USING SCC WITH Z8000 IN SDLC PROTOCOL

INTRODUCTION

This application note describes the use of the Z8030 Serial Communications Controller (SCC) with the Z8000™ CPU to implement a communications controller in a Synchronous Data Link Control (SDLC) mode of operation. In this application, the Z8002 CPU acts as a controller for the SCC. This application note also applies to the non-multiplexed Z8530.

One channel of the SCC communicates with the remote station in Half Duplex mode at 9600 bits/second. To test this application, two Z8000 Development Modules are used. Both are loaded with the same software routines for initialization and for transmitting and receiving messages. The main program of one module requests the transmit routine to send a message of the length indicated by "COUNT" parameter. The other system receives the incoming data stream, storing the message in its resident memory.

DATA TRANSFER MODES

The SCC system interface supports the following data transfer modes:

- **Polled Mode.** The CPU periodically polls the SCC status registers to determine if a received character is available, if a character is needed for transmission, and if any errors have been detected.
- **Interrupt Mode.** The SCC interrupts the CPU when certain previously defined conditions are met.
- **Block/DMA Mode.** Using the Wait/Request (/W//REQ) signal, the SCC introduces extra wait cycles in order to synchronize the data transfer between a controller or DMA and the SCC.

The example given here uses the block mode of data transfer in its transmit and receive routines.

SDLC PROTOCOL

Data communications today require a communications protocol that can transfer data quickly and reliably. One such protocol, Synchronous Data Link Control (SDLC), is the link control used by the IBM Systems Network Architecture (SNA) communications package. SDLC is a subset of the International Standard Organization (ISO) link control called High-Level Data Link Control (HDLC), which is used for international data communications.

SDLC is a bit-oriented protocol (BOP). It differs from byte-control protocols (BCPs), such as Bisync, in that it uses only a few bit patterns for control functions instead of several special character sequences. The attributes of the SDLC protocol are position dependent rather than character dependent, so the data link control is determined by the position of the byte as well as by the bit pattern.

A character in SDLC is sent as an octet, a group of eight bits. Several octets combine to form a message frame, in which each octet belongs to a particular field. Each message contains: opening flag, address, control, information, Frame Check Sequence (FCS), and closing flag (Figure 1).

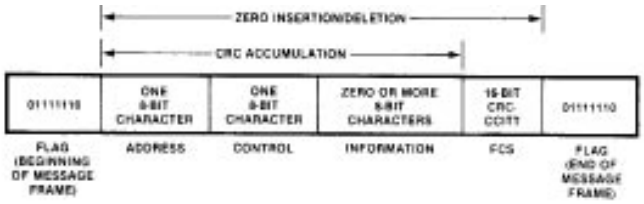

Figure 1. Fields of the SDLC Transmission Frame

Both flag fields contain a unique binary pattern, 0111110, which indicates the beginning or the end of the message frame. This pattern simplifies the hardware interface in receiving devices so that multiple devices connected to a common link do not conflict with one another. The receiving devices respond only after a valid flag character has been detected. Once communication is established with a particular device, the other devices ignore the message until the next flag character is detected.

The address field contains one of more octets, which are used to select a particular station on the data link. An address of eight 1s is a global address code that selects all the devices on the data link. When a primary station sends a frame, the address field is used to select one of several secondary stations. When a secondary station sends a message to the primary station, the address field contains the secondary station address, i.e., the source of the message.

The control field follows the address field and contains information about the type of frame being sent. The control field consists of one octet that is always present.

The information field contains any actual transferred data. This field may be empty or it may contain an unlimited number of octets. However, because of the limitations of the error-checking algorithm used in the frame-check sequence, however, the maximum recommended block size is approximately 4096 octets.

The frame check sequence field follows the information or control field. The FCS is a 16-bit Cyclic Redundancy Check (CRC) of the bits in the address, control, and information fields. The FCS is based on the CRC-CCITT code, which uses the polynomial ($x^{16} + x^{12} + x^5 + 1$). The Z8030 SCC contains the circuitry necessary to generate and check the FCS field.

Zero insertion and deletion is a feature of SDLC that allows any data pattern to be sent. Zero insertion occurs when five consecutive 1s in the data pattern are transmitted. After the fifth 1, a 0 is inserted before the next bit is sent. The extra 0 does not affect the data in any way and is deleted by the receiver, thus restoring the original data pattern.

