— 88C0022-001 DOCUMENT CONTROL —
MASTER Rev. B

Z80182 EVALUATION BOARD USER MANUAL

TABLE OF CONTENTS

Z80182 Evaluation Board
September 1992

This manual provides the user instructions for the "Z80182 Evaluation Board" from Zilog. The Z80182 EV Board was designed to aid evaluators and potential users of the Z80182 modem controller. The Z8018200ZC is based around the Z80182 processor and supplies the user with an excellent hardware example of how to design with the Z80182. The board is provided with a debug monitor, ESCC device driver, and MIMIC device driver. The debug monitor provides an easy way to download and run programs assembled in the Intel hex format. The ESCC and MIMIC drivers demonstrate the various operations of internal Z80182 devices.

Two EPROMs are provided with the Z80182. One EPROM contains the ESCC Driver and the Debug Monitor for stand-alone use. The other EPROM contains a MIMIC Driver for use with the Z80182 evaluation board when connected to the PC/XT/AT bus.

GENERAL DESCRIPTION

The Z80182 EV Board has the following configurations:

- IBM-PC plug-in I/O card configuration. Host side is connected to 550 MIMIC interface and complies with the IBM-PC I/O card standard.
- Stand-alone system configuration. In this case the Z80182 Ch. B is active instead of the 550 MIMIC interface.

The Z80182 EV Board includes the following hardware (Reference one page schematic diagram at the rear of this manual):

- Zilog Z80182 modem controller;
- 28-pin EPROM socket, suitable for 2764 through 27512 devices;
- 32-pin (or 28-pin) SRAM socket, suitable for 8kx8, 32kx8, or 64kx8 devices;
- Glue logic between PC/XT/AT bus and 550 MIMIC interface;
- Two RS-232C line drivers and receivers
- One hard-wired DB-25 socket for ESCC channel A.
- PC back plate DB-25 socket and ribbon cable for ASCI channel 0 use in IBM-PC plug-in card configuration.
- DB-25 socket and ribbon cable for ASCI channel 0 use in stand-alone system configuration.
- One RS-422 line driver, receiver, and Appletalk connector.
- Several jumpers for configuring the Z80182 EV Board.
- One 28-pin 32kx8 SRAM
- Two EPROMs;
  1. ESCC driver + Debug Monitor
  2. MIMIC driver + Debug Monitor

1. INSTALLATION

The first step to using the Z80182 evaluation board is to install appropriate jumpers and connect the board to a terminal and PC bus if appropriate. Lets start by loading the evaluation board with RAM and ROM memory.

1.1 Memory configuration.

EPROM
28-pin EPROM socket is provided. Jumper headers J13 and J14 allows the socket to be compatible with 2764, 27128, 27256, or 27512 EPROMS.

For 27512 connect jumper J13-3 to J13-2 & J14-2 to J14-3 and leave open J13-1 & J14-1.
For 27256 connect jumper J13-2 to J13-3 & J14-1 to J14-2 and leave open J13-1 & J14-3.
For 27128 connect jumper J13-1 to J13-2 & J14-1 to J14-2 and leave open J13-3 & J14-3.
For 2764 connect jumper J13-1 to J13-2 & J14-1 to J14-2 and leave open J13-3 & J14-3.

When shipping, this board has 2-27512 EPROMs (or, 27C512). The jumpers J13-2 and J13-3 are shunted along with J14-2 and J14-3 in order to effectively use the 27512 EPROMs. Each EPROM contains a debug monitor. In addition to that, the EPROMs also contain a device driver. One EPROM has a ESCC driver while the other has a MIMIC driver. The physical addresses for the EPROMs are listed below:

EPROM type	Physical Addresses
2764	00000 to 01FFF
27128	00000 to 03FFF
27256	00000 to 07FFF
27512	00000 to 07FFF and 10000 to 17FFF

It is recommended that MIMIC/Debug EPROM is used in PC Plug-in configuration and ESCC/Debug EPROM is used in stand-alone configuration!

SRAM
32-pin SRAM socket is provided. VCC is provided at both 32 and 30 pin so 28-pin 32kx8 SRAM can be installed in the socket. Jumper header J9 also allows the socket to be compatible with 8kx8 SRAM.

For 8kx8 SRAM connect jumper J9-1 to J9-2 and leave open J9-3.
For 32kx8 SRAM connect jumper J9-2 to J9-3 and leave open J9-1.
For 64kx8 SRAM connect jumper J9-2 to J9-3 and leave open J9-1.

When shipping, this board has 32kx8 bit type SRAM in the RAM socket (U3) location and J9 has shunt between J9-2 and J9-3. The physical RAM address range is shown below:

SRAM type	Physical Addresses
64K Bit	08000 to 09FFF
256K Bit	08000 to 0FFFF

1.2 Emulation mode selection

EV1 and EV2 inputs to the Z80182 controller determine the emulation mode for Z8S182 MPU. For normal operation, use the jumper settings shown below for mode 0.

Mode 0 - connect J10-1 to J10-2 and connect J10-3 & J10-4. Use this setting for normal operation;
Mode 1 - connect J10-3 to J10-4 and leave open J10-1 & J10-2;
Mode 2 - connect J10-1 to J10-2 and leave open J10-3 & J10-4;
Mode 3 - leave open J10-1 & J10-2 & J10-3 & J10-4;

1.3 IBM-PC plug-in configuration (for MIMIC/Debug EPROM)

The evaluation board can be used in stand alone configuration or IBM-PC plug-in card configuration. If you are using the MIMIC\Debug EPROM, the board should be configured for IBM-PC Plug-in configuration. If you are using the ESCC\Debug EPROM, skip this section and set the board in stand-alone configuration (Section 1.5).

To set the board for IBM-PC plug in configuration, connect J4-1 to J4-2 and leave open J11-1 & J11-2. Also select the com port and interrupt level that the evaluation board will respond to by setting jumpers on J1 & J2 accordingly. For more information on jumper settings, refer to sections below labelled "Jumper Settings for PC Plug-In Configuration".

Jumper Settings for PC Plug-In Configuration

1. Address selection

Jumper J1 is used to set the address for the 550 MIMIC interface. Be sure that only one shunt is set. Note that leaving all eight connectors open will disable the port, if desired.

Shunt Position			com port	550 Mimic Port Address	
J1-1	to	J1-2		0E8-0EF	- reserved not use
J1-3	to	J1-4		0F8-0FF	- use only if no numeric co-processor
J1-5	to	J1-6	4	1E8-1EF	- use only if no COM4: installed
J1-7	to	J1-8	2	1F8-1FF	- use only if no COM2: installed
J1-9	to	J1-10		2E8-2EF	- use only for PC/AT
J1-11	to	J1-12		2F8-2FF	- use only for PC/AT
J1-13	to	J1-14	3	3E8-3EF	- use only if no COM3: installed
J1-15	to	J1-16	1	3F8-3FF	- use only if no COM1: installed

2. Interrupt level selection

Jumper J2 selects the interrupt level for the 550 MIMIC interface. IRQ3, IRQ4, IRQ5, IRQ10, or IRQ11 can be selected, depending on shunt position. If no interrupt is desired, remove the jumper.

SHUNT POSITION			550 MIMIC INTERRUPT LEVEL
J2-1	to	J2-2	IRQ5
J2-3	to	J2-4	IRQ4
J2-5	to	J2-6	IRQ3
J2-7	to	J2-8	IRQ10
J2-9	to	J2-10	IRQ11

IRQ10 and IRQ11 usage requires PC/AT bus connection.

3. DMA channel selection
To enable DMA transfer mode for the 550 MIMIC interface connect J5-1 to J5-2.
There are two sets of jumpers here. Jumper J6 position selects the DACK for the transmitter and receiver. Jumper J7 position selects the DREQ for the transmitter and receiver. DMA Ch1, Ch3, Ch5, or Ch7 can be selected depending on the shunt positions. If DMA is not used these jumpers may be removed.

SHUNT POSITION (*)			550 MIMIC TX DMA CHANNEL
J6-1	to	J6-2	DACK Ch1
J7-1	to	J7-2	DREQ Ch1
J6-3	to	J6-4	DACK Ch3
J7-3	to	J7-4	DREQ Ch3
J6-5	to	J6-6	DACK Ch5
J7-5	to	J7-6	DREQ Ch5
J6-7	to	J6-8	DACK Ch7
J7-7	to	J7-8	DREQ Ch7

(*) use only one shunt for J6 and one shunt for J7 when selecting TX DMA channel

SHUNT POSITION (*)			550 MIMIC RX DMA CHANNEL
J6-1	to	J6-2	DACK Ch1
J7-1	to	J7-2	DREQ Ch1
J6-3	to	J6-4	DACK Ch3
J7-3	to	J7-4	DREQ Ch3
J6-5	to	J6-6	DACK Ch5
J7-5	to	J7-6	DREQ Ch5
J6-7	to	J6-8	DACK Ch7
J7-7	to	J7-8	DREQ Ch7

(*) use only one shunt for J6 and one shunt for J7 when selecting RX DMA channel

4. Reset and NMI
J16 and J17 headers enable the user to bring Reset and NMI switches outside the PC. Connect switches to J16 and J17 and route these switches to a convenient location outside the PC.
This allows the user to reset the Z182 MPU without resetting the PC or opening the case.

The Z80182 evaluation board can be connected directly into the ISA or XT bus socket of an IBM PC. Although the board has an ISA (16-bit bus) connector labeled P2, it does not need to be connected unless PC IRQ 10, IRQ 11, DMA Ch5, or DMA Ch7 is used for the evaluation board. Therefore, the evaluation board can also be connected by only P1 (XT bus connector) on a XT bus socket.

PC installation is very simple, first remove the PC case, remove the blank metal slot cover, and insert the board. Since the MIMIC Driver and Debug Monitor also require an extra terminal to control the MPU, a PC Plate connector is provided to route the J16 ASCI channel 0 headers to an external DB-25 connector that can be installed on an empty PC slot. A terminal can then be connected to the PC Plate with a standard RS-232 connector. To install the PC Plate connector, remove another blank metal slot cover and install the included DB-25 connector plate. Attach the ribbon cable to the headers of J16. Make sure that the red pin 1 marking on the ribbon cable is aligned with pin 1 of the J16 headers. Then replace the PC cover, and you are done.

1.5 Stand-alone system (for ESCC/Debug EPROM)

The evaluation board can also be used in stand-alone configuration with a terminal and power supply. To select this configuration connect J4-2 to J4-3 and J11-1 to J11-2. The Z80182 EV Board can be configured to use Z182s ASCI Channel 0 or ESCC Channel B for connection to terminal. All software provided in this kit is programmed to utilize ASCI Channel 0, so set jumpers for ASCI Channel 0 usage unless your custom application requires ESCC Channel B.

To enable the console on the Z80182 EV Board	**Connect Jumpers as follow:**
Z182 ACSI Ch. 0	J3-3 to J3-1, J3-4 to J3-2 header pins 5 thru 18 are left open.
ESCC Ch. B	J3-3 to J3-5, J3-4 to J3-6 J3-1 open, J3-2 open header pins 7 thru 14 are left open.

Also, connect J11-1 to J11-2 for Stand-Alone configuration!

The evaluation board kit should contain a ribbon cable that connects the headers located on P8 to a standard DB-25 connector. Connect this cable to the RS-232 port on your terminal. Make sure that the pin 1 marking on the ribbon cable is aligned with pin 1 of the P8 headers. Also, connect a 5V power supply and GND to the evaluation board by attaching the included power cable to both P7 and the power supply and you are done.

An IBM PC can also be used with the evaluation board if terminal emulator software is used. There are many terminal emulation programs available today. If you plan to use them, set the software for ANSI mode 9600 baud, 8-n-1. The evaluation kit includes a very simple terminal emulation program. The program TERM.EXE provides facility for simple terminal emulation and allows downloading of user code in Intel Hex format. The Z80182 board can be connected to either COM1 or COM2 of the IBM PC.

Type TERM to invoke terminal emulation program. A '>' prompt will appear on the terminal. When there is no configuration file 'TERM.CFG', the terminal program will prompt for configuration parameters, which will be saved in 'TERM.CFG'.

Alternately, port configuration may be specified on the command line as:

```
TERM  p  bbbb
```

where p (=1 or 2) specifies the port (COM1 or COM2) and bbbb is the baud rate. The board is configured for 9600 baud, 8 data bits, 1 stop bit, and no parity.

1.5 Serial interfacing.

The serial I/O pins of the Z80182 serial controller are connected to the J12 and J3 connector blocks. The jumpers connected on block J3 determine what signals come out of P8 and J16 (either ASCI channel 0 or ESCC channel B). The evaluation kit provides 2 connectors for console use. One ribbon cable attaches to the evaluation board port labeled J16, and connects this port to a PC back-plate DB-25 connector for the PC plug-in card configuration. This allows access to J16 even when the PC casing is put on.

The other cable connects P8 to a standard RS-232 connector for use in stand alone configuration. Make sure that the ribbon cable pin 1 marking (shown by a red streak across one side of ribbon cable) is oriented such that the marking is aligned with pin 1 of the P8 or J16 header.

The jumpers on J12 determine what signals come out of P3 (ESCC Channel A signals). Since clock and handshaking can be routed to P3, this hard-wired DB-25 DTE connector can be used for synchronous as well as asynchronous serial communications.

The 8-pin circular DIN connector, P14, is compatible with Apple macintosh Plus, later Macintoshes, and with other AppleTalk/LocalTalk equipment. The J15 jumper controls whether the RS-422 driver for Transmit Data is turned "on" and "off" under control of the associated Request to Send signal, as on the Mac, or is "on" full time which is suitable for the use of DB25-pin RS-422 connector. The evaluation board provides 2 sockets for RS-422 drivers/receivers. U11 is for 75ALS194 driver and U12 is for 75ALS195 receiver.

TECHNICAL DESCRIPTION

The serial interface RS-232C option is selected when the RS-232C line driver and receiver are installed at locations U13 (MAX 238) and the RS-422 chips removed (U11, U12). The RS-422 option is installed when the RS-422 chips are installed at U12 (75ALSO195 RCVR) and U11 (75ALSO194 DRIVER) and the RS-232C chip removed. **Only install one set of drivers/receivers or the other, never both.**

Direct memory access (DMA) can be used to transfer data at very high rate in 550 MIMIC mode. This requires extensive programming and very good understanding of operation of the PC hardware. Z80182/ESCC Ch. A serial interface can use two Z8S182/DMA channels. For transfer in DMA mode connect Z80812/ESCC Ch. A DMA request outputs to the Z8S182/DMA request inputs (P6-15 to P6-17 and P6-20 to P6-18).

P3 Connector Pinout:
(DB25-pin female DTE connector mounted on the board)

SIGNAL NAME		PIN#	mode	
GND	Ground	7		
TD	Transmit Data	2	output	RS-232
RTS	Request To Send	4	output	RS-232
DTR	Data Terminal Ready	20	output	RS-232
TXC	Transmit Clock	24	output	RS-232
RD	Receive Data	3	input	RS-232
CTS	Clear To Send	5	input	RS-232
DCD	Data Curry Detect	6	input	RS-232
RXC	Receive Clock	15	input	RS-232
TX+	Transmit Positive	14	output	RS-422
TX-	Transmit Negative	2	output	RS-422
RTS+	Request To Send Pos.	4	output	RS-422
RTS-	Request To Send Neg.	19	output	RS-422
DTR+	Data Terminal Ready Pos.	23	output	RS-422
DTR-	Data Terminal Ready Neg.	20	output	RS-422
TXC+	Transmit Clock Pos.	24	output	RS-422
TXC-	Transmit Clock Neg.	11	output	RS-422
RX+	Receive Positive	16	input	RS-422
RX-	Receiver Negative	3	input	RS-422
CTS+	Clear To Send Pos.	13	input	RS-422
CTS-	Clear To Send Neg	5	input	RS-422
DCD+	Data Curry Detect Pos.	22	input	RS-422
DCD-	Data Curry Detect Neg.	6	input	RS-422
RXC+	Receive Clock Pos.	12	input	RS-422
RXC-	Receive Clock Neg.	15	input	RS-422

ZILOG

Z80182 DEBUG MONITOR

2. DEBUG MONITOR

The Z80182 evaluation board contains the Debug Monitor on two separate EPROMs. Place one of these EPROMs in the EPROM socket. The monitor source code is provided to allow you the option to upgrade if you find a problem. Zilog regrets that it does not support the monitor although it is fully operational to the best of our knowledge.

