Z84C15 EVALUATION KIT

ZILOG

UM016001-0703

Z84C15 EVALUATION KIT
P/N Z84C1500ZC0

TABLE OF CONTENTS

1. DESCRIPTION

1.1 FUNCTIONAL DESCRIPTION

The Z84C15 Evaluation Kit serves multiple purposes. It demonstrates the use of the Z84C15 in a high performance, high integration, low cost environment. It also demonstrates its use as a test facility for both hardware and software.

The Z84C15 Evaluation Kit can be used as a test bench to debug and analyze hardware designs. Target boards with specific peripherals can be driven by the Z84C15 through its expansion interface.

A set of powerful software tools is included to assist the user in his application. Assembly and linking utilities are included with the Evaluation KIt.

1.2 HARDWARE DESCRIPTION

The Z84C15 Evaluation Kit is composed of the following elements:

1. The Z84C15 10MHz CMOS Intelligent Peripheral Controller.
2. A 28 pin EPROM socket.
3. A 28 pin SRAM socket.
4. One NMI switch.
5. One Reset switch.
6. One RS-232 serial interface.
7. Three expansion interfaces.

The Zilog Z84C15 Evaluation Kit provides the user with a platform for developing and debugging Z84C15 application hardware and software. The kit comes configured to run using the Debug Monitor. Serial communication with the host computer (PC) is accomplished via the SIO channel B of the Z84C15 with its baud rate set by the Z84C15 Counter/Timer 1.

As indicated on figure 2, P2 is a 40 pin header that mirrors the Z80 pin out. P3 is a 60 pin header that interfaces to the Z84C15 PIO, SIO, CTC,watch dog timer and external control lines. Jumpers within these three interfaces can be used to configure data, clock and control paths during stand alone operation of the Evaluation Kit. Figure 1 shows this relationship.

There are seven jumper groupings shown in figure 3. J1 is a connector for the /Reset and the /Watch Dog Timer Out. This is shipped with the connection open. J2 determines whether the Z84C15 is in evaluation mode. This jumper is shipped with the Z84C15 in non-EV mode . The Z84C15 could use either its on board clock generator or an external clock. Jumper J3 is shunted so that the on-chip generator is used. J4 determines the memory size used. With an 8Kx8 EPROM J4's pin 1 and 2 are shorted (note that A0-A12 are active and A13/A14 are unused in an 8Kx8). J5 determines the SRAM size. The Evaluation Kit is shipped with J5's pin 1 shunted to pin 2 (8Kx8 SRAM). J6 is an 8 pin header that allows selection between two Z84C15 counter/timers for the baud rate clock for the SIO channels. RS232 level shifting is performed by the MAX232. Input and output signals from the MAX232 are determined by the 12 pin jumper J7.

1.2.1 BLOCK DIAGRAM

Figure 1 shows the Z84C15 Evaluation Kit block diagram. This shows the Z84C15-memory-expansion relationship.

1.2.2 Z84C15

The Z84C15 is a CMOS 8-bit microprocessor integrated with the CTC, SIO, PIO, CGC and WDT into a single 100-pin quad flat pack package. The Z84C15 is upward compatible with the Z84015.

This high end superintegrated intelligent peripheral controller is targeted for a broad range of applications ranging from error correcting modems to enhancement/cost reductions of existing hardware using Z80-based discrete peripherals.

Its features include:

1. Z84C00 Z80 CPU
2. On-chip two channel SIO (Z80 SIO).
3. On-chip four channel Counter Timer Controller (Z80 CTC).
4. On-chip two 8 bit ports (Z80 PIO).
5. Built-in Clock Generator Controller (CGC).
6. Built-in Watch Dog Timer (WDT).
7. Noise filter to CLK/TRG inputs to the CTC.
8. Power-on Reset.
9. Additional two chip select pins.
10. A 32-bit CRC for Channel A of SIO.
11. Wait state generator.
12. Simplified EV mode selection.
13. Schmitt-trigger inputs to transmit and receive clocks for the SIO.
14. Crystal divide-by-one mode.

1.2.3 EXPANSION INTERFACE

Referring to figure 2, P2 and P3 are provided to allow for proper connection of plug-on cabled target boards to the Evaluation Kit. The interface is sectioned off into functional groups :Z84C15 CPU (P2), Z84C15 peripheral signals (P3) and RS-232-C signals (P1).

1.3 SOFTWARE DESCRIPTION

The Z84C15 Evaluation Kit comes with a set of software packages. Utilities for assembling and linking source codes are contained with the kit. Zilog's Z800 Cross Assembler (ASM800) takes a source file containing Z8, Z80 or Z8000 assembly language statements and translates it into a corresponding object file. ASM800 can also produce a listing containing the source code, object code and comments. Zilog's Microprocessor Universal Format for Object Modules (MUFOM) utilities allow the programmer to combine, display and load machine-language object modules.

The Evaluation Kit contains three 5.25", double density, double sided diskettes for use in IBM PC's or IBM PC compatibles:

1. ZASM diskette contain ASM800 and utilities.
2. MOBJ diskette contain MUFOM utilities.
3. SRC diskette contain example application object/source codes, debug monitor source code, batch file examples

Please refer to the individual "READ.ME" files contained in each of the above diskettes for a complete listing of the essential files.