Zero insertion and deletion insures that the data stream will not contain a flag character or abort sequence. Six 1s preceded and followed by 0s indicate a flag sequence character. Seven to fourteen 1s signify and abort; Seven to fourteen 1s signify an abort; 15 or more 1s indicate an idle (inactive) line. Under these three conditions, zero insertion and deletion are inhibited. Figure 2 illustrates the various line conditions.

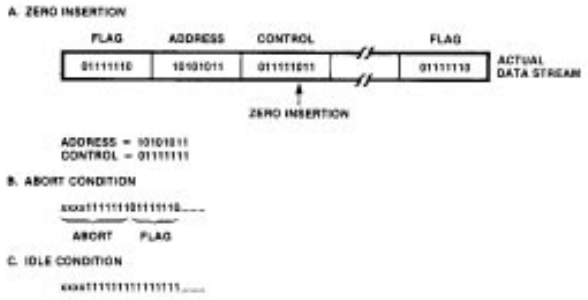

Figure 2. Bit Patterns for Various Line Conditions

The SDLC protocol differs from other synchronous protocols with respect to frame timing. In Bisync mode, for example, a host computer might temporarily interrupt transmission by sending sync characters instead of data. This suspended condition continues as long as the receiver does not time out. With SDLC, however, it is invalid to send flags in the middle of a frame to idle the line. Such action causes an error condition and disrupts orderly operation. Thus, the transmitting device must send a complete frame without interruption. If a message cannot be transmitted completely, the primary station sends an abort sequence and restarts the message transmission at a later time.

SYSTEM INTERFACE

The Z8002 Development Module consists of a Z8002 CPU, 16K words of dynamic RAM, 2K words of EPROM monitor, a Z80A SIO providing dual serial ports, a counter/timer channels, two Z80A PIO devices providing 32 programmable I/O lines, and wire wrap area for prototyping. The block diagram is depicted in Figure 3. Each of the peripherals in the development module is connected in a prioritized daisy chain configuration. The SCC is included in this configuration. The SCC is included in this configuration by tying its IEI line to the IEO line of another device, thus making it one step lower in interrupt priority compared to the other device.

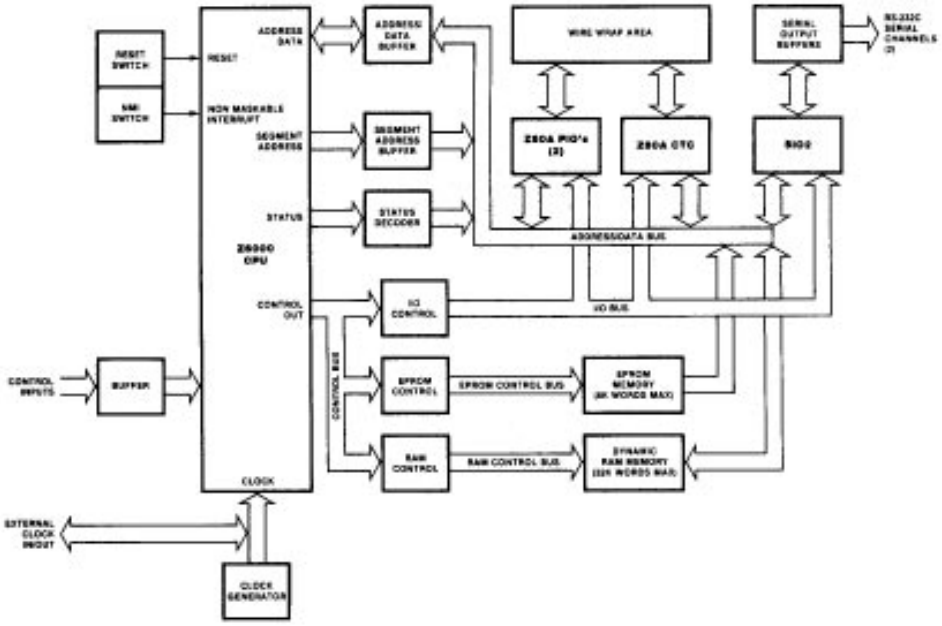

Figure 3. Block Diagram of Z8000 DM

SYSTEM INTERFACE (Continued)

Two Z8000 Development Modules containing SCCs are connected as shown in Figure 4 and Figure 5. The Transmit Data pin of one is connected to the Receive Data pin of the other and vice versa. The Z8002 is used as a host CPU for loading the modules; memories with software routines.