The debug monitor provides facilities to download a program from PC, to run the program with/without break point, to display/fill memory locations, to compare contents of two different memory blocks, to display/modify registers, and to read/write from/to a port.

All addresses from logical address 0D400h upwards are free to the user. The users stack pointer is initialized to 0D400h, and there are 182h bytes free for his stack below this address.

The breakpoint routine address and all of the command interpreter addresses are initialized in RAM at power up. These can be modified by the user if he really feels a burning urge.

Before Debug Monitor can work, the evaluation board must be configured correctly. Zilog Debug Monitor runs at 9.216MHz in clock divide-by-two mode. Make sure X1 is loaded with a 18.432 MHz crystal. The Debug Monitor is initialized to use ASCI Channel 0 as a console in both Stand-Alone and PC Plug-in Configuration. When the Debug Monitor is configured for PC Plug-in , you still need another terminal device connected to ASCI channel 0 for use as a Debug Monitor Console. Refer to Installation section before attempting to use Debug Monitor.

2.1 OPENING MESSAGE:

If the connection with your terminal or PC is correct, you should see following opening message by pressing RESET button on the board.

```
                                   Zilog
                         Z80182 Evaluation Card Monitor
                                Version 2.0b
DEBUG MONITOR  >>
```

2.2. COMMANDS

The monitor will prompt for all commands. The prompt for an EPROM resident monitor is "DEBUG MONITOR >>". If you type in some incorrect statement and want to get out of a command at any point you can hit the escape key. If the monitor doesn't recognize a command it will beep at you. Character case is un-important.

Numerical values are expected to be in hex. If a non-hex character is input the command will be aborted. You can type as many digits as you like, the monitor will pick as many bits as it needs from the least significant end of what you have typed. If you don't type a value (but just press return) then a zero value is assumed.

NOTE: If you want to exit from the Debug Monitor while using TERM.EXE , type <CTRL C> on PC keyboard to exit to DOS.

2.2.1 ALTER MEMORY - A

The alter command allows modification of bytes in logical memory. The display shows the address and current data and the display direction. When a byte is entered it is written to the address shown, and the next address in the appropriate direction is displayed. The t (toggle) key changes the scroll direction. The t is not echoed, but the direction indicator is changed to the opposite sign.If the toggle key is typed after entering some data but before <cr>, the data is NOT written to memory.

Example:

```
DEBUG MONITOR >> Alter Memory From: 8000
8000 FF +
8001 FF + aa
8002 FF + t (not echoed and line is overwritten with the next line)
8002 FF -
8001 AA - <esc>
DEBUG MONITOR >>
```

2.2.2 SET A BREAKPOINT - B

Only one breakpoint can be set at any one time. The code at the break address supplied is replaced with the RST 038h instruction (opcode 0ffh). When a breakpoint is hit, it is automatically reset and the users registers are displayed. The following will be displayed when <B> is pressed:

Example:

```
DEBUG MONITOR >> Breakpoint At What Address? abcd
DEBUG MONITOR >> Go From Address: 9000

A  F  B  C  D  E  H  L  A' F' B' C' D' E' H' L' I  IX   IY   SP   PC
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 00 FFFF FFFF 2000 ABCD
```

(CAUTION: This command is operational upon proper code is downloaded)

2.2.3 COMPARE MEMORY - C

Compare a specified number of memory bytes and display the differences. The following will be displayed when <C> is pressed:

Example:

```
DEBUG MONITOR >> Compare Data At Address: 0000
           With Data At Address: 8000
      How Many Bytes To Compare? 3
0000 = 18 : 8000 = 00
0001 = 3C : 8001 = FF
0002 = 00 : 8002 = FF
DEBUG MONITOR >>
```

2.2.4 DISPLAY MEMORY - D

Displays memory bytes in hex and ASCII format. The display can be paused by typing ^S. Typing a further ^S while paused displays one more line. Any other key quits the pause. The <esc> key stops the display after the current line has been printed.

Example:

```
DEBUG MONITOR >> Display Memory From : 0
                 How Many bytes? 20
0000 18 3C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 *.<..............*
0010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 *................*
DEBUG MONITOR >>
```

2.2.5 FILL MEMORY - F

Fill from memory address for range with specified byte. The following will be displayed when <F> is pressed:

Example:

```
DEBUG MONITOR >> Fill Memory From: 9000
          How Many bytes? 28
          With What Data? 55
DEBUG MONITOR >> Display Memory From : 9000
              How Many bytes? 30
9000 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 *UUUUUUUUUUUUUUUU*
9010 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 *UUUUUUUUUUUUUUUU*
9020 55 55 55 55 55 55 55 55 FF FF FF FF 00 00 00 00 *UUUUUUUU........*
DEBUG MONITOR >>
```

(CAUTION:
D000-D400 is used by the Debug Monitor as workspace. F and M commands will not work if the command is applied within this memory range)

2.2.6 GO COMMAND - G

Execute code. If an address is supplied, it is used as the users PC for execution. If no address is supplied the value that stored in the PC register is used as the starting address.

Example:

```
DEBUG MONITOR >> Breakpoint At What Address? 9008
DEBUG MONITOR >> Go From Address: 9000

A  F  B  C  D  E  H  L  A' F' B' C' D' E' H' L' I  IX   IY   SP   PC
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 00 FFFF FFFF 2000 9008
```

If you try to Go from an address which has been specified as the breakpoint a warning message is displayed.

2.2.7 HELP - H

Typing "h" displays the following:

```
DEBUG MONITOR >>Help Display

 A - (A)lter memory from logical address
 B - set a (B)reakpoint
 C - (C)ompare logical addresses
 D - (D)isplay logical memory
 F - (F)ill logical memory
 G - (G)o from logical address
 H - (H)elp screen (thicko!)
 I - (I)nput byte from 16-bit I/O address
 L - (L)oad an intel hex file from the host
 J - (J)ump to Device Driver Software
 M - (M)ove memory blocks
 O - (O)utput byte to 16-bit I/O address
 R - display/modify cpu (R)egisters
 U - specify the (U)ART delay
 V - display the software (V)ersion
 X - e(X)amine the memory map

DEBUG MONITOR >>
```

2.2.8 INPUT FROM ADDRESS - I

This performs an 8-bit input from a 16-bit I/O address. The following will be displayed when <I> is pressed:

Example:

```
DEBUG MONITOR >> Input from a Port Address : 00c0
00C0 : 74
DEBUG MONITOR >>
```

2.2.9 JUMP TO DEVICE DRIVER SOFTWARE - J

This command will switch the Z80182 evaluation board from 'Debug Monitor' mode to 'Device Driver Demonstration Software' mode. Two EPROMs are provided in the Z80182 evaluation kit. One EPROM contains Debug Monitor with Mimic Device Driver. The other EPROM contains the same Debug Monitor along with a ESCC Device Driver. The Device driver is addressed starting from 2000h. Therefore, you can also jump to device driver by executing <G> and entering 2000 as start address.

2.2.10 LOADING AN INTEL HEX FILE - L

This command is intercepted by the PC and on hitting return after the command line the PC will sent the required file to the board. If you have any grief at all h it a random combination of <esc> and <cr> until you get beeped at.

Example:

```
DEBUG MONITOR >> l program.hex
DEBUG MONITOR >>
```

2.2.11 MOVING MEMORY - M

Move specified number of bytes from one address to another.

Example:

```
DEBUG MONITOR >> Move Memory From: 3
                             To: 9000
          How Many Bytes? 1
DEBUG MONITOR >>
```

(CAUTION:
D000-D400 is used by the Debug Monitor as workspace. F and M commands will not work if the command is applied within this memory range)

2.2.12 OUTPUT TO ADDRESS - O

This performs an 8-bit output to the a 16-bit I/O address.

Example:

```
DEBUG MONITOR >> Output to a Port Address : c3
                   Data to Write : 00
DEBUG MONITOR >>
```

2.2.13 REGISTER DISPLAY AND MODIFICATION - R

You will be asked for a starting register. If no register name is supplied all CPU registers will be displayed. If a register name is supplied, display starts from that register, and the values can be modified. <cr> moves the display on to the next register, and <esc> stops the display. The PC cannot be modified using the R command.

Example:

```
DEBUG MONITOR >> Display/Modify Registers (cr for all) :
A  F  B  C  D  E  H  L  A' F' B' C' D' E' H' L' I   IX   IY   SP   PC
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 00 FFFF FFFF 6000 8000
DEBUG MONITOR >> Display/Modify Registers (cr for all) : h'
H'   FF : 00
L'   FF : 00
I    00 : <esc>
Display/Modify Registers (cr for all) :
A  F  B  C  D  E  H  L  A' F' B' C' D' E' H' L' I   IX   IY   SP   PC
FF FF FF FF FF FF FF FF FF FF FF FF FF FF 00 00 00 FFFF FFFF 6000 8000
DEBUG MONITOR >>
```

2.2.14 UART DELAY - U

The monitor starts up with a delay inserted prior to any character output. This is to allow slower PC's to keep up with the display. If you are using an AT or a Terminal, this delay can be removed because they are fast enough to cope with 9600 Baud without any delays. The only options are Y or N.

Example:

```
DEBUG MONITOR >>Is padding needed for host comms? (y or n): No, UART is
            full speed
DEBUG MONITOR >>
```

2.2.15 VERSION NUMBER - V

Typing V displays the software version number:

```
DEBUG MONITOR >>Z80182 Evaluation Card Monitor Version 1.2
DEBUG MONITOR >>
```

2.2.16 EXAMINE MMU MAP - X

This command is to understand Z182's MMU. By giving the values for the MMU control registers (CBAR, CBR and BBR), you will get the map of the physical memory.

Example:

```
DEBUG MONITOR >> X
Examine MMU register values
Value for MMU Common/Bank Area Register(CBAR: Addr 3Ah) ?00

Value for MMU Bank Base Register       (BBR:  Addr 39h) ?30

Value for MMU Common Base Register     (CBR:  Addr 38h) ?20

                  Above            Mapped to
                  Logical addr     Physical Addr
Common Area 1 :       00000h ==> 30000h
Bank Area     :       Do not exist
```

```
Common Area 0 :          Do not exist
```

If the value for CBAR is not a reasonable number, then you will have following message, and returns to the prompt for CBAR:

```
Value for MMU Common/Bank Area Register(CBAR: Addr 3Ah) ?0f
Upper nibble must be grater than or equal to Lower nibble!
Value for MMU Common/Bank Area Register(CBAR: Addr 3Ah) ?
```

You must then enter another value for CBAR.

2.2.17 OTHER FEATURES

(a) NMI

If you press NMI button, you will have the following message, and return to monitor.

```
NMI received !!!
A  F  B  C  D  E  H  L  A' F' B' C' D' E' H' L' I   IX   IY   SP   PC
FC 7C 00 00 81 53 00 BF 62 04 01 0E 90 00 80 89 00 904A 800F 827A 09AA
DEBUG MONITOR >>
```

(b) TRAP

Z182 has a feature to cause "TRAP" by executing undefined instruction. Upon detection of it, you will have the following message, and return to monitor.

```
Trap !!!
Undefined instruction found at location 9000
```

(c) Using channels other than ASCI0 for console

The source is provided in the file '182demon.s'. To modify this configuration in order to use ESCC Channel A or B, go to the file '182demon.s'. At the front of this file you will find the line;

```
escc_com:      equ     00h      ; using ESCC for comm.
                                ; If you want ESCC Channel B for comm,
                                ; change 00h to 001h!
```

Change the equate to 001h, the monitor will initialize the ESCC Ch.B at 9600 baud, 8-bits per character, 1 stop bit, no parity. If you want to set console for ESCC Ch.A look for the following code in the beginning of '182demon.s':

```
escc_a:        equ     000h     ;using escc ch b for comm.
                                ;If you want escc ch a for comm,
                                ;change 00h to 001h!
```

For both cases, X'tal in X1 location has to be 19.6608 MHz. Jumpers must also be changed to reflect the console used. Refer to Installation section for more details on jumper settings.

(d) RAM resident Debug Monitor

You can configure the Debug Monitor to load itself into RAM. This feature allows for easy modification of the Debug Monitor code. In the file named '182demon.s', the first page should include the statement below. Change this variable and recompile code for RAM resident operation.

```
ram_res:    equ  000h        ;If you want to make it as RAM resident,
                             ;change 000h to 001h!
```

ZILOG

ESCC DEVICE DRIVER DEMONSTRATION PACKAGE

3. ESCC DEVICE DRIVER DEMONSTRATION SOFTWARE

3.1 ESCC Device Driver Demonstration Program User Guide

The ESCC Device Driver Demonstration program supports Zilog's internal ESCC channel A (Enhanced Serial Communication Controller) on the Z80182. The Device Driver gives the user the ability to work with and learn the various operations of the ESCC. Registers can be programmed at the bit level, frames can be sent and received, and options can be looked at before a design has been started. This following sessions provide a short and concise description on how to install, operate, and run the device driver demonstration program.

3.1.1 Installation
The evaluation kit contains an EPROM with a Debug Monitor/ESCC Driver. Insert this EPROM in the evaluation board EPROM socket. The ESCC device driver can be used in Stand-Alone or PC Plug-in configuration. The ESCC Device Driver uses ASCI Channel 0 as a console in both Stand-Alone and PC Plug-in Configuration. Since the Device Driver runs at 18.432 MHz in divide-by-1 mode, load X1 with a 18.432 MHz crystal (using the Debug Monitor automatically enters divide-by-two mode for 9.216MHz operation). When the ESCC Device Driver is configured for PC Plug-in , you still need another terminal device connected to ASCI channel 0 for use as a ESCC Device Driver Console.

Evaluation Board jumpers must be set to use ASCI channel 0 as a console (refer to Chapter 1 - installation). The jumpers below must also be added in order to make use of DMA transfer features:

P6-20 to P6-18 and P6-15 to P6-17

3.1.2 To run the ESCC Device Driver
ESCC Device Driver can be activated in the Z80182 evaluation board Debug Monitor when :
(a) the program control is transferred to address 2000h OR
(b) J command is typed in Debug Monitor.

Example:

```
(a) DEBUG MONITOR >> J

(b) DEBUG MONITOR >> Go From Address: 2000
```

The Device Driver is activated and the following menu will be displayed:

```
-------------------------------------------------------------
|==================================================================|
                    Z80182 Datacom Evaluation Kit
             Z85230 ESCC Enhanced Serial Communication Controller
                  Device Driver and Demonstration Program
|==================================================================|

1.  Resetting the ESCC

2.  Initializing the ESCC

Default baud rate = 9600 bps
Default TxMethod = DMA
Default RxMethod = DMA
Mode: Rx Interrupt on Special condition only (default)

3.  Initializing Transmit and Receive Buffers
```

```
Possible buffer content:
   1: line
   2: pattern (/^\_)
   3. binary (from 00h up)
   4: zero
Enter buffer type:
-----------------------------------------------------------------
```

and the Z80182-ESCC Device Driver has been activated.