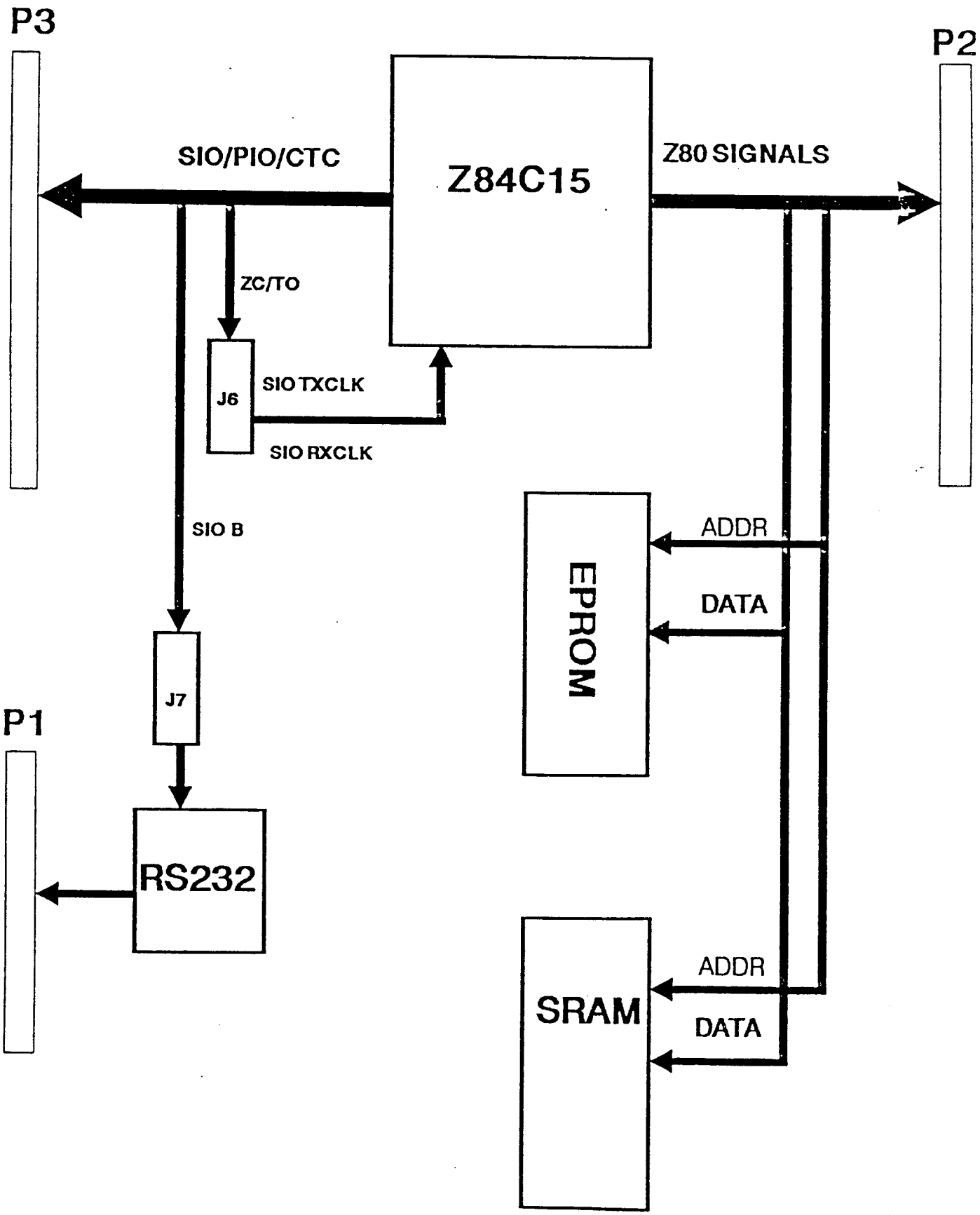
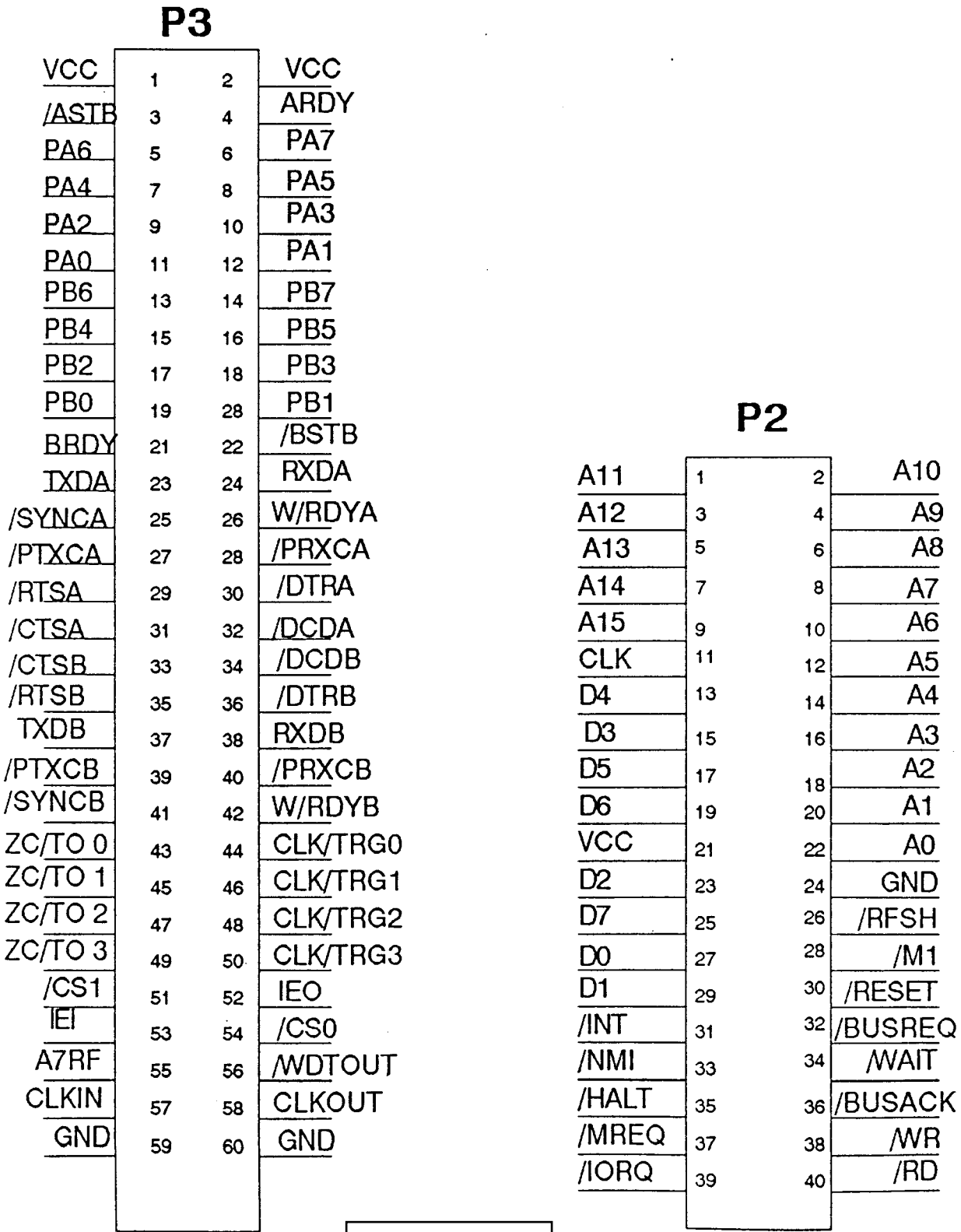
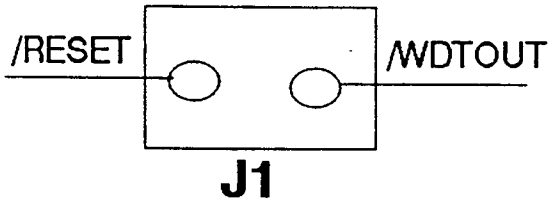


