rx overrun, CRC/framing error, SDLC end of frame) are generated.

RCA Interrupt on First Character Only

In this mode, the IOP is interrupted by the next data character in the asynchronous mode, by the first non-sync character in the byte synchronous mode, and by the first valid address byte (Address Search Enabled), or by the first non-flag (Address Search Disabled) in the SDLC mode. The RCA interrupt request is removed by reading the character from the Rx FIFO. Thereafter, the received data stream itself does not interrupt the IOP unless an RXSPCL condition (excluding parity errors) occurs, and in this case, both the RXSPCL condition in RR1 and the character on which it occurred are held (even if read) until an RXSPCL Reset command is issued.

RCA Interrupt on All Characters

In these two modes, the IOP is interrupted by RXSPCL conditions and by **all** received characters. In the asynchronous mode, an RCA interrupt is generated by every character. In byte synchronous modes, an interrupt is generated by every non-sync character when control bit Sync Character Load Inhibit is set, and by every character when control bit Sync Character Load Inhibit is reset (including CRC). In SDLC mode, an interrupt is generated by every non-flag character except data which matches the programmed value of WR6 if control bit Sync Character Load Inhibit is set, or by every non-flag character this control bit is reset (including CRC). If the parity affects vector option is selected, then parity errors generate RXSPCL interrupts provided control bit Parity Enable is set. If the parity does not affect vector option is selected, then parity errors cannot generate RXSPCL interrupts (a valid Parity Error status bit may still be polled from RR1, however).

D2 Status Affects Vector (Channels 1 & 3 only)

This bit can be written to channels 1 and 3 only. The bit written to channel 1 applies to both channel 0 and channel 1, while the bit written to channel 3 applies to both channel 2 and channel 3. If this bit is reset, then the fixed interrupt vector written to WR2 is both read from RR2 and placed on the C-Bus by the SIO during interrupt acknowledge from the IOP, regardless of the SIO channel interrupt source. If this control bit is set, then bits V3, V2 and V1 of the interrupt vector read from RR2 and supplied during interrupt acknowledge will vary with the interrupt source as shown below:

V3	V2	V1	Interrupt Source	
0	0	0	Ch1/Ch3	TBE
0	0	1	Ch1/Ch3	EXSTAT
0	1	0	Ch1/Ch3	RCA
0	1	1	Ch1/Ch3	RXSPCL
1	0	0	Ch0/Ch2	TBE
1	0	1	Ch0/Ch2	EXSTAT
1	1	0	Ch0/Ch2	RCA
1	1	1	Ch0/Ch2	RXSPCL

The other interrupt vector bits V7 through V4 and V0 written to WR2 are unaltered during interrupt acknowledge, or when read from RR2. In an IM2 interrupt driven configuration, this control bit would be set. The byte written to the IOP Z80 I-register would define the interrupt page; the bytes written to channel 1 WR2 and channel 3 WR3 would define the base addresses on this page for the two IM2 indirect jump tables. The variable vectors supplied during interrupt acknowledge would then point to the indirect jump addresses for each service routine.

D1 TBE Interrupts Enable

If this bit is set then TBE conditions generate interrupts to the IOP. In this context a TBE interrupt is only generated after the Tx buffer has emptied into the transmitter. If TBE interrupts are enabled when the Tx buffer and the transmitter are already empty no TBE interrupt results; the Tx buffer must first be loaded and then become empty before the first TBE interrupt occurs. If this control bit is reset then TBE conditions do not generate interrupts.

D0 EXSTAT Interrupts Enable

If this bit is set then a transition in <u>any of</u> the five EXSTAT status bits (Break/Abort, CTS, $\overline{\text{Sync}}$/Hunt, DCD, and Tx Underrun/EOM) generate EXSTAT interrupts to the IOP. A transition in either direction generates an EXSTAT interrupt for all bits except Tx Underrun/EOM; for this bit only a reset to set transition generates an EXSTAT interrupt, not a set to reset change which can occur only by issuing a Reset Tx Underrun/EOM Latch command to WR0. If an EXSTAT interrupt occurs then the interrupt itself and the five EXSTAT bits are latched until a Reset EXSTAT Interrupts command is issued to WR0 which removes the interrupt and frees (unlatches) the EXSTAT bits. If control bit EXSTAT Interrupts Enable is reset then transitions within the EXSTAT bit group do latch the EXSTAT bits in RR0, but they do not generate interrupts to the IOP.

WRITE REGISTER 2 (Channels 1 & 3 Only)

D7	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**	**Function**
V7	V6	V5	V4	V3	V2	V1	V0	Interrupt Vector

Registers WR2 exist in channels 1 and 3 only. An interrupt vector written to channel 1 is shared by both channel 0 and channel 1 while an interrupt vector written to channel 3 is shared by both channel 2 and channel 3. If control bit Status Effects Vector is set then a modified version of this vector, with V3, V2 and V1 varying according to the interrupt source, is placed on the C-Bus during interrupt acknowledge from the IOP. This same vector is available for reading at RR2 in the interrupt service routine. If control bit Status Affects Vector is reset, then the WR2 vector is placed, unmodified, onto the C-Bus during interrupt acknowledge from the IOP and an unmodified version of this vector is also read from RR2.

WRITE REGISTER 3

D7	D6	Function
0	0	Rx 5-Bit Character Length
0	1	Rx 6-Bit Character Length
1	0	Rx 7-Bit Character Length
1	1	Rx 8-Bit Character Length

These two bits determine the number of serial receiver data bits assembled to form an Rx character. The Rx character length may be changed while a character is in the process of being assembled provided the newly programmed Rx character length exceeds the number of bits currently assembled in the receiver. Characters are always assembled right justified in the asynchronous mode with the extraneous high order bits set to logic 1. Characters are also assembled right justified in the byte synchronous and SDLC modes provided the Rx character length is not changed after acquiring sync. If the Rx character length is changed on the fly then special precautions must be taken to right justify the read data -- see Section 4.4. When the Rx character length is changed on the fly in any mode, the read data should be sampled quickly enough so that only one character is in the Rx FIFO at any time, otherwise there is no way to determine on which character the change takes affect.

D5 Auto Enables

If this bit is set, then the channel modem input line DCD functions as a channel receiver enable, and the channel modem input line CTS serves as a channel transmitter enable (circuit off to disable, on to enable in both cases). If this bit is reset, then modem input lines DCD and CTS have no effect on the channel transmitter and receiver. Also see DCD and CTS descriptions in the Read Register 0 group.

