TBUG™
Z-80 Monitor and Debugging Aid

Catalog Number 26-2001

Radio Shack®
TRS-80
MICRO
COMPUTER
SYSTEM

SOFTWARE

Table of Contents

CUSTOM MANUFACTURED IN THE USA FOR RADIO SHACK A DIVISION OF TANDY CORPORATION

Introduction

TBUG is a powerful, machine-language monitor designed to give you direct access to the Z-80 CPU, which is the heart of the Radio Shack TRS-80 Microcomputer. TBUG provides the capability to:

1. Create and modify machine-language programs.
2. Debug machine-language programs through the use of breakpoints and register displays.
3. Examine and modify the contents of RAM memory and Z-80 registers.
4. Save and load machine-language programs using cassette tape.
5. Execute machine-language programs created using TBUG (if you have a LEVEL II TRS-80, you can also execute programs created with the Editor/Assembler program via TBUG).

This manual will tell you how to use TBUG. There is also considerable summary information on the Z-80 CPU Instruction Set. However, this manual does not pretend to teach you to write machine language programs. To do that would require at least one full length book. Several such books are listed in Appendix G.

How this manual is organized

TBUG is available for both LEVEL I and LEVEL II machines. Because of differences between LEVEL I and LEVEL 11 memory allocation and ROM routines, there are a few differences in the way TBUG loads and operates in the two machines.

That's why there are two sections titled, Loading and Using TBUG one for each type of machine. A few other sections are broken up this way. Read the sections that pertain to your TRS-80.

Notation Conventions

Before going on, you should be aware of the following notational conventions, used for the sake of clarity and brevity throughout this manual.

ENTER	Represents the phrase, "Press the ENTER key".
CAPITALS and punctuation	Indicate material which must be entered exactly as it appears. Example: B nnnn
INVERSE CAPITALS	Only the command B is entered verbatim; you supply a value for nnnn.
	Represent input you supply, upon prompting from the Computer. This convention will only be used where necessary to distinguish between Computer prompting and your input. Example: # M 2C00 20
	The Computer supplies the # character, 20, and all the blanks; you supply the characters shown with gray background.
lowercase italics	Represent words, letters or values you supply from a set of acceptable values for that situation. Example: P *aaaa bbbb cccc name*
	aaaa,bbbb,cccc are addresses you supply, and *name* is a name you supply.

nnnnH	The letter H following a number indicates the number is in hexadecimal form. See the decimal/hexadecimal conversion chart, Appendix F. Examples: 3C00H is the hexadecimal number equivalent to decimal number 15360 0AH is the hex number equivalent to decimal 10
ddddd	Numbers appearing without the H suffix are in decimal form, unless described otherwise.
lsb	Indicates the least significant byte in a register pair or address specification. Example: IX (lsb) refers to the least significant byte of the IX register
msb	Indicates the most significant byte in a register pair or address specification. Example: IX (msb) refers to the most significant byte of the IX register

Before Loading TBUG...

The TBUG monitor program uses only the first 16 columns of the screen for displaying Computer output and your input. The other 48 columns will be unaffected by TBUG (exception: LEVEL II Punch command).

For the sake of a few of the examples in this manual, you should clear the display (CLS statement or CLEAR key) before loading TBUG.

One final note: TBUG provides a "live keyboard" - you don't have to press ENTER after each command.

Loading and Using TBUG
(LEVEL I only)

If your TRS-80 is equipped with LEVEL I BASIC, simply load the LEVEL I TBUG tape as you would a BASIC Program tape, using the CLOAD command. Asterisks will blink on and off as the tape loads. When the program is loaded, control is transferred to the TBUG monitor, and a number sign # appears in the top left of the Display. This sign takes the place of BASIC's > prompt.

Note: If there is an error during loading, the message

```
WHAT?
>_
```

will be displayed.
Rewind the TBUG tape and start over. You may need to try a slightly higher or lower volume setting on your recorder.

To exit TBUG and return to LEVEL I BASIC: The simplest way is to force the Computer into its power-up sequence by typing:

```
# J 0000
```

Remember, you only type in the characters which are highlighted by the gray background.

Or you can manually reset the Computer by pressing the Reset button on the left rear of the case.

(The J command is explained under TBUG Commands.)

To return to TBUG from LEVEL I BASIC, you must reload the TBUG tape.

Using TBUG in LEVEL I

On a LEVEL I TRS-80, TBUG is loaded into memory locations 4000H through 43FFH. Thus, your machine-language programs may be written using address 4400H through the end of memory (4FFFH for 4K, 7FFFH for 16K). The Stack Pointer is preset to 43FDH, although you may easily change this to another address in memory (above 4400H), if desired.

One of the features of TBUG, which makes it such a powerful and versatile monitor, is the Register Save Area. This area is used to preserve the Z-80 registers while TBUG is in control. This assures that the register contents will not be disturbed by TBUG.

The TBUG commands, which access the Register Save Area, are: Breakpoint, Register, Jump, and Go. When TBUG is entered from a user program via the Breakpoint command, the Z-80 registers are placed in the Register Save Area. The Register command will display the contents of these registers by calling out the Register Save Area addresses. Prior to execution of the user program (via Jump and Go commands), the Z-80's registers are loaded from the Register Save Area. See the command descriptions for further details.

On the LEVEL I TRS-80, the Register Save Area is located in memory addresses 43B7H through 43CEH. Table 1 indicates where each of the Z-80's registers is stored. **Note:** you may set the initial values of the CPU registers by using the Memory command to modify the addresses in the Register Save Area prior to executing a Jump or Go command.

Register	Location	Register	Location	Register	Location
A'	43B8H	A	43C0H	IX(MSB)	43C8H
F'	43B7H	F	43BFH	IX(LSB)	43C7H
B'	43BAH	B	43C2H	IY(MSB)	43CAH
C'	43B9H	C	43C1H	IY(LSB)	43C9H
D'	43BCH	D	43C4H	SP(MSB)	43CCH
E'	43BBH	E	43C3H	SP(LSB)	43CBH
H'	43BEH	H	43C6H	PC(MSB)	43CEH
L'	43BDH	L	43C5H	PC(LSB)	43CDH

Table 1. Register Save Area for LEVEL I

Loading and Using TBUG
(LEVEL II only)

If your TRS-80 is equipped with LEVEL II BASIC, you load TBUG under the SYSTEM command. Insert the tape into the recorder and adjust the volume to a LEVEL II setting (around 4-6 on the CTR-41). Now type

```
>SYSTEM ENTER

*? TBUG ENTER
```

The tape will begin to load, as indicated by the flashing asterisks on the Display. When the tape has loaded, the SYSTEM prompt will reappear. Type:

```
*? / ENTER
```

A number sign # will appear at the top left of the Display. This is the TBUG prompt - comparable to the BASIC prompt > .

Note: If there is an error during the TBUG loading process, a C will appear beside the asterisk in the upper right of the Display. In this case, rewind the tape and start over. You may need to use a slightly higher or lower volume setting.

To exit TBUG and return to LEVEL II BASIC:

The simplest way is to force the Computer into its power-up sequence by typing:

```
# J 0000
```

Remember, you only type the characters highlighted by the gray background.

The J command is explained under TBUG Commands.

Or, you can manually reset the Computer by pressing the Reset button on the left rear of the keyboard case.

Note: If your machine language program has altered the contents of locations 4000H through 42E8H, pressing the Reset button may not have the desired effect of resetting the Computer and returning you to BASIC. In such cases, you'll have to turn the Computer off and on again.

To re-enter TBUG from LEVEL II BASIC:

Assuming you have not altered the address area where TBUG resides, you can return to TBUG from BASIC by typing:

```
>SYSTEM ENTER

*? /17312 ENTER
```

Notice that this address (decimal 17312) is too low to be protected via the MEMORY SIZE specification. Therefore, BASIC program execution will typically alter the TBUG resident area, and you'll have to reload TBUG if you do any BASIC program execution.

