# Z8030 Z-BUS SCC/ Z8530 SCC Serial Communications Controller

## Technical Manual

September 1986

Zilog

# Z8030 Z-BUS SCC/ Z8530 SCC Serial Communications Controller

Zilog

**Related Zilog Documents**
*Data Book*, 00-2034-04
*Z8030/8530 SCC Product Specification*, 00-2439-01
*Design Considerations Using Quartz Crystals With Zilog's Components*, 00-2802-01

The information contained herin is subject to change without notice. Zilog assumes no responsibility for the uses of any circuitry other than circuitry embodied in a Zilog product. No other circuit patent licenses are implied.

All specifications (parameters) are subject to change without notice. The applicable Zilog test documentation will specify which parameters are tested.

# Table of Contents

# PREFACE

Thank you for your interest in the SCC, one of the most versatile and most popular Serial Data Communications ICs. This manual is intended to provide answers to all technical questions about the Z8030 and the Z8530, the two slightly different versions of the SCC. The Product Specification, document 00-2439-01, contains ac and dc characteristics. Please ready this Preface where we try to anticipate your questions.

- If you are new to serial data communications, you will need additional tutorial information. Of the many introductory texts on this subject, *Technical Aspects of Data Communications* by John E. McNamara, published by Digital Press (DEC) 1982, is one of the best.
- If you have designed with simpler UARTs and USARTs, and HDLC/SDLC devices, the SCC offers you far greater flexibility, but also requires an in-depth study and understanding of the impact and the use of its many powerful features. This manual contains important information.
- If you are familiar with the Z80-SIO, you will feel right at home with the SCC, for it is really a functionally enhanced superset of the Z80-SIO.

Most users read only chapters that are of interest to them. If you are designing the microcomputer hardware structure using the SCC as a peripheral, you will want to read the Applications Section in Chapter 8, and the Interrupt Routine Section in the same chapter.

If you are programming a system using the SCC, you will be more interested, on the worksheet in Chapter 8.

Points To Watch Out For :

1. Follow the worksheet for initialization (Chapter 8.1). Unexplainable operations may occur if this procedure is not followed.
2. Watch out for Write Recovery time violation (Chapter 3). Both the CPU clock rate and the SCC clock rate will affect the Write Recovery time.
3. Ensure Mode bits are not changed when writing Commands. (Chapter 8.1.2). Each Mode bit affects only one function and a Command bit entry requires a rewrite of the entire register; therefore, care must be taken to insure the integrity of the Mode bits whenever a new command is issued.
4. Data must be valid prior to falling edge of $\overline{WR}$ or $\overline{DS}$.
5. If not used, $\overline{INTACK}$ should be tied high.

CHAPTER 1

# GENERAL INFORMATION

## 1.0 INTRODUCTION

The SCC Serial Communications Controller is a dual-channel, multiprotocol data communications peripheral designed for use with 8- and 16-bit microprocessors. The SCC functions as a serial-to-parallel, parallel-to-serial converter/controller. The SCC can be software-configured to satisfy a wide variety of serial communications applications. The device contains a variety of new, sophisticated internal functions including on-chip baud rate generators, digital phase-lock loops, and crystal oscillators, which dramatically reduce the need for external logic.

The SCC handles asynchronous formats, Synchronous byte-oriented protocols such as IBM Bisync, and Synchronous bit-oriented protocols such as HDLC and IBM SDLC. This versatile device supports virtually any serial data transfer application (telecommunications, cassette, diskette, tape drivers, etc.).

The device can generate and check CRC codes in any Synchronous mode and can be programmed to check data integrity in various modes. The SCC also has facilities for modem controls in both channels. In applications where these controls are not needed, the modem controls can be used for general-purpose I/O.

With access to 14 Write registers and 7 Read registers per channel, the user can configure the SCC so that it can handle all asynchronous formats regardless of data size, number of stop bits, or parity requirements. The SCC also accommodates all synchronous formats including character, byte, and bit-oriented protocols.

Within each operating mode, the SCC also allows for protocol variations by checking odd or even parity bits, character insertion or deletion, CRC generation and checking, break and abort generation and detection, and many other protocol-dependent features.

The SCC is offered in two versions. The Z8030 is directly compatible with the Z8000 and 8086 CPUs. The Z8530 is designed for non-multiplexed buses and is easily interfaced to CPUs such as the 8080, Z80, 6800, 68000 and *Multibus.

## 1.1 CAPABILITIES

**Two independent full-duplex channels.**

**Synchronous/Isosynchronous data rates:**

- Up to 1/4 of the PCLK (i.e., 1 Mbit/sec. maximum data rate with 4 MHz PCLK. Using external phase-lock loop.
- Up to 375 Kbit/sec. with a 6 MHz clock rate. Up to 250 Kbit/sec. with a 4 MHz clock rate (FM encoding using digital phase-locked loop).
- Up to 187.5 Kbit/sec. with a 6 MHz clock rate Up to 125 Kbit/second with a 4 MHz clock rate (NRZI encoding using digital phase-locked loop).

**Asynchronous capabilities:**

- 5, 6, 7, or 8 bits per character
- 1, 1-1/2, or 2 stop bits
- Odd or even parity
- Times 1, 16, 32, or 64 clock modes
- Break generation and detection
- Parity, overrun and framing error detection.

**Byte-oriented synchronous capabilities:**

- Internal or external character synchronization
- 1or 2 sync characters (6 or 8 bits/character) in separate registers
- Automatic Cyclic redundancy check (CRC) generation/detection

**SDLC/HLDC capabilities:**

- Abort sequence generation and checking
- Automatic zero insertion and deletion
- Automatic flag insertion between messages
- Address field recognition
- I-field residue handling
- CRC generation/detection
- SDLC loop mode with EOP recognition/loop entry and exit.

**Receiver data registers quadruply buffered. Transmitter data registered double buffered.**

**NRZ, NRZI, or FM encoding/decoding.**

**Baud-rate generator in each channel.**

**Digital phase-locked loop for clock recovery.**

**Crystal oscillator.**

## 1.2 BLOCK DIAGRAM

Figure 1-1 and Figure 2-1 are block diagrams of the SCC. Received data enters the receive data pins and follows one of several data paths, depending on the state of the control logic. The contents of the registers and the state of the external control pins establish the internal control logic. Transmitted data follows a similar pattern of control, register, and external pin definition.

## 1.3 PIN FUNCTIONS

The SCC pins are divided into seven functional groups: Address/Data, Bus Timing and Reset, Device Control, Interrupt, Serial Data (both channels), Peripheral Control (both channels), and Clocks (both channels). Figure 1-2, 1-3 show the pins in each functional group for both the Z8030 and Z8530. Notice the pin functions unique to each version in the Address/Data group, Bus Timing and Reset group, and Control groups.

The Address/Data group consists of the bidirectional lines used to transfer data between the CPU and the SCC (Addresses in the Z8030 are latched by $\overline{AS}$). The direction of these lines depends on whether the SCC is selected and whether the operation is a Read or a Write.

The Timing and Control groups designate the type of transaction to occur and when this transaction will occur. The Interrupt group provides inputs and outputs to conform to the Z-BUS specifications for handling and prioritizing interrupts. The remaining groups are divided into Channel A and Channel B groups for serial data (transmit or receive), peripheral control (such as DMA or modem), and the input and output lines for the receive and transmit clocks.

## 1.4 PIN DESCRIPTIONS

The SCC is available with two sets of bus interface timings: one for multiplexed systems and one for nonmultiplexed.

The pin descriptions here describe the Z8030-specific pins, the Z8530-specific pins, and the pins that are identical for both versions. Figure 1-3 designates the pin locations and signal names for the Z8030 SCC. Figure 1-4 designates the pin locations and signal names for the Z8530 SCC version.

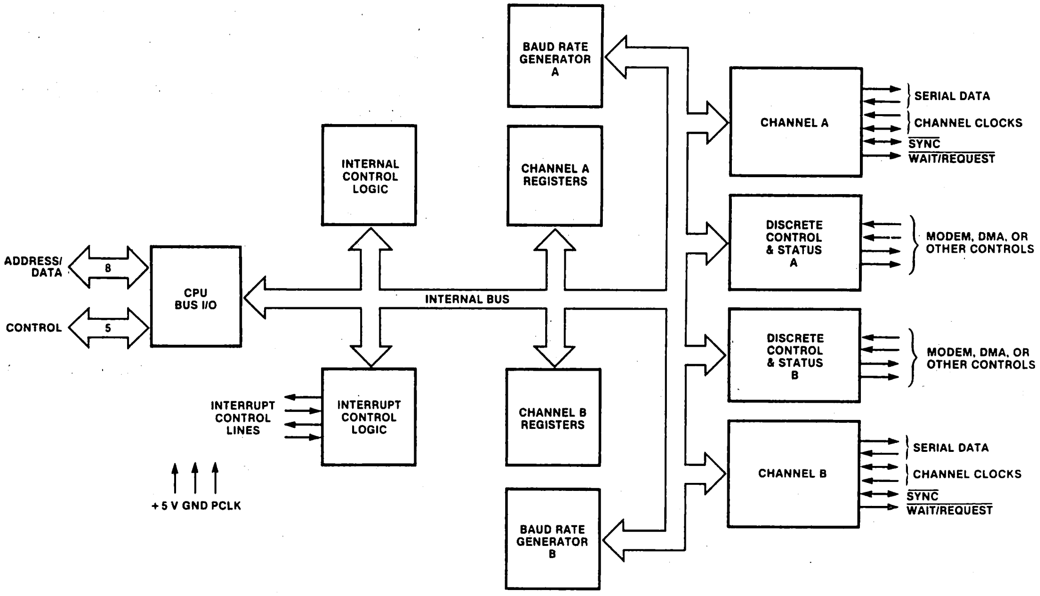


**Figure 1-1 Z-SCC Block Diagram**

### 1.4.1 Pin Descriptions (Z8030 SCC only)

**AD0–AD7.** Address/Data Bus (bidirectional, active HIGH 3-state). These multiplexed lines carry register addresses to the SCC as well as data or control information to and from the SCC.

**$\overline{\text{AS}}$.** Address Strobe (input, active LOW). Addresses on AD0–AD7 are latched by the rising edge of this signal.

**$\overline{\text{CS}}_0$.** Chip Select 0 (input active LOW). This signal is latched concurrently with the addresses on AD0–AD7 and must be active for the intended bus transaction to occur.

**$\text{CS}_1$.** Chip Select 1 (input, active HIGH). This second select signal must also be active before the intended bus transaction can occur. CS1 must remain active throughout the transaction.

**$\overline{\text{DS}}$.** Data Strobe (input, active LOW). This signal provides timing for the transfer of data into and out of the SCC. If AS and DS are both LOW, this is interpreted as a Reset.

**R/$\overline{\text{W}}$.** Read/Write (input, Read active HIGH). This signal specifies whether the operation to be

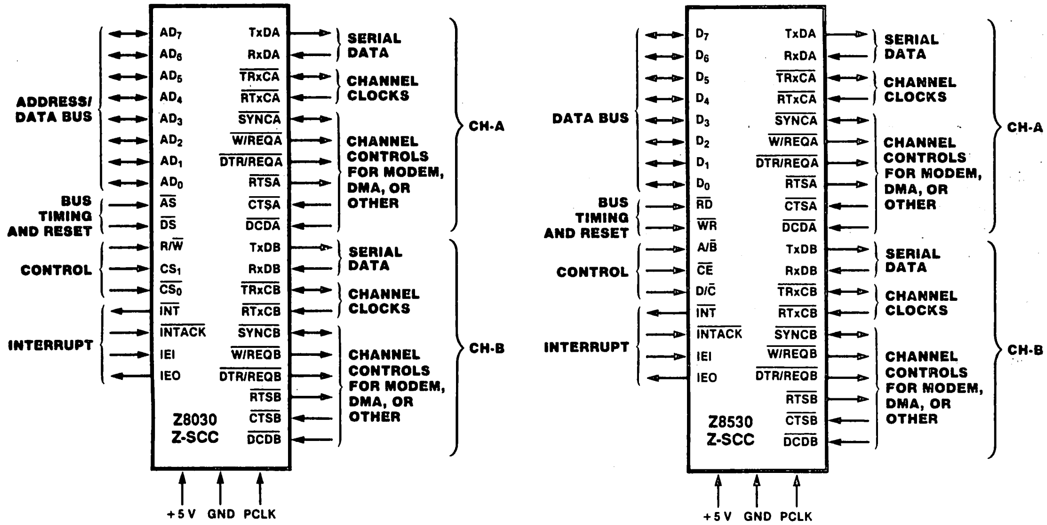
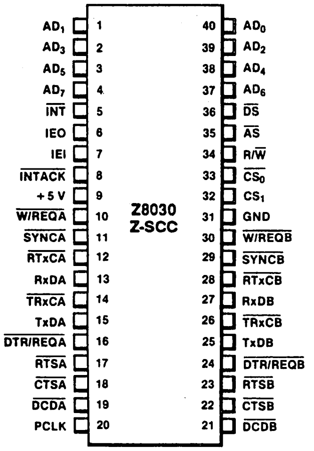
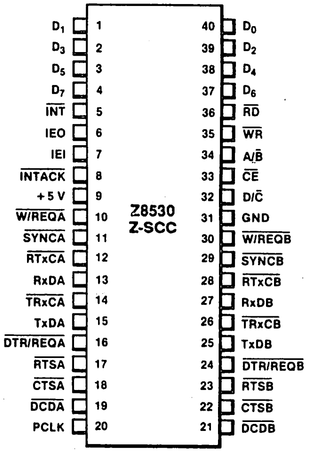


**Figure 1-2 Pin Functions SCC/AmZ-SCC**





07613A 1-3

07613A 1-4

**Figure 1-3 Pin Designation for AmZ8030 SCC**

**Figure 1-4 Pin Designation for AmZ8530 SCC**

performed is a Read or a Write.

### 1.4.2 Pin Descriptions (Z8530 SCC only)

**A/$\overline{\text{B}}$.** Channel A/Channel B Select (input, Channel A active HIGH). This signal selects the channel in which the Read or Write operation occurs.

**$\overline{\text{CE}}$.** Chip Enable (input, active LOW). This signal selects the SCC for operation. It must remain active throughout the bus transaction.

**D0–D7.** Data Lines (bidirectional, 3-state). These I/O lines carry data or control information to and from the SCC.

**D/$\overline{\text{C}}$.** Data/Control (input, Data active HIGH). This signal defines the type of information transfer performed by the SCC: data or control.

**$\overline{\text{RD}}$.** Read (input, Active LOW). This signal indicates a Read operation and when the SCC is selected, enables the SCC bus drivers. During the interrupt acknowledge cycle, this signal gates the interrupt vector onto the bus if the SCC is the highest priority device requesting an interrupt.

**$\overline{\text{WR}}$.** Write (input, active LOW). When the SCC is selected, this signal indicates a Write operation. The coincidence of $\overline{\text{RD}}$ and $\overline{\text{WR}}$ is interpreted as a Reset.

### 1.4.3 Pin Descriptions (both Z8030 and Z8530)

**$\overline{\text{CTSA}}$, $\overline{\text{CTSB}}$.** Clear to Send (inputs, active LOW). If these pins are programmed as auto enables, a LOW on these inputs enables the respective transmitters. If not programmed as auto enables, they may be used as general-purpose inputs. Both inputs are Schmitt-trigger buffered to accommodate slow rise-time inputs. The SCC detects transitions on these inputs and can interrupt the CPU on either logic level transitions.

**$\overline{\text{DCDA}}$, $\overline{\text{DCDB}}$.** Data Carrier Detect (inputs, active LOW). These pins function as receiver enables if they are programmed as auto enable bits; otherwise they may be used as general-purpose input pins. Both pins are Schmitt-trigger buffered to accommodate slow rise-time signals. The SCC detects transitions on these pins and can interrupt the CPU on either logic level transitions.

**$\overline{\text{DTR}}$/$\overline{\text{REQA}}$, $\overline{\text{DTR}}$/$\overline{\text{REQB}}$.** Data Carrier Detect (~~inputs~~ OUTPUT, active LOW). These pins function as receiver enables if they are programmed into the DTR bit. They can also be used as general-purpose outputs (transmit) or as request lines for the DMA controller. The SCC allows full duplex DMA transfers.

**IEI.** Interrupt Enable In (input, active HIGH). IEI is used with IEO to form an interrupt daisy chain when there is more than one interrupt-driven device. A HIGH on IEI indicates that no other higher priority device has an Interrupt Under Service (IUS) or is requesting an interrupt.

**IEO.** Interrupt Enable Out (output, active HIGH). IEO is HIGH only if IEI is HIGH and the CPU is not servicing an SCC or SCC interrupt or the controller is not requesting an interrupt (interrupt acknowledge cycle only). IEO is connected to the next lower priority device's IEI input and thus inhibits interrupts from lower priority devices.

**$\overline{\text{INTACK}}$.** Interrupt Acknowledge (input, active LOW). This signal indicates an active interrupt acknowledge cycle. During this cycle, the interrupt daisy chain settles. When $\overline{\text{RD}}$ or $\overline{\text{DS}}$ becomes active, the SCC places an interrupt vector on the data bus (if IEI is HIGH). $\overline{\text{INTACK}}$ is latched by the rising edge of $\overline{\text{AS}}$ or PCLK.

**$\overline{\text{INT}}$.** Interrupt Request (output, open-drain, active LOW). This signal is activated when the SCC is requesting an interrupt.

**PCLK.** Clock (input). This is the master clock used to synchronize internal signals. PCLK is not required to have any phase relationship with the master system clock. PCLK is a TTL level signal.

**$\overline{\text{RTSA}}$, $\overline{\text{RTSB}}$.** Request to Send (outputs, active LOW). When the Request to Send (RTS) bit in Write Register 5 (Figure 7-7) is set, the $\overline{\text{RTS}}$ signal goes LOW. When the RTS bit is reset in the Asynchronous mode and auto enables is on, the signal goes HIGH after the transmitter is empty. In Synchronous mode or in Asynchronous mode with auto enables off, the $\overline{\text{RTS}}$ pins strictly follow the state of the RTS bit. Both pins can be used as general-purpose outputs.

**$\overline{\text{RTxCA}}$, $\overline{\text{RTxCB}}$.** Receive/Transmit Clocks (inputs, active LOW). The functions of these pins are under program control. In each channel, $\overline{\text{RTxC}}$ may supply the receive clock, the transmit clock, the clock for the baud rate generator, or the clock for the digital phase-locked loop (refer to Section 4 for bit configurations). These pins can also be programmed for use the the respective $\overline{\text{SYNC}}$ pins as a crystal oscillator. The receive clock may be 1, 16, 32, or 64 times the data rate in asynchronous modes.

**RxDA, RxDB.** Receive Data (inputs, active HIGH).

These input signals receive serial data at standard TTL levels.

**$\overline{\mathrm{SYNCA}}$, $\overline{\mathrm{SYNCB}}$.** Synchronization (inputs/outputs, active LOW). These pins can act as either inputs, outputs, or as part of the crystal oscillator circuit. In the Asynchronous Receive mode (crystal oscillator option not selected), these pins are inputs similar to $\overline{\mathrm{CTS}}$ and $\overline{\mathrm{DCD}}$. In this mode, transitions on these lines affect the state of the Sync/Hunt status bits in Read Register 0 (Figure 7-18), but have no other function.

In External Synchronization mode with the crystal oscillator not selected, these lines also act as inputs. In this mode, $\overline{\mathrm{SYNC}}$ must be driven LOW two receive clock cycles after the last bit in the sync character is received. Character assembly begins on the rising edge of the receive clock immediately preceding the activation of $\overline{\mathrm{SYNC}}$.

In the Internal Synchronization mode, (Monosync and Bisync) with the crystal oscillator not selected, these pins act as outputs and are active only during the part of the receive clock cycle in which sync characters are recognized. The sync condition is not latched, so these outputs are active each time a sync character is recognized (regardless of character boundaries). In SDLC mode, these pins act as outputs and are valid on receipt of a flag.

**$\overline{\mathrm{TRxCA}}$, $\overline{\mathrm{TRxCB}}$.** Transmit/Receive Clocks (inputs or outputs, active LOW). The functions of these pins are under program control. $\overline{\mathrm{TRxC}}$ may supply the receive clock or the transmit clock in the Input mode or supply the output of the digital phase-locked loop, the crystal oscillator, the baud rate generator, or the transmit clock in the output mode. (Refer to Section 4 for bit configuration.)

**TxDA, TxDB.** Transmit Data (outputs, active HIGH). This output signal transmits serial data at standard TTL levels.

**$\overline{\mathrm{W}}/\overline{\mathrm{REQA}}$, $\overline{\mathrm{W}}/\overline{\mathrm{REQB}}$.** Wait/Request (outputs, open drain when programmed for Wait function, driven HIGH or LOW when programmed for a Request function). These dual-purpose outputs can be programmed as Request (receive) lines for a DMA controller or as Wait lines to synchronize the CPU to the SCC data rate. The reset state is Wait. The SCC allows full duplex DMA transfer.

CHAPTER 2

# OVERVIEW

## 2.0 INTRODUCTION

The SCC internal structure provides all the interrupt and control logic necessary to interface with multiplexed and non-multiplexed buses. Interface logic is also provided to monitor modem or peripheral control inputs and outputs. All of the control signals are general purpose and can be applied to various peripheral devices as well as used for modem control.

The center for data activity revolves around the internal read and write registers. The programming of these registers provides the SCC with functional "personality"; i.e., register values can be assigned before or during program sequencing to determine how the SCC will establish a given communication protocol.

## 2.1 REGISTER FUNCTIONS

All modes of communication are established by the bit values of the write registers. As data is received or transmitted, read register values may change. These changed values can promote software action or internal hardware action for further register changes.

The register set for each channel includes 14 write registers and seven read registers. Ten write registers are used for control, two for sync character generation, and two for baud rate generation. In addition there are two write registers which are shared by both channels; one is the interrupt vector register and one is the master interrurpt control and reset register. Four read registers indicate status information, two are for baud rate generation, and one for the receive buffer. In addition there are two read registers which are shared by both channels; one for the interrupt pending bits and one for interrupt vector.

Table 2-1 lists the assigned functions for each read and write register. The SCC contains only one WR2 (interrupt vector) and one WR9 (master interrupt control). Both registers are accessed and shared by either channel. Chapter 7 provides a detailed bit legend and description of each register.

**Table 2-1 Register Set**

| **Read Register Functions** | |
|---|---|
| RR0 | Transmit/Receive buffer status, and External status |
| RR1 | Special Recieve Condition status, residue codes, error conditons |
| RR2 | Modified (Channel B only) interrupt vector and Unmodified interrupt vector (Channel A only) |
| RR3 | Interrupt Pending bits (Channel A only) |
| RR8 | Receive buffer |
| RR10 | Miscellaneous XMTR, RCVR status parameters |
| RR12 | Lower byte of baud rate generator time constant |
| RR13 | Upper byte of baud rate generator time constant |
| RR15 | External/Status interrupt control information |
| **Write Register Functions** | |
| WR0 | Command Register, (Register Pointers, Z8530 only), CRC initialization, resets for various modes |
| WR1 | Interrupt conditions, Wait/DMA request control |
| WR2 | Interrupt vector (access through either channel) |
| WR3 | Receive/Control parameters, number of bits per character, Rx CRC enable |
| WR4 | Transmit/Receive miscellaneous parameters and codes, clock rate, number of sync characters, stop bits, parity |
| WR5 | Transmit parameters and control, number of Tx bits per character, Tx CRC enable |
| WR6 | Sync character (1st byte) or SDLC flag |
| WR8 | Transmit buffer |
| WR9 | Master interrupt control and reset (accessed through either channel), reset bits, control interrupt daisy chain |
| WR10 | Miscellaneous transmitter/receiver control bits, NRZI, NRZ, FM encoding, CRC reset |
| WR11 | Clock mode control, source of Rx and Tx clocks |
| WR12 | Lower byte of baud rate generator time constant |
| WR13 | Upper byte of baud rate generator time constant |
| WR14 | Miscellaneous control bits: baud rate generator, Phase-Locked Loop control, auto echo, local loopback |
| WR14 | External/Status interrupt control information-control external conditions causing interrupts |

## 2.2 DATA PATHS

Figure 2-1 illustrates the data paths involved in the six major areas of the SCC:

- Transmitter
- Receiver
- Baud rate generator
- DPLL
- Clocking options
- Data encoding.

All communication modes are established by programming the write registers. As data is received or transmitted, read register values may change, altering the direction of the data path. These changed values can promote software action or internal hardware action for further register changes.

### 2.2.1 Transmitter

The transmitter has an 8-bit Transmit Data register (WR8) loaded from the internal data bus and a Transmit Shift register loaded from either WR6, WR7, or the Transmit Data register. In byte-oriented modes, WR6 and WR7 can be programmed with sync characters. In Monosync mode, an 8-bit or 6-bit sync character is used (WR6), whereas a 16-bit sync character is used (WR6 and WR7) in Bisync mode. In bit-oriented synchronous modes, the flag contained in WR7 is loaded into the Transmit Shift register at the beginning and end of a message.

If asynchronous data is processed, WR6 and WR7 are not used and the Transmit Shift register is formatted with start and stop bits shifted out to the transmit multiplexer at the selected clock rate. Synchronous data (except SDLC/HDLC) is shifted to the CRC generator as well as to the transmit multiplexer.

SDLC/HDLC data is shifted to the CRC Generator and out through the zero insertion logic (which is disabled while the flags are being sent). A "0" is inserted in all address, control, information, and frame check fields following five contiguous "1s" in the data stream. The result of the CRC generator for SDLC data is also routed through the zero insertion logic and then to the transmit multiplexer.

### 2.2.2 Receiver

The receiver has a three deep 8-bit Data FIFO (paired with an 8-bit Error FIFO), and an 8-bit shift register. This arrangement creates a 3-byte delay time, which allows the CPU time to service an interrupt at the beginning of a block of high-speed data. With each Receive Data FIFO, the Error FIFO stores parity and framing errors and other types of status information. The Error FIFO is readable in Read Register 1.

Incoming data is routed through one of several paths depending on the mode and character length. In Asynchronous mode, serial data enters the 3-bit delay (Figure 2-1) if the character length of seven or eight bits is selected. If a character length of five or six bits is selected, data enters the receive shift register directly.

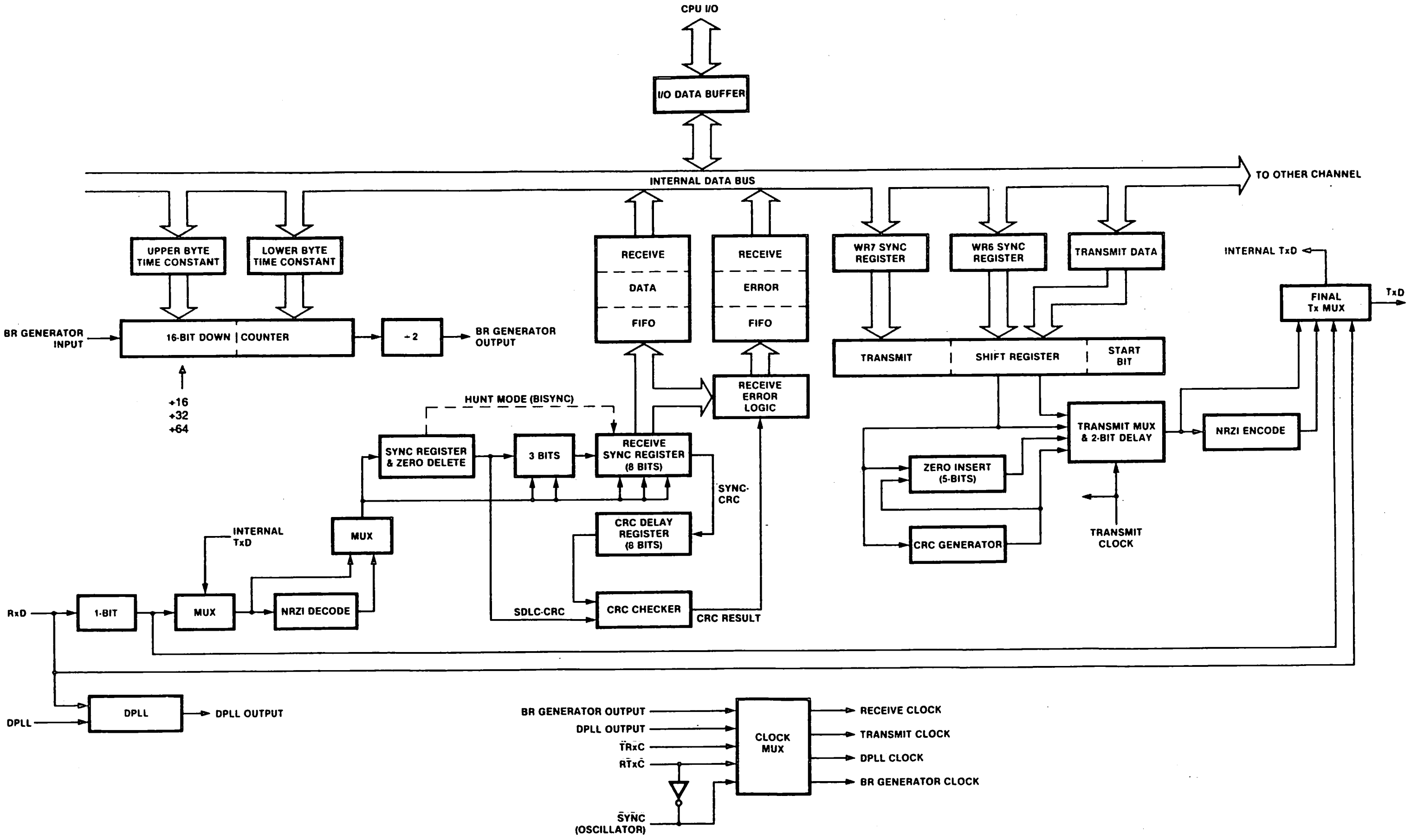
In synchronous modes, the data path is determined by the phase of the receive process currently in operation. A synchronous receive operation begins with a hunt phase in which a bit pattern that matches the programmed sync characters (6-, 8-, or 16-bit is searched.

The incoming data then passes through the Sync register and is compared to a sync character stored in WR6 or WR7 (depending on which mode it is in). The Monosync mode matches the sync character programmed in WR7 and the character assembled in the Receive Sync register to establish synchronization.

Synchronization is achieved differently in the Bisync mode. Incoming data is shifted to the Receive Shift register while the next eight bits of the message are assembled in the Receive Sync register. If these two characters match the programmed characters in WR6 and WR7, synchronization is established. Incoming data can then bypass the Receive Sync register and enter the 3-bit delay directly.

The SDLC mode of operation uses the Receive Sync register to monitor the receive data stream and to perform zero deletion when necessary; i.e., when five continuous "1s" are received, the sixth bit is inspected and deleted from the data stream if it is "0." The seventh bit is inspected only if the sixth bit equals one. If the seventh bit is "0," a flag sequence has been received and the receiver is synchronized to that flag. If the seventh bit is a "1," an abort or an EOP (End Of Poll) is recognized, depending on the selection of either the normal SDLC mode or SDLC Loop mode.

The same path is taken by incoming data for both SDLC modes. The reformatted data enters the 3-bit delay and is transferred to the Receive Shift register. The SDLC receive operation begins in the hunt phase by attempting to match the



07513A 2-1

Figure 2-1 Data Paths

assembled character in the Receive Shift Register with the flag pattern in WR7. Then the flag character is recognized, subsequent data is routed through the same path, regardless of character length.

Either the CRC-16 or CRC-SDLC cyclic redundancy check (CRC) polynomial can be used for both Monosync and Bisync modes, but only the CRC-SDLC polynomial is used for SDLC operation. The data path taken for each mode is also different. Bisync protocol is a byte-oriented operation that requires the CPU to decide whether or not a data character is to be included in CRC calculation. An 8-bit delay in all synchronous modes except SDLC is allowed for this process. In SDLC mode, all bytes are included in the CRC calculation.

### 2.2.3 Baud Rate Generator

Each channel in the SCC contains a programmable baud rate generator. Each generator consists of two 8-bit, time-constant registers forming a 16-bit time constant, a 16-bit down counter, and a flip-flop on the output that makes the output a square wave. On start-up, the flip-flop on the output is set High so that it starts in a known state, the value in the time-constant register is again loaded into the counter, and the counter begins counting down. When a count of zero is reached, the output of the baud rate generator toggles, the value in the time-constant register is loaded into the counter, and the process starts over. The time constant can be changed at any time, but the new value does not take effect until the next load of the counter.

No attempt is made to synchronize the loading of a new time constant with the clock used to drive the generator. When the time constant is to be changed, the generator should be stopped by writing to an enable bit in WR14. This ensures the loading of a correct time constant.

If neither the transmit clock nor the receive clock are programmed to come from the $\overline{\text{TRXC}}$ pin, the output of the baud rate generator may be made available for external use on the $\overline{\text{TRXC}}$ pin. Section 3.9 presents the formula for determining the time constant for a given baud rate.

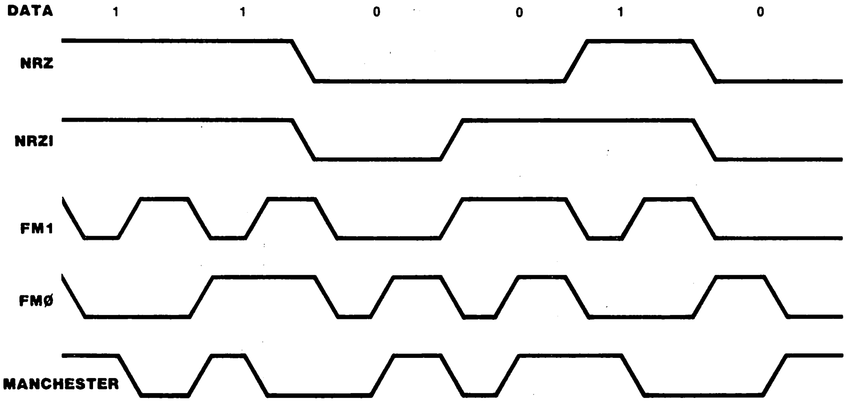
### 2.2.4 Digital Phase-Locked Loop (DPLL)

The SCC contains a digital phase-locked loop that can be used to recover clock information from a data stream with NRZI or FM coding. The DPLL is driven by a clock nominally 32 (NRZI) or 16 (FM) times the data rate. The DPLL uses this clock, along with the data stream, to construct a receive clock for the data. This clock can then be used as the SCC receive clock, the transmit clock, or both. Section 3.10 details the clock recovery for each of the different forms of encoding.

### 2.2.5 Clocking Options

The SCC can select several clock sources for internal and external use. Write Register 11 is the Clock Mode Control register for both the receive and transmit clocks. It determines the type of signal on the $\overline{\text{SYNC}}$ and $\overline{\text{RTxC}}$ pins and the direction of the $\overline{\text{TRxC}}$ pin.

Write Register 11 also controls the output of the baud rate generator, the DPLL output, and the



07666A 2-2

Figure 2-2 Data Encoding Methods

selection of either a TT1 or XTAL output for the $\overline{\text{RTxC}}$ pin. (Section 3.11 gives a detailed description of the clocking options.)

### 2.2.6. Data Encoding

Figure 2-2 illustrates the four encoding methods used by the SCC. In NRZ encoding, a "1" is represented by a High level and a "0" is represented by a Low level. In NRZI encoding, a "1" is represented by no change in level and a "0" is represented by a change in level. In FM1 (more properly, biphase mark), a transition occurs at the beginning of every bit cell. A "1" is represented by an additional transition at the center of the bit cell and a "0" is represented by the absence of a transition at the center of the bit cell. In FM0 (more properly, biphase space), a transition occurs at the beginning of every bit cell. A "0" is represented by an additional transition at the center of the bit cell and a "1" is represented by the absence of a transition at the center of the bit cell.

In addition to these four methods, the SCC can be used to decode Manchester (biphase level) data by using the DPLL in the FM mode and programming the receiver for NRZ data. Manchester encoding always produces a transition at the center of the bit cell. If the transition is Low to High, the bit is "0." If the transition is High to Low, the bit is "1."

## 2.3 DATA COMMUNICATIONS CAPABILITIES

SCC logic handles all asynchronous, byte-oriented synchronous, and bit-oriented synchronous modes of operation. The following section briefly describes asynchronous, synchronous, and SDLC modes of communication.

### 2.3.1 Asynchronous

Figure 2-3 represents a typical asynchronous message format using one start bit, seven data bits, one parity bit, and one stop bit. A start bit is a High-to-Low transition detected by an asynchronous receiver and is actually an information bit notifying the receiver of an incoming message.

The start bit also initiates a clock circuit to provide latching pulses during expected data bit intervals. The parity bit is provided for error checking. The parity bit is calculated in both the receiver and the transmitter; the two results are compared to ensure that the expected and the actual bit values match.

The stop bit returns the message unit to the quiescent marking state; i.e., a constant high state condition lasts until the next High-to-Low start bit indicates an incoming data byte. During reception, the start and stop bits are stripped away and checked for errors, leaving only the working data for CPU interaction. The number of selected bits for each asynchronous function may differ between the transmitter and the receiver.

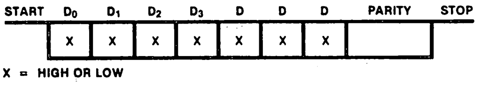
### 2.3.2 Monosync Mode

Monosync and Bisync modes require clocking information to be transmitted along with the data either by a method of encoding data that contains clocking information, or by a modem that encodes or decodes clock information in the modulation process.

Start and stop bits are not required in synchronous modes. All bits are used to transmit data. This eliminates the "waste" characteristic of asynchronous communication.

Figure 2-4 shows the character format for synchronous transmission. For example, bits 1–8 might be one character and bits 9–13 part of another character; or bit 1 might be part of one character, bits 2–9 part of a second character, and bits 10–13 part of a third character. The framing (where each character begins) of each character is accomplished by defining a synchronization character, commonly called a "sync character."

The CPU places the receiver in Hunt mode whenever transmission begins (or whenever a data dropout has occurred and the hardware determines that resynchronization is necessary). In Hunt mode, the receiver shifts a bit into the Receive Shift register and compares the contents



07666A 2-3

Figure 2-3 Asynchronous Message Format

of the Receive Shift register and with the sync character (stored in another register), repeating the process until a match occurs. When a match occurs, the receiver begins transferring bytes to the receive FIFO.

### 2.3.3 Bisynchronous Mode

The Bisync mode of operation (Figure 2-5) is similar to the Monosync mode, except that two sync characters are provided instead of one. Bisync attempts a more structured approach to synchronization through the use of special characters as message "headers" or "trailers". A detailed description of IBM's Bisync can be found in McNamara's Book (See Preface).

### 2.3.4 External Sync Mode

External Sync mode (Figure 2-6) eliminates the use of sync characters in the serial data stream by providing an external sync signal to mark the beginning of a data field; i.e., an external input pin (Sync) waits for an active state change to indicate the beginning of an information field.

### 2.3.5 SDLC Mode

Synchronous Data Link Control mode (SDLC) uses synchronization characters similar to Bisync and Monosync modes (such as flags and pad characters), but it is a bit-oriented protocol instead of byte-oriented protocol.

Any data communication link involves at least two stations. The station that is responsible for the data link and issues the commands to control the link is called the "primary station." The other station is a "secondary station." Not all information transfers need to be initiated by a primary station. In SDLC mode, a secondary station can be the initiator.

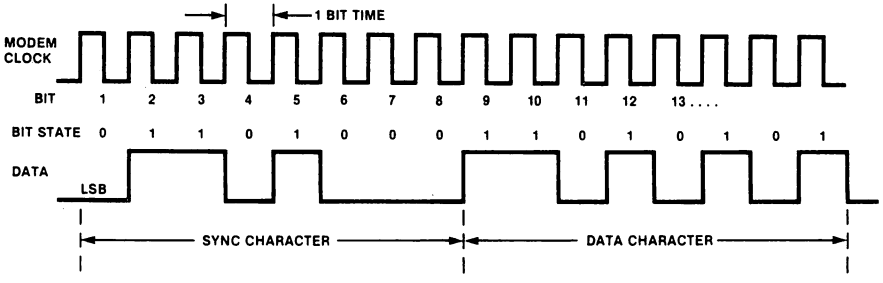
The basic format for SDLC is a "frame" (Figure 2-7). The information field is not restricted in format or content and can be of any reasonable length (including zero). Its maximum length is that which can be expected to arrive at the receiver error- free most of the time. Hence, the determination of maximum length is a function of communication channel error rate.

The two flags that delineate the SDLC frame serve as reference points when positioning the address and control fields, and they initiate the transmission error check. The ending flag indicates to the receiving station that the 16 bits just received constitute the frame check. The ending flag could be followed by another frame, another flag, or an idle. This means that when two frames follow one another, the intervening flag may simultaneously be the ending flag of the first frame and the begin-ning flag of the next frame. Since the SDLC mode does not use characters of defined length, but rather works on a bit-by-bit basis, the 01111110 (7EH) flag can be recognized at any time.

To ensure that the flag is not sent accidentally, SDLC procedures require a binary "0" to be inserted by the transmitter after the transmission of any five contiguous "1s." The receiver then removes the "0" following a received succession of five "1s." Inserted and removed "0s" are not included in the CRC calculation.

The address field is 8 bits long and designates the number of secondary station to which the commands or data from the primary station are sent. The control field is eight bits long and is used to initiate all SDLC activities (see Section 3.6).

The SCC can also serve the High-level synchro-



07666A 2-4

Figure 2-4 Monosync Data Character Format

nous Data Link Communication (HDLC) protocol, which is identical to SDLC except for differences in framing.

### 2.3.6 SDLC Loop Mode

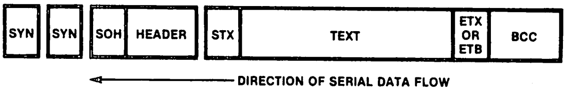
The SCC supports SDLC Loop mode in addition to normal SDLC. SDLC Loop mode is very similar to normal SDLC but is usually used in application where a point-to-point network is not appropriate (for example, Point-Of-Sale terminals). In an SDLC Loop there is a primary station, called the controller, that manages the message traffic flow on the loop, and there are any number of secondary stations.

A secondary station in an SDLC loop is always listening to the messages being sent around the loop, and must pass these messages to the rest of the loop by retransmitting them with a one-bit- time delay. The secondary station can only place its own message on the loop at specific times. The controller signals that secondary stations may transmit messages by sending a special character, called an EOP (End of Poll), around the loop. The EOP character is the bit pattern 11111110. Because of zero insertion during messages this bit pattern is unique and thus is easily recognized.

When a secondary station has a message to transmit and it recognizes an EOP on the line, the first thing that it does is to change the last 1 or the EOP to a "0" before transmitting it. This turns the EOP into a Flag sequence. The secondary station now places its message on the loop and terminates its message with an EOP. Any secondary stations further down the loop with messages to transmit can then append its message to the message of the first secondary station by the same process. All secondary stations without messages to send merely echo the incoming messages and are prohibited from placing messages on the loop, except upon recognizing an EOP.