D4 Enter Hunt Mode

After Channel Reset command is issued and the channel receiver is then initialized in any synchronous mode, the channel receiver automatically defaults to the hunt mode, and status bit $\overline{\text{Sync}}$/Hunt is set. Once the receiver establishes character synchronization the hunt mode is automatically exited and status bit $\overline{\text{Sync}}$/Hunt is reset. A transition in the $\overline{\text{Sync}}$/Hunt bit (in either direction) causes an EXSTAT interrupt if control bit EXSTAT Interrupts Enable is set. Setting the Enter Hunt Mode bit in all synchronous modes immediately forces the channel receiver to the hunt mode and sets status bit $\overline{\text{Sync}}$/Hunt thereby generating an EXSTAT interrupt, if enabled. The Enter Hunt Mode bit is not latched and the receiver automatically exits the hunt mode once character synchronization is achieved, generating an EXSTAT interrupt if it is enabled. Resetting the Enter Hunt Mode bit is a null operation. This bit has no affect in the asynchronous mode.

D3 Rx CRC Enable

This bit is primarily used in the byte synchronous mode to exclude certain control and sync characters from the CRC calculation; this bit should always be set in the SDLC mode. To properly use this bit in the byte synchronous mode, the received data must be read rapidly enough so that only one character is stored in the Rx FIFO at any time. Assume that character N has just been assembled and transferred to the FIFO, setting status bit RCA. Character N is read from an SIO Data Port and the decision is then made to either include or exclude character N from the CRC calculation, based on the character's context. To include it in the CRC calculation set control bit Rx CRC Enable; to exclude it reset bit Rx CRC Enable. Setting/resetting bit Rx CRC Enable must be accomplished **before** status bit RCA is again set to be valid. This bit has no affect in the asynchronous mode.

D2 SDLC Address Search Mode

This bit may be used to either accept or reject an SDLC frame based on its 8-bit address field. Accepting a message here means that the frame address field, control field, I-Field and CRC bits are assembled by the channel receiver and transferred to the Rx FIFO. Message rejection means that status bit RCA remains reset and no RCA interrupt is generated until a valid address field is received (the eight bits following a flag are interpreted as an address field). If this control bit is set, only SDLC messages with an address field matching the WR6 programmed address, or those messages with global address 11111111b (0ffh) are accepted and all others are rejected. This control bit has no affect in the byte synchronous or asynchronous modes.

D1 Sync Character Load Inhibit

This bit is intended to be used primarily in the byte synchronous mode; it should always be reset in the SDLC mode. Setting this bit in the byte synchronous or SDLC modes inhibits all characters matching that programmed in WR6 if 8-bit sync or SDLC, or those in WR6 and WR7 if 16-bit sync, from being transferred to the Rx FIFO. Any such characters not loaded into the Rx FIFO are still routed to the CRC checker, however, and are included in the CRC calculation if control bit Rx CRC Enable is set. If control bit Sync Character Load Inhibit is reset in the byte synchronous or SDLC modes, then all receiver characters are transferred to the Rx FIFO, setting status bit RCA with each transfer. This bit is typically set only while waiting for a message opening control character like STX. Sync characters in the message body itself should be read to determine whether to include or exclude them from the CRC calculation. This bit has no affect in the asynchronous mode.

D0 Rx Enable

In all modes, setting this bit enables all channel receiver operations to begin functioning (assembling characters, looking for sync characters, generating RCA interrupts, and so on). The channel receiver must not be enabled until the receiver is completely initialized. Resetting this bit immediately disables the receiver; any character in the process of being assembled is lost, and no RCA interrupts are generated. Any received and error data already stacked in the FIFOs may be correctly read when the receiver is disabled, however.

WRITE REGISTER 4

D7	D6	Function
0	0	X1 Clock
0	1	X16 Clock
1	0	X32 Clock
1	1	X64 Clock

These bits define the factor which, when multiplied by the TxD and RxD data rates in bits per second, yield the required TxC and RxC clock rates. This clock multiplier applies to both the channel transmitter and receiver. A X1 clock multiplier **must** be used in all synchronous modes, and a X16 clock multiplier is recommended in the asynchronous mode although X32 and X64 multipliers may also be used. The X1 clock rate may **not** be used in the asynchronous mode as no provision has been made for external bit synchronization on the Quadart.

D5	D4	Function
0	0	8-Bit Sync Character
0	1	16-Bit Sync Character
1	0	SDLC Mode
1	1	External Sync (Not Used)

If synchronous modes are enabled by resetting WR4 bits D3 and D2 (see below), then bits D5 and D4 select among the three synchronous modes; if asynchronous mode is selected these bits have no affect. Selecting the 8-bit sync character option places the channel in the byte synchronous mode. The value written to WR6 defines the sync character sent during transmitter idle, and the value written to WR7 defines the sync character used by the channel receiver to acquire character synchronization. The 16-bit sync character option again selects the byte synchronous mode and the 16-bit value written to WR6 and WR7 is used by both the channel transmitter and receiver. If the SDLC mode is selected, then the SDLC flag 01111110b which must be written to WR7 is used as a sync character by both the channel transmitter and receiver. The SIO $\overline{\text{Sync}}$ pins are uncommitted on the Quadart so the External Sync mode should not be selected.

D3	D2	Function
0	0	Enable Sync Modes
0	1	1 Stop Bit Per Character
1	0	1-1/2 Stop Bits Per Character
1	1	2 Stop Bits Per Character

Resetting both of these bits enables one of the synchronous modes (see above). The other three combinations select the asynchronous mode and the number of stop bits to be used. Either 1, 1-1/2 or 2 spacing stop bits are added to the end of each character by the channel transmitter and the same number of spacing stop bits are expected by the channel receiver. Status bit Framing Error is set if the channel receiver samples marking data where it expects to see a spacing stop bit. The number of programmed stop bits applies to both the transmitter and receiver.

D1 Parity Even/$\overline{\text{Odd}}$

Resetting this bit selects odd parity (the number of logic 1 bits in the character together with the parity bit is odd), and setting this bit selects even parity (the number of logic 1 bits in the character together with the parity bit is even) if control bit Parity Enable is set. This bit has no affect if control bit Parity Enable is reset.

D0 Parity Enable

This bit is active in all modes although it should always be reset in the SDLC mode since the protocol excludes character parity checking. Setting this bit enables channel transmitter parity generation, receiver parity checking, and parity error status reporting at RR1 bit D4. Additionally, parity errors generate RXSPCL interrupts if mode RCA And RXSPCL Interrupts On All Characters (Parity Affects Vector) is enabled when this bit is set. The parity bit is appended at the end of each data character (after the MSB) by the transmitter in addition to the programmed Tx character length, and it is expected by the channel receiver in addition to the programmed number of Rx character bits. The parity bit is read with the received data to the left of the MSB if the programmed Rx character length is less than 8 bits/character. The parity bit does not appear in the read data if the Rx character length is 8 bits/character. If control bit Parity Enable is reset, then no parity bit is appended or expected, no error reporting to RR1 bit D4 occurs, and parity error interrupts cannot occur.