Initially, the program will reset and initialize the ESCC, preset the baud rate to 9600 bps, set Tx and Rx method to DMA, and default the Rx Interrupt for Special Condition Only.

There are four ways to initialize the Tx buffer: line, pattern, binary, or zero (see details below). Choose one of the options by typing the number in front of the option. Enter the desired buffer length and follow the instructions that are displayed.

Example:

```
Enter buffer type: 2 <CR>

Enter buffer length (no CRC) : 1000 <CR>
```

Pattern buffer is selected and buffer length of 1000 bytes are reserved.

The following main menu will be displayed next:

```
--------------------------------------------------------------------------
|========================================================================|
|        ||      ESCC  (Enhanced Serial Communication Controller)      || |
|        ==============================================================  |
|   GENERAL FUNCTION     |      TRANSMISSION          |    RECEPTION     |
|------------------------+----------------------------+------------------|
|                        | [1] Init Tx buffer         | [2] Init Rx buffer
|  [5] Start Transfers   | [3] TxMethod=DMA           | [4] RxMethod=DMA |
|  [R] Eval ESCC reg.    | [A] Show Tx buffer         | [B] Show Rx buffer
|                        | [U] IdleMode=Mark          | [Y] StatusFIFO=No|
|                        | [D] TxLevel=NotFull        | [E] RxLevel=NotEmpt
|                        | [M] RTS_Timing=Normal      | [N] RxIntMode=Sp.On
|  ESC Zilog Monitor     | [J] DTR_Timing=AsW_REQ     | [K] Purge SDLC FIFO
|------------------------+----------------------------+------------------|
|  [7] Expert Menu       | [6] Modify baud rate (Current baud rate=9600) |
|                        | [L] Loopback=Yes                              |
|                        | [X] ExtendRead=No                             |
|                        | [I] ExtStatIE=No                              |
|  >>>   H D L C   <<<   | [P] Purge All                                 |
|   p r o t o c o l      | [C] Compare Tx - Rx                           |
|========================================================================|

Enter choice:
--------------------------------------------------------------------------
```

Initialization of the Device Driver is therefore completed.

3.1.3 To exit the ESCC Device Driver

Typing <ESC> in the main menu will return the control to the debug monitor of the Z80182 evaluation board. The program will return to 0000h and essentially re-initialize the whole system.

3.1.4 Operation of the Device Driver

By typing in various commands in the main menu, the device driver can:
a. Initialize the Rx buffer with the [2]- Init Rx Buffer command.
b. Use the [5]- Start Transfers command to transfer the Tx buffer programmed in #1 to the Rx buffer.
c. Send X number of frames.
d. Use the [C]- Compare Tx-Rx command to verify if the transfers were performed correctly.
e. Use the [B]- Show Rx buffer command to view the Rx buffer.
You can now experiment with the other options available on the ESCC device driver to see what effects they have on the device. The following sections give a detailed description of the commands that are available to the user.

3.1.5 TRANSMITTER DEVICE DRIVER COMMANDS

This section contains descriptions of the various device driver commands that are available for the transmitter portion of the ESCC.

[1] - Init Tx buffer

This command allows the user to initialize the memory buffer that will be transmitted. The device driver gives the user four options to choose from to fill this buffer. They are:

a) line - This allows the entry of ASCII characters into the Tx buffer.The maximum Tx buffer size is currently 1K Bytes. This value may be changed in the source code of the "buffer.c" file. The program would then need to be recompiled.

b) pattern - Allows the transmission of a preset pattern.

c) binary - This command will fill the buffer with bytes starting at 00h and filling each consecutive byte of the buffer with an incremented value over the last one written. For example, a buffer would contain the following if the buffer length is programmed to 5 and the binary command was chosen:

```
00 01 02 03 04
```

d) zero - Initializes the Tx buffer to zero for the desired number of bytes.

[A] - Show Tx buffer

This command will display the current Tx buffer contents. This buffer may be changed by the "Init Tx buffer" command. The first four bytes, if they are used, are:

- frame address
- control field

[3] - TxMethod= ???

The transmitter may be programmed to transmit characters by one of three methods: POLLING, INTERRUPTS, or DMA. The transmitter may also be disabled with this command. The current setting will be displayed on the main menu.

[U] - IdleMode= ???

The transmitter may be toggled to send either FLAGS (01111110) or MARK (11111111) during an idle condition. This feature is only relevant in HDLC/SDLC mode. This command directly toggles bit WR10 D3. The current setting will be displayed on the main menu. Refer to the ESCC technical manual on page 5-14 for further details.

[D] - Txlevel= ???

The ESCC contains a four byte transmit FIFO rather than a one byte Tx buffer as in the SCC. This command triggers when the ESCC sets the transmit buffer empty (TBE) interrupt bit. The TBE interrupt can be generated when the transmit FIFO is completely empty ("Empty" value) or when the top byte of the Tx FIFO is empty ("Not Full" value). Similarly in DMA mode, the Tx level controls when the DMA request signal is asserted. For further details, see page 5-12 of the ESCC technical manual.

[M] - RTS_Timing= ???

This command controls the timing of the deassertion of the /RTS pin
(WR7' D2). For a detailed description of the operation of this feature, refer to page 5-12 of the ESCC technical manual. The current setting will be displayed on the main menu.

[J] - DTR_Timing= ???

This command also controls another enhanced feature of the ESCC. When this bit is set (WR7' D4), the deactivation of the /DTR//REQ pin is identical to the /W//REQ pin. If this feature is disabled, the /DTR//REQ pin will behave identically as it did on the SCC. For further details, see page
5-12 of the ESCC technical manual.

3.1.6 RECEIVER DEVICE DRIVER COMMANDS

This section contains descriptions of the various device driver commands that are available for the receiver portion of the ESCC.

[2] - Init Rx buffer

This commands clears the Rx buffer contents to zero. Similar to the Tx buffer, the Rx buffer is currently 1K Bytes. This value may be changed in the source code of the "buffer.c" file. This program would then need to be recompiled.

[B] - Show Rx buffer

This command will display the current Rx buffer contents. This buffer may be cleared by issuing the "Init Rx buffer" command. The first four bytes, if they are used, are:
- frame address
- control field

[4] - RxMethod= ???

The receiver may be programmed to receive characters by one of three methods: POLLING, INTERRUPTS, or DMA. The receiver may also be disabled with this command. The current setting will be displayed on the main menu.

[Y] - Status FIFO= ???

This command enables or disables the SDLC Frame Status FIFO (WR15 D2). The SDLC Frame Status FIFO is 10 deep by 19-bit wide that contains count and status/error information about each received SDLC frame. This feature is fully described on page 4-25 of the ESCC technical manual.

[E] - RxLevel= ???

The ESCC contains an eight byte receive FIFO rather than a three byte Rx buffer as in the SCC. This command triggers when the ESCC sets the receive character available (RCA) interrupt bit (WR7' D3). The RCA interrupt can be generated when the receive FIFO is half full (Half Full or EOF) or on every received character (Not Empty) as in the SCC. For further details, see page 5-12 of the ESCC technical manual.

[N] - RxIntMode= ???

This command controls the various conditions that may cause a Rx interrupt request (WR1 D3,D4). This is described on page 5-4 of the ESCC technical manual.

[K] - Purge SDLC FIFO

This command purges the SDLC Frame Status FIFO.

3.1.7 TRANSMISSION AND RECEPTION DEVICE DRIVER COMMANDS

This section describes some general purpose commands that affect both the receiver and the transmitter sections of the device driver.

[6] - Modify baud rate (Current baud rate = ???)

This command will change the baud rate of transmission and reception of the ESCC. The maximum value is 4.608 MBits/sec with 20 MHz Z80182 and 20 MHz ESCC running at 18.432 MHz. The current value will always be displayed.
Not all baud rates are directly achievable since the baud rate is a function of the clock used. See page 3-2 of the ESCC technical manual for the operation of the baud rate generators and equation that determines the baud rate.

[L] - Loopback= ???

Local loopback is selected when WR14 D4 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. See page 2-37 of the ESCC technical manual.

[X] - Extended Read= ???

Setting WR7' D6 enables the reading of WR3, WR4, WR5, WR7' and WR10. See page 5-12 of the ESCC technical manual for further details.

[T] - ExtStatIE= ???

Allows the user to enable all of the external interrupts (WR15 D1,D3,D4,D5,D6,D7) and enables the External/Status Master Interrupt bit (WR1 D0). See page 5-19 and 5-6 for further details.

[P] - Purge All

This command purges the SDLC Frame Status FIFO, the receive and transmit FIFOs and issues all general reset commands to the ESCC.

[C] - Compare Tx - Rx

This command usually follows the "Start Transfers" command to look at what has been received compared to what was transmitted. Local loopback must be enabled or an external loopback plug or jumper connected between the TxDA and RxDA pins.

3.1.8 GENERAL FUNCTIONS

[R] - Eval ESCC reg

This gives the user direct access to the ESCC registers. The contents of all internal registers are displayed and may be changed by the user. See Chapter 5 of the ESCC technical manual for a detailed description of each of the registers.
[5] - Start Transfers

This command will transfer the contents of the Tx buffer to the ESCC transmitter by the method that the user programmed in the TxMethod. If local loopback is enabled, the receiver of the ESCC will receive the data and place it in the Rx buffer. The "Compare Tx-Rx" command can then be used to verify that the frame or frames of data were sent correctly. The "Show Rx buffer" command will also show the two bytes of CRC that is calculated and sent with every transmitted frame.

[ESC] - Zilog monitor

This command gets the user out of the device driver and back to the debug monitor program.

[7] - Expert Menu

This command will display another sub-menu that contains lower level device drivers that are used by the main menu software drivers.

```
-----------------------------------------------------------------------

==================================================================
      ||                 ESCC EXPERT MENU                    ||
      =======================================================
  GENERAL FUNCTION    |      TRANSMISSION        |      RECEPTION
----------------------+--------------------------+-------------------
                      | [1] Init  Tx             | [2] Init Rx
 [D] Eval DMA reg.    | [3] Start Tx             |
                      | [4] Tx                   | [5] Rx
                      | [6] End Tx               | [7] End Rx
----------------------+--------------------------+-------------------
                      | [E] End Tx and Rx (common features)
                      | [R] Reset ESCC
                      | [I] Init ESCC  (default)
                      |
 ESC Main Menu        |
==================================================================

-----------------------------------------------------------------------
```

The following commands are used as defined by the [3] TxMethod command in the main menu:

[1] - Initiate the transmitter
[3] - Start transmission of the frame
[4] - Transmit data
[6] - Disable the transmitter

The following commands are used as defined by the [4] RxMethod command in the main menu:

[2] - Initiate the receiver
[5] - Receive data
[7] - Disable the receiver

[E] - Disable the transmission and reception (common features)

This command disables both the transmitter and receiver of the ESCC.

[R] - Reset the ESCC device
This command issues a software reset to the device.

[I] - Initialize the ESCC device with default value

is programmed to when the device driver is first run. For example, the device will be reset, programmed for DMA operation on the transmitter and receiver, and the baud rate set to 9600.

[D] - Direct access to the DMA registers (DMA Controller) of the Z80182

This command will give the user access to the DMA controller on the Z80182.

[ESC] - return to MAIN MENU

This will return the user back to the main menu.

3.1.9 To Modify the Source Code

You will need a C compiler for the Z80182 processor, since this is not provided in this kit. All source codes are compatible with Microtec's ANSI C compiler version for the 80182.

You can contact Microtec Research at the following address and mention Zilog's Z80182 Evaluation Kit.

Microtec Research Inc.
2350 Mission College Blvd.
Santa Clara, CA 95054
Tel. 408-980-1300
Toll free 800-950-5554

Alternately, if another vendor's C compiler is used, modifications to the source code may need to be performed to solve any incompatibility issues.

If you use the Microtec tools, install the compiler, assembler and linker on your hard disk. Include the tools in your path (refer to Microtec's documentation).

The following additional files help you drive the tools to recompile the source code:

- ESCC180CMD linker command file
  This file is a command file for the Microtec linker utility. Depending where you install the Microtec tools (which drive and directory), you may have to modify the ESCC180.CMD file. Just edit the file and modify the path in the "load" commands for the libraries:

```
load c:\mcc80\c80isf.lib -> load XXX\c80isf.lib
load c:\mcc80\c80isc.lib -> load XXX\c80isc.lib
load c:\mcc80\c80isz.lib -> load XXX\c80isz.lib
```

  where "XXX" is the complete path where you installed the Microtec libraries.

- ESCC180.BAT batch file
  These files contain all the necessary commands to compile and link the source code. If your path contains the Microtec directories, running ESCC180.BAT (which calls TOOBJ.BAT) will generate the final program ESCC180.HEX in Intel Hex format.

Modify the provided source code, compile its different modules, link them and locate the code in the available memory. You can either generate a file in Intel Hex format and download using Zilog's on-board monitor, or use another high performance debug monitor such as Microtec's XRAY package.

ZILOG

MIMIC DEVICE DRIVER DEMONSTRATION PROGRAM

4. MIMIC DEVICE DRIVER DEMONSTRATION SOFTWARE

4.1 Mimic Device Driver Demonstration Program User Guide

The MIMIC Device Driver Demonstration program supports Z80182's internal MIMIC (PC Standard UART) . The Device Driver gives the user the ability to work with and learn the various operations of the 16550 MIMIC. This Device Driver is a useful aid in development, in that developers may familiarize themselves with the MIMIC device and learn how the device must be programmed to fit in their applications. Registers can be programmed at the bit level, data can be sent and received, and features can be observed before implemented in final designs. The MIMIC device driver allows the developer to become comfortable with the MIMIC and its features even before any development begins. This User Guide provides a short and concise description on how to install, operate, and run the device driver demonstration program. Please take time to read Z80182 product specification before attempting to use Device Drivers.

4.1.1 Installation

The evaluation kit contains an EPROM with a Debug Monitor/MIMIC Driver. Insert this EPROM in the evaluation board EPROM socket. The MIMIC device driver can only be used in PC Plug-in configuration. The MIMIC Device Driver uses ASCI Channel 0 as a console and communicates with the PC through the PC XT/AT bus. Since the Debug Monitor/MIMIC Driver runs at 9.216 MHz in divide-by-2 mode, load X1 with a 18.432 MHz crystal. Refer to Chapter 1-Installation to set the evaluation board jumpers and connections for PC Plug-in Configuration.
Make sure the following ESCC/DMA Jumpers are Disconnected prior to using MIMIC Device Driver:

P6-20 to P6-18 and P6-15 to P6-17 **are NOT connected**

The MIMIC Device Driver consists of two sets of code or programs. One set of code which is stored on the included EPROM controls and monitors the Z80182 and MIMIC MPU registers. This program uses the ASCI channel 0 and an external terminal as a console. The other program, ASYNC.EXE, runs on the PC and controls and monitors the MIMIC PC registers. This executable file can be installed on a floppy, hard drive, or simply executed from the provided diskette. Run this executable file on the PC by typing ASYNC whenever you use the MIMIC Device Driver. The remaining sections of this user guide is split into three parts. One section describes operation on the external terminal, while the section following describes operations on the IBM-PC. It is suggested that these sections, along with the z182 technical manual, is read and understood prior to attempting to go on.

In order to use the MIMIC Device Driver effectively, both the PC and external terminal must be used simultaneously. The third section demonstrates the initiation of a simple data transfer using both the PC and the external terminal. This section is given in a step-by-step form that can easily be used as a stepping stone to becoming familiar with the Z80182's MIMIC. From here on, the manual will often refer to the Z182 program as the "MPU Side Program" and the PC program, ASYNC.EXE, as the "PC Side Program".