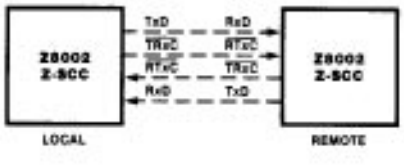

Figure 4. Block Diagram of Two Z8000 CPUs

The Z8002 CPU can address either of the two bytes contained in 16-bit words. The CPU uses an even address (16 bits) to access the most significant byte of a word and an odd address for the least significant byte of a word.

When the Z8002 CPU uses the lower half of the Address/Data bus (AD7-AD0 the least significant byte) for byte read and write transactions during I/O operations, these transactions are performed between the CPU and I/O ports located at odd I/O addresses. Since the SCC is attached to the CPU on the lower half of the A/D bus, its registers must appear to the CPU at odd I/O addresses. To achieve this, the SCC can be programmed to select its internal registers using lines AD5-AD1. This is done either automatically with the Force Hardware Reset command in WR9 or by sending a Select Shift Left Mode command to WR0B in channel B of the SCC. For this application, the SCC registers are located at I/O port address “Fexx”. The Chip Select signal (/CSO) is derived by decoding I/O address “FE” hex from lines AD15-AD8 of the controller.

To select the read/write registers automatically, the SCC decodes lines AD5-AD1 in Shift Left mode. The register map for the SCC is depicted in Table 1.

Table 1. Register Map

Address (Hex)	Write Register	Read Register
FE01	WR0B	RR0B
FE03	WR1B	RR1B
FE05	WR2	RR2B
FE07	WR3B	RR3B
FE09	WR4B	
FE0B	WR5B	
FE0D	WR6B	
FE0F	WR7B	
FE11	B DATA	B DATA
FE13	WR9	
FE15	WR10B	RR10B
FE17	WR11B	
FE19	WR12B	RR12B
FE1B	WR13B	RR13B
FE1D	WR14B	
FE1F	WR15B	RR15B
FE21	WR0A	RR0A
FE23	WR1A	RR1A
FE25	WR2	RR2A
FE27	WR3A	RR3A
FE29	WR4A	
FE2B	WR5A	
FE2D	WR6A	
FE2F	WR7A	
FE31	A DATA	A DATA
FE33	WR9	
FE35	WR10A	RR10A
FE37	WR11A	
FE39	WR12A	RR12A
FE3B	WR13A	RR13A
FE3D	WR14A	
FE3F	WR15A	RR15A