WRITE REGISTER 5

D7 DTR

This bit controls channel modem output line DTR (Data Terminal Ready) at Quadart connectors J2 through J9. Setting this bit turns the DTR circuit on (spacing); resetting this bit forces the DTR circuit off (marking).

D6	D5	Function
0	0	Tx 5-Bit Or Less Character Length
0	1	Tx 6-Bit Character Length
1	0	Tx 7-Bit Character Length
1	1	Tx 8-Bit Character Length

In all modes these two bits define the number of bits which are transferred from an SIO Data Port to the transmitter to form a character. The bits written to an SIO Data Port must be right justified (LSB at position D0, and so on). The extra high order character bits are irrelevant unless the 5-Bit Or Less character length is selected. In this case the high order bits must be formatted as shown below to be properly transmitted:

D7	D6	D5	D4	D3	D2	D1	D0	Character Length
1	1	1	1	0	0	0	D	Tx 1-Bit Length
1	1	1	0	0	0	D	D	Tx 2-Bit Length
1	1	0	0	0	D	D	D	Tx 3-Bit Length
1	0	0	0	D	D	D	D	Tx 4-Bit Length
0	0	0	D	D	D	D	D	Tx 5-Bit Length

Again all bits are right justified (LSB at position D0, and so on). Changing the Tx character length as a character is being shifted out of the transmitter has no affect on that character. The Tx character length is sampled as each character is loaded from the Tx buffer into the transmitter.

D4 Send Break

Setting this bit unconditionally forces the channel transmitter $\overline{TxD}$ output to a spacing state in all modes. Resetting this bit frees the transmitter to resume normal operation. The Send Break bit has no affect on status bit TBE. The Send Break bit is typically used only in the asynchronous mode since an SDLC Abort sequence is sent by issuing Send Abort command to WR0, and break sequences are not used in the byte synchronous mode. In the asynchronous mode setting the Send Break bit forces the transmitter output to a spacing state, but it does not suspend transmitter loading and timing. The Send Break bit must remain set long enough to allow any character in the transmitter, and any character currently held in the Tx buffer to time out and clear the transmitter. This is most easily accomplished by monitoring status bit All Sent which is set only when all character bits (including parity and stop bits) have cleared the transmitter. The All Sent bit may also be used to time a break interval by loading dummy characters into the transmitter buffer with the Send Break bit set and observing the All Sent bit set after each dummy character times out. The Send Break bit is then reset when the desired number of character times have transpired.

D3 Tx Enable

A Quadart POC or channel reset disables the channel transmitter with its $\overline{TxD}$ output marking in all modes. Setting control bit Tx Enable permits the transmitter to begin normal operation. This bit should not be set until the channel transmitter has been completely initialized. Resetting this control bit disables the channel transmitter and again forces its $\overline{TxD}$ output marking. Data or sync characters in the process of being shifted out of the transmitter are completely sent before the $\overline{TxD}$ output begins marking. If the transmitter is disabled while CRC is being sent, sync or flag bits replace CRC bits until a full 16 bits are transmitted, then the $\overline{TxD}$ output begins marking.

D2 CRC-16/SDLC(CRC-CCITT)

Setting this control bit selects a CRC-16 generating polynomial, X16 + X15 + X2 + 1, and resetting it selects a CRC-CCITT generating polynomial, X16 + X12 + X5 + 1, for both the channel transmitter and receiver. The CRC-CCITT polynomial must be used in the SDLC mode where both the CRC generator and checker are preloaded with ones when reset and a special check sequence is used. Either polynomial may be used in the byte synchronous mode, but both the CRC generator and checker are preloaded with zeroes by the Reset Rx CRC Checker/Tx CRC Generator commands, regardless of the polynomial selected.

D1 RTS

This bit controls the channel modem output line RTS (Request To Send) at connectors J2 through J9. Setting this bit in a synchronous mode turns the RTS circuit on (spacing), and resetting the bit turns the circuit off (marking). Setting the RTS control bit in the asynchronous mode again turns circuit RTS on, but unlike the synchronous modes the circuit does not turn off until control bit RTS is reset **and** the channel Tx buffer and transmitter are completely empty. The RTS control bit is then typically controlled as follows in the asynchronous mode: (1) set control bit RTS, (2) load a character in the Tx buffer, (3) reset control bit RTS before the first character clears the transmitter, and (4) continue sending characters -- RTS automatically turns off when character loading stops and all data clears the transmitter.

D0 Tx CRC Enable

If this bit is set at the time a character is transferred from the Tx buffer to the transmitter, then CRC is calculated on the character; if this bit is reset at the time the character is loaded from the Tx buffer to the transmitter, then CRC is not calculated on the character in the synchronous modes. To insure that each character is properly included or excluded from the CRC calculation the following procedure is recommended on **each** Tx character: (1) wait for a TBE condition which signifies character N-1 is now in the transmitter, or the transmitter is empty if the transmission is just beginning, (2) set/reset the Tx CRC Enable bit to include/exclude character N from the CRC calculation, and (3) load character N into the Tx buffer through a channel SIO Data Port. The Tx CRC Enable bit has no effect in the asynchronous mode.

WRITE REGISTER 6

In the 8-bit synchronous mode this register is loaded with the 8-bit **transmit** sync character (the sync characters for the channel transmitter and receiver may differ); in the 16-bit synchronous mode this register is loaded with the 8 low order bits of the 16-bit sync character used by both the channel transmitter and receiver (the low order 8 bits are transmitted and received before the high order 8 bits). In the SDLC mode this register is loaded with the 8-bit address which is compared with the frame Address Field if control bit SDLC Address Search Mode is set. The contents of this register are irrelevant if control bits SDLC Address Search Mode and Sync Character Load Inhibit are reset. This register is used by the receiver only in the SDLC mode since the frame Address Field byte is loaded and sent as ordinary data by the transmitter at the beginning of a frame. This register has no function in the asynchronous mode.

WRITE REGISTER 7

In the 8-bit synchronous mode this register is loaded with the 8-bit **receive** sync character; in the 16-bit synchronous mode this register is loaded with the 8 high order bits of the 16-bit sync character used by both the channel transmitter and receiver. This register must be loaded with the 01111110b (7Eh) flag character in SDLC mode. This register has no function in the asynchronous mode.