4.2 MIMIC Device Driver Operation - External Terminal (MPU Side)

4.2.1 Getting Started

Make sure that the external terminal is set for 9600 baud 7-n-2 or 8-n-1. If all jumpers and connections are set up properly (PC Plug-in Configuration), the message below should be shown on the external monitor. If garbage characters appear, set the external terminal baud rate to 9600 baud and toggle between 7-n-2 and 8-n-1. If no characters appear, refer to Chapter 1- Installation to make sure the evaluation board is properly set up for PC Plug-in Configuration.

```
                         Zilog
              Z80182 Evaluation Card Monitor
                    Version 2.0b
Debug Monitor  >>
```

The MIMIC Device Driver can be activated in one of the following methods:

(a) program control is transferred to address 2000h by pressing <G> and entering 2000 as starting address.

(b) J command is typed in Debug Monitor.

Example:

```
(a) DEBUG MONITOR >> J

(b) DEBUG MONITOR >> Go From Address: 2000
```

The MIMIC Device Driver is activated and the following will be displayed:

```
---------------------------------------------------------------------------------
|================================================================|
                      Z80182 Datacom Evaluation Kit
            MIMIC Device Driver and Demonstration Program
                              MPU SIDE
|================================================================|


Default TxMethod = INTERRUPTS
Default RxMethod = INTERRUPTS


Possible buffer content:
   1: line
   2: pattern (/^\_)
   3. binary (from 00h up)
   4: zero
Enter buffer type:
```

Initially, the program will reset MIMIC and default transmission method to interrupts.

There are four ways to initialize the Tx buffer: line, pattern, binary, or zero. These choices are described in a later section. For now, choose one of the options by typing the number in front of it. Enter the desired buffer length and follow the instructions that are displayed.

Example:

```
Enter buffer type:  2 <CR>

Enter buffer length (no CRC)  :  100 <CR>
```

The example shown selected a continuous pattern of characters (/^_/^_/^_ ... etc) of 100 byte length to be stored in a transmit buffer. Whenever a transfer is initiated, this pattern of characters will be transmitted to the PC bus.

The following main menu will be displayed next:

```
------------------------------------------------------------------------
|======================================================================|
|      ||                  *  MPU Z80182 SIDE  *                  ||   |
|      ||                  MIMIC DEVICE DRIVER                    ||   |
|      =============================================================    |
|  GENERAL FUNCTION     |      TRANSMISSION          |    RECEPTION    |
|-----------------------+----------------------------+-----------------|
| [M] Eval MIMIC reg.   | [1] Init RBR buffer        | [2] Init THR buffer
| [H] Eval PORT A/B/C   | [3] Show RBR buffer        | [4] Show THR buffer
| [5] Start Transfers   | [6] RBR_Method=INTERRUPTS  | [7] THR_Method=INTE
| [D] Eval DMA reg.     | [8] RBR_Delay=Yes          | [9] THR_Delay=Yes
| [T] Trace             | [A] RTCR = 8H              | [B] TTCR = 8H
| [ESC] Zilog Monitor   | [W] Write Character        | [R] Read Character
|-----------------------+----------------------------+-----------------|
|                       |   [G] WR12_CHB = ffH       [O] WR13_CHB = 0H
| ***** 16450 *****     |             [C] Compare RBR - THR
| ***** Mode  *****     |             [E} Enable Interrupt(s)
|                       |             [P] Purge All
|======================================================================|
Enter choice:

------------------------------------------------------------------------
```

Note: [7] should be - [7] THR_Method = INTERRUPTS. View was compressed to fit page.

4.2.2 Exiting the MIMIC Device Driver

Typing <ESC> in the main menu will return control to the debug monitor. The whole system will re-initialize and the program will return to 0000h (Debug Monitor start address).

4.2.3 Operation of the Device Driver

By typing in various commands in the main menu, the device driver can:

- Initialize the THR and RBR buffers
- Use the [5]- Start Transfers command to transfer the RBR buffer to the PC bus.
- Use the [C] - Compare RBR-THR buffers command to verify if the transfers were performed correctly.
- Use the [4] - Show THR buffer command to view the THR buffer.

You can now experiment with the other options available on the MIMIC Device Driver and observe what effects they have on MIMIC data transfers. The following sections give a detailed description of the commands that are available to the user.

4.2.4 TRANSMITTER DEVICE DRIVER COMMANDS

This section contains descriptions of the various MIMIC Device Driver commands that are available for the transmitter portion of the MIMIC Driver.

[1] - Init RBR buffer

This command allows the user to initialize the memory buffer that will be transmitted to the PC Bus. The device driver gives the user four options to choose from to fill this buffer. They are:

a) line - This allows the entry of ASCII characters into the Tx buffer.The maximum Tx buffer size is currently 1K Bytes. This value may be changed in the source code of the "buffer.c" file. The program would then need to be recompiled.

b) pattern - Allows the transmission of a preset pattern. The ASCII representation of this pattern is
"/^_/^_/^_ .... etc "

c) binary - This command will fill the buffer with bytes starting at 00h and filling each consecutive byte of the buffer with an incremented value over the last one written. For example, a buffer would contain the following if the buffer length is programmed to 5 and the binary command was chosen:

```
00 01 02 03 04
```

d) zero - Initializes the Tx buffer to zero for the desired number of bytes.

[3] - Show RBR buffer

This command will display the current RBR buffer contents. This buffer may be changed by the "Init RBR buffer" command.

[6] - RBR_Method= ???

The transmitter may be programmed to transmit characters to the PC bus by one of three methods: POLLING, Z182 INTERRUPTS, or Z182 DMA. The transmitter may also be disabled with this command. The current setting will be displayed on the main menu.

[8] - RBR_Delay = ?

The purpose of adding a RBR data transfer delay is to simulate the 16550 UART's delay caused by the shifting of data required in serial communication. This feature is added to eliminate any software problems a high speed continuous data transfer might cause to existing software that uses the 16550 UART. Adding an RBR delay allows the RBR transmission to occur at a slower rate.

[A] - RTCR = ??
Receive Timer Delay Counter Register - When RBR_Delay is enabled, the RTCR is loaded into a timer and delays setting the Data Ready bit in the LSR until the timer counts to zero. The clock rate in which this register counts down is determined by registers 12 and 13 of ESCC channel B. This allows for a 24-bit division of Z182s internal clock rate. Provisions for programming these registers will be discussed in a later section. If data transfers do not receive/transmit entire buffer length, try increasing RTCR value to account for software overhead.

[W] - Write Character
This writes one character to the PC bus. When choosing this action, "char to be written? " will enquire what character will be placed in MIMIC's RBR register. Press the character on the external terminal keyboard that you wish to send to the PC bus. By initiating a <R> read command on the PC HOST, the written character can be read.

4.2.5 RECEIVER DEVICE DRIVER COMMANDS

This section contains descriptions of the various device driver commands that are available for the receiver portion of the MIMIC Device Driver.

[2] - Init THR buffer

This command allows the user to clear contents of THR buffer. The THR receive buffer should be cleared prior to any data transfers to provide an accurate representation of what was received.

4] - Show THR buffer

This command will display the current THR buffer contents along with buffer length and physical address. If you initiate the Show THR buffer command after initializing the THR buffer, no data will be displayed.

[7] - THR_Method= ???

The receiver may be programmed to receive characters by one of three methods: POLLING, Z182 INTERRUPTS, or Z182 DMA. The receiver may also be disabled with this command. The current setting will be displayed on the main menu.

[8] - THR_Delay = ?

The purpose of adding a THR data transfer delay is to simulate the 16550 UART's delay caused by the shifting of data required in serial communication. This feature is added to eliminate any software problems a high speed continuous data transfer might cause to existing software that uses the 16550 UART. Adding an THR delay allows the THR transmission to occur at a slower rate.

[A] - TTCR = ??
Transmit Timer Delay Counter Register - When THR_Delay is enabled, the TTCR is loaded into a timer and delays setting the THRE (Transmitter Holding Register Empty) bit in the LSR until the timer counts to zero. The clock rate in which this register counts down is determined by registers 12 and 13 of ESCC channel B. This allows for a 24-bit division of Z182s internal clock rate. Provisions for programming these registers will be discussed in a later section. If data transfers do not receive/transmit entire buffer length, try increasing RTCR value to account for software overhead.

[R] - Read Character
This command reads one character from the PC bus. When choosing this action, "char read =" will be followed by the appropriate character that was written to the MIMIC THR register by the PC.

4.2.6 TRANSMISSION AND RECEPTION DEVICE DRIVER COMMANDS

This section describes some general purpose commands that affect both the receiver and the transmitter sections of the MIMIC device driver.

[Q] and [G] - WR13_CHB and WR12_CHB

Registers 13 and 12 of ESCC Channel B provide a 16-bit counter that is clocked by the Z182 internal clock. Register 13 provides the upper byte of the time constant while Register 12 provides the lower byte. Although this counter is conventionally used as a baud rate generator for the ESCC, it is also used in conjunction with MIMIC Timer Delays and Timeouts to provide a 24-bit time delay. To increase the THR and RBR delay, the hex values in these registers can be increased.

[C] - Compare RBR - THR

This command usually follows the "Start Transfers" command to look at what has been received compared to what was transmitted. This is useful only if Transmit buffers on the PC side are set up in the same manner as the MPU side. A simple MIMIC test would send a pattern to the PC. That same pattern is also sent from the PC to the Z182 MPU. By comparing what was received to what was sent you can determine whether the MPU is receiving data from the PC correctly.

[E] - Enable Interrupt(s)

This command allows the user to enable the following interrupts:

a. FCR - This interrupt occurs when PC writes to FIFO Control Register
b. DIV - This interrupt occurs when PC writes to Divisor Latch
c. MCR - This interrupt occurs when PC writes to Modem Control Register
d. LCR - This interrupt occurs when PC writes to Modem Control Register

THR full, RBR empty, and Transmit Timeout interrupts are automatically set according to transfer methods selected and Timeout enable.

[P] - Purge All

This command is used to reset all interrupts, reset Trace memory, and clear THR register/fifo.

4.2.7 GENERAL FUNCTIONS

[R] - Eval MIMIC reg

This gives the user direct access to the MIMIC registers. The contents of all internal registers are displayed and may be changed by the user. Refer to pgs. 57-68 for more information about MIMIC registers. The registers are displayed in two sections: The top register display are read only registers that cannot be modified by MPU side device driver (These registers must be modified by the PC using the ASYNC.EXE program). The lower section displays registers that can be modified (or partially modified). Pressing <ESC> will return control to the MIMIC Driver Main Menu.

[H] - Eval PORT A/B/C

The Z182 also has 3 internal Parallel ports labelled A,B, and C. The data direction (input or output) and data values can be read or written with this command. Refer to pgs. 69-71 of Z182 product specification for more information.

[5] - Start Transfers

This command initiates transfers from the Z182 MPU to the PC. The contents of RBR buffer are transferred to the PC via the MIMIC and PC bus. This command simultaneously allows data to be transferred from the PC to the Z182 MPU. **The Start Transfer command must be initiated in both the MPU side and the PC side for a transfer to occur.**

[D] - Eval DMA reg.

This command allows the user to have direct access to DMA registers. The contents of these registers are displayed and can easily be modified by choosing register of interest and writing hexadecimal value to that register. Refer to pgs. 35-38 of Z182 product specification for more information.

[T] - Trace

This command is usually given after a data transfer. An option to display all Trace or Display value change is given. When Display All Trace is chosen, each service routine has a table entry. When Display Value Change is chosen, only those service routines that transferred a different number of bytes than previous service routine (first service routine is always shown in table) are represented in the table.

Initiating the Trace will display a table of events on the monitor screen. This table will be given in the form of;

T=INT_Nbr/TMDR0H/TMDR0L   V=?

T= 6 digit hexadecimal number representing the time in which the particular data transfer was completed. When data transfer starts, a 2-byte internal timer is initiated. TMDR0H and TMDR0L are the timer registers. When these registers reach a value of ffffH, INT_Nbr increments by 1. Therefore, the T value is analogous to a time stamp for that service routine.

V= The number of bytes of data that was transferred in each service routine for INTERRUPT mode and number of all bytes of data that was trasfer for DMA mode

Two tables are shown; one for MPU data reception and another for data transmission. The first table represents data reception service routines. Press any key after this to view table for data transmission service routines.

[ESC] - Zilog monitor

This command gets the user out of the device driver and back to the debug monitor program.

4.2.8 FIFO FUNCTIONS

In order to access these commands, FIFO mode must be selected on the PC side by choosing the <F> command in the MIMIC Device Driver Menu in the ASYNC.EXE program. Once this is selected, press <CR> on the MPU side terminal to re-display the MIMIC Driver menu. The menu should now have a section for FIFO FUNCTIONS as shown below:

```
-----------------------------------------------------------------------------
|=============================================================================|
|       ||                  *  MPU Z80182 Side  *                       ||    |
|       ||                  MIMIC DEVICE DRIVER                         ||    |
|       ==================================================================    |
|  GENERAL FUNCTION    |     TRANSMISSION          |      RECEPTION          |
|----------------------+---------------------------+-------------------------|
| [M] Eval MIMIC reg.  | [1] Init RBR buffer       | [2] Init THR buffer     |
| [H] Eval PORT A/B/C  | [3] Show RBR buffer       | [4] Show THR buffer     |
| [5] Start Transfers  | [6] RBR_Method=INTERRUPTS | [7] THR_Method=INTE     |
| [D] Eval DMA reg.    | [8] RBR_Delay=Yes         | [9] THR_Delay=Yes       |
| [T] Trace            | [A] RTCR = 8H             | [B] TTCR = 8H           |
| [ESC] Zilog Monitor  | [W] Write Character       | [R] Read Character      |
-----------------------------------------------------------------------------
| >>>>>>>>>><<<<<<<<<< | [I] RBR_Timeout=No        | [J] THR_Timeout=No      |
| >> FIFO FUNCTIONS << | [X] RTTC = ffH            | [Y] TTTC = ffH          |
| >>>>>>>>>><<<<<<<<<< | RCVR_Trigger_Level=01     | [Z] XMIT_Trigger_Le     |
|----------------------+---------------------------+-------------------------|
|                      |    [G] WR12_CHB = ffH      [O] WR13_CHB = 0H         |
| *****  16450  *****  |              [C] Compare RBR - THR                   |
| *****  Mode   *****  |              [E} Enable Interrupt(s)                 |
|                      |              [P] Purge All                           |
|=============================================================================|

Enter choice:
-----------------------------------------------------------------------------
```

Note: [7] should be - [7] THR_Method = INTERRUPTS and [Z] should be - [Z] XMIT_Trigger_Level=01.
View was compressed to fit page.

[I] - RBR_Timeout=?

RBR_Timeout feature allows the Z80182 Receive Timeout Timer to emulate the four character timeout delay specified by the 16550 UART. If no read or write to the RCVR FIFO has taken place and data bytes are still available, but are below the PC trigger level, a timeout interrupt is sent to PC so that these bytes can be read.

[X] RTTC = ?

This register contains an 8-bit constant for emulation of the 16550 four character timeout feature. This timer is enabled to down count when the RBR_Timeout is enabled. When the down count reaches 00H, the PC timeout interrupt occurs to enable reading of remaining bytes in the RBR FIFO. This timer receives its input from the baud rate generator of ESCC Channel B (discussed previously) that can be modified with commands [Q] and [G].