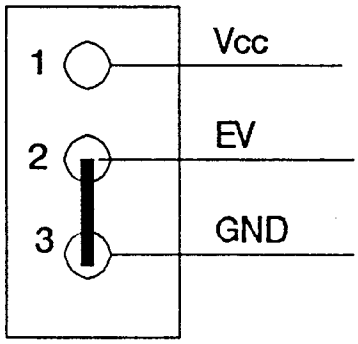
FIGURE 1 BLOCK DIAGRAM

FIGURE 2
Z84C15 I/O SIGNALS

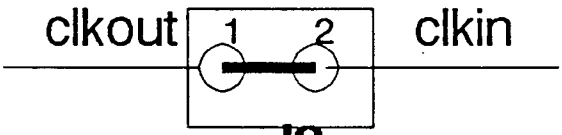




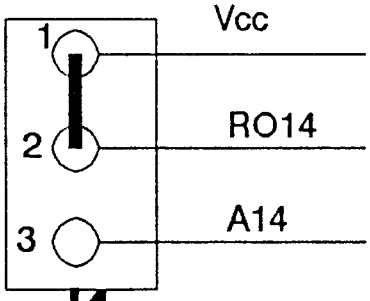
J1



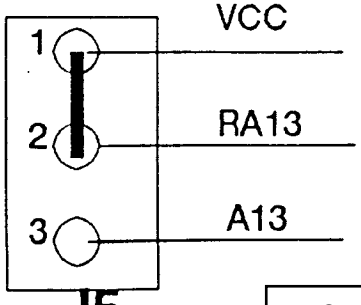
J2



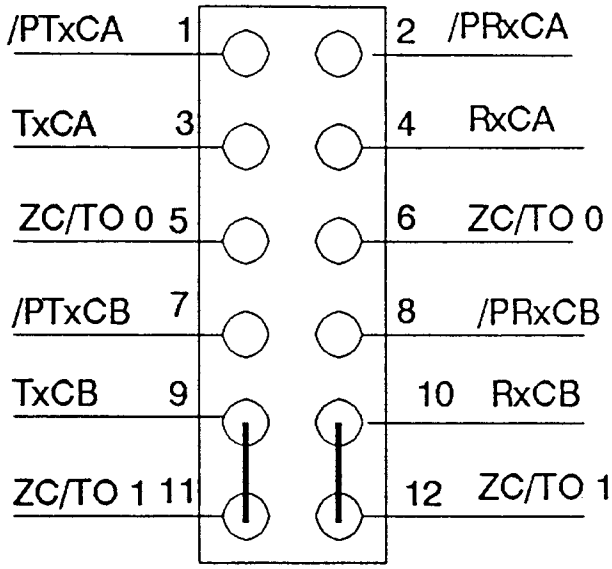
J3



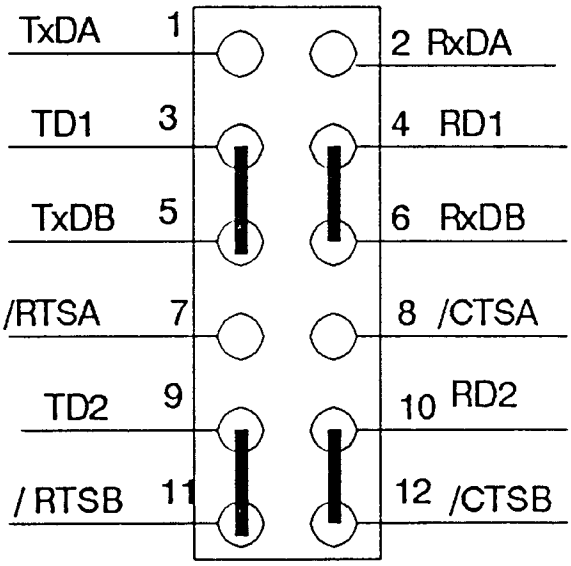
J4



J5



J6



J7

JUMPERS AS SHIPPED

FIGURE 3

2. INSTALLATION

2.1 INVENTORY CHECKLIST

The included diskettes are double sided, double density, 360K capacity for use in I.B.M. PC'S AND 100% compatibles.

1. Z84C15 Evaluation Board P/N 99C0184-001.
2. RS232 cable P/N 59-0445-00.
3. SRC software diskette P/N 25C0100-001:
   SRC.EXE Compressed source code files.
   READ.ME List of files on the diskette and instructions to install the files.
4. ZASM software diskette P/N 25C0097-001:
   ZASM.EXE Compressed Z80, Z8 and Z8000 assemblers.
   READ.ME Instructions to install the files
5. MOBJ software diskette P/N 25C0098-001:
   MOBJ.EXE Compressed MUFOM utilities
   READ.ME Instructions to install the files.
6. Z84C15 Preliminary Product Specification P/N 00-2507-02
7. Zilog Components Short Form P/N 00-2490-01
8. asm800/Z800 Cross Assembler Users' Guide P/N 03-8232-02
9. Zilog Universal Object File Utilities Users' Guide P/N 03-8236-03

2.2 WHAT ELSE IS NEEDED?

In addition to the material included in the kit, the following elements are needed:

1. A 5.0V power supply(delivering 0.5A).
2. An IBM PC or compatible with a 5.25 inch floppy disk drive.

The PC will work as a terminal emulator. It will communicate with the Z84C15 via a serial link on its COM1 or COM2 port.

2.3 SOFTWARE INSTALLATION

The file Z84C15.EXE is a self extracting archive file with compression utilities. To install the software you will require approximately 500K bytes free on your hard disk. An IBM PC XT was used in the example installation below.

Example Installation:

1. Choose which hard drive to use (e.g. C).
2. Make a directory on the chosen hard disk (e.g. ZILOG).
3. Insert the diskette labeled ZASM.EXE into Drive A:

Type:

```
A:ZASM -D C:\ZILOG\ZASM
```

This will put asm800 and its utilities into your directory C:\ZILOG\ZASM.

4. Insert the diskette labeled MOBJ.EXE into drive A: and type:

```
A:MOBJ -D C:\ZILOG\ZASM
```

This will put the MUFOM utilities into your directory C:\ZILOG\ZASM.

5. Insert the diskette labeled SRC.EXE into drive A: and type:

```
A:Z84C15 -D C:\ZILOG\SRC
```

This will create subdirectory C:\ZILOG\SRC. Source codes will be loaded into this directory.

6. Add the drive and the directory names to the path command in your AUTOEXEC.BAT file (e.g. C:\ZILOG\ZASM);. Add the command SET ASM800=C:\ZILOG\ZASM\ASM800 to your autoexec.bat file.

8. Reboot your system.

2.4 HARDWARE INSTALLATION

1. JUMPERS AS SHIPPED

J1	/RESET TO /WDT OUT	OPEN
J2	EV	2-3
J3	CLOCK SOURCE	1-2
J4	EPROM SIZE SELECT(27C64)	1 TO 2
J5	SRAM SIZE SELECT(8Kx8 SRAM)	1 TO 2
J6	SIO BAUD RATE SOURCE	9 TO 11 and 10 TO 12
J7	MAX 32 INPUT SELECT	3 TO 5
		4 TO 6
		9 TO 11
		10 TO 12

2. Connect the RS-232 port to either a terminal or a PC. See description of "Term" in the TARGET-HOST INTERFACE section.
3. Set the terminal or PC to 9600 baud, 1 start bit, 1 stop bit, no parity, and 8 bit/character.
4. Connect a 5V power supply to the board.
5. Turn the power supply on.

3. DEBUG MONITOR

3.1 INTRODUCTION

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

The commands should be in lower case and followed by depressing the "Enter" key. Addresses should be entered in hexadecimal.

3.2 DEBUG MONITOR DESCRIPTION

The commands recognized by the debug monitor are as follows :

COMMAND TYPE	SYNTAX	FUNCTION
BREAK POINT	b addr	Set break point at 'addr. 'When a user program fetches an instruction from 'addr', execution is stopped, the contents of all user registers are displayed, the break point is cleared and control is passed to the debug monitor.
COMPARE	c addr1 addr2 count	Compare memory starting at 'addr1'with memory starting at 'addr2' for 'count' bytes. Monitor displays differences.
DISPLAY REGISTERS	r	Displays all user registers.
	r reg	Displays register 'reg' and go into input mode to allow user modification. Q to quit.
DISPLAY MEMORY	d addr count	Display memory starting at'addr' for 'count' bytes.
	d addr	Display content of 'addr' and go into input mode to allow modification if desired. Q to quit.
FILL MEMORY	f addr1 addr2 byte	Fill memory from 'addr1' to 'addr2' with 'byte' value.
GO	g	Pass control to user program starting at current user program counter.