Using TBUG in LEVEL II

On a LEVEL II TRS-80, TBUG is loaded into memory locations 4380H through 497FH. Thus, your machine language programs may be written using address 4980H through the end of memory (4FFFH for 4K machines, 7FFFH for 16K, BFFFH for 32K, FFFFH for 48K). The Stack Pointer is preset to 4980H, although you may easily change this to another address in memory (above 4980H), if desired.

One of the features of TBUG, which makes it such a powerful and versatile monitor, is the Register Save Area. This area is used to preserve the Z-80 registers while TBUG is in control. This assures that the register contents will not be disturbed by TBUG.

The TBUG commands, which access the Register Save Area, are: Breakpoint, Register, Jump, and Go. When TBUG is entered from a user program via the Breakpoint command, the Z-80 registers are placed in the Register Save Area. The Register command will display the contents of these registers by calling out the Register Save Area addresses. Prior to execution of the user program (via Jump and Go commands), the Z-80's registers are loaded from the Register Save Area. See the command descriptions for further details.

On the LEVEL II TRS-80, the Register Save Area is located in memory addresses 4825H through 483CH. Table 2 indicates where each of the Z-80's registers is stored. NOTE: You may set the initial values of the CPU registers by using the Memory command to modify the addresses in the Register Save Area prior to executing a Jump or Go command.

Register	Location	Register	Location	Register	Location
A'	4826H	A	482EH	IX(MSB)	4836H
F'	4825H	F	482DH	IX(LSB)	4835H
B'	4828H	B	4830H	IY(MSB)	4838H
C'	4827H	C	482FH	IY(LSB)	4837H
D'	482AH	D	4832H	SP(MSB)	483AH
E'	4829H	E	4831H	SP(LSB)	4839H
H'	482CH	H	4834H	PC(MSB)	483CH
L'	482BH	L	4833H	PC(LSB)	483BH

Table 2. Register Save Area for LEVEL II

TBUG Commands

In following the sample displays below, remember that keyboard input (what you type in) is highlighted with a gray background. Computer prompts and output appear without any highlighting. Note that you can only enter a command when the TBUG # prompt appears alone as the last line displayed on the screen.

Memory

The Memory command lets you examine the contents of any memory location, either ROM or RAM. It also lets you modify the contents of any RAM location. To initiate this command, type the letter M followed by the address you want to change:

Mnnnn

where nnnn is a 4-digit hexadecimal address.

(All numbers input to or output from TBUG are in hexadecimal.)

For example, suppose you want to examine the contents of memory location 3C20H. You simply type M3C20 as follows:

```
# MW3C20 20
```

Notice that TBUG displays the contents of 3C20H immediately after you type the fourth digit of the address; also that TBUG inserts spaces in the line for easy reading.

Now that you have initiated the Memory command, you can either leave the current address unchanged and look at the next higher address in memory, or change the contents of the current address and then advance to the next one.

To change the contents of the displayed location, type the new 2-digit value (hex form) after the displayed value. For example, to change the contents of 3C20H from 20H to 4CH, type:

```
# M3C20 20 4C
3C21 20
```

If you tried the above example, you noticed two things:

1. TBUG automatically displays the next address in memory along with its contents;
2. the letter L appears toward the middle of the top line of the Display.

To understand why the L appears, you must realize that addresses 3C00 through 3FFF are memory-mapped to the Display. In other words, whatever values are stored in these locations will automatically be displayed on the screen in ASCII form.

Since 4CH is the ASCII code for the letter L, that letter appears in the Display location which corresponds to address 3C20H.

After you type in the new 2-digit value for an address, TBUG modifies the address accordingly and displays the next address. To leave an address unchanged, simply press ENTER and TBUG will move on to the next higher address.

For example:

```
# M 3C20 20 4C
3C21 26
3C22 20
3C23 20 X
#
```

To exit from the Memory command at any time, type an X. This character will not be displayed, because it has a control function, causing TBUG to return with the prompt. You can then enter any TBUG command.

Jump

The Jump command lets you start execution of a machine-language program at a specified address. Its form is:

Jnnnn

where nnnn is a 4-digit hexadecimal address where execution is to begin.

When you type in the Jump command, TBUG does the following:

1. Loads nnnn into the Program Counter (PC) register;
2. Loads the other Z80 registers from the Register Save Area;
3. Begins execution.

For example, to execute a program beginning at 4A00H, type:

```
# J 4A00
```

You should take care that the address specified contains an executable instruction. That is, it should be a one-byte instruction or the first byte of a multiple-byte instruction.

You may cancel the Jump command any time prior to entering the fourth character of the address, by typing the letter X. This will return control to TBUG, and you can enter another command.

Breakpoint

This command lets you break the normal flow of a program at a pre-specified "point" or instruction. It is a very powerful tool and may be used to debug or simply terminate a program.

To initiate the Breakpoint command, type B followed by the hex address where you wish the break in execution to occur:

Bnnnn

where nnnn is the specified break address. This address should contain a single-byte instruction or the first byte of a multiple-byte instruction.

The effect of Bnnnn is to replace the contents of nnnn through nnnn+2 with a breakpoint sequence. The previous contents of these locations are saved and can be restored later with the Fix command (see below).

Note: You can cancel the Breakpoint command prior to typing the fourth digit of the address, by typing an X.

When program execution reaches the breakpoint you have entered, control will be transferred to TBUG. The Z-80's register contents will be placed in the Register Save Area, where you may examine or modify them (see Register command).

Don't try to place more than one breakpoint in a program at once, since only the last entered breakpoint can be restored to its former contents via the Fix command.

Example:

```
# M 5200 C3
5201 A0 ENTER
5202 52 X
# B 5200
# M 5200 CD
```

In this example, addresses 5200H through 5202H contain a Z-80 instruction (JUMP to 52AOH). A breakpoint is inserted at 5200H. If you examine 5200H, you see that it now contains CDH, which is the first byte of the breakpoint sequence.

When you attempt to execute beginning at 5200H, the breakpoint is encountered immediately, so that TBUG regains control and returns with the # prompt. To demonstrate this, type:

```
# J 5210
```

At this point, you can examine registers and memory, and type in any TBUG command. However, if you plan to restore the breakpoint's previous contents with Fix, you should do it now, or before the user PC contents are modified.

```
# F
# M 5200 C3 ENTER
5201 A0 ENTER
5202 52
```

Notice that 5200H through 5202H have been restored with their previous contents. You can now continue execution with a Go or Jump command.

Fix

Fix is used in conjunction with the Breakpoint command. After a breakpoint has been reached during execution of a machine language program, and control returns to TBUG, type F. This tells TBUG to retrieve the previous contents of the breakpoint, and place it in the address specified by the user PC. See Breakpoint for examples.

Use the F command after executing a breakpoint and before changing the user PC contents. **If you use it any other time, it may destroy part of your program.**

Go

The Go command tells TBUG to begin execution at the address specified by the user PC. You should only use it when you know what's in the user PC register, or immediately upon returning from and Fixing a breakpoint. Otherwise, Go may send the Computer on a "voyage into nowhere".

To use the Go command, simply type the letter G. TBUG will immediately load the Z-80 registers from the Register Save Area and begin execution.

Register

The register command lets you examine the contents of the Register Save Area. Simply type the letter R, and TBUG will display the Register Save Area as follows:

A'F'	B'C'
D'E'	H 'L'
AF	BC
DE	HL
IX(msb,lsb)	IY(msb,lsb)
SP(msb,lsb)	PC(msb,lsb)

To modify the contents of any register, place the desired value in the appropriate Register Save Area address. The next time you execute a Jump or Go command, the Z-80 registers will be loaded from the Save Area, and any changes you've made will be implemented then.

Punch

The Punch command lets you dump the contents of memory onto tape. This is the only command which must be specified differently for LEVEL I and LEVEL II machines. Program tapes created with Punch can be loaded just as TBUG is loaded.

Note: The cassette recorder should be in the Record mode prior to executing the Punch command.

Punch
LEVEL I only

In LEVEL I TBUG, the Punch command has the following form:

P aaaa bbbb

where aaaa	is the hexadecimal starting address of the memory block you are dumping to tape;
bbbb	is the ending address of the memory block.