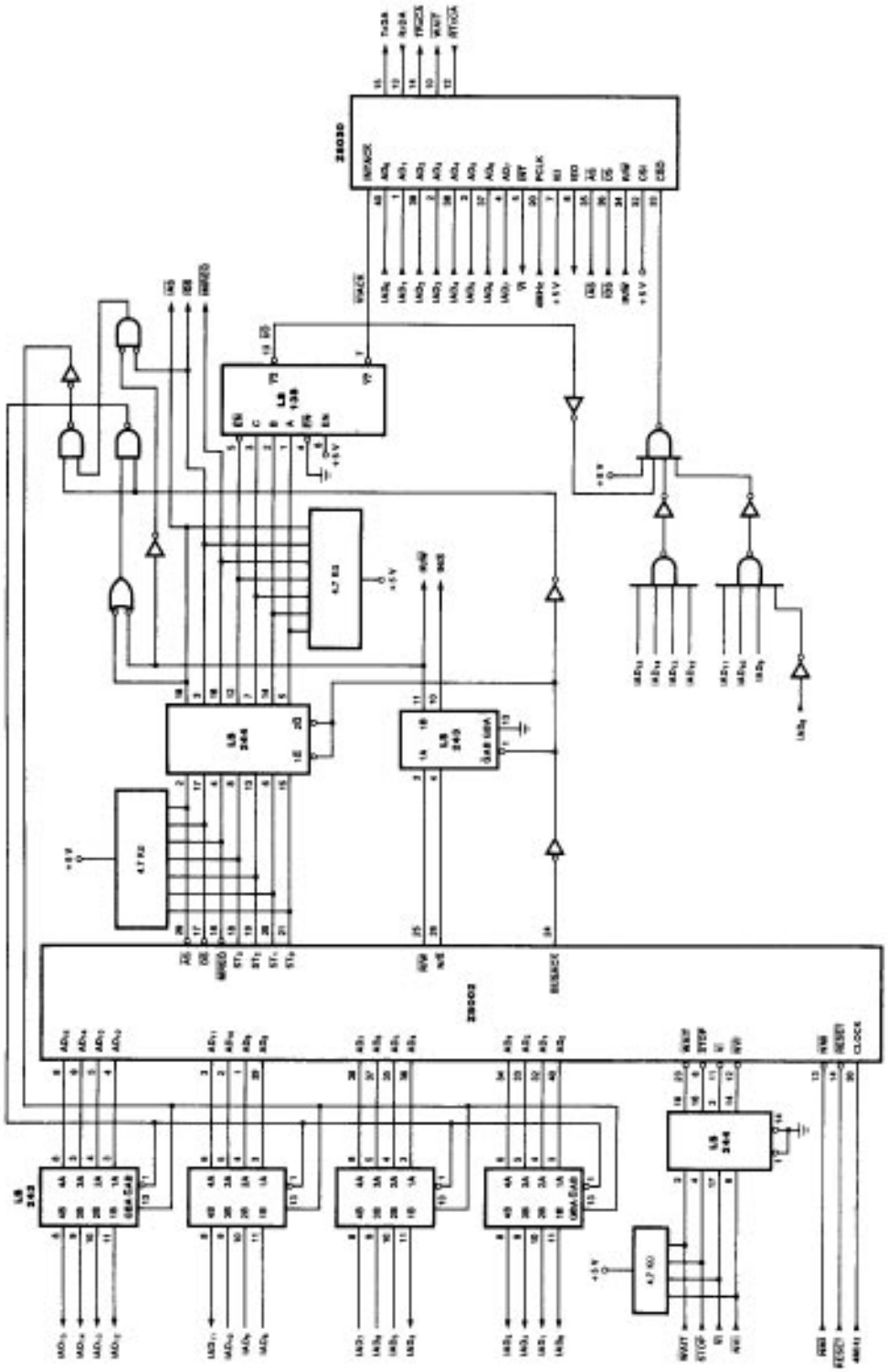

Figure 5. Z8002 With SCC

INITIALIZATION

The SCC can be initialized for use in different modes by setting various bits in its write registers. First, a hardware reset must be performed by setting bits 7 and 6 of WR9 to one; the rest of the bits are disabled by writing a logic zero.

SDLC protocol is established by selecting a SDLC mode, sync mode enable, and a x1 clock in WR4. A data rate of 9600 baud, NRZ encoding, and a character length of eight bits are among the other options that are selected in this example (Table 2).

Note that WR9 is accessed twice, first to perform a hardware reset and again at the end of the initialization sequence to enable the interrupts. The programming sequence depicted in Table 2 establishes the necessary parameters for the receiver and transmitter so that they are ready to perform communication tasks when enabled.

Table 2. Programming Sequence for Initialization

Register	Value (Hex)	Effect
WR9	C0	Hardware reset
WR4	20	x1 clock, SDLC mode, sync mode enable
WR10	80	NRZ, CRC preset to one
WR6	AB	Any station address e.g. “AB”
WR7	7E	SDLC flag (01111110) = “7E”
WR2	20	Interrupt vector “20”
WR11	16	Tx clock from BRG output, /TRxC pin = BRG out
WR12	CE	Lower byte of time constant = “CE” for 9600 baud
WR13	0	Upper byte = 0
WR14	03	BRG source bit =1 for PCKL as input, BRG enable
WR15	00	External Interrupt Disable
WR5	60	Transmit 8 bits/character SDLC CRC
WR3	C1	Rx 8 bits/character, Rx enable (Automatic Hunt mode)
WR1	08	ext int. disable
WR9	09	MIE, VIS, status Low

The Z8002 CPU must be operated in System mode to execute privileged I/O instructions. So the Flag and Control Word (FCW) should be loaded with system normal (S//N), and the Vectored Interrupt Enable (VIE) bits set. The Program Status Area Pointer (PSAP) is loaded with address %4400 using the Load Control Instruction (LDCTL). If the Z8000 Development Module is intended to be used, the PSAP need not be loaded by the programmer because the development module’s monitor loads it automatically after the NMI button is pressed.