INPUT	i ppaa	Read from port 'aa' on page 'pp' and display value. Page address is 00h if not specified.
JUMP	j addr	Pass control to user program starting at 'addr'.
LOAD	l path\filename.ext	Load 'path\filename' file in Intel-hex format into RAM. User program counter is automatically updated with starting address specified in the hex file.
MOVE MEMORY	m dest src length	Move 'length' bytes from 'src' to'dest' address.
OUTPUT	o ppaa val	Write 'val' to port 'aa' on page 'pp'. Page address is 00h if not specified.
SET MEMORY	s addr byte	Set content of 'addr' to 'byte'.

3.3 MONITOR PROGRAM

In the supplied program, location 38H contains the instruction RETI and location 66H contains the instruction RETN. The user can modify these locations to meet specific needs in conjunction with using Interrupt Mode 1 and Non Maskable Interrupt.

The debug monitor uses single character commands. To add a new command, simply add a command routine and add two new lines in the command interpretation portion of the code labelled COM? :

CP 'x'	where x is any single character.
JP Z,cmd_rtine	where cmd_rtine is the starting address ofthe newly added command routine.

There are two stacks in the Debug Monitor. One is for the monitor itself, and the other is for user programs. The monitor stack starts at address STACK and is 128 bytes deep. The user stack is stored in the RAM variable 'SAVSTK'. The initial value of the user stack pointer is 3FFFH, but it can be modified manually or by a user program.

User register contents are stored in the RAM area between 'STATUS' and 'SAVSTK'. This storage structure may not be changed. It can, however, be moved to another address, but the whole structure must always be allocated on the same 256 byte page. Likewise, for the pointer table 'REGTAB' the order may not be changed and it must fit on one 256 byte page.

Interrupts are disabled when the Debug Monitor is in control. When a user program comes to a break point, control is returned to the Debug Monitor and further interrupts are disabled. They will be enabled again before control is passed to the user program using the JUMP or GO commands.

The RST 30H instruction is used for BREAK POINTS. The object code of this instruction is 0f7H and is patched at the desired break point address. Therefore, break point can only be set in RAM resident code. Only one break point can be set at a time. A break point is automatically cleared when a user program reaches the break point address. The starting address of the break point service routine is stored in the RAM variable 'BRKRT' located at the lowest RAM address.

3.4 TARGET-HOST INTERFACE

The program TERM.EXE provides a facility for simple terminal emulation and allows down loading of user code in Intel Hex format. The Board can be connected to either COM1 or COM2 of the PC. Type TERM to invoke the 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 then 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 Debug Monitor as supplied is configured for 9600 baud, 8 bits, 1 stop bit, and no parity. To stop display from scrolling,

type CTRL-S, and to continue type CTRL-Q. CTRL-X aborts on going display and CTRL-C terminates the emulation program.

4. SAMPLE SESSION

An IBM PC XT with a 360k floppy disk drive was used in the sample session below. USER15.HEX is the example software resident on the SRC diskette which may be used by you in trying out some of the Debug Monitor commands that was listed out on page 9.

1. Refer to "SOFTWARE INSTALLATION" section regarding the loading of the necessary software needed to get started.

2. Necessary hardware hookups are described in the "HARDWARE INSTALLATION" section.

3. Invoke the Debug Monitor as shown below(per "SOFTWARE INSTALLATION", you should be in directory C:\ZILOG\SRC):

```
C:\ZILOG\SRC>TERM
```

4. The Debug Monitor prompt is shown below:

```
>
```

5. Wake up your Debug Monitor by pressing the reset button. If you don't get the prompt below then check to make certain that you have 5V from your power supply.

```
>***** Z80 DEBUG MONITOR *****
```

6. Once in the Debug Monitor, load the example program provided for you.

```
>L USER15.HEX
```

7. This example program will be executed by typing:

```
>G
```

8. The contents of the Z80 registers will be shown at the program's break point. This program resides on SRAM location 3000HEX so try your own break points at different locations. Remember that break points are allowed only at the beginning of an opcode (so look at the listing!).

9. To exit from the Debug Monitor hit Ctrl-C on your terminal. This will end your session and should bring you back to your PC's operating system.

APPENDIX 1: BOARD IMPLEMENTATION

The memory and I/O address maps are shown in appendix 1 figure 1. The pin out for both EPROM's and SRAM's are arranged such that devices of different capacities can be used with only a minor change to the boards' circuitry.

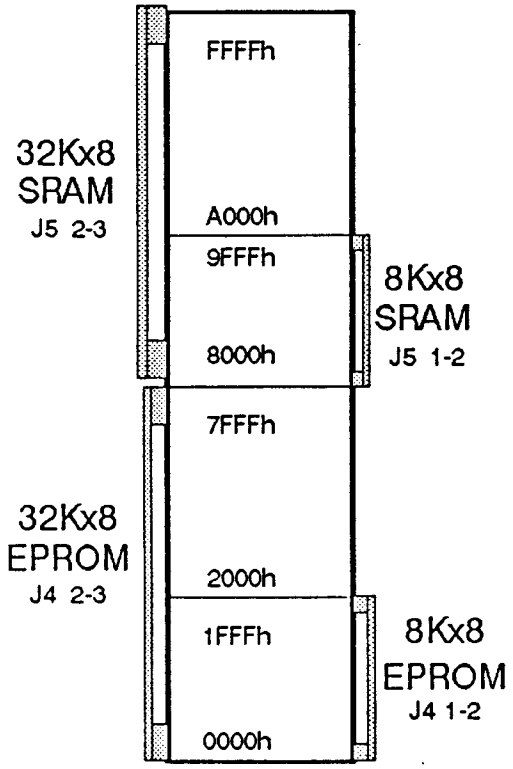
The 28 pin socket U3 can accept either an 8Kx8 or a 32Kx8 SRAM. Addithional RAM may be implemented on a target board by using the /CS1 chip select signal at P3-53 after removing the on board SRAM.

The 28 pin socket U2 can accept an 8/16/32Kx8 EPROM. Target board EPROM can be selected by /CS0 at connector P3-54 but the on board EPROM must be removed.