Below command [X] is a field that is displayed as follows:
`RCVR_Trigger_Level=01`

This value cannot be modified by the MPU side but can be changed on the PC HOST side device driver. This value indicates the number of available bytes in the receiver FIFO before an interrupt to the PC occurs to transfer these bytes.

[J] - THR_Timeout=?

This command enables the Z80182 timer that is used to interrupt the MPU if characters are available in the Transmit FIFO but are below the trigger level. The timer will timeout and interrupt the MPU if no read or write to the Transmit FIFO takes place within the interval specified by register TTTC.

[Y] - TTTC = ?

This register contains an 8-bit constant for determining the interval for the Transmit Timeout Timer. When allowed to timeout, this timer interrupts the MPU by setting the THR bit in the IUS/IP register. This will allow bytes below trigger level in Transmit FIFO to be transmitted to MPU. This constant is reloaded into timer every time there is a read or write to the Transmit FIFO. The timer receives its input from the baud rate generator of ESCC Channel B (discussed previously) that can be modified with commands [Q] and [G].

[Z] - XMIT_Trigger_Level=01

This command allows you to choose the number of bytes available to read in the Transmit FIFO before an interrupt occurs to the MPU. The choices are given as 1,4,8, and 14 bytes.

In the next section, the PC HOST SIDE operation will be discussed.

4.3 MIMIC Device Driver Operation - ASYNC.EXE (PC HOST SIDE)

ASYNC.EXE runs on the PC and controls and monitors the MIMIC PC registers. This program is included in the evaluation kit diskettes. Upon startup, the program will ask what com port is the evaluation board using. Header J1 can be set to respond to com ports 1-4 or other designated addresses. Header J2 can be set to use interrupts 3, 4, 5, 10, or 11. The program is designed with interrupts paired with certain com ports. A table of com port - interrupt pairs are shown below:

Com 1	---	IRQ4	J1 (15-16)	&	J2 (3-4)
Com 2	---	IRQ3	J1 (7-8)	&	J2 (5-6)
Com 3	---	IRQ4	J1 (13-14)	&	J2 (3-4)
Com 4	---	IRQ3	J1 (5-6)	&	J2 (5-6)

Make sure the jumper on J2 is connected such that its interrupt level corresponds to the com port selected on J1.

4.3.1 Getting Started - ASYNC.EXE

Make sure jumpers on J1 and J2 are set properly and evaluation board is set up for PC Plug-in configuration described in Chapter 1 - Installation. I

At the DOS prompt type the following: ASYNC <CR>

If the software was installed properly, the following message is displayed on the PC monitor:

```
------------------------------------------------------------------
|================================================================|
                    Z80182 Datacom Evaluation Kit
          MIMIC Device Driver and Demonstration Program
                             HOST SIDE
|================================================================|


Default TxMethod = INTERRUPTS
Default RxMethod = INTERRUPTS


[1] - COM1
[2] - COM2
[3] - COM3
[4] - COM4
---------------------------
Select COM[1,2,3,4]:  3

------------------------------------------------------------------
```

Com 3 is automatically selected as default value. If evaluation board is configured to respond to Com 3 press Enter. Otherwise, press the key corresponding to the com port selection.
Once this is done, the following message should appear on the PC monitor:

```
------------------------------------------------------------------------------------
Possible buffer content:
   1: line
   2: pattern (/^\_)
   3. binary (from 00h up)
   4: zero
Enter buffer type:
------------------------------------------------------------------------------------
```

Initially, the program will reset MIMIC and default transmission method to interrupts.

There are four ways to initialize the Tx buffer: line, pattern, binary, or zero. These choices are described in a later section. For now, choose one of the options by typing the number in front of it. Enter the desired buffer length and follow the instructions that are displayed.

Example:

```
Enter buffer type:  2 <CR>

Enter buffer length (no CRC)  :  100 <CR>
```

The example shown selected a continuous pattern of characters (/^_/^_/^_ ... etc) of 100 byte length to be stored in a transmit buffer. Whenever a transfer is initiated, this pattern of characters will be transmitted to the Z182 MIMIC.

The following main menu will be displayed next:

```
-----------------------------------------------------------------------
|======================================================================|
|      ||              *  HOST SIDE     *                      || |    |
|      ||              MIMIC DEVICE DRIVER                     || |    |
|      =================================================================
|  GENERAL FUNCTION    |     TRANSMISSION         |     RECEPTION      |
|----------------------+--------------------------+--------------------|
| [M] Eval MIMIC reg.  | [1] Init THR buffer      | [2] Init RBR buffer|
|                      | [3] Show THR buffer      | [4] Show RBR buffer|
| [5] Start Transfers  | [6] THR_Method=INTERRUPTS| [7] RBR_Method=INTE|
| [ESC] Exit           | [W] Write Character      | [R] Read Character |
|----------------------+--------------------------+--------------------|
|                      |           [A] IE_MSR_Enable=No                |
|                      |           [B] IE_LSR_Enable=No                |
|                      |           [F] FIFO_Mode=No                    |
|                      |           [C] Compare RBR - THR               |
|                      |           [P] Purge All                       |
|======================================================================|

Enter choice:

-----------------------------------------------------------------------
```

Note: [7] should be - [7] RBR_Method = INTERRUPTS. View was compressed to fit page.

4.3.2 Exiting the MIMIC Device Driver

Typing <ESC> in the main menu will return control to the DOS Prompt.

4.3.3 Operation of the Device Driver

By typing in various commands in the main menu, the device driver can:

a. Initialize the THR and RBR buffers
b. Use the [5]- Start Transfers command to transfer the THR buffer to the Z182 MIMIC.
c. Use the [C] - Compare RBR-THR buffers command to verify if the transfers were performed correctly.
d. Use the [4] - Show RBR buffer command to view what was received from the Z182.

You can now experiment with the other options available on the ASYNC program and observe what effects they have on MIMIC data transfers. The following sections give a detailed description of the commands that are available in the ASYNC program.

4.3.4 TRANSMITTER DEVICE DRIVER COMMANDS

This section contains descriptions of the various commands that are available for the transmitter portion of the ASYNC program.

[1] - Init THR buffer

This command allows the user to initialize the memory buffer that will be transmitted to the Z182 MIMIC. The device driver gives the user four options to choose from to fill this buffer. They are:

a) line - This allows the entry of ASCII characters into the Tx buffer.The maximum Tx buffer size is currently 1K Bytes. This value may be changed in the source code of the "buffer.c" file. The program would then need to be recompiled.

b) pattern - Allows the transmission of a preset pattern. The ASCII representation of this pattern is "/^_/^_/^_ .... etc "

c) binary - This command will fill the buffer with bytes starting at 00h and filling each consecutive byte of the buffer with an incremented value over the last one written. For example, a buffer would contain the following if the buffer length is programmed to 5 and the binary command was chosen: 00 01 02 03 04

d) zero - Initializes the Tx buffer to zero for the desired number of bytes.

[3] - Show THR buffer

This command will display the current THR buffer contents. This buffer may be changed by the "Init THR buffer" command.

[6] - THR_Method= ???

The transmitter may be programmed to transmit characters to the Z182 MIMIC by one of three methods: POLLING, PC - INTERRUPTS, or PC - DMA. The transmitter may also be disabled with this command. The current setting will be displayed on the main menu.

[W] - Write Character

This writes one character to the Z182 MIMIC. When choosing this action, "char to be written? " will enquire what character will be transferred to MIMIC's THR register. Press the character on the external terminal keyboard that you wish to send to the Z182 MIMIC. By initiating the <R> - read command on the MPU side, you can read the character that you just wrote from the PC HOST.

4.3.5 RECEIVER DEVICE DRIVER COMMANDS

This section contains descriptions of the various device driver commands that are available for the receiver portion of the MIMIC Device Driver.

[2] - Init RBR buffer

This command allows the user to clear contents of RBR buffer. The RBR receive buffer should be cleared prior to any data transfers to provide an accurate representation of what was received.

[4] - Show RBR buffer

This command will display the current RBR buffer contents along with buffer length and physical address.

[7] - RBR_Method= ???

The receiver may be programmed to receive characters by one of three methods: POLLING, PC -INTERRUPTS, or PC - DMA. The receiver may also be disabled with this command. The current setting will be displayed on the main menu.

[R] - Read Character
This command reads one character from the Z182 MIMIC. When choosing this action, "char read =" will be followed by the appropriate character that was written to the MIMIC RBR register by the MPU Side.

4.3.6 TRANSMISSION AND RECEPTION DEVICE DRIVER COMMANDS

This section describes some general purpose commands that affect both the receiver and the transmitter sections of the ASYNC program.

[A] - IE_MSR_Enable=?

This command allows a PC Interrupt to occur when bits 0, 1, 2, or 3 of the Modem Status Register is set.

[B] - IE_LSR_Enable=?

This command allows a PC Interrupt to occur when bits 1, 2, 3, or 4 of the Line Status Register is set.

[F] - FIFO_Mode=?

This command enables the 16550 MIMIC FIFO mode. Initiating this command opens a new set of commands in the Main Menu labeled "FIFO FUNCTIONS". These functions can only be used when FIFO mode is enabled.

[C] - Compare RBR - THR

This command usually follows the "Start Transfers" command to look at what has been received compared to what was transmitted. This is useful only if Transmit buffers on the PC side are set up in the same manner as the MPU side. A simple MIMIC test would send a pattern to the Z182 MIMIC device. That same pattern is also sent from the Z182 MIMIC device to the PC bus. By comparing what was received to what was sent you can determine whether the Z182 MIMIC is transmitting data correctly.

[P] - Purge All

This command is used to reset all interrupts and clear RBR register/fifo.

4.3.7 GENERAL FUNCTIONS

[M] - Eval MIMIC reg

This gives the user direct access to the PC-MIMIC registers. The contents of all internal registers that are accessible to the PC are displayed and may be changed by the user. Refer to pgs. 57-68 for more information about MIMIC registers. The registers are displayed in two sections: The top register display are read only registers that cannot be modified by PC HOST side (These registers must be modified by the MPU side terminal). The lower section displays registers that can be modified. Note that 'XX' symbolizes registers that are not accessible unless register #3 (Z182_PC_LCR) is modified such that bit 7 (most significant bit) is equal to 1. Pressing <ESC> will return control to the MIMIC Driver Main Menu.

[5] - Start Transfers

This command initiates transfers from the PC to the Z182 MIMIC. The contents of THR buffer are transferred to the Z182 via the MIMIC and PC bus. This command simultaneously allows data to be transferred from the Z182 MPU to the PC. **The Start Transfer command must be initiated in both the MPU side and the PC side for a transfer to occur.**

[ESC] - Exit

This command gets the user out of the ASYNC program and back to the DOS prompt. Program exit can be initiated anytime by simple pressing Cntrl-C.

4.3.8 FIFO FUNCTIONS

In order to access these commands, FIFO mode must be selected on the PC side by choosing the <F> command in the MIMIC Device Driver Menu of the ASYNC.EXE program. The menu should now have a section for FIFO FUNCTIONS as shown:

```
---------------------------------------------------------------------------
|==========================================================================|
|        ||                    *  HOST SIDE     *                 ||| |
|        ||                    MIMIC DEVICE DRIVER                ||| |
|        ===============================================================   |
|  GENERAL FUNCTION     |      TRANSMISSION           |     RECEPTION      |
|-----------------------+-----------------------------+--------------------|
|  [M] Eval MIMIC reg.  | [1] Init THR buffer         | [2] Init RBR buffer|
|                       | [3] Show THR buffer         | [4] Show RBR buffer|
|  [5] Start Transfers  | [6] THR_Method=INTERRUPTS   | [7] RBR_Method=INTE|
|  [ESC] Exit           | [W] Write Character         | [R] Read Character |
|-----------------------+-----------------------------+--------------------|
|  >>>>>>>>>><<<<<<<<<< | [X] Reset THR FIFO          | [Y] Reset RBR FIFO |
|  >> FIFO FUNCTIONS << |                             | [V] RCVR_Trigger_Le|
|  >>>>>>>>>><<<<<<<<<< |            [I] PC_DMA_Mode=Single               |
|-----------------------+-----------------------------+--------------------|
|                       |            [A] IE_MSR_Enable=No                 |
|                       |            [B] IE_LSR_Enable=No                 |
|                       |            [F] FIFO_Mode=No                     |
|                       |            [C] Compare RBR - THR                |
|                       |            [P] Purge All                        |
|==========================================================================|

Enter choice:
---------------------------------------------------------------------------
```

Note: [7] should be - [7] RBR_Method = INTERRUPTS. Also, [V] should be - [V] RCVR_Trigger_level=01. View was compressed to fit page.

[X] - Reset THR FIFO

This command will cause the transmitter FIFO pointer logic to be reset; any data in the THR FIFO will be lost.

[Y] - Reset RBR FIFO

This command will cause the receiver FIFO pointer logic to be reset; any data in the RBR FIFO will be lost.

[V] - RCVR_Trigger_Level=?

This command allows selection of number of available bytes in the receiver FIFO before an interrupt to the PC occurs. The interrupt trigger levels available are given as 1, 4, 8, and 14 bytes.

[I] - PC_DMA_Mode=?

This function allows selection of Single byte DMA transfer or Multiple byte DMA transfer. This command will toggle back and forth between Single and Multiple modes.

MIMIC DATA TRANSFER

STEP BY STEP TUTORIAL

5. MIMIC DATA TRANSFER - STEP BY STEP TUTORIAL

5.1 Introduction

This section is provided to introduce MIMIC Device Driver usage. A step-by-step basic data transfer is explained in an easy to follow tutorial. After studying the Z182 product specification and sections 3 and 4, it is suggested this tutorial is utilized to become better acquainted with the MIMIC Device Driver Software. After completing this basic tutorial, more advanced methods of PC data transfer can be explored simply by modifying certain registers.

5.1.1 Initiating a Basic Data Transfer

Follow installation procedures described in Chapter 1 - Installation, and install evaluation board in PC Plug-in Configuration. Load EPROM socket with Debug Monitor/MIMIC Driver EPROM. Plug evaluation board into PC bus and install bracket faceplate with RS-232 connector. In order to use the MIMIC Device driver, an external terminal must be attached to this RS-232 plug.
Set Terminal to respond at 9600 baud 7-n-2 or 8-n-1.

1. Turn on PC. Once PC is turned on, the external terminal should respond with the following:

```
                              Zilog
                   Z80182 Evaluation Card Monitor
                            Version 2.0b
Debug Monitor >>
```

2. Press <J> on the external terminal. This should cause the MPU SIDE startup screen to appear.

The external terminal gives you the following choices:

```
Possible buffer content:
   1: line
   2: pattern (/^\_)
   3. binary (from 00h up)
   4: zero
Enter buffer type:
```

3. Press <2> on the external terminal.

You should see "Enter buffer length (no CRC):"

4. Press <1><0><0><enter> on the external terminal.

You have just set up the RBR buffer with a pattern of bytes, 100 bytes in length. Later you will transmit the contents of this buffer to the PC.

5. Start the PC HOST program ASYNC.EXE by typing "ASYNC" at the DOS prompt. Given that the file ASYNC.EXE is in the computer's current directory, the PC screen should display the HOST SIDE startup screen and prompt you (as shown below) to select the evaluation board's COM PORT:

```
[1] - COM1
[2] - COM2
[3] - COM3
[4] - COM4
----------------------------
Select COM[1,2,3,4]:  3
```

6. Type in the COM PORT number in which the evaluation board is set. The PC screen will then give options as to filling the THR buffer. This is shown below:

```
Possible buffer content:
   1: line
   2: pattern (/^\_)
   3. binary (from 00h up)
   4: zero
Enter buffer type:
```

7. Press <2> on the PC Keyboard.

You should see "Enter buffer length (no CRC):"

8. Press <1><0><0><enter> on the PC Keyboard.

You have just set up the PC's THR buffer with a pattern of bytes, 100 bytes in length. Note that the same pattern and length was previously set up for the Z182's RBR buffer. Later you will transmit the contents of the THR buffer to the Z182 and the PC will receive the contents of the RBR buffer. Since these buffer contents were set up exactly the same way, doing a compare on what was received versus what was transmitted should reveal no dissimilarities.