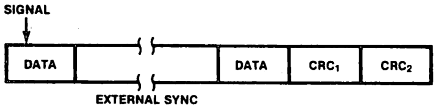
There are also restrictions as to when and how a secondary station physically becomes part of the loop. A secondary station that has just powered up must monitor the loop, without the one- bit-time delay, until it recognizes an EOP. When an EOP is recognized the one-bit-time delay is switched on. This does not disturb the loop because the line is marking idle between the time that the controller sends the EOP and the time that it receives the EOP back. The secondary station that has gone on-loop cannot place a message on the loop until the next time that an EOP is issued by the controller. A secondary station goes off-loop in a similar manner. When given a command to go off-loop, the secondary station waits until the next EOP to remove the one-bit-time delay.

To operate the SCC in SDLC Loop mode, the SCC must first be programmed just as if normal SDLC were to be used. Loop mode is then selected by writing the appropriate control word in WR10. The



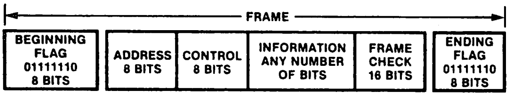
07666A 2-5

Figure 2-5 Bisynchronous Message Format



07666A 2-6

Figure 2-6 External Sync Format


FRAME
BEGINNING FLAG 01111110 8 BITS
ADDRESS 8 BITS
CONTROL 8 BITS
INFORMATION ANY NUMBER OF BITS
FRAME CHECK 16 BITS
ENDING FLAG 01111110 8 BITS

07666A 2-7

Figure 2-7 SDLC Message Format

SCC is now waiting for the EOP so that it can go on loop. While waiting for the EOP, the SCC ties TxD to RxD with only the internal gate delays in the signal path. When the first EOP is recognized by the SCC, the Break/Abort/EOP bit is set in RR0, generating an External/Status interrupt (if so enabled). At the same time, the On-Loop bit in RR10 is set to indicate that the SCC is indeed on-loop, and a one-bit time delay is inserted in the TxD to the RxD patch.

The SCC is now on-loop but cannot transmit a message until a flag and the next EOP are received. The requirement that a flag be received ensures that the SCC cannot erroneously send messages until the controller ends the current polling sequence and starts another one.

A secondary station on the loop is prohibited from transmitting a message during a polling sequence unless it captures the line at the moment the EOP passes by. The SCC does this automatically. If the CPU in the secondary station with SCC needs to transmit a message, the Go-Active-On-Poll bit in WR10 must be set. If this bit is set when the EOP is detected, the SCC changes the EOP to a flag and starts sending another flag. The EOP is reported in the Break/Abort/EOP bit in RR0 and the CPU should write its data bytes to the SCC, just as in normal SDLC frame transmission. When the frame is complete and CRC has been sent, the SCC closes with a flag and reverts to One-Bit-Delay mode. The last zero of the flag, along with the marking line echoed from the RxD pin, form an EOP for secondary stations further down the loop. If the Go-Active-On-Poll bit is not set at the time the EOP passes by, the SCC cannot send a message until a flag (terminating the current polling sequence) and another EOP are received. While the SCC is actually transmitting a message, the loop-sending bit in R10 is set to indicate this.

If SDLC loop is de-selected, the SCC is designed to exit from the loop gracefully. When SDLC Loop mode is de-selected by writing to WR10, the SCC waits until the next polling cycle to remove the on-bit time delay. If a polling cycle is in progress at the time the command is written, the SCC finishes sending any message that it may be transmitting, ends with an EOP, and disconnects TxD from RxD. If no message was in progress, the SCC immediately disconnects TxD from RxD. To ensure proper loop operation after the SCC goes off the loop, and until the external relays take the SCC completely out of the loop, the SCC should be programmed for Mark idle instead of Flag idle. When the SCC goes off the loop, the On-Loop bit is reset.

The SCC allows the user the option of using NRZI in SDLC Loop mode by programming WR20 appropriately. With NRZI encoding, the outputs of secondary stations in the loop may be inverted from their inputs because of messages that they have transmitted. Removing the stations from the loop (removing the one-bit time delay ) may cause problems further down the loop because of extraneous transitions on the line. The SCC avoids this problem by making transparent adjustments at the end of each frame it sends in response to an EOP. A response frame from the SCC is terminated by a flag and an EOP. Normally, the flag and the EOP share a zero, but if such sharing would cause the RxD and TxD pins to be of opposite polarity after the EOP, the SCC adds another zero between the flag and the EOP. This causes an extra line transition so that RxD and TxD are identical after the EOP is sent. This extra zero is completely transparent because it only means that the flag and the EOP no longer share a zero. All that a proper loop exit needs, therefore, is the removal of the one-bit time delay.

## 2.4 I/O CAPABILITIES

The SCC can work with three basic forms of I/O operations: polling, interrupts, and block transfer. All three I/O types involve register manipulation durng initialization and data transfer. However, the Interrupt mode also incorporates Z-BUS® interrupt protocol for a faster and more efficient data transfer.

### 2.4.1 Polling

During a polling sequence, the status of Read Register 0 is examined in each channel. This register indicates whether or not a receive or transmit data transfer is needed and whether or not any special conditions are present, e.g., errors.

This method of I/O transfer avoids interrupts. All interrupt functions must be disabled in order to operate the device in a polled environment. With no interrupts enabled, this mode of operation must initiate a read cycle of Read Register 0 to detect an incoming character before jumping to a data handler routine.

### 2.4.2 Interrupts

The SCC provides interrupt capability through the use of pins and a hardware scheme that enhances the maximum speed of serial data. Whenever the interrupt (INT) pin is active, the SCC is ready to transfer data.

Read and write registers are programmed so that an interrupt vector points to an interrupt service routine. The interrupt vector can also be modified to reflect various status conditions. Therefore, as many as eight different interrupt routines can be referenced.

Transmit interrupts, receive interrupts, and external/status interrupts are the main sources of interrupts. Each interrupt source is enabled under program control, with channel A having a higher priority than channel B and with receive, transmit, and external/status interrupts prioritized respectively within each channel. (Section 4 provides a detailed description of the interrupt scheme and the various interrupt types.)

### 2.4.3 Block Transfers

The SCC provides a Block Transfer mode to accommodate CPU block transfer functions and DMA controllers. The Block Transfer mode uses the W/REQ output in conjunction with the Wait/Request bits in Write Register 1. The W/REQ output can be defined by software as a WAIT line in the CPU Block Transfer mode or as a REQUEST line in the DMA Block Transfer mode.

To a DMA controller, the SCC REQUEST output indicates that the SCC is ready to transfer data to or from memory. To the CPU, the WAIT output indicates that the SCC is not ready to transfer data, thereby requesting the CPU to extend the I/O cycle. (Section 3.5 describes the registers used in block transfers.)

CHAPTER 3

# INTERFACING THE SCC

## 3.0 INTRODUCTION

This chapter covers the details of interfacing the Z8030 and Z8530 to a system. The general timing requirements for both devices are described but the respective data sheets must be referred to for specific A.C. numbers.

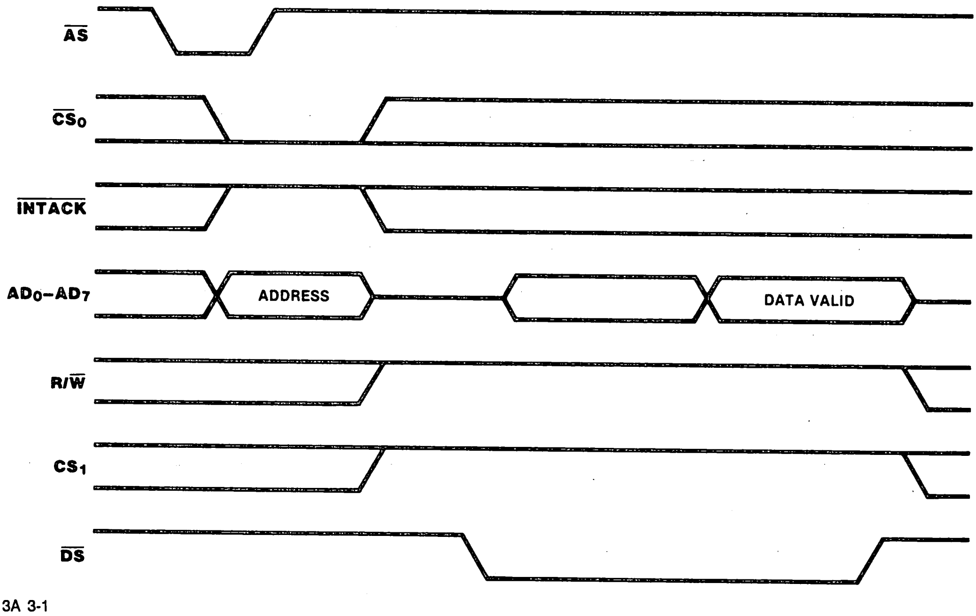
## 3.1 INTERFACING THE Z8030

The Z-BUS compatible SCC is suited for system applications with multiplexed address/data buses similar to the Z800 or Z8. The SCC complies with Z-BUS protocol established in the *Z-BUS Component Interconnect Summary*, in the Zilog Data Book, #00-2034-04.

Two control signals, $\overline{\text{AS}}$ and $\overline{\text{DS}}$, are used by the Z8030 to time bus transactions. In addition, four other control signals ($\overline{\text{CS1}}$, CS1, R/$\overline{\text{W}}$, and $\overline{\text{INTACK}}$) are used to control the type of bus transaction that will occur. A bus transaction is initiated by $\overline{\text{AS}}$, the rising edge of which latches the register address on the Address/Data bus and the state of $\overline{\text{INTACK}}$ and $\overline{\text{CS}}$. In addition to timing bus transactions, $\overline{\text{AS}}$ is used by the interrupt section to set the Interrupt Pending (IP) bits. Because of this, $\overline{\text{AS}}$ must be kept cycling for the interrupt section to function properly. The Z8030 generates internal control signals in response to a register access. Since $\overline{\text{AS}}$ and $\overline{\text{DS}}$ have no phase relationship with PCLK, the circuitry generating these internal control signals provide time for metastable conditions to disappear. This results in a recovery time related to PCLK. This recovery time applies only to transactions involving the Z8030, and any intervening transactions are ignored. This recovery time is four PCLK cycles, measured from the falling edge of $\overline{\text{DS}}$ in the case of a read or write of any register other than RR8 or WR8. In the case of a read of RR8 or a write to WR8, the recovery time is measured from the rising edge of $\overline{\text{DS}}$.

### 3.1.1 Z8030 Read Cycle Timing

The Read cycle timing for the Z8030 is shown in Figure 3-1. The register address on AD0–AD7



07513A 3-1

Figure 3-1 Z8030 Read Cycle Timing

as well as the state of CS0 and INTACK, are latched by the rising edge of AS. R/W must be HIGH before DS falls to indicate a Read cycle. The Z8030 data bus drivers are enabled while CS1 is HIGH and DS is LOW.

### 3.1.2 Z8030 Write Cycle Timing

The Write cycle timing for the Z8030 is shown in Figure 3-2. The register address on AD0–AD7, as well as the state if CS0 and INTACK are latched by the rising edge of AS. R/W must be LOW when DS falls to indicate a Write cycle. The leading edge of the coincidence of CS1 HIGH and DS LOW latches the write data on AD0–AD7 as well as the state of R/W.

### 3.1.3 Z8030 Interrupt Acknowledge Cycle Timing

The Interrupt Acknowledge cycle timing for the Z8030 is shown in Figure 3.3. The address on AD0–AD7 and the state of CS0 and INTACK are latched by the rising edge of AS. However, if INTACK is LOW, the address, CS0, CS1 and R/W are ignored for the duration of the interrupt acknowledge cycle. between the rising edge of AS and the falling edge of DS, the internal and external daisy chains settle (this is A.C. parameter #29 TdAS(DSA) in the data sheet. If there is an interrupt pending in the SCC, and IEI is HIGH when DS falls, the acknowledge cycle was intended for the SCC. This being the case, the Z8030 sets the appropriate Interrupt-Under-Service latch, as well as placing an interrupt vector on AD0–AD7. The INT pin also goes inactive in response of the falling edge of DS. Note that there should be only one DS per acknowledge cycle. Another important fact is that the IP bits in the Z8030 are up-dated by AS, which may delay interrupt requests if the processor does not supply AS strobes during the time between accesses of the Z8030.

### 3.1.4 Z8030 Register Access

The registers in the Z8030 are addressed via the address on AD0–AD7, latched by the rising edge of AS. The Shift Right/Shift Left bit in WR0B controls which bits will be decoded to form the register address. It is placed in this register to simplify programming when the current state of the Shift Right/Shift Left bit is not known. A hardware reset forces Shift Left mode, where the address is decoded from AD5–AD1. In Shift Right mode, the address is decoded from AD4–AD0. The Shift Right/Shift Left bit is written via command to make the software writing to WR0 independent of the state of the Shift Right/Shift Left bit. While in Shift Left mode, the register address itself is placed on AD4–AD1 and the Channel Select bit, A/B, is decoded from AD5. The register map for this case is shown in Table 3-1. In Shift Right mode, the

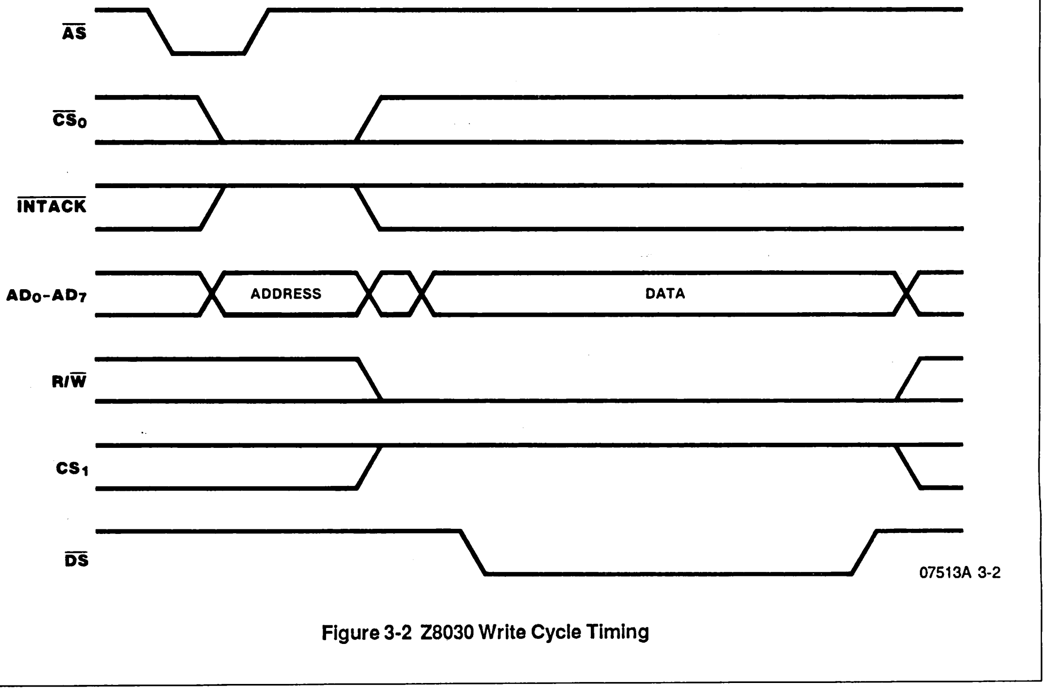


Figure 3-2 Z8030 Write Cycle Timing

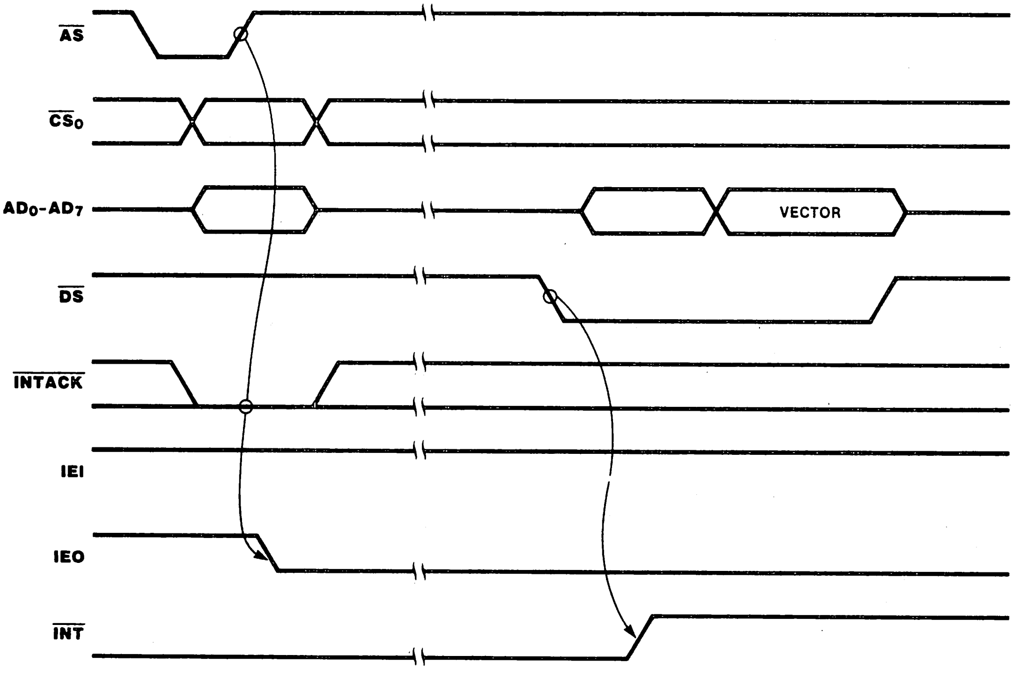
register address is again placed on AD4–AD1 but the channel select A/$\overline{B}$ is decoded from AD0. The register map for this case is shown in Table 3-2. Because the Z8030 does not contain 16 read registers, the decoding of the read registers is not complete; this is indicated in Table 3-1 and Table 3-2 by parentheses around the register name. These addresses may also be used to access the read registers. Note also that the Z8030 contains only one WR2 and WR9; these registers may be written from either channel. Shift Left Mode is used when channel A and B are to be programmed differently. This allows the software to sequence through the registers of one channel. The Shift Right Mode is used when the channels are programmed the same. By incrementing the address, the programmer can program the same data value into Channel A's register and Channel B's register.

### 3.1.5 Z8030 Reset

The Z8030 may be reset by either hardware of software. Hardware reset occurs when $\overline{AS}$ and $\overline{DS}$ are both LOW at the same time, which is normally an illegal condition. As long as both $\overline{AS}$ and $\overline{DS}$ are LOW, the Z8030 recognizes the reset condition. However, once this condition is removed, the reset condition is asserted internally for an additional four to five PCLK cycles. During this time any attempt to access the Z8030 will be ignored. The Z8030 has three software resets, encoded into two command bits in WR9. There are two channel resets, which only affect one channel in the device and some of the bits in the write registers. The third command forces the same result as does a hardware reset. As in the case of the hardware reset, the Z8030 stretches the reset signal an additional four to five PCLK cycles beyond the ordinary valid access recovery time. The bits in WR9 may be written at the same time as the reset command because these bits are affected only by a hardware reset. The reset values of the various registers are shown in Figure 3-4.

### 3.2 INTERFACING THE Z8530

Two control signals, $\overline{RD}$ and $\overline{WR}$, are used by the Z8530 to time bus transactions. In addition, four other control signals, $\overline{CE}$, D/$\overline{C}$, A/$\overline{B}$ and $\overline{INTACK}$, are used to control the type of bus transaction that will occur. A bus transaction starts when the addresses on D/$\overline{C}$ and A/$\overline{B}$ are asserted before $\overline{RD}$



07513A 3-3

Figure 3-3 Z8030 Interrupt Acknowledge Details

**Table 3-1 Z8030 Register Map (Shift Left)**

| AD$_5$ | AD$_4$ | AD$_3$ | AD$_2$ | AD$_1$ | WRITE | READ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | WR0B | RR0B |
| 0 | 0 | 0 | 0 | 1 | WR1B | RR1B |
| 0 | 0 | 0 | 1 | 0 | WR2 | RR2B |
| 0 | 0 | 0 | 1 | 1 | WR3B | RR3B |
| 0 | 0 | 1 | 0 | 0 | WR4B | (RR0B) |
| 0 | 0 | 1 | 0 | 1 | WR5B | (RR1B) |
| 0 | 0 | 1 | 1 | 0 | WR6B | (RR2B) |
| 0 | 0 | 1 | 1 | 1 | WR7B | (RR3B) |
| 0 | 1 | 0 | 0 | 0 | WR8B | RR8B |
| 0 | 1 | 0 | 0 | 1 | WR9 | (RR13B) |
| 0 | 1 | 0 | 1 | 0 | WR10B | RR10B |
| 0 | 1 | 0 | 1 | 1 | WR11B | (RR15B) |
| 0 | 1 | 1 | 0 | 0 | WR12B | RR12B |
| 0 | 1 | 1 | 0 | 1 | WR13B | RR13B |
| 0 | 1 | 1 | 1 | 0 | WR14B | (RR10B) |
| 0 | 1 | 1 | 1 | 1 | WR15B | RR15B |
| 1 | 0 | 0 | 0 | 0 | WR0A | RR0A |
| 1 | 0 | 0 | 0 | 1 | WR1A | RR1A |
| 1 | 0 | 0 | 1 | 0 | WR2 | RR2A |
| 1 | 0 | 0 | 1 | 1 | WR3A | RR3A |
| 1 | 0 | 1 | 0 | 0 | WR4A | (RR0A) |
| 1 | 0 | 1 | 0 | 1 | WR5A | (RR1A) |
| 1 | 0 | 1 | 1 | 0 | WR6A | (RR2A) |
| 1 | 0 | 1 | 1 | 1 | WR7A | (RR3A) |
| 1 | 1 | 0 | 0 | 0 | WR8A | RR8A |
| 1 | 1 | 0 | 0 | 1 | WR9 | (RR13A) |
| 1 | 1 | 0 | 1 | 0 | WR10A | RR10A |
| 1 | 1 | 0 | 1 | 1 | WR11A | (RR15A) |
| 1 | 1 | 1 | 0 | 0 | WR12A | RR12A |
| 1 | 1 | 1 | 0 | 1 | WR13A | RR13A |
| 1 | 1 | 1 | 1 | 0 | WR14A | (RR10A) |
| 1 | 1 | 1 | 1 | 1 | WR15A | RR15A |

**Table 3-2 Z8030 Register Map (Shift Right)**

| AD$_5$ | AD$_4$ | AD$_3$ | AD$_2$ | AD$_1$ | WRITE | READ |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | WR0B | RR0B |
| 0 | 0 | 0 | 0 | 1 | WR0A | RR0A |
| 0 | 0 | 0 | 1 | 0 | WR1B | RR1B |
| 0 | 0 | 0 | 1 | 1 | WR1A | RR1A |
| 0 | 0 | 1 | 0 | 0 | WR2 | RR2B |
| 0 | 0 | 1 | 0 | 1 | WR2 | RR2A |
| 0 | 0 | 1 | 1 | 0 | WR3B | RR3B |
| 0 | 0 | 1 | 1 | 1 | WR3A | RR3A |
| 0 | 1 | 0 | 0 | 0 | WR4B | RR0B |
| 0 | 1 | 0 | 0 | 1 | WR4A | RR0A |
| 0 | 1 | 0 | 1 | 0 | WR5B | (RR1B) |
| 0 | 1 | 0 | 1 | 1 | WR5A | (RR1A) |
| 0 | 1 | 1 | 0 | 0 | WR6B | RR2B |
| 0 | 1 | 1 | 0 | 1 | WR6A | RR2A |
| 0 | 1 | 1 | 1 | 0 | WR7B | (RR3B) |
| 0 | 1 | 1 | 1 | 1 | WR7A | (RR3A) |
| 1 | 0 | 0 | 0 | 0 | WR8B | RR8B |
| 1 | 0 | 0 | 0 | 1 | WR8A | RR8A |
| 1 | 0 | 0 | 1 | 0 | WR9 | (RR13B) |
| 1 | 0 | 0 | 1 | 1 | WR9 | (RR13A) |
| 1 | 0 | 1 | 0 | 0 | WR10B | RR10B |
| 1 | 0 | 1 | 0 | 1 | WR10A | RR10A |
| 1 | 0 | 1 | 1 | 0 | WR11B | (RR15B) |
| 1 | 0 | 1 | 1 | 1 | WR11A | (RR15A) |
| 1 | 1 | 0 | 0 | 0 | WR12B | RR12B |
| 1 | 1 | 0 | 0 | 1 | WR12A | RR12A |
| 1 | 1 | 0 | 1 | 0 | WR13B | RR13B |
| 1 | 1 | 0 | 1 | 1 | WR13A | RR13A |
| 1 | 1 | 1 | 0 | 0 | WR14B | (RR10B) |
| 1 | 1 | 1 | 0 | 1 | WR14A | (RR10A) |
| 1 | 1 | 1 | 1 | 0 | WR15B | RR15B |
| 1 | 1 | 1 | 1 | 1 | WR15A | RR15A |

or $\overline{WR}$ fall. The coincidence of $\overline{CE}$ and $\overline{RD}$ or $\overline{CE}$ and $\overline{WR}$ latches the state of D/$\overline{C}$ and A/$\overline{B}$ and starts the internal operation. The $\overline{INTACK}$ signal must have been previously sampled High by a rising edge of PCLK for a read or write cycle to occur. In addition to sampling $\overline{INTACK}$, PCLK is used by the interrupt section to set the IP bits. The Z8530 generates internal control signals in response to a register access. Since $\overline{RD}$ and $\overline{WR}$ have no phase relationship with PCLK, the circuitry generating these internal control signals provides time for metastable conditions to disappear. This results in a recovery time related to PCLK. This recovery time applies only between transactions involving the Z8530, and any intervening transactions are ignored. This recovery time is four PCLK cycles, measured from the falling edge of $\overline{RD}$ or $\overline{WR}$ in the case of a read or write of any register.

### 3.2.1 Z8530 Read Cycle Timing

The Read cycle timing for the Z8530 is shown in Figure 3-5. The address on A/$\overline{B}$ and D/$\overline{C}$ is latched by the coincidence of $\overline{RD}$ and $\overline{CE}$ active. $\overline{CE}$ must remain LOW and $\overline{INTACK}$ must remain HIGH throughout the cycle. The Z8530 bus drivers are enabled while $\overline{CE}$ and $\overline{RD}$ are both LOW. A read with D/$\overline{C}$ HIGH does not disturb the state of the pointers and a read cycle with D/$\overline{C}$ LOW resets the pointers to zero after the internal operation is complete.

### 3.2.2 Z8530 Write Cycle Timing

The Write cycle timing for the Z8530 is shown in Figure 3-6. The address on A/$\overline{B}$ and D/$\overline{C}$, as well as the data on D0–D7, is latched by the coincidence of $\overline{WR}$ and $\overline{CE}$ active. $\overline{CE}$ must remain LOW and $\overline{INTACK}$ must remain HIGH throughout the cycle. A write cycle with D/$\overline{C}$ HIGH does not disturb the state of the pointers and a write cycle with D/$\overline{C}$ LOW resets the pointers to zero after the internal operation is complete.

### 3.2.3 Z8530 Interrupt Acknowledge Cycle Timing

The Interrupt Acknowledge cycle timing for the Z8530 is shown in Figure 3.7. The state of INTACK is latched by the rising edge of PCLK. While INTACK is LOW, the state of A/B, CE, D/C, and WR are ignored. Between the time INTACK is first sampled LOW and the time RD falls, the internal and external IEI/IEO daisy chains settle; this is A.C. parameter #38 TdIAi (RD). If there is an interrupt pending in the Z8530, and IEI is HIGH when RD falls, the Interrupt Acknowledge cycle was intended for the Z8530. This being the case, the Z8530 sets the appropriate Interrupt Under Service latch, and places an interrupt vector on D0–D7. If the falling edge of RD sets an IUS bit in the Z8530, the INT pin goes active in response to the falling edge. Note that there should be only one RD per Acknowledge cycle. Another important fact is that the IP bits in the Z8530 are updated by PCLK divided by two, and this clock to update IPs is stopped while the pointers point to RR2 and RR3. This prevents data changing during a read, but will delay interrupt requests if the pointers are left pointing at these registers.

### 3.2.4 Z8530 Register Access

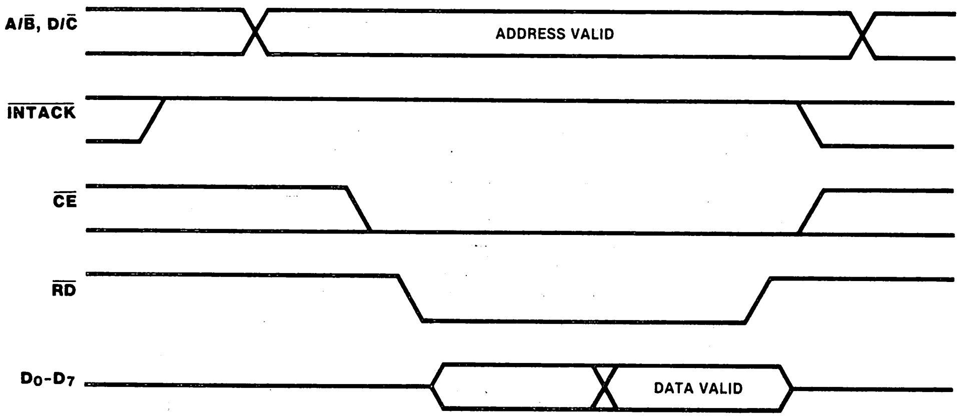
The registers in the Z8530 are accessed in a two-step process, using a Register Pointer to perform

| HARDWARE RESET | | | | | | | | CHANNEL RESET | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | WR0 |
| 0 | 0 | . | 0 | 0 | . | 0 | 0 | 0 | 0 | . | 0 | 0 | . | 0 | 0 | WR1 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR2 |
| . | . | . | . | . | . | . | 0 | . | . | . | . | . | . | . | 0 | WR3 |
| . | . | . | . | . | 1 | . | . | . | . | . | . | . | 1 | . | . | WR4 |
| 0 | . | . | 0 | 0 | 0 | 0 | . | 0 | . | . | 0 | 0 | 0 | 0 | . | WR5 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR6 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR7 |
| 1 | 1 | 0 | 0 | 0 | 0 | . | . | . | . | 0 | . | . | . | . | . | WR9 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | . | . | 0 | 0 | 0 | 0 | 0 | WR10 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | . | . | . | . | . | . | . | . | WR11 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR12 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR13 |
| . | . | 1 | 0 | 0 | 0 | 0 | 0 | . | . | 1 | 0 | 0 | 0 | . | . | WR14 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | WR15 |
| 0 | 1 | . | . | . | 1 | 0 | 0 | 0 | 1 | . | . | . | 1 | 0 | 0 | RR0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | RR1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RR3 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RR10 |

07513A 3-4

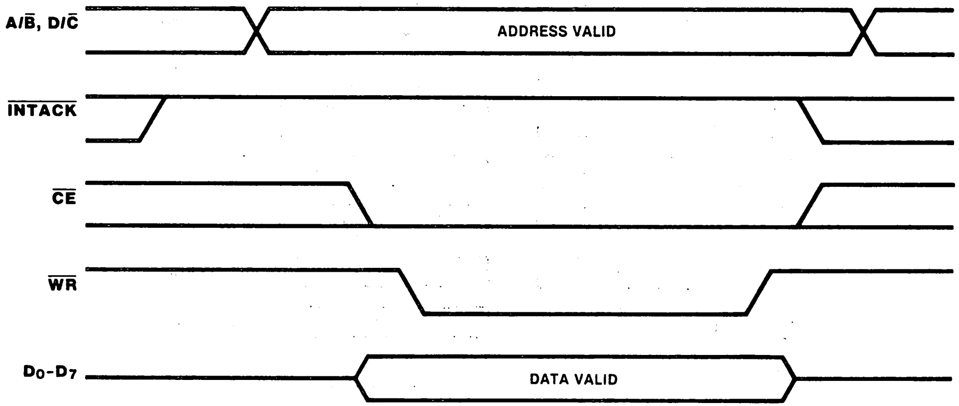
**Figure 3-4 Z8030 Register Reset Values**

the addressing. To access a particular register, the pointer bits must be set by writing to WR0. The pointer bits may be written in either channel because only one set exists in the Z8530. After the pointer bits are set, the next read or write cycle of the Z8530 having D/$\overline{C}$ LOW will access the desired register. At the conclusion of this read or write cycle the pointer bits are reset to "0s," so that the next control write will be to the pointers in WR0. A read or RR8 (the receive data buffer) or a write to WR8 (the transmit data buffer) may either be done in this fashion or by accessing the Z8530 having D/$\overline{C}$ pin HIGH. A read or write with D/$\overline{C}$ HIGH accesses the data registers directly, and independently, of the state of the pointer bits. This allows single-cycle access to the data registers and does not disturb the pointer bits. The fact that the pointer bits are reset to "0," unless explicitly set otherwise, means that WR0 and RR0 may also be accessed in a single cycle. That is, it is not necessary to write the pointer bits with "0" before accessing WR0 or RR0. There are three pointer bits in WR0, and these allow access to the registers with addresses 0 through 7. Note that a command may be written to WR0 at the same time that the pointer bits are written. To access the registers with addresses 8 through 15, a special command must accompany the pointer bits. This



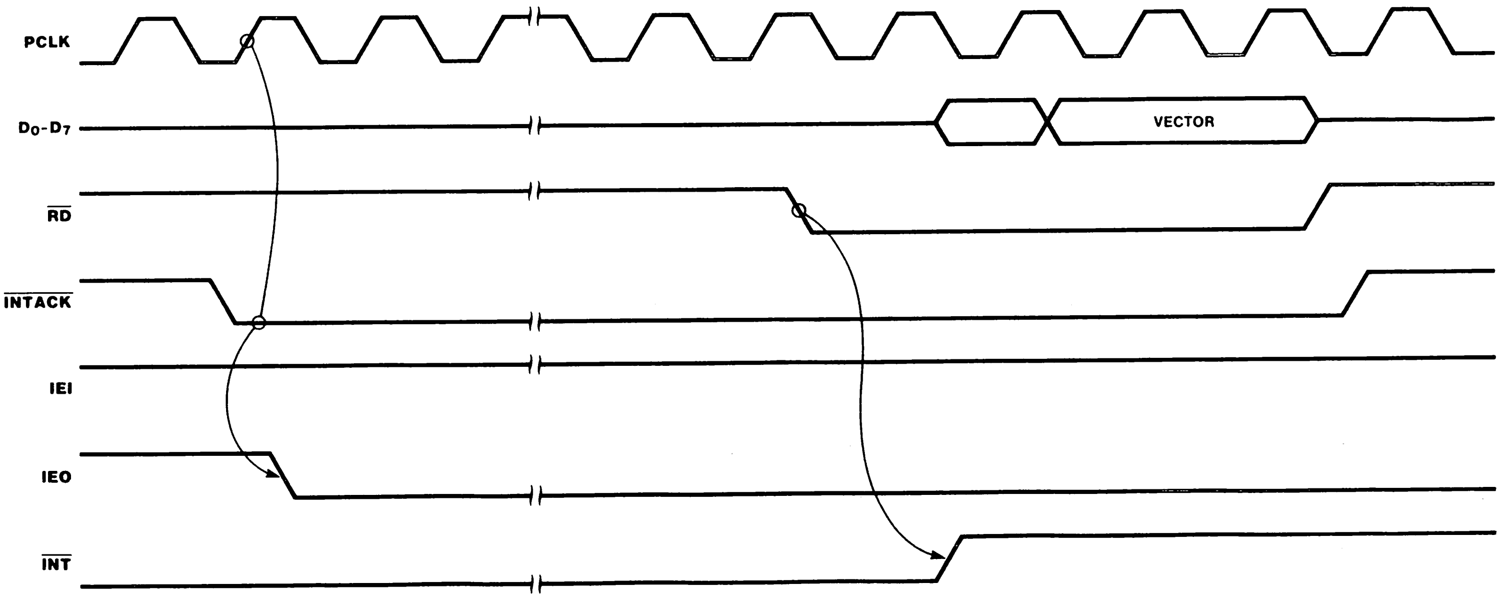
07513A 3-5

Figure 3-5 Z8530 Read Cycle Timing



07513A 3-6

Figure 3-6 Z8530 Write Cycle Timing



07513A 3-7

Figure 3-7 Z8530 Interrupt Acknowledge Details

precludes concurrently issuing a command when pointing to these registers. The register map for the AmZ8530 is shown in Table 3.3. If, for some reason, the state of the pointer bits is unknown they may be reset to "0" by performing a read cycle with the D/$\overline{C}$ pin held LOW. Once the pointer bits have been set, the desired channel is selected by the state of the A/$\overline{B}$ pin during the actual read or write of the desired register.

### 3.2.5 Z8530 Reset

The Z8530 may be reset by either hardware or software. Hardware reset occurs when $\overline{RD}$ and $\overline{WR}$ are both LOW, simultaneously, which is normally an illegal condition. As long as both $\overline{RD}$ and $\overline{WR}$ are LOW, the Z8530 recognizes the reset condition. Once this condition is removed, however, the reset condition is asserted internally for an additional four to five PCLK cycles. During this time any attempt to access the Z8530 will be ignored. The Z8530 has three software resets, encoded into command bits in WR9. There are two channel resets, which affect only one channel in the device and some of the bits in the write registers. The third command forces the same result as does a hardware reset. As in the case of a hardware reset, the Z8530 stretches the reset signal an additional four to five PCLK cycles beyond the ordinary valid access recovery time. The bits in WR9 may be written at the same time as the reset command because these bits are affected only by a hardware reset. The reset values of the various registers are shown in Figure 3-8.

**Table 3-3 Z8530 Register Map**

| A/B | $PNT_2$ | $PNT_1$ | $PNT_0$ | WRITE | READ |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | WR0B | RR0B |
| 0 | 0 | 0 | 1 | WR1B | RRIB |
| 0 | 0 | 1 | 0 | WR2 | RR2B |
| 0 | 0 | 1 | 1 | WR3B | RR3B |
| 0 | 1 | 0 | 0 | WR4B | (RR0B) |
| 0 | 1 | 0 | 1 | WR5B | (RR1B) |
| 0 | 1 | 1 | 0 | WR6B | (RR2B) |
| 0 | 1 | 1 | 1 | WR7B | (RR3B) |
| 1 | 0 | 0 | 0 | WR0A | RR0A |
| 1 | 0 | 0 | 1 | WR1A | RR1A |
| 1 | 0 | 1 | 0 | WR2 | RR2A |
| 1 | 0 | 1 | 1 | WR3A | RR3A |
| 1 | 1 | 0 | 0 | WR4A | (RR0A) |
| 1 | 1 | 0 | 1 | WR5A | (RR1A) |
| 1 | 1 | 1 | 0 | WR6A | (RR2A) |
| 1 | 1 | 1 | 1 | WR7A | (RR3A) |
| With the Point High command: | | | | | |
| 0 | 0 | 0 | 0 | WR8B | RR8B |
| 0 | 0 | 0 | 1 | WR9 | RRI3B |
| 0 | 0 | 1 | 0 | WR10B | RR10B |
| 0 | 0 | 1 | 1 | WR11B | (RR15B) |
| 0 | 1 | 0 | 0 | WR12B | RR12B |
| 0 | 1 | 0 | 1 | WR13B | RR13B |
| 0 | 1 | 1 | 0 | WR14B | (RR10B) |
| 0 | 1 | 1 | 1 | WR15B | RR15B |
| 1 | 0 | 0 | 0 | WR8A | RR8A |
| 1 | 0 | 0 | 1 | WR9 | (RR13A) |
| 1 | 0 | 1 | 0 | WR10A | RR10A |
| 1 | 0 | 1 | 1 | WR11A | (RR15A) |
| 1 | 1 | 0 | 0 | WR12A | RR12A |
| 1 | 1 | 0 | 1 | WR13A | RR13A |
| 1 | 1 | 1 | 0 | WR14A | RR14A |
| 1 | 1 | 1 | 1 | WR15A | RR15A |

| HARDWARE RESET | | | | | | | | CHANNEL RESET | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | WR0 |
| 0 | 0 | . | 0 | 0 | . | 0 | 0 | 0 | 0 | . | 0 | 0 | . | 0 | 0 | WR1 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR2 |
| . | . | . | . | . | . | . | 0 | . | . | . | . | . | . | . | 0 | WR3 |
| . | . | . | . | . | 1 | . | . | . | . | . | . | . | 1 | . | . | WR4 |
| 0 | . | . | 0 | 0 | 0 | 0 | . | 0 | . | . | 0 | 0 | 0 | 0 | . | WR5 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR6 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR7 |
| 1 | 1 | 0 | 0 | 0 | 0 | . | . | . | . | 0 | . | . | . | . | . | WR9 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | . | . | 0 | 0 | 0 | 0 | 0 | WR10 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | . | . | . | . | . | . | . | . | WR11 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR12 |
| . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | . | WR13 |
| . | . | 1 | 0 | 0 | 0 | 0 | 0 | . | . | 1 | 0 | 0 | 0 | . | . | WR14 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | WR15 |
| 0 | 1 | . | . | . | 1 | 0 | 0 | 0 | 1 | . | . | . | 1 | 0 | 0 | RR0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | RR1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RR3 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | RR10 |

07513A 3-8

**Figure 3-8 Z8530 Register Reset Values**

CHAPTER 4

# I/O PROGRAMMING CAPABILITIES

## 4.0 INTRODUCTION

Regardless of the version of the SCC, all communication modes can use a choice of polling, interrupt and block transfer. These modes must be selected by the user to select the proper hardware and software required to supply data at the rate required.

## 4.1 POLLING

This is the simpliest mode to implement. The software must poll the SCC to determine when data is to be inputted or outputted from the SCC. In this mode, MIE (WR9 bit 3), and Wait/DMA Request Enable (WR1 bit7) are both reset to 0 to disable any requests. The software must then poll RR0 to determine the status of the receive buffer, transmit buffer and external status.

## 4.2 INTERRUPT OPERATIONS

The SCC, as a microprocessor peripheral, will request an interrupt only when it needs servicing. This allows the CPU to perform other operations while the SCC does not need service. The SCC has an internal priority resolution method to allow the highest priortiy interrupt to be serviced first.

The SCC is flexible with its interrupt method. The interrupt may be acknowledged with a vector transferred, acknowledged without a vector, or not acknowledged at all.