Since VIS and Status Low are selected in WR9, the vectors listed in Table 3 will be returned during the Interrupt Acknowledge cycle. Of the four interrupts listed, only two, Ch A Receive Character Available and Ch A Special Receive Condition, are used in the example given here.

Table 3. Interrupt Vectors

Vector (Hex)	PS Address	Interrupt
28	446E	Ch A Transmit Buffer Empty
2A	4472	Ch A External Status Change
2C	4476	Ch A Receive Char. Available
2E	447A	Ch A Special Receive Condition

* Assuming that PSAP has been set to 4400 hex, “PS Address” refers to the location in the Program Status Area where the service routine address is stored for that particular interrupt.

TRANSMIT OPERATION

To transmit a block of data, the main program calls up the transmit data routine. With this routine, each message block to be transmitted is stored in memory, beginning with location “TBUF” The number of characters contained in each block is determined by the value assigned to the “COUNT” parameter in the main module.

To prepare for transmission, the routine enables the transmitter and selects the Wait On Transmit function; it then enables the wait function. The Wait on Transmit function indicates to the CPU whether or not the SCC is ready to accept data from the CPU. If the CPU attempts to send data to the SCC when the transmit buffer is full, the SCC asserts its /WAIT line and keeps it Low until the buffer is empty. In response, the CPU extends its I/O cycles until the /WAIT line goes inactive, indicating that the SCC is ready to receive data.

The CRC generator is reset and the Transmit CRC bit is enabled before the first character is sent, thus including all the characters sent to the SCC in the CRC calculation.

The SCC transmit underrun/EOM latch must be reset sometime after the first character is transmitted by writing a Reset Tx Underrun/EOM command to WR0. When this latch is reset, the SCC automatically appends the CRC characters to the end of the message in the case of an underrun condition.

Finally, a three-character delay is introduced at the end of the transmission, which allows the SCC sufficient time to transmit the last data byte and two CRC characters before disabling the transmitter.

RECEIVE OPERATION

Once the SCC is initialized, it can be prepared to receive the message. First, the receiver is enabled, placing the SCC in Hunt mode and thus setting the Sync/Hunt bit in status register RR0 to 1. In Hunt mode, the receiver searches the incoming data stream for flag characters. Ordinarily, the receiver transfers all the data received between flags to the receive data FIFO. If the receiver is in Hunt mode, however, no data transfer takes place until an opening flag is received. If an abort sequence is received, the receiver automatically re-enters Hunt mode. The Hunt status of the receiver is reported by the Sync/Hunt bit in RR0.

The second byte of an SDLC frame is assumed by the SCC to be the address of the secondary stations for which the frame is intended. The SCC provides several options for handling this address. If the Address Search Mode bit D2 in WR3 is set to zero, the address recognition logic is disabled and all the received data bytes are transferred to the receive data FIFO. In this mode, software must perform any address recognition. If the Address Search Mode bit is set to one, only those frames with addresses that match the address programmed in WR6 or the global address (all 1s) will be transferred to the receive data FIFO. If the Sync Character Load Inhibit bit (D1) in WR3 is set to zero, the address comparison is made across all eight bits of WR6. The comparison can be modified so that only the four most significant bits of WR6 need match the received address. This alterations made by setting the Sync Character Load Inhibit bit to one. In this mode, the address field is still eight bits wide and is transferred to the FIFO in the same manner as the data. In this application, the address search is performed.

When the address match is accomplished, the receiver leaves the Hunt mode and establishes the Receive Interrupt on First Character mode. Upon detection of the receive interrupt, the CPU generates an Interrupt Acknowledge Cycle. The SCC returns the programmed vector %2C. This vector points to the location %4472 in the Program Status Area which contains the receive interrupt service routine address.

The receive data routine is called from within the receive interrupt service routine. While expecting a block of data, the Wait on Receive function is enabled. Receive read buffer RR8 is read and the characters are stored in memory location RBUF. The SCC in SDLC mode automatically enables the CRC checker for all data between opening and closing flags and ignores the Receive CRC Enable bit (D3) in WR3. The result of the CRC calculation for the entire frame in RR1 becomes valid only when the End of Frame bit is set in RR1. The processor does not use the CRC bytes, because the last two bits of the CRC are never transferred to the receive data FIFO and are not recoverable.