At this point, the MIMIC Driver Menu should be on both the external terminal and the PC screen. On the very bottom ot the terminal monitor and PC screen you should see "Enter Choice: " .

9. Press <5> on the external terminal. You have just started a data transfer. You should see the "Wait for the RTS signal from the PC side" message on the bottom of the external terminal. The data transfer will not begin until the Start Transfer command is initiated on the PC HOST.

10. Press <5> on the PC Keyboard. Congratulations! In 10 easy steps, you have successfully initiated a MIMIC-PC Data Transfer. The contents of the Z182 RBR buffer was transferred to the PC RBR buffer. Also, the contents of the PC THR buffer was transferred to the Z182 THR buffer.

On the very bottom of the external monitor and PC screen, you should see the following:

```
Current number / Total number for:
Tx:  100/100;  Rx:  100/800
```

This shows that 100 bytes out of 100 bytes in transmit buffer were transmitted and 100 bytes out of 800 bytes (maximum size of receive buffer) were received.

11. Press <N> on the PC Keyboard. You should see the following:

```
Modem_Int = 1
Error_Int = 0
Unexpected_Int = 0
TimeOut = 0

Hit key to go on
```

This shows that 1 the MODEM CONTROL INTERRUPT occurred once. When the PC set the RTS bit in the Modem Control Register, this interrupt occured. A running total of the other interrupts revealed that no other LSR, Timeout, or Unexpected Interrupts occurred in data transfer.

12. Press any key on the PC Keyboard. This should bring you back to the HOST SIDE menu.

13. Press <N> on the external terminal. This should bring the external terminal back to the MPU SIDE menu.

14. Press C on the PC Keyboard. You should see "Data was transferred O.K." on your PC screen. This shows that the MIMIC data transfer to PC was successful. You can type command <4> - Show RBR buffer, to observe that the pattern was actually received by the PC.

15. Press C on the external terminal. You should see "Data was transferred O.K." on your terminal. This shows that the PC data transfer to MIMIC was successful. You can type command <4> -Show THR buffer, to observe that the pattern was actually received by the Z182 MIMIC.

You can now experiment with advanced features such as FIFO and DMA transfers. This can be accomplished by simply choosing certain commands and modifying registers. The world of Z182 MIMIC now awaits your exploration and conquest!

APPENDICES

A. ESCC DEVICE DRIVER SUPPORT ROUTINES

This section describes the routines that are used by the device driver demonstration software.

A.1 C MODULES

The device driver demonstration software is made up of 8 different C modules. Details of the modules are described as follows:

A.1.1 CPU180.C functions:

The CPU180 module defines and controls miscellaneous DMA and interrupt operations. This is accomplished by programming the Z80182 differently:

DMA180_Init	Initialize the source and destination address in DMA transfers and the DMA transfer modes
DMA180_StartXfer	Enable DMA operation
DMA180_StopXfer	Disable DMA Channel and DMA operation
DMA180_GetCount	Read DMA Byte Count
DMA180_ShowReg	Display various DMA registers value
DMA180_ChgReg	Change various DMA registers value
DMA180_Done	Determine whether all characters were transferred by the DMA channel
ESCC_SetIntHdlr	Install interrupt handlers

A.1.2 UTILS.C functions:

The UTILS module controls the I/O to console terminal.

getchf	Get a character from a list of valid characters
HitKeyToGoOn	Display prompt message and get any character from the console to go on
WriteMenu	Output Menu to the console

A.1.3 SCC.C functions:

The SCC module sets the attributes and returns the status of the ESCC.

ESCC_Reset	Issue channel reset to the ESCC
ESCC_Init	Initialize various ESCC attributes
SCC_InitSDLC_Paramset	Initialize various SDLC parameters
SCC_InitSDLC_BaudRate	Initialize SDLC baud rate
SCC_BRG_Enable	Enable Baud Rate Generator
SCC_TxEnable	Enable Transmitter
SCC_RxEnable	Enable Receiver
SCC_RxDisable	Disable Receiver
ESCC_Enables	Enable Baud Rate Generator, Receiver, Transmitter and issue Reset commands
SCC_TxBufferNotFull	Determine whether Transmit FIFO is full
SCC_WriteChar	Write a character to the ESCC
SCC_CharAvailable	Determine whether Receive FIFO is half-full
SCC_ReadChar	Read a character from the ESCC
SCC_EnTxInt	Enable transmit interrupt
SCC_DisTxIntPend	Reset the pending transmit interrupts
SCC_DisTxInt	Disable transmit interrupt
SCC_EnRxInt	Enable receive interrupt

SCC_EnRxIntSpecCond	Enable 'Receive interrupt on Special Conditions Only'
SCC_EnRxInt1stAndSpecial	Enable 'Receive interrupt on 1st character or Special Conditions only'
SCC_EnRxIntAllSpecial	Enable 'Receive interrupt on all characters or Special Conditions'
SCC_DisRxInt	Disable receive interrupt
SCC_ResetHighestIUS	Issue 'Reset Highest IUS' command
SCC_SetMIE	Enable Master Interrupt
SCC_ResetMIE	Disable Master Interrupt
SCC_StartTxDMA_Xfer	Enable ESCC transmit DMA request
SCC_StopTxDMA_Xfers	Disable ESCC transmit DMA request
SCC_StartRxDMA_Xfer	Enable ESCC receive DMA request
SCC_StopRxDMA_Xfers	Disable ESCC receive DMA request
SCC_StartTxSDLC_Frame	Reset Transmit CRC Generator, send out the 1st byte and reset transmit underrun/EOM latch
RTS_Pin_Low	Assert /RTS pin
RTS_Pin_High	Deactivate /RTS
SCC_ResetStatus	Issue 'Error Reset' command
SCC_SetLoopBack	Enable local loopback
SCC_ResetLoopBack	Disable local loopback
ESCC_ToggleEnhdFeature	Toggle ESCC SDLC enhancement features
ESCC_SetAutoFlag	Enable Automatic Opening Flag transmission
SCC_ToggleExStatIE	Toggle external status interrupts
ESCC_ToggleIdleMode	Toggle Mark/Flag Idle
ESCC_ToggleSDLC_FIFO	Toggle SDLC Status FIFO
DisSDLC_FIFO	Disable SDLC FIFO
SCC_ShowRRs	Display SCC Read Registers
SCC_ShowWRs	Display SCC Write Registers
SCC_ChgReg	Change ESCC register value
ShowRR0	Show RR0 register value and meanings
ShowRR1	Show RR1 register value and meanings
ShowRegisters	Show all the register values
ShowRR15	Show RR15 register value and meanings
ESCC_ToggleAbort	Toggle Abort Interrupt Enable bit
ESCC_ToggleTxUnderrunEOM	Toggle Transmit Underrun/EOM bit
ESCC_ToggleCTS	Toggle CTS Interrupt Enable bit
ESCC_ToggleSYNC_Hunt	Toggle SYNC/HUNT Interrupt Enable bit
ESCC_ToggleDCD	Toggle DCD Interrupt Enable bit
ESCC_ToggleZeroCount	Toggle Zero Count Interrupt Enable bit

A.1.4 BUFFER.C functions:
The BUFFER module manages the user buffers and return the user buffers status.

SetToValue	Set the user buffer with the specified size to the chosen value
InitBuffers	Initialize the user buffer with the specified size with '0'
AllocateBuffer	Allocate total user buffer size
SetToPattern	Preset the whole user buffer with a predetermined pattern
SetToSuite	Reserve the whole user buffer for storing characters
SetToZero	Reset the whole user buffer with '0'
SetBuffer	Prompt user for user buffer allocation scheme
BlankBuffer	Preset the user buffer with '0' before processing
DisplayBuffer	Display a user buffer in both hex and ascii
ShowBuffer	Display status of the user buffers
CompareBuffers	Compare two user buffers, return first index where they differ

A.1.5 TOHOST.C functions:
The TOHOST module control communications of the Z80182 evaluation board with the host terminals. In Z8018200ZCO, the host terminal is communicated via RS232.

GetByte	Get a byte from the host terminal
GetWord	Get a word from the host terminal
SendByte	Send a byte to the host terminal
SendWord	Send a word to the host terminal
GetChar	Get a character from the host terminal
GetUnsigned	Get an unsigned number from the host terminal
GetLine	Get a line from the host terminal

A.1.6 LLC.C functions:
LLC module is the heart of the device driver. The module controls various operations of ESCCs and returns the results to higher level software.

ShowStatus	Show the status of the transfer operations
ControlMenu	Display the ESCC Main menu
InitTxBuffer	Initialize the transmit buffer starting address and the transmit block size of the user buffer
InitRxBuffer	Initialize the receive buffer starting address and the receive Purge the receive data FIFO of the ESCC
CompareTxRx	Compare the transmited and received frames attributes
PurgeAll	Purge the transmited and received frames and clear all the error conditions and external status interrupts
ReadSDLC_FIFO	Determine whether frame information is loaded into the SDLC Status FIFO. If loaded, display the frame information
PurgeSDLC_FIFO	Purge all the SDLC frame information and reset the SDLC status FIFO
TxPOLL_Init	Initialize transmitter in polling mode
TxPOLL_Start	Start frame transmission in polling mode
TxPOLL	Repeat sending out data in polling mode until transmit buffer is full
RxPOLL_DMA_SpecialCondHandlerr	Special conditions receive interrupt handler in polling mode, no SDLC FIFO is enabled, data FIFO is locked when special condition is detected
RxPOLL	Repeat receiving data in polling mode until receive buffer empty
RxPOLL_Special empty	Repeat polling receive status until receive buffer
RxPOLL_Init	Initialize receive polling modes
Unexpectedd	Display RR0, RR2 and RR3 status when unexpected interrupt is received
ESCC_TxINT_BuffEmptyHandlerr	Transmit interrupt handler. Interrupt when transmit FIFO is completely empty (WR7' D5 = 1). Keep writing to the transmit FIFO until the FIFO is full. Disable the pending transmit interrupt when all the data are sent
TxINT_BuffNotFullHandlerr	Transmit interrupt handler. Interrupt when entrance of the transmit FIFO is empty. Write one byte to the transmit FIFO. Disable the pending transmit interrupt when all the data are sent

AdjustBurstLimit	Adjust transmit burst limit
TxINT_Init	Initialize transmitter in interrupt mode
TxINT_Start	Initialize and trigger frame transmission
TxINT	Determine whether all character were transferred in interrupt mode
RxINT_CharAvailHandlerr	Handler for 'Receive Interrupts on All Characters or Special Conditions'. Use when transmit FIFO interrupt level is not enabled
ESCC_RxINT_CharAvailHandlerr	Handler for 'Receive Interrupts on All Characters or Special Conditions'. Use when transmit FIFO interrupt level is enabled. i.e. at least 4 bytes are in the data FIFO
RxDMA_CharAvailHandlerr	Handler for 'Receive Interrupts on 1st Characters or Special Conditions'. Use with DMA-driven systems. The 1st character is read by the DMA, not by the interrupt handler
RxDMA_FIFO_SpecialCondHandlerr	Handler for 'Receive Interrupts on Special Conditions Only'. Use with SDLC FIFO in DMA-driven environments. Interrupt occurs when EOF is detected. Data FIFO is not locked
RxDMA_FIFO_SpecialCondHandlerr	Handler for 'Recieve Interrupts on Special Conditions Only'. Use with SDLC FIFO in DMA-driven environments. Interrupt occurs when overrun or parity is detected. Data FIFO is locked
RxINT_SpecialCondHandlerr	Handler for 'Receive Interrupts on All Characters or Special Conditions Only'. Data FIFO is not locked
ExtStatChange_Handlerr	Handler for 'External Status Interrupt'. Used when Transmit Underrun/EOM interrupt is enabled.
Abort_Handlerr	Handler for 'External Status Interrupt'. Detect Abort condition
RxINT_Init	Initialize receive interrupt mode
TxDMA_Init	Initialize transmit DMA transfers
TxDMA_Start	Trigger transmit DMA transfers
TxDMA	Determine whether all characters are transferred in DMA mode
DisTxDMA	Terminate transmit DMA transfer
RxDMA_Init	Initialize receive DMA and trigger DMA transfer
RxDMA	Return receive DMA transfer status
DisRxDMA	Terminate receive DMA transfer
InstallUnderrun	Transmit Underrun/EOM or Abort interrupt handler
SetTxMethod	Initialize transmit method
SelectTxMethod	Prompt user for selection of transfer methods
SetRxMethod	Initialize receive method
SelectRxMethod	Prompt user for selectiom of receive methods
EndRest	Reset Master Interrupts
StartXfers	Initializes, manages and terminates transfers
ToggleSDLC_FIFO	Toggle SDLC FIFO enhancements
ToggleTxLevel	Toggle transmit FIFO interrupt levels
ToggleRxLevel	Toggle receive FIFO interrupt levels
ControlExtStatIE	Enables external/status interrupts
SelectRxIntMode	Prompts user to select receive interrupt modes
SetBaudRate	Prompts user to select baud rate and initialize the baud rate accordingly
ReadChar	Display the character read
WriteChar	Display the character to be written

A.1.7 ESCC182.C functions:
The escc182 module is the main program in the device driver demonstration software. It interfaces to the user and distributes the parameters defined by the user programs to various modules.

Intro	Prints greeting messages
SubMenu	Prints submenu
OtherMenu	Control sub-menu
InitTxRxBuffers	Initialize transmit and receive buffers

A.1.8 SYS_IO.C functions (Interface to Microtec C Compiler):
sys_io functions are low level interface to Microtec C libaries. The user may need to modify the implementations if a different C compiler is used. The user may also need to modify the memory and I/O address assignment if a different hardware platform is used.

outp	Output a data to the console
inp	Input a data from the console
read	Read a character from the console
kbhit	Return whether the console key is hit
getch	Get a signed character from the console
write	Write a character to the console
putch	Put a character to the console

A.1.9 Microtec C libraries:
The Microtec C libaries used in the program are as follows:

printf	Print characters on the formatted output device
scanf	Read characters from the formatted input device
putc	Write a character to console
puts	Write a string to standard output device
sscanf	Unformats a string
toupper	Converts characters to uppercase

Note: These libraries are developed by Microtec and linked with the compiled code. This kit does not include these modules. Customer need to contact Microtec for further details.

A.2 ASSEMBLY MODULES

The assembly modules are interrupt service routines. Based on the different interrupting conditions, status before interrupt is pushed onto user stack, control is transferred to corresponding handler compiled separately in LLC.C, and status before interrupt is restored after handler processing is completed.

ASSEMBLY MODULE NAME	CORRESPONDING INTERRUPT HANDLER IN LLC.C
txint0	TxINT_BuffNotFullHandlerr
txint1	ESCC_TxINT_BuffEmptyHandlerr
rxint0	RxINT_CharAvailHandlerr
rxint1	ESCC_RxINT_CharsAvailHandlerr
rxint2	RxDMACharAvailHandlerr
rxspe0	RxINT_SpecialCondHandlerr
rxspe1	RxPOLL_DMA_SpecialCondHandlerr
rxspe2	RxDMA_FIFO_EOF_Handlerr
rxspe3	RxDMA_FIFO_SpecialCondHandlerr
extint	ExtStatChange_Handlerr
unexpect	Unexpectedd
abortint	Abort_Handlerr

Note: These modules are assembled by Microtec Z180 Assembler. The customer need to contact Microtec to recompile the code. Modification may be necessary if other assemblers are used.