### Interrupt Without Acknowledge

In this mode, the Interrupt Acknowledge signal does not have to be generated. This allows a simpler hardware design that does not have to meet the Interrupt acknowledge timing. Soon after the $\overline{\text{INT}}$ goes active, the interrupt controller will jump to the interrupt routine. In the interrupt routine, the code must read RR2 from Channel B to read the vector including status. When the vector is read from Channel B, it always includes the status regardless of the VIS bit (WR9 bit 0). The status given will decode the highest priority interrupt pending at the time it is read. The vector is not latched so that the next read could produce a different vector if another interrupt occurs. The register is disabled from change during the read operation to prevent an error if a higher interrupt occurs exactly during the read operation.

Once the status is read, the interrupt routine must decode the interrupt pending, and clear the condition. Removing the interrupt condition will clear the IP and bring $\overline{\text{INT}}$ inactive, as long as there are no other IP bits set. For example, writing a character to the transmit buffer will clear the transmit buffer empty IP.

When the interrupt IP, decoded from the status, is cleared RR2 can be read again. This allows the interrupt routine to clear all of the IP's within one interrupt request to the CPU.

### Interrupt with Acknowledge

After the SCC brings $\overline{\text{INT}}$ active, the CPU must respond by bringing $\overline{\text{INTACK}}$ active. After enough time has elapsed to allow the daisy-chain to settle, the SCC will set the IUS bit for the highest priority IP. If the No Vector bit is not set (WR9 bit 1), the SCC will then place the interrupt vector on the data bus during a read. To speed the intrrupt response time, the SCC can also modify 3 bits in the vector to inidicate status. To include the status, the VIS bit (WR9 bit 0) must be set. The service routine must then clear the interrupting condition. For example, writing a character to the transmit buffer will clear the transmit buffer empty IP. After the interrupting condition is cleared, the routine can read RR3 to determine if any other IP's are set and clear them. At the end of the interrupt routine, a Reset IUS command (WR0) must be issued to unlock the daisy-chain and allow lower-priority interrupt requestes. This is the only way, short of a software or hardware reset, that an IUS bit may be reset.

If the No Vector bit (WR9 bit 1) is set, the SCC will not place the vector on the data bus. An interrupt controller must then vector the code to the interrupt routine. The interrupt routine must then read RR2 from Channel B to read the status. This is the same as the case of an interrupt without an acknowledge except the IUS is set and the vector will not change until the Reset IUS command in RR0 is issued.

### Interrupt Sources

Each channel in the SCC contains 3 sources of interrupt, making a total of 6. These 3 sources of interrupts are the receiver, the transmitter, and External/Status conditions. In addition, there are several conditions that may cause these interrupts.

The receive interrupt request may either be caused by a receive character available or a special condition. The receive character available interrupt is generated when a character is loaded into the FIFO and is ready to be read. The special conditions are receive FIFO overrun, CRC/framming error, End of frame, and parity. The parity special condition can be included as a special condition or not depending on bit 2 WR1. The special condition status can be read from RR1.

The transmit interrupt request has only one source. It can only be set when the transmit buffer goes from full to empty. Note that this means that the transmit interrupt will not be set until after the first character is written to the SCC.

The External/status Interrupts have several sources which may be individually enabled in WR15. The sources are zero count, DCD, Sync/Hunt, CTS, transmitter underrun/EOM and Break/Abort.

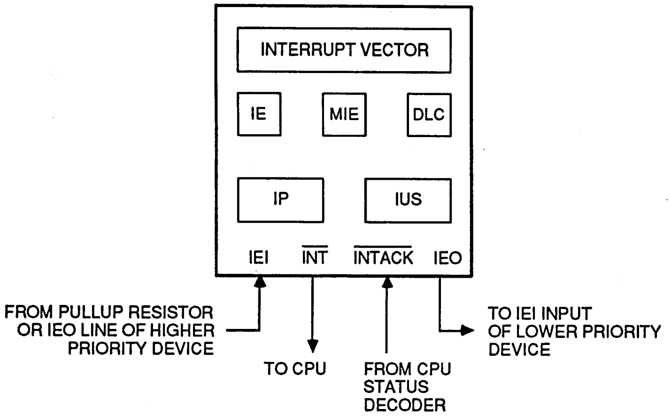
Each source of interrupt in the SCC has three control/status bits associated with it. There are Interrupt Enable (IE), Interrupt Pending (IP), and Interrupt Under Service (IUS) (See Figure 4.1). The IE bit is written by the processor and serves to control interrupt requests from the SCC. If the IE bit is set for a given source of interrupt, then that source may cause an interrupt request when all of the necessary conditions are met. If the IE bit is reset, no interrupt request will be generated by that source. The IE bits are write-only in the SCC. The IP bit for a given source of interrupt may be set by the presence of an interrupt condition in the SCC and is reset directly by the processor, or indirectly by some action that the processor may take. If the corresponding IE bit is not set, the IP for that source of interrupt will never be set. The IP bits in the SCC are read-only in RR3A. The IUS bits are completely hidden from the processor's view. An IUS is set during an Interrupt Acknowledge cycle for the highest-priority IP. See Table 4-1 for the interrupt priority. IUS is used to control the operation of the interrupt daisy chain (see Section 4.1) by masking lower-priority interrupts. At the end of an interrupt service routine, the processor must issue a Reset Highest IUS command in WR0 to allow lower-priority interrupts. This is the only way, short of a software or hardware reset, that an IUS bit may be reset.

**Table 4-1 Interrupt Source Priority**

| | |
|---|---|
| Receiver Channel A | High |
| Transmit Channel A | |
| External/Status Channel A | ↓ |
| Receiver Channel B | ↓ |
| Transmit Channel B | |
| External/Status Channel B | Low |

### Daisy-Chain Priority Resolution

The six sources of interrupt in the SCC are prioritized in a fixed order via a daisy chain; provision is made, via the IEI and IEO pins, for use of an external daisy chain as well. All Channel A



07513A 4-1

**Figure 4-1 Peripheral Interrupt Structure**

interrupts are higher-priority than any Channel B interrupts, with the receiver, transmitter, and External/Status interrupts prioritized in that order within each channel. The SCC requests an interrupt by pulling the $\overline{INT}$ pin Low from its open-drain state. This is controlled by the IP bits and the IEI input, among other things. A flowchart of the interrupt sequence for the SCC is shown in Figure 4-2. The internal daisy chain links the six sources of interrupt in a fixed order, chaining the IUS bits for each source. While an IUS is set, all lower-priority interrupt requests are masked off; during an Interrupt Acknowledge cycle the IP bits are also gated into the daisy chain. This insures that the highest-priority IP will be selected to have its IUS set. The internal daisy chain may be controlled by the MIE bit in WR9. This bit, when reset, has the same effect as pulling the IEI pin Low, thus disabling all interrupt requests.

### External Daisy Chain Operations

The SCC generates an interrupt request by pulling $\overline{INT}$ Low, but only if such interrupt requests are enabled (IE is 1, MIE is 1), an IP is set without a higher-priority IUS being set, or no higher-priority IUS being set, or no higher-priority interrupt is being serviced (IEI is High), or no Interrupt Acknowledge transaction is taking place. It is not pulled Low by the SCC at this time, but instead continues to follow IEI until an Interrupt Acknowledge transaction occurs. Some time after $\overline{INT}$ has been pulled Low, the processor initiates an Interrupt Acknowledge transaction. Between the time the the SCC recognizes that an Interrupt Acknowledge cycle is in progress and the time during the acknowledge that the processor requests an interrupt vector, the IEI/IEO daisy chain settles. Any peripheral in the daisy chain having an Interrupt Pending (IP is 1) or an Interrupt Under Service (IUS is 1) holds its IEO line Low and all others make IEO follow IEI.

When the processor requests an interrupt vector, only the highest-priority interrupt source with a pending interrupt (IP is 1) has its IEI input High, its IE bit set to "1," and its IUS bit set to "0." This is the interrupt source being acknowledged, and at this point it sets its IUS bit to "1." If its NV bit is "0," the SCC identifies itself by placing the interrupt vector from WR2 on the data bus. If the NV bit is "1," the SCC data bus remains floating, allowing external logic to supply a vector. If the VIS bit in the SCC is "1," the vector also contains status information, encoded as shown in Table 4-2, which further describes the nature of the SCC interrupt. If the VIS bit is "0," the vector held in WR2 is returned without modification. If the SCC is programmed to include status information in the vector, this status may be encoded and placed in either bits 1-3 or in bits 4-6. This operation is selected by programming the Status High/Status Low bit in WR9. At the end of the interrupt service routine, the processor should issue the Reset Highest IUS command to unlock the daisy chain and allow lower-priority interrupt requests. The IP is reset during the interrupt service routine either directly by command, or indirectly, through some action taken by the processor. The external daisy chain may be controlled by the DLC bit in WR9. This bit, when set, forces IEO Low, disabling all lower-priority devices.

**Table 4-2 Interrupt Vector Modification**

| V3 | V2 | V1 | Status High/Status Low = 0 |
|---|---|---|---|
| V4 | V5 | V6 | Status High/Status Low = 1 |
| 0 | 0 | 0 | Ch B Transmit Buffer Empty |
| 0 | 0 | 1 | Ch B External/Status Change |
| 0 | 1 | 0 | Ch B Receive Character Avail. |
| 0 | 1 | 1 | Ch B Special Receive Condition |
| 1 | 0 | 0 | Ch A Transmit Buffer Empty |
| 1 | 0 | 1 | Ch A External/Status Change |
| 1 | 1 | 0 | Ch A Receive Character Avail. |
| 1 | 1 | 1 | Ch A Special Receive Condition |

### 4.2.1 Receive Interrupts

The Receive Interrupt mode is controlled by WR1 bits 4 and 3. These select one of the four interrupt modes. The four modes are, Interrupt disabled, Interrupt on first character or special conditions, Interrupt on all characters or special conditions, and Interrupt on special conditions.

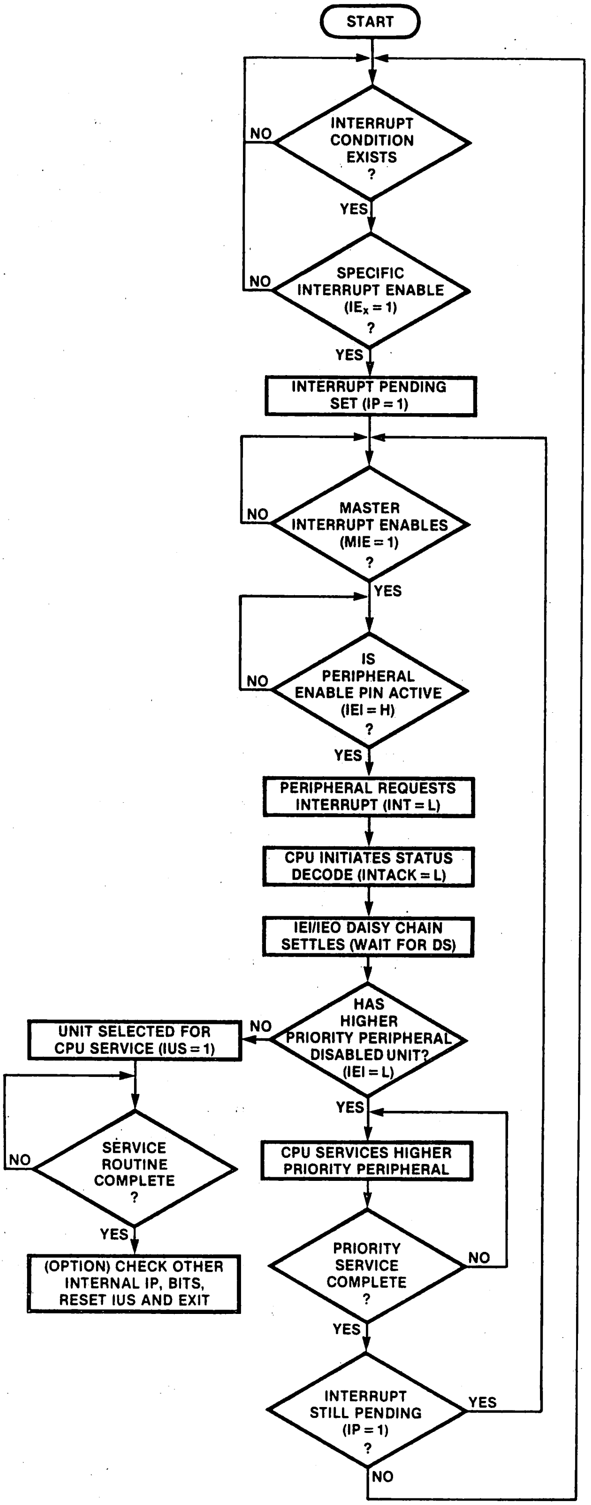
### Receive Interrupts Disabled

This mode prevents the receiver from requesting an interrupt. It is used in a polled environment where either the status bits in RR0 or the modified vector in RR2 (Channel B) is read. Although the receiver interrupts are disabled, the interrupt logic can still be used to provide status.

When these bits indicate that a received character has reached the top of the FIFO, the status in RR1 should be checked and then the data should be read. If status is to be checked, it must be done before the data is read, because the act of reading the data pops both the data and error FIFOs.

### Receive Interrupt on First Character or Special Condition

This mode is designed for use with DMA transfers of the receive characters. After this mode is


START
INTERRUPT CONDITION EXISTS ?
NO
YES
SPECIFIC INTERRUPT ENABLE ($IE_x = 1$) ?
NO
YES
INTERRUPT PENDING SET (IP = 1)
MASTER INTERRUPT ENABLES (MIE = 1) ?
NO
YES
IS PERIPHERAL ENABLE PIN ACTIVE (IEI = H) ?
NO
YES
PERIPHERAL REQUESTS INTERRUPT (INT = L)
CPU INITIATES STATUS DECODE (INTACK = L)
IEI/IEO DAISY CHAIN SETTLES (WAIT FOR DS)
HAS HIGHER PRIORITY PERIPHERAL DISABLED UNIT? (IEI = L)
NO
UNIT SELECTED FOR CPU SERVICE (IUS = 1)
YES
CPU SERVICES HIGHER PRIORITY PERIPHERAL
SERVICE ROUTINE COMPLETE ?
NO
YES
(OPTION) CHECK OTHER INTERNAL IP, BITS, RESET IUS AND EXIT
PRIORITY SERVICE COMPLETE ?
NO
YES
INTERRUPT STILL PENDING (IP = 1) ?
YES
NO

07513A 4-2

Figure 4-2 Interrupt Flowchart

selected, the first character received, or the first character already stored in the FIFO, will set the receiver IP. This IP will be reset when this character is removed from the SCC. No further receive interrupts will occur until the processor issues an Enable Interrupt on Next Receive Character command in WR0 or until a special receive condition occurs. The SCC recognizes several special receive conditions. A receive overrun (where a character in the FIFO is written over) is a special receive condition, as is a framing error in Asynchronous mode, or the end-of-frame condition in SDLC mode. In addition, if D2 of WR1 is set, any character with a parity error will generate a special receive condition interrupt. The correct sequence of events when using this mode is to first select the mode and wait for the receive character available interrupt. When the interrupt occurs the processor should read the character and then enable the DMA to transfer the remaining characters. A special receive condition interrupt may occur any time after the first character is received, but is guaranteed to occur after the character having the special condition has been read. The status is not lost in this case, however, because the FIFO is locked by the special condition. In the service routine the processor should read RR1 to obtain the status, and may read the data again if necessary. The FIFO is unlocked by issuing an Error Reset command in WR0. If the special condition was End-of-Frame, the processor should now issue the Enable Interrupt on Next Receive Character command to prepare for the next frame. The first character interrupt and special condition interrupt are distinguished by the status included in the interrupt vector. In all other respects they are identical, including sharing the IP and IUS bits.

**Interrupt on All Receive Characters or Special Conditions**

This mode is designed for an interrupt-driven system. In this mode the SCC will set the receiver IP on every received character, whether or not it has a special receive condition. This includes characters already in the FIFO when this mode is selected. In this mode of operation the IP is reset when the character is removed from the FIFO, so if the processor requires status for any character, this status must be read before the data is removed from the FIFO. The special receive conditions are identical to those previously mentioned, and as before, the only difference between a "receive character available" interrupt and a "special receive condition" interrupt is the status encoded in the vector. In this mode a special receive condition does not lock the receive data FIFO so that the service routine must read the status in RR1 before reading the data. At moderate to high data rates, where the interrupt overhead is significant, time can usually be saved by checking for another received character before exiting the service routine. This technique eliminates the Interrupt Acknowledge and the processor-state-saving time, but care must be exercised because this receive character must be checked for special receive conditions before it is removed from the SCC.

**Receive Interrupt on Special Conditions**

This mode is designed for use with DMA transfers of the receive characters. In this mode, only receive characters with special conditions will cause the receiver IP to be set. All other characters are assumed to be transferred via DMA. No special initialization sequence is needed in this mode. Usually the DMA is initialized and enabled, and then this mode is selected in the SCC. A special receive condition interrupt may occur at any time after this mode is selected but the logic guarantees that the interrupt will not occur until after the character with the special condition has been read from the SCC. The special condition locks the FIFO so that the status will be valid when read in the interrupt service routine, and it guarantees that the DMA will not transfer any characters until the special condition has been serviced. In the service routine the processor should read RR1 to obtain the status and unlock the FIFO by issuing an Error Reset command. DMA transfer of the receive characters will then resume.

### 4.2.2 Transmit Interrupts

Transmit interrupts are controlled by the Transmit Interrupt Enable bit (D1) in WR1. If the interrupt capabilities of the SCC are not required, polling may be used. This is selected by disabling the transmit interrupts and polling the Transmit Buffer Empty bit in RR0. When the Transmit Buffer Empty bits is set a character may be written to the SCC without fear of writing over previous data. Another way of polling the SCC is to enable the transmit interrupt and then reset the MIE bit in WR9. The processor may then poll the IP bits in RR3A to determine when the transmit buffer is empty. Transmit interrupts should also be disabled in the case of DMA transfer of the transmitted data.

While the transmit interrupts are enabled the SCC will set the transmit IP whenever the transmit buffer becomes empty. This means that the transmit buffer must have been full before the transmit IP can be set. Thus when the transmit interrupts are first enabled, the transmit IP will not be set until after the first character is written to the SCC. In synchronous modes one other condition can cause the transmit IP to be set. This occurs at the end of a transmission after CRC is sent. When the last bit of CRC has cleared the Transmit Shift register and the flag or sync character is loaded into the Transmit Shift register, the SCC will set the transmit IP. Data for the new frame or block to be transmitted may be written at this time. In this particular case the Transmit Buffer Empty bit in RR0 is not set; only the transmit IP is set. If the transmit Buffer Empty bit is, in fact, set for the transmit interrupt that occurs immediately after CRC transmission, this indicates that data was written while CRC was being sent. This is an indication that the transmitter underflowed, without the CPU being aware of it. The transmit IP is reset either by writing data to the transmit buffer or by issuing the Reset Transmit IP command in WR0. Ordinarily the response to a transmit interrupt is to write more data to the SCC; however, at the end of a frame or block of data where CRC is to be sent next, the Reset Transmit IP command should be issued in lieu of data.

### 4.2.3 External/Status Interrupts

There are several sources of External/Status interrupts, each of which may be individually enabled in WR15. The master enable for the External/Status interrupts is located in WR1 (D0). The individual enable bits in WR15 control whether or not latches will be present in the path from the source of interrupt to the status bit in RR0. If an individual enable bit in WR15 is set to "0" the latches are not present in the signal path and the value read in RR0 reflects the current status. An interrupt source whose individual enable in WR15 is "0" is not a source of External/Status interrupts even though the External/Status Interrupt Enable bit is set. When an individual enable in WR15 is set to "1," the latch is present in the signal path. The latches for the sources of External/Status interrupts are not independent. Rather, they all close at the same time as a result of a state change by one of the sources of interrupt. Thus, a read of RR0 returns the current status for any bits whose individual enable is "0" and either the current state or the latched state of the remainder of the bits. To guarantee the current status the processor should issue a Reset External/Status Interrupts command in WR0 to open the latches. The External/Status IP is set by the closing of the latches and remains set as long as they are closed. If the master enable for the External/Status interrupts is not set, the IP will never be set, even though the latches may be present in the signal paths and working as described. Because the latches close on the current status but give no indication of change, the processor must maintain a copy of RR0 in memory. When the SCC generates an External/Status interrupt the processor should read RR0 and determine which condition changed state and take appropriate action. The copy of RR0 in memory must then updated and the Reset External/Status Interrupt command issued. Care must be taken in writing the interrupt service routine for the External/Status interrupts because it is possible for more than one status condition to change state at the same time. All of the latch bits in RR0 should be compared to the copy of RR0 in memory. If none have changed and the ZC interrupt is enabled, the Zero Count condition caused the interrupt.

The operation of the individual enable bits in WR15 for each of the six sources of External/Status interrupts is identical, but subtle differences exist in the operation of each source of interrupt. The six sources are Break/Abort, Underrun/EOM, CTS, DCD, Sync/Hunt and Zero Count. The Break/Abort, Underrun/EOM, and Zero Count conditions are internal to the SCC, while Sync/Hunt may be internal or external, and CTS and DCD are purely external signals. In the following discussions each source is assumed to be enabled, so that the latches are present, and the External/Status interrupts are enabled as a whole. Recall that the External/Status IP is set while the latches are closed and that the state of the signal is reflected immediately in RR0 if the latches are not present.

The Break/Abort status is used in asynchronous and SDLC modes but is always "0" in synchronous modes other than SDLC. In asynchronous modes this bit is set when a break sequence (null character plus framing error) is detected in the receive data stream, and remains set as long as "0s" continue to be received. This bit is reset when a "1" is received. A single null character is left in the receive FIFO each time that the break condition is terminated. This character should be read and discarded. In SDLC mode this bit is set by the detection of an abort sequence, which is seven or more contiguous "1s" in the receive data stream. The bit is reset when a "0" is received. A received abort forces the receiver into Hunt, which is also an external/status condition. Though these two bits change state at roughly the same time, one or two External/Status interrupts may be generated as a result. The Break/Abort bit is unique in that both transitions are guaranteed to cause the latches to close, even if another External/Status interrupt is pending at the time these transitions occur. This guarantees that a break or abort will be caught.

The Transmit Underrun/EOM bit is used in synchronous modes to control the transmission of CRC. This bit is reset by issuing the Reset Transmit Underrun/EOM command in WR0. However, this transition does not cause the latches to close; this occurs only when the bit is set. To inform the processor of this fact, the SCC sets this bit when CRC is loaded into the Transmit Shift register. This bit will also be set if the processor issues the Send Abort command in WR0. The bit is always set in Asynchronous mode.

The CTS bit reports the state of the $\overline{\text{CTS}}$ input, and the DCD bit reports the status of the DCD input. Both bits latch on either input transition. In both cases, after the Reset External/Status Interrupt command is issued, if the latches are closed, they remain closed if there is any odd number of transitions on an input; they will be open if there is an even number of transistions on the input.

The Zero Count bit is set when the counter in the baud rate generator reaches a count of "0" and is reset when the counter is reloaded. The latches are closed only when this bit is set to "1," and the status in RR0 always reflects the current status. While the Zero Count IE bit in WR15 is reset this bit is forced to "0."

There are a variety of ways in which the Sync/Hunt may be set and reset, depending on the SCC's mode of operation. In the Asynchronous mode this bit reports the state of the $\overline{\text{SYNC}}$ pin, latching on both input transitions. The same is true of External Sync mode. However, if the crystal oscillator is enabled while in Asynchronous mode this bit will be forced to "0" and the latches will not be closed. Selecting the crystal option in External Sync mode is illegal, but the result will be the same. In Synchronous modes other than SDLC the Sync/Hunt reports the Hunt state of the receiver. Hunt mode is entered when the processor issues the Enter Hunt command in WR3. This forces the receiver to search for a sync character match in the receive data stream. Because both transitions of the Hunt bit close the latches, issuing this command will cause an External/Status interrupt. The SCC resets this bit when character synchronization has been achieved, causing the latches to again be closed. In these synchronous modes the SCC will not reenter the Hunt mode automatically; only the Enter Hunt command will set this bit. In SDLC mode this bit is also set by the Enter Hunt command, but the receiver will also automatically enter the Hunt mode if an Abort sequence is received. The receiver leaves Hunt upon receipt of a flag sequence. Both transitions of the Hunt bit will cause the latches to be closed. In SDLC mode the receiver will automatically synchronize on Flag characters. The receiver is in Hunt mode when it is enabled, so the Enter Hunt command will probably never be needed.

If careful attention is paid to details, the interrupt service routine for External/Status interrupts is straightforward. To determine which bit or bits changed state, the routine should first read RR0 and compare it to a copy from memory. For each changed bit the appropriate action should be taken and the copy in memory updated. The service routine should close with a Reset External/Status interrupts command to reopen the latches. The copy of RR0 in memory should always have the Zero Count bit set to "0," since this will be the state of the bit after the Reset External/Status interrupts command at the end of the service routine. When the processor issues the Reset Transmit Underrun/EOM latch command in WR0, the Transmit Underrun/EOM bit in the copy of RR0 in memory should be reset because this transition does not cause an interrupt.

## 4.3 BLOCK TRANSFERS

The SCC offers several alternatives for the block transfer of data. The various options are selected by WR1 (bits D7 through D5) and WR14 (bit D2). Each channel in the SCC has two pins which may be used to control the block transfer of data. Both pins in each channel may be programmed to act as DMA Request signals, and one pin in each channel may be programmed to act as a Wait signal for the CPU. In either mode, it is advisable to select and enable the mode in two separate accesses of the appropriate register. The first access should select the mode and the second access should enable the function. This procedure prevents glitches on the output pins. Reset forces Wait mode, with $\overline{\text{W/REQ}}$ open-drain.

### 4.3.1 Wait on Transmit

The Wait function on transmit is selected by setting both D6 and D5 to "0" and then enabling the function by setting D7 of WR1 to "1." In this mode the $\overline{\text{W/REQ}}$ pin carries the $\overline{\text{WAIT}}$ signal, and is open-drain when inactive and Low when active. When the processor attempts to write to the transmit buffer when it is full, the SCC will assert $\overline{\text{WAIT}}$ until the buffer is empty. This allows the use of a block-move instruction to transfer the transmit data. In the case of the AmZ8030, $\overline{\text{WAIT}}$ will go active in response to $\overline{\text{DS}}$ going active, but only if WR8 is being accessed and a write is attempted. In all other cases $\overline{\text{WAIT}}$ will remain open-drain. In the case of the AmZ8530, $\overline{\text{WAIT}}$ will go active in response to WR going active, but only if the data buffer is being accessed, either directly or via the pointers. The WAIT pin is released in response to the falling edge of PCLK. Details of the timing are shown in Figure 4-3.

### 4.3.2 Wait on Receive

The Wait function on receive is selected by setting D6 or WR1 to "0," D5 of WR1 to "1," and then enabling the function by settling D7 of WR1 to "1." In this mode the W/REQ pin carries the WAIT signal, and is open-drain when inactive and Low when active. When the processor attempts to read data from the receive FIFO when it is empty, the SCC will assert WAIT until a character has reached the top of the FIFO. This allows the use of a block-move instruction to transfer the receive data. In the case of the AmZ8030, WAIT will go active in response to RD going active, but only if RR8 is being accessed and a read is attempted. In all other cases WAIT will remain open-drain. In the case of the AmZ8530, WAIT will go active in response to RD going active, but only if the receive data FIFO is being accessed, either directly or via the pointers. The WAIT pin is released in response to the falling edge of PCLK. Details of the timing are shown in Figure 4-4.

### 4.3.3 DMA Requests

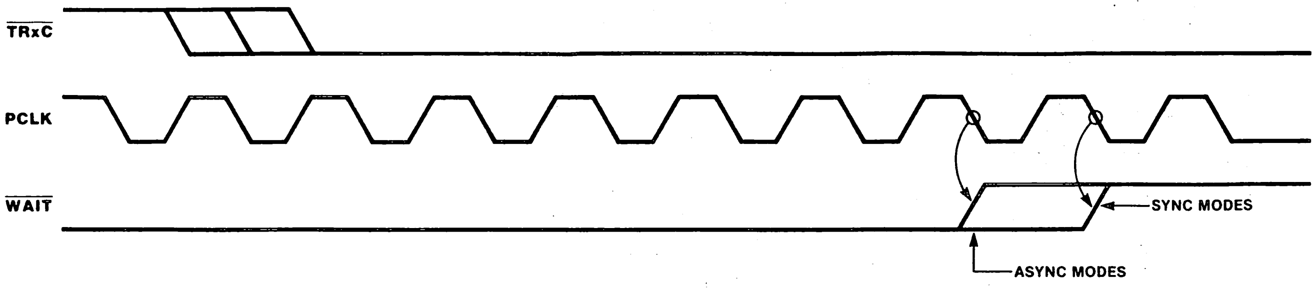
The two DMA request pins W/REQ and DTR/REQ can be programmed to be used as DMA requests. The W/REQ pin can be used as either a transmit or a receive request and the DTR/REQ pin can only be used as a receive request. For full-duplex operation, the W/REQ is, therefore, used for transmit and the DTR/REQ is used for receive. These modes are described below.

### 4.3.4 DMA Request on Transmit (using W/REQ)

The Request on Transmit function is selected by setting D6 of WR to "1," D5 of WR1 to "0," and then enabling the function by setting D7 of WR1 to "1." In this mode the W/REQ pin carries the REQUEST signal, which is active Low. When this mode is selected, but not yet enabled, the W/REQ is driven High.. When the enable bit is set, REQUEST goes Low if the transmit buffer is empty at the time, or will remain High until the transmit buffer becomes empty. Note that the REQUEST pin will follow the state of the transmit buffer even though the transmitter is disabled. Thus, if the REQUEST is enabled, the DMA may write data to the SCC before the transmitter is enabled. This will not cause a problem in Asynchronous mode but may cause problems in Synchronous mode because the SCC will send data in preference to flags or sync characters. It may also complicate the CRC initialization, which cannot be done until after the transmitter is enabled. With only one exception, the REQUEST pin directly follows the state of the transmit buffer in this mode. REQUEST goes Low when the transmit buffer empties and remains Low until the transmit buffer is filled. The SCC generates only one falling edge on REQUEST per character requested and the timing for this is shown in Figure 4-5 The one exception occurs in synchronous modes at the end of CRC transmission. At the end of CRC transmission, when the closing flag or sync character is loaded into the Transmit Shift register, REQUEST is pulsed High for one PCLK cycle. The DMA may use this falling edge on REQUEST to write the first character of the next frame or block to the SCC. In the case of the AmZ8030, REQUEST will go High in response to the falling edge of DS, but only if the appropriate transmit buffer in the SCC is accessed. This is shown in Figure 4-6. In the case of the AmZ8530, REQUEST will go High in response to the falling edge of WR, but only when the appropriate transmit buffer in the SCC is accessed. This is shown in Figure 4-7.

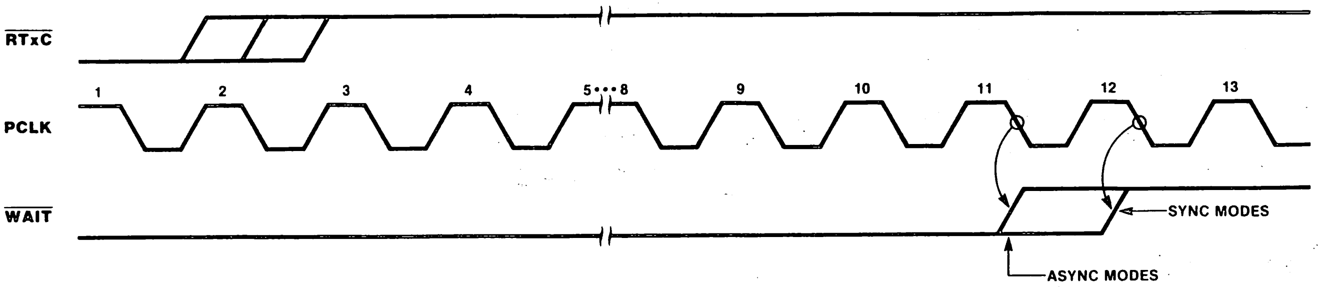
### 4.3.5 DMA Request on Transmit (using DIR/REQ)

A second Request on Transmit function is available on the DTR/REQ pin. This mode is selected by setting D2 of WR14 to "1." When this bit is set to "1," REQUEST goes Low if the transmit buffer is empty at the time, or will go High until the transmit buffer becomes empty. While D2 of WR14 is set to "0," the DTR/REQ pin is DTR and follows the inverted state of D7 in WR5. This pin will be High after a channel or hardware reset and in the DTR mode. In the Request mode REQUEST will follow the state of the transmit buffer even though the transmitter is disabled. Thus if REQUEST is enabled before the transmitter is enabled, the DMA may write data to the SCC before the transmitter is enabled. This will not cause a problem in Asynchronous mode, but may cause problems in Synchronous mode because the SCC will send data in preference to flags or sync characters. It may also complicate the CRC initialization, which cannot be done until after the transmitter is enabled. With only one exception, the REQUEST pin directly follows the state of the transmit buffer in this mode. REQUEST goes Low when the transmit buffer empties and remains Low until the transmit buffer is filled. The SCC generates only one falling edge on REQUEST per



07513A 4-3

Figure 4-3 Wait on Transmit



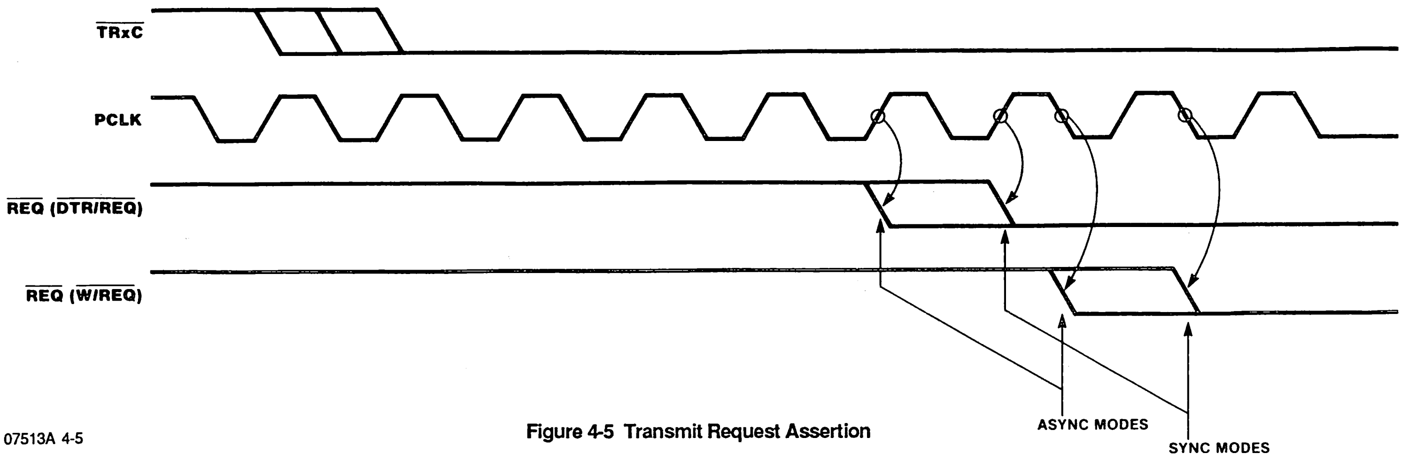
07513A 4-4

Figure 4-4 Wait on Receive

character requested and the timing for this is shown in Figure 3-14. The one exception occurs in synchronous modes at the end of CRC transmission. At the end of CRC transmission, when the closing flag or sync character is loaded into the Transmit Shift register, REQUEST is pulsed High for one PCLK cycle. The DMA may use this falling edge on REQUEST to write the first character of the next frame or block to the SCC. The Request signal on DTR/REQ differs from the one on W/REQ in that it does not go immediately High in response to the access which writes to the transmit buffer. This is because the registers in the SCC are not written during the actual access, but are delayed by some number of PCLK cycles. The Request signal on DTR/REQ follows the state of the transmit buffer exactly while the Request signal on W/REQ goes inactive in anticipation of the transmit buffer becoming full. The timing of the Request signal on both pins is shown for the AmZ8030 in Figure 4-6 and for the AmZ8530 in Figure 4-7.

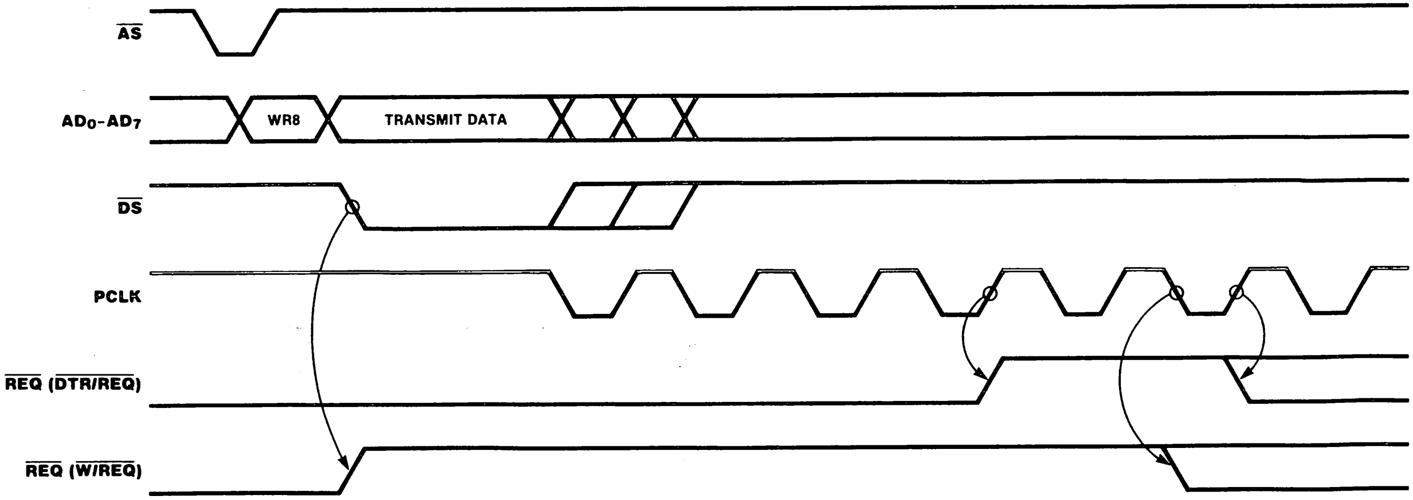
### 4.3.6 DMA Request on Receive

The Request on Receive function is selected by setting D6 and D5 of WR1 to "1" and then enabling the function by setting D7 of WR1 to "1." In this mode the W/REQ pin carries the REQUEST signal, which is active Low. When this mode is selected, but not yet enabled, the W/REQ pin is driven High. When the enable bit is set REQUEST goes Low if the receive buffer contains a character at the time, or will remain High until a character enters the receive buffer. Note that the REQUEST pin will follow the state of the receive buffer even though the receiver is disabled. Thus, if the receiver is disabled and REQUEST is still enabled, the DMA will transfer the previously received data correctly. In this mode the REQUEST pin directly follows the state of the receive buffer with only one exception. REQUEST goes Low when a character enters the receive buffer and remains Low until this character is removed from the receive buffer. The SCC generates only one falling edge on REQUEST per character transfer requested and the timing for this is shown in Figure 4-8. The one exception occurs in the case of a special receive condition in the Receive Interrupt on First Character or Special Condition mode, or the Receive Interrupt on Special Condition Only mode. In the two interrupt modes any receive character with a special receive condition is locked at the top of the FIFO until an Error Reset command is issued. This character in the receive FIFO would ordinarily cause additional DMA Requests after the first time it is read. However, the logic in the SCC guarantees only one falling edge on REQUEST by holding REQUEST High from the time the character with the special receive condition is read, and the FIFO locked, until after the Error Reset command has been issued. Once the FIFO is unlocked by the Error Reset command, REQUEST again follows the state of the receive buffer. In the case of the Z8030, REQUEST will go High in response to the falling edge of DS, but only if the appropriate receive buffer in the SCC is accessed. This is shown in Figure 4-9. In the case of the AmZ8530, REQUEST will go High in response to the falling edge of RD, but only when the appropriate receive buffer in the SCC is accessed. This is shown in Figure 4-10.