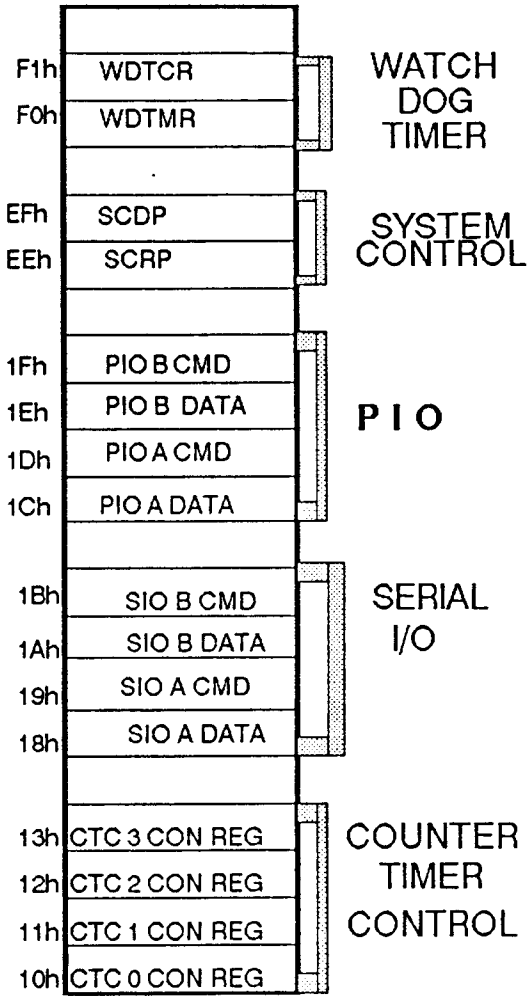
The board is shipped with the Chip Select Boundary Register (CSBR) programmed to generate /CS0 for the Eprom (0000h to 1FFFh). /CS1 will be asserted for the SRAM (2000h to 3FFFh). The debug monitor will have the Z84C15's CSBR programmed to 31h to obtain this particular memory mapping.

Appendix 1 figure 1 depicts a memory mapping scheme with the Z84C15's CSBR programmed to F7h (7FFFh >= /CS0 >= 0000h; FFFFh >= /CS1 >= 8000h). This figure also shows the I/O space for the Z84C15.

APPENDIX 1 FIGURE 1
ADDRESS MAP



MEMORY MAP FOR CSBR EQUAL TO F7H



I/O ADDR MAP

APPENDIX 2: ZASM UTILITIES

The ASM800 cross assembler takes a source file containing assembly language statements and translates it into a corresponding object file. A simple example of program assembly is shown below.

It assumed that the contents of ZASM have been downloaded.

```
C:\ZILOG\SRC>asm800 user15.s -l -r -o user15.o -s asm800
```

where:	asm800	is the reserved word to invoke assembly.
	user15.s	is your Z80 source code
	-l	is an option that will create a listing file user15.l
	-r	restricts all instructions to a Z80 subset
	-o user15.o	specifies object file name other than m.out
	-s asm800	gets the assembler symbol initialization table from asm800. Note that asm800 symfile resides in C:\ZILOG\ZASM, if so this path name must be included as \ZILOG\ZASM\asm800.

Complete documentation of the asm800 utilities is contained in "asm800 Z800 Cross Assembler User's Guide" contained with the Evaluation Kit.

APPENDIX 3: MUFOM UTILITIES

MLINK

MUFOM is an acronym for Microprocessor Universal Format for Object Modules. The utilities allow the programmer to combine, display, and load machine-language object modules.

The mlink utility is used to assign absolute addresses to relocatable sections in MUFOM input modules in order to link two or more separate object modules into one module.

user15.o

0000	user_code
	user_ram

A common problem that occurs in cross-software development is when the target system has both PROM and RAM, and it is necessary to put the Debug Monitor in EPROM and the user code in RAM. The -address option specifies the base address of the next section to be selected. The selection sections from the input list (user15.o) is placed in specific order to the output list (user15.lnk).

```
mlink user15.o -3000s user_code -s user_ram -e 3000 -o
user15.lnk
(the above is all on one line)
```

mlink user15.o	inputs the specified file
-3000s user_code	sets the location for the next section
-s user_ram	selects input section
-e 3000	specifies the entry point for the output file
-o user15.lnk	specify which output file

Thus the above mlink command simply places all the sections of user15.o to user15.lnk starting at address 3000hex.

user15.lnk

0000	
3000	user_code
	user_ram

MLOAD

The mload utility is a format conversion program which translates MUFOM files into one of three formats suitable for moving object modules from host system to target system. The three output formats are: Intel Hex, Tektronix and simplified MUFOM.

mload user15.lnk -i -o user15.hex

where:

mload user15.lnk	is a reserved word to invoke the mload utility for input user15.lnk.
-i	specifies output to be Intel Hex format.
-o user15.hex	specifies the output file name.

For more information on the MUFOM utilities please refer to "Z8 Universal Object File Utilities" enclosed with the Evaluation Kit.

APPENDIX 4: USE OF ORION

The Orion Emulator for the Z84C15 does not use the standard emulation module which is designed for the Dip packaged microprocessors. Instead, the module can be a 28 pin or a 24 pin device that plugs into the EPROM socket. Be sure to position the module to the bottom of the socket leaving pins 1, 2, 27 and 28 open when using the 24 pin device. The following connnections are made when using the 24 pin device.

IN CIRCUIT EMULATOR LEAD		P2
A12	to	A12 (P2-3)
A13	to	A13 (P2-5)
A14	to	A14 (P2-7)

The other leads come from the "48 Channel Bus State Analyzer" connection on the front of the Orion case, via a "cable E". The following are the required connections:

cable E		P3
A15	to	A15 (P2-9)
A19	to	MREQ (P2-37)
WR	to	WR (P2-38)
RD	to	RD (P2-40)
GND	to	GND (P2-24)
NMI	to	NMI (P2-33)
RES	to	RES (P2-30)
K1	to	M1 (P2-28)
K2	to	IORQ (P2-39)

To down load a HEX file to the ORION, use the Orion Command "hex load"; set the Emulation Memory to the size of your code; start the execution with the "start up" command. You can also down load a symbol file, to allow symbolic analysis and debugging. To generate this file for programs that have been linked, the commands are:

```
MNM -L -O %1.SYM %1.LNK
LOADER MUFOM %1.SYM
```

MNM is a part of the MUFOM utilities. It generates a symbol file in format that is accepted by the "Loader" program from Orion.

There is an example file called "ORIONSYM.BAT" on the SRC.EXE diskette. This will create a file "orion.sym" that can be down loaded to the Orion by using the command "symload orion".

Be sure to use the correct EPROM Personality module when attempting to program EPROMs with the Orion.