READ REGISTER 0

D7 Break/Abort

This EXSTAT status bit is active in the asynchronous and SDLC modes only. In the asynchronous mode the bit is set when a break sequence (a null character plus a framing error) is detected in the RxD data, and is reset when the RxD line stops spacing (a marking bit is sampled). In the SDLC mode status bit Break/Abort is set when seven or more successive ones (marking) are detected in the received data, and is reset when the marking RxD data condition terminates. In both the asynchronous and SDLC modes a transition of this bit (either direction) latches all EXSTAT bits and additionally generates an EXSTAT interrupt if control bit EXSTAT Interrupts Enable is set. The Reset EXSTAT Interrupts command is used to remove the interrupt and/or to free the latched EXSTAT bits to reflect current conditions. The standard procedure for handling break/abort sequences with EXSTAT interrupts enabled should then be: (1) a break/abort sequence generates an EXSTAT interrupt, (2) the service routine issues a Reset EXSTAT Interrupts command to free the Break/Abort bit from its latched state, (3) the termination of break/abort causes the bit to be reset generating another EXSTAT interrupt, (4) the service routine issues another Reset EXSTAT Interrupts command to free the five EXSTAT bits, and (5) the extraneous null character assembled from the break data bits (asynchronous mode only) is read and discarded.

D6 Tx Underrun/EOM

This EXSTAT status bit is reset by a Quadart POC, a channel reset, or by a Reset Tx Underrun/EOM command written to WR0, and is set by a channel transmitter underrun (when both the Tx buffer and transmitter become empty). Once the transmitter underruns this bit must be reset by direct command to detect the next transmitter underrun. A reset to set change in the Tx Underrun/EOM status bit latches all EXSTAT bits in RR0 and generates an EXSTAT interrupt if control bit EXSTAT Interrupts Enable is set, but a set to reset change caused by a Reset Tx Underrun/EOM Latch command does not latch EXSTAT bits nor generate interrupts. The Reset EXSTAT Interrupts command is used to remove an EXSTAT interrupt and/or to free the EXSTAT bits to reflect current conditions. If in the synchronous modes control bit Tx CRC is set **and** the Tx Underrun/EOM latch is reset any time prior to a transmitter underrun, then the current

16-bit CRC character is automatically appended to the last data character sent followed by sync (or flag) characters. Sync (or flag) characters only follow the data if either condition is not met.

D5 CTS

This EXSTAT bit monitors the state of modem input line CTS (Clear To Send) at Quadart connectors J2 through J9. Circuit CTS may serve in its normal capacity as a transmitter enable if control bit Auto Enables is set, or it may be user defined. Reading a logic 0 implies circuit CTS is off (marking); reading a logic 1 implies CTS is on (spacing). A CTS transition (either direction) latches all EXSTAT bits and additionally generates an EXSTAT interrupt if control bit EXSTAT Interrupts Enable is set. The Reset EXSTAT Interrupts command is used to remove the interrupt and/or to free the latched EXSTAT bits to reflect current conditions.

D4 $\overline{\text{Sync}}$/Hunt

This EXSTAT status bit is active in the synchronous modes only. A Channel Reset command forces the receiver into the hunt mode setting status bit $\overline{\text{Sync}}$/Hunt. Receiving one sync character forces the receiver out of the hunt mode resetting status bit $\overline{\text{Sync}}$/Hunt. Setting control bit Enter Hunt Mode also forces the channel receiver into the hunt mode which sets status bit $\overline{\text{Sync}}$/Hunt, and again the status bit is reset as character synchronization is reacquired. In the byte synchronous mode control bit Enter Hunt Mode is typically set at the end of each message, or when it is determined from the received data content that synchronization has been lost. In the SDLC mode control bit Enter Hunt Mode is typically set after it has been determined from the frame address that the frame is not needed. A $\overline{\text{Sync}}$/Hunt transition (either direction) latches all EXSTAT bits and additionally generates an EXSTAT interrupt if control bit EXSTAT Interrupts Enable is set. The Reset EXSTAT Interrupts command is used to remove the interrupt and/or to free the latched EXSTAT bits to reflect current conditions.

D3 DCD

This EXSTAT bit monitors the state of modem input line DCD (Data Carrier Detect) at Quadart connectors J2 through J9. Circuit DCD may serve in its normal capacity as a receiver enable if control bit Auto Enables is set, or it may be user defined. Reading a logic 0 implies circuit DCD is off (marking); reading a logic 1 implies DCD is on (spacing). A DCD transition (either direction) latches all EXSTAT bits and additionally generates an EXSTAT interrupt if control bit EXSTAT Interrupts Enable is set. The Reset EXSTAT Interrupts command is used to remove the interrupt and/or to free the latched EXSTAT bits to reflect current conditions.

D2 TBE

Status bit TBE (Tx Buffer Empty) is set whenever the Tx buffer is empty and CRC is not being sent. It is reset when the Tx buffer is full or when CRC is being sent. Status bit TBE is initially set after a channel reset, and thereafter it is set as the Tx buffer empties into the transmitter itself. The TBE condition which exists immediately after initialization with TBE interrupts enabled does not cause a TBE interrupt. The Tx buffer must first be loaded and then become empty for a TBE interrupt to occur. From this point on TBE interrupts are generated every time the Tx buffer empties. The first and all following TBE interrupts are removed by either loading the Tx buffer through a Quadart Data Port or by issuing a Reset TBE Interrupts Pending command to WR0. After this command is issued, status bit TBE remains set and another TBE interrupt does not occur until the Tx buffer is loaded and then becomes empty. Status bit TBE may be polled in real time if TBE interrupts are disabled. Status bit TBE is set while sync characters or flags are being transmitted in the synchronous modes. While CRC is being shifted out bit TBE is reset and bit Tx Underrun/EOM is set. After CRC is completely sent, status bit TBE is set signaling a sync character or flag has been loaded into the transmitter.

D1 Interrupt Pending (Channels 0 and 2 Only)

This bit may be read from channels 0 and 2 only. Reading this bit from channels 1 and 3 always returns logic 0. This bit is set in channel 0 when there is any channel 0 or channel 1 interrupt pending. Likewise, this bit is set in channel 2 when there is any channel 2 or channel 3 interrupt pending. This bit is reset when all interrupts have been removed by reading any available data characters, loading all empty Tx buffers, and issuing any necessary RXSPCL Reset, Reset TBE Interrupt Pending, or Reset EXSTAT Interrupts commands.

D0 RCA

In all modes, this bit is set when at least one character is available for reading on top of the Rx FIFO through the channel SIO Data Port. This bit is reset when all characters have been read from the Rx FIFO. Inputting data from a SIO Data Port with no new character available merely rereads the previous data character. In all modes each received character is loaded into the Rx FIFO and status bit RCA is set except for: (1) sync characters in the byte synchronous when control bit Sync Character Load Inhibit is set, (2) characters matching the contents of WR6 in SDLC mode when control bit Sync Character Load Inhibit is set, and (3) all flags in SDLC mode.