When the SCC recognizes the closing flag, the contents of the Receive Shift register are transferred to the receive data FIFO, the Residue Code (not applicable in this application) 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, a special receive condition interrupt occurs. The special receive condition register RR1 is read to determine the bit is zero, the frame received is assumed to be correct; if the bit is 1, an error in the transmission is indicated.

Before leaving the interrupt service routine, Reset Highest IUS (Interrupt Under Service), Enable Interrupt on Next Receive Character, and Enter Hunt Mode commands are issued to the SCC.

RECEIVE OPERATION (Continued)

If receive overrun error is made, a special condition interrupt occurs. The SCC presents vector %2E to the CPU, and the service routine located at address %447A is executed. Register RR1 is read to determine which error occurred. Appropriate action to correct the error should be taken by the user at this point. Error Reset and Reset Highest IUS commands are given to the SCC before returning to the main program so that the other low-priority interrupts can occur.

In addition to searching the data stream for flags, the receiver also scans for seven consecutive 1s, which indicates an abort condition. This condition is reported in the Break/Abort bit (D7) in RR0. This is one of many possible external status conditions. As a result transitions of this bit can be programmed to cause an external status interrupt. The abort condition is terminated when a zero is received, either by itself or as the leading zero of a flag. The receiver leaves Hunt mode only when a flag is found.

SOFTWARE

Software routines are presented in the following pages. These routines can be modified to include various other options (e.g., SDLC Loop, Digital Phase Locked Loop etc.). By modifying the WR10 register, different encoding methods (e.g., NRZI, FM0, FM1) other than NRZ can be used.

Appendix

Software Routines

```
plzasm   1.3
LOC    OBJ CODE     STMT SOURCE STATEMENT

                      1
                      2
                      3          SDLC MODULE
                        $LISTON $TTY
                        CONSTANT
                        WRDA     :=  %FE21              !BASE ADDRESS FOR WR0 CHANNEL A!
                        RRDA     :=  %FE21              !BASE ADDRESS FOR RR0 CHANNEL A!
                        RBUF     :=  %5400              !BUFFER AREA FOR RECEIVE CHARACTER!
                        PSAREA   :=  %4400              !START ADDRESS FOR PROGRAM STAT AREA!
                        COUNT    :=   12                !NO. OF CHAR. FOR TRANSMIT ROUTINE!
0000                    GLOBAL MAIN PROCEDURE
                        ENTRY

0000 7601                       LDA     R1,PSAREA
0002 4400
0004 7D1D                       LDCTL   PSAPOFF,R1      !LOAD PSAP!
0006 2100                       LD      R0,#%5000
0008 5000
000A 3310                       LD      R1(#%1C),R0     !FCW VALUE(%5000) AT %441C FOR VECTORED!
000C 001C                                               !INTERRUPTS!
000E 7600                       LDA     R0,REC
0010 00D6'
0012 3390                       LD      R1(#%76),R0     !EXT. STATUS SERVICE ADDR. AT %4476 IN!
0014 0076                                               !PSA!
0016 7600                       LDA     R0,SPCOND
0018 00FA'
001A 3310                       LD      R1(#%7A),R0     !SP.COND.SERVICE ADDR AT %447A IN PSA!
001C 007A
001E 5F00                       CALL    INIT
0020 0034'
0022 5F00                       CALL    TRANSMIT
0024 008C'
0026 EBFF                       JR      $

0028 AB                 TBUF:   BVAL    %AB             !STATION ADDRESS!
0029 48                         BVAL    'H'
002A 45                         BVAL    'E'
002B 4C                         BVAL    'L'
002C 4C                         BVAL    'L'
002D 4F                         BVAL    'O'
002E 20                         BVAL    ' '
002F 54                         BVAL    'T'
0030 48                         BVAL    'H'
0031 45                         BVAL    'E'
0032 52                         BVAL    'R'
0033 45                         BVAL    'E'

0034                            END     MAIN
```