APPENDIX 5: JUMPER DEFINITION

HEADER	FUNCTION	SHIPPED	OPTION
J1	WATCH DOG TIMER OUTPUT		
	TO /RESET		1 TO 2
		OPEN	
J2	EV	OPEN	
			1 TO 2
J3	CLOCK SOURCE		
	ON CHIP GENERATOR	1 TO 2	
	EXTERNAL SOURCE		OPEN
J4	EPROM SIZE		
	8Kx8	1 TO 2	
	32K x8		2 TO 3
J5	SRAM SIZE		
	8Kx8	1 TO 2	
	32Kx8		2 TO 3
J6	SIO A TRANSMIT CLOCK (/TXCA) SOURCE		
	/PTXCA EXTERNAL SIGNAL FROM P3-27		1 TO 3
	/PTXCB EXTERNAL SIGNAL FROM P3-39		7 TO 3
	ZC/TO 0 COUNTER/TIMER 0 OUTPUT		5 TO 3
	ZC/TO 0 COUNTER/TIMER 0 OUTPUT		6 TO 3
	ZC/TO 1 COUNTER TIMER 1 OUTPUT		11 TO 3
	ZC/TO 1 COUNTER/TIMER 1 OUTPUT		2 TO 3
	SIO A RECEIVE CLOCK (/RXCA) SOURCE		
	/PRXCA EXTERNAL SIGNAL FROM P3-27		1 TO 4
	/PRXCB EXTERNAL SIGNAL FROM P3-39		7 TO 4
	ZC/TO 0 COUNTER/TIMER 0 OUTPUT		5 TO 4
	ZC/TO 0 COUNTER/TIMER 0 OUTPUT		6 TO 4
	ZC/TO 1 COUNTER/TIMER 1 OUTPUT		11 TO 4
	ZC/TO 1 COUNTER/TIMER 1 OUTPUT		12 TO 4
	SIO B TRANSMIT CLOCK (/TXCB) SOURCE		
	/PTXCA EXTERNAL SIGNAL FROM P3-27		1 TO 9
	/PTXCB EXTERNAL SIGNAL FROM P3-39		7 TO 9
	ZC/TO 0 COUNTER/TIMER 0 OUTPUT		5 TO 9
	ZC/TO 0 COUNTER/TIMER 0 OUTPUT		6 TO 9
	ZC/TO 1 COUNTER/TIMER 1 OUTPUT	11 TO 9	
	ZC/TO 1 COUNTER/TIMER 1 OUTPUT		12 TO 9
	SIO B RECIEVE CLOCK (/RXCB) SOURCE		

/PRXCA	EXTERNAL SIGNAL FROM P3-27	1 TO 10
/PRXCB	EXTERNAL SIGNAL FROM P3-39	7 TO 10
ZC/TO 0	COUNTER/TIMER 0 OUTPUT	5 TO 10
ZC/TO 0	COUNTER/TIMER 0 OUTPUT	6 TO 10
ZC/TO 1	COUNTER/TIMER 1 OUTPUT	11 TO 10
ZC/TO 1	COUNTER/TIMER 1 OUTPUT	12 TO 10

J7 TRANSMIT DATA (TD1) SOURCE

TXDA	SIO A TRANSMIT DATA		1 TO 3
TXDB	SIO B TRANSMIT DATA	5 TO 3	

RECEIVE DATA (RD1) SOURCE

RXDA	SIO A RECEIVE DATA		2 TO 4
RXDB	SIO B RECEIVE DATA	6 TO 4	

TRANSMIT DATA (TD2) SOURCE

RSTA	SIO A REQUEST TO SEND		7 TO 9
RSTB	SIO B REAUEST TO SEND	11 TO 9	

RECEIVE DATA (RD2) SOURCE

CTSA	SIO A CLEAR TO SEND		8 TO 10
CTSB	SIO B CLEAR TO SEND	12 TO 10	

APPENDIX 6: CONNECTOR DEFINITION

DESIG.	SCHEM. LOC.	FUNCTION
P1	B3	RS232 port
P2	B1	40 pin that mirrors Z80 pin out.
P3	D1	60 pin with grouped I/O signals.

APPENDIX 7: LISTINGS OF READ.ME FILES

These are the program source files and batch files for the Z84C15EV1. The following files have been compressed using PKwares' PKSFX utility.

DMC15.S	Source code for the debug monitor.
USER15.S	A short program to test the debug monitors' down load feature.
ASM15.BAT	A file to compile, link and load the debug monitor.
ORIONSYM.BAT	A file to allow the Orion emulator to do symbolic debugging.
USER15.BAT	A file to compile, link and load USER15.S.

The file SRC.EXE is a self extracting archive file that was produced by PKWAREs' PKSFX compression utility. Follow the instructions below if you wish to use an assembler\linker other than that supplied on the included ZASM and MOBJ diskettes.

To install the software you will require 103,981 bytes free on your hard disk.

Example Installation:

1. Choose which hard drive to use (e.g. C:).

2. Make a directory and subdirectory on the chosen hard drive (e.g. ZILOG\SRC).

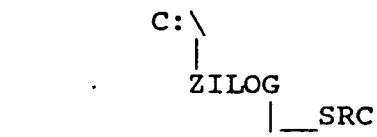
3. Add the drive and directory names to the PATH command in your AUTOEXEC.BAT file (e.g. C:\ZILOG\SRC;)

4. Insert the diskette labeled SRC.EXE into Drive "A:". Type:

   A:SRC -D C:\ZILOG\SRC

   PKSFX self extracts and installs the files from A:SRC.EXE.

Your directory structure will now be:



The SRC subdirectory contains the applications and test programs listed above.

PKSFX Quick Reference:

Syntax:
Z84C15AP [options] [d:path\] [file...]

Where:

Z84C15AP	= Name of the .EXE file
options	= Any of the options listed below.
d:path\	= Output drive and/or path.
file	= Name(s) of files to extract. Wildcards *, ?, are ok. Default is all files.

Options:

-c[m]	= Extract to screen [with More]
-d	= Create directories stored in ZIP file
-p[a,b,c]	= Extract to printer [ASC mode, Bin mode, COM port] [port number]
-x	= Extract (default)

The following is the "Read.me" file for the MOBJ and ZASM diskettes.

The file <filename>.EXE is a self extracting archive file that was produced by PKWAREs' PKSFX compression utility.

To install the software you will require 508,894 bytes free on your hard disk.

Example Installation:

1. Choose which hard drive to use (e.g. C).
2. Make a directory on the chosen hard drive (e.g. ZILOG).
3. Insert the diskette labeled ZASM.EXE into Drive "A:". Type:

```
A:ZASM -D C:\ZILOG\ZASM
```

PKSFX creates a target subdirectory named "ZASM" in "C:\ZILOG" and self
extracts files from A:ZASM.EXE into it.

4. Insert the diskette labled MOBJ.EXE into drive A: and type:

```
A:MOBJ -D C:\ZILOG\ZASM
```

This will put the MUFOM utilities into the same directory with the Z80, Z8, AND Z8000 assemblers.

5. Insert the diskette labeled SRC.EXE into drive A: and type:

```
A:SRC -D C:\ZILOG\SRC
```

This will create a subdirectory named SRC and load the source code into it.

Your directory structure will now be:

6. Add the drive and directory names to the path command in your AUTOEXEC.BAT file (e.g. C:\ZILOG\ZASM;)

7. Add the command SET ASM800=C:\ZILOG\ZASM\ASM800 to your autoexec.bat file.

8. Reboot your system.

The ZASM directory contains all the utilities necessary to operate the kit and more. They are:

ASM800	MLIB.EXE TERM.EXE
ASM800.EXE	MLIST.EXE
MLINK.EXE	MLORDER.EXE
MLOAD.EXE	MNM.EXE
MCONV.EXE	MDUMP.EXE
ASM80K	ASM80K.EXE
ASM8	ASM8.EXE

The SRC subdirectory contains the applications and test programs.

Please note that the compiler calls include the option "-s". This tells the compiler where to find the symbol init table generating file which is named "Asm8xx" (no extention). If you attempt to compile your own code and get this part wrong you will receive the error message "Cannot open symbol init table".