For example, suppose your program resides from 5000H through 521 FH. You may save this program on tape by typing the following (be sure the recorder is in the Record mode):

 # P 5000 521F

The TBUG monitor will turn on the recorder, write the contents of the specified memory block onto tape, turn off the recorder and display the prompt character on the next Display line.

The data from memory is put onto tape in the following format:

4. A 128-byte leader consisting of zeroes.
5. The synchronization code (A5H).
6. The starting address for the data (lsb, msb).
7. The ending address for the data plus 1 to include checksum (lsb, msb).
8. The data.
9. A one-byte checksum, which is the sum of all the data bytes, in two's complement form.

You may cancel the Punch command prior to typing the last character of the ending address, by typing the letter X.

Punch
LEVEL II only

In LEVEL II TBUG, the Punch command has the following form:

P aaaa bbbb cccc name

where aaaa	is the hex starting address of the block to be dumped;
bbbb	is the ending address of the block;
cccc	is the entry point - this is where execution is to begin after the tape is loaded;
name	is the file name for the program you are saving on tape. (More below.)

Tapes created by the LEVEL II TBUG Punch command can be loaded under the SYSTEM command – just as TBUG itself is loaded. All SYSTEM tapes have a file name, and that's why the Punch command requires that you specify one.

This file name can be any sequence of characters – from one to six total characters. If you wish to use a name shorter than six characters, you must press the ENTER key to start the Punch process. If you enter a six-character file name, the Punch process starts automatically. Use a file name that's easy to remember and type in.

For example, suppose your program resides from 5000H to 521FH, and you want to save the program on tape under the name TEST. Then type:

 # P 5000 521F 5000 TEST ENTER

if you wanted to save the same program under the name TESTOI, you'd type:

 # P 5000 521F 5000 TEST01

In the first case, the Punch command begins immediately after the ENTER key is pressed. In the second case, as soon as you type the sixth character of the file name, the Punch process begins. Be sure the recorder is in the Record mode before you type in this command.

You can cancel the Punch command while you are entering the start, end or entry address, by typing the letter X. If, however, you are typing in the file name, you must hit the BREAK key, since X is a valid character for a file name.

To execute Punch, TBUG turns on the recorder, writes the specified memory contents onto tape, turns off the recorder and returns with the prompt character. The format of the output tape is:

1. A 255-byte leader consisting of zeroes.
2. The synchronization code (A5H).
3. The file name header code (55H).
4. The six-byte file name padded with trailing blanks (20H).
5. One or more data blocks as followed:
    a. The data header code (3CH).
    b. The number of bytes in this block (from 1 to 256, where 256 is represented by 00H).
    c. The start address for this block (lsb, msb).
    d. The data (1 to 256 bytes as specified in 5b.
    e. A one-byte checksum, which is the sum of the start address and the data.
6. The entry point header code (78H).
7. The entry point address (lsb, msb).

There is no limit as to how many data blocks (item 5 above) are written; TBUG will figure how many are required based on the starting and ending addresses you supply.

Loading stops when the entry point address is read.

Note: that this tape format may only be read by the TBUG Load command (described below) or the LEVEL II BASIC SYSTEM command.

Load

The Load command is used to transfer the contents of a previously recorded machine-language tape into the TRS-80's memory. This command is specified by typing the letter L. TBUG will execute the command immediately.

During execution, TBUG will not accept any keyboard input. Therefore the only way to cancel the Load command is to press the reset button on the left rear of the TRS-80 keyboard case.

During loading, an asterisk will appear on the Display and blink as data is input to the Computer. This asterisk will be in the upper left of the Display for LEVEL I and in the lower left for LEVEL II. The prompt character will appear on the next Display line following completion of the load function.

If a checksum error occurs during loading, TBUG will display the letter E on the Display line next to the Load command. In this case, try to load the tape again, perhaps using a slightly higher or lower volume setting.

Conversion Considerations

If you currently own a LEVEL I TRS-80 but plan to eventually upgrade to a LEVEL II, you will probably want to convert several of your TBUG created programs to LEVEL II as well. You should be aware that there are several important differences between LEVEL I and LEVEL II which will require you to modify your LEVEL I machine–language programs so that they will run on LEVEL II. This section will provide you with information and hints, to make the conversion much simpler.

There are three subjects which require attention when converting from LEVEL I to LEVEL II:

1. Program location.
2. CALL statements to ROM subroutines.
3. Cassette tape format and transfer rate.

The first item refers to the fact that LEVEL I TBUG is located from address 4000H through 43FFh, while LEVEL II is located from 4380H through 497FH. Therefore, if you have a LEVEL I machine–language program which begins below 4980H, you will have to "relocate" it under LEVEL II. This is somewhat cumbersome since all CALL and JP statements (along with some others) will have to be changed. The easiest way to avoid this problem is to create your LEVEL I programs above 4980H to begin with.

The second item refers to the problem of changing CALL statements that utilize ROM subroutines. In the Appendix of this manual, you will find instructions on how to use certain ROM subroutines. You will also note that the LEVEL I CALL routines are somewhat different from the LEVEL II CALLS. As an example, let's say you want to use the keyboard input routine in ROM to input a byte from the keyboard. In LEVEL I, you would code the following:

```
4800 CD400B  00110 AGN     CALL  0640H        ; SCAN KEYBOARD
4803 28FB    00120         JR    Z,AGN        ; GO AGAIN IF KB CLEAR
```

However, in LEVEL II, the same coding would look like this:

```
4A00 D5      00110         PUSH  DE           ; MIUIST SAVE DE
4A01 FDE5    00120         PUSH  IY           ; AND IY
4A03 CD2B00  00130 AGN     CALL  2BH          ;SCAN ROUTINE
4A06 B7      00140         OR    A            ; A=0 I F KB CLEAR
4A07 28F7    00150         JR    Z,AGN        ;BRANCH IF NO BYTE
4A09 FDE1    00160         POP   IY           ;RESTORE IY
4A0B D1      00170         POP   DE           ;AND DE
```

As you can see, there is a substantial difference in the number of instructions and the instructions themselves.

The easiest way to convert this type of program from LEVEL I to LEVEL II is to prepare your LEVEL I program with the conversion in mind. That is, while you are creating your LEVEL I TBUG programs and you desire to use a LEVEL I ROM CALL to perform some function, look at both the LEVEL I and LEVEL II subroutine calls to see how they differ. If LEVEL II requires more bytes than LEVEL I, you should "pad" your LEVEL I routine with NOP (00H) instructions until it is the same size. In the above example, addresses 4A03H through 4A0AH of your LEVEL I routine would contain NOP instructions - the next actual instruction in the routine would be at location 4A0BH. Alternatively, if you want to reduce the conversion work even more, you could code your LEVEL I routine as follows:

```
4A00 D5      00110          PUSH  DE          ;REQ'D FOR LEVEL II
4A01 FDE5    00120          PUSH  IY          ;  TO SAVE DE & IY
4A03 CD4008  00130 AGN      CALL  0B40H       ;CHANGE CALL FOR LEVEL II
4A06 00      00140          NOP               ;USE 'OR A' FOR LEVEL II
4A07 28F7    00150          JR    Z,AGN       ;DO AGAIN IF KB CLEAR
4A09 FDE1    00160          POP   IY          ;RESTORE IY
4A0B Dl      00170          POP   DE          ;  AND DE
```

Note that only source lines 130 and 140 need be changed to adapt the routine to LEVEL II. Of course, the LEVEL I routine automatically displays the byte; the LEVEL II does not. If needed, the display byte routine will have to be added.

The last conversion item comes from the fact that you cannot load LEVEL I tapes on a LEVEL II machine. The obvious solution to this problem is to re-enter your machine-level programs using the Memory command under LEVEL II. This assumes that you have accurate hard-copy versions of these programs from which to copy.