SOFTWARE (Continued)

```
                        !****************** INITIALISATION ROUTINE FOR Z-SCC ***********************!

0034                            GLOBAL  INIT PROCEDURE
                                ENTRY
0034 2100                       LD      R0,#15          !NO.OF PORTS TO WRITE TO!
0036 00DF
0038 7602                       LDA     R2,SCCTAB       !ADDRESS OF DATA FOR PORTS!
003A 004E'
003C 2101               ALOOP:  LD      R1,#WR0A
003E FE21
0040 0029                       ADDB    RL1,#R2
0042 A920                       INC     R2
0044 3A22                       OUTIB   @R1,@R2,R0      !POINT TO WR0A,WR1A ETC THRO LOOP!
0046 0018
0048 8D04                       TEST    R0              !END OF LOOP?!
004A EEFB                       JR      NE,ALOOP        !NO,KEEP LOOPING!
004C 9E08                       RET
004E 12                 SCCTAB: BVAL    2*9
004F C0                         BVAL    %C0             !WR9=HARDWARE RESET!
0050 08                         BVAL    2*4
0051 20                         BVAL    %20             !WR4=X1 CLK,SDLC,SYNC MODE!
0052 14                         BVAL    2*10
0053 80                         BVAL    %80             !WR10=CRC PRESET ONE,NRZ,FLAG ON IDLE,!
                                                            !FLAG ON UNDERRUN!
0054 0C                         BVAL    2*6
0055 AB                         BVAL    %AB             !WR6= ANY ADDRESS FOR SDLC STATION!
0056 0E                         BVAL    2*7
0057 7E                         BVAL    %7E             !WR7=SDLC FLAG CHAR!
0058 04                         BVAL    2*2
0059 20                         BVAL    %20             !WR2=INT VECTOR %20!
005A 16                         BVAL    2*11
005B 16                         BVAL    %16             !WR11=Tx CLOCK & TRxC OUT=BRG OUT!
005C 18                         BVAL    2*12
005D CE                         BVAL    %CE             !WR12= LOWER TC=CE!
005E 1A                         BVAL    2*13
005F 00                         BVAL    0               !WR13= UPPER TC=0!
0060 1C                         BVAL    2*14
0061 03                         BVAL    %03             !WR14=BRG ON,BRG SRC=PCLK!
0062 1E                         BVAL    2*15
0063 00                         BVAL    %00             !WR15=EXT INT. DISABLE!
0064 0A                         BVAL    2*5
0065 60                         BVAL    %60             !WR5=Tx 8 BITS/CHAR, SDLC CRC!
0066 06                         BVAL    2*3
0067 C5                         BVAL    %C5             !WR3=ADDR SRCH,REC ENABLE!
0068 02                         BVAL    2*1
0069 08                         BVAL    %08             !WR1=RX INT ON 1ST & SP COND,!
                                                            !EXT INT DISABLE!
006A 12                         BVAL    2*9
006B 09                         BVAL    %09             !WR9= NIE,VIS,STATUS LOW!
006C                    END     INIT


                        !****************** RECEIVE ROUTINE ************************************!

                        !                   RECEIVE A BLOCK OF MESSAGE                          !

006C                            GLOBAL  RECEIVE PROCEDURE
                                ENTRY
006C C828                       LDB     RL0,#%28        !WAIT ON RECV.!
006E 3A86                       OUTB    WR0A+2,RL0
0070 FE23
0072 6008                       LDB     RL0,%A8
0074 00A8
0076 3A86                       OUTB    WR0A+2,RL0      !ENABLE WAIT FNC. SP. COND. INT!
0078 FE23
007A 2101                       LD      R1,#RR0A+16
007C FE31
007E 2102                       LD      R2,#COUNT+2     !COUNT+2 CHARACTERS TO READ!
0080 000E
0082 2103                       LD      R3,#RBUF        !RECEIVE BUFFER IN MEMORY!
0084 5400
0086 3A18                       INDRB   @R3,@R1,R2      !READ THE ENTIRE MESSAGE!
0088 0230
008A 9E08                       RET
008C                    END RECEIVE
```

```
                        !**************  TRANSMIT ROUTINE ***********************************!
                        !               SEND A BLOCK OF EIGHT  DATA CHARACTERS               !
                        !               THE BLOCK STARTS AT LOCATION TBUF                    !

008C                            GLOBAL  TRANSMIT PROCEDURE
                                ENTRY
008C 2102                               LD      R2,#TBUF        !PTR TO START OF BUFFER!
008E 0028'
0090 C868                               LDB     RL0,#%68
0092 3A86                               OUTB    WR0A+10,RL0     !ENABLE TRANSMITTER!
0094 FE2B
0096 C800                               LDB     RL0,#%00        !WAIT ON TRANSMIT!
0098 3A86                               OUTB    WR0A+2,RL0
009A FE23
009C C888                               LDB     RL0,#%88
009E 3A86                               OUTB    WR0A+2,RL0      !WAIT ENABLE!
00A0 FE23
00A2 C880                               LDB     RL0,#%80
00A4 3A86                               OUTB    WR0A,RL0        !RESET TxCRC GENERATOR!
00A6 FE21
00A8 2101                               LD      R1,#WR0A+16     !WR8A SELECTED!
00AA FE31
00AC 2100                               LD      R0,#1
00AE 0001
00B0 C869                               LDB     RL0,#%69        !SDLC CRC!
00B2 3A86                               OUTB    WR0A+10,RL0     !WR5A-TxCRC ENABLE!
00B4 FE2B
00B6 3A22                               OTIRB   @R1,@R2,R0      !SEND ADDRESS!
00B8 0010
00BA C8C0                               LDB     RL0,#%C0
00BC 3A86                               OUTB    WR0A,RL0        !RESET TxUND/EOM LATCH!
00BE FE21
00C0 2100                               LD      R0,#COUNT-1
00C2 000B
00C4 3A22                               OTIRB   @R1,@R2,R0      !SEND MESSAGE!
00C6 0010
00C8 2100                               LD      R0,#926         !CREATE DELAY BEFORE DISABLING!
00CA 039E
00CC F0B1                       DEL:    DJNZ    R0,DEL          !TRANSMITTER SO THAT CRC CAN BE!
00CE C8D0                               LDB     RL0,#0          !SENT!
00D0 3A86                               OUTB    WR0A+10,RL0     !DISABLE TRANSMITTER!
00D2 FE2B
00D4 9ED8                               RET
00D6                            END TRANSMIT


                        !************** RECEIVE INT. SERVICE ROUTINE ************************!

00D6                              GLOBAL REC PROCEDURE
                                  ENTRY
00D6 93F3                               PUSH    @R15,R3
00D8 93F2                               PUSH    @R15,R2
00DA 93F1                               PUSH    @R15,R1
00DC 93F0                               PUSH    @R15,R0
00DE 3A84                               INB     RL1,RR0A        !READ STATUS REG RR0A!
00E0 FE21
00E2 A690                               BITB    RL1,#0          !TEST IF Rx CHAR SET!
00E4 E602                               JR      Z,RESET         !YES CALL RECEIVE ROUTINE!
00E6 5FD0                               CALL    RECEIVE
00E8 006C'
00EA C838                       RESET:  LDB     RL0,#%38
00EC 3A86                               OUTB    WR0A,RL0        !RESET HIGHEST IUS!
00EE FE21
00F0 97F0                               POP     R0,@R15
00F2 97F1                               POP     R1,@R15
00F4 97F2                               POP     R2,@R15
00F6 97F3                               POP     R3,@R15
00F8 7B00                               IRET
00FA                            END REC
```