PKSFX Quick Reference:

Syntax:
<filename>AP [options] [d:path\] [file...]

Where:

<filename>AP	= Name of the .EXE file
options	= Any of the options listed below.
d:path\	= Output drive and/or path.
file	= Name(s) of files to extract. Wildcards *, ?, are ok. Default is all files. Options:
-c[m]	= Extract to screen [with More]
-d	= Create directories stored in ZIP file
-p[a,b,c]	= Extract to printer [ASC mode, Bin mode, COM port] [port number]
-x	= Extract (default)

If you would like more information on the PKWARE products contact:

PKWARE, Inc.
7545 North Port Washington Road
Glendale, WI 53217

Voice (414) 352-3670
Fax (414) 352-3815

APPENDIX 8: SELECTED SOURCE LISTINGS

The following is the text of USER15.BAT.

```
asm800 USER15.s -l -r -o USER15.o -s %asm800%
mlink USER15.o -3000s USER_CODE -s USER_RAM -e 3000 -o
USER15.lnk
mload USER15.lnk -i -o USER15.hex
```

The following is the text of ASM.BAT.

```
Asm800 dmc15.s -l -r -o dmc15.OBJ -s %asm800%
mlink dmc15.OBJ -0S DM_CODE -o dmc15.lnk -2000S DM_RAM -o
mload dmc15.lnk -i -o dmc15.hex
```