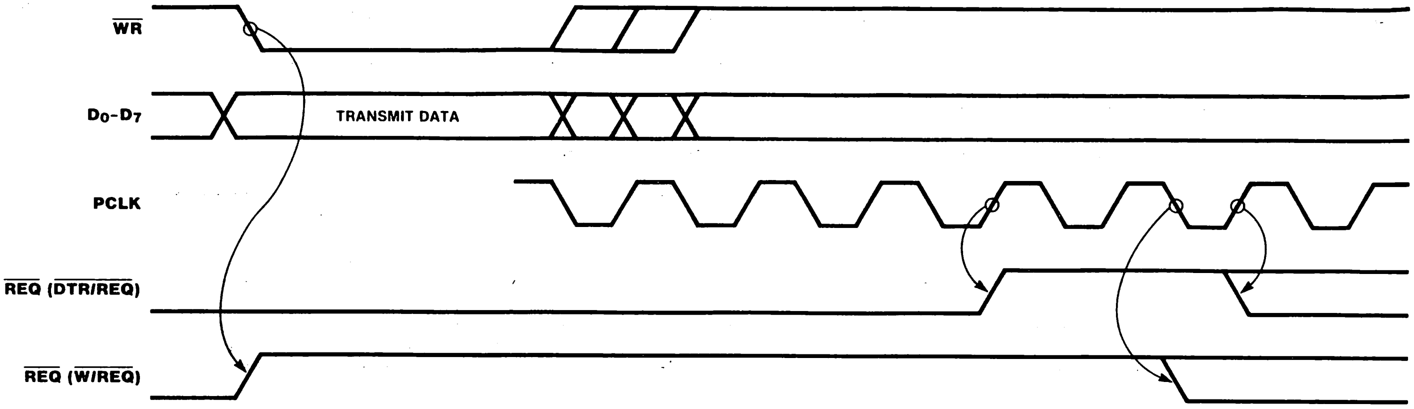
07513A 4-5

Figure 4-5 Transmit Request Assertion



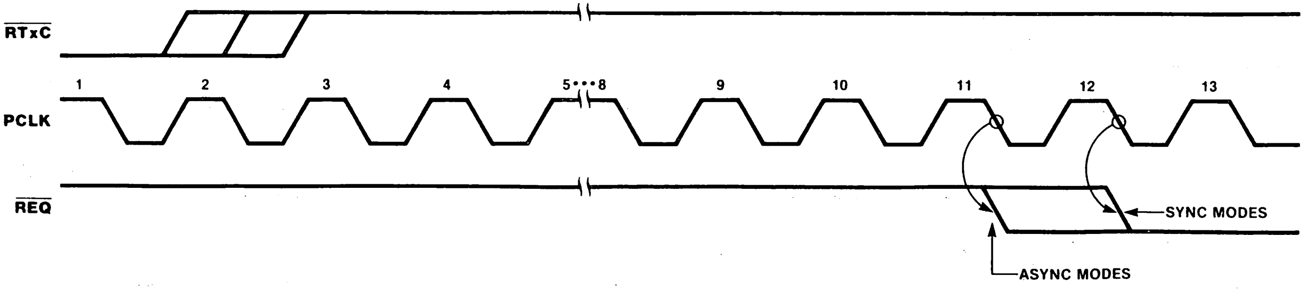
07513A 4-6

Figure 4-6 Z8030 Transmit Request Release



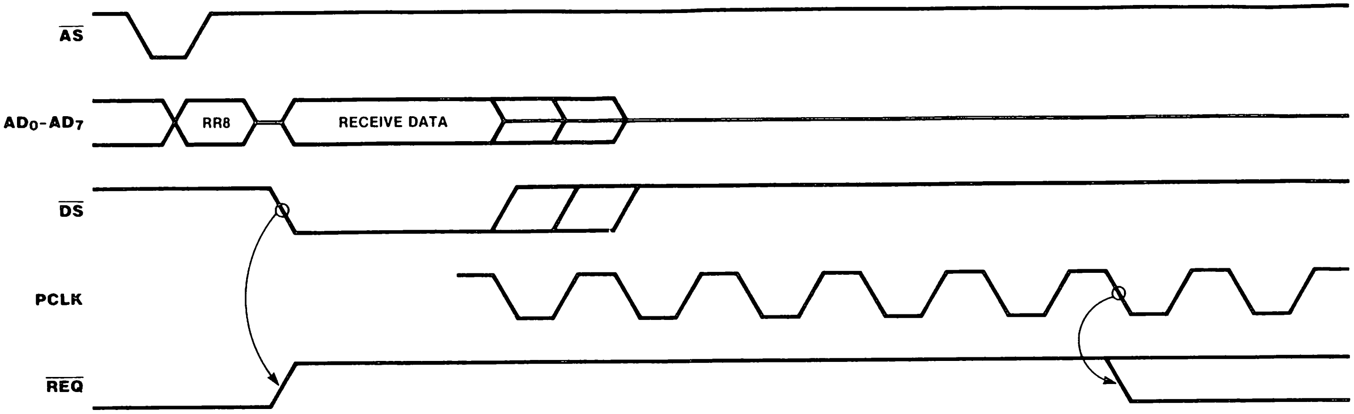
07513A 4-7

Figure 4-7 Z8530 Transmit Request Release



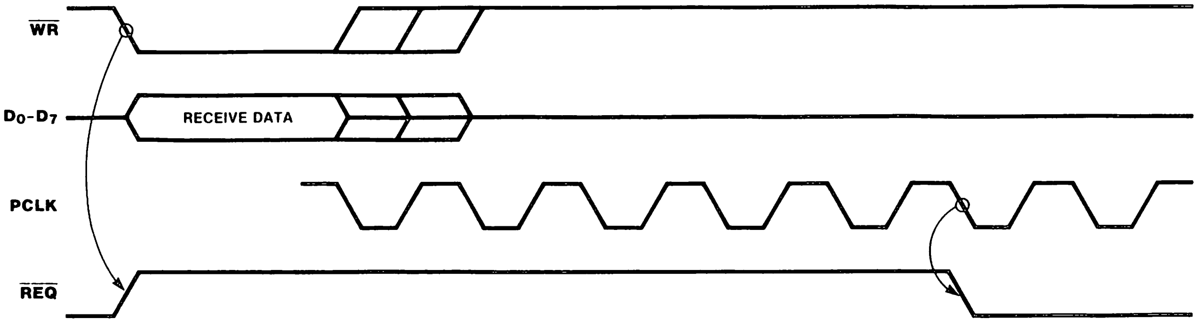
07513A 4-8

Figure 4-8 Receive Request Assertion



07513A 4-9

Figure 4-9 Z8030 Receive Request Release



07513A 4-10

Figure 4-10 Z8530 Receive Request Release

CHAPTER 5

# PROGRAMMING DATA COMMUNICATION MODES

## 5.0 INTRODUCTION

The SCC provides two independent full-duplex channels programmable for use in any common asynchronous or synchronous data communication protocol. These include asynchronous, synchronous byte-oriented protocols, monosync, IBM Bisync, and bit-oriented protocols such as HDLC and SDLC. This chapter is divided into 3 sections: Asynchronous, Synchronous, and SDLC.

## 5.1 ASYNCHRONOUS MODE

The SCC supports Asynchronous mode with a number of programmable options including the number of bits per character, the number of stop bits, the clock factor, modem interface signals and break detect and generation. Asynchronous mode is selected by programming the desired number of stop bits in $D_3$ and $D_2$ of WR4. Programming these two bits with other than "00" places both the receiver and transmitter in Asynchronous mode. In this mode, the SCC ignores the state of bits $D_4$, $D_3$, $D_2$, and $D_1$ of WR3, bits $D_5$ and $D_4$ of WR4, bits $D_2$ and $D_0$ of WR5, all of WR6 and WR7 and all of WR10 except $D_6$ and $D_5$. Bits that are ignored may be programmed with "1" or "0" or not at all.

### 5.1.1 Asynchronous Receive

Asynchronous mode is selected by specifying the number of stop bits per character in WR4. This selection applies only to the transmitter, however, as the receiver always checks for one stop bit. If after character assembly the receiver finds this stop bit to be a "0", the Framing Error bit in the receive error FIFO is set at the same time that the character is transferred to the receive data FIFO. This error bit accompanies the data to the top of the FIFO, where it generates a special receive condition. The Framing Error bit is not latched, and so must be read in RR1 before the accompanying data is read.

The number of bits per character is controlled by bits D7 and D6 of WR3. Five, six, seven, or eight bits per character may be selected via these two bits. Data is right-justifed with the unused bits set to "1s". An additional bit, carrying parity information, may be selected by setting bit D0 of WR4 to "1". Note that this also enables parity for the transmitter. The parity sense is selected by bit D1 of WR4. If this bit is set to "1", the received character is checked for even parity, if set to "0", the received character is checked for odd parity. The additional parity bit per character is transferred to the receive data FIFO along with the data if the data plus parity is eight bits or less. The Parity Error bit in the receive error FIFO may be programmed to cause a special receive condition interrupt by setting bit D2 of WR1 to "1". This error bit is latched and so will remain active, once set, until an Error Reset command has been issued. If interrupts are not used to transfer data, the Parity Error, Framing Error, and Overrun Error bits in RR1 should be checked before the data is removed from the receive data FIFO.

The break condition is continuous "0s", as opposed to the usual continuous ones during an idle. The SCC recognizes the Break condition upon seeing a null character (all "0s") plus a framing error. Upon recognizing this sequence the Break bit in RR0 will be set and will remain set until a "1" is received. At this point the break condition is no longer present. At the termination of a break the receive data FIFO contains a single null character, which should be read and discarded. The Framing Error bit will not be set for this character, but if odd parity has been selected, the Parity Error bit will be set. Caution should be exercised if the receive data line contains a switch that is not debounced to generate breaks. Switch bounce may cause multiple breaks, recognized by the SCC to be additional characters assembled in the receive data FIFO. It may also cause a receive overrun condition being latched.

The SCC may be programmed to accept a receive clock that is one, sixteen, thirty-two, or sixty-four times the data rate. This is selected by bits D7 and D6 in WR4. The 1X mode is used when bits are synchronized external to the receiver. The 1X mode is the only mode in which a data encoding method other than NRZ may be used. The clock factor is common to the receiver and transmitter.

The SCC provides up to three modem control signals associated with the receiver. The $\overline{\text{SYNC}}$ pin is a general-purpose input whose state is reported in the Sync/Hunt bit in RR0. If the crystal oscillator is enabled, this pin is not available and the Sync/Hunt bit is forced to "0". Otherwise, the SYNC pin may be used to carry the Ring Indicator signal. The DTR/REQ pin carries the inverted state of the DTR bit (D7) in WR5 unless this pin has been programmed to carry a DMA Request signal. The DCD pin is ordinarily a simple input to the DCD bit in RR0. However, if the Auto Enables mode is selected by setting D5 of WR3 to "1", this pin becomes an enable for the receiver. That is, if Auto Enables is on and the DCD pin is HIGH, the receiver is disabled. While the DCD pin is LOW, the receiver is enabled.

The initialization sequence for the receiver in Asynchronous mode is: WR4 first to select the mode, then WR3 and WR5 to select the various options. At this point, the other registers should be initialized as necessary. When all of this is complete the receiver may be enabled by setting bit $D_0$ or WR3 to "1".

### 5.1.2 Asynchronous Transmit

Asynchronous mode is selected by specifying the number of stop bits per character in bits $D_3$ and $D_2$ of WR4. The three options available are one, one-and-a-half, or two stop bits per character. These two bits only select the number of stop bits for the transmitter, as the receiver always checks for one stop bit.

The number of bits per transmitted character is controlled both by Bits $D_6$ and $D_5$ in WR5 and the way the data is formatted within the transmit buffer. The bits in WR5 allow the option of five, six, seven, or eight bits per character. When five bits per character is selected the data may be formatted before being written to the transmit buffer to allow transmission of from one to five bits per character.

This formatting is shown in Table 5-1.

**Table 5-1 Data Format—Five Bits Or Less**

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | One data bit |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | Two data bits |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Three data bits |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Four data bits |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Five data bits |

In all cases the data must be right-justified, with the unused bits being ignored except in the case of five bits per character. An additional bit, carrying parity information, may be automatically appended to every transmitted character by setting bit D0 of WR4 to "1". This bit is sent in addition to the number of bits specified in WR4 or by the data format. The parity sense is selected by bit D1 of WR4. If this bit is set to "1", the transmitter sends even parity, if set to "0", the parity is odd.

The transmitter may be programmed to send a Break by setting bit D4 of WR5 to "1". The transmitter will send continuous "0s" from the first transmit clock edge after this command is issued, until the first transmit clock edge after this bit is reset. The transmit clock edges referred to here are those that define transmitted bit cell boundaries.

An additional status bit for use in Asynchronous mode is available in bit $D_0$ or RR1. This bit, called All Sent, is set when the transmitter is completely empty and any previous data or stop bits have reached the TxD pin. The All Sent bit can be used by the processor as an indication that the transmitter may be safely disabled.

The SCC may be programmed to accept a transmit clock that is one, sixteen, thirty-two, or sixty-four time the data rate. This is selected by bits $D_7$ and $D_6$ of WR4, in common with the clock factor for the receiver. Note that the chosen clock factor may restrict the number of stop bits that may be transmitted. In particular, when the clock rate and data rate are identical, one-and-a-half stop bits are not allowed. If any length other than one stop bit is desired in the times one mode, only two stop bits may be used.

There are two modem control signals associated with the transmitter provided by the SCC. The RTS pin is a simple output that carries the inverted state of the RTS bit ($D_1$) in WR5, unless the Auto Enables bit ($D_5$) is set in WR3. When Auto Enables is set the RTS pin will immediately go LOW when the RTS bit is set. However, when the RTS bit is reset the RTS pin remains LOW until the transmitter is completely empty and the last stop bit has left the TxD pin. Thus the RTS pin may be used to disable external drivers for the transmit data. The CTS pin is ordinarily a simple input to the CTS bit in RR0. However, if Auto Enables mode is selected this pin becomes an enable for the transmitter. That is, if Auto Enables is on and the CTS pin is HIGH, the transmitter is disabled; the transmitter is enabled while the CTS pin is LOW.

The initialization sequence for the transmitter in Asynchronous mode is: WR4 first to select the mode, then WR3 and WR5 to select the various options. At this point the other registers should be initialized as necessary. When all of this is complete, the transmitter may be enabled by setting bit $D_3$ of WR5 to "1". Note that the transmitter and receiver may be initialized at the same time.

## 5.2 SYNCHRONOUS MODE

In synchronous modes of operation a special bit pattern is used to provide character synchronization. The SCC offers several options to support Synchronous mode including various sync character lengths, the number of bits per data character, parity generation and checking, CRC generation and checking, as well as modem controls and a transmitter to receiver synchronization function. Synchronous mode is selected by programming bits $D_3$ and $D_2$ of WR4 with "0s". This selects Synchronous mode, as opposed to Asynchronous mode, but this selection is further modified by bits $D_5$ and $D_7$ of WR4 as well as bits $D_1$ and $D_0$ of WR10. The sync character or characters are written in WR6 and WR7. In all synchronous modes, except External Sync the state of bits $D_7$ and $D_6$ of WR4 are forced to "0" to select the times one clock mode. In External Sync mode these two bits must be programmed with "0s".

### 5.2.1 Synchronous Receive

The receiver in the SCC searches for character synchronization only while it is in Hunt mode. In this mode the receiver is idle except that it is searching the incoming data stream for a sync character match. The receiver is in Hunt mode when it is first enabled, and may be placed in Hunt mode by command from the processor. This is accomplished by issuing the Enter Hunt Mode command in WR3. This bit ($D_4$) is a command; writing a "0" to it has no effect. The Hunt status of the receiver is reported by the Sync/Hunt bit in RR0. Sync/Hunt is one of the possible sources of external/status interrupts, with both transitions causing an interrupt. This is true even if the Sync/Hunt bit is set as a result of the processor issuing the Enter Hunt Mode command.

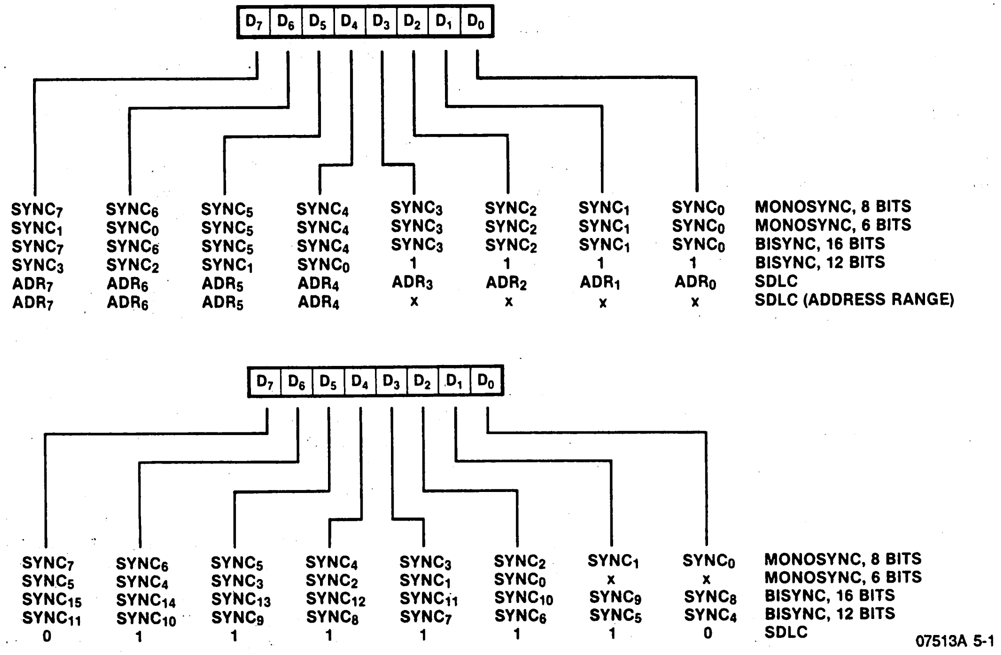
An 8-bit sync character is selected by setting bits $D_5$ and $D_4$ of WR4, as well as bit $D_0$ of WR10, to "0". With this option the receiver searches the data stream for a match with the eight bits in WR7. The 6-bit sync option requires the same programming except that D0 of WR10 is set to "1" and the sync character is held in the high-order six bits of WR7. The SCC also allows the option of double length sync characters. This is selected by setting bit $D_5$ of WR4 to "0" and bit $D_4$ of WR4 to "1". The selection between 12 and 16 bits of sync character is controlled by bit D0 of WR10. A "0" selects 16 bits of sync character, while a "1" in this bit selects a 12-bit sync character. The arrangement of the sync character in WR6 and WR7 is shown in Figure 5-1. For those applications requiring any other sync character length, the SCC makes provision for an external circuit to provide a character synchronization signal on the $\overline{\text{SYNC}}$ pin. This mode is selected by setting bits $D_5$ and $D_4$ of WR4 to "1". In this mode the Sync/Hunt bit in RR0 reports the state of the $\overline{\text{SYNC}}$ pin but the receiver must still be placed in Hunt mode when the external logic is searching for a sync character match. When the receiver is in Hunt mode and the $\overline{\text{SYNC}}$ pin is driven LOW, two receive clock cycles after the last bit of the sync character is received, character assembly will begin on the rising edge of the receive clock immediately preceding the activation of $\overline{\text{SYNC}}$. This is shown in Figure 5-2. The receiver leaves Hunt mode when $\overline{\text{SYNC}}$ is driven LOW. In all cases except External Sync mode the $\overline{\text{SYNC}}$ pin is an output that is driven LOW by the SCC to signal that a sync character has been received. The $\overline{\text{SYNC}}$ pin is activated regardless of character boundaries so any external circuitry using it should only respond the $\overline{\text{SYNC}}$ pulse that occurs while the receiver is in Hunt mode. The timing for the $\overline{\text{SYNC}}$ signal is shown in Figure 5-3.

The number of bits per character is controlled by bits $D_7$ and $D_6$ of WR3. Five, six, seven, or eight bits per character may be selected via these two bits. The data is right-justified in the receive data buffer. The SCC merely takes a snapshot of the receive data stream at the appropriate times so the "unused" bits in the receive buffer are only the bits following the character in the data stream. An additional bit, carrying parity information, may be selected by setting bit $D_0$ of WR4 to "1". If this bit is set to "1", the received character is checked for even parity, if set to "0", the received character is checked for odd parity. The additional bit per character is visible in the receive data FIFO if the data plus parity is eight bits or less. The parity bit is not visible when there are eight data bits per character. The Parity Error bit in the receive error FIFO may be programmed to cause a Special Receive Condition interrupt by setting bit $D_2$ of WR1 to "1". This error bit is latched and so will remain active, once set, until an Error Reset command has been issued. If interrupts are not used to transfer data the Parity Error, CRC Error, and Overrun Error bits in RR1 should be checked before the data is removed from the receive data FIFO. The character length may be changed at any time before the new number of bits has been assembled by the receiver, but, care should be exercised as unexpected results may occur. A representative example, switching from five bits to eight bits and back to five bits is shown in Figure 5-4.

It is sometimes desirable to prevent sync characters in the receive data stream from being transferred to the receive data FIFO. This function is available in the SCC by setting the Sync Character Load Inhibit bit (D1) in WR3 to "1". While this bit is set to "1", character about to be loaded into the receive data FIFO is compared with the contents of WR6. If all eight bits match the character, it is not loaded into the receive data FIFO. Because the comparison is across eight bits, this function works correctly only when the number of bits per character is the same as the sync character length. Thus it cannot be used with 12- or 16-bit sync characters. Both leading sync characters and sync characters embedded in the data will be properly removed in the case of an 8-bit sync character, but only the leading sync characters may be properly removed in the case of a 6-bit sync character. Care must be exercised in using this feature because sync characters not transferred to the receive data FIFO will automatically be excluded from CRC calculation. This works properly only in the 8-bit case.

Either of two CRC polynomials may be used in synchronous modes, selected by bit D$_2$ in WR5. If this bit is set to "1", the CRC-16 polynomial is used, if this bit is set to "0", the CRC-CCITT polynomial is used. This bit controls the polynomial selection for both the receiver and transmitter. The initial state of the generator and checker is controlled by bit D$_7$ of WR10. When this bit is set to "1", both the generator and checker will have an initial value of all ones, if this bit is set to "0", the initial values will be all "0s". The SCC presets the checker whenever the receiver is in Hunt mode so a CRC reset command is not strictly necessary. However, the CRC checker may be preset by issuing the Reset CRC Checker command in WR0. This command is encoded in bits D$_7$ and D$_6$ of WR0. If CRC is to be used the CRC checker must be enabled by setting bit D0 of WR3 to "1". If sync characters are being stripped from the data stream, this may be done at any time before the first non-sync character is received. If the sync strip feature is not being used, CRC must not be enabled until after the first data character has been transferred to the receive data FIFO. As previously mentioned, 8-bit sync characters stripped form the data stream are automatically excluded from CRC calculation.

Some synchronous protocols require that certain characters be excluded from CRC calculation. This is possible in the SCC because CRC calculation may be enabled and disabled on the fly. To give the processor sufficient time to decide whether or not a particular character should be included in the CRC calculation, the SCC contains an 8-bit time delay between the receive shift register and the

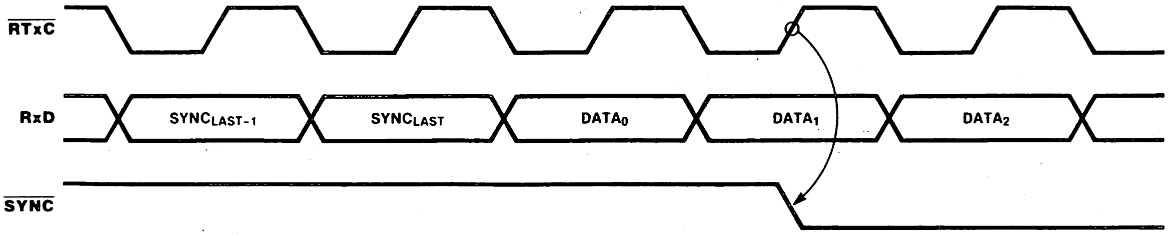


| D$_7$ | D$_6$ | D$_5$ | D$_4$ | D$_3$ | D$_2$ | D$_1$ | D$_0$ | |
|---|---|---|---|---|---|---|---|---|
| SYNC$_7$ | SYNC$_6$ | SYNC$_5$ | SYNC$_4$ | SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | MONOSYNC, 8 BITS |
| SYNC$_1$ | SYNC$_0$ | SYNC$_5$ | SYNC$_4$ | SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | MONOSYNC, 6 BITS |
| SYNC$_7$ | SYNC$_6$ | SYNC$_5$ | SYNC$_4$ | SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | BISYNC, 16 BITS |
| SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | 1 | 1 | 1 | 1 | BISYNC, 12 BITS |
| ADR$_7$ | ADR$_6$ | ADR$_5$ | ADR$_4$ | ADR$_3$ | ADR$_2$ | ADR$_1$ | ADR$_0$ | SDLC |
| ADR$_7$ | ADR$_6$ | ADR$_5$ | ADR$_4$ | x | x | x | x | SDLC (ADDRESS RANGE) |

| D$_7$ | D$_6$ | D$_5$ | D$_4$ | D$_3$ | D$_2$ | D$_1$ | D$_0$ | |
|---|---|---|---|---|---|---|---|---|
| SYNC$_7$ | SYNC$_6$ | SYNC$_5$ | SYNC$_4$ | SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | MONOSYNC, 8 BITS |
| SYNC$_5$ | SYNC$_4$ | SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | x | x | MONOSYNC, 6 BITS |
| SYNC$_{15}$ | SYNC$_{14}$ | SYNC$_{13}$ | SYNC$_{12}$ | SYNC$_{11}$ | SYNC$_{10}$ | SYNC$_9$ | SYNC$_8$ | BISYNC, 16 BITS |
| SYNC$_{11}$ | SYNC$_{10}$ | SYNC$_9$ | SYNC$_8$ | SYNC$_7$ | SYNC$_6$ | SYNC$_5$ | SYNC$_4$ | BISYNC, 12 BITS |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | SDLC |

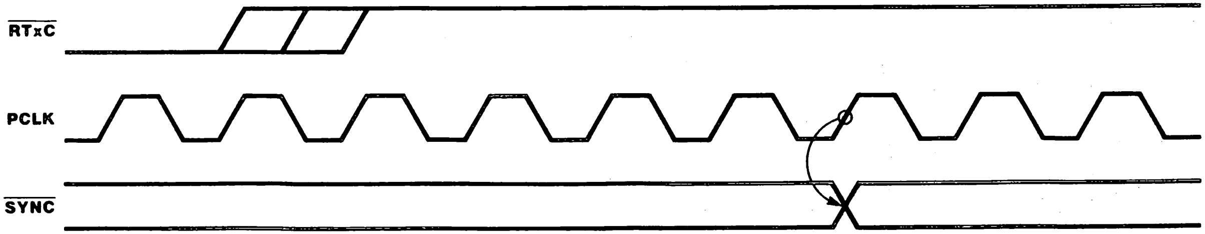
07513A 5-1

**Figure 5-1 Sync Character Programming**



07513A 5-2

Figure 5-2 SYNC as an Output

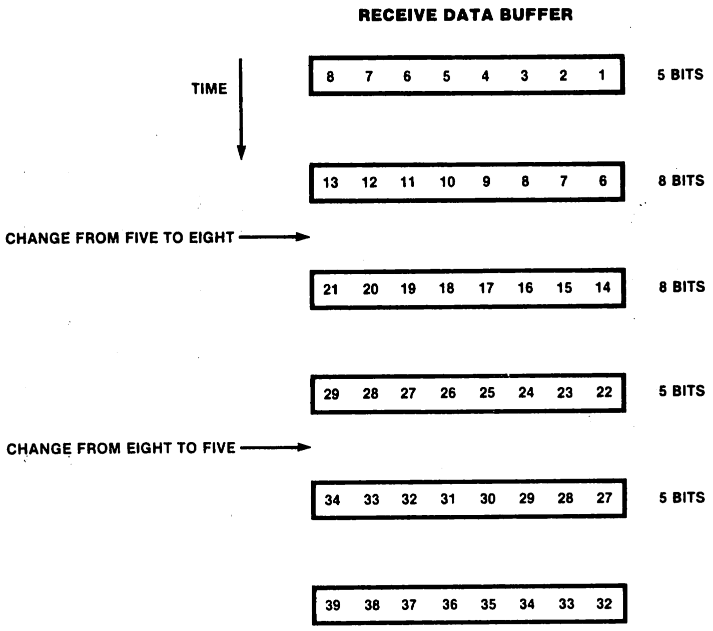


07513A 5-3

Figure 5-3 SYNC as an Input

CRC checker. The logic also guarantees that the calculation will only start or stop on a character boundary by delaying the enable or disable until the next character is loaded into the receive data FIFO. To understand how this works refer to Figure 5-5 and the following explanation. Consider a case where the SCC receives a sequence of eight bytes, called A, B, C, D, E, F, G and H with A received first. Now suppose that A is the sync character, that CRC is to be calculated on B, C, E, and F, and that F is the last byte of this message. Before A is received the receiver is in Hunt mode and the CRC is disabled. When A is in the receive shift register it is compared with the contents of WR7. Since A is the sync character, the bit patterns match and receiver leaves Hunt mode, but character A is not transferred to the receive data FIFO. The CRC remains disabled even though somewhere during the next eight-bit-time the processor reads B and enables CRC. At the end of an eight-bit-time, B is in the 8-bit delay and C is in the receive shift register. At this point, B is loaded into the receive data FIFO. The CRC remains disabled even though somewhere during the next eight bit times the processor reads B and enables CRC. At the end of the eight-bit-time, B is in the 8-bit delay and C is in the receive shift register. Character C is loaded into the receive data FIFO and at the same time the CRC checker is enabled. During the next eight-bit-time, the processor reads C and leaves the CRC enabled. At the end of these eight-bit-times the SCC has calculated CRC on B, character C is the 8-bit delay and D is in the Receive Shift register. D is then loaded into the receive data buffer and at some point during the next eight-bit-time the processor reads D and disables CRC. At the end of these eight-bit-times CRC has been calculated on C, character D is in the 8-bit delay and E is in the Receive Shift register.

Now E is loaded into the receive data FIFO and, at the same time, the CRC is disabled. During the next eight-bit-times the processor reads E and enables the CRC. During this time E shifts into the 8-bit delay, F enters the Receive Shift register and CRC is not being calculated on D. After these eight-bit-times have elapsed, E is in the 8-bit delay, and F is in the Receive Shift register. Now F is transferred to the receive data FIFO and CRC is enabled. During the next eight-bit-times the processor reads F and leaves the CRC enabled. The processor is usually aware that this is the last character in the message and so prepares to check the result of the CRC computation. However, another sixteen bit-times are required before CRC has been calculated on all of character F. At the end of eight-bit-times F is in the 8-bit delay and G is in the Receive Shift register. At this time G is transferred to the receive data FIFO. Character G



07513A 5-4

Figure 5-4 Changing Character Length

must be read and discarded by the processor. Eight bit times later H is transferred to the receive data FIFO also. The result of a CRC calculation is latched in the receive error FIFO at the same time as data is written to the receive data FIFO. Thus the CRC result through character F accompanies character H in the FIFO and will be valid in RR1 until character H is read from the receive data FIFO. The CRC checker may be disabled and reset at any time after character H is transferred to the receive data FIFO. Recall, however, that internally CRC will not be disabled until a character is loaded into the receive data FIFO so the reset command should not be issued until after this occurs. A better alternative is to place the receiver in Hunt mode, which automatically disables and resets the CRC checker.

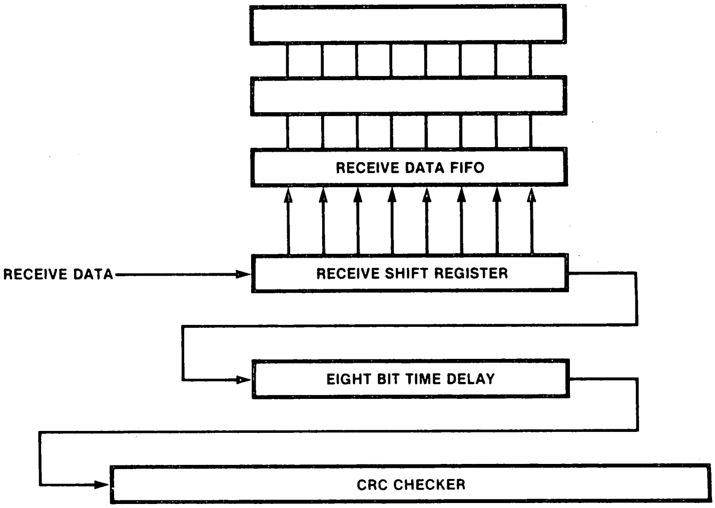
Up to two modem control signals associated with the receiver are available in synchronous modes. The $\overline{\text{DTR}}/\overline{\text{REQ}}$ pin carries the inverted state of the DTR bit (D7) in WR5 unless this pin has been programmed to carry a DMA Request signal. The $\overline{\text{DCD}}$ pin is ordinarily a simple input to the DCD bit in RR0. However, if the Auto Enables mode is selected by setting D5 of WR3 to "1", this pin becomes an enable for the receiver. That is, if Auto Enables is on and the $\overline{\text{DCD}}$ pin is HIGH the receiver is disabled; while the $\overline{\text{DCD}}$ pin is LOW the receiver is enabled.

The initialization sequence for the receiver in synchronous modes is WR4 first, to select the mode, then WR10 to modify it if necessary, WR6 and WR7 to program the sync characters and then WR3 and WR5 to select the various options. At this point the other registers should be initialized as necessary. When all of this is complete the receiver is enabled by setting bit D of WR3 to "1".

### 5.2.2 Synchronous Transmit

Once Synchronous mode has been selected, any of three sync character lengths may be selected. An 8-bit sync character is selected by setting bits $D_5$ and $D_4$ or WR4, as well as bit $D_0$ of WR10 to "0". With this option selected the transmitter sends the contents of WR6 when it has no data to send. The 6-bit sync option requires the same programming except that bit D0 of WR10 is set to "1" and only the least significant six bits of WR6 and used as a time fill. For a 16-bit sync character, set bit $D_4$ of WR4 to "1" and bit $D_5$ of WR4 and bit $D_0$ of WR10 to "0". In this mode the transmitter sends the concatenation of WR6 and WR7 as a time fill. Because the receiver requires that sync characters be left-justified in the registers, while the transmitter requires them to be right-justified, only the receiver will work with a 12-bit sync character. While the receiver is in External Sync mode the transmitter sync length may be six or eight bits, as selected by bit D0 of WR10.

The number of bits per transmitted character is controlled by bits $D_6$ and $D_5$ of WR5 and the way the data is formatted within the transmit buffer. The bits in WR5 allow the option of five, six, seven, or eight bits per character. When five bits per character is selected the data may be formatted



07513A 5-5

Figure 5-5 Receive CRC Data Path

before being written to the transmit buffer to allow transmission of from one to five bits per character. This formatting is shown in Table 5-1. In all cases the data must be right-justified, with the unused bits being ignored except in the case of five bits per character. An additional bit, carrying parity information, may be automatically appended to every transmitted character by setting bit $D_0$ of WR4 to "1". This parity bit is sent in addition to the number of bits specified in WR4 or by the data format. If this bit is set to "1", the transmitter will send even parity, if set to "0", the transmitted parity will be odd.

Either of two CRC polynomials may be used in synchronous modes, selected by bit $D_2$ in WR5. If this bit is set to "1", the CRC-16 polynomial is used and, if this bit is set to "0", the CRC-CCITT polynomial is used. This bit controls the selection for both the transmitter and receiver. The initial state of the generator and checker is controlled by bit $D_7$ of WR10. When this bit is set to "1", both the generator and checker will have an initial value of all ones, if this bit is set to "0", the initial values will be all zeros. The SCC does not automatically preset the CRC generator, so this must be done in software. This is accomplished by issuing the Reset Tx CRC Generator command, which is encoded in bits $D_7$ and $D_6$ of WR0. For proper results this command must be issued while the transmitter is enabled and sending sync characters. If CRC is to be used, the transmit CRC generator must be enabled by setting bit D0 of WR5 to "1". This bit may also be used to exclude certain characters from the CRC calculation. Sync characters are automatically excluded from the CRC calculation and any characters written as data may also be excluded from the calculation by using bit $D_0$ of WR5. Internally, the CRC is enabled or disabled for a particular character at the same time as the character is loaded from the transmit buffer to the Transmit Shift register. Thus, to exclude a character from CRC calculation bit, $D_0$ of WR5 should be set to "0" before the character is written to the transmit buffer. This guarantees that the internal disable will occur when the character moves from the buffer to the shift register. Once the buffer becomes empty, the Tx CRC Enable bit may be written for the next character.

Enabling the CRC generator is not sufficient to control the transmission of CRC. In the SCC this function is controlled by the Tx Underrun/EOM bit, which may be reset by the processor and set by the SCC. When the transmitter underruns (both the transmit buffer and Transmit Shift register are empty) the state of the Tx Underrun/EOM bit determines the action taken by the SCC. If the tx Underrun/EOM bit is set when the underrun occurs, the transmitter will send sync characters, if this bit is reset when the underrun occurs, the transmitter will send the accumulated CRC followed by sync characters. When the CRC is loaded into the transmit Shift register for transmission, the SCC will set the Tx Underrun/EOM bit to indicate this. This transition may be programmed to cause an external/status interrupt, or the Tx Underrun/EOM is available in RR0. The Reset Tx Underrun/EOM Latch command is encoded in bits $D_7$ and $D_6$ of WR0. For correct transmission of the CRC at the end of a block of data, this command must be issued after the first character is written to the SCC but before the transmitter underruns after the last character written to the SCC. The command is usually issued immediately after the first character is written to the SCC so that CRC will be sent if an underrun occurs inadvertently during the block of data.

In synchronous modes, if the transmitter is disabled during transmission of a character, that character will be sent completely. This applies to both data and sync characters. However, if the transmitter is disabled during the transmission of CRC, the 16-bit transmission will be completed, but the remaining bits will come from the SYNC registers rather than the remainder of the CRC.

There are two modem control signals associated with the transmitter provided by the SCC. The $\overline{RTS}$ pin is a simple output that carries the inverted state of the RTS bit ($D_1$) in WR5. The $\overline{CTS}$ pin is ordinarily a simple input to the CTS bit in RR0. However, if Auto Enables mode is selected this pin becomes an enable for the transmitter. That is, if Auto Enables is on and the $\overline{CTS}$ pin is HIGH the transmitter is disabled. While the $\overline{CTS}$ pin is LOW, transmitter is enabled

The initialization sequence for the transmitter in synchronous modes is: WR4 first, to select the mode, then WR10 to modify it if necessary, WR6 and WR7 to program the sync characters, then WR3 and WR5 to program the sync characters, and then WR3 and WR5 to select the various options. At this point, the other registers should be initialized as necessary. When all of this is complete the transmitter may be enabled by setting bit $D_3$ or WR5 to "1". Now that the transmitter is enabled the CRC generator may be initialized by issuing the Reset Tx CRC Generator command in WR0.

### 5.2.3 Transmitter to Receiver Synchronization

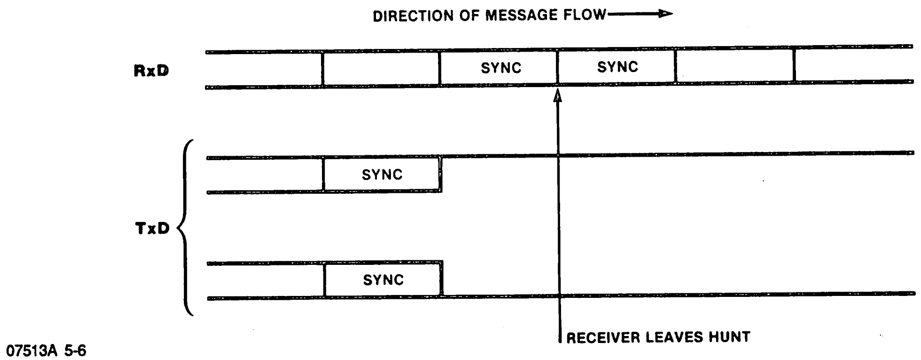
The SCC contains a transmitter-to-receiver synchronization function that may be used to guarantee that the character boundaries for the received and transmitted data are the same. In this mode the receiver is in Hunt and the transmitter is idle, sending either all "1s" or all "0s". When the receiver recognizes a sync character, it leaves Hunt mode and one character time later the transmitter is enabled and begins sending sync characters. Beyond this point the receiver and transmitter are again completely independent, except that the character boundaries are now aligned. This is shown in Figure 5-6. There are several restrictions on the use of this feature in the SCC. First, it will only work with 6-bit, 8-bit or 16-bit sync characters, and the data character length for both the receiver and the transmitter must be six bits with a 6-bit sync character or eight bits with an 8-bit or 16-bit sync character. Of course, the receive and transmit clocks must have the same rate as well as the proper phase relationship.

A specific sequence of operations must be followed to synchronize the transmitter to the receiver. Both the receiver and transmitter must have been initialized for operation in Synchronous mode sometime in the past, although this initialization need not be redone each time the transmitter is synchronized to the receiver. The transmitter is disabled by setting bit $D_3$ of WR5 to "0". At this point the transmitter will send continuous "1s". If it is desired that continuous "0s" be transmitted, the Send Break bit ($D_4$) in WR5 should be set to "1". The transmitter is now idling but must still be placed in the transmitter to receiver synchronization mode. This is accomplished by setting the Loop Mode bit ($D_1$) in WR10 and then enabling the transmitter by setting bit $D_3$ of WR5 to "1". At this point the processor should set the Go Active on Poll bit ($D_4$) in WR10. The final step is to force the receiver to search for sync characters. If the receiver is currently disabled the receiver will enter Hunt mode when it is enabled by setting bit $D_0$ of WR3 to "1". If the receiver is already enabled it may be placed in Hunt mode by setting bit $D_4$ of WR3 to "1". Once the receiver leaves Hunt mode the transmitter is activate on the following character boundary.

## 5.3 SDLC MODE

SDLC mode is useful in bit-oriented protocols. That is, protocols which use the technique of "0" insertion to achieve data transparency. In SDLC mode, frames of information are opened and closed by a unique bit pattern called a flag. The Flag character has a bit pattern of "01111110" and this sequence is unique because all data between the opening and closing flags is prohibited from having more than five consecutive "1s". The transmitter guarantees this by watching the transmit data stream and inserting a "0" after five consecutive ones, irrespective of character boundaries. In turn, the receiver searches the receive data stream for five consecutive "1s" and deletes the next bit if it is a "0". CRC may be used in SDLC mode but only with the CRC- CCITT polynomial, because the transmitter in the SCC automatically inverts the CRC before transmission, and the receiver —to compensate for this—checks the CRC result for the bit pattern "0001110100001111". This is consistent with bit-oriented protocols such as SDLC, HDLC, and ADCCP. There are two unique bit patterns in SDLC mode besides the flag sequence. They are the Abort and EOP (End of Poll) sequence. An Abort is a sequence of from seven to thirteen consecutive "1s" and is used to signal the premature termination of a frame. The EOP is the bit pattern "11111110", which is used in loop applications as a signal to a secondary station that it may begin transmission.

SDLC mode is selected by setting bit D5 of WR4 to "1" and bits $D_4$, $D_3$, and $D_2$ of WR4 to "0". In addition, the flag sequence must be written to WR7. Additional control bits for SDLC mode are located in WR10.



07513A 5-6

Figure 5-6 Transmitter to Receiver Synchronization

### 5.3.1 SDLC Receive

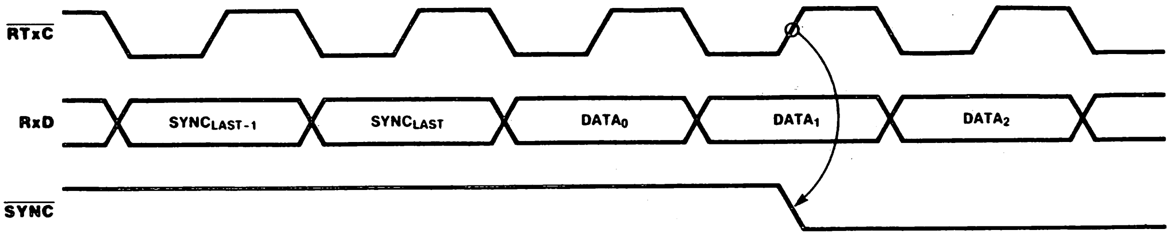
The receiver in the SCC always searches the receive data stream for flag characters in SDLC mode. Ordinarily, the receiver transfers all received data between flags to the receive data FIFO. However, if the receiver is in Hunt mode no flag is received. The receiver is in Hunt mode when first enabled, or the receiver may be placed in Hunt mode by the processor issuing the Enter Hunt mode command in WR3. This bit ($D_4$) is a command, and writing a "0" to it has no effect. The Hunt status of the receiver is reported by the Sync/Hunt bit in RR0. Sync/Hunt is one of the possible sources of external/status interrupts, with both transitions causing an interrupt. This is true even if the Sync/Hunt bit is set as a result of the processor issuing the Enter Hunt mode command. The receiver will automatically enter Hunt mode if an abort is received. Because the receiver always searches the receive data stream for flags and automatically enter Hunt Mode when an abort is received, the receiver will always handle frames correctly, and the Enter Hunt Mode command should never to needed. The SCC will drive the $\overline{\text{SYNC}}$ pin LOW to signal that a flag has been recognized. The timing for the $\overline{\text{SYNC}}$ signal is shown in Figure 5-7.