READ REGISTER 1

D7 SDLC End Of Frame

This RXSPCL status bit is active in SDLC mode only. A reset bit is loaded into the error FIFO each time an assembled character is loaded into the Rx FIFO except at the end of an SDLC frame. The end of an SDLC frame is noted by the detection of a closing flag, and at this moment, trailing CRC bits are loaded into the Rx FIFO as a set SDLC End Of Frame bit is loaded into the error FIFO. When SDLC End Of Frame is set in RR1 then status bit CRC Error and the three SDLC Residue bits in RR1 are also valid (and only then). Enabling any type of RCA interrupts also automatically enables RXSPCL interrupts, and in this case status bit SDLC End Of Frame generates an RXSPCL interrupt when it is set in RR1. The RXSPCL interrupt must be removed by issuing a RXSPCL Reset command to WR0. Since this command both removes the interrupt and resets all four RXSPCL bits to logic 0, all RXSPCL bits should be read **before** issuing this command to determine the interrupt source. Status bit SDLC End Of Frame may be polled if RCA and RXSPCL interrupts are disabled (command RXSPCL Reset may still be used to reset all four RXSPCL bits).

D6 CRC/Framing Error

This RXSPCL status bit is active in all modes. In the asynchronous mode it reports framing errors and it can generate RXSPCL interrupts; in the synchronous modes it reports CRC errors and it cannot generate interrupts of any kind. Like the other three RXSPCL status bits, CRC/Framing Error status is loaded into the error FIFO each time an assembled character is loaded into the Rx FIFO. The status bit is set to indicate a CRC/Framing error, and reset to indicate no error. In the asynchronous mode a set CRC/Framing Error bit implies a spacing stop bit was detected on the character loaded into the Rx FIFO. In the SDLC mode the CRC/Framing Error bit is valid only when the accompanying RXSPCL status bit SDLC End Of Frame is set. In the byte synchronous mode CRC/Framing Error status must be sampled when the 16-bit CRC character is wholly in the CRC checker (see Section 6.3). Enabling any type of RCA interrupt automatically enables RXSPCL interrupts also. An RXSPCL interrupt is generated by a set CRC/Framing Error bit in the asynchronous mode only. This interrupt must be removed by issuing an RXSPCL Reset command to WR0. Since the command both removes the interrupt and resets all four RXSPCL bits to logic 0, all RXSPCL bits should be read **before** issuing this command to determine the interrupt source. Status bit CRC/Framing error may be polled if RCA and RXSPCL interrupts are disabled (command RXSPCL Reset may still be used to reset all four RXSPCL bits).

D5 Rx Overrun Error

This RXSPCL status bit is active in all modes. The bit is loaded into the error FIFO each time an assembled character is loaded into the Rx FIFO. A reset Rx Overrun Error bit is loaded each time if data is read quickly enough from the Rx FIFO so that no characters are overrun. If there are three unread characters in the data FIFO as a fourth character is written to the FIFO, then the fourth character overwrites the third character at the bottom of the Rx FIFO and a set Rx Overrun Error bit along with the fourth character's other three status bits are loaded into the error FIFO (i.e., the third character and its RXSPCL status bits are overwritten). If a character is now read from the Rx FIFO top, all characters and status move up one position in the Rx and error FIFOs. As the fifth character arrives, it and its RXSPCL status bits are stored at the bottom of the FIFOs where they may (or may not) be overrun in the same way. RR1 status bit Rx Overrun Error is latched set as a set Rx Overrun Error bit appears at the top of the error FIFO. This generates an RXSPCL interrupt if RCA interrupts (any mode) are enabled. The interrupt must be removed by issuing a RXSPCL Reset command to WR0. Since the command both removes the interrupt and resets all four RXSPCL bits to logic 0, all RXSCPL bits should be read **before** issuing this command to determine the interrupt source. The Rx Overrun Error bit may be polled if RCA and RXSPCL interrupts are disabled, and the RXSPCL Reset command may be used to reset the latched Rx Overrun Error bit to detect the next overrun.

D4 Parity Error

This RXSPCL status bit is typically used in the asynchronous and byte synchronous modes only although it is active in the SDLC mode also. A Parity Error bit is loaded into the error FIFO each time an assembled character is loaded into the Rx FIFO. A set parity bit is loaded only if control bit Parity Enable is set and the parity of the character loaded into the Rx FIFO does not match that programmed with control bit Parity Even/$\overline{\text{Odd}}$; a reset Parity Error bit is loaded otherwise. RR1 status bit Parity Error is latched set when a set

Parity Error bit appears at the top of the error FIFO. Only if RCA And RXSPCL Interrupts On All Characters (Parity Affects Vector) mode is enabled does the set Parity Error bit result in an RXSPCL interrupt. This RXSPCL interrupt is removed by issuing an RXSPCL Reset command to WR0. Since the command both removes the interrupt and resets all four RXSPCL bits to logic 0, all RXSPCL bits should be read **before** this command is issued to determine the interrupt source. If RCA And RXSPCL Interrupts On All Characters (Parity Affects Vector) is not selected then status bit Parity Error may be polled, and the RXSPCL Reset command may be used to reset the latched Parity Error bit to detect the next parity error.

D3 D2 D1 SDLC Residue Code Bits

These bits are valid only in the SDLC mode when RXSPCL status bit SDLC End Of Frame is set in RR1. The code is used to separate valid data bits from CRC bits near the end of an SDLC frame. These bits do not generate interrupts of any type. Refer to Section 7.4 for a full description of SDLC residue codes.

D0 All Sent

This bit is active in the asynchronous mode only. It is useful for timing break intervals and for modem line control. The All Sent bit is set when all bits have cleared the channel transmitter and the Tx buffer is empty, and it is reset when a character is loaded into the channel Tx buffer. The All Sent bit does not generate an interrupt of any kind.

READ REGISTER 2 (CHANNELS 1 & 3 ONLY)

This register exists in channels 1 and 3 only, and is provided on the SIO to accommodate processors which do not have vectored interrupt capabilities. If RCA interrupts are enabled and if control bit Status Affects Vector is set, then reading RR2 returns the interrupt vector written to WR2 with bits V3, V2 and V1 modified to point to the highest priority interrupt source currently requesting service. If no interrupt is currently pending, then V3=0, V2=1 and V1=1 is returned. If control bit Status Affects Vector is reset, then the vector is read exactly as written to WR2.