Fortunately, the obvious solution has an alternative – you can create a machine-language program under LEVEL II which can allow you to read a LEVEL I tape! The routine which performs this function is shown in Listing 1. You may enter this program using TBUG or the TRS-80 EDITOR/ASSEMBLER. Load the program into memory along with TBUG. Using the Jump command, type

```
# J4100
```

Load your LEVEL I TBUG-created tape into the cassette. press the Play button, and hit . The familiar asterisks will appear at the upper left on the Video display as the program is loaded from tape. Upon completion, control will return to TBUG. You may now make any modifications to your LEVEL I program you desire; then, using TBUG, output it to a tape in LEVEL II format. NOTE: you will find a "relocation" function built into this program. The instruction at location 4130H:

```
4130 010000   04320              LD       BC,0
```

may be used to change the load address of your program. For example, suppose your LEVEL I program was originally located at memory locations 4400H through 4623H. If you try to load this program from tape, it will destroy part of TBUG. You may, however, change the load address to elsewhere in memory by putting in a displacement value in locations 4131H and 4132H. Your program may be loaded into 5000H through 5223H, for example, by changing the instruction at 4130H to:

```
4130 01000C   04320              LD       BC,0C00H0
```

Because of the way the program is written, the value 0C00H would be added to 4400H and 4623H before the data is loaded.

This will allow you to read your LEVEL I tapes on your LEVEL II machine, and perform program relocation if necessary. However, you will still have to make the other conversions described above.

Note: Be sure you remember to turn the volume up to between 7-9 on the recorder when you load the LEVEL I program for conversion.

If an error occurs during loading of the LEVEL I tape, the leftmost asterisk will be replaced by an E, and TBUG will regain control. Reload the LEVEL I tape (try a different volume setting).

```
                04000 ; LEVEL I CASSETTE LOAD
                04005 ;
                04010 ; THIS PROGRAM LOADS A LEVEL I OBJECT FILE TAPE
                04020 ; INTO A LEVEL II TRS-80
                04025 ;
                04027 ; VOLUME ON THE RECORDER MUST BE
                04028 ; AT A LEVEL I SETTING (7-9)
                04029 ;
                04030 ; ADDRESSES 4131H (LSB) AND 4132H (MSB) MAY BE
                04040 ; CHANGED TO RELOCATE INPUT PROGRAM
                04050 ;
                04055 ; ROM AND TBUG ADDRESSES
                04065 ;
4380            04070 TBUG    EQU   4380H       ;RETURN TO TBUG ADDRESS
3C00            04080 VIDEO   EQU   3C00H       ;UPPER LEFT CORNER VIDEO
002B            04090 KEY     EQU   2BH         ;KEYBOARD SCAN ROUTINE
                04093 ;
4100            04095         ORG   4100H
4100 310041     04100 START   LD    SP,$        ;SET STACK POINTER
4103 CD2B00     04110 CHKEY   CALL  KEY         ;SEE IF THERE IS
4106 A7         04120         AND   A           ; ANYTHING FROM KEYBOARD
4107 28FA       04130         JR    Z,CHKEY     ;NO - CHKEY
4109 FE0D       04140         CP    0DH         ;YES - CARRIAGE RETURN?
410B 20F6       04150         JR    NZ,CHKEY    ;N0 - CHKEY
410D CD8F41     04160         CALL  CTON        ;YES - TURN ON CASSETTE
4110 AF         04170         XOR   A
4111 CD6741     04180 CLOAD1  CALL  GETBIT      ;HAVE WE FOUND
4114 FEA5       04190         CP    0A5H        ; THE SYNC CHARACTER?
4116 20F9       04200         JR    NZ,CLOAD1   ;NO - CLOAD1
4118 3E2A       04210         LD    A,'*'       ;TURN ON
411A 32003C     04220         LD    (VIDEO),A   ;BOTH ASTERISKS
411D 32013C     04230         LD    (VIDEO+1),A
4120 CD8741     04240         CALL  GTBYTE      ;GET
4123 57         04250         LD    D,A         ; START
4124 CD8741     04260         CALL  GTBYTE      ;  ADDRESS
4127 5F         04270         LD    E,A         ;   FOR DE
4128 CD8741     04280         CALL  GTBYTE      ;GET
412B 67         04290         LD    H,A         ; ENDING
412C CD8741     04300         CALL  GTBYTE      ;  ADDRESS
412F 6F         04310         LD    L,A         ;   FOR HL
4130 010000     04320         LD    BC,0        ;CHANGE CONSTANT
4133 09         04330         ADD   HL,BC       ; TO RELOCATE ***
4134 EB         04340         EX    DE,HL       ;  RELOCATION
4135 09         04350         ADD   HL,BC       ;   AMOUNT
4136 EB         04360         EX    DE,HL       ;    TO DE & HL
4137 0E00       04370         LD    C,0         ;ZERO OUT CHECKSUM
4139 23         04371         INC   HL          ;INCLUDE CHECKSUM
```

Listing 1. LEVEL I Loader for LEVEL II

```
            04380 ;
            04390 ; NOW WE WILL LORD EACH BYTE OFF THE TAPE FIND
            04400 ; STORE IN MEMORY AS INDICATED BY DE AND HL
            04410 ;
413A CD8741 04420 CLOAD2  CALL  GTBYTE       ;GET BYTE FROM TAPE
413D 12     04430         LD    (DE),A      ;STORE IT IN MEM
413E 13     04440         INC   DE          ; AND STEP COUNTER
413F FE0D   04450         CP    0DH         ;IS IT A CARRIAGE RETURN?
4141 200A   04460         JR    NZ,CLOAD3   ;NO - CLOAD3
4143 F5     04470         PUSH  AF          ;YES - CONTINUE
4144 3A013C 04480         LD    A,(VIDEO+1)
4147 EE0A   04490         XOR   0AH         ;BLINK ASTERISK
4149 32013C 04500         LD    (VIDEO+1),A
414C F1     04510         POP   AF
414D 81     04520 CLOAD3  ADD   A,C         ;ADD BYTE TO CHECKSUM
414E 4F     04530         LD    C,A         ; AND SAVE IT
414F 7C     04540         LD    A,H         ;SEE
4150 BA     04550         CP    D           ; IF
4151 20E7   04560         JR    NZ,CLOAD2   ;  WE
4153 7C     04570         LD    A,L         ;   ARE
4154 BA     04580         CP    E           ;    DONE
4155 20E3   04590         JR    NZ,CLOAD2   ; NO - CLOAD2
4157 CD9441 04600         CALL  CTOFF       ;YES - TURN OFF CASSETTE
415A 79     04610         LD    A,C         ;IF CHECKSUM
415B A7     04620         AND   A           ; EQUALS ZERO
415C CA8043 04630         JP    Z,TBUG      ;  RETURN TO TBUG
415F 3E45   04640         LD    A,'E'       ;NO - TURN FIRST ASTERISK
4161 32003C 04650         LD    (VIDEO),A   ; INTO AN "E"
4164 038043 04660         JP    TBUG        ;  AND GO TO TBUG
            04670 ;
            04600 ; THIS SUBROUTINE READS A SINGLE BYTE FROM THE TAPE
            04690 ; PORT AND 'ADDS' INTO THE A REGISTER
            04700 ;
4167 D9     04710 GETBIT  EXX               ;USE ALTERNATE REGISTERS
4168 08     04720         EX    AF,AF'
4169 DBFF   04730 GB1     IN    A,(255)     ;INPUT SYNC BIT
416B 17     04740         RLA               ;DO WE HAVE ONE?
416C 30FB   04750         JR    NC,GB1      ;NO - GB1
416E 067C   04760         LD    B,7CH       ;DELAY
4170 10FE   04770         DJNZ  $           ; LOOP
4172 CD9941 04780         CALL  GTSTAT      ;CLEAR INPUT LATCH
4175 06F8   04790         LD    B,0F8H      ;DELAY
4177 10FE   04800         DJNZ  $           ; LOOP
4179 DBFF   04810         IN    A,(255)     ;GET DATA BIT
417B 47     04820         LD    B,A
417C 08     04830         EX    AF,AF'      ;DATA BIT IS IN BIT 7
417D CB10   04840         RL    B           ; OF B REGISTER
417F 17     04850         RLR               ;ADD IT INTO R REGISTER
```