The first byte in an SDLC frame is assumed by the SCC to be the address of the secondary station for which the frame is intended. The SCC provides several options for handling this address. If the Address Search Mode bit ($D_2$) in WR3 is set to "0", the address recognition logic is disabled and all received frames are transferred to the receive data FIFO. In this mode the software must perform any address recognition. If the Address Search Mode bit is set to "1", only those frames whose address matches the address programmed in WR6 or the global address (all "1s") will be transferred to the receive data FIFO. The address comparison will be across all eight bits of WR6 if the Sync Character Load Inhibit bit ($D_1$) in WR3 is set to "0". The comparison may be modified so that only the four most significant bits of WR6 must match the received address. This mode is selected by setting the Sync Character Load Inhibit bit to "1". In this mode, however, the address field is still eight bits wide. The address field is transferred to the receive data FIFO in the same manner as data. It is not treated differently than data.

The number of bits per character is controlled by bits $D_7$ and $D_6$ of WR3. Five, six, seven, or eight bits per character may be selected via these two bits. The data is right-justified in the receive buffer. The SCC merely takes a snapshot of the receive data stream at the appropriate times, so the "unused" bits in the receive buffer are only the bits following the character in the data stream. An additional bit carrying parity information may be selected by setting bit $D_6$ of WR4 to "1". This also enables parity in the transmitter. The parity sense is selected by bit $D_1$ of WR4. Parity is not normally used in SDLC mode. The character length may be changed at any time before the new number of bits have been assembled by the receiver. Care should be exercised, however, as unexpected results may occur. A representative example, switching from five bits to eight bits and back to five bits is shown in Figure 5-8.

Most bit-oriented protocols allow an arbitrary number of bits between opening and closing Flags. The SCC allows for this by providing three bits of Residue Code in RR1 that indicate which bits in the last few bytes transferred from the receive data FIFO by the processor are actually valid data bits. The meaning of these three bits with each character length option is shown in Table 5-2. As indicated in the table, these bits allow the processor to determine those bits in the information (and not CRC) field. This allows transparent retransmission of the received frame. The Residue Code bits do not go through a FIFO so they change in RR1 when the last character of the frame is loaded into the receive data FIFO. If there are any characters already in the receive data FIFO the Residue Code will be updated before they are read by the processor. Thus these three bits of RR1 should be ignored by the processor unless the End of Frame bit in RR1 is set.

Only the CRC-CCITT polynomial may be used for CRC calculation in SDLC mode, although the generator and checker may be preset to all "1s" or all "0s". The CRC-CCITT polynomial is selected by setting bit $D_2$ of WR5 to "0", bit $D_7$ of WR10 controls the preset value. If this bit is set to "1", the generator and checker are preset to "1s", if this bit is reset, the generator and checker are preset to all "0s". The receiver expects the CRC to be inverted before transmission and so checks the CRC result against the value "0001110100001111". The SCC presets the CRC checker whenever the receiver is in Hunt mode or whenever a flag is received so a CRC reset command is not strictly necessary. However, the CRC checker may be preset by issuing the Reset CRC Checker command in WR0. The CRC checker is automatically enabled for all data between the opening and closing flags by the SCC in SDLC mode, and the Rx CRC Enable bit ($D_3$) in WR3 is ignored. The result of the CRC calculation for the entire frame is valid in RR1 only when accompanied by the End of Frame bit being set in RR1. At all other times the CRC Error bit in RR1 should be ignored by the processor. Care must be exercised so that the processor does not attempt



07513A 5-7

Figure 5-7 SYNC as an Output

Table 5-2 Residue Codes

| Residue Code | | | Bits in Previous Byte | | | | Bits in Second Previous Byte | | | | Bits in Third Previous Byte | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 1 | 0 | 8B/C | 7B/C | 6B/C | 5B/C | 8B/C | 7B/C | 6B/C | 5B/C | 8B/C | 7B/C | 6B/C | 5B/C |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 8 | 7 | 5 | 2 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 4 | 2 | 0 | 0 | 8 | 7 | 6 | 3 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 5 | 3 | 1 | 0 | 8 | 7 | 6 | 4 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 6 | 4 | 2 | 0 | 8 | 7 | 6 | 5 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 7 | 5 | 3 | 1 | 8 | 7 | 6 | 5 |
| 0 | 1 | 1 | 0 | 0 | 0 | - | 8 | 6 | 4 | - | 8 | 7 | 6 | - |
| 1 | 1 | 1 | 1 | 0 | - | - | 8 | 7 | - | - | 8 | 7 | - | - |
| 0 | 0 | 0 | 2 | - | - | - | 8 | - | - | - | 8 | - | - | - |

to use the CRC bytes that are transferred as data because not all of the bits are transferred properly. The last two bits of CRC are never transferred to the receive data FIFO and are not recoverable.

A frame is terminated by a closing flag. When the SCC recognizes this flag the contents of the Receive Shift register are transferred to the receive data FIFO, the Residue Code is latched, the CRC Error bit is latched in the status FIFO and the End Of Frame bit is set in the receive status FIFO. The End Of Frame bit, upon reaching the top of the FIFO, will cause a special receive condition. The processor may then read RR1 to determine the result of the CRC calculation as well as the Residue Code. If either the Rx Interrupt on Special Condition Only or the Rx Interrupt on First Character or Special Condition modes are selected, the processor must issue an Error Reset command in WR0 to unlock the receive FIFO.

In addition to searching the data stream for flags, the receiver in the SCC also watches for seven consecutive "1s", which is the abort condition. The presence of seven consecutive "1s" is reported in the Break/Abort bit in RR0. This is one of the possible external/status interrupts, so transitions of this status may be programmed to cause interrupts. Upon receipt of an abort the receiver is forced into Hunt mode, where it looks for flags. The Hunt status is also a possible external/status condition whose transition may be programmed to cause an interrupt. The transitions of these two bits occur very close together but either one or two external/status interrupts may result. The abort condition is terminated when a "0" is received, either by itself or as the leading "0" of a flag. The receiver does not leave Hunt mode until a flag has been received so two discrete external/status conditions will occur at the end of an abort. An abort received in the middle of a frame terminates the frame reception, but no in an orderly manner, because the character being assembled is lost.

Up to two modem control signals associated with the receiver are available in SDLC mode. The DTR/$\overline{\text{REQ}}$ pin carries the inverted state of the DTR bit ($D_7$) in WR5 unless this pin has been programmed to carry a DMA Request signal. The DCD pin is ordinarily a simple input to the DCD bit in RR0. However, if the Auto Enables mode is selected by setting bit D5 of WR3 to "1", this pin becomes an enable for the receiver. That is, if Auto Enable is on and the $\overline{\text{DCD}}$ pin is HIGH the receiver is disabled. While the $\overline{\text{DCD}}$ pin is LOW, the receiver is enabled

The initialization sequence for the receiver in SDLC mode is: WR4 first, to select the mode, then WR10 to modify it if necessary, WR6 to program the address, WR7 to program the flag and the WR3 and WR5 to select the various options. At this point the other registers should be initialized as

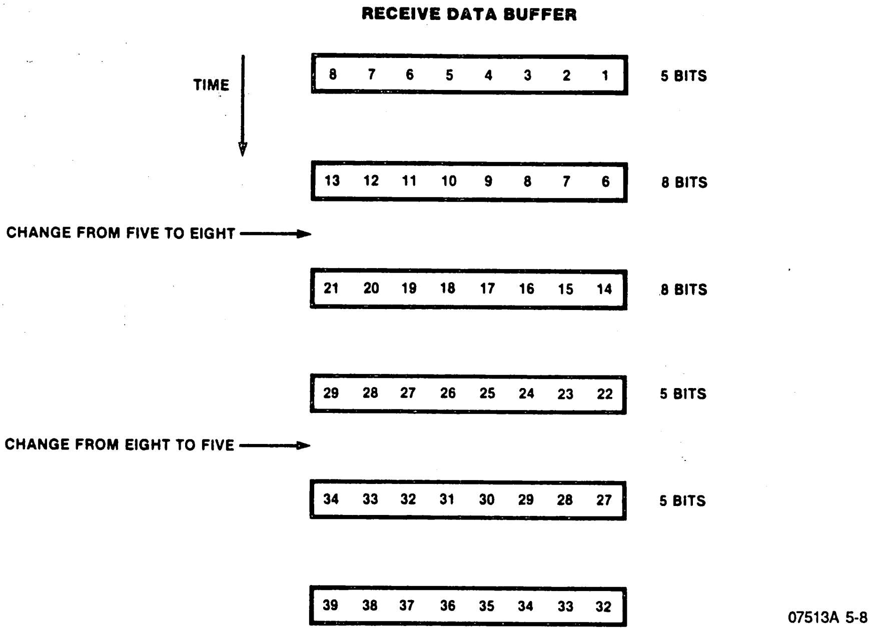


**Figure 5-8 Changing Character Length**

necessary. When all of this is complete the receiver may be enabled by setting bit $D_6$ of WR3 to "1".

### 5.3.2 SDLC Transmit

Once SDLC mode has been selected, the flag must be written in WR7, to be used to open and close the transmitted frames. The SCC does not automatically send the address byte; it merely encapsulates the data supplied by the processor with flags and CRC. Ordinarily, a frame will be terminated by the SCC with CRC and a flag but the SCC may be programmed to send an abort and a flag in place of the CRC. This option allows the SCC to abort a frame transmission in progress if the transmitter is accidentally allowed to underrun. This is controlled by the Abort/Flag On Underrun bit ($D_2$) in WR10. When this bit is set to "1" the transmitter will send an abort and a flag in place of the CRC when an underrun occurs. The frame will be terminated normally, with CRC and a flag, if this bit is set to "0". The SCC is also able to send an abort by command of the processor. The Send Abort command, issued in WR0, will send eight consecutive "1s" and then the transmitter will idle. Since up to five consecutive "1s" may have been sent prior to the command being issued, a Send Abort will cause a sequence of from eight to thirteen "1s" to be transmitted. The Send Abort command also empties the transmit buffer register. The idle condition for the transmitter is continuous flags, but this is under program control. By setting the Mark/Flag Idle bit ($D_3$) in WR10 to "1", the transmitter will send continuous "1s" in place of the idle flags. Note that the closing flag will be transmitted correctly even if this mode is selected. The Mark/Flag Idle must be set to "0", allowing a flag to be transmitted, before data is written to the transmit buffer. Care must be exercised in doing this because the continuous "1s" are transmitted eight at a time, and all eight must leave the Transmit Shift register, so that a flag may be loaded into it before the first data is written to the transmit buffer. When using the transmitter in SDLC mode, recall that all data passes through the zero inserter, which adds an extra five bit times of delay between the Transmit Shift register and the Transmit Data pin.

The number of bits per transmitted character is controlled by bits $D_6$ and $D_5$ of WR5 and the way the data is formatted within the transmit buffer. The bits in WR5 allow the option of five, six, seven, or eight bits per character. When five bits per character is selected, the data may be formatted before being written to the transmit buffer, to allow transmission of one to five bits per character. This formatting is shown in Table 5-1. In all cases the data must be right-justified, with the unused bits being ignored, except in the case of five bits per character.

An additional bit, carrying parity information, may be automatically appended to every transmitted character by setting bit $D_6$ of WR4 to "1". This bit is sent in addition to the number of bits specified in WR4 or by the data format. The parity sense is selected by bit $D_1$ of WR4. Parity is not normally used in SDLC mode. The character length may be changed on the fly, but the desired length must be selected before the character is loaded into the transmit shift register from the transmit buffer. The easiest way to ensure this is to write to WR5 to change the character length before writing the data to the transmit buffer.

Only the CRC-CCITT polynomial may be used in SDLC mode. This is selected by setting bit $D_2$ in WR5 to "0". This bit controls the selection for both the transmitter and receiver. The initial state of the generator and checker is controlled by bit $D_7$ of WR10. When this bit is set to "1", both the generator, and checker sill have an initial value of all "1s" and, if this bit is set to "0", the initial values will be all "0s". The SCC does not automatically preset the CRC generator so this must be done in software. This is accomplished by issuing the Reset Tx CRC generator command, which is encoded in bits $D_7$ and $D_6$ of WR0. For proper results, this command must be issued while the transmitter is enabled and idling. If CRC is to be used the transmit CRC generator must be enabled by setting bit D0 of WR5 to "1". CRC is normally calculated on all characters between opening and closing flags, so this bit is usually set to "1" at initialization and never changed.

Enabling the CRC generator is not sufficient to control the transmission of CRC. In the SCC this function is controlled by the Tx Underrun/EOM bit, which may be reset by the processor and set by the SCC. When the transmitter underruns (both the transmit buffer and transmit shift register are empty) the state of the Tx Underrun EOM bit determines the action taken by the SCC. If the Tx Underrun/EOM bit is set to "1" when the underrun occurs, the transmitter will send flags; if this bit is reset to "0" when the underrun occurs, the transmitter will send either the accumulated CRC followed by flags, or an abort followed by flags, depending the the state of the Abort/Flag on Underrun bit in the WR10. When the CRC or abort is loaded into the Transmit Shift register for transmission, the SCC will set the Tx Underrun/EOM bit to indicate this. This transition may be programmed to cause an external/status interrupt, or the Tx Underrun/EOM bit is available in RR0. The Reset Tx Underrun/EOM Latch command is encoded in bits $D_7$ and $D_6$ of WR0.

For correct transmission of the CRC at the end of a frame, this command must be issued after the first character is written to the SCC but before the transmitter underruns after the last character written to the SCC. The command is usually issued immediately after the first character is written to the SCC so that the abort or CRC is sent if an underrun occurs inadvertently. The Abort/$\overline{\text{Flag}}$ on Underrun bit (D2) in WR10 is usually set to "1" at the same time as the Tx Underrun/EOM bit is reset so that an abort sill be sent if the transmitter underruns. The bit is then set to "0" near the end of the frame to allow the correct transmission of CRC.

In this paragraph the term "completely sent" means shifted out of the Transmit Shift register, not shifted out of the zero inserter, which is an additional five bit times of delay. In SDLC mode, if the transmitter is disabled during transmission of a character, that character will be "completely sent". This applies to both data and flags. However, if the transmitter is disabled during the transmission of CRC, the 16-bit transmission will be completed but the remaining bits will be from the Flag register rather than the remainder of the CRC.

There are two modem control signals associated with the transmitter provided by the SCC. The $\overline{\text{RTS}}$ pin is a simple output that carries the inverted state of the RTS bit ($D_1$) in WR5. The $\overline{\text{CTS}}$ pin is ordinarily a simple input to the $\overline{\text{CTS}}$ bit in RR0. However, if Auto Enables mode is selected, this pin becomes an enable for the transmitter. That is, if Auto Enables is on and the $\overline{\text{CTS}}$ pin is HIGH the transmitter is disabled. If the $\overline{\text{CTS}}$ pin is LOW, the transmitter is enabled.

The initialization sequence for the transmitter in SDLC mode is: WR4 first, to select the mode, then WR10 to modify it if necessary, WR7 to program the flag, and then WR3 and WR5 to select the various options. At this point the other registers should be initialized as necessary. When all of this is complete, the transmitter may be enabled by setting bit $D_3$ of WR5 to "1". Now that the transmitter is enabled, the CRC generator may be initialized by issuing the Reset Tx CRC Generator command in WR0.

### 5.3.3 SDLC Loop Mode

SDLC Loop mode is quite similar to SDLC mode except that two additional control bits are used. They are the Loop Mode bit ($D_1$) and the Go Active on Poll bit ($D_4$) in WR10. In addition to these two extra control bits, there are also two status bits in RR10. They are the On Loop bit ($D_1$) and the Loop Sending bit ($D_4$). Before Loop mode is selected both the receiver and transmitter must be completely initialized for SDLC operation. Once this is done, Loop mode is selected by setting bit $D_1$ of WR10 to "1". At this point the SCC connects TxD to RxD with only gate delays in the path. At the same time a flag is loaded into the Transmit Shift register, and is shifted to the end of the zero inserter, ready for transmission. The SCC will remain in this state until the Go Active on Poll bit ($D_4$) in WR10 is set to "1". When this bit is set to "1" the receiver begins looking for a sequence of seven consecutive "1s", indicating either an EOP or an idle line. When the receiver detects this condition the Break/Abort bit in RR0 is set to "1" and a one-bit time delay is inserted in the path from RxD to TxD. The On Loop bit in RR10 is also set to "1" at this time, and the receiver enters the Hunt mode. The SCC cannot transmit on the loop until a flag is received, causing the receiver to leave Hunt mode, and another EOP (bit pattern "11111110" ) is received. The SCC is now on the loop and capable of transmitting on the loop. As soon as this status is recognized by the processor, the Go Active On Poll bit in WR10 should be set to "0" to prevent the SCC from transmitting on the loop without the consent of the processor.

To transmit a message on the loop, the Go Active On Poll bit in WR10 must be set to "1". Once this is done, the SCC will change the next received EOP into a Flag and begin transmitting on the loop. When the EOP is received, the Break/Abort and Hunt bits in RR0 will be set to "1", and the Loop Sending bit in RR10 will also be set to "1". Data to be transmitted may be written after the Go Active On Poll bit has been set or after the receiver enters Hunt mode. If the data is written immediately after the Go Active On Poll bit has been set, the SCC will only insert one flag after the EOP is changed into a flag. If the data is not written until after the receiver enters the Hunt mode, flags will be transmitted until the data is written. If only one frame is to be transmitted on the loop in response to an EOP, the processor must set the Go Active on Poll bit to "0" before the last data is written to the transmitter. In this case the transmitter will close the frame with a single flag, and then revert to the one-bit delay. The Loop Sending bit in RR10 is set to "0" when the closing Flag has been sent. If more than one frame is to be transmitted, the Go Active On Poll bit should not be set to "0" until the last frame is being sent. If this bit is not set to "0" before the end of a frame, the trans-mitter will send Flags until either more data is written to the transmitter, or until the Go Active On Poll bit is set to "0". Note that the state of the Abort/ $\overline{\text{Flag}}$ on Underrun and Mark/$\overline{\text{Flag}}$ Idle bits in WR10 are ignored by the SCC in SDLC Loop mode.

To go off the loop in an orderly manner requires actions similar to those taken to go on the loop. First, the Go Active On Poll bit must be set to "0" and any transmission in progress completed, if the SCC is currently sending on the loop. Once the SCC is not sending on the loop, an exit from the loop is accomplished by setting the Loop Mode bit in WR10 to "0", and at the same time writing the Abort/$\overline{\text{Flag}}$ on Underrun and Mark/$\overline{\text{Flag}}$ Idle bits with the desired values. The SCC will revert to normal SDLC operation as soon as an EOP is received, or immediately, if the receiver is already in Hunt mode because of the receipt of an EOP.

The initialization sequence for the SCC in SDLC Loop mode is similar to the sequence used in SDLC mode, except that it is somewhat longer. The processor should program WR4 first, to select SDLC mode, and then WR10 to select the CRC preset value and program the Mark/Flag Idle bit. The Loop Mode and Go Active On Poll bits in WR10 should not be set to "1" yet. The flag is written in WR7 and the various options are selected in WR3 and WR5. At this point the other registers should be initialized as necessary, then the Loop Mode bit ($D_1$) in WR10 should be set to "1". When all of this is complete the transmitter may be enabled by setting bit $D_3$ of WR5 to "1". Now that the transmitter is enabled, the CRC generator may be initialized by issuing the Reset Tx CRC Generator command in WR0. The receiver is enabled by setting the Go Active on Poll bit ($D_4$) in WR10 to "1". The SCC will go on the loop when seven consecutive "1s" are received, and will signal this by setting the On Loop bit in RR10. Note that the seven consecutive "1s" will set the Break/Abort and Hunt bits in RR0 also. Once the SCC is on the loop, the Go Active on Poll bit should be set to "0" until a message is to be transmitted on the loop. To transmit a message on the loop, the Go Active on Poll bit should be set to "1". At this point the processor may either write the first character to the transmit buffer and wait for a transmit buffer empty condition, or wait for the Break/Abort and Hunt bits to be set in RR10 and the Loop Sending bit to be set in RR10 before writing the first data to the transmitter. The Go Active On Poll bit should be set to "0" after the transmission of the frame has begun. To go off of the loop, the processor should set the Go Active On Poll bit in WR10 to "0" and then wait for the Loop Sending bit in RR10 to be set to "0". At this point the Loop Mode bit ($D_1$) in WR10 is set to "0" to request an orderly exit from the loop. The SCC will exit SDLC Loop mode when seven consecutive "1s" have been received; at the same time the Break/Abort and Hunt bits in RR0 will be set to "1", and the On Loop bit in RR10 will be set to "0".

CHAPTER 6

# SUPPORT CIRCUITRY PROGRAMMING

## 6.0 INTRODUCTION

The SCC incorporates additional circuitry to aid in serial communications. This circuitry includes clocking options, baud rate generator, data encoding, and internal loopback. This chapter discusses how to program these functions.

## 6.1 CLOCK OPTIONS

The SCC may be programmed to select one of several sources to provide the transmit and receive clocks. In addition, the SCC requires a fundamental, parallel resonant crystal oscillator in each chanel, as well as the ability to echo one of several internal clock sources to the outside world. These options are controlled by the bits in WR11. For further details on the crystal, refer to the Zilog Application Note, *Design Considerations Using Quartz Crystals With Zilog's Components*.

The crystal oscillator option is controlled by bit D$_7$ in WR11. When this is set to "0", the crystal oscillator is disabled and all pins function normally. When this bit is set to "1" the crystal oscillator is enabled and a high-gain amplifier is connected between the $\overline{\text{RTxC}}$ pin and the $\overline{\text{SYNC}}$ pin. While the crystal oscillator is enabled, anything that has $\overline{\text{RTxC}}$ selected as its clock source will automatically be connected to the output of the crystal oscillator. While the crystal oscillator is enabled, the $\overline{\text{SYNC}}$ pin is obviously unavailable for other use. In synchronous modes no sync pulse is output, and the External Sync mode cannot be selected. In asynchronous modes the state of the Sync/Hunt bit in RR0 is no longer controlled by the $\overline{\text{SYNC}}$ pin. Instead, the Sync/Hunt bit is forced to "0". The crystal oscillator requires some finite time to stabilize. The oscillator must be allowed to stabilize before it is used as a clock source.

The source of the receive clock is controlled by bits D$_6$ and D$_5$ of WR11. The receive clock may be programmed to come from the $\overline{\text{RTxC}}$ pin, the $\overline{\text{TRxC}}$ pin, the output of the baud rate generator, or the transmit output of the DPLL.

The source of the transmit clock is controlled by bits D$_4$ and D$_3$ of WR11. The transmit clock may be programmed to come from the $\overline{\text{RTxC}}$ pin, the $\overline{\text{TRxC}}$ pin, the output of the baud rate generator, or the transmit output of the DPLL.

Ordinarily the TRxC pin is an input, but it becomes on output if this pin has not been selected as the source for the transmitter or the receiver, and bit D$_2$ of WR11 is set to "1". The selection of the signal provided on the $\overline{\text{TRxC}}$ output pin is controlled by bits D$_1$ and D0 of WR11. The $\overline{\text{TRxC}}$ pin may be programmed to provide the output of the crystal oscillator, the output of the baud rate generator, the receive output of the DPLL or the actual transmit clock. If the output of the crystal oscillator is selected but the crystal oscillator has not been enabled the $\overline{\text{TRxC}}$ pin will be driven HIGH. The option of placing the transmit clock signal on the $\overline{\text{TRxC}}$ pin when it is an output allows access to the transmit output of the DPLL.

Figure 6-1 shows a simplified schematic diagram of the circuitry used in the clock multiplexing. It shows the inputs to the multiplexer section as well as the various signal inversions that occur in the paths to the outputs. Also shown are the edges used by the receiver, transmitter, baud rate generator and DPLL to sample or send data or otherwise change state. For example, the receiver samples data on the falling edge, but since there is an inversion in the clock path between the $\overline{\text{RTxC}}$ pin and the receiver, a rising edge of the $\overline{\text{RTxC}}$ pin samples the data for the receiver.

Selection of the clocking options may be done anywhere in the initialization sequence, but the final values must be selected before the receiver, transmitter, baud rate generator, or DPLL are enabled to prevent problems from arbitrarily narrow clock signals out of the multiplexers. The same is true of the crystal oscillator, in that the output should be allowed to stabilize before it is used as a clock source.

## 6.2 BAUD RATE GENERATOR

Figure 6-2 shows a block diagram of the baud rate generator. It consists of a 16-bit down-counter, two 8-bit time constant registers and an output divide-by-two. The baud rate generator input comes from the output of a two-input multiplexer, the zero count condition is output to the External/Status Interrupt Section. The baud rate generator may be enabled and disabled by command and is disabled by a hardware reset.

The time constant for the baud rate generator is programmed in WR12 and WR13, with the least-significant byte in WR12. The formulas relating the baud rate to the time constant and vice versa are shown in Table 6-1 with an example. In these formulas the baud rate generator clock frequency is in Hertz, the desired baud rate in bits/second and the time constant is dimensionless. The example in Table 6-2 assumes a 2.4576 MHz clock factor of 16 and shows the time constant for a number of popular baud rates.

**Table 6-1. Time Constant Formulas**

$$\text{Time Constant} = \frac{\text{Clock Frequency}}{2 \cdot (\text{Clock Mode}) \cdot (\text{Baud Rate})} - 2$$

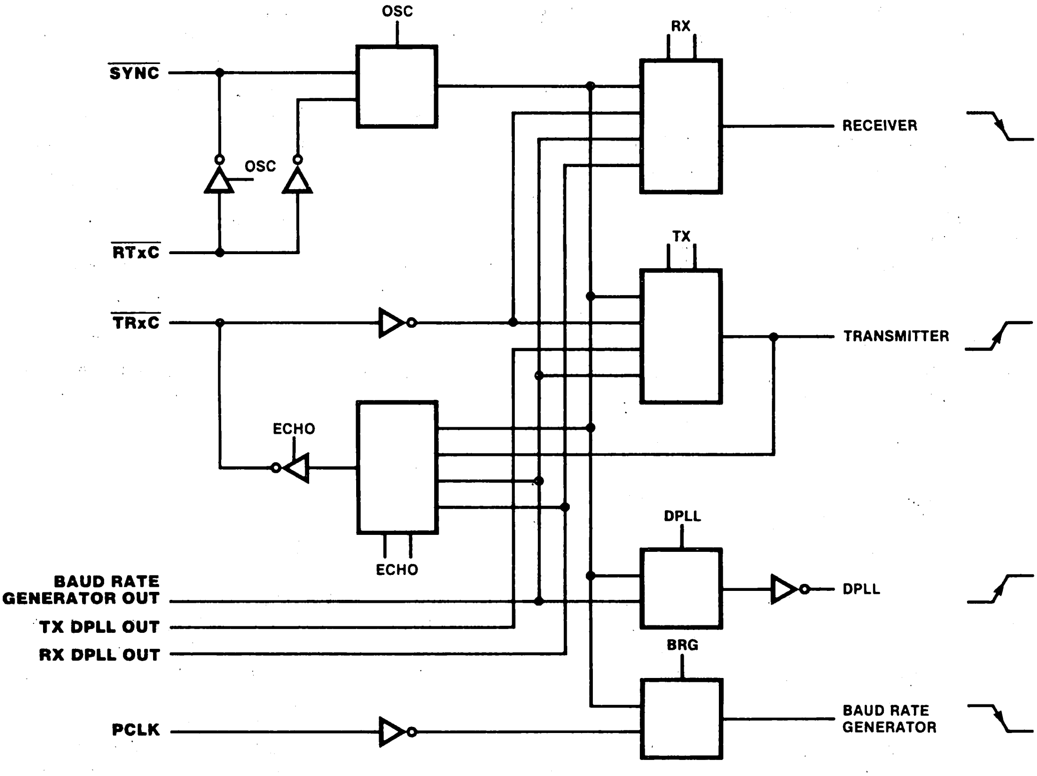
$$\text{Baud Rate} = \frac{\text{Clock Frequency}}{2 \cdot (\text{Clock Mode}) \cdot (\text{Time Constant} + 2)}$$

**Table 6-2 Baud Rate Example**

| Baud Rate | Divider | |
|---|---|---|
| | Decimal | Hex |
| 38400 | 0 | 0000H |
| 19200 | 2 | 0002H |
| 9600 | 6 | 0006H |
| 4800 | 14 | 000EH |
| 2400 | 30 | 001EH |
| 1200 | 62 | 003EH |
| 600 | 126 | 007EH |
| 300 | 254 | 00FEH |
| 150 | 510 | 01FEH |

For 2.4576 MHz Clock, X16 Mode

The clock source for the baud rate generator is selected by bit $D_1$ of WR14. When this bit is set to "0" the baud rate generator uses the signal on the TxC pin as its clock, independent of whether the



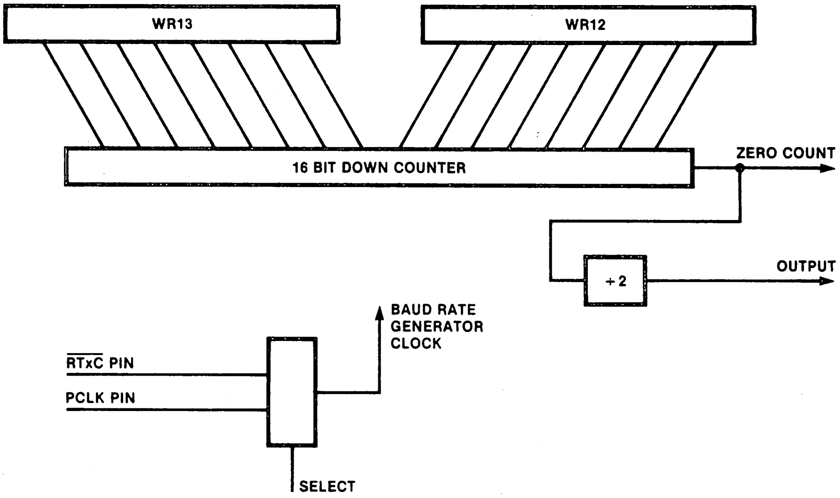
07513A 6-1

Figure 6-1 Clock Multiplexer

$\overline{\text{TxC}}$ pin is a simple input or part of the crystal oscillator circuit. When this bit is set to "1" the baud rate generator is clocked by PCLK. To avoid metastable problems in the counter, this bit should be changed only while the baud rate generator is disabled, since arbitrarily narrow pulses can be generated at the output of the multiplexer when it changes status.

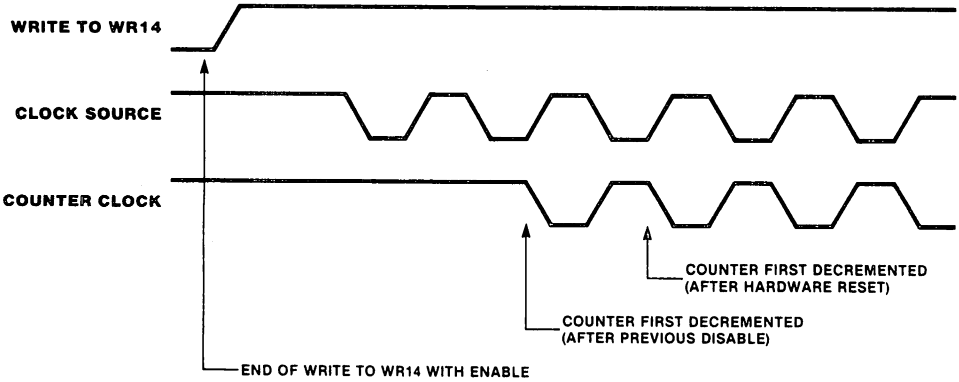
The baud rate generator is enabled while bit $D_0$ of WR14 is set to "1" and is disabled while this bit is set to "0". To prevent metastable problems when the baud rate generator is first enabled, the enable bit is synchronized to the baud rate generator clock. This introduces an additional delay then the baud rate generator is first enabled and this is shown in Figure 6-3. The baud rate generator is disabled immediately when bit $D_0$ of WR14 is set to "0", because the delay is only necessary on startup. The baud rate generator may be enabled and disabled on the fly, but this delay on startup must be taken into consideration.

Upon reaching a count of "0" the time constant



07513A 6-2

Figure 6-2 Baud Rate Generator


WRITE TO WR14
CLOCK SOURCE
COUNTER CLOCK
COUNTER FIRST DECREMENTED (AFTER HARDWARE RESET)
COUNTER FIRST DECREMENTED (AFTER PREVIOUS DISABLE)
END OF WRITE TO WR14 WITH ENABLE

07513A 6-3

Figure 6-3 Baud Rate Generator Start-Up

held in WR12 and WR13 is reloaded into the downcounter so that the process of counting down may start over. In addition to reloading the time constant, the output of the baud rate generator toggles, and for the clock cycle with a zero count, the zero count signal goes active to the External/Status Section. This zero count condition from the baud rate generator does not persist, so if it is to be used by the processor, it should be latched in the External/Status latch. While the baud rate generator is disabled the state of the zero count signal is held. This signal is forced active by a hardware reset.

Initializing the baud rate generator is done in four steps. First, the time constant is determined and loaded into WR12 and WR13. Next, the processor must select the clock source for the baud rate generator by writing to bit $D_1$ of WR14. Finally, the baud rate generator is enabled by setting bit $D_0$ of WR14 to "1". Note that the first write to WR14 is not necessary after a hardware reset if the clock source is to be the $\overline{\text{RTxC}}$ pin. This is because a hardware reset automatically selects the $\overline{\text{RTxC}}$ pin as the baud rate generator clock source.

## 6.3 DATA ENCODING

The SCC provides four different data encoding methods, selected by bits $D_6$ and $D_5$ in WR10. An example of these four encoding methods is shown in Figure 6-4. Any encoding method may be used in any X1 mode in the SCC, asynchronous or synchronous. The data encoding selected is active even though the transmitter or receiver may be idling or disabled.

In NRZ encoding a "1" is represented by a HIGH level and a "0" is represented by a LOW level. In this encoding method only a minimal amount of clocking information is available in the data stream in the form of transitions on bit-cell boundaries. In an arbitrary data pattern this may not be sufficient to generate a clock for the data from the data itself.

In NRZI encoding a "1" is represented by no change in the level and a "0" is represented by a change in the level. As in NRZ only a minimal amount of clocking information is available in the data stream, in the form of transitions on bit cell boundaries. In an arbitrary data pattern this may not be sufficient to generate a clock for the data from the data itself. In the case of SDLC, where the number of consecutive "1s" in the data stream is limited, a minimum number of transitions to generate a clock are guaranteed.

In FM1 encoding, also known as biphase mark, a transition is present on every bit cell boundary, and an addition transition may be present in the middle of the bit cell. In FM1 a "0" is sent as no transition in the center of the bit cell and a "1" is sent as a transition in the center of the bit cell. FM1 encoded data contains sufficient information to recover a clock from the data.

In FM0 encoding, also known as biphase space, a transition is present on every bit cell boundary and an additional transition may be present in the

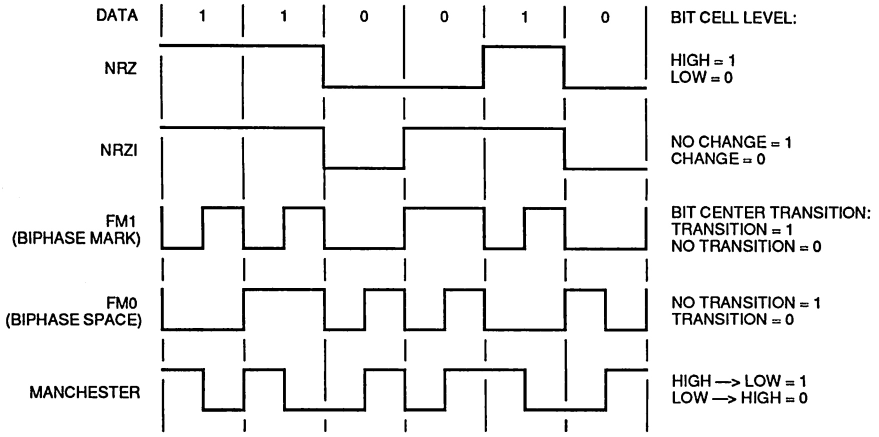


**Figure 6-4 Data Encoding Methods**

middle of the bit cell. In FM0 a "1" is sent as no transition in the center of the bit cell and a "0" is sent as a transition in the center of the bit cell. FM0 encoded data contains sufficient information to recover a clock from the data.

The data encoding method should be selected in the initialization procedure before the transmitter and receiver are enabled but no other restrictions apply. Note, in Figure 6-4, that in NRZ and NRZI the receiver samples the data only on one edge. However, in FM1 and FM0 the receiver samples the data on both edges. Also, as shown in Figure 6-4, the transmitter defines bit cell boundaries by one edge in all cases and uses the other edge in FM1 and FM0 to create the mid-bit transition.

## 6.4 DIGITAL PHASE-LOCKED LOOP

Figure 6-5 shows a block diagram of the digital phase-locked loop. It consists of a 5-bit counter, an edge detector, and a pair of output decoders. The clock for the DPLL comes from the output of a two-input multiplexer, and the two outputs go to the transmitter and receive clock multiplexers. The DPLL is controlled by the seven commands that are encoded in bits $D_7$, $D_6$, and $D_5$ of WR14.

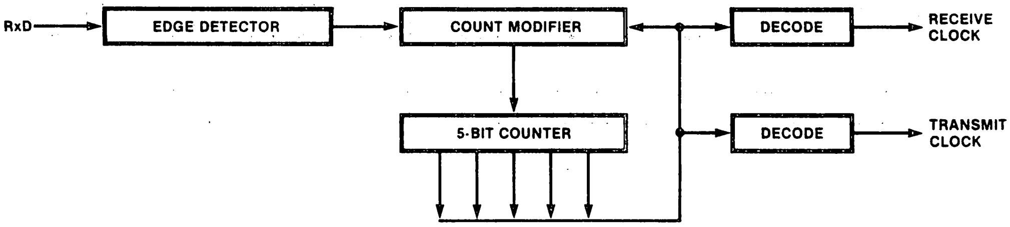
The clock for the DPLL is selected by two of the commands in WR14. One command selects the output of the baud rate generator as the clock source, and the other command selects the $\overline{\text{RTxC}}$ pin as the clock source, independent of whether the $\overline{\text{RTxC}}$ pin is a simple input or part of the crystal oscillator circuit. To avoid metastable problems in the counter, the clock source selection should be made only while DPLL is disabled, since arbitrarily narrow pulses can be generated at the output of the multiplexer when it changes status.

The DPLL is enabled by issuing the Enter Search Mode command in WR14. This command is also used to reset the DPLL to a known state if it is suspected that synchronization has been lost. When used to enable the DPLL, the Enter Search Mode command unlocks the counter, which is held while the DPLL is disabled, and enables the edge detector. If the DPLL is already enabled when this command is issued, the DPLL also enters Search Mode. While in Search mode, the counter is held at a specific count and no outputs are provided. The DPLL remains in this status until an edge is detected in the receive data stream. This first edge is assumed to occur on a bit cell boundary, and the DPLL will begin providing an output to the receiver that will properly sample the data. From this point on the DPLL will adjust its output to remain in phase with the receive data. If the first edge that the DPLL sees does not occur on a bit cell boundary, the DPLL will eventually lock on to the receive data but it will take longer to do so.

The DPLL may be programmed to operate in either of two modes, as selected by command in WR14. In the NRZI mode the DPLL clock must be 32 times the date rate. In this mode the transmit and receive clock outputs of the DPLL are identical, and the clocks are phased so that the receiver samples the data in the middle of the bit cell. In NRZI mode the DPLL does not require a transition in every bit cell, so this mode is useful for recovering the clocking information from NRZ and NRZI data streams. In the FM mode the DPLL clock must be 16 times the data rate. In this mode the transmit clock output of the DPLL lags the receive clock outputs by 90o, to make the transmit and receive bit cell boundaries the same, because the receiver must sample FM data at one-quarter and three-quarters bit time. In FM mode the DPLL requires a transition in every bit cell, and if this transition is not present in two consecutively sampled bit cells, the DPLL will automatically enter the search mode. As in the case of the clock source selection, the mode of operation should only be changed while the DPLL is disabled to prevent unpredictable results.

## 6.5 NRZI MODE OPERATION

To operate in NRZI mode the DPLL must be supplied with a clock that is 32 times the data rate. The DPLL uses this clock, along with the receive data, to construct receive and transmit clock



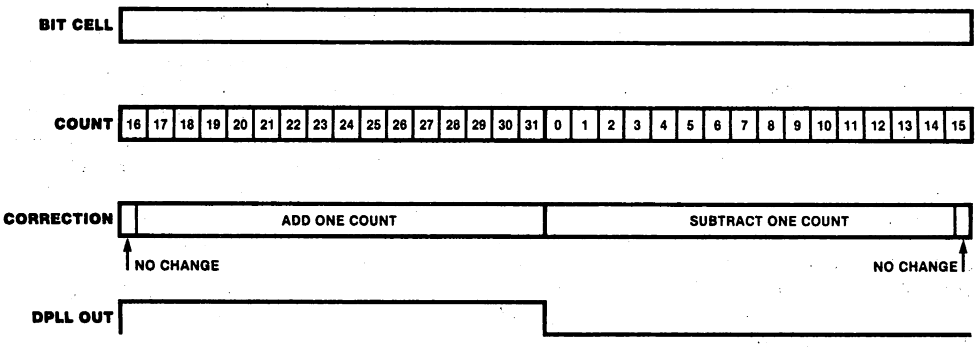
07513A 6-5

Figure 6-5 Digital Phase-Locked Loop

outputs that are phased to properly receive and transmit data. To do this, the DPLL divides each bit cell into four regions, and makes an adjustment to the count cycle of the 5-bit counter dependent upon in which region a transition on the receive data input occurred. This is shown in Figure 6-6. Ordinarily, a bit cell boundary will occur between count 15 and count 16, and the DPLL output will cause the data to be sampled in the middle of the bit cell. The DPLL actually allows the transition marking a bit cell boundary to occur anywhere during the second half of count 15 or the first half of count 16 without making a correction to its count cycle. However, if the transition marking a bit cell boundary occurs between the middle of count 16 and count 31 the DPLL is sampling the data too early in the bit cell. In response to this the DPLL extends its count by one during the next 0 to 31 counting cycle, which effectively moves the edge of the clock that samples the receive data closer to the center of the bit cell. In a similar manner, if the transition occurs between count 0 and the middle of count 15, the output of the DPLL is sampling the data too late in the bit cell. To correct this, the DPLL shortens its count by one during the next 0 to 31 counting cycle, which effectively moves the edge of the clock that samples the receive data closer to the center of the bit cell. In NRZI mode, if the DPLL does not see any transition during a counting cycle, no adjustment is made in the following counting cycle. If an adjustment to the counting cycle is necessary the DPLL modifies count five, either deleting it or doubling it. Thus only the LOW time of the DPLL output will be lengthened or shortened. While the DPLL is in search mode, the counter remains at count 16, where the DPLL outputs are both HIGH. The missing clock latches in the DPLL which may be accessed in RR10, are not used in NRZI mode. An example of the DPLL in operation is shown in Figure 6-7.