Appendix B

SIX BIT TRANSCODE CHARACTER SET

00h	SOH	20h	-
01h	A	21h	/
02h	B	22h	S
03h	C	23h	T
04h	D	24h	U
05h	E	25h	V
06h	F	26h	W
07h	G	27h	X
08h	H	28h	Y
09h	I	29h	Z
0Ah	STX	2Ah	ESC
0Bh	.	2Bh	,
0Ch	<	2Ch	%
0Dh	BEL	2Dh	ENQ
0Eh	SUB	2Eh	ETX
0Fh	ETB	2Fh	HT
10h	&	30h	0
11h	J	31h	1
12h	K	32h	2
13h	L	33h	3
14h	M	34h	4
15h	N	35h	5
16h	O	36h	6
17h	P	37h	7
18h	Q	38h	8
19h	R	39h	9
1Ah	Space	3Ah	SYN
1Bh	$	3Bh	#
1Ch	*	3Ch	@
1Dh	US	3Dh	NAK
1Eh	EOT	3Eh	EM
1Fh	DLE	3Fh	DEL

Appendix C

USASCII CHARACTER SET

00h	NUL	(^@)	20h	SPACE	40h	@	60h	'
01h	SOH	(^A)	21h	!	41h	A	61h	a
02h	STX	(^B)	22h	"	42h	B	62h	b
03h	ETX	(^C)	23h	~	43h	C	63h	c
04h	EOT	(^D)	24h	$	44h	D	64h	d
05h	ENG	(^E)	25h	%	45h	E	65h	e
06h	ACK	(^F)	26h	&	46h	F	66h	f
07h	BEL	(^G)	27h	'	47h	G	67h	g
08h	BS	(^H)	28h	(	48h	H	68h	h
09h	HT	(^I)	29h	)	49h	I	69h	i
0Ah	LF	(^J)	2Ah	*	4Ah	J	6Ah	j
0Bh	VT	(^K)	2Bh	+	4Bh	K	6Bh	k
0Ch	FF	(^L)	2Ch	,	4Ch	L	6Ch	l
0Dh	CR	(^M)	2Dh	-	4Dh	M	6Dh	m
0Eh	SO	(^N)	2Eh	.	4Eh	N	6Eh	n
0Fh	SI	(^O)	2Fh	/	4Fh	O	6Fh	o
10h	DLE	(^P)	30h	0	50h	P	70h	p
11h	DC1	(^Q)	31h	1	51h	Q	71h	q
12h	DC2	(^R)	32h	2	52h	R	72h	r
13h	DC3	(^S)	33h	3	53h	S	73h	s
14h	DC4	(^T)	34h	4	54h	T	74h	t
15h	NAK	(^U)	35h	5	55h	U	75h	u
16h	SYN	(^V)	36h	6	56h	V	76h	v
17h	ETB	(^W)	37h	7	57h	W	77h	w
18h	CAN	(^X)	38h	8	58h	X	78h	x
19h	EM	(^Y)	39h	9	59h	Y	79h	y
1Ah	SUB	(^Z)	3Ah	:	5Ah	Z	7Ah	z
1Bh	ESC	(^[)	3Bh	;	5Bh	[	7Bh	{
1Ch	FS	(^\)	3Ch	<	5Ch	\	7Ch	|
1Dh	GS	(^])	3Dh	=	5Dh	]	7Dh	}
IEh	RS	(^^)	3Eh	>	5Eh	^	7Eh	~
1Fh	VS	(^)	3Fh	?	5Fh	_	7Fh	DEL

Note: The ^ symbol denotes the **CNTRL** key.

Appendix D

EBCDIC CHARACTER SET

00h	NUL	2Ah	SM	7Dh	'	C7h	G
01h	SOH	2Dh	ENQ	7Eh	=	C8h	H
02h	STX	2Eh	ACK	7Fh	"	C9h	I
03h	ETX	2Fh	BEL	81h	a	D0h	}
04h	PF	32h	SYN	82h	b	D1h	J
05h	HT	34h	PN	83h	c	D2h	K
06h	LC	35h	RS	84h	d	D3h	L
07h	DEL	36h	UC	85h	e	D4h	M
09h	RLF	37h	EOT	86h	f	D5h	N
0Ah	SMM	3Ch	DC4	87h	g	D6h	O
0Bh	VT	3Dh	NAK	88h	h	D7h	P
0Ch	FF	3Fh	SUB	89h	i	D8h	Q
0Dh	CR	40h	SPACE	91h	j	D9h	R
0Eh	SO	4Ah	¢	92h	k	E0h	\
0Fh	SI	4Bh	.	93h	l	E2h	S
10h	DLE	4Ch	<	94h	m	E3h	T
11h	DC1	4Dh	(	95h	n	E4h	U
12h	DC2	4Eh	+	96h	o	E5h	V
13h	DC3	4Fh	|	97h	p	E6h	W
14h	RES	50h	&	98h	q	E7h	X
15h	NL	5Ah	!	99h	r	E8h	Y
16h	BS	5Bh	$	A1h	~	E9h	Z
17h	IL	5Ch	*	A2h	s	F0h	0
18h	CAN	5Dh	)	A3h	t	F1h	1
19h	EM	5Eh	;	A4h	u	F2h	2
1Ah	CC	5Fh	^	A5h	v	F3h	3
1Ch	IFS	60h	-	A6h	w	F4h	4
1Dh	IGS	61h	/	A7h	x	F5h	5
1Eh	IRS	6Ah	|	A8h	y	F6h	6
1Fh	IUS	6Bh	,	A9h	z	F7h	7
20h	DS	6Ch	*	C0h	{	F8h	8
21h	SOS	6Eh	>	C1h	A	F9h	9
22h	FS	6Fh	?	C2h	B		
24h	BYP	79h	`	C3h	C		
25h	LF	7Ah	:	C4h	D		
26h	EOB/ETB	7Bh	#	C5h	E		
27h	PRE/ESC	7Ch	@	C6h	F		