RECEIVE OPERATION (Continued)

```
                ;************* SPECIAL CONDITION INTERRUPT SERVICE ROUTINE ***************;

00FA                    GLOBAL SPCOND PROCEDURE
                        ENTRY

00FA 93F0                       PUSH    @R15,R0
00FC 3A84                       INB     RL0,RRDA+2      ;READ ERRORS;
00FE FE23
0100 A607                       BITB    RL0,#7          ;END OF FRAME ?;
                        ;PROCESS OVERRUN,FRAMING ERRORS IF ANY;
0102 E603                       JR      Z,RESE
0104 C820                       LDB     RL0,#%20
0106 3A86                       OUTB    WRDA,RL0        ; YES,ENABLE INT ON NEXT RBC CHAR;
0108 FE21
010A C830               RESE:   LDB     RL0,#%30
010C 3A86                       OUTB    WRDA,RL0        ;ERROR RESET;
010E FE21
0110 C808                       LDB     RL0,#%08
0112 3A86                       OUTB    WRDA+2,RL0      ;WAIT DISABLE,RxINT ON 1ST OR SP COND.;
0114 FE23
0116 C838                       LDB     RL0,#%38
0118 3A86                       OUTB    WRDA,RL0        ;RESET HIGHEST IUS;
011A FE21
011C 97F0                       POP     R0,@R15
011E 7B00                       IRET

0120                    END SPCOND

                        END SDLC
```