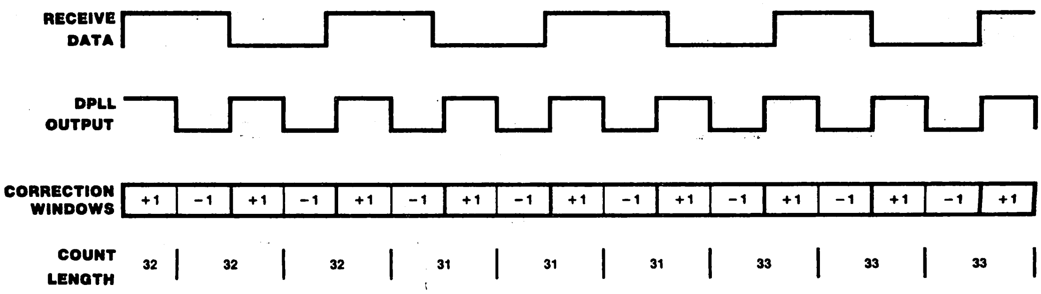
## 6.6 FM MODE OPERATION

To operate in FM mode the DPLL must be supplied with a clock that is 16 times the data rate. The DPLL uses this clock, along with the receive



07513A 6-6

**Figure 6-6 DPLL In NRZI Mode**



07513A 6-7

**Figure 6-7 DPLL Operating Example**

data, to construct receive and transmit clock outputs that are phased to receive and transmit data properly. In FM mode that the counter in the DPLL still counts from 0 to 31 but now each cycle corresponds to 2-bit cells. To make adjustments to remain in phase with the receive data, the DPLL divides a pair of bit cells into five regions, making the adjustment to the counter dependent upon which region the transition on the receive data input occurred. This is shown in Figure 6-8. Ordinarily a bit cell boundary will occur between count 15 or count 16, and the DPLL receive output will cause the data to be sampled at one-fourth and three-fourths of the way through the bit cell. The DPLL actually allows the transition marking a bit-cell boundary to occur anywhere during the second half of count 15 or the first half of count 16 without making a correction to its count cycle. However, if the transition marking a bit cell boundary occurs between the middle of count 16 and the middle of count 19 the DPLL is sampling the data too early in the bit cell. In response to this the DPLL extends its count by 1 during the next 0 to 31 counting cycle, which effectively moves the receive clock edges closer to where they should be. In FM mode any transitions occurring between the middle of count 19 in one cycle and the middle of count 12 during the next cycle are ignored by the DPLL. This is necessary to guarantee that any data transitions in the bit cells will not cause an adjustment to the counting cycle.

In FM mode the transmit clock and receive clock outputs from the DPLL are not in phase. This is necessary to make the transmit and receive bit cell boundaries coincide, since the receive clock must sample the data one-fourth and three-fourths of the way through the bit cell. As in NRZI mode, if an adjustment to the counting cycle is necessary, the DPLL modifies count 5, either deleting it or doubling it. If no adjustment is necessary, the count sequence proceeds normally. While the DPLL is in search mode, the counter remains at count 16, where the receive output is LOW and the transmit output is LOW. This fact can be used to provide a transmit clock under software control since the DPLL is in search mode while it it disabled. While the DPLL is disabled the transmit clock output of the DPLL may be toggled by alternately selecting FM and NRZI move in the DPLL. The same is true of the receive clock.

In addition to FM encoded data, the DPLL may also be used to recover the clock from Manchester encoded data, which contains a transition at the center of every bit cell. Here it is the direction of the transition that distinguishes a “1” from a “0”. Another way of looking at Manchester encoding is to realize that, during the first half of the bit cell the data is sent, during the second half of the bit cell the complement of the data is sent. This is shown in Figure 6-9, along with the DPLL output if it thinks that the mid-bit transitions are really bit cell boundaries. As is obvious from the figure, if the receiver samples the data on the falling edge of the DPLL receive clock output, the Manchester data will be properly decoded. This occurs if the receiver is programmed to accept NRZ data.

From the above discussion together with an examination of FM0 and FM1 data encoding, it should be obvious that only clock transitions should exist on the receive data pin when the DPLL is programmed to enter search mode. If this is not the case the DPLL may attempt to lock on to the data transitions. With FM0 encoding this

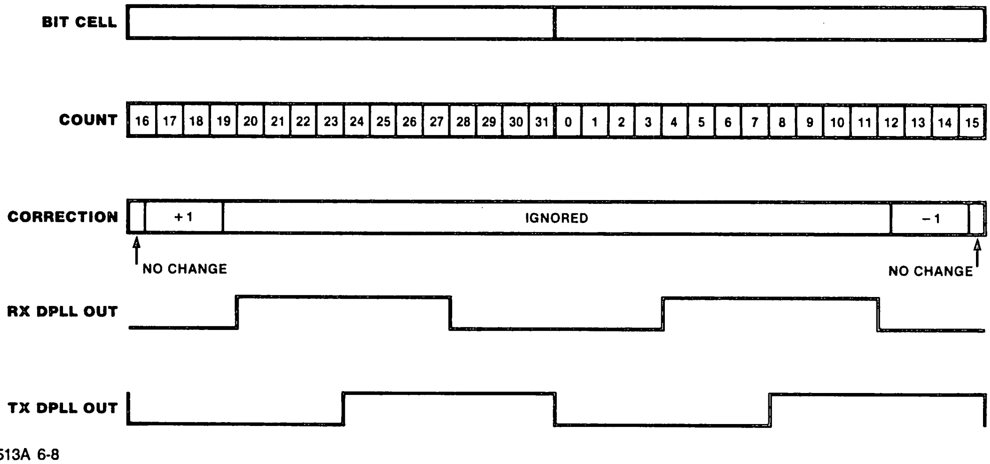


Figure 6-8 DPLL in FM Mode

requires continuous "1s" received when leaving search. In FM1 encoding it is continuous "0s;" with Manchester encoded data this means alternating "1s" and "0s". With all three of these data encoding methods there will always be at least one transition in every bit cell, and in FM mode the DPLL is designed to expect this transition. In particular, if no transition occurs between the middle of count 12 and the middle of count 19, the DPLL is probably not locked onto the data properly. When the DPLL misses an edge the One Clock Missing bit is RR10 is set to "1" and latched. It will hold this value until a Reset Mission Clock command is issued in WR14 or until the DPLL is disabled or programmed to enter the Search mode. Upon missing this one edge the DPLL takes no other action and does not modify its count during the next counting cycle. However, if the DPLL does not see an edge between the middle of count 12 and the middle of count 19 in two successive 0 to 31 count cycles, a line error condition is assumed. If this occurs, the two Clocks Mission bit in RR10 is set to "1" and latched. At the same time the DPLL enters the Search mode. The DPLL makes the decision to enter Search mode during count 2, where both the receive clock and transmit clock outputs are LOW. This prevents any glitches on the clock outputs when search mode is entered. While in search mode no clock outputs are provided by the DPLL. The Two Clocks Missing bit in RR10 is latched until a Reset Missing Clock command is issued in WR14, or until the DPLL is disabled or programmed to enter the Search mode.

### 6.7 DPLL INITIALIZATION

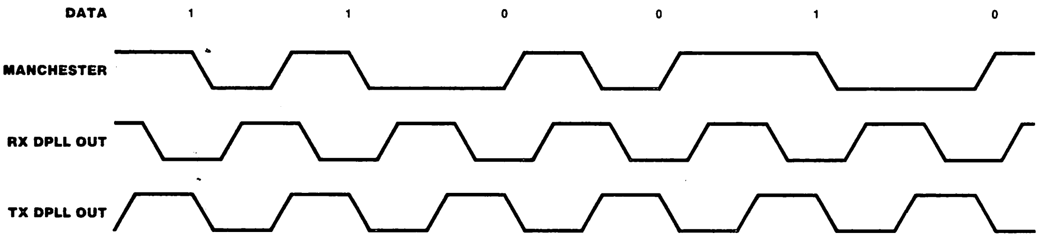
Initialization of the DPLL may be done at any time during the initialization sequence, but should probably be done after the clock modes have been selected in WR11, and before the receiver and transmitter are enabled. When initializing the DPLL the clock source should be selected first, followed by the selection of the operating mode. At this point the DPLL, may be enabled by issuing the Enter Search Mode command in WR14. Note that a channel or hardware reset disables the DPLL, selects the $\overline{RTxC}$ pin as the clock source for the DPLL, and places it in the NRZI mode.

### 6.8 INTERNAL LOOPBACK/AUTO ECHO

The SCC contains two other features useful for diagnostic purposes, controlled by bits in WR14. They are local loopback and auto echo.

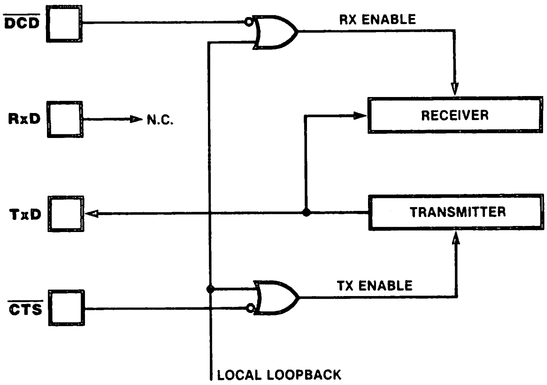
Local loopback is selected when bit D4 of WR14 is set to "1". In this mode the output of the transmitter is internally connected to the input of the receiver. At the same time the TxD pin remains connected to the transmitter. In this mode the $\overline{DCD}$ pin is ignored as a receive enable and the $\overline{CTS}$ pin is ignored as a transmitter enable even if the Auto Enables mode has been selected. Note that the DPLL input is connected to the RxD pin, not to the input of the receiver. This precludes the use of the DPLL in local loopback.

Auto echo is selected when bit D3 of WR14 is set to "1". In this mode the TxD pin is connected directly to the RxD pin, and the receiver input is connected to the RxD pin. In this mode the $\overline{CTS}$ pin is ignored as a transmitter enable and the output of the transmitter does not connect to anything. If both the Local Loopback and Auto Echo bits are set to "1", the auto echo mode will be selected, but both the $\overline{CTS}$ pin and $\overline{DCD}$ pin will be ignored as auto enables. This, however, should not be considered a normal operating mode, however. Local Loopback is shown schematically in Figure 6-10 and auto echo is shown schematically in Figure 6-11.



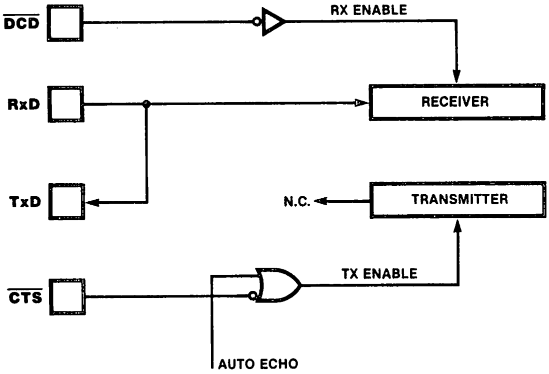
07513A 6-9

Figure 6-9 Manchester Clock Recovery



07513A 6-10

Figure 6-10 Local Loopback



07513A 6-11

Figure 6-11 Auto Echo

CHAPTER 7

# REGISTER DESCRIPTION

## 7.0 INTRODUCTION

The following sections describe the SCC registers. Each register is detailed in terms of bit configuration, the active states (See Table 7-1) of each bit, their definitions, their functions, and their effects upon the internal hardware and external pins.

**Table 7-1 SCC Register Description**

| Read Register | Description |
|---|---|
| RR0 | Xmit/Receive Buffer Status and Ext Status |
| RR1 | Receive Condition Status/Residue Codes |
| RR2 | Interrupt Vector (modified in B Channel) |
| RR3 | Interrupt Pending (Channel A only) |
| RR8 | Receive Buffer |
| RR10 | Loop/Clock Status |
| RR12 | Lower Byte of Time Constant |
| RR13 | Upper Byte of Time Constant |
| RR15 | External Status Interrupt Enable |

| Write Register | Description |
|---|---|
| WR0 | Command Register |
| WR1 | Tx/Rx Interrupt and Data Xfer Mode Definition |
| WR2 | Interrupt Vector |
| WR3 | Receive Parameters and Control |
| WR4 | Tx/Rx Miscellaneous Parameters and Modes |
| WR5 | Transmit Parameter and Controls |
| WR6 | Sync Character or SDLC Address Field |
| WR7 | Sync Character or SDLC Flag |
| WR8 | Transmit Buffer |
| WR9 | Master Interrupt Control |
| WR10 | Misc Transmitter/Receiver Control Bits |
| WR11 | Clock Mode Control |
| WR12 | Lower Byte of Baud Rate Generator Time Constant |
| WR13 | Upper Byte of Baud Rate Generator Time Constant |
| WR14 | Miscellaneous Control Bits |
| WR15 | External Status/Interrupt Control |

## 7.1 WRITE REGISTERS

The SCC write register set in each channel includes ten control registers (among them is the transmit buffer), two sync character registers, and two baud rate time constant registers. The interrupt control register and the master interrupt control and reset register are shared by both channels.

The only difference in register definition between the Z8030 and Z8530 versions of the SCC exists in the command decode structure. The following sections describe in detail each write register and the associated bit configuration for each.

### 7.1.1 Write Register 0 (Command Register)

WR0 is the command register and the CRC reset code register. WR0 in the AmZ8030 version varies slightly from that in the AmZ8530 version. Figure 7-1 shows the bit configuration for the AmZ8530 version and includes register select bits in addition to command and reset codes. Figure 7-2 shows the bit configuration for the AmZ8030 version and includes (in Channel B only) the address decoding select described in the Programming section. The following bit description for WR0 is identical for both versions except where specified.

**Bits D7 and D6: CRC Reset Codes 0 And 1**

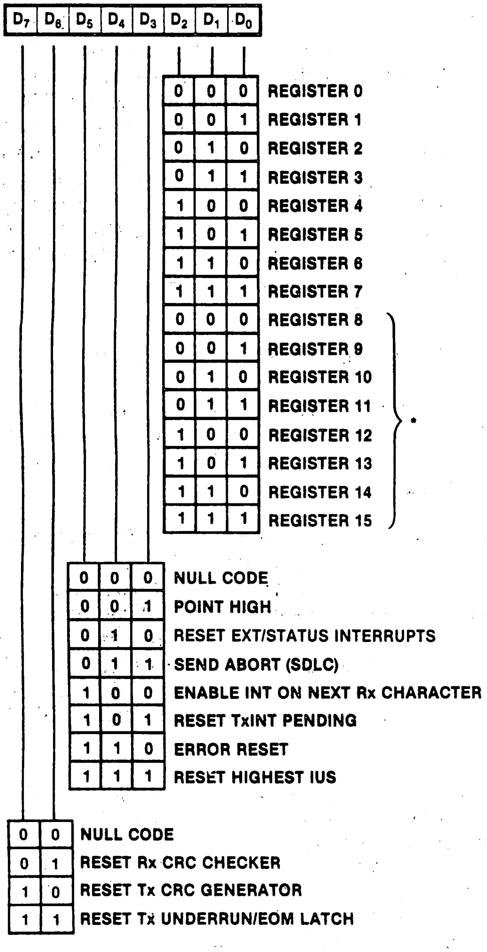
**Null code (00).** This command has no effect on the SCC and is used when a write to WR0 is necessary for some reason other than a CRC Reset command.

**Reset Receive CRC Checker (01).** This command is used to initialize the receive CRC circuitry. It is necessary in synchronous modes (except SDLC) if the Enter Hunt Mode command in Write Register 3 is not issued between received messages. Any action that disables the receiver initializes the CRC circuitry. Resetting the Receive CRC Checker command is accomplished automatically in SDLC mode.

**Reset Transmit CRC Generator (10).** This command initializes the CRC generator. It is usually issued in the initialization routine and after the CRC has been transmitted. A Channel Reset will not initialize the generator and this command should not be issued until after the transmitter has been enabled in the initialization routine.

**Reset Transmit Underrun/EOM Latch (11).** This command controls the transmission of CRC at the end of transmission (EOM). If this latch has been reset, and a transmit underrun occurs, the SCC automatically appends CRC to the mes-sage. In SDLC mode with Abort on Underrun se-lected, the SCC sends an abort, and Flag on under-run if the TX Underrun/EOM latch as been reset.

At the start of the CRC transmission, the Tx Underrun/EOM latch is set. The Reset command can be issued at any time during a message. If the transmitter is disabled, this command will not reset the latch. However, if no External Status interrupt is pending, or if a Reset External Status Int command accompanies this command while the transmitter is disabled, an External/Status interrupt is generated with the Tx Underrun/EOM bit reset in RR0.



*WITH POINT HIGH COMMAND

07513A 7-1

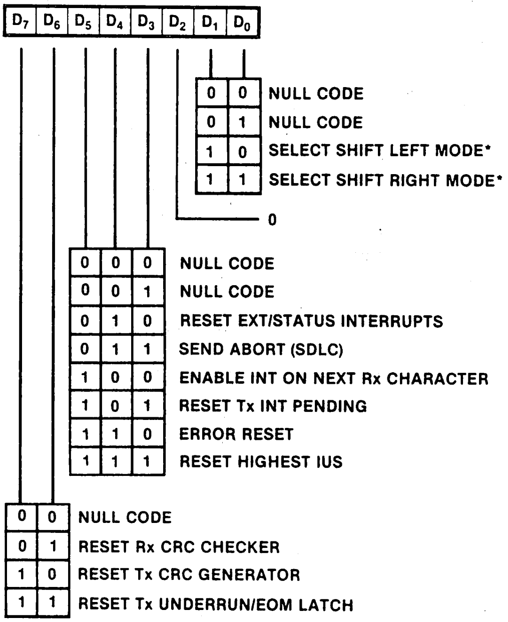
Figure 7-1 Write Register 0 (Z8530)

**Bits D5–D3: Command Codes**

**Null Code (000).** The Null command has no effect on the SCC.

**Point High (001).** This command effectively adds eight to the Register Pointer (B2–B0) by allowing WR8 through WR15 to be accessed. The Point High command and the Register Pointer bits are written simultaneously. This command is used only in the Z8530 version of the SCC. In the Z8030 version, the registers are accessed as described in the Programming section.

**Reset External/Status Interrupts (010).** After an External/Status interrupt (a change on a modem line or a break condition, for example), the status bits in RR0 are latched. This command re-enables the bits and allows interrupts to occur again as a result of a status change. Latching the status bits captures short pulses until the CPU has time to read the change. The SCC contains simple queueing logic associated with most of the external status bits in RR0. If another External/Status condition changes while a previous condition is still pending (Reset External/Status Interrupts has not yet been issued) and this condition persists until after the command is issued, this second change causes another


D7 D6 D5 D4 D3 D2 D1 D0
0 0 NULL CODE
0 1 NULL CODE
1 0 SELECT SHIFT LEFT MODE*
1 1 SELECT SHIFT RIGHT MODE*
0
0 0 0 NULL CODE
0 0 1 NULL CODE
0 1 0 RESET EXT/STATUS INTERRUPTS
0 1 1 SEND ABORT (SDLC)
1 0 0 ENABLE INT ON NEXT Rx CHARACTER
1 0 1 RESET Tx INT PENDING
1 1 0 ERROR RESET
1 1 1 RESET HIGHEST IUS
0 0 NULL CODE
0 1 RESET Rx CRC CHECKER
1 0 RESET Tx CRC GENERATOR
1 1 RESET Tx UNDERRUN/EOM LATCH

07513A 7-2

Figure 7-2 Write Register 0 (Z8030)

External/Status interrupt. However, if this second status change does not persist (there are two transitions), another interrupt is not generated. Exceptions to this rule are detailed in the RR0 description.

**Send Abort (011).** This command is used in SDLC mode to transmit a sequence of eight to thirteen "1s." This command always empties the transmit buffer and sets Tx Underrun/EOM bit in Read Register 0.

**Enable Interrupt on Next Rx Character (100).** If the interrupt on the First Received Character mode is selected, this command is used to reactivate that mode after each message is received. The next character to enter the receive FIFO causes a Receive interrupt. Alternatively, the first previously stored character in the FIFO will cause a Receive interrupt.

**Reset Tx Interrupt Pending (101).** This command is used in cases where there are no more characters to be sent; e.g., at the end of a message. This command prevents further transmit interrupts until after the next character has been loaded into the transmit buffer or until CRC has been completely sent. This command is necessary to prevent the transmitter from requesting an interrupt when the transmit buffer becomes empty (with Transmit Interrupt Enabled).

**Error Reset (110).** This command resets the error bits in RR1. If Interrupt on First Rx Character or Interrupt on Special Condition modes are selected and a special condition exists, the data with the special condition is held in the receive FIFO until this command is issued. If either of these modes is selected and this command is issued before the data has been read from the receive FIFO, the data is lost.

**Reset Highest IUS (111).** This command resets the highest priority Interrupt Under Service (IUS) bit, allowing lower priority conditions to request interrupts. This command allows the use of the internal daisy chain (even in systems without an external daisy chain) and should be the last operation in an interrupt service routine.

**Bits 2 through 0:**
**Resister Selection Code**

These three bits select Registers 0 through 7. With the Point High command, Registers 8 through 15 are selected. The Register Selection Code bits are used only in the AmZ8530 version. In the AmZ8030 version, bit D2 is always "0." Bits D1 and D0 select Shift Left/Right.

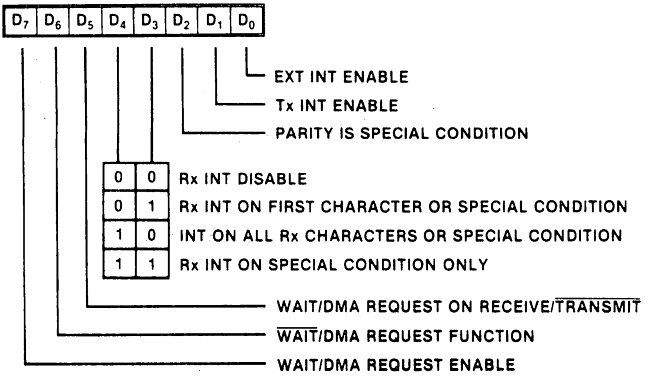
The following is a summary of the bit descriptions for each write register (WR1–WR15) used in both the AmZ8530 and AmZ8030 SCC.

### 7.1.2 Write Register 1 (Transmit/Receive Interrupt and Data Transfer Mode Definition).

Write Register 1 is the control register for the various SCC interrupt and Wait/Request modes. Figure 7-3 shows the bit assignments for WR1.

**Bit 7: WAIT/DMA Request Enable**

This bit enables the Wait/Request function in conjunction with the Request/Wait Function Select bit (B6). If bit 7 is set to "1," the state of bit 6 determines the activity of the WAIT/REQUEST pin (Wait or Request). If bit 7 is set to "0," the selected function (bit 6) forces the WAIT/REQUEST pin in



$D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$
EXT INT ENABLE
Tx INT ENABLE
PARITY IS SPECIAL CONDITION
0 0 Rx INT DISABLE
0 1 Rx INT ON FIRST CHARACTER OR SPECIAL CONDITION
1 0 INT ON ALL Rx CHARACTERS OR SPECIAL CONDITION
1 1 Rx INT ON SPECIAL CONDITION ONLY
WAIT/DMA REQUEST ON RECEIVE/TRANSMIT
WAIT/DMA REQUEST FUNCTION
WAIT/DMA REQUEST ENABLE
07513A 7-3

Figure 7-3 Write Register 1

to the appropriate inactive state (High for Request, floating for Wait).

### Bit 6: WAIT/DMA Request Function

The request function is selected by setting this bit to "1." In the Request mode, the WAIT/REQUEST pin switches from High to Low when the SCC is ready to transfer data. When this bit is "0," the wait function is selected. In the Wait mode, the WAIT/REQUEST pin switches from floating to Low when the CPU attempts to transfer data before the SCC is ready.

### Bit 5: WAIT/DMA Request On Receive Transmit

This bit determines whether the WAIT/REQUEST pin operates in the Transmit mode or the Receive mode. When set to "1," this bit allows the wait/request function to follow the state of the receive buffer; i.e., depending on the state of bit 6, the WAIT/REQUEST pin is active or inactive in relation to the empty or full state of the receive buffer. Conversely, if this bit is set to "0," the state of the WAIT/REQUEST pin is determined by bit 6 and the state of the transmit buffer. (Note that a transmit request function is available on the DTR/REQUEST pin. This allows full-duplex operation under DMA control for both channels.)

The request function may occur only when the SCC is not selected; e.g., if the internal request becomes active while the SCC is in the middle of a read or write cycle, the external request will not become active until the cycle is complete. An active request output causes a DMA controller to initiate a read or write operation. If the request on Transmit mode is selected in either SDLC or Synchronous Mode, the Request pin is pulsed Low for one PCLK cycle at the end of CRC transmission to allow the immediate transmission of another block of data.

If the Wait On Receive mode, the WAIT pin is active if the CPU attempts to read SCC data that has not yet been received. In the Wait On Transmit mode, the WAIT pin is active if the CPU attempts to write data when the transmit buffer is still full. Both situations can occur frequently when block transfer instructions are used.

### Bits 4 and 3: Receive Interrupt Modes

These two bits specify the various character-available conditions that may cause interrupt requests.

**Receive Interrupts Disabled (00).** This mode prevents the receiver from requesting an interrupt and is normally used in a polled environment where either the status bits in RR0 or the modified vector in RR2 (Channel B) can be monitored to initiate a service routine. Although the receiver interrupts are disabled, a special condition can still provide a unique vector status in RR2.

**Receive Interrupt on First Character or Special Condition (01).** The receiver requests an interrupt in this mode on the first available character (or stored FIFO character) or on a special condition. Sync characters to be stripped from the message stream do not cause interrupts.

Special receive conditions are: receiver overrun, framing error, end of frame, or parity error (if selected). If a special receive condition occurs, the data containing the error is stored in the receive FIFO until an Error Reset command is issued by the CPU.

This mode is usually selected when a Block Transfer mode is used. In this interrupt mode, a pending special receive condition remains set until either an Error Reset command, a channel or hardware reset, or until receive interrupts are disabled.

The Receive Interrupt on First Character or Special Condition mode can be re-enabled by the Enable Rx Interrupt on Next Character command in WR0.

**Interrupt on All Receive Characters or Special Condition (10).** This mode allows an interrupt for every character received (or character in the receive FIFO) and provides a unique vector when a special condition exists. The Receiver Overrun bit and the Parity Error bit in RR1 are two special conditions that are latched. These two bits must be reset by the Error Reset command. Receiver overrun is always a special receive condi-tion, and parity can be programmed to be a special condition.

Data characters with special receive conditions are not held in the receive FIFO in the Interrupt On All Receive Characters or Special Conditions Mode as they are in the other receive interrupt modes.

**Receive Interrupt on Special Condition (11).** This mode allows the receiver to interrupt only on characters with a special receive condition. When an interrupt occurs, the data containing the error is held in the receive FIFO until an Error Reset command is issued. When using this mode in conjunction with a DMA, the DMA can be initialized and enabled before any characters have been received by the SCC. This eliminates the time-critical section of code required in the Receive Interrupt on First Character or Special condition mode; i.e., all data can be transferred via the DMA so that the CPU need not handle the first received character as a special case.

**Bit 2: Parity Is Special Condition**

If this bit is set to "1," any received characters with parity not matching the sense programmed in WR4 give rise to a Special Receive Condition. If parity is disabled (WR4), this bit is ignored. A special condition modifies the status of the interrupt vector stored in WR2. During an interrupt acknowledge cycle, this vector can be placed on the data bus.

**Bit 1: Transmitter Interrupt Enable**

If this bit is set to "1," the transmitter requests an interrupt whenever the transmit buffer becomes empty.

**Bit 0: External/Status Master Interrupt Enable**

This bit is the master enable for External/Status interrupts including $\overline{\text{DCD}}$, $\overline{\text{CTS}}$, $\overline{\text{SYNC}}$ pins, break, abort, the beginning of CRC transmission when the Transmit/Underrun/EOM latch is set, or when the counter in the baud rate generator reaches "0." Write Register 15 contains the individual enable bits for each of these sources of External/Status interrupts. This bit is reset by a channel or hardware reset.

### 7.1.3 Write Register 2 (interrupt vector)

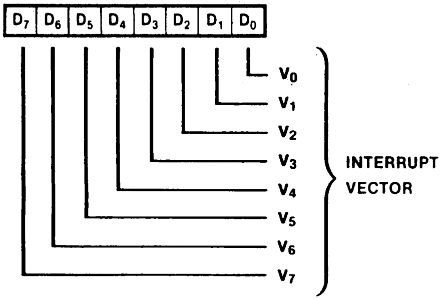
WR2 is the interrupt vector register. Only one vector register exists in the SCC, but it can be accessed through either channel. The interrupt vector can be modified by status information. This is controlled by the Vector Includes Status (VIS) and the Status High/Status Low bits in WR9. The bit positions for WR2 are shown in Figure 7-4.

### 7.1.4 Write Register 3 (Receive Parameters and Control)

This register contains the control bits and parameters for the receiver logic as illustrated in Figure 7-5.

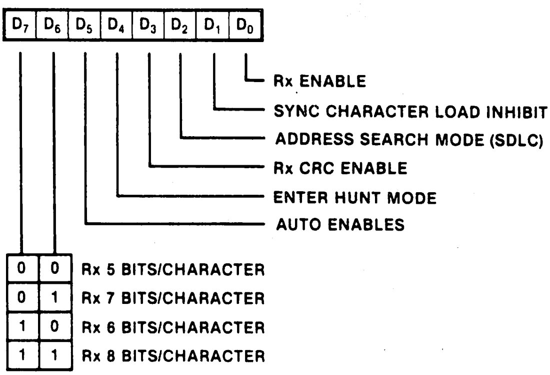
**Bits 7 and 6: Receiver Bits/Character**

The state of these two bits determines the number of bits to be assembled as a character in the received serial data stream. The number of bits per character can be changed while a character is being assembled but only before the number of bits currently programmed is reached. Unused bits in the Received Data Register (RR8) are set to "1" in asynchronous modes. In synchronous modes and SDLC modes, the SCC merely transfers an 8-bit section of the serial data stream to the receive FIFO at the appropriate time. Table 7-2 lists the number of bits per character in the assembled character format.



07513A 7-4

Figure 7-4 Write Register 2



07513A 7-5

Figure 7-5 Write Register 3

Table 7-2 Receive Bits/Character

| $B_7$ | $B_6$ | |
|---|---|---|
| 0 | 0 | 5 Bits/Character |
| 0 | 1 | 7 Bits/Character |
| 1 | 0 | 6 Bits/Character |
| 1 | 1 | 8 Bits/Character |

### Bit 5: Auto Enables

This bit programs the function for both the $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$ pins. $\overline{\text{CTS}}$ becomes the transmitter enable and $\overline{\text{DCD}}$ becomes the receiver enable when this bit is set to "1." However, the Receiver Enable and Transmit Enable bits must be set before the $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$ pins can be used in this manner. When the Auto Enables bit is set to "0," the $\overline{\text{DCD}}$ and $\overline{\text{CTS}}$ pins are merely inputs to the corresponding status bits in Read Register 0. The state of $\overline{\text{DCD}}$ is ignored in the Local Loopback mode. The state of $\overline{\text{CTS}}$ is ignored in both Auto Echo and Local Loopback modes.

### Bit 4: Enter Hunt Mode

This command forces the comparison of sync characters or flags to assembled receive characters for the purpose of synchronization. After reset, the SCC automatically enters the Hunt mode (except asynchronous). Whenever a flag or sync character is matched, the Sync/Hunt bit in Read Register 0 is reset and, if External/Status Interrupt Enable is set, an interrupt sequence is initiated. The SCC automatically enters the Hunt mode when an abort condition is received or when the receiver is disabled.

### Bit 3: Receiver CRC Enable

This bit is used to initiate CRC calculation at the beginning of the last byte transferred from the Receiver Shift register to the receive FIFO. This operation occurs independently of the number of bytes in the receive FIFO. When a particular byte is to be excluded from CRC calculation, this bit should be reset before the next byte is transferred to the receive FIFO. If this feature is used, care must be taken to ensure that eight bits per character is selected in the receiver because of an inherent delay from the Receive Shift register to the CRC checker.

This bit is internally set to "1" in SDLC mode and the SCC calculates CRC on all bits except inserted zeros between the opening and closing character flags. This bit is ignored in asynchronous modes.

### Bit 2: Adress Search Mode (SDLC)

Setting this bit in SDLC mode causes messages with addresses not matching the address programmed in WR6 to be rejected. No receiver interrupts can occur in this mode unless there is an address match. The address that the SCC attempts to match can be unique (1 in 256) or multiple (16 in 256), depending on the state of Sync Character Load Inhibit bit. The Address Search mode bit is ignored in all modes except SDLC.

### Bit 1: SYNC Character Load Inhibit

If this bit is set to "1" in any synchronous mode except SDLC, the SCC compares the byte in WR6 with the byte about to be stored in the FIFO, and it inhibits this load if the bytes are equal. The SCC does not calculate the CRC on bytes stripped from the Data stream in this manner. If the 6-bit sync option is selected while in Monosync mode, the compare is still across eight bits, so WR6 must be programmed for proper operation.

If the 6-bit sync option is selected with this bit set to "1," all sync characters except the one immediately preceding the data are stripped from the message. If the 6-bit sync option is selected while in the Bisync mode, this bit is ignored.

The address recognition logic of the receiver is modified in SDLC mode if this bit is set to "1;" i.e., only the four most significant bits of WR6 must match the receiver address. This procedure allows the SCC to receive frames from up to 16 separate sources without programming WR6 for each source (if each station address has the four most significant bits in common). The address field in the frame is still eight bits long.

This bit is ignored in SDLC mode if Address Search mode has not been selected.

### Bit 0: Receiver Enable

When this bit is set to "1," receiver operation begins. This bit should be set only after all other receiver parameters are established and the receiver is completely initialized. This bit is reset by a channel or hardware reset command, and it disables the receiver.

### 7.1.5 Write Register 4 (Transmit/Receiver Miscellaneous Parameters and Modes)

WR4 contains the control bits for both the receiver and the transmitter. These bits should be set in the transmit and receiver initialization routine before issuing the contents of WR1, WR3, WR6, and WR7. Bit positions for WR4 are shown in Figure 7-6.

**Bits 7 and 6: Clock Rate 1 And 0**

These bits specify the multiplier between the clock and data rates. In synchronous modes, the 1S mode is forced internally and these bits are ignored unless External Sync mode has been selected.

**1X Mode (00).** The clock rate and data rate are the same. In External Sync mode, this bit combination specifies that only the $\overline{\text{SYNC}}$ pin can be used to achieve character synchronization.

**16X Mode (01).** The clock rate is 16 times the data rate. In External Sync mode, this bit combin-ation specifies that only the $\overline{\text{SYNC}}$ pin can be used to achieve character synchronization.

**32X Mode (10).** The clock rate is 32 times the data rate. In External Sync mode, this bit combination specifies that either the $\overline{\text{SYNC}}$ pin or a match with the character stored in WR7 will signal character synchronization. The sync character can be either six or eight bits long as specified by the 6-bit/8-bit Sync bit in WR10.

**64X Mode (11).** The clock rate is 64 times the data rate. With this bit combination in External Sync mode, both the receiver and transmitter are placed in SDLC mode. The only variation from normal SDLC operation is that the $\overline{\text{SYNC}}$ pin can be used to start or stop the reception of a frame by forcing the receiver to act as though a flag had been received.

**Bits 5 and 4: SYNC Modes 1 And 0**

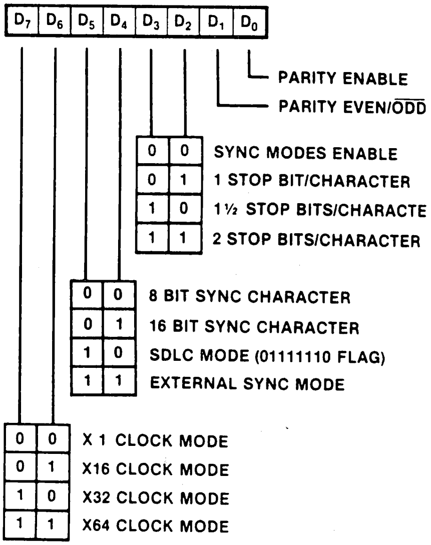
These two bits select the various options for character synchronization. They are ignored unless synchronous modes are selected in the stop bits field of this register.

**Monosync (00).** In this mode, the receiver achieves character synchronization by matching the character stored in WR7 with an identical character in the received data stream. The transmitter uses the character stored in WR6 as a time fill. The sync character can be either six or eight bits, depending on the state of the 6-bit/8-bit Sync bit in WR10. If the Sync Character Load Inhibit bit is set, the receiver strips the contents of WR6 from the data stream if received within character boundaries.

**Bisync (01).** The concatenation of WR7 with WR6 is used for receiver synchronization and as a time fill by the transmitter. The sync character can be 12 or 16 bits in the receiver, depending on the state of the 6-bit/8-bit Sync bit in WR10. The transmitted character is always 16 bits.

**SDLC Mode (10).** In this mode, SDLC is selected and requires a Flag (01111110) to be written to WR7. The receiver address field should be written to WR6. The SDLC CRC polynomial must also be selected (WR5) in SDLC mode.

**External Sync Mode (11).** In this mode, the SCC expects external logic to signal character synchronization via the $\overline{\text{SYNC}}$ pin. If the crystal oscillator option is selected (in WR11), the internal $\overline{\text{SYNC}}$ signal is forced to "0." In this mode, bits B7–B6 of this register select special version of External Sync mode. In this mode, the transmitter is in Monosync mode using the contents of WR6 as the time fill with the sync character length specified by the 6-bit/8-bit Sync bit in WR10.



$D_7$ $D_6$ $D_5$ $D_4$ $D_3$ $D_2$ $D_1$ $D_0$
PARITY ENABLE
PARITY EVEN/$\overline{\text{ODD}}$
0 0 SYNC MODES ENABLE
0 1 1 STOP BIT/CHARACTER
1 0 1½ STOP BITS/CHARACTE
1 1 2 STOP BITS/CHARACTER
0 0 8 BIT SYNC CHARACTER
0 1 16 BIT SYNC CHARACTER
1 0 SDLC MODE (01111110 FLAG)
1 1 EXTERNAL SYNC MODE
0 0 X 1 CLOCK MODE
0 1 X16 CLOCK MODE
1 0 X32 CLOCK MODE
1 1 X64 CLOCK MODE

07513A 7-6

**Figure 7-6 Write Register 4**

### Bits 3 and 2: Stop Bits 1 and 0

These bits determine the number of stop bits added to each asynchronous character that is transmitted. The receiver always checks for one stop bit in Asynchronous mode. A Special mode specifies that a Synchronous mode is to be selected. B2 is always set to "1" by a channel or hardware reset to ensure that the $\overline{\text{SYNC}}$ pin is in a known state after a reset.

**Synchronous Modes Enable (00).** This bit combination selects one of the synchronous modes specified by bits B4, B5, B6, and B7 of this register and forces the 1X Clock mode internally.

**1 Stop Bit/Character (01).** This bit selects Asynchronous mode with one stop bit per character.

**1 1/2 Stop Bits/Character (10).** These bits select Asynchronous mode with 1-1/2 stop bits per character. This mode can not be used with the 1X clock mode.

**2 Stop Bits/Character (11).** These bits select Asynchronous mode with two stop bits per transmitted character and check for one received stop bit.

### Bit 1: Parity Even/$\overline{\text{Odd}}$

This bit determines whether parity is checked as even or odd. A "1" programmed here selects even parity, and a "0" selects odd parity. This bit is ignored if the Parity Enable bit is not set.

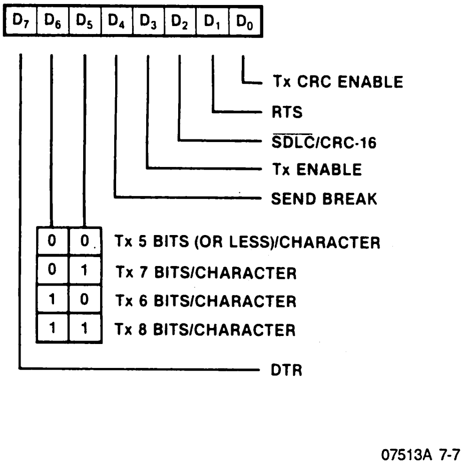


Figure 7-7 Write Register 5

### Bit 0: Parity Enable

When this bit is set, an additional bit position beyond those specified in the bits/character control is added to the transmitted data and is expected in the receive data. The Received Parity bit is transferred to the CPU as part of the data unless eight bits per character is selected in the receiver.

## 7.1.6 Write Register 5 (Transmit Parameter and Controls)

WR5 contains control bits that affect the operation of the transmitter. B2 affects both the transmitter and the receiver. Bit positions for WR5 are shown in Figure 7-7.

### Bit 7: Data Terminal Ready

This is the control bit for the $\overline{\text{DTR}}/\overline{\text{REQ}}$ pin while the pin is in the DTR mode (selected in WR14). When set, $\overline{\text{DTR}}$ is Low; when reset, $\overline{\text{DTR}}$ is High. This bit is ignored when $\overline{\text{DTR}}/\overline{\text{REQ}}$ is programmed to act as a $\overline{\text{REQUEST}}$ pin. This bit is reset by a channel or hardware reset.

### Bits 6 and 5: TX Bits/Chacter 1 and 0

These bits control the number of bits in each byte transferred to the transmit buffer. Bits sent must be right justified with lease significant bits first.

The Five Or Less mode allows transmission of one to five bits per character; however, the CPU should format the data character as shown below in Table 7.3. In the Six or Seven Bits/Character modes, unused data bits are ignored.

Table 7-3 Tx Bits/Character 1 and 0

| Tx BITS/ CHAR 1 | Tx BITS/ CHAR 0 | |
|---|---|---|
| 0 | 0 | 5 or less bits/ character |
| 0 | 1 | 7 bits/character |
| 1 | 0 | 6 bits/character |
| 1 | 1 | 8 bits/character |

| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ | |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | Sends one data bit |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | Sends two data bits |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Sends three data bits |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Sends four data bits |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Sends five data bits |

**Bit 4: Send Break**

When set, this bit forces the TxD output to send continuous "0s" beginning with the following transmit clock, regardless of any data being transmitted at the time. This bit functions whether or not the transmitter is enabled. When reset, TxD continues to send the contents of the Transmit Shift register, which might be syncs, data, or all "1s." If this bit is set while in the X21 mode (Monosync and Loop mode selected) and character synchronization is achieved in the receiver, this bit is automatically reset and the transmitter begins sending syncs or data. This bit can also be reset by a channel or hardware reset.