QUADART PARTS LIST

	Part No.	Description	Qty
Resistors			
R1-4	001-0030	10 Kohm, 1/4 W	4
R5	001-0018	1 Kohm, 1/4 W	1
R6	001-0012	330 Kohm, 1/4 W	1
R7	001-0018	1 Kohm, 1/4 W	1
Resistor Networks			
RN1-3	003-0013	330 ohm, 7 resistor, 8 pin	3
RN4	003-0009	4.7 Kohm, 7 resistor, 8 pin	1
RN5	033-0024	10 Kohm, 9 resistor, 10 pin	1
RN6-8	003-0013	330 ohm, 7 resistor, 8 pin	3
RN9	003-0009	4.7 Kohm, 7 resistor, 8 pin	1
Capacitors			
C1-2	004-0032	10 uF @ 20 VDC	2
C3-8	004-0061	.047 uF @ 50 VDC	6
C9	004-0034	6.8 uF tant	1
C10	004-0032	10 uF @ 20 VDC	1
C11-21	004-0061	.047 uF @ 50 VDC	11
C22	004-0034	6.8 uF tant	1
C23	004-0032	10 uF @ 20 VDC	1
C24-26	004-0061	.047 uF @ 50 VDC	3
Diodes			
D1-28	008-0020	LED, green, TIL-211	28
Integrated Circuits			
IC1	010-0076	75188/1488	1
IC2	010-0108	74LS367	1
IC3	010-0076	75188/1488	1
IC4	010-0048	74LS153	1
IC5-6	010-0076	75188/1488	2
IC7	010-0108	74LS367	1
IC8	010-0076	75188/1488	1
IC9	010-0048	74LS153	1
IC10	010-0049	74LS151	1
IC11	010-0053	74LS85	1
IC12	010-0058	74LS32	1
IC13	010-0068	74LS02	1
IC14	012-0014	Regulator, 7912	1
IC15-16	010-0077	75189/1489	2
IC17	010-0048	74LS153	1
IC18-22	010-0077	75189/1489	5

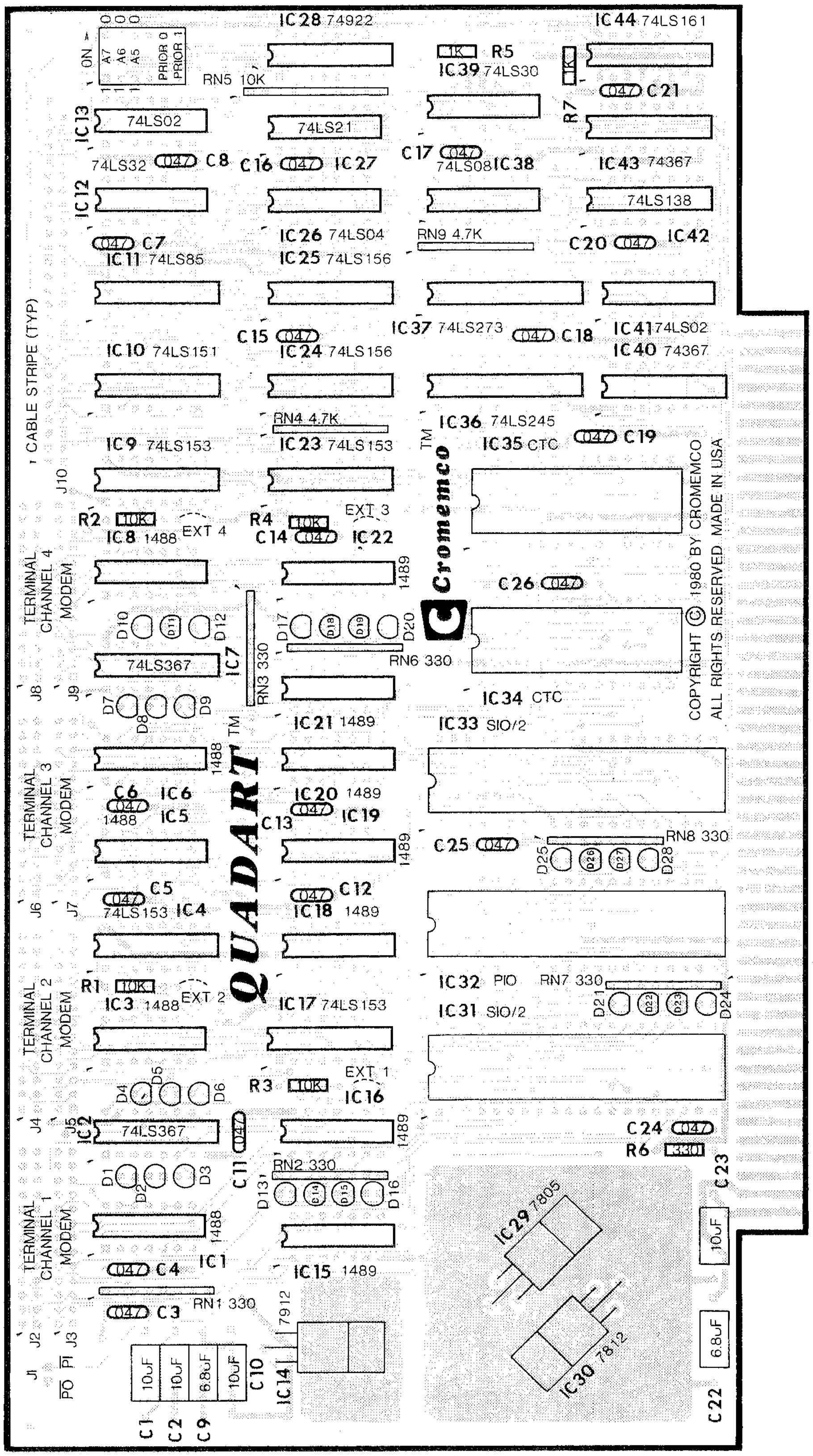
IC23	010-0048	74LS153	1
IC24-25	010-0188	74LS156	2
IC26	010-0066	74LS04	1
IC27	010-0060	74LS21	1
IC28	010-0311	74922	1
IC29	012-0001	Regulator, 7805/340T-5	1
IC30	012-0002	Regulator, 7812	1
IC31	011-0040	Z80A-SIO/2 APS	1
IC32	011-0039	Z80A-PIOAPS	1
IC33	011-0040	Z80A-SIO/2 APS	1
IC34-35	011-0038	Z80A-CTCAPS	2
IC36	010-0120	74LS245	1
IC37	010-0107	74LS273	1
IC38	010-0064	74LS08	1
IC39	010-0059	74LS30	1
IC40	010-0080	74367	1
IC41	010-0068	74LS02	1
IC42	010-0096	74LS138	1
IC43	010-0080	74367	1
IC44	010-0044	74LS161	1

Miscellaneous

SW1	013-0025	6 position DIP switch	1
	015-0013	6-32 hex nut, sm pat, 1/4 cad	5
	015-0020	#6 split ring lock wash, cad	5
	015-0044	6-23x1/2 pan hd slot screw, cad	5
	017-0001	sckt, 14 pin, Burndy 14P-11T	19
	017-0002	sckt, 16 pin, Burndy 16P-11T	13
	017-0004	sckt, 20 pin, Burndy 20P-11T	2
	017-0006	sckt, 40 pin, Burndy 40P-11T	3
	017-0009	sckt wafer, Molex 22-12-2021	1
	017-0014	conn 50 pin r/a, Amp 2-86479-9	1
	017-0030	26 pin conn strip, straight	4
	017-0034	16 pin DIP plug, CA16PLC-08	1
	017-0071	sckt, 28 pin, Burndy 28P-11T	2
	020-0043	Quadart	1
	021-0016	small heatsink	1
	021-0017	lg heatsink, TO-220 Wakefield	1
	021-0109	silicon pad, regulator	3

Documentation

	023-2005	Quadart Instruction Manual	1



Quadart Parts Location Diagram

LIMITED WARRANTY

Cromemco, Inc. ("Cromemco") warrants this product against defects in material and workmanship to the original purchaser for ninety (90) days from the date of purchase, subject to the following terms and conditions.