Listing 1, cont.

```
4180 F5      04860          PUSH  AF
4181 CD9941  64870          CALL  GTSTAT       ;CLEAR INPUT LATCH
4184 F1      04880          POP   AF
4185 D9      04896          EXX                ;REGISTERS NORMAL AGAIN
4186 C9      04988          RET                ;FINISHED
             04910 ;
             04920 ; THIS SUBROUTINE CALLS GETBIT 8 TIMES
             04930 ; TO GET A FULL BYTE
             04940 ;
4187 0608    04950 GTBYTE   LD    B,8          ;SET COUNT
4189 CD6741  04960 GB2      CALL  GETBIT       ;GET R BIT
418C 10FB    04970          DJNZ  GB2          ;LOOP 'TIL WE HAVE A BYTE
418E C9      04980          RET
             04996 ;
             05000 ; THIS SUBROUTINE TURNS ON THE CASSETTE
             05010 ;
418F 2104FF  05020 CTON     LD    HL,0FF04H    ;SET BIT 2
4192 1808    05030          JR    CSTAT
             05040 ;
             05050 ; THIS SUBROUTINE TURNS OFF THE CASSETTE
             05060 ;
4194 2100FB  05070 CTOFF    LD    HL,0FB00H    ;CLEAR BIT 2
4197 1803    05080          JR    CSTAT
             05090 ;
             05100 ; THIS SUBROUTINE CHECKS THE PORT STATUS
             05110 ;
4199 2100FF  05120 GTSTAT   LD    HL,0FF00H    ;CLEAR READ LATCH
419C 3AA741  05130 CSTAT    LD    A,(STATUS)
419F A4      05140          AND   H            ;SET AND
41A0 B5      05150          OR    L            ; CLEAR BITS
41A1 D3FF    05160          OUT   (255),A      ;OUTPUT TO PORT
41A3 32A741  05170          LD    (STATUS),A
41A6 C9      05180          RET
41A7 00      05190 STATUS   DEFB  0
4100         05200          END   START
```

Listing 1, cont.

Appendix A. Z-80 Instruction Set by Mnemonics

OPCODE	SOURCE CODE		OPCODE	SOURCE CODE		OPCODE	SOURCE CODE	
8E	ADC	A,(HL)	CB45	BIT	0,L	CB73	BIT	6,E
DD8E05	ADC	A,(IX+IND)	CB4E	BIT	1,(HL)	CB74	BIT	6,H
FD8E05	ADC	A,(IY+IND)	DDCB054E	BIT	1,(IX+IND)	CB75	BIT	6,L
8F	ADC	A,A	FDCB054E	BIT	1,(IY+IND)	CB7E	BIT	7,(HL)
88	ADC	A,B	CB4F	BIT	1,A	DDCB057E	BIT	7,(IX+IND)
89	ADC	A,C	CB48	BIT	1,B	FDCB057E	BIT	7,(IY+IND)
8A	ADC	A,D	CB49	BIT	1,C	CB7F	BIT	7,A
8B	ADC	A,E	CB4A	BIT	1,D	CB78	BIT	7,B
8C	ADC	A,H	CB4B	BIT	1,E	CB79	BIT	7,C
8D	ADC	A,L	CB4C	BIT	1,H	CB7A	BIT	7,D
CE20	ADC	A,N	CB4D	BIT	1,L	CB7B	BIT	7,E
ED4A	ADC	HL,BC	CB56	BIT	2,(HL)	CB7C	BIT	7,H
ED5A	ADC	HL,DE	DDCB0556	BIT	2,(IX+IND)	CB7D	BIT	7,L
ED6A	ADC	HL,HL	FDCB0556	BIT	2,(IY+IND)	DC8405	CALL	C,NN
ED7A	ADC	HL,SP	CB57	BIT	2,A	FC8405	CALL	M,NN
86	ADD	A,(HL)	CB50	BIT	2,B	D48405	CALL	NC,NN
DD8605	ADD	A,(IX+IND)	CB51	BIT	2,C	CD8405	CALL	NN
FD8605	ADD	A,(IY+IND)	CB52	BIT	2,D	C48405	CALL	NZ,NN
87	ADD	A,A	CB53	BIT	2,E	F48405	CALL	P,NN
80	ADD	A,B	CB54	BIT	2,H	EC8405	CALL	PE,NN
81	ADD	A,C	CB55	BIT	2,L	E48405	CALL	PO,NN
82	ADD	A,D	CB5E	BIT	3,(HL)	CC8405	CALL	Z,NN
83	ADD	A,E	DDCB055E	BIT	3,(IX+IND)	3F	CCF	
84	ADD	A,H	FDCB055E	BIT	3,(IY+IND)	BE	CP	(HL)
85	ADD	A,L	CB5F	BIT	3,A	DDBE05	CP	(IX+IND)
C620	ADD	A,N	CB58	BIT	3,B	FDBE05	CP	(IY+IND)
09	ADD	HL,BC	CB59	BIT	3,C	BB	CP	E
19	ADD	HL,DE	CB5A	BIT	3,D	B8	CP	H
29	ADD	HL,HL	CB5B	BIT	3,E	BC	CP	H
39	ADD	HL,SP	CB5C	BIT	3,H	FE20	CP	N
DD09	ADD	IX,BC	CB5D	BIT	3,L	BF	CPA	
DD19	ADD	IX,DE	CB66	BIT	4,(HL)	B9	CPC	
DD29	ADD	IX,IX	DDCB0566	BIT	4,(IX+1ND)	BA	CPD	
DD39	ADD	IX,SP	FDCB0566	BIT	4,(IY+IND)	EDA9	CPD	
FD09	ADD	IY,BC	CB67	BIT	4,A	EDB9	CPDR	
FD19	ADD	IY,DE	CB61	BIT	4,C	EDA1	CPI	
FD29	ADD	IY,IY	CB62	BIT	4,D	EDB1	CPIR	
FD39	ADD	IY,SP	CB63	BIT	4,E	2F	CPL	
A6	AND	(HL)	CB60	BIT	4,H	BD	CPL	
DDA605	AND	(IX+IND)	CB64	BIT	4,H	27	DAA	
FDA605	AND	(IY+IND)	CB65	BIT	4,L	35	DEC	(HL)
A7	AND	A	CB6E	BIT	5,(HL)	DD3505	DEC	(IX+IND)
A0	AND	B	DDCB056E	BIT	5,(IX+IND)	FD3505	DEC	(IY+IND)
A1	AND	C	FDCB056E	BIT	5,(IY+IND)	3D	DEC	A
A2	AND	D	CB6F	BIT	5,A	05	DEC	B
A3	AND	E	CB68	BIT	5,B	0B	DEC	BC
A4	AND	H	CB69	BIT	5,C	0D	DEC	C
A5	AND	L	CB6A	BIT	5,D	15	DEC	D
E620	AND	N	CB6B	BIT	5,E	1B	DEC	DE
CB46	BIT	0,(HL)	CB6C	BIT	5,H	1D	DEC	E
DDCB0546	BIT	0,(IX+IND)	CB6D	BIT	5,L	25	DEC	H
FDCB0546	BIT	0,(IY+IND)	CB76	BIT	6,(HL)	2B	DEC	HL
CB47	BIT	0,A	DDCB0576	BIT	6,(IX+IND)	DD2B	DEC	IX
CB40	BIT	0,B	FDCB0576	BIT	6,(IY+IND)	FD2B	DEC	IY
CB41	BIT	0,C	CB77	BIT	6,A	2D	DEC	L
CB42	BIT	0,D	CB70	BIT	6,B	3B	DEC	SP
CB43	BIT	0,E	CB71	BIT	6,C	F3	DI	
CB44	BIT	0,H	CB72	BIT	6,D	102E	DJNZ	DIS