B. MIMIC DEVICE DRIVER (MPU SIDE) SUPPORT ROUTINES

This section describes the routines that are used by the MIMIC device driver software demonstration software (MPU Side). Note that C modules of same name also reside in other Device Drivers, yet the contents of the module may be modified for the application.

B.1 C MODULES

The MIMIC device driver demonstration software is made up of nine different C modules. Details of the modules are described as follows:

B.1.1 ASYNC.C functions

This module is the main program code. Upon device driver startup, this module is used and calls all other modules and functions.

main	Calls functions that initialize the transmit and receive buffers. Then program goes into a continuous loop; displays menu, input choice of command, execute command, then back to displaying menu.

B.1.2 MIMIC.C functions

This module deals with the management of all Z182 register values. All registers are arranged into arrays for register names and values (hex value read/written to register).

MIMIC_Reg_Init	Initializes MIMIC registers and values into arrays labelled Z182_550_Name and Z182_550_Off.
MISC_Reg_Init	Initializes Miscellaneous Z182 registers and values into arrays labelled Z182_MISC_Name and Z182_MISC_off.
Port_Reg_Init	Initializes Parallel Port registers and values into arrays labelled Z182_PORT_Name and Z182_PORT_Off.
Z182_MimicWRRs	Displays MIMIC Read/Write registers and values
Z182_MISC_WRRs	Displays Z182 Miscellaneous registers and values
Z182_PORT_WRRs	Displays Parallel Port registers and values
Z182_MimicRRs	Displays MIMIC Read only registers and values
Z182_Mimic_ChgReg	Displays all MPU addressable MIMIC registers and values and allows modification of register values.
Z182_MISC_ChgReg	Displays Z182 Miscellaneous registers and values, and also allows modification of register values.
Z182_PORT_ChgReg	Displays Parallel Port registers and values, and also allows for modification of register values.
MIMIC_EnTHRInt	Enables THR not empty interrupts
MIMIC_DisTHRInt	Disables THR not empty interrupts
MIMIC_EnRBRInt	enable RBR empty interrupt
MIMIC_DisRBRInt	Disables RBR empty interrupts
MIMIC_ResetHighestIUS	Resets MIMIC highest interrupt under service
MIMIC_SetMIE	Sets Master MIMIC Interrupt enable
MIMIC_ResetMIE	Resets Master MIMIC Interrupt enable
MIMIC_StartRBR_DMA_Xfer	Enables and initiates DMA Ch.0&1 transfer from memory to MIMIC RBR register/fifo.
MIMIC_StopRBR_DMA_Xfer	Disables DMA Ch.0
MIMIC_StartTHR_DMA_Xfer	Enables and initiates DMA Ch.0&1 transfer from MIMIC THR register/fifo to memory.
MIMIC_StopTHR_DMA_Xfers	Disables DMA Ch.1
MIMIC_Toggle_THR_Timer	Enable/Disables THR delay timer

MIMIC_Toggle_RBR_Timer	Enables/Disables RBR delay
Modify_TTTC	Allows modification of THR delay time constant
Modify_RTCR	Allows modification of RBR delay time constant
Modify_TTTC	Allows modification of THR timeout constant
Modify_RTTC	Allows modification of RBR timeout constant
MIMIC_Toggle_RBR_Timeout	Enable/Disables RBR timeout timer
MIMIC_Toggle_THR_Timeout	Enable/Disables THR timeout timer
MIMIC_Set_Level_01	Sets transmit FIFO interrupt trigger level at 1 byte
MIMIC_Set_Level_04	Sets transmit FIFO interrupt trigger level at 4 bytes
MIMIC_Set_Level_08	Sets transmit FIFO interrupt trigger level at 8 bytes
MIMIC_Set_Level_14	Sets transmit trigger FIFO interrupt level at 14 bytes
XMIT_Trigger_Level	Displays transmit trigger level choices, inputs choice, and executes
MIMIC_Toggle_IE_FCR	Enables/Disables Fifo control register interrupt
MIMIC_Toggle_IE_MCR	Enables/Disables Modem control register interrupt
MIMIC_Toggle_IE_LCR	Enables/Disables Line control register interrupt
MIMIC_Toggle_IE_MIE	Enables/Disables all MIMIC interrupts
MIMIC_Enable_INT	Displays interrupt enable choices, inputs choice, and executes.

B.1.3 TRACE.C functions:

The MIMIC device driver traces all data transfer to/from MIMIC device. The trace "time-stamps" each MIMIC transfer and lists table with Time-stamp and number of bytes sent/received in transfer.

TraceBuffer	Assigns descriptor to Trace data buffer and initializes Z182 timers
TraceData	Fills Trace buffer with Trace Data
DisplayAllTrace	Formats and displays all Trace Data
DisplayValueChange	Displays only trace data whose consequetive traces differ in bytes transferred.
ClearBuffer	Reassignes buffer pointer to buffer base; all data is lost
FreeBuffer	Frees memory assigned for Trace buffer
PRT0_HANDLERR	Inputs data from timer0 and resets Interrupts under service
PRT0_HANDLERR	Inputs data from timer1 and resets Interrupts under service
main	Manages entire tracing routine

B.1.4 CPU180.C functions:

Initializes Z182's DMA registers and provides console interrupt routines..

DMA_Data_Init	Initializes an array for DMA register names and values
ESCC_SetIntHdlr	ESCC serial communication interrupt handlers
DMA180_Init	Initialization of DMA registers
DMA180_ShowReg	Displays DMA registers and values
DMA180_ChgReg	Allows modification of DMA registers
DMA180_StartXfer	Enables assigned DMA channel
DMA180_StopXfer	Disables DMA channels
DMA180_GetCount	returns byte count for assigned channel
DMA180_Done	returns completion status of assigned DMA channel

B.1.5 TOHOST.C functions:

The functions in this code provide communications to console terminal.

GetByte	gets a byte of data from the console.
GetWord	gets a group of bytes from the console.
SendByte	sends a byte of data to the console.
SendWord	sends a group of bytes to the console.

GetChar	Get data in character format from console
GetUnsigned	Get data in unsigned format from console
GetLine	Puts input data into keyboard buffer

B.1.6 UTILS.C functions:
Various utilities for Device Driver (some functions not used in MIMIC Driver).

getchf	gets a character and compares to list of valid characters. Valid character ASCII code is returned.
palloc	alerts user when unable to allocate memory
HitKeyToGoOn	displays "Hit key to go on ", and waits for key entry
WriteMenu	Displays menu entries
AreYouSure	Prompts user to verify a choice

B.1.7 BUFFERS.C functions:
This section of code provides functions for initialization of transmit and receive buffers.

SetToValue	Initializes memory buffer to required size
InitBuffers	Initializes buffer and determines maximum data bytes in buffer. If contents of buffer are used for transmission, buffer is initialized with data.
AllocateBuffer	Assigns a valid pointer value to Buffer pointer variable
SetToPattern	Fills assigned data buffer with pattern
SetToSuite	Fills assigned data buffer with bytes starting at 00h and filling each consecutive byte with incremented value.
SetToZero	Fills data buffer with zeros (00h).
SetBuffer	Resets buffer to zero, prompts user to choose data to fill buffer, and fills data buffer with requested data and length.
BlankBuffer	Initializes buffer with zero data using SetToZero Function
DisplayBuffer	Displays the data buffer with both Hex and ASCII
ShowBuffer	Displays buffer information such as address and size
CompareBuffers	Compares transmit and receive buffers and attempts to find first index where buffers differ.

B.1.8 SYS_IO.C functions:
Functions in this source code manage input output routines to the ASCI channel 0 console.

delay	introduces delay; Simple 0fh countdown to zero
outp	ouputs data to I/O address specified
inp	input data from I/O address specified
read	reads specified number of character and place data in buffers
kbhit	tests for any keyboard entry
getch	Waits for receive data register full and gets this data
getche	Similar to 'getch' function but utilizes WR function
write	writes specified number of characters from buffers
putch	Simple method to write character to screen

B.1.9 FIFO.C functions:
This module contains functions used for FIFO usage on the MIMIC and ESCC. Some general MIMIC functions are also found here.

OUT_CHB	Outputs data to ESCC CHB
IN_CHB	Inputs data from ESCC CHB

PurgeBuffer	Purges Data in THR FIFO
PurgeAll	Clears contents of MIMIC status registers, trace buffers, and resets highest interrupt under service.
InitTxBuffer	Sets global variables used for transmission.
InitRxBuffer	Sets global variables used for reception.
CompareTxRx	Compares Tx and Rx buffers and sends results to monitor
TxPOLL_Init	Sets global variable used for Transmiting in polling mode
RBR_Burst	Allows for burst transfers in polling mode
TxPOLL	Transmits in polling mode and creates trace data
RxPOLL_Init	Sets global variables used for Reception in polling mode
RxPOLL	Receives in polling mode and creates trace data
TxINT_Init	Enables transmit interrupts and installs interrupt handlers
DisTxINT	Disables RBR Interrupts
RxINT_Init	Enables receive interrupts and installs interrupt handlers
RxINT	Checks status of transfer completion
DisRxINT	Disables interrupts and removes handlers
Unexpected	Unexpected Interrupt handler, displays "unexpected interrupt"
Bugg	Interrupt handler for possible bugs in program
MIMIC_INT_CHAR_AVAIL	MIMIC THR Interrupt handler routine
MIMIC_TOO_HANDLER	MIMIC THR Timeout with data in FIFO Interrupt Handler
MIMIC_INT_FIFO_EMPTY	MIMIC RBR Interrupt Handler
MIMIC_MCR_HANDLER	MIMIC Modem Control Register Write Interrupt Handler
MIMIC_LCR_HANDLER	MIMIC Line Control Register Write Interrupt Handler
MIMIC_DIV_HANDLER	MIMIC Division Latch LS/MS Write Interrupt Handler
MIMC_FCR_HANDLER	MIMIC FIFO Control Register Write Interrupt Handler
TxDMA_Init	Initializes DMA registers and variables. Initiates data transfer
TxDMA	checks status of transfer completion
DisTxDMA	Disables Tx DMA transfers
RxDMA_Init	Initiates Rx DMA data transfer
RxDMA	checks status of Rx DMA transfer completion
DisRxDMA	Stop Rx DMA transfer
ReadChar	Reads data from MIMIC THR
WriteChar	Writes data to MIMIC RBR
ShowStatus	Displays method of data transmission
ControlMenu	Displays MIMIC Device driver Menu
SelectTxMethod	Allows selection of Rx transfer type
SetRxMethod	Selects interrupt modes depending on Rx transfer type
SelectRxMethod	Allows selection of Tx transfer type
InitTx	Initialization type chosen according to selected transfer type
Tx	Initiates Tx data transfer
EndTx	Disables Tx data transfer
InitInterrupts	Determines necessary interrupt handlers and installs them
InitRx	Initializes Rx interrupts and handlers
Rx	Initiates Rx data transfer
EndRx	Disables Rx data transfers
EndRest	Removes handlers and removes all interrupts
StartXfers	Initiatializes transfers, starts transfer, and ends transfer
InitMIMIC	Initializes MIMIC registers and loads MIMIC interrupt handlers
TraceTransfer	Provides menu to choose trace type and displays selected trace
Change_WR12	Changes ESCC CHB time constant - low byte
Change_WR13	Changes ESCC CHB time constant - high byte

C. MIMIC DEVICE DRIVER (PC HOST SIDE) SUPPORT ROUTINES

This section describes the routines that are used by the MIMIC device driver software demonstration software (PC HOST Side). Note that C modules of same name also reside in other Device Drivers, yet the contents of the module may be modified for the application.

C.1 C MODULES

The details of the modules are described as follows:

C.1.1 ASYNC.C functions

This module is the main program code. Upon device driver startup, this module is used and calls all other modules and functions.

main	Calls functions that initialize the transmit and receive buffers. Then program goes into a continuous loop; displays menu, input choice of command, execute command, then back to displaying menu.

C.1.2 UTILS.C functions:

The UTILS module controls the I/O to console terminal.

getchf	Get character
HitkeyToGoOn	Printf a message
palloc	Printf "...failed to allocate mem.."
WriteMenu	Write baudrate menu
AreYouSure	Printf a message

C.1.3 TOHOST.C functions:

The functions in this code provide communications to console terminal.

GetByte	gets a byte of data from the console.
GetWord	gets a group of bytes from the console.
SendByte	sends a byte of data to the console.
SendWord	sends a group of bytes to the console.

C.1.4 CIRCBUFI.C functions:

Functions to manage circular buffer used in Device Driver.

BufferI_Full	Test for buffer full
BufferI_Empty	Test for buffer empty
PutInBufferI	Place char in buffer
GetFromBufferI	Get char from buffer
StringCircBufI	Transfer circ buffer into a string
InitCircBufI	Initialize circular buffer

C.1.5 BUFFERS.C functions:
Initializes and manages transmit and receive buffers.

InitBuffers	Set size of buffer
AllocateBuffer	Assigns buffer type
SetToPattern	Sets specific pattern to be transferred
SetToSuite	Sets buffer1=1..buffern=n
SetToZero	Sets values to zero
SetBuffer	Resets whole buffer to zero
BlankBuffer	From addrs to addrs+size,zero data
DisplayBuffer	Display buffer contents
ShowBuffer	Printf message, showing buffer data
CompareBuffers	Compares two buffers, returns index where they differ

C.1.6 DMA37.C functions:
Controls miscellaneous DMA operations.

DMA37_InitXfer	Initialize DMA related registers
DMA37_StartXfer	Initiate DMA transfer
DMA180_GetCount	Get DMA transfer count

C.1.7 PIC.C functions:
Initializes interrupt routines..

PIC_ResetIUS	Rest Interrupt under Service
PIC_EnInt	Enable interrupt as specified
PIC_DisInt	Diable interrupt as specified

C.1.8 UART.C functions:
This module includes commands for the MIMIC device.

SelectComPort	Printf the com port menu
Z182_Mimic WRRs	Printf mimic r/w, w only menu
Z182_MimicRRs	Printf com port
Z182_Mimic_ChgReg	Mimic change register routine
MIMIC_RBR_CharAvailable	Poll LSR's data ready
MIMIC_Read	Read mimic register
MIMIC_THR_BufferEmpty	Poll LSR's data ready
MIMIC_Write	Write to mimic register
MIMIC_EnTHRInt	Enable IER's THRE interrupt
MIMIC_DisTHRInt	Disable IER's THRE interrupt
MIMIC_EnRBRInt	Enable IER's RBR interrupt
MIMIC_DisRBRInt	Disable IER's RBR interrupt
MIMIC_Taggle_IE_LSR	Change IER's LSR enable bit
MIMIC_Taggle_IE_MSR	Change IER's MSR enable bit
MIMIC_SetMIE	Set HINTR to 3-state in MCR
MIMIC_SetRTS	Set RTS in Modem Control Reg
MIMIC_ResetRTS	Reset RTS in Modem Control Reg
MIMIC_ResetMIE	Resets HINTR back in MCR
MIMIC_Set_Level_01	Set mimic receiver level to 1
MIMIC_Set_Level_04	Set mimic receiver level to 4

MIMIC_Set_Level_08	Set mimic receiver level to 8
MIMIC_Set_level_14	Set mimic receiver level to 14
PC_RCVR_Trigger_Level	Printf the trigger level menu
MIMIC_Taggle_FIFO_Enable	enabled on the FIFO Control Reg
MIMIC_Taggle_DMA_Mode	DMA multibyte enabled on FCR
MIMIC_Reset_XMIT_FIFO	XMIT Fifo reset on FCR
MIMIC_Reset_RCVR_FIFO	RCVR Fifo reset on FCR

C.1.9 FIFO.C functions:
Manages certain MIMIC and ESCC FIFO routines.