What Is Covered By This Warranty

During the ninety (90) day warranty period Cromemco will, at its option, repair or replace this Cromemco product or repair or replace with new or used parts any parts or components, manufactured by Cromemco, which prove to be defective, provided the product is returned to an Authorized Cromemco Dealer as set forth below.

How To Obtain Warranty Service

You should immediately notify IN WRITING your Authorized Cromemco Dealer or Cromemco Inc of problems encountered during the warranty period. In order to obtain warranty service, first obtain a return authorization number by contacting the Authorized Cromemco Dealer from whom you purchased the product. Then attach to the product:

1. Your name, address and telephone number,
2. the return authorization number,
3. a description of the problem, and
4. proof of the date of retail purchase.

Ship or otherwise return the product, transportation and insurance costs prepaid, to the Authorized Cromemco Dealer. If you are unable to receive warranty repair from the Authorized Cromemco Dealer from whom you purchased the product, you should contact Cromemco Customer Support at: Cromemco, Inc., 280 Bernardo Ave., Mountain View, Ca. 94043.

What Is Not Covered By This Warranty

Cromemco does not warrant any products, components or parts not manufactured by Cromemco.

This warranty does not apply if the product has been damaged by accident, abuse, misuse, modification or misapplication; by damage during shipment; or by improper service. This product is not warranted to operate satisfactorily with peripherals or products not manufactured by Cromemco. Transportation and insurance charges incurred in transporting the product to and from the Authorized Cromemco Dealer or Cromemco are not covered by this Warranty.

THIS WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, WHETHER ORAL OR WRITTEN, EXPRESS OR IMPLIED. ANY IMPLIED WARRANTIES, INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO NINETY (90) DAYS FROM THE DATE OF PURCHASE OF THIS PRODUCT. CROMEMCO SHALL NOT BE LIABLE FOR INCIDENTAL AND/OR CONSEQUENTIAL DAMAGES FOR THE BREACH OF ANY EXPRESS OR IMPLIED WARRANTY, INCLUDING DAMAGE TO PROPERTY AND, TO THE EXTENT PERMITTED BY LAW, DAMAGES FOR PERSONAL INJURY, EVEN IF CROMEMCO HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOFTWARE, TECHNICAL INFORMATION AND FIRMWARE IS LICENSED "AS IS" AND WITH ALL FAULTS. THE AGENTS, DEALERS, AND EMPLOYEES OF CROMEMCO ARE NOT AUTHORIZED TO MAKE MODIFICATIONS TO THIS WARRANTY, OR ADDITIONAL WARRANTIES BINDING ON CROMEMCO ABOUT OR FOR PRODUCTS SOLD OR LICENSED BY CROMEMCO. ACCORDINGLY, ADDITIONAL STATEMENTS WHETHER ORAL OR WRITTEN EXCEPT SIGNED WRITTEN STATEMENTS FROM AN OFFICER OF CROMEMCO DO NOT CONSTITUTE WARRANTIES AND SHOULD NOT BE RELIED UPON.

SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES OR LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

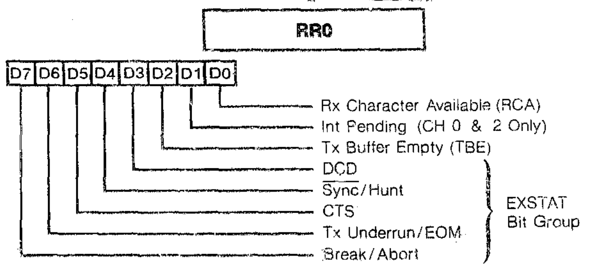
THIS WARRANTY AND THE STATUTE OF LIMITATIONS SHALL RUN CONCURRENTLY WITH ANY ACCEPTANCE PERIOD. THIS WARRANTY IS NOT TRANSFERABLE. NO SUIT, LITIGATION, OR ACTION SHALL BE BROUGHT BASED ON THE ALLEGED BREACH OF THIS WARRANTY OR IMPLIED WARRANTIES MORE THAN ONE YEAR AFTER THE DATE OF PURCHASE IN THOSE JURISDICTIONS ALLOWING SUCH A LIMITATION, OTHERWISE NO SUCH ACTION SHALL BE BROUGHT MORE THAN ONE YEAR AFTER THE EXPIRATION OF THIS WARRANTY.

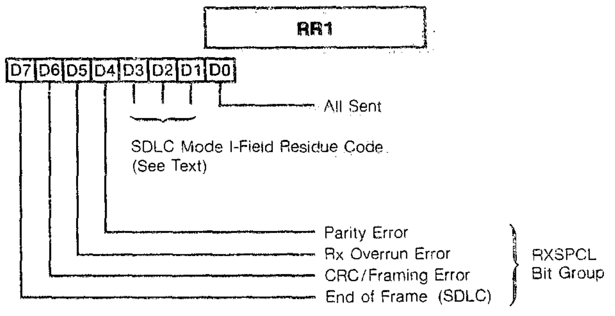
THIS WARRANTY SHALL NOT BE APPLICABLE TO THE EXTENT THAT ANY PROVISION OF THIS WARRANTY IS PROHIBITED BY ANY FEDERAL, STATE OR MUNICIPAL LAW WHICH CANNOT BE PREEMPTED. THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.

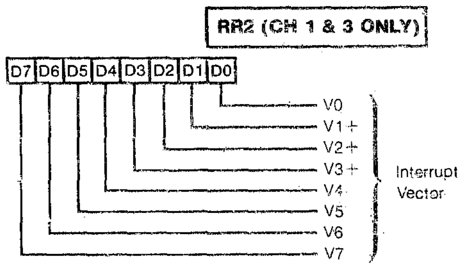
5/81

SIO READ REGISTERS

(Each Channel)







+ V1, V2, V3 Variable If Status Affects Vector Mode Programmed.

2006-33