OPCODE	SOURCE CODE		OPCODE	SOURCE CODE		OPCODE	SOURCE CODE	
EB	EI		74	LD	(HL),H	4D	LD	C,L
DDE3	EX	(SP),IX	75	LD	(HL),L	0E20	LD	C,N
FDE3	EX	(SP),IY	3620	LD	(HL),N	56	LD	D,(HL)
08	EX	AF,AF'	DD7705	LD	(IX+IND),A	DD5605	LD	D,(IX+IND)
EB	EX	DE,HL	DD7005	LD	(IX+IND),B	FD5605	LD	D,(IY+IND)
E3	EX	SP),HL	DD7105	LD	(IX+IND),C	57	LD	D,A
D9	EXX		DD7205	LD	(IX+IND),D	50	LD	D,B
76	HALT		DD7305	LD	(IX+IND),E	51	LD	D,C
ED46	IM	0	DD7405	LD	(IX+IND),H	52	LD	D,D
ED56	IM	1	DD7505	LD	(IX+IND),L	53	LD	D,E
ED5E	IM	2	DD360520	LD	(IX+IND),N	54	LD	D,H
ED78	IN	A,(C)	FD7705	LD	(IY+IND),A	55	LD	D,L
DB20	IN	A,N	FD7005	LD	(IY+IND),B	1620	LD	D,N
ED45	IN	C,(C)	FD7105	LD	(IY+IND),C	ED5B8405	LD	DE,(NN)
ED50	IN	D,(C)	FD7205	LD	(IY+IND),D	118405	LD	DE,NN
ED58	IN	E,(C)	FD7305	LD	(IY+IND),E	5E	LD	E,(HL)
ED40	IN	H,(C)	FD7405	LD	(IY+IND),H	DD5E05	LD	E,(IX+IND)
ED60	IN	H,(C)	FD7505	LD	(IY+IND),L	FD5E05	LD	E,(IY+IND)
ED65	IN	L,(C)	FD360520	LD	(IY+IND),N	5F	LD	E,A
34	INC	(HL)	328405	LD	(NN),A	58	LD	E,B
DD3405	INC	(IX+IND)	ED438405	LD	(NN),BC	59	LD	E,C
FD3405	INC	(IY+IND)	ED538405	LD	(NN),DE	5A	LD	E,D
3C	INC	A	228405	LD	(NN),HL	5B	LD	E,E
04	INC	B	DD228405	LD	(NN),IX	5C	LD	E,H
03	INC	BC	FD225405	LD	(NN),IY	5D	LD	E,L
0C	INC	C	ED738405	LD	(NN),SP	1E20	LD	E,N
14	INC	D	0A	LD	A,(BC)	66	LD	H,(HL)
13	INC	DE	1A	LD	A,(DE)	DD6605	LD	H,(IX±IND)
1C	INC	E	7E	LD	A,(HL)	FD6605	LD	H,(IY+IND)
24	INC	H	DD7E05	LD	A,(IX+IND)	67	LD	H,A
23	INC	HL	FD7E05	LD	A,(IY+IND)	60	LD	H,B
DD23	INC	IX	3A8405	LD	A,(NN)	61	LD	H,C
FD23	INC	IY	7F	LD	A,A	62	LD	H,D
2C	INC	L	78	LD	A,B	63	LD	H,E
33	INC	SP	79	LD	A,C	64	LD	H,H
EDAA	IND		7A	LD	A,D	65	LD	H,L
EDBA	INDR		7B	LD	A,E	2620	LD	H,N
EDA2	INI		7C	LD	A,H	2A8405	LD	HL,(NN)
EDB2	INIR		ED57	LD	A,I	218405	LD	HL,NN
E9	JP	(HL)	7D	LD	A,L	ED47	LD	I,A
DDE9	JP	(IX)	3E20	LD	A,N	DD2A8405	LD	IX,(NN)
FDE9	JP	(IY)	46	LD	B,(HL)	DD218405	LD	IX,NN
DA8405	JP	C,NN	DD4605	LD	B,(IX+IND)	FD2A8405	LD	IY,(NN)
FA8405	JP	M,NN	FD4605	LD	B,(IY+IND)	FD215405	LD	IY,NN
D28405	JP	NC,NN	47	LD	B,A	6E	LD	L,(HL)
C38405	JP	NN	40	LD	B,B	DD6E05	LD	L,(IX+IND)
C28405	JP	NZ,NN	41	LD	B,C	FD6E05	LD	L,(IY+IND)
F28405	JP	P,NN	42	LD	B,D	6F	LD	L,A
EA8405	JP	PE,NN	43	LD	B,E	68	LD	L,B
E28405	JP	PO,NN	44	LD	B,H,NN	69	LD	L,C
CA8405	JP	Z,NN	45	LD	B,L	6A	LD	L,D
382E	JR	C,DIS	0620	LD	B,N	6B	LD	L,E
182E	JR	DIS	ED4B8405	LD	BC,(NN)	6C	LD	L,H
302E	JR	NC,DIS	018405	LD	BC,NN	6D	LD	L,L
202E	JR	NZ,DIS	4E	LD	C,(HL)	2E20	LD	L,N
282E	JR	Z,DIS	DD4E05	LD	C,(IX+IND)	ED7B8405	LD	SP,(NN)
02	LD	(BC),A	FD4E05	LD	C,(IY+IND)	F9	LD	SP,HL
12	LD	(DE),A	4F	LD	C,A	DDF9	LD	SP,IX
77	LD	(HL),A	48	LD	C,B	FDF9	LD	SP,IY
70	LD	(HL),B	49	LD	C,C	318405	LD	SP,NN
71	LD	(HL),C	4A	LD	C,D	EDA5	LDD	
72	LD	(HL),D	4B	LD	C,E	EDB8	LDDR	
73	LD	(HL),E	4C	LD	C,H	EDA0	LDI	