**Bit 3: Transmit Enable**

Data is not transmitted until this bit is set, and the TxD output sends continuous "1s" unless Auto Echo mode or SDLC Loop mode is selected. If this bit is reset after transmission started, the transmission of data or sync characters is completed. If the transmitter is disabled during the transmission of a CRC character, sync or flag characters are sent instead of CRC. This bit is reset by a channel or hardware reset.

**Bit 2: $\overline{\text{SDLC}}$/CRC-16**

This bit selects the CRC polynomial used by both the transmitter and receiver. When set, the CRC-16 polynomial is used; when reset, the SDLC polynomial is used. The SDLC/CRC polynomial must be selected when SDLC mode is selected. The CRC generator and checker can be preset to all "0s" or all "1s", depending on the state of the Preset 1/Preset 0 bit in WR10.

**Bit 1: Request To Send**

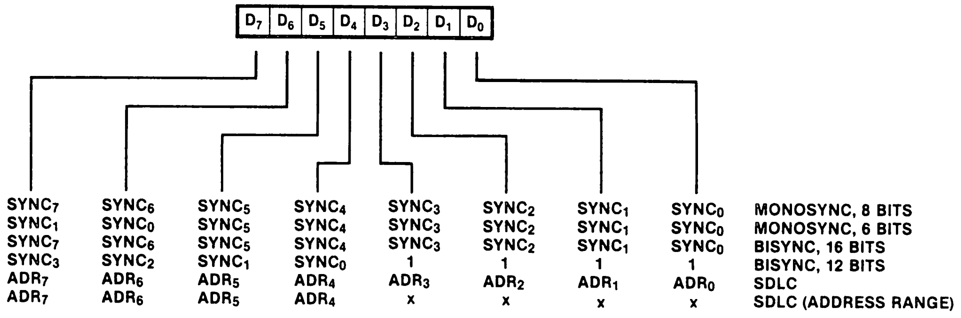
This is the control bit for the $\overline{\text{RTS}}$ pin. When the RTS bit is set, the $\overline{\text{RTS}}$ pin goes Low; when reset, $\overline{\text{RTS}}$ goes High. In the Asynchronous mode with the Auto Enables bit set, $\overline{\text{RTS}}$ goes High only after all bits of the character have been sent and the transmit buffer is empty. In synchronous modes of the Asynchronous mode with auto enables off, the pin directly follows the state of this bit. This bit is reset by a channel or hardware reset.

**Bit 0: Transmit CRC Enable**

This bit determines whether or not CRC is calculated on a transmit character. If this bit is set at the time the character is loaded from the transmit buffer to the Transmit Shift register, CRC is calculated on that character. CRC is not automatically sent unless this bit is set when the transmit underrun exists.

### 7.1.7 Write Register 6 (Sync Characters or SDLC Address Field)

WR6 is programmed to contain the transmit sync character in the Monosync mode, the first byte of a 16-bit sync character in the External Sync mode. WR6 is not used in asynchronous modes. In the SDLC modes, it is programmed to contain the secondary address field used to compare against the address field of the SDLC Frame. In SDLC mode, the SCC does not automatically transmit the station address at the beginning of a response frame. Bit positions for WR6 are shown in Figure 7-8.



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| SYNC$_7$ | SYNC$_6$ | SYNC$_5$ | SYNC$_4$ | SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | MONOSYNC, 8 BITS |
| SYNC$_1$ | SYNC$_0$ | SYNC$_5$ | SYNC$_4$ | SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | MONOSYNC, 6 BITS |
| SYNC$_7$ | SYNC$_6$ | SYNC$_5$ | SYNC$_4$ | SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | BISYNC, 16 BITS |
| SYNC$_3$ | SYNC$_2$ | SYNC$_1$ | SYNC$_0$ | 1 | 1 | 1 | 1 | BISYNC, 12 BITS |
| ADR$_7$ | ADR$_6$ | ADR$_5$ | ADR$_4$ | ADR$_3$ | ADR$_2$ | ADR$_1$ | ADR$_0$ | SDLC |
| ADR$_7$ | ADR$_6$ | ADR$_5$ | ADR$_4$ | x | x | x | x | SDLC (ADDRESS RANGE) |

07513A 7-8

Figure 7-8 Write Register 6

### 7.1.8 Write Register 7 (SYNC Character or SDLC Flag)

WR7 is programmed to contain the receive sync character in the Monosync mode, a second byte (the last eight bits) of a 16-bit sync character in the Bisync mode, or a Flag character (01111110) in the SDLC modes. WR7 may hold the receive sync character or a flag if one of the special versions of the External Sync mode is selected. WR7 is not used in Asynchronous mode. Bit positions for WR7 are shown in Figure 7-9.

### 7.1.9 Write Register 8 (Transmit Buffer)

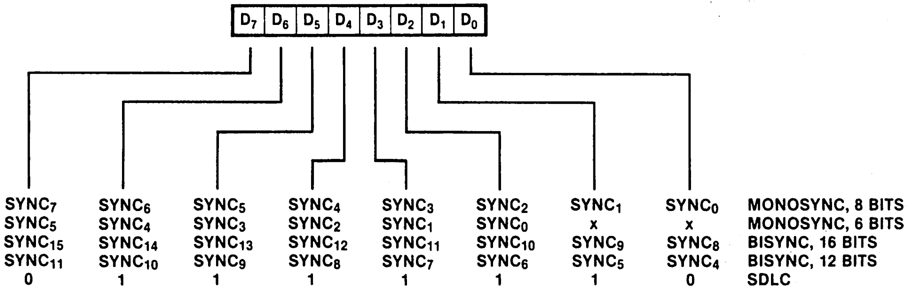
WR8 is the transmit buffer register.

### 7.1.10 Write Register 9 (Master Interrupt Control)

WR9 is the Master Interrupt Control register and contains the Reset command bits. Only one WR9 exists in the SCC and can be accessed from either channel. The interrupt control bits can be programmed at the same time as the Reset command because these bits are only reset by a hardware reset. Bit positions for WR9 are shown in Figure 7-10.

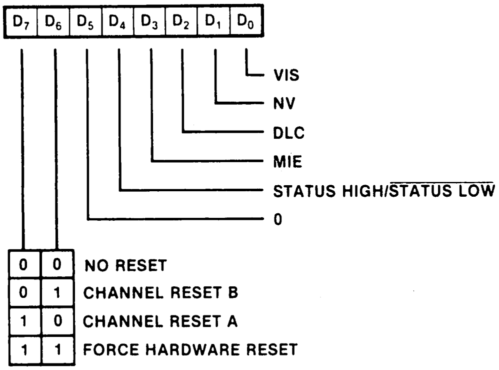
**Bits 7 and 6: Reset Command Bits**

Together, these bits select one of the reset commands for the SCC. Setting either of these bits to "1" disables both the receiver and the transmitter in the corresponding channel, forces TxD for that channel marking, forces the modem control signals High in that channel, resets all IPs and IUSs and disables all interrupts in that channel. Four extra PCLK cycles must be allowed beyond the usual cycle time after any of the active reset commands is issued before any additional commands or controls are written to the channel affected. In the AmZ8530, four extra PCLK cycles


D$_7$ D$_6$ D$_5$ D$_4$ D$_3$ D$_2$ D$_1$ D$_0$
SYNC$_7$ SYNC$_6$ SYNC$_5$ SYNC$_4$ SYNC$_3$ SYNC$_2$ SYNC$_1$ SYNC$_0$ MONOSYNC, 8 BITS
SYNC$_5$ SYNC$_4$ SYNC$_3$ SYNC$_2$ SYNC$_1$ SYNC$_0$ x x MONOSYNC, 6 BITS
SYNC$_{15}$ SYNC$_{14}$ SYNC$_{13}$ SYNC$_{12}$ SYNC$_{11}$ SYNC$_{10}$ SYNC$_9$ SYNC$_8$ BISYNC, 16 BITS
SYNC$_{11}$ SYNC$_{10}$ SYNC$_9$ SYNC$_8$ SYNC$_7$ SYNC$_6$ SYNC$_5$ SYNC$_4$ BISYNC, 12 BITS
0 1 1 1 1 1 1 0 SDLC

07513A 7-9

Figure 7-9 Write Register 7


D$_7$ D$_6$ D$_5$ D$_4$ D$_3$ D$_2$ D$_1$ D$_0$
VIS
NV
DLC
MIE
STATUS HIGH/$\overline{\text{STATUS LOW}}$
0
0 0 NO RESET
0 1 CHANNEL RESET B
1 0 CHANNEL RESET A
1 1 FORCE HARDWARE RESET

07513A 7-10

Figure 7-10 Write Register 9

must be allowed beyond the usual cycle time before any additional command or controls are written to the SCC.

**No Reset (00).** This command has no effect. It is used when a write to WR9 is necessary for some reason other than an SCC Reset command.

**Channel Reset B (01).** Issuing this command causes a channel reset to be performed on Channel B.

**Channel Reset A (10).** Issuing this command causes a channel reset to be performed on Channel A.

**Force Hardware Reset (11).** The effects of this command are identical to those of a hardware reset, except that the Shift Right/Shift Left bit is not changed and the MIE, Status High/Status Low and DLC bits take the programmed values that accompany this command.

### Bit 5: Not Used

Must be "0."

### Bit 4: Status High/$\overline{\text{Status Low}}$

This bit controls which vector bits the SCC will modify to indicate status. When set to "1," the SCC modifies bits V6, V5, and V4 according to Table 7-4. When set to "0," the SCC modifies bits V1, V2, and V3 according to Table 7-34 This bit controls status in both the vector returned during an interrupt acknowledge cycle and the status in RR2B. This bit is reset by a hardware reset.

**Table 7-4 Interrupt Vector Modification**

| V3 | V2 | V1 | Status High/Status Low = 0 |
|---|---|---|---|
| V4 | V5 | V6 | Status High/Status Low = 1 |
| 0 | 0 | 0 | Ch B Transmit Buffer Empty |
| 0 | 0 | 1 | Ch B External/Status Change |
| 0 | 1 | 0 | Ch B Receive Character Avail. |
| 0 | 1 | 1 | Ch B Special Receive Condition |
| 1 | 0 | 0 | Ch A Transmit buffer Empty |
| 1 | 0 | 1 | Ch A External/Status Change |
| 1 | 1 | 0 | Ch A Receive Character Avail. |
| 1 | 1 | 1 | Ch A Special Receive Condition |

### Bit 3: Master Interrupt Enable

This bit is set to 1 to globally enable interrupts, and cleared to zero to disable interrupts. Clearing this bit to zero forces the IEO pin to follow the state of the IEI pin unless there is an IUS bit set in the SCC. No IUS bit can be set after the MIE bit is cleared to zero. This bit is reset by a hardware reset.

### Bit 2: Disable Lower Chain

The Disable Lower Chain bit can be used by the CPU to control the interrupt daisy chain. Setting this bit to "1" forces the IEO pin Low, preventing lower-priority devices on the daisy chain from requesting interrupts. This bit is reset by a hardware reset.

### Bit 1: No Vector

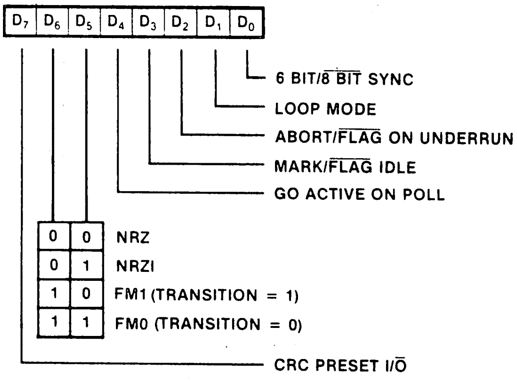
The No Vector bit controls whether or not the SCC will respond to an interrupt acknowledge cycle by placing a vector on the data bus if the SCC is the highest-priority device requesting an interrupt. If this bit is set, no vector is returned; i.e., AD0–AD7 remain three-stated during an interrupt acknowledge cycle, even if the SCC is the highest-priority device requesting an interrupt.

### Bit 0: Vector Includes Status

The Vector Includes Status Bit controls whether or not the SCC will include status information in the vector it places on the bus in response to an interrupt acknowledge cycle. If this bit is set, the vector returned is variable, with the variable field depending on the highest-priority IP that is set. Table 7-4 shows the encoding of the status information. This bit is ignored if the No Vector (NV) bit is set.

## 7.1.11 Write Register 10 (Miscellaneous Transmitter/Receiver Control Bits)

WR10 contains miscellaneous control bits for both the receiver and the transmitter. Bit positions for WR10 are shown in Figure 7-11.



07513A 7-11

**Figure 7-11 Write Register 10**

### Bit 7: CRC Presets I/$\overline{O}$

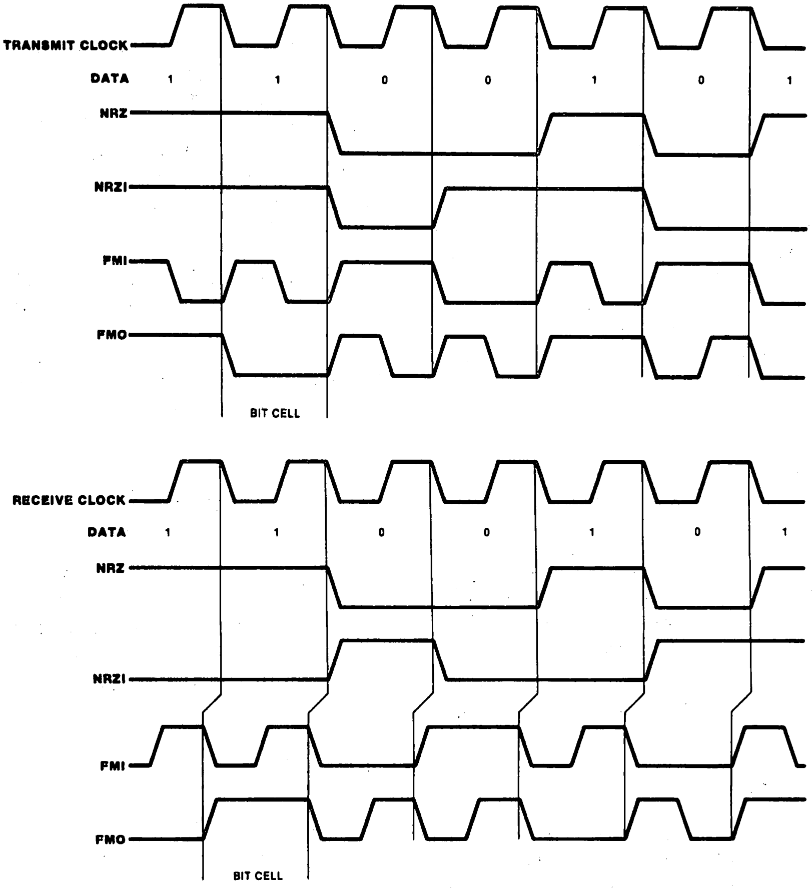
This bit specifies the initialized condition of the receive CRC checker and the transmit CRC generator. If this bit is set to "1," the CRC generator and checker are preset to "1." If this bit is set to "0," the CRC generator and checker are preset to "0." Either option can be selected with either CRC polynomial. In SDLC mode, the transmitted CRC is inverted before transmission and the received CRC is checked against the bit pattern "0001110100001111." This bit is reset by a channel or hardware reset. This bit is ignored in Asynchronous mode.

### Bits 6 and 5: Data Encoding 1 And 2

These bits control the coding method used for both the transmitter and the receiver, as illustrated in Table 7-5. All of the clocking options are available for all coding methods. The DPLL in the SCC is useful for recovering clocking information in NRZI and FM modes. Any coding method can be used in X1 mode. A hardware reset forces NRZ mode. Timing for the various modes in shown in Figure 7-12.

**Table 7-5 Data Encoding**

| Data | Data Encoding | Encoding |
|---|---|---|
| 0 | 0 | NRZ |
| 0 | 1 | NRZI |
| 1 | 0 | FM1 (transition = 1) |
| 1 | 1 | FM0 (transition = 1) |



07513A 7-12

**Figure 7-12 NRZ(NRZI)FM1(FM0) Timing**

### Bit 4: Go Active On Poll

When Loop mode is first selected during SDLC operation, the SCC connects RxD to TxD with only gate delays in the path. The SCC does not go on-loop and insert the 1-bit delay between RxD and TxD until this bit has been set and an EOP received. When the SCC is on-loop, the transmitter cannot go active unless this bit is set at the time an EOP is received. The SCC examines this bit whenever the transmitter is active in SDLC Loop mode and is sending a flag. If this bit is set at the time the flag is leaving the Transmit Shift register, another flag or data byte (if the transmit buffer is full) is transmitted. If the Go Active on Poll bit is not set at this time, the transmitter finishes sending the flag and reverts to the 1-Bit Delay mode. Thus, to transmit only one response frame, this bit should be reset after the first data byte is sent to the SCC but before CRC has been transmitted. If the bit is not reset before CRC is transmitted, extra flags are sent, slowing down response time on the loop. If this bit is reset before the first data is written, the SCC completes the transmission of the present flag and reverts to the 1-Bit Delay mode. After gaining control of the loop, the SCC is not able to transmit again until a flag and another EOP have been received. Though not strictly necessary, it is good practice to set this bit only upon receipt of a poll frame to ensure that the SCC does not go on loop without the CPU noticing it.

In synchronous modes other than SDLC with the Loop Mode bit set, this bit must be set before the transmitter can go active in response to a received sync character.

This bit is always ignored in Asynchronous mode and Synchronous modes unless the Loop Mode bit is set. This bit is reset by a channel or hardware reset.

### Bit 3: Mark/$\overline{\text{Flag}}$ Idle

This bit affects only SDLC operation and is used to control the idle line condition. If this bit is set to "0," the transmitter sends flags as an idle line. If this bit is set to "1," the transmitter sends continuous "1s" after the closing flag of a frame. The idle line condition is selected byte by byte; i.e., either a flag or eight "1s" are transmitted. The primary station in an SDLC loop should be programmed for Mark Idle to create the EOP sequence. Mark Idle must be deselected at the beginning of a frame before the first data is written to the SCC, so that an opening flag can be transmitted. This bit is ignored in Loop mode, but the programmed value takes effect upon exiting the Loop mode. This bit is reset by a channel or hardware reset.

### Bit 2: Abort /$\overline{\text{Flag}}$ On Underrun

This bit affects only SDLC operation and is used to control how the SCC responds to a transmit underrun condition. If this bit is set to "1" and a transmit underrun occurs, the SCC sends an abort and a flag instead of CRC. If this bit is reset, the SCC sends CRC on a transmit underrun. At the beginning of this 16- bit transmission, the Transmit Underrun/EOM bit is set, causing an External/ Status interrupt. The CPU uses this status, along with the byte count from memory or the DMA, to determine whether the frame must be retransmitted. A transmit buffer Empty interrupt occurs at the end of this 16-bit transmission to start the next frame. If both this bit and the Mark/Flag Idle bit are set to "1," all "1s" are transmitted after the transmit underrun. This bit should be set after the first byte of data is sent to the SCC and reset immediately after the last byte of data so that the frame will be terminated properly with CRC and a flag. This bit is ignored in Loop mode, but the programmed value is active upon exiting Loop mode. This bit is reset by a channel or hardware reset.

### Bit 1: Loop Mode

In SDLC mode, the initial set condition of this bit forces the SCC to connect TxD to TxD and to begin searching the incoming data stream so that it can go on loop. All bits pertinent to SDLC mode operation in other registers must be set before this mode is selected. The transmitter and receiver should not be enabled until after this mode has been selected. As soon as the Go Active On Poll bit is set and an EOP is received, the SCC goes on loop. If this bit is reset after the SCC goes on loop. If this bit is reset after the SCC is on loop, the SCC waits for the next EOP to go off loop.

In synchronous modes, the SCC uses this bit, along with the Go Active On Poll bit, to synchronize the transmitter to the receiver. The receiver should not be enabled until after this mode is selected. The TxD pin is held marking when this mode is selected unless a break condition is programmed. The receiver waits for a sync character to be received and then enables the transmitter on a character boundary. The break condition, if programmed, is removed. This mode works properly with sync characters of 6, 8, or 16 bits. This bit is ignored in Asynchronous mode and is reset by a channel or hardware reset.

### Bit 0: 6 Bit/8 Bit SYNC

This bit is used to select a special case of synchronous modes. If this bit is set to "1" in Monosync mode, the receiver and transmitter sync characters are six bits long instead of the usual eight. If this bit is set to "1" in Bisync mode, the received sync will be 12 bits and the transmitter sync character will remain 16 bits long. This bit is ignored in SDLC and Asynchronous modes but still has effect in the special external sync modes. This bit is reset by a channel or hardware reset.

## 7.1.12 Write Register 11 (Clock Mode Control)

WR11 is the Clock Mode Control register. The bits in this register control the sources of both the receive and transmit clocks, the type of signal on the SYNC and RTxC pins, and the direction of the TRxC pin. Bit positions for WR11 are shown in Figure 7-13.

### Bit 7: RTxC—XTAL / NO XTAL

This bit controls the type of input signal the SCC expects to see on the RTxC pin. If this bit is set to "0," the SCC expects a TTL-compatible signal as an input to this pin. If this bit is set to "1," the SCC connects a high-gain amplifier between the RTxC and SYNC pins in expectation of a quartz crystal being placed across the pins.

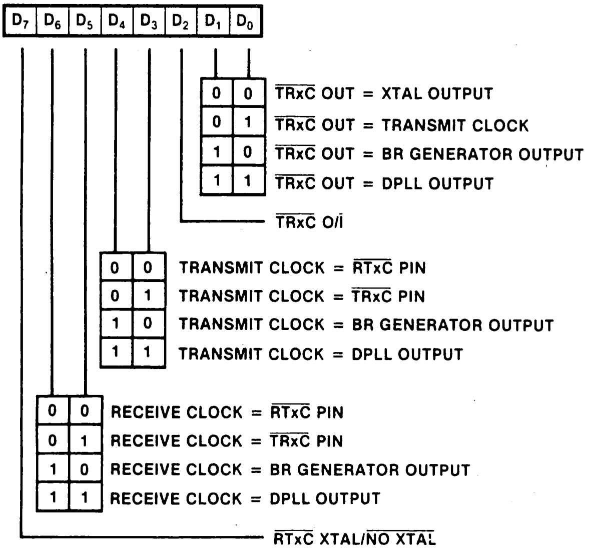
The output of this oscillator is available for use as a clocking source. In this mode of operation, the SYNC pin is unavailable for other use. The SYNC signal is forced to "0" internally. A hardware reset forces NO XTAL. (At least 20 ms should be allowed after this bit is set to allow the oscillator to stabilize.)

### Bits 6 and 5: Receiver Clock 1 And 0

These bits determine the source of the receive clock as shown in Table 7-6. They do not interfere with any of the modes of operation in the SCC but simply control a multiplexer just before the internal receive clock input. A hardware reset forces the receive clock to come from the TRxC pin.

Table 7-6 Receive Clock Source

| Receive Clock 1 | Receive Clock 0 | |
|---|---|---|
| 0 | 0 | Receive Clock = RTxC pin |
| 0 | 1 | Receive Clock = TRxC pin |
| 1 | 0 | Receive Clock = BR output |
| 1 | 1 | Receive Clock = DPLL output |


D7 D6 D5 D4 D3 D2 D1 D0
0 0 TRxC OUT = XTAL OUTPUT
0 1 TRxC OUT = TRANSMIT CLOCK
1 0 TRxC OUT = BR GENERATOR OUTPUT
1 1 TRxC OUT = DPLL OUTPUT
TRxC O/I
0 0 TRANSMIT CLOCK = RTxC PIN
0 1 TRANSMIT CLOCK = TRxC PIN
1 0 TRANSMIT CLOCK = BR GENERATOR OUTPUT
1 1 TRANSMIT CLOCK = DPLL OUTPUT
0 0 RECEIVE CLOCK = RTxC PIN
0 1 RECEIVE CLOCK = TRxC PIN
1 0 RECEIVE CLOCK = BR GENERATOR OUTPUT
1 1 RECEIVE CLOCK = DPLL OUTPUT
RTxC XTAL/NO XTAL

07513A 7-13

Figure 7-13 Write Register 11

### Bits 4 and 3: Transmit Clock 1 and 0

These bits determine the source of the transmit clock as shown in Table 7-7. They do not interfere with any of the modes of operation of the SCC but simply control a multiplexer just before the internal transmit clock input. The DPLL output that may be used to feed the transmitter in FM modes lags by 90 the output of the DPLL used by the receiver. This makes the received and transmitted bit cells occur simultaneously, neglecting delays. A hardware reset selects the TRxC pin as the source of the transmit clocks.

**Table 7-7 Transmit Clock Source**

| Transmit Clock 1 | Transmit Clock 0 | |
|---|---|---|
| 0 | 0 | Transmit Clock = RTxC pin |
| 0 | 1 | Transmit Clock = TRxC pin |
| 1 | 0 | Transmit Clock = BR output |
| 1 | 1 | Transmit Clock = DPLL output |

### Bit 2: TRxC O/I

This bit determines the direction of the TRxC pin. If this bit is set to "1," the TRxC pin is an output and carries the signal selected by D1 and D0 of this register. However, if either the receive or the transmit clock is programmed to come from the TRxC pin, TRxC will be an input, regardless of the state of this bit. The TRxC pin is also an input if this bit is set to "0." A hardware reset forces this bit to "0."

### Bits 1 and 0: TRxC Output Source 1 And 0

These bits determine the signal to be echoed out of the SCC via the TRxC pin. No signal is produced if TRxC has been programmed as the source of either the receive or the transmit clock. If TRxC O/I (bit 2) is set to "0," these bits are ignored.

If the XTAL oscillator output is programmed to be echoed, and the Xtal oscillator has not been enabled, the TRxC pin goes High. The DPLL signal that is echoed is the DPLL signal used by the receiver. Hardware reset selects the XTAL oscillator as the output source.

**Table 7-8 Transmit External Control Selection**

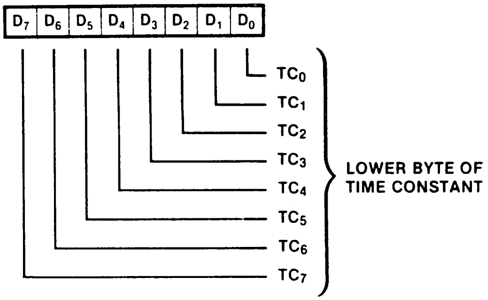
| Output Signal | Output Signal | |
|---|---|---|
| 0 | 0 | TRxC = XTAL oscillator output |
| 0 | 1 | TRxC = Transmit Clock |
| 1 | 0 | TRxC = BR outbut |
| 1 | 1 | TRxC = DPLL output (receive) |

## 7.1.13 Write Register 12 (Lower Byte of Baud Rate Generator Time Constant)

WR12 contains the lower byte of the time constant for the baud rate generator. The time constant can be changed at any time, but the new value does not take effect until the next time the time constant is loaded into the down counter. No attempt is made to synchronize the loading of the time constant into WR12 and WR13 with the clock driving the down counter. For this reason, it is advisable to disable the baud rate generator while the new time constant is loaded into WR12 and WR13. Ordinarily, this is done anyway to prevent a load of the down counter between the writing of the upper and lower bytes of the time constant.

The formula for determining the appropriate time constant for a given baud is shown below with the desired rate in bits per second and the BR clock period in seconds. This formula is derived because the counter decrements from N down to "0"-plus-one-cycle for reloading the time constant and is then fed to a toggle flip-flop to make the output a square wave. Bit positions for WR12 are shown in Figure 7-14.

Time constant = [1/2 • desired rate • BR clock period] – 2



07513A 7-14

**Figure 7-14 Write Register 12**

### 7.1.14 Write Register 13 (Upper Byte of Baud Rate Generator Time Constant)

WR13 contains the upper byte of the time constant for the baud rate generator. Bit positions for WR13 are shown in Figure 7-15.

### 7.1.15 Write Register 14 (Miscellaneous Control Bits)

WR14 contains some miscellaneous control bits. Bit positions for WR14 are shown in Figure 7-16.

#### Bits 7 and 5: Digital Phase-Locked Loop Command Bits

These three bits encode the eight commands for the Digital Phase-Locked Loop. A channel or hardware reset disables the DPLL, resets the mission clock latches, sets the source to the RTxC pin and selects NRZI mode. The Enter Search Mode command enables the DPLL after a reset.

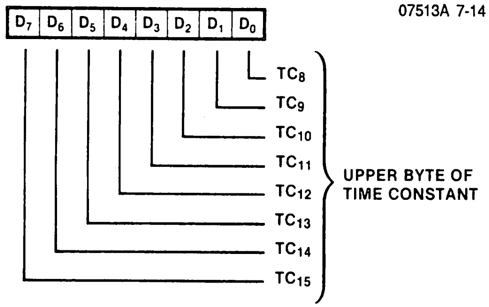
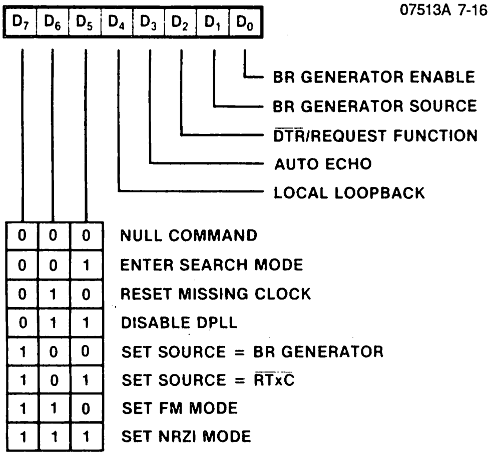


**Figure 7-15 Write Register 13**



| | | | |
|---|---|---|---|
| 0 | 0 | 0 | NULL COMMAND |
| 0 | 0 | 1 | ENTER SEARCH MODE |
| 0 | 1 | 0 | RESET MISSING CLOCK |
| 0 | 1 | 1 | DISABLE DPLL |
| 1 | 0 | 0 | SET SOURCE = BR GENERATOR |
| 1 | 0 | 1 | SET SOURCE = $\overline{\text{RTxC}}$ |
| 1 | 1 | 0 | SET FM MODE |
| 1 | 1 | 1 | SET NRZI MODE |

**Figure 7-16 Write Register 14**

**Null Command (000).** This command has no effect on the DPLL.

**Enter Search Mode (001).** Issuing this command causes the DPLL to enter the Search mode, where the DPLL searches for a locking edge in the incoming data stream. The action taken by the DPLL upon receipt of this command depends on the operating mode of the DPLL.

In NRZI mode, the output of the DPLL is High while the DPLL is waiting for an edge in the incoming data stream. After the Search mode is entered, the first edge the DPLL sees is assumed to be a valid data edge, and the DPLL begins the clock recovery operation from that point. The DPLL clock rate must be 32 times the data rate in NRZI mode. Upon leaving the Search mode, the first sampling edge of the DPLL occurs 16 of these 32X clocks after the first data edge and the second sampling edge occurs 48 of these 32X clocks after the first data edge. Beyond this point, the DPLL begins normal operation, adjusting the output to remain in sync with the incoming data.

In FM mode, the output of the DPLL is Low while the DPLL is waiting for an edge in the incoming data stream. The first edge the DPLL detects is assumed to be a valid clock edge. For this to be the case, the line must contain only clock edges; i.e., with FM1 encoding, the line must be continuous "0s." With FM0 encoding the line must be continuous "1s," whereas Manchester encoding requires alternating "1s" and "0s" on the line. The DPLL clock rate must be 16 times the data rate in FM mode. The DPLL output causes the receiver to sample the data stream in the nominal center of the two halves of the bit cell to decide whether the data was a "1" or a "0." After this command is issued, as in NRZI mode, the DPLL starts sampling immediately after the first edge is detected. (In FM mode, the DPLL examines the clock edge of every other bit cell to decide what correction must be made to remain in sync.) If the DPLL does not see an edge during the expected window, the one clock mission bit in RR10 is set. If the DPLL does not see an edge after two successive attempts, the two clocks missing bit in RR10 is set and the DPLL automatically enters the Search mode. This command resets both clock missing latches.

**Reset Clock Missing (010).** Issuing this command disables the DPLL, resets the clock missing latches in RR10, and forces a continuous Search mode state.

**Disable DPLL (001).** Issuing this command disables the DPLL, resets the clock missing latches in RR10, and forces a continuous Search mode state.

**Set Source = BR Gen (100).** Issuing this command forces the clock for the DPLL to come from the output of the baud rate generator.

**Set Source = RTxC (101).** Issuing the command forces the clock for the DPll to come from the RTxC pin or the crystal oscillator, de-pending on the state of the XTAL/no XTAL bit in WR11. This mode is selected by a channel or hardware reset.

**Set FM Mode (110).** This command forces the DPLL to operate in the FM mode and is used to recover the clock from FM or Manchester-encoded data. (Manchester is decoded by placing the receiver in NRZ mode while the DPLL is in FM mode.)

**Set NRZI Mode (111).** Issuing this command forces the DPLL to operate in the NRZI mode. This mode is also selected by a hardware or channel reset.

### Bit 4: Local Loopback

Setting this bit to "1" selects the Local Loopback mode of operation. In this mode, the internal transmitted data is routed back to the receiver, as well as to the TxD pin. The CTS and DCD inputs are ignored as enables in Local Loopback mode, even if auto enables is selected. (If so programmed, transitions on these inputs still cause interrupts.) This mode works with any Transmit/ Receive mode ex-cept Loop mode. For meaningful results, the frequency of the transmit and receive clocks must be the same. This bit is reset by a channel or hardware reset.

### Bit 3: Auto Enable

Setting this bit to "1" selects the Auto Enable mode of operation. In this mode, the TxD pin is connected to RxD, as in Local Loopback mode, but the receiver still listens to the RxD input. Transmitted data is never seen inside or outside the SCC in this mode, and CTS is ignored as a transmit enable. This bit is reset by a channel or hardware reset.

### Bit 2: DTR/Request Function

This bit selects the function of the DTR/REQ pin follows the state of the DTR bit in WR5. If this bit is set to "0" the DTR/REQ pin follows the state of the DTR bit in WR5. If this bit is set to "1," the DTR/REQ pin goes Low whenever the transmit buffer becomes empty and in any of the synchronous mode when CRC has been sent at the end of a message. The request function on the DTR/REQ pin differs somewhat from the transmit request function available on the W/REQ pin in that REQUEST does not go inactive until the internal operation satisfying the request is complete, which occurs four to five PCLK cycles after the rising edge of DS, READ or WRITE. If the DMA used is edge-triggered, this difference is unimportant. This bit is reset by a channel or hardware reset.

### Bit 1: Baud Rate Generator Source

This bit selects the source of the clock for the baud rate generator. If this bit is set to "0," the baud rate generator clock comes from either the RTxC pin or the XTAL oscillator (depending on the state of the XTAL/no XTAL bit). If this bit is set to "1," the clock for the baud rate generator is the SCC's PCLK input. Hardware reset sets this bit to "0," selecting the RTxC pin as the clock source for the baud rate generator.

### Bit 0: Baud Rate Genterator Enable

This bit controls the operation of the baud rate generator. The counter in the baud rate generator is enabled for counting when this bit is set to "1," and counting is inhibited when this bit is set to "0." When this bit is set to "1," change in the state of this bit is not reflected by the output of the baud rate generator for two counts of the counter. This allows the command to be synchronized. However, when set to "0," disabling is immediate. This bit is reset by a hardware reset.

## 7.1.16 Write Register 15 (External/Status InterruptControl)

WR15 is the External/Status Source Control register. If the External/Status interrupts are enabled as a group via WR1, bits in this register control which External/Status conditions can cause an interrupt. Only the External/Status conditions that occur after the controlling bit are sent to "1" will cause an interrupt. This is true even if an External/Status condition is pending at the time the bit is set. Bit positions for WR15 are shown in Figure 7-17.

**Bit 7: Break/Abort IE**

If this bit is set to "1," a change in the Break/Abort status of the receiver causes an External/Status interrupt. This bit is set by a channel or hardware reset.

**Bit 6: Tx Underrun/EOM**

If this bit is set to "1," a change of state by the Tx Underrun/EOM latch in the transmitter causes an External/Status interrupt. This bit is set to "1" by a channel or hardware reset.

**Bit 5: CTS IE**

If this bit is set to "1," a change of state on the $\overline{\text{CTS}}$ pin causes an External/Status interrupt. This bit is set by a channel or hardware reset.

**Bit 4: SYNC/Hunt IE**

If this bit is set to "1," a change of state on the $\overline{\text{SYNC}}$ pin causes an External/Status interrupt in Asynchronous mode, and a change of state in the Hunt bit in the receiver causes and External/Status interrupt in synchronous modes. This bit is set by a channel or hardware reset.

**Bit 3: DCD IE**

If this bit is set to "1," a change of state on the $\overline{\text{DCD}}$ pin causes an External/Status interrupt. This bit is set by a channel or hardware reset.

**Bit 2: Not Used**

Must be "0."

**Bit 1: Zero Count IE**

If this bit is set to "1," an External/Status interrupt is generated whenever the counter in the baud rate generator reaches "0." This bit is set to "0" by a channel or hardware reset.

**Bit 0: Not Used**

Must be "0."

## 7.2 READ REGISTERS

Both the Z8030 and the Z8530 versions of the SCC contain seven read registers in each channel. In addition there are two registers which are shared by both channels. The status of these registers is continually changing and depends on the mode of communication, received and transmitted data, and the manner in which this data is transferred to and from the CPU. The following description details the bit assignments for each register.

### 7.2.1 Read Register 0 (Transmit/receive buffer Status and External Status)

Read Register 0 contains the status of the receive and transmit buffers. RR0 also contains the status bits for the six sources of External/Status interrupts. The bit configuration is illustrated in Figure 7-18.

**Bit 7: Break /Abort**

In the Asynchronous mode, this bit is set when a Break sequence (null character plus framing error) is detected in the receive data stream. This bit is reset when the sequence is terminated, leaving a single null character in the receive FIFO. This

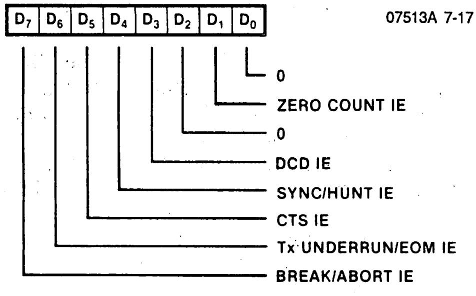


**Figure 7-17 Write Register 15**

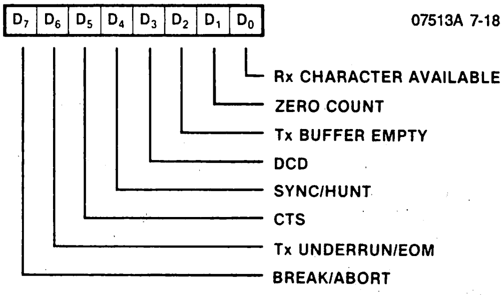


**Figure 7-18 Read Register 0**

character should be read and discarded. In SDLC mode, this bit is set by the detection of an Abort sequence (seven or more "1s"), then reset automatically at the termination of the Abort sequence. In either case, if the Break/Abort IE bit is set, an External/Status interrupt is initiated. Unlike the remainder of the External/Status bits, both transitions are guaranteed to cause an External/ Status interrupt, even if another External/Status interrupt is pending at the time these transitions occur. This procedure is necessary because Abort or Break conditions may not persist.

**Bit 6: TX Underrun/EOM**

This bit is set by a channel or hardware reset and when the transmitter is disabled or a Send Abort command is issued. This bit can only be reset by the reset Tx Underrun/EOM Latch command in WR0. When the Transmit Underrun occurs, this bit is set and causes an External/Status interrupt (if the Tx Underrun/EOM IE bit is set).

Only the 0-to-1 transition of this bit causes an interrupt. This bit is always "1" in Asynchronous mode, unless a reset Tx Underrun/EOM Latch command has been erroneously issued. In this case, the Send Abort command can be used to set the bit to one and at the same time cause an External/Status interrupt.

**Bit 5: Clear to Send**

If the CTS IE bit in WR15 is set, this bit indicates the state of the $\overline{\text{CTS}}$ pin the last time any of the enabled External/Status bits changed. Any transition on the $\overline{\text{CTS}}$ pin while no interrupt is pending latches the state of the $\overline{\text{CTS}}$ pin and generates an External/Status interrupt. Any odd number of transitions on the $\overline{\text{CTS}}$ pin while another External/Status interrupt is pending also causes an External/Status interrupts condition. If the CTS IE bit is reset, it merely reports the current unlatched state of the $\overline{\text{CTS}}$ pin.

**Bit 4: SYNC/Hunt**

The operation of this bit is similar to that of the $\overline{\text{CTS}}$ bit, except that the condition monitored by the bit varies depending on the mode in which the SCC is operating.

When the XTAL oscillator option is selected in asynchronous modes, this bit is forced to "0" (no External/Status interrupt is generated). Selecting the XTAL oscillator in synchronous or SDLC modes had no effect on the operation of this bit. The XTAL oscillator should not be selected in External Sync mode.

In Asynchronous mode, the operation of this bit is identical to that of the CTS status bit, except that this bit reports the state of the $\overline{\text{SYNC}}$ pin.

In External sync mode the $\overline{\text{SYNC}}$ pin is used by external logic to signal character synchronization. When the Enter Hunt Mode command is issued in External Sync mode, the $\overline{\text{SYNC}}$ pin must be held High by the external sync logic until character synchronization is achieved. A High on the $\overline{\text{SYNC}}$ pin holds the Sync/Hunt bit in the reset condition.

When external synchronization is achieved, $\overline{\text{SYNC}}$ must be driven Low on the second rising edge of the Receive Clock after the last rising edge of the Receive Clock on which the last bit of the receive character was received. Once $\overline{\text{SYNC}}$ is forced Low, it is good practice to keep it Low until the CPU informs the external sync logic that synchronization has been lost or that a new message is about to start. Both transitions on the $\overline{\text{SYNC}}$ pin cause External/Status interrupts if the Sync/Hunt IE bit is set to "1."

The Enter Hunt Mode command should be issued whenever character synchronization is lost. At the same time, the CPU should inform the external logic that character synchronization has been lost and that the SCC is waiting for $\overline{\text{SYNC}}$ to become active.

In the Monosync and Bisync Receive modes, the Sync/Hunt status bit is initially set to "1" by the Enter Hunt Mode command. The Sync/Hunt bit is reset when the SCC established character synchronization. Both transitions cause External/ Status interrupts if the Sync/Hunt IE bit is set. When the CPU detects the end of message or the loss of character synchronization, the Enter Hunt Mode command should be issued to set the Sync/Hunt bit and cause an External/Status interrupt. In this mode, the $\overline{\text{SYNC}}$ pin is an output, which goes Low every time a sync pattern is detected in the data stream.