UART_IntHandler	PC-side interrupt handler
UART_InitInterrupt	Interrupt initialization
UART_DisInterrupt	Disable interrupt routine
OUT_CHB	Output to ESCC CHB
IN_CHB	Input from ESCC CHB
PurgeBuffer	Purge data fifo
PurgeAll	Purge data, MCR, LCR etc..
InitTxBuffer	Set global variables for transmitter
InitRxBuffer	Set global variables for receiver
CompareTxRx	Compare Tx and Rx buffers
TxPOLL_Init	Reset Tx write count to 0
RBR_Burst	Burst count for RBR
TxPOLL	Poll number of Tx write counts
RxPOLL_Init	Initialize DMA transfer
RxPOLL	Poll of receive write number
Unexpected	Printf "unexpected interrupt"
Bugg	For debugging purposes
MIMIC_INT_CHAR_AVAIL_HANDLER	Handler for Int char availbl
MIMIC_TOO_HANDLERR	THR timeout handler
MIMIC_INT_FIFO_EMPTY_HANDLERR	Handler for FIFO empty condition
MIMIC_MCR_HANDLERR	Modem Control Reg write Int handler
MIMIC_LCR_HANDLERR	Line Control Register write int handler
MIMIC_DIV_HANDLERR	Division Latch write interrupt handler
MIMIC_FCR_HANDLERR	Fifo Control Register write int handler
TxDMA_Init	Tx DMA Initialization
TxDMA	Determine Tx write count
DisTxDMA	Stop mimic and Z180 DMA transfer
RxDMA_Init	Rx DMA initialization
RxDMA	Determine Rx write count
DisRxDMA	Stop MIMIC and Z180 DMA transfer
ReadChar	Character read
WriteChar	Character to be written
ShowStatus	Printf RBR and THR transfer method
ControlMenu	Printf control menu
SetTxMethod	Set the transmit method
SelectTxMethod	Polling,int or DMA for Tx method
SetRxMethod	Set the receive method
SelectRxMethod	Polling,int or DMA for Rx method
InitTx	Initialize Tx for polling/int/DMA
Tx	Returns with method of transfer for Tx
EndTx	Returns with ending of transfer for Tx
InitInterrupt	Setup for initializing interrupts

InitRx	Printf messages for initializing Rx
Rx	Returns with ending of transfer for Rx
EndRest	Resets transfer, transfer bytes and ends transfer
InitMIMIC	Initializes mimic register
TraceTransfer	Display RBR and THR trace
Change_WR12	WR12 modify
Change_WR13	WR13 modify

D. HEADER AND MISCELLANEOUS FILES

Various header files are required to define the data types of the variables. It is included and compiled together with various C modules. The header files used in the device driver programs include the following:

Developed by Zilog
sys_io.h
cpu182.h
DMA.h
scc_df.h
settings.h
tohost.h
bastypes.h
scc.h
escc.h
esccio_d.h
utils.h
buffer.h
llc.h

Supported by Microtec (Compiler specific)
stdio.h
stdlib.h
ctype.h

Miscellaneous Files

Additional files are required in the Device Driver Demonstration Software.

ESCC.C
ESCC.C is the definition files used by the SCC.C module.

INIT.SRC
INIT.SRC is the assembly modules to initialize the configuration of Z80182 evaluation board.

Include Files
Include files are required to define the various data types of the variables. It is included and assembled together with various Assembly modules. The include files used in the Device Driver Demonstration Software are as follows:

sys.inc
confg.inc

APPENDIX E --- MEMORY AND I/O INTERFACE REQUIREMENT FOR THE Z80182 EVALUATUION BOARD

This appendix outlines the speed requirement on memory and peripherals for Z80182. The Z80182board is used as a design example.

BACKGROUND

(1) 20 MHz Z182 timing is based on the Z80182 Product Specification

20 MHZ Z182 SYSTEM REQUIREMENTS

(1) Memory Requirement: (Measured from Address Valid to Read Data Valid, see Fig.3)

Memory read access time < T1 Rise to T3 Rise - T1 rise to Address valid - Data Read setup time
= 2 * tcyc - tAD - tDRS
= 2(50) - 30 - 10
= 60 ns

==> EPROM and SRAM access time have to be less than 60 ns for 0 wait states access.

(2) Peripheral requirement:
(a) Peripheral read cycle (Measured from /RD Fall to Read Data Valid, see Fig.4)
(Assume that /IORQ and /RD is used to decode I/O Read Cycle)

Peripheral access time < T1 Fall to T3 Fall + Tw - T1 Fall to /RD Fall - Data Read setup time
= 3 * tcyc - tRDD1 - tDRS
= 3(50) - 25 - 10
= 115 ns (1 wait state)

==> Peripherals access time has to be less than 115 ns if one
I/O wait states are programmed in DCNTL register during I/O cycle. (IWI1 and IWI0 are '00')

(b) Interrupt acknowledge cycle
(Assume that /M1 is used for decoding Interrupt Acknowledge Cycle and /IORQ is used for initiating the interrupt vector read)

Daisy-Chain settling time : (Measured from M1 Fall to /IORQ Fall, see Fig.5)
= T1 Rise to Tw1 Fall - Clock Rise to /M1 Fall Delay + Clock Fall to /IORQ Fall Delay
= 2.5 * tcyc - tM1D1 + tIOD1
= 2.5(50) - 35 + 25
= 115 ns

Interrupt Vector Read Cycle Access Time : (Measured from /IORQ Fall to /M1 Rise, see Fig.5)
Interrupt read access time = Tw1 Fall to T3 Rise + Clock Rise to M1 Rise Delay - Clock Fall to /IORQ Fall delay - Data Read Setup Time
= 1.5 * tcyc + tM1D2 - tIOD1 - tDRS
= 75 + 40 - 25 - 10
= 80 ns

==> Daisy-Chain Settling Time must be less than 115 ns if two wait states are added in the

interrupt acknowledge cycle
==> Interrupt Vector Read within 80 ns

The memory and I/O requirement at 20 MHz Z80182 is summarised as follows:

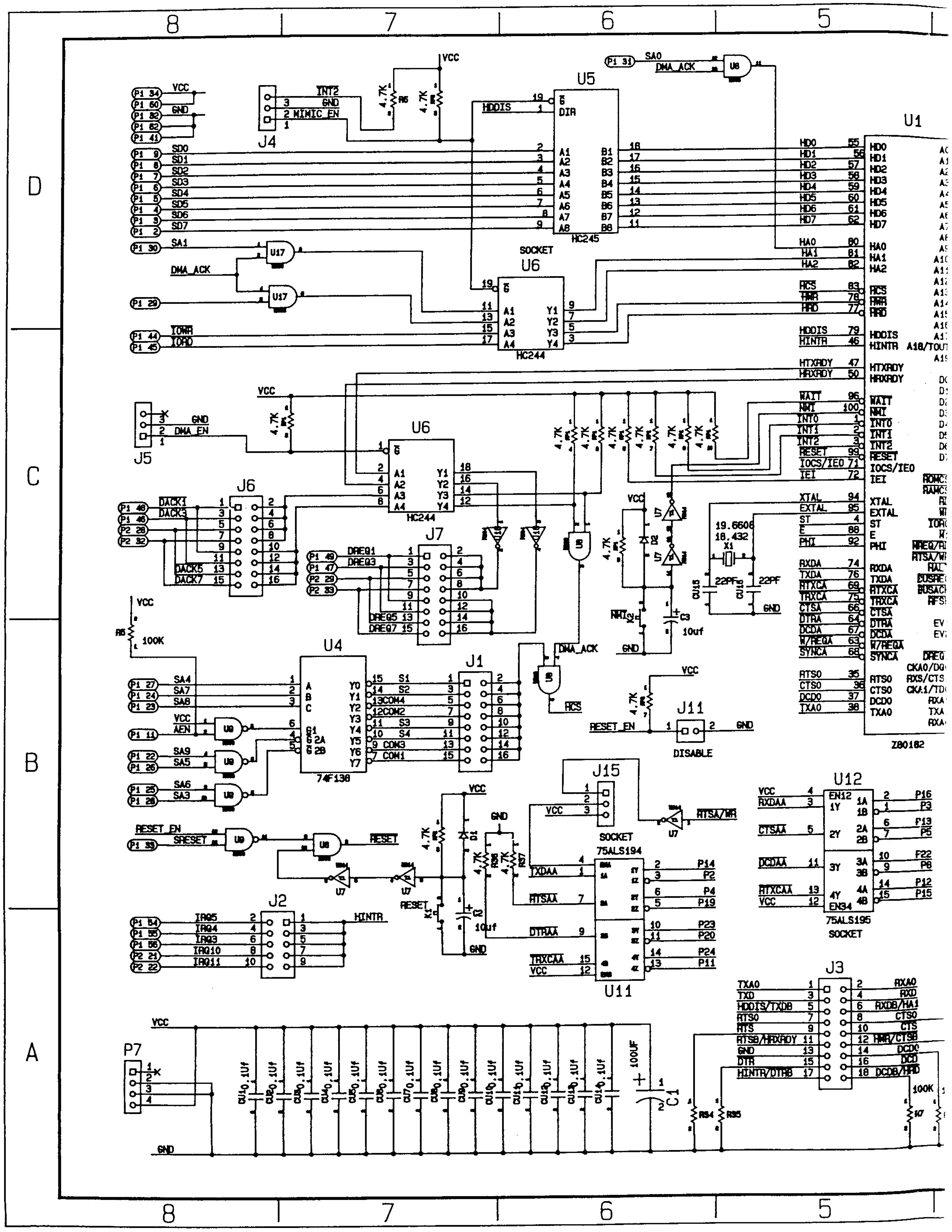
(a) Memory Access Requirements

MWI1:MWI0 of DCNTL	00	01	10	11
Mmeory wait states	0 wait	1 wait	2 wait	3 wait
Memory Access time	60 ns	110 ns	160 ns	210 ns

(b) I/O Access Requirements

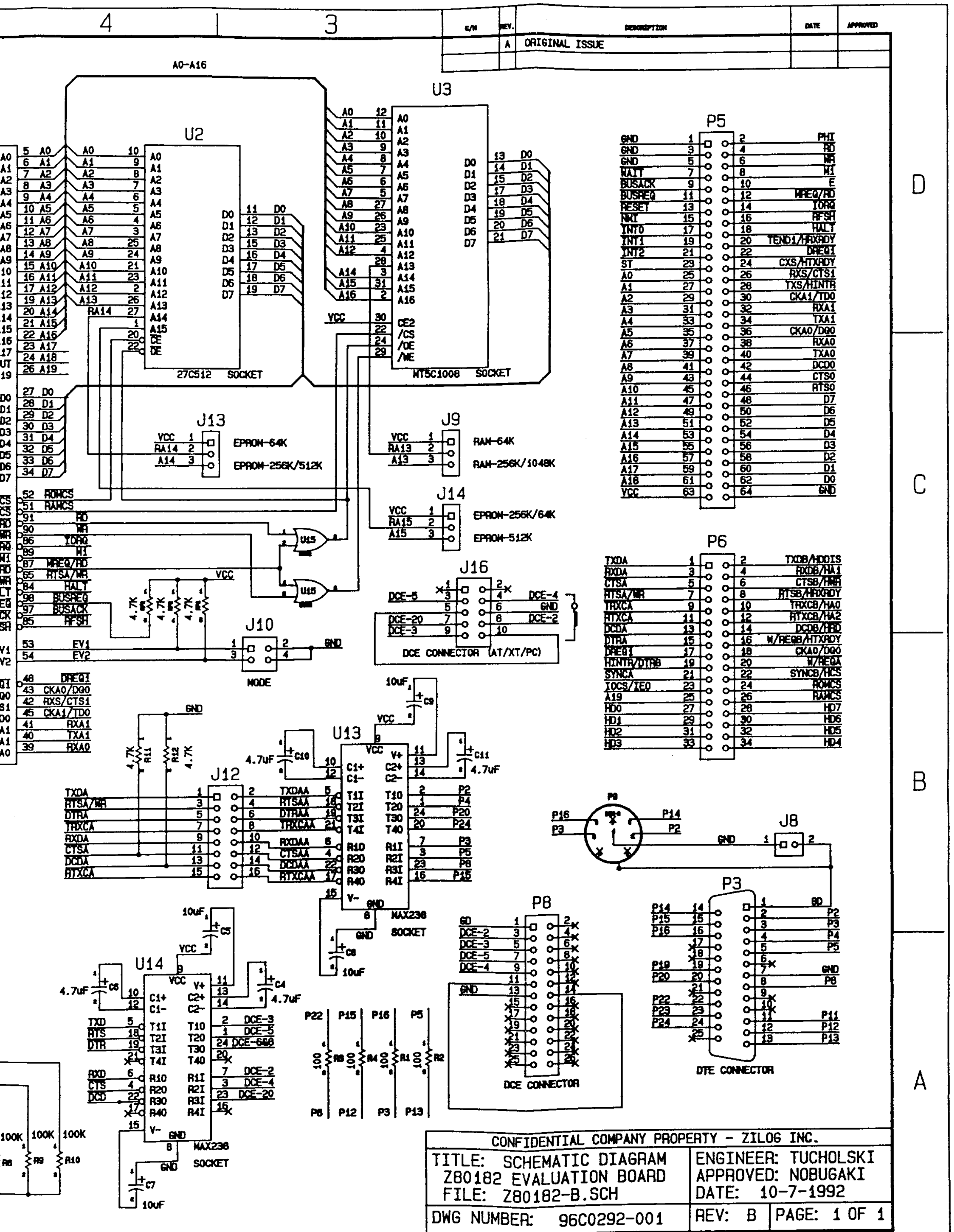
IWI1:IWI0 of DCNTL	00	01	10	11
I/O wait states	1 wait	2 wait	3 wait	4 wait
I/O Access time	115 ns	165 ns	215 ns	265 ns
INT0 interrupt acknowledge cycle	2 wait	4 wait	5 wait	6 wait
Daisy-Chain Settling time	115 ns	215 ns	265 ns	315 ns
Interrupt Vector Read Access Time	80 ns	80 ns	80 ns	80 ns

--oOo--



8
7
6
5
D
C
B
A
U1
Z80182
U5
HC245
SOCKET
U6
HC244
U4
74F138
U11
75ALS194
U12
75ALS195
SOCKET
J1
J2
J3
J4
J5
J6
J7
J11
DISABLE
J15
P7
VCC
GND
INT2
MIMIC_EN
DMA_EN
DMA_ACK
HDDIS
RESET
RESET_EN
SRESET
HINTR
HCS
RTSA/WR
19.6608
18.432
X1
22PF
10uf
4.7K
100K
100UF
C1
C2
C3
0.1Uf

S/N	REV.	DESCRIPTION	DATE	APPROVED
	A	ORIGINAL ISSUE		



CONFIDENTIAL COMPANY PROPERTY - ZILOG INC.

TITLE: SCHEMATIC DIAGRAM Z80182 EVALUATION BOARD FILE: Z80182-B.SCH	ENGINEER: TUCHOLSKI APPROVED: NOBUGAKI DATE: 10-7-1992
DWG NUMBER: 96C0292-001	REV: B PAGE: 1 OF 1
DWG SIZE: B	

88C0022-001