OPCODE	SOURCE CODE	OPCODE	SOURCE CODE	OPCODE	SOURCE CODE
EDB0	LDIR	CB91	RES 2,C	C8	RET Z
ED44	NEG	CB92	RES 2,D	ED4D	RETI
00	NOP	CB93	RES 2,E	ED45	RETN
B6	OR (HL)	CB94	RES 2,H	CB16	RL (HL)
DDB605	OR (IX+IND)	CB95	RES 2,L	DDCB0516	RL (IX+IND)
FDB605	OR (IY+IND)	CB9E	RES 3,(HL)	FDCB0516	RL (IY+IND)
B7	OR A	DDCB059E	RES 3,(IX+IND)	CB17	RL A
B0	OR B	FDCB059E	RES 3,(IY+IND)	CB10	RL B
B1	OR C	CB9F	RES 3,A	CB11	RL C
B2	OR D	CB98	RES 3,B	CB12	RL D
B3	OR E	CB99	RES 3,C	CB13	RL E
B4	OR H	CB9A	RES 3,D	CB14	RL H
B5	OR L	CB9B	RES 3,E	CB15	RL L
F620	OR N	CB9C	RES 3,H	17	RLA
EDBB	OTDR	CB9D	RES 3,L	CB06	RLC (HL)
EDB3	OTIR	CBA6	RES 4,(HL)	DDCB0506	RLC (IX+IND)
ED79	OUT (C),A	DDCB05A6	RES 4,(IX+1ND)	FDCB0506	RLC (IY+IND)
ED41	OUT (C),B	FDCB05A6	RES 4,(IY+IND)	CB07	RLC A
ED49	OUT (C),C	CBA7	RES 4,A	CB00	RLC B
ED51	OUT (C),D	CBA0	RES 4,B	CB01	RLC C
ED59	OUT (C),E	CBA1	RES 4,C	CB02	RLC D
ED61	OUT (C),H	CBA2	RES 4,D	CB03	RLC E
ED69	OUT (C),L	CBA3	RES 4,E	CB04	RLC H
D320	OUT N,A	CBA4	RES 4,H	CB05	RLC L
EDAB	OUTD	CBA5	RES 4,L	07	RLCA
EDA3	OUTI	CBAE	RES 5,(HL)	ED6F	RLD
F1	POP AF	DDCB05AE	RES 5,(IX+IND)	FDCB051E	RR (IY+IND)
C1	POP BC	FDCB05AE	RES 5,(IY+IND)	CB18	RR B
D1	POP DE	CBAF	RES 5,A	CB19	RR C
E1	POP HL	CBA8	RES 5,B	CB1A	RR D
DDE1	POP IX	CBA9	RES 5,C	CB1B	RR E
FDE1	POP IY	CBAA	RES 5,D	CB1C	RR H
F5	PUSH AF	CBAB	RES 5,E	CB1D	RR L
C5	PUSH BC	CBAC	RES 5,H	CB1E	RR (HL)
D5	PUSH DE	CBAD	RES 5,L	DDCB051E	RR(IX+IND)
E5	PUSH HL	CBB6	RES 6,(HL)	1F	RRA
DDE5	PUSH IX	DDCB05B6	RES 6,(IX+IND)	CB1F	RRA
FDE5	PUSH IY	FDCB05B6	RES 6,(IY+IND)	CB0E	RRC (HL)
CB86	RES 0,(HL)	CBB7	RES 6,A	DDCB050E	RRC (IX+IND)
DDCB0556	RES 0,(IX+IND)	CBB0	RES 6,B	FDCB050E	RRC (IY+IND)
FDCB0586	RES 0,(IY+IND)	CBB1	RES 6,C	CB0F	RRC A
CB87	RES 0,A	CBB2	RES 6,D	CB08	RRC B
CB80	RES 0,B	CBB3	RES 6,E	CB09	RRC C
CB81	RES 0,C	CBB4	RES 6,H	CB0A	RRC D
CB82	RES 0,D	CBB5	RES 6,L	CB0B	RRC E
CB83	RES 0,E	CBBE	RES 7,(HL)	CB0C	RRC H
CB84	RES 0,H	DDCB05BE	RES 7,(IX+IND)	CB0D	RRC L
CB85	RES 0,L	FDCB05BE	RES 7,(IY+IND)	0F	RRCA
CB8E	RES 1,(HL)	CBBF	RES 7,A	ED67	RRD
DDCB055E	RES 1,(IX+IND)	CBB8	RES 7,B	C7	RST 0
FDCB055E	RES 1,(IY+IND)	CBB9	RES 7,C	D7	RST 10H
CB8F	RES 1,A	CBBA	RES 7,D	DF	RST 18H
CB88	RES 1,B	CBBB	RES 7,E	E7	RST 20H
CB89	RES 1,C	CBBC	RES 7,H	EF	RST 28H
CB8A	RES 1,D	CBBD	RES 7,L	F7	RST 30H
CB8B	RES 1,E	C9	RET	FF	RST 38H
CB8C	RES 1,H	D8	RET C	CF	RST 8
CB8D	RES 1,L	F8	RET M	9E	SBC A,(HL)
CB96	RES 2,(HL)	D0	RET NC	DD9E05	SBC A,(IX+IND)
DDCB0596	RES 2,(IX+IND)	C0	RET NZ	FD9E05	SBC A,(IY+IND)
FDCB0596	RES 2,(IY+IND)	F0	RET P	9F	SBC A,A
CB97	RES 2,A	E8	RET PE	98	SBC A,B
CB90	RES 2,B	E0	RET PO	99	SBC A,C

OPCODE	SOURCE CODE		OPCODE	SOURCE CODE		OPCODE	SOURCE CODE	
9A	SBC	A,D	CBE6	SET	4,(HL)	CB2E	SRA	(HL)
9B	SBC	A,E	DDCB05E6	SET	4,(IX+1ND)	DDCB052E	SRA	(IX+IND)
9C	SBC	A,H	FDCB05E6	SET	4,(IY+IND)	FDCB052E	SRA	(IY+IND)
9D	SBC	A,L	CBE7	SET	4,A	CB2F	SRA	A
DE20	SBC	A,N	CBE0	SET	4,B	CB28	SRA	B
ED42	SBC	HL,BC	CBE1	SET	4,C	CB29	SRA	C
ED52	SBC	HL,DE	CBE2	SET	4,D	CB2A	SRA	D
ED62	SBC	HL,HL	CBE3	SET	4,E	CB2B	SRA	E
ED72	SBC	HL,SP	CBE4	SET	4,H	CB2C	SRA	H
37	SCF		CBE5	SET	4,L	CB2D	SRA	L
CBC6	SET	0,(HL)	CBEE	SET	5,(HL)	CB3E	SRL	(HL)
DDCB05C6	SET	0,(IX+IND)	DDCB05EE	SET	5,(IX+IND)	DDCB053E	SRL	(IX+IND)
FDCB05C6	SET	0,(IY+IND)	FDCB05EE	SET	5,(IY+IND)	FDCB053E	SRL	(IY+IND)
CBC7	SET	0,A	CBEF	SET	5,A	CB3F	SRL	A
CBC0	SET	0,B	CBE8	SET	5,B	CB38	SRL	B
CBC1	SET	0,C	CBE9	SET	5,C	CB39	SRL	C
CBC2	SET	0,D	CBEA	SET	5,D	CB3A	SRL	D
CBC3	SET	0,E	CBEB	SET	5,E	CB3B	SRL	E
CBC4	SET	0,H	CBEC	SET	5,H	CB3C	SRL	H
CBC5	SET	0,L	CBED	SET	5,L	CB3D	SRL	L
CBCE	SET	1,(HL)	CBF6	SET	6,(HL)	96	SUB	(HL)
DDCB05CE	SET	1,(IX+IND)	DDCB05F6	SET	6,(IX+IND)	DD9605	SUB	(IX+IND)
FDCB05CE	SET	1,(IY+IND)	FDCB05F6	SET	6,(IY+IND)	FD9605	SUB	(IY+IND)
CBCF	SET	1,A	CBF7	SET	6,A	97	SUB	A
CBC8	SET	1,B	CBF0	SET	6,B	90	SUB	B
CBC9	SET	1,C	CBF1	SET	6,C	91	SUB	C
CBCA	SET	1,D	CBF2	SET	6,D	92	SUB	D
CBCB	SET	1,E	CBF3	SET	6,E	93	SUB	E
CBCC	SET	1,H	CBF4	SET	6,H	94	SUB	H
CBCD	SET	1,L	CBF5	SET	6,L	95	SUB	L
CBD6	SET	2,(HL)	CBFE	SET	7,(HL)	D620	SUB	N
DDCB05D6	SET	2,(IX+IND)	DDCB05FE	SET	7,(IX+IND)	AE	XOR	(HL)
FDCB05D6	SET	2,(IY+IND)	FDCB05FE	SET	7,(IY+IND)	DDAE05	XOR	(IX+IND)
CBD7	SET	2,A	CBFF	SET	7,A	FDAE05	XOR	(IY+IND)
CBD0	SET	2,B	CBF9	SET	7,C	AF	XOR	A
CBD1	SET	2,C	CBFA	SET	7,D	A8	XOR	B
CBD2	SET	2,D	CBFB	SET	7,E	A9	XOR	C
CBD3	SET	2,E	CBF8	SET	7,H	AA	XOR	D
CBD4	SET	2,H	CBFC	SET	7,H	AB	XOR	E
CBD5	SET	2,L	CBFD	SET	7,L	AC	XOR	H
CBDE	SET	3,(HL)	CB26	SLA	(HL)	AD	XOR	L
DDCB05DE	SET	3,(IX+IND)	DDCB0526	SLA	(IX+IND)	EE20	XOR	N
FDCB05DE	SET	3,(IY+IND)	FDCB0526	SLA	(IY+IND)	NN	DEFS	2
CBDF	SET	3,A	CB27	SLA	A	IND	EQU	5
CBD8	SET	3,B	CB20	SLA	B	M	EQU	10H
CBD9	SET	3,C	CB21	SLA	C	N	EQU	20H
CBDA	SET	3,D	CB22	SLA	D	DIS	EQU	30H
CBDB	SET	3,E	CB23	SLA	E		END	
CBDC	SET	3,H	CB24	SLA	H			
CBDD	SET	3,L	CB25	SLA	L			