In the SDLC modes, the Sync/Hunt bit is initially set by the Enter Hunt Mode command or when the receiver is disabled. It is reset when the opening flag of the first frame is detected by the SCC. An External/Status interrupt is also generated if the Sync/Hunt IE bit is set. Unlike the Monosync and Bisync modes, once the Sync/Hunt bit is reset in SDLC mode, it does not need to be set when the end of the frame is detected. The SCC automatically maintains synchronization. The only way the Sync/Hunt bit can be set again is by the Enter Hunt Mode command or by disabling the receiver.

### Bit 3: Data Carrier Detect

If the DCD IE bit in WR15 is set, this bit indicates the state of the $\overline{\text{DCD}}$ pin the last time the Enabled External/Status bits changed. Any transition on the $\overline{\text{DCD}}$ pin while no interrupt is pending latches the state of the $\overline{\text{DCD}}$ pin and generates an External/Status interrupt. Any odd number of transitions on the $\overline{\text{DCD}}$ pin while another External/Status interrupt is pending also causes an External/Status interrupt condition. If the DCD IE is reset, this bit merely reports the current, unlatched state of the $\overline{\text{DCD}}$ pin.

### Bit 2: TX Buffer Empty

This bit is set to "1" when the transmit buffer is empty. It is reset while CRC is sent in a synchronous or SDLC mode and while the transmit buffer is full. The bit is reset when a character is loaded into the transmit buffer. This bit is always in the set condition after a hardware or channel reset.

### Bit 1: Zero Count

If the Zero Count Interrupt Enable bit is set in WR15, this bit is set to one while the counter in the baud rate generator is at the count of zero. If there is no other External/Status interrupt condition pending at the time this bit is set, an External/Status interrupt is generated. However, if there is another External/Status interrupt pending at this time, no interrupt is initiated until interrupt service is complete. If the Zero Count condition does not persist beyond the end of the interrupt service routine, no interrupt will be generated. This bit is not latched High, even thought the other External/Status latches close as a result of the Low-to-High transition on ZC. The interrupt service routine should check the other External/Status conditions for changes. If none changed, ZC was the source. In polled applications, check the IP bit in RR3A for a status change and then proceed as in the interrupt service routine.

### Bit 0: RX Character Avaliable

This bit is set to "1" when at least one character is available in the receive FIFO and is reset when the receive FIFO is completely empty. A channel or hardware reset empties the receive FIFO.

## 7.2.2 Read Register 1

RR1 contains the Special Receive Condition status bits and the residue codes for the I-field in SDLC mode. Figure 7-19 shows the bit positions for RR1.

### Bit 7: End of Frame (SDLC)

This bit is used only in SDLC mode and indicates that a valid closing flag has been received and that the CRC Error bit and residue codes are valid. This bit can be reset by issuing the Error Reset command. It is also updated by the first character of the following frame. This bit is reset in any mode other than SDLC.

### Bit 6: CRC/Framing Error

If a framing error occurs (in Asynchronous mode), this bit is set (and not latched) for the receive character in which the framing error occurred. Detection of a framing error adds an additional one-half bit to the character time so that the framing error is not interpreted as a new Start bit. In Synchronous and SDLC modes, this bit indicates the result of comparing the CRC checker to the appropriate check value. This bit is reset by issuing an Error Reset command, but the bit is never latched. Therefore, it is always updated when the next character is received. When used for CRC error status in Synchronous or SDLC modes, this bit is usually set since most bit combination, except for a correctly completed message, result in a non-zero CRC.

### Bit 5: Receiver Overrun Error

This bit indicates that the receive FIFO has overflowed. Only the character that has been written over is flagged with this error, and when the character is read, the Error condition is latched until reset by the Error Reset command. The overrun character and all subsequent characters received until the Error Reset command is issued causes a Special Receive Condition vector to be returned.

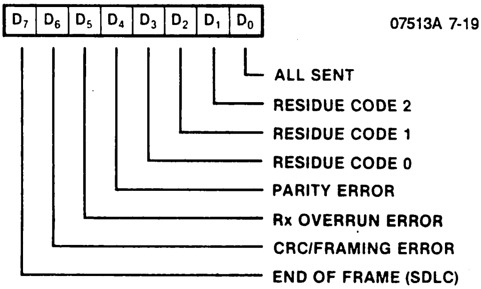


**Figure 7-19 Read Register 1**

### Bit 4: Parity Error

When parity is enabled, this bit is set for the characters whose parity does not match the programmed sense (even/odd). This bit is latched so that once an error occurs, it remains set until the Error Reset command is issued. If the parity in Special Condition bit is set, a parity error causes a Special Receive Condition vector to be returned on the character containing the error and on all subsequent characters until the Error Reset command is issued.

### Bits 3, 2, and 1: Residue Codes 2, 1, And 0

In those cases in SDLC mode where the received I-Field is not an integral multiple of the character length, these three bits indicate the length of the I-Field and are meaningful only for the transfer in which the end of frame bit is set. This field is set to "011" by a channel or hardware reset and is forced to this state in Asynchronous mode. These three bits can leave this state only if SDLC is selected and a character is received. The codes signify the following (Reference Table 7-9 when a receive character length is eight bits per character.

I-Field bits are right-justified in all cases. If a receive character length other than eight bits is used for the I-Field, a table similar to Table 7-9 can be constructed for each different character length. Table 7-10 shows the residue codes for no residue (The I-Field boundary lies on a character boundary).

**Table 7-9 I-Field Bit Selection (8 Bits Only)**

| Residue 2 | Residue 1 | Residue 0 | I-Field Bits in Last Byte | I-Field Bits in Previous Byte |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 3 |
| 0 | 1 | 0 | 0 | 4 |
| 1 | 1 | 0 | 0 | 5 |
| 0 | 0 | 1 | 0 | 6 |
| 1 | 0 | 1 | 0 | 7 |
| 0 | 1 | 1 | 0 | 8 |
| 1 | 1 | 1 | 1 | 8 |
| 0 | 0 | 0 | 2 | 8 |

**Table 7-10 Residue Bits/Character**

| Bits/Char | Residue | Residue | Residue |
|---|---|---|---|
| 8 | 0 | 1 | 1 |
| 7 | 0 | 0 | 0 |
| 6 | 0 | 1 | 0 |
| 5 | 0 | 0 | 1 |

### Bit 0: All Sent

In Asynchronous mode, this bit is set when all characters have completely cleared the transmitter pins. Most modems contain additional delays in the data path, which requires the modem control signals to remain active until after the data has cleared both the transmitter and the modem. This bit is always set in synchronous and SDLC modes.

## 7.2.3 Read Register 2

RR2 contains the interrupt vector written into WR2. When the register is accessed in Channel A, the vector returned is the vector actually stored in WR2. When this register is accessed in Channel B, the vector returned includes status information in bits 1, 2, and 3 or in bits 6, 5, and 4, depending on the state of the Status High/Status Low bit in WR9 and independent of the state of the VIS bit in WR9. The vector is modified according to Table 7-4 shown in the explanation of the VIS bit in WR9. If no interrupts are pending, the status is V3, V2, V1 = 011, or V6, V5, V4 = 110. Figure 7-20 shows the bit positions for RR2.

## 7.2.4 Read Register 3

RR3 is the Interrupt Pending register. The status of each of the Interrupt Pending bits in the SCC is reported in this register. This register exists only in Channel A. If this register is accessed in Channel B, all "0s" are returned. The two unused bits are always returned as "0." Figure 7-21 shows the bit positions for RR3.

## 7.2.5 Read Register 8

RR8 is the Receive Data register.

## 7.2.6 Read Register 10

RR10 contains some miscellaneous status bits. Unused bits are always "0." Bit positions for RR10 are shown in Figure 7-22.

### Bit 7: One Clock Missing

While operating in the FM mode, the DPLL sets this bit to "1" when it does not see a clock edge on the incoming lines in the window where it expects one. This bit is latched until reset by a Reset Missing Clock or Enter Search Mode command in

WR14. In the NRZI mode of operation and while the DPLL is disabled, this bit is always "0."

**Bit 6: Two Clocks Missing**

While operating in the FM mode, the DPLL sets this bit to "1" when it does not see a clock edge in two successive tries. At the same time the DPLL enters the Search mode. This bit is latched until reset by a Reset Missing Clock or Enter Search Mode command in WR10. In the NRZI mode of operation and while the DPLL is disabled, this bit is always "0."

**Bit 4: Loop Sending**

This bit is set to "1" in SDLC Loop mode while the transmitter is in control of the Loop, that is, while the SCC is actively transmitting on the loop. This bit is reset at all other times.

This bit can be polled in SDLC mode to determin when the closing flag has been sent.

**Bit 1: On Loop**

This bit is set to "1" while the SCC is actually on-loop in SDLC Loop mode. This bit is set to "1" in the X.21 mode (Loop mode selected while in monosync) when the transmitter goes active. This bit is "0" at all other times. This bit can also be pulled in SDLC mode to determine when the closing flag has been sent.

### 7.2.7 Read Register 12

RR12 returns the value stored in WR12, the lower byte of the time constant for the baud rate generator. Figure 7-23 shows the bit positions for RR12.

### 7.2.8 Read Register 13

RR13 returns the value stored in WR13, the upper byte of the time constant for the baud rate generator. Figure 7-24 shows the bit positions for RR13.

### 7.2.9 Read Register 15

RR15 reflects the value stored in WR15, the External/Status IE bits. The two unused bits are always returned as "0s." Figure 7-25 shows the bits positions for RR15.

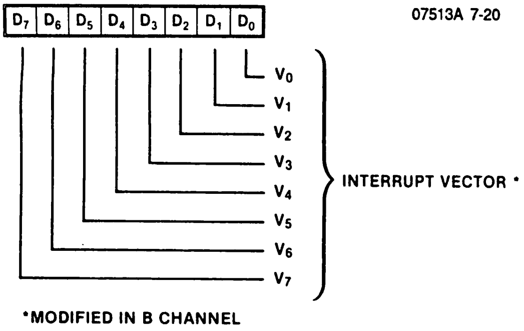


Figure 7-20 Read Register 2

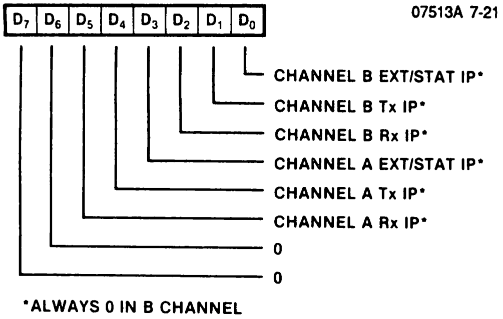


Figure 7-21 Read Register 3

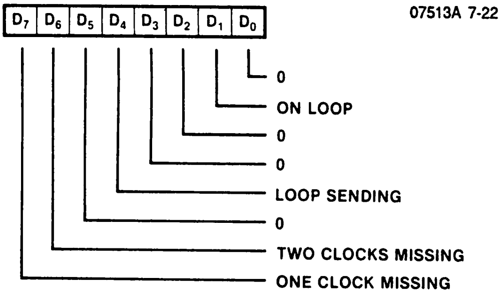


Figure 7-22 Read Register 10

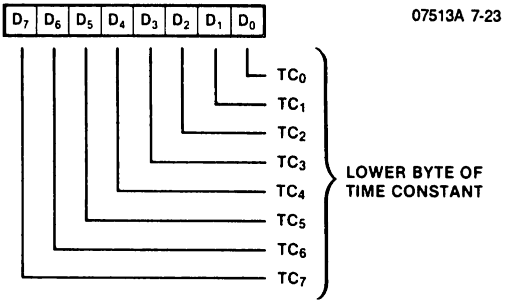


Figure 7-23 Read Register 12

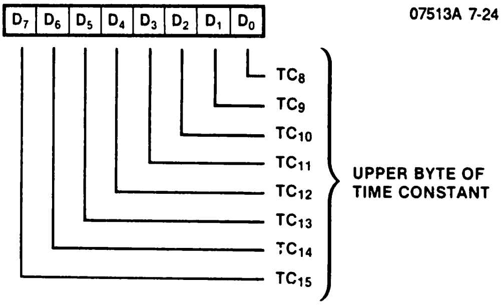


Figure 7-24 Read Register 13

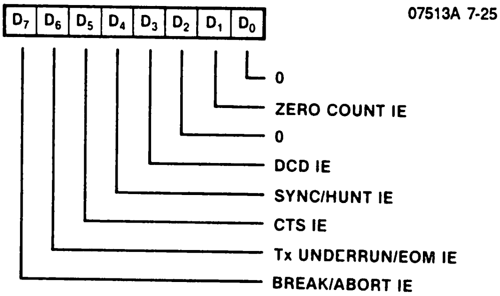


Figure 7-25 Read Register 15

CHAPTER 8

# SCC APPLICATION NOTES

## 8.1 Z8530 and Z8030 SCC Initialization: A Worksheet

### 8.1.1 Introduction

This application note describes the software initialization procedure for the Serial Communications Controller (SCC), and is applicable to both the Z8030 and the Z8530. Although the Z8030 and Z8530 have different bus interfaces, their registers are programmed in the same order.

Figure 8-1 provides a worksheet that can be used as an aid when initializing the SCC. Since all SCC operation modes are initialized in a similar manner, the worksheet can be used to tailor the SCC device to the user's individual need. Specific examples are given in the following chapters.

### 8.1.2 Register Overview

Each of the SCC's two channels has its own separate Write registers that are programmed to initialize different operating modes. There are two types of bits in the Write registers: Command bits and Mode bits. An example of a register that contains both types of bits is Write Register 9 (WR9), and is shown in Figure 8-2.

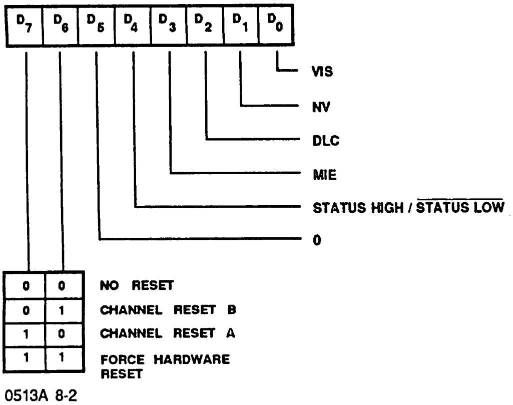


Figure 8-2 Write Register 2

WR9 is the Master Interrupt Control register and contains the Reset command bits. Command bits are denoted by having boxes drawn around them in register diagrams. Bit D 5 in this register is not used in this register and must be 0 at all times.

The Command bits, D7 and D6, select one of the reset commands for the SCC. Setting either of these bits to 1 disables both the receiver and the transmitter in the corresponding channel, forces TxD for the channel marking, forces the modem control signals High in that channel, resets all IPs and IUSs, and disables all interrupts in that channel. Functions controlled by the Command bits can only be enabled or disabled, they cannot be toggled.

Bits D4–D0 are Mode bits that can be enabled or disabled either by being set to "1" or reset to "0. Each Mode bit affects only one function. For example, Bit D1 is the No Vector mode bit, it controls whether or not the SCC will respond to an interrupt acknowledge cycle by placing a vector on the data bus. If this bit is set, no vector is returned. In Command bits entry, each new command requires a separate rewrite of the entire register. Care must be taken when issuing a command, so that the Mode bits are not changed accidentally.

### 8.1.3 Initialization Procedure

The SCC initialization procedure is divided into three parts. The first part consists of programming the operation modes (e.g., bits-per-character, parity) and loading the constants (e.g., interrupt vector, time constants). The second part enables the hardware functions (e.g., transmitter, receiver, baud-rate generator). It is important that the operating modes are programmed before the hardware functions are enabled. The third part, if required, consists of enabling the different interrupts.

Table 8-1 shows the order (from top to bottom) in which the SCC registers are to be programmed. Those registers that need not be programmed are listed as optional in the comments column. The bits in the registers that are marked with an "X" are to be programmed by the user. The bits marked with an "S" are to be set to their previous programmed value. For example, in part 2, Write Register 3, bits D1-D7 are shown with an "S" because they have been programmed in part 1 and must remain set to the same value.

### 8.1.4 Initialization Table Generation

Figure 8-1 is a worksheet for the initialization of the

Label of SCC Table: ______________________________ SCC Base Address ______________________________

Description: ____________________________________________________________

| | REGISTER | HEX | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | COMMENTS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MODES | WR9 | C | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | SOFTWARE RESET |
| | WR0 | 0 | | 0 | 0 | 0 | 0 | 0 | 0 | | | |
| | WR4 | | | | | | | | | | | |
| | WR1 | | | 0 | | | 0 | 0 | | 0 | 0 | |
| | WR2 | | | | | | | | | | | |
| | WR3 | | | | | | | | | | 0 | |
| | WR5 | | | | | | | 0 | | | | |
| | WR6 | | | | | | | | | | | |
| | WR7 | | | | | | | | | | | |
| | WR9 | | | 0 | 0 | 0 | | 0 | | | | |
| | WR10 | | | | | | | | | | | |
| | WR11 | | | | | | | | | | | |
| | WR12 | | | | | | | | | | | |
| | WR13 | | | | | | | | | | | |
| | WR14 | | | | | | | | | | 0 | |
| | WR14 | | | | | | | | | | 0 | |
| ENABLES | WR14 | | | 0 | 0 | 0 | | | | | 1 | |
| | WR3 | | | | | | | | | | 1 | |
| | WR5 | | | | | | | 1 | | | | |
| | WR0 | 8 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | RESET TxCRC |
| | WR1 | | | | | | | | | | | |
| INTERRUPT | WR15 | | | | | | | | | | | |
| | WR0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | RESET Ext/STATUS |
| | WR0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | RESET Ext/STATUS |
| | WR1 | | | | | | | | | | | |
| | WR9 | | | 0 | 0 | 0 | | | | | | |

0513A 8-1

Figure 8-1 SCC Initialization Worksheet

| REGISTER | HARDWARE RESET | | | | | | | | CHANNEL RESET | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| WR0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| WR1 | 0 | 0 | | 0 | 0 | | 0 | 0 | 0 | 0 | | 0 | 0 | | 0 | 0 |
| WR2 | | | | | | | | | | | | | | | | |
| WR3 | | | | | | | | 0 | | | | | | | | 0 |
| WR4 | | | | | | 1 | | | | | | | | 1 | | |
| WR5 | 0 | | | 0 | 0 | 0 | | | 0 | | | 0 | 0 | 0 | 0 | |
| WR6 | | | | | | | | | | | | | | | | |
| WR7 | | | | | | | | | | | | | | | | |
| WR9 | 1 | 1 | 0 | 0 | 0 | 0 | | | | | 0 | | | | | |
| WR10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | 0 | 0 | 0 | 0 |
| WR11 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | | | | | | | | |
| WR12 | | | | | | | | | | | | | | | | |
| WR13 | | | | | | | | | | | | | | | | |
| WR14 | | | 0 | 0 | 0 | 0 | 0 | 0 | | | 1 | 0 | 0 | 0 | | |
| WR15 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| RR0 | 0 | 0 | | | | 0 | 0 | 0 | 0 | 1 | | | | 1 | 0 | 0 |
| RR1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| RR3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RR10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

0513A 8-3

**Figure 8-3 Z8030 Register Reset Values**

*Part 1. Modes and Constants*

| | | |
|---|---|---|
| WR9 | 1100000 | Hardware Reset |
| WR0 | 000000XX | Select Shift M0de (8030 only) |
| WR4 | XXXXXXXX | Tx/Rx con, Async or Sync Mode |
| WR1 | 0XX00X00 | Select W/REQ (opt) |
| WR2 | XXXXXXXX | Program Interrupt Vector (opt) |
| WR3 | XXXXXXX0 | Select Rx Control |
| WR5 | XXXX0XXX | Selects Tx Control |
| WR6 | XXXXXXXX | Program sync character (opt) |
| WR7 | XXXXXXXX | Program sync character (opt) |
| WR9 | 000X0XXX | Select Interrupt Control |
| WR10 | XXXXXXXX | Miscellaneous Control (opt) |
| WR11 | XXXXXXXX | Clock Control |
| WR12 | XXXXXXXX | Time constant lower byte (opt) |
| WR13 | XXXXXXXX | Time constant upper byte (opt) |
| WR14 | XXXXXXX0 | Miscellaneous Control |
| WR14 | XXXSSSSS | Commands (opt) |

*Part 2. Enables*

| | | |
|---|---|---|
| WR14 | 000SSSS1 | Baud Rate Enable |
| WR3 | SSSSSSS1 | Rx Enable |
| WR5 | SSSS1SSS | Tx Enable |
| WR0 | 10000000 | Reset Tx CRG (opt) |
| WR1 | XSS00S00 | DMA Enable (opt) |

*Part 3. Interrupt Status*

| | | |
|---|---|---|
| WR15 | XXXXXXXX | Enable External/Status |
| WR0 | 00010000 | Reset External Status |
| WR0 | 00010000 | Reset External Status twice |
| WR1 | SSSXXSXX | Enable Rx, Tx and Ext/Status |
| WR9 | 000SXSSS | Enable Master Interrupt Enable |

| | |
|---|---|
| 1 = Set to one | X= User defined |
| 0 = Reset to zero | S = Same as previously prog. |

**Table 8-1 SCC Initialization Order**

SCC. All the bits that must be programmed as either a "0" or a "1" are already filled in; the remaining bits are left blank and are to be programmed by the user according to the desired mode of operation. The binary value can then be converted to a hexadecimal number and placed in the table, following the Write register notation in the column labeled "HEX". A Program Initialization Table is produced when this worksheet is completed.

### 8.1.5 Reset Conditions

Prior the initialization, the SCC should be reset by either hardware or software. A hardware reset can be accomplished by simultaneously grounding $\overline{RD}$ and $\overline{WR}$ on the Z8530, or $\overline{AS}$ and $\overline{DS}$ on the Z8030. A software reset can be executed by writing a 0H to Write Register 9. The state of the SCC registers, after reset, is shown in Figure 8-3.

## 8.2 Polled Asynchronous Mode

### 8.2.1 Introduction

This chapter describes the use of the SCC in polled Asynchronous Mode. The device can be set with 5 to 8 bits per character, 1, 1.5, or 2 stop bits, and a wide range of baud rates. In this particular example, 8 bits per character, 2 stop bits and 9600 baud rate are used. An external 2.4576 MHz, crystal oscillator is used for baud-rate generation. The SCC can be programmed for local loopback for on-board diagnostics. The user can make use of this feature to test-program the part without additional hardware to simiulate an actual transmit and receive environment.

### 8.2.2 SCC Interface

Figure 8-4 shows the SCC to CPU interface required for this application. The 8-bit data bus and control lines all come from the user's CPU. The 8030 control lines are $\overline{DS}$, $\overline{AS}$, R/$\overline{W}$, $\overline{CS0}$ and CS1. The 8530 control lines are $\overline{RD}$, $\overline{WR}$, A/$\overline{B}$, D/$\overline{C}$ and $\overline{CE}$. PCLK comes from the system clock, or an external crystal, up to the maximum rate of the SCC (ex. 6 MHz for the Z8530A). The IEI and the $\overline{INTACK}$ pins should be pulled up. The baud-rate generator clock is connected to the $\overline{RTxC}$ pin.

### 8.2.3 SCC Initialization

Initialization of the SCC for polled asynchronous communication is divided into two parts; part one programs the operating modes of the SCC and part two enables them. Care must be taken when writing the software to meet the SCC's Cycle and Reset Recovery times. The Cycle Recovery time, 6 PCLK cycles, applies to the period between any Read or Write cycles affecting the SCC. The Reset Recovery time is the period after a hardware reset caused either by hardware or software; this recovery time extends the Cycle Recovery time to 11 PCLK cycles. For more details about these recovery times, see the SCC Technical Manual (Chapter 3).

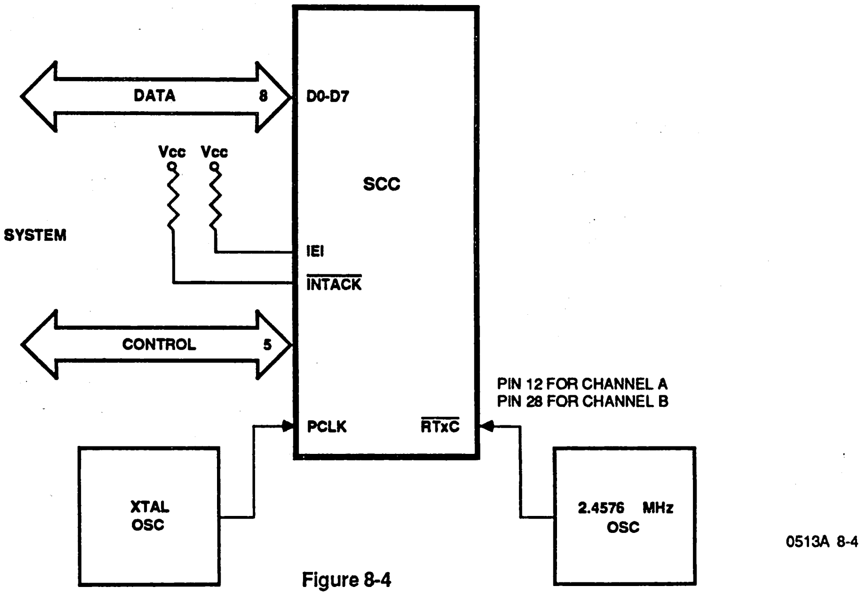


Figure 8-4

Table 8-2 **Polled Asynchronous Initialization Procedure**

| Register | Value | Comments |
|---|---|---|
| WR9 | C0H | Force Hardware Reset |
| WR4 | 4CH | x16 clock,2 stop bits,no parity |
| WR3 | C0H | R x 8 bits, Rx disabled |
| WR5 | 60H | T x 8 bits, DTR,RTS,Tx off |
| WR9 | 00H | Int. Disabled |
| WR10 | 00H | NRZ |
| WR11 | 56H | Tx & Rx = BRG out, TR x C = BRG out |
| WR12 | 06H | Time constant = 6 |
| WR13 | 00H | Time constant high =0 |
| WR14 | 10H | BRG in = RT x C, BRG off, loopback |
| | | **Enables** |
| WR14 | 11H | BRG enable |
| WR3 | C1H | Rx enable |
| WR5 | 68H | Tx enable |

### 8.2.3.1 SCC Operating Mode Programming

**WR9** resets the SCC to a known state by writing a C0 hex. The command, Force Hardware Reset, is identical to a hardware reset.

**WR4** selects the asynchronous, x 16 mode, with 2 stop bits and no parity. The x 16 mode means that clock rate is16 times the data rate.

**WR3** selects 8 bits per character and does not enable the receiver. The 8 bits per character allows 8 bits to be assembled from the data stream. The receiver is not enabled at this time because the SCC has not been initialized.

**WR5** selects 8 bits per character and does not enable the transmitter. The 8 bits per character allows 8 bits to be sent, as data, with the least significant bit first. The transmitter is not enabled at this time because the SCC has not been initialized.

**WR9** selects that there are no interrupts enabled. This inhibits the SCC from requesting an interrupt from the CPU.

**WR10** selects NRZ encoding. This NRZ coding is used on the transmitter as well as the receiver.

**WR11** selects the $\overline{\text{RTxC}}$ pin to TTL clock; the baud-rate generator is the transmit and receive clocks source, and the TRxC pin as a baud-rate generator output.

**WR12 & WR13** and select the baud-rate generator's time constant. The WR13 time constant is determined by the equation:

$$\text{Time Constant} = \frac{\text{Clock Frequency}}{2 \times \text{Baud Rate} \times \text{clock mode}} - 2$$

In this example, the clock frequency is 2.4576 MHz, the baud rate is 9600, and the clock mode is 16, the time constant is, therefore, 6; expressed as a 16-bit, hexadecimal number, it is 0006H. The time constant LOW (WR12) is, therefore, 06H and the time constant HIGH (WR13) is 00H. The baud rate for this example can be varied, as long as the data rate is less than 1/4 of the PCLK rate. Table 8-3 shows the time constants for other common baud rates.

**Table 8-3 Time constants for common baud rates**

| Baud | Divider | |
|---|---|---|
| Rate | Dec | Hex |
| 38400 | 0 | 0000H |
| 19200 | 2 | 0002H |
| 9600 | 6 | 0006H |
| 4800 | 14 | 000EH |
| 2400 | 30 | 001EH |
| 1200 | 62 | 003EH |
| 600 | 126 | 007EH |
| 300 | 254 | 00FEH |
| 150 | 510 | 01FEH |

*For 2.4576 MHz Clock, X16 Mode*

**WR14** selects the baud-rate generator as the $\overline{\text{RTxC}}$ pin, baud-rate generator disabled, and internal loopback. The baud-rate generator uses the $\overline{\text{RTxC}}$ pin as the clock source and is not enabled at this tim e because the SCC initialization is not complete.

### 8.2.3.2 SCC Operating Mode Enables

**WR14** enables the baud-rate generator. Bit 0 (LSB) is changed to a 1 to enable the baud-rate generator; all other bits must maintain the value selected during initialization.

**WR3** enables the receiver. Bit 0 (LSB) is changed to a 1 to enable the receiver, all other bits must maintain the value selected during initialization.

**WR5** enables the transmitter. Bit 3 is changed to a 1 to enable the transmitter, all other bits must maintain the value selected during initialization.

### 8.2.4 Transmit and Receive Routines

After initialization, and after all enables have been selected, the SCC is ready for communication.

The transmitter buffer and the receive FIFO are empty. The example shown below is coded to transmit and receive characters.

```
;Transmit a character

TXCHAR: INPUT   RR0      ;Read RR0
        TEST    BIT 2    ;Test transmit
                          buffer empty
        JZ      TXCHAR   ;Loop if not empty
        OUTPUT  CHAR     ;Output character to

                          data port
        RET              ;Return

;Receive a character

RXCHAR: INPUT   RR0      ;Read RR0
        TEST    BIT 0    ;Test Receive
buffer
        JZ      RXCHAR   ;Loop if not full
        INPUT   CHAR     ;Input character
                          from data port
        RET              ;Return
```

**Figure 8-5 Transmit and Receive Routine**

## 8.3 INTERRUPT WITHOUT INTACK ASYNCHRONOUS MODE

### 8.3.1 Introduction

This chapter describes the use of the SCC for interrupt-driven Asynchronous Mode. As with the example in the previous chapter, the SCC is set with 8 bits per character, 2 stop bits, at 9600 baud rate. An external 2.4576 MHz, crystal oscillator is used for baud-rate generation. Interrupt acknowledge is not generated because of the extra hardware required to produce this signal. In this chapter, the SCC is also programmed for local loopback so that no external loop between the transmit and the receive data lines is needed for on-board diagnostics. This feature allows the user to test-program the part without additional hardware to simiulate an actual transmit and receive environment.

### 8.3.2 SCC INTERFACE

Figure 8-6 shows the SCC to CPU interface required for this application. The 8-bit data bus and control lines all come from the user's CPU. For

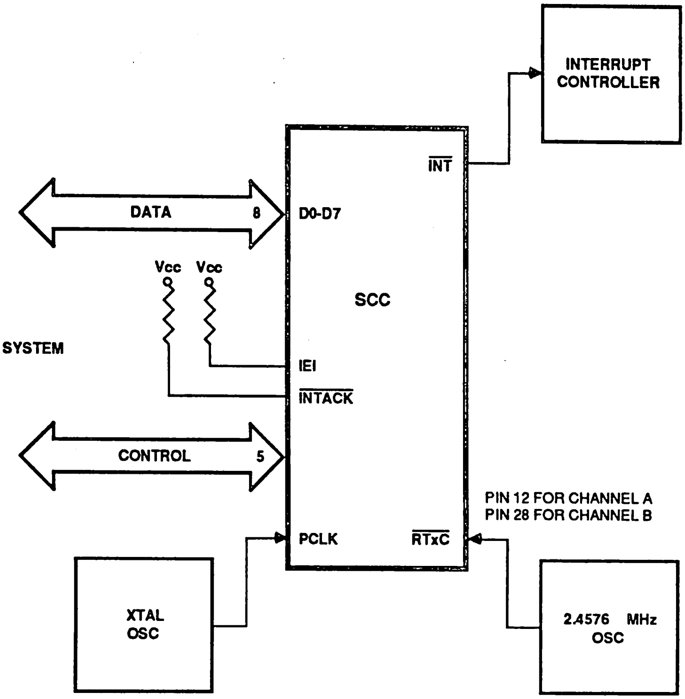


**Figure 8-6**

the 8030, the control lines are $\overline{DS}$, $\overline{AS}$, $R/\overline{W}$, $\overline{CS0}$ and CS1. For the 8530, the control lines are $\overline{RD}$, $\overline{WR}$, A/$\overline{B}$, D/$\overline{C}$ and $\overline{CE}$. The $\overline{INT}$ signal goes to an interrupt controller which must produce the interrupt vector to the CPU. The PCLK comes from the system clock, or an external crystal oscillator, up to the maximum rate of the SCC (ex. 6 MHz for the Z8530A). The IEI and the $\overline{INTACK}$ pins should be pulled up. The baud-rate generator clock is connected to the $\overline{RTxC}$ pin.

### 8.3.3 SCC INITIALIZATION

The initialization of the SCC for interrupt-driven asynchronous communication is divided into three parts. Part one programs the operating modes of the SCC, part two and three enable them. Care must be taken when writing the code to meet the SCC's Cycle and Reset Recovery times. The Cycle Recovery time applies to the period between any Read or Write cycles to the SCC, and is 6 PCLK cycles. The Reset Recovery time applies to a hardware reset caused either by hardware or software; this recovery time extends the Cycle Recovery time to 11 PCLK cycles. More details about these recovery times can be found in the SCC Technical Manual (Chapter 3).

Table 8-4

| Register | Value | Comments |
|---|---|---|
| WR9 | C0H | Force Hardware Reset |
| WR4 | 4CH | x16 clock, 2 stop bits, no parity |
| WR2 | 00H | Interrupt Vector 00WR3 |
| | C0H | Rx 8 bits, Rx disabled |
| WR5 | 60H | Tx 8 bits, DTR, RTS, Tx off |
| WR9 | 00H | Int DisabledWR10 |
| | 00H | NRZ |
| WR11 | 56H | Tx & Rx = BRG out, TRxC = BRG out |
| WR12 | 06H | Time constant = 6 |
| WR13 | 00H | Time constant high = 0 |
| WR14 | 10H | BRG in = RTxC, BRG off, |
| loopback | | |
| | | **Enables** |
| WR14 | 11H | BRG enable |
| WR3 | C1H | Rx enable |
| WR5 | 68H | Tx enable |
| | | **Enable Interrupts** |
| WR1 | 12H | Rx Int on all char and Tx Int enables |
| WR9 | 08H | MIE |

#### 8.3.3.1 SCC Operating Modes Programming

**WR9** resets the SCC to a known state by writing a C0 hex. This command, Force Hardware Reset, is identical to a hardware reset.

**WR4** selects asynchronous mode, x16 mode, 2 stop bits and no parity. The x16 mode means that the clock rate is 16 times the data rate.

**WR2** is the interrupt vector of the SCC. Even though a vector is mot placed on the bus in this mode the vector including status is read from RR2. By writing 00H to this register the status read will be the only bits set in RR2.

**WR3** selects 8 bits per character and does not enable the receiver. The 8 bits per character allows 8 bits to be assembled from the data stream. The receiver is not enabled at this time because the SCC is not completely initialized.

**WR5** selects 8 bits per character and does not enable the transmitter. The 8 bits per character allows 8 bits to be sent as data with the least significant bit first. The transmitter is not enabled at this time because the SCC is not completely initialized.

**WR9** selects that there are no interrupts enabled. This will inhibit the SCC from requesting an interrupt from the CPU.

**WR10** selects NRZ encoding. This selects NRZ coding is to be used on the transmitter and the receiver.

**WR11** selects the $\overline{RTxC}$ pin to TTL clock, the transmit and receive clocks source as the baud-rate generator and the TRxC pin as a baud-rate generator output.

**WR12 & WR13** select the baud-rate generators time constant. The time constant is determined by the equation:

$$\text{Time Constant} = \frac{\text{Clock Frequency}}{2 \times \text{Baud Rate} \times \text{clock mode}} - 2$$

In this example, the clock frequency is 2.4576 MHz, the baud rate is 9600, and the clock mode is 16; the time constant is 6. Converting this time constant to a 16-bit hexadecimal number, it becomes 0006H. The time constant LOW (WR12) is 06H and the time constant HIGH (WR13) is 00H. The baud rate for this example can be varied for as long as the data rate is less than 1/4 of the PCLK rate. Table 8-5 gives the time constants for other common baud rates.

**Table 8-5 Time constants for common baud rates**

| Baud Rate | Divider | |
|---|---|---|
| | Dec | Hex |
| 38400 | 0 | 0000H |
| 19200 | 2 | 0002H |
| 9600 | 6 | 0006H |
| 4800 | 14 | 000EH |
| 2400 | 30 | 001EH |
| 1200 | 62 | 003EH |
| 600 | 126 | 007EH |
| 300 | 254 | 00FEH |
| 150 | 510 | 01FEH |

*For 2.4576 MHz Clock, X16 Mode*

**WR14** selects the baud rate source as the $\overline{\text{RTxC}}$ pin, baud rate generator disabled, and internal loopback. The baud-rate generator will use the $\overline{\text{RTxC}}$ pin as the clock source for the baud-rate generator. The baud-rate generator is not enabled at this time because the SCC initialization is not complete.

### 8.3.3.2 SCC Operating Mode Enables

**WR14** enables the baud-rate generator. Bit 0 (LSB) is changed to a 1 to enable the baud-rate generator; all other bits must maintain the value selected during initialization.

**WR3** enables the receiver. Bit 0 (LSB) is changed to a 1 to enable the receiver; all other bits must maintain the value selected during initialization.

**WR5** enables the transmitter. Bit 3 is changed to a 1 to enable the transmitter; all other bits must maintain the value selected during initialization.

### 8.3.3.3 SCC Operating Mode Interrupts

**WR1** enables the Tx and the Rx interrupts. The Rx interrupt is programmed to generate an interrupt an all received characters or special conditions. This provides an interrupt on every character received by the SCC. The external/status interrupts are not enabled in this application.

**WR9** sets the master interrupt enable (MIE) bit 3. Setting this bit enables the interrupts pending to generate and interrupt on the INT pin.

### 8.3.4 Interrupt Routine

When the SCC has been initialized and enabled, it is ready for communication. The transmitter buffer and the receive FIFO are both empty. An interrupt will not be generated until the software writes the first character to the transmit buffer. Once the first character is in the SCC shift register, the first transmit interrupt will occur. The SCC will then keep setting transmit and receive interrupts to the interrupt controller until the end of the message. At the end of the message, a Reset Transmitter Interrupt Pending (WR0) is issued to clear the transmit interrupt. After the last character is read into the SCC, the interrupts will cease until another message is written into the transmitter.

Once an interrupt is received and the interrupt controller vectors to the interrupt routine, RR2 is read from channel B. The value read from RR2 is the vector, including status. This vector shows the status of the highest priority interrupt pending (IP) at the time it is read. Once the highest priority interrupt condition is cleared, RR2 will show the status of the next highest interrupt pending, if one is present. This allows multiple interrupts to be serviced without the overhead of the interrupt acknowledge cycle of the interrupt controller.

The following example shows how the interrupt routine should be coded.

```
BEGIN:   INPUT    RR2         ;Read RR2 from channel B
         TEST     Bit 4       ;Test for Tx Empty
         JE       TXEMPTY     ;Jump to Transmit Routine

         TEST     Bit 5       ;Test for Rx full
         JUMP     RXFULL      ;Jump to Receive Routine
         OUT      EOI         ;Output EOI to Interrupt Controller
         IRET                 ;Return to Main
;
;
TXEMPTY:TEST      NOMORE      ;Test a last character flag
         JE       LAST        ;Jump to LAST if no more characters
         OUTPUT   CHAR        ;Output character to data port
         DEC      CHARCOUNT   ;Decrement character count
         JUMP     BEGIN       ;Jump to BEGIN to test for more IP
;
LAST:    OUTPUT   RR0,28H     ;Reset Tx Interrupt Pending
         JUMP     BEGIN       ;Jump to BEGIN to test for more IP
;
;
RXFULL:  INPUT    RR1         ;Read RR1
         COMPARE  RR1,00      ;Test for special condition bit set
         JUMP     NE          ;Jump to SPECIAL

         INPUT    CHAR        ;Input character from data port
         JUMP     BEGIN       ;Jump to BEGIN to test for more IP
;
SPECIAL: .
         .

          This means a framing error, receive overrun error or parity error has occurred.
          Character may be read but data is not correct. A flag should be set to post the
          error.
         .
         .
         OUTPUT RR0,30H   ;Reset Error Command
         JUMP BEGIN       ;Jump to BEGIN to test for more IP
```

**Figure 8-7**

# INDEX

Zilog

# Reader Response

*Your feedback about this document helps us ascertain your needs and fulfill them in the future. Please take the time to fill out this questionnaire and return it to us. This information will be helpful to us, and to future users of Zilog products.*

□ First-time user reading document for general information

□ Experienced user needing specific information

Is there any additional information you would like presented about this product? □ Yes □ No

If Yes, what type of information?____________________________

What would have improved the material?____________________________

Do you have a unique application that could benefit other users of Zilog products? □ Yes □ No

If Yes, briefly describe____________________________

Other comments and suggestions____________________________

Your Name____________________________

Company Name____________________________

Address____________________________

City____________________ State__________

Zip________________ Phone (        )________________

□ Check here to receive a copy of our current literature list.

## ***ZILOG DOMESTIC SALES OFFICES AND TECHNICAL CENTERS***

| | |
|---|---|
| **CALIFORNIA** | |
| Agoura | 818-707-2160 |
| Campbell | 408-370-8000 |
| Costa Mesa | 714-261-1281 |
| **COLORADO** | |
| Boulder | 303-440-3971 |
| **FLORIDA** | |
| Largo | 813-585-2533 |
| **GEORGIA** | |
| Atlanta | 404-451-8425 |
| **ILLINOIS** | |
| Schaumburg | 312-885-8080 |
| **MASSACHUSETTS** | |
| Burlington | 617-273-4222 |
| **MINNESOTA** | |
| Edina | 612-831-7611 |
| **NEW JERSEY** | |
| Hasbrouck Heights | 201-288-3737 |
| Mt. Laurel | 609-778-8070 |
| **OHIO** | |
| Seven Hills | 216-447-1480 |
| **TEXAS** | |
| Richardson | 214-231-9090 |

## ***INTERNATIONAL SALES OFFICES***

| | |
|---|---|
| **CANADA** | |
| Toronto | 416-673-0634 |
| **GERMANY** | |
| Munich | 49-89-612-6046 |
| **JAPAN** | |
| Tokyo | 81-3-587-0528 |
| **HONG KONG** | |
| Kowloon | 852-3-723-8979 |
| **UNITED KINGDOM** | |
| Berkshire | 44-628-39200 |

Zilog, Inc. 1315 Dell Ave., Campbell, California 95008 Telephone (408)370-8000 TWX 910-338-7621

00-2057-03 Printed in USA