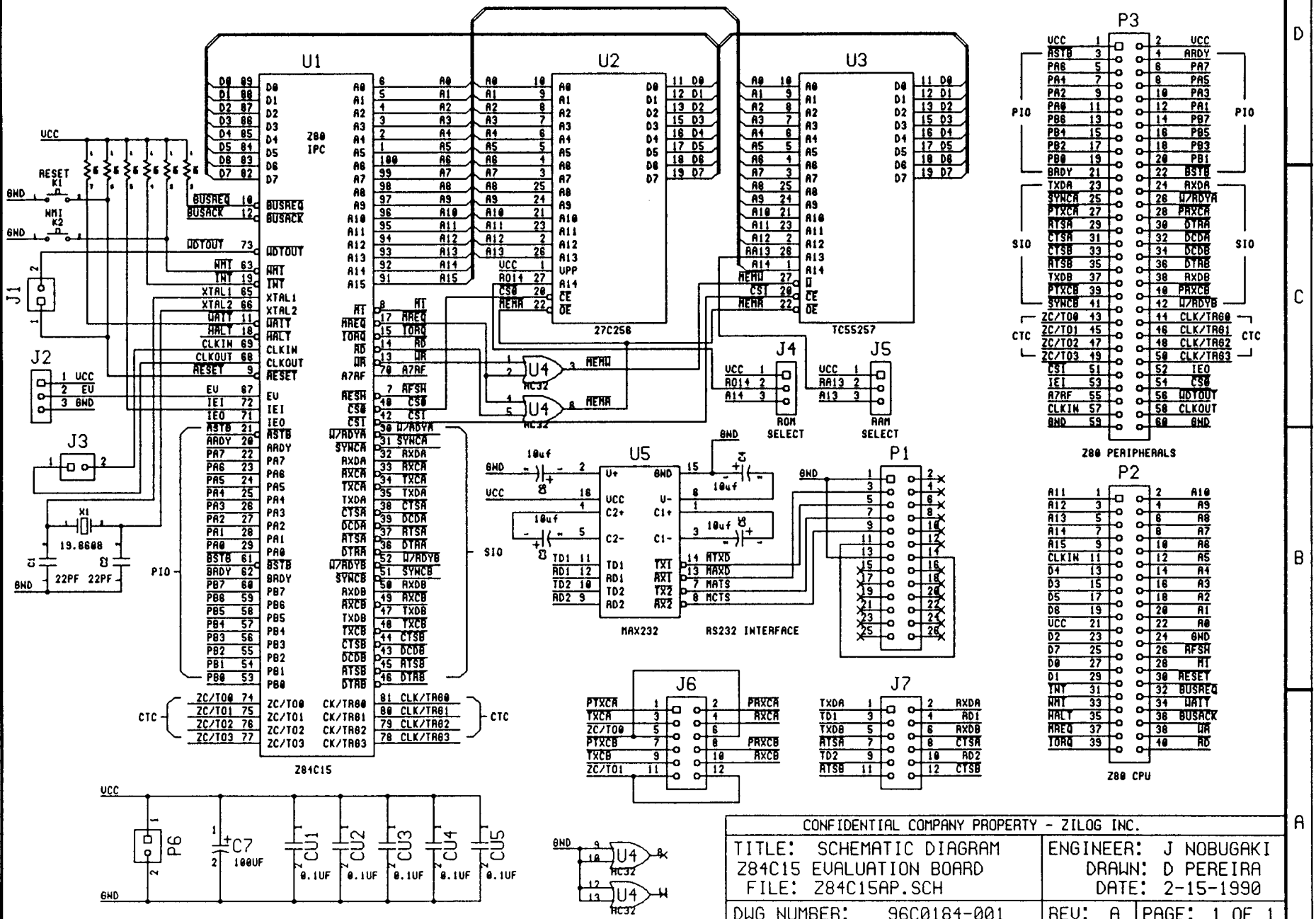
The following is the text of USER15.S.

```
;*************************************************************
*****
;
; Filename : USER15.S
; Date     : 1-12-1990
; Updated  : 2-28-1990
;
; SIMPLE MAIN PROGRAM TO TEST DOWNLOAD COMMAND OF Z84C15
DEBUG MONITOR.
;
; - to be downloaded in RAM at location 3000h
;*************************************************************
*****

; KIO REG. ADDR. :
KIO  EQU  00H  ;BASE ADDR. OF KIO
SIO_BD    EQU  KIO+1AH   ;ADDR. OF SIO B - DATA
SIO_BC    EQU  KIO+1BH   ;ADDR. OF SIO B - COMMAND

RXRDY     EQU  0    ;BIT 0 OF COMMAND REG.
TXRDY     EQU  2    ;BIT 2 OF COMMAND REG.
XON  EQU  11H  ;RESUME TRANSMISSION
XOFF EQU  13H  ;STOP TRANSMISSION

     SECTION USER_CODE

BEGIN:
     LD B,30h
LOOP:     DEC B          ; LOOP ON B REGISTER CONTENT
     LD A,B         ; USE BREAKPOINTS TO SEE EVOLUTION
     CP 0
```

```
      JP NZ,LOOP
      LD HL,MSG ; DISPLAY MESSAGE ON SCREEN
      CALL PUTCHAR
CONT:      LD HL,MEM ; LOADS HL WITH ADDRESS OF MEMORY
LOCATION
      LD (MEM),HL     ; STORES THAT ADDRESS IN MEMORY TOO
      RST 30H         ; SOFTWARE BREAKPOINT - DISPLAYS
REGISTERS
                ; CHECK REGISTER CONTENTS (A,B,HL) AND
                ; MEMORY CONTENTS (D 3300,2)

; SUBROUTINE PUTCHAR
; The SIO is well initialized in the Debug Monitor
PUTCHAR:PUSH BC           ;***** SAVE BC
      PUSH AF         ;&&&&& SAVE AF
      LD B,(HL) ;LOAD COUNT VALUE TO B
NEXT:      INC HL
      LD A,B          ;FINISH ?

      CP 0       ;
      JP Z,OUT   ;
      IN A,(SIO_BC)   ;CHECK RX BUFFER
      BIT RXRDY,A     ;
      JR Z,CHK_TX     ;NOTHING IN THE RX BUFFER
      IN A,(SIO_BD)   ;SOMETHING IN THE RX BUFFER, READ IT.
      CP XOFF         ;STOP TRANSMISSION ?
      JR NZ,CHK_TX    ;
CHK_XON:IN A,(SIO_BC)     ;WAIT FOR XON
      BIT RXRDY,A     ;
      JR Z,CHK_XON    ;
      IN A,(SIO_BD)   ;
      CP XON          ;
      JR NZ,CHK_XON   ;
CHK_TX:    IN A,(SIO_BC)   ;CHECK TX
      BIT TXRDY,A     ;RX READY ?
      JP Z,CHK_TX     ;
      LD A,(HL)
      OUT (SIO_BD),A ;TRANSMIT CHAR.
      DEC B
      JP NEXT
OUT:  POP AF          ;&&&&& SAVE AF
      POP BC          ;***** SAVE BC
      RET

MSG:  DEFB 22
      DEFM '******* ZILOG ******'
      DEFB 0DH
      DEFB 0AH

      SECTION USER_RAM
MEM:  DEFW 0aah

      END
```

```
asm800 version 2.4
Thu Jul 12 22:46:58 1990       USER15
LOC      OBJ             LINE# --- SOURCE ---
                             1 ;*****************************************************************************
                             2 ;
                             3 ; Filename : USER15.S
                             4 ; Date     : 1-12-1990
                             5 ; Updated  : 2-28-1990
                             6 ;
                             7 ; SIMPLE MAIN PRCGRAM TO TEST DOWNLOAD COMMAND OF Z84C15 DEBUG MONITOR.
                             8 ;
                             9 ; - to be downloaded in RAM at location 3000h (above stand-alone DMC50)
                            10 ;
                            11 ;*****************************************************************************
                            12
                            13 ; KIO REG. ADDR. :
     0000000000000000000    14 KIO     EQU     00H     ;BASE ADDR. OF KIO
     0000000000000000001a   15 SIO_BD  EQU     KIO+1AH ;ADDR. OF SIO B - DATA
     0000000000000000001b   16 SIO_BC  EQU     KIO+1BH ;ADDR. OF SIO B - COMMAND
                            17
     0000000000000000000    18 RXRDY   EQU     0       ;BIT 0 OF COMMAND REG.
     0000000000000000002    19 TXRDY   EQU     2       ;BIT 2 OF COMMAND REG.
     0000000000000000011    20 XON     EQU     11H     ;RESUME TRANSMISSION
     0000000000000000013    21 XOFF    EQU     13H     ;STOP TRANSMISSION
                            22
                            23         SECTION USER_CODE
                            24
00000000                    25 BEGIN:
00000000 0630               26         LD B,30h
00000002 05                 27 LOOP:   DEC B           ; LOOP ON B REGISTER CONTENT
00000003 78                 28         LD A,B          ; USE BREAKPOINTS TO SEE EVOLUTION
00000004 fe00               29         CP 0
00000006 c2R000+0002,       30         JP NZ,LOOP
00000009 21Wwww             31         LD HL,MSG       ; DISPLAY MESSAGE ON SCREEN
0000000c cdWwww             32         CALL PUTCHAR
0000000f 21Wwww             33 CONT:   LD HL,MEM       ; LOADS HL WITH ADDRESS OF MEMORY LOCATION
00000012 22Wwww             34         LD (MEM),HL     ; STORES THAT ADDRESS IN MEMORY TOO
00000015 f7                 35         RST 30H         ; SOFTWARE BREAKPOINT - DISPLAYS REGISTERS
                            36                         ; CHECK REGISTER CONTENTS (A,B,HL) AND
                            37                         ; MEMORY CONTENTS (D 3300,2)
                            38
                            39 ; SUBROUTINE PUTCHAR
                            40 ; The SIO is well initialized in the Debug Monitor
00000016 c5                 41 PUTCHAR:PUSH BC         ;***** SAVE BC
00000017 f5                 42         PUSH AF         ;&&&&& SAVE AF
00000018 46                 43         LD B,(HL)       ;LOAD COUNT VALUE TO B
00000019 23                 44 NEXT:   INC HL
0000001a 78                 45         LD A,B          ;FINISH ?
0000001b fe00               46         CP 0            ;
0000001d caWwww             47         JP Z,OUT        ;
00000020 db1b               48         IN A,(SIO_BC)   ;CHECK RX BUFFER
00000022 cb47               49         BIT RXRDY,A     ;
00000024 28**               50         JR Z,CHK_TX     ;NOTHING IN THE RX BUFFER
00000026 db1a               51         IN A,(SIO_BD)   ;SOMETHING IN THE RX BUFFER, READ IT.
00000028 fe13               52         CP XOFF         ;STOP TRANSMISSION ?
0000002a 20**               53         JR NZ,CHK_TX    ;
0000002c db1b               54 CHK_XON:IN A,(SIO_BC)   ;WAIT FOR XON
0000002e cb47               55         BIT RXRDY,A     ;
00000030 28fa               56         JR Z,CHK_XON    ;
00000032 db1a               57         IN A,(SIO_BD)   ;
00000034 fe11               58         CP XON          ;
00000036 20f4               59         JR NZ,CHK_XON   ;
00000038 db1b               60 CHK_TX: IN A,(SIO_BC)   ;CHECK TX
0000003a cb57               61         BIT TXRDY,A     ;RX READY ?
0000003c caR000+0038,       62         JP Z,CHK_TX     ;
0000003f 7e                 63         LD A,(HL)
00000040 d31a               64         OUT (SIO_BD),A  ;TRANSMIT CHAR.
00000042 05                 65         DEC B
00000043 c3R000+0019,       66         JP NEXT
00000046 f1                 67 OUT:    POP AF          ;&&&&& SAVE AF
00000047 c1                 68         POP BC          ;***** SAVE BC
00000048 c9                 69         RET
                            70
00000049 16                 71 MSG:    DEFB 22
0000004a 2a2a2a2a2a2a2a20   72         DEFM '******* ZILOG *******'
00000052 5a494c4f47202a2a
0000005a 2a2a2a2a
0000005e 0d                 73         DEFB 0DH
0000005f 0a                 74         DEFB 0AH
                            75
                            76         SECTION USER_RAM
00000000 aa00               77 MEM:    DEFW 0aah
```

C/N	REV	DESCRIPTION	DATE	APPROVED
	A	ORIGINAL ISSUE		



CONFIDENTIAL COMPANY PROPERTY - ZILOG INC.

TITLE: SCHEMATIC DIAGRAM Z84C15 EVALUATION BOARD FILE: Z84C15AP.SCH	ENGINEER: J NOBUGAKI DRAWN: D PEREIRA DATE: 2-15-1990
DWG NUMBER: 96C0184-001	REV: A | PAGE: 1 OF 1
DWG SIZE: A	