Appendix B. Z-80 Instruction Set by Operation Codes

OPCODE	SOURCE CODE		OPCODE	SOURCE CODE		OPCODE	SOURCE CODE	
00	NOP		3B	DEC	SP	76	HALT	
018405	LD	BC,NN	3C	INC	A	77	LD	(HL),A
02	LD	(BC),A	3D	DEC	A	78	LD	A.B
03	INC	BC	3E20	LD	A,N	79	LD	A,C
04	INC	B	3F	CCF		7A	LD	A,D
05	DEC	B	40	LD	B,B	7B	LD	A,E
0620	LD	B,N	41	LD	B,C	7C	LD	A,H
07	RLCA		42	LD	B,D	7D	LD	A,L
08	EX	AF,AF'	43	LD	B,E	7E	LD	A,(HL)
09	ADD	HL,BC	44	LD	B,H,NN	7F	LD	A,A
0A	LD	A,(BC)	45	LD	B,L	80	ADD	A,B
0B	DEC	BC	46	LD	B,(HL)	81	ADD	A,C
0C	INC	C	47	LD	B,A	82	ADD	A,D
0D	DEC	C	48	LD	C,B	83	ADD	A,E
0E20	LD	C,N	49	LD	C,C	84	ADD	A,H
0F	RRCA		4A	LD	C,D	85	ADD	A,L
102E	DJNZ	DIS	4B	LD	C,E	86	ADD	A,(HL)
118405	LD	DE,NN	4C	LD	C,H	87	ADD	A,A
12	LD	(DE),A	4D	LD	C,L	88	ADC	A,B
13	INC	DE	4E	LD	C,(HL)	89	ADC	A,C
14	INC	D	4F	LD	C,A	8A	ADC	A,D
15	DEC	D	50	LD	D,B	8B	ADC	A,E
1620	LD	D,N	51	LD	D,C	8C	ADC	A,H
17	RLA		52	LD	D,D	8D	ADC	A,L
182E	JR	DIS	53	LD	D,E	8E	ADC	A,(HL)
19	ADD	HL,DE	54	LD	D,H	8F	ADC	A,A
1A	LD	A,(DE)	55	LD	D,L	90	SUB	B
1B	DEC	DE	56	LD	D,(HL)	91	SUB	C
1C	INC	E	57	LD	D,A	92	SUB	D
1D	DEC	E	58	LD	E,B	93	SUB	E
1E20	LD	E,N	59	LD	E,C	94	SUB	H
1F	RRA		5A	LD	E,D	95	SUB	L
202E	JR	NZ,DIS	5B	LD	E,E	96	SUB	(HL)
218405	LD	HL,NN	5C	LD	E,H	97	SUB	A
228405	LD	(NN),HL	5D	LD	E,L	98	SBC	A,B
23	INC	HL	5E	LD	E,(HL)	99	SBC	A,C
24	INC	H	5F	LD	E,A	9A	SBC	A,D
25	DEC	H	60	LD	H,B	9B	SBC	A,E
2620	LD	H,N	61	LD	H,C	9C	SBC	A,H
27	DAA		62	LD	H,D	9D	SBC	A,L
282E	JR	Z,DIS	63	LD	H,E	9E	SBC	A,(HL)
29	ADD	HL,HL	64	LD	H,H	9F	SBC	A,A
2A8405	LD	HL,(NN)	65	LD	H,L	A0	AND	B
2B	DEC	HL	66	LD	H,(HL)	A1	AND	C
2C	INC	L	67	LD	H,A	A2	AND	D
2D	DEC	L	68	LD	L,B	A3	AND	E
2E20	LD	L,N	69	LD	L,C	A4	AND	H
2F	CPL		6A	LD	L.D	A5	AND	L
302E	JR	NC,DIS	6B	LD	L,E	A6	AND	(HL)
318405	LD	SP,NN	6C	LD	L,H	A7	AND	A
328405	LD	(NN),A	6D	LD	L,L	A8	XOR	B
33	INC	SP	6E	LD	L,(HL)	A9	XOR	C
34	INC	(HL)	6F	LD	L,A	AA	XOR	D
35	DEC	(HL)	70	LD	(HL),B	AB	XOR	E
3620	LD	(HL),N	71	LD	(HL),C	AC	XOR	H
37	SCF		72	LD	(HL),D	AD	XOR	L
382E	JR	C,DIS	73	LD	(HL),E	AE	XOR	(HL)
39	ADD	HL,SP	74	LD	(HL),H	AF	XOR	A
3A8405	LD	A,(NN)	75	LD	(HL),L	B0	OR	B

OPCODE	SOURCE CODE		OPCODE	SOURCE CODE		OPCODE	SOURCE CODE	
B1	OR	C	F3	DI		CB3B	SRL	E
B2	OR	D	F48405	CALL	P,NN	CB3C	SRL	H
B3	OR	E	F5	PUSH	AF	CB3D	SRL	L
B4	OR	H	F620	OR	N	CB3E	SRL	(HL)
B5	OR	L	F7	RST	30H	CB3F	SRL	A
B6	OR	(HL)	F8	RET	M	CB40	BIT	0,B
B7	OR	A	F9	LD	SP,HL	CB41	BIT	0,C
B8	CP	H	FA8405	JP	M,NN	CB42	BIT	0,D
B9	CP	C	EB	E1		CB43	BIT	0,E
BA	CP	D	FC8405	CALL	M,NN	CB44	BIT	0,H
BB	CP	E	FE20	CP	N	CB45	BIT	0,L
BC	CP	H	FF	RST	38H	CB46	BIT	0,(HL)
BD	CPL		CB00	RLC	B	CB47	BIT	0,A
BE	CP	(HL)	CB01	RLC	C	CB48	BIT	1,B
BF	CPA		CB02	RLC	D	CB49	BIT	1,C
C0	RET	NZ	CB03	RLC	E	CB4A	BIT	1,D
C1	POP	BC	CB04	RLC	H	CB4B	BIT	1.E
C28405	JP	NZ,NN	CB05	RLC	L	CB4C	BIT	1,H
C38405	JP	NN	CB06	RLC	(HL)	CB4D	BIT	1,L
C48405	CALL	NZ,NN	CB07	RLC	A	CB4E	BIT	1,(HL)
C5	PUSH	BC	CB08	RRC	B	CB4F	BIT	1,A
C620	ADD	A,N	CB09	RRC	C	CB50	BIT	2,B
C7	RST	0	CB0A	RRC	D	CB51	BIT	2,C
C8	RET	Z	CB0B	RRC	E	CB52	BIT	2,D
C9	RET		CB0C	RRC	H	CB53	BIT	2,E
CA8405	JP	Z,NN	CB0D	RRC	L	CB54	BIT	2,H
CC8405	CALL	Z,NN	CB0E	RRC	(HL)	CB55	BIT	2,L
CD8405	CALL	NN	CB0F	RRC	A	CB56	BIT	2,(HL)
CE20	ADC	A,N	CB10	RL	B	CB57	BIT	2,A
CF	RST	8	CB11	RL	C	CB58	BIT	3,B
D0	RET	NC	CB12	RL	D	CB59	BIT	3,C
D1	POP	DE	CB13	RL	E	CB5A	BIT	3,D
D28405	JP	NC,NN	CB14	RL	H	CB5B	BIT	3,E
D320	OUT	N,A	CB15	RL	L	CB5C	BIT	3,H
D48405	CALL	NC,NN	CB16	RL	(HL)	CB5D	BIT	3,L
D5	PUSH	DE	CB17	RL	A	CB5E	BIT	3,(HL)
D620	SUB	N	CB18	RR	B	CB5F	BIT	3,A
D7	RST	10H	CB19	RR	C	CB60	BIT	4,H
D8	RET	C	CB1A	RR	D	CB61	BIT	4,C
D9	EXX		CB1B	RR	E	CB62	BIT	4,D
DA8405	JP	C,NN	CB1C	RR	H	CB63	BIT	4,E
DB20	IN	A,N	CB1D	RR	L	CB64	BIT	4,H
DC8405	CALL	C,NN	CB1E	RR	(HL)	CB65	BIT	4,L
DE20	SBC	A,N	CB1F	RRA		CB66	BIT	4,(HL)
DF	RST	18H	CB20	SLA	B	CB67	BIT	4,A
E0	RET	PO	CB21	SLA	C	CB68	BIT	5,B
EI	POP	HL	CB22	SLA	D	CB69	BIT	5.C
E28405	JP	PO,NN	CB23	SLA	E	CB6A	BIT	5,D
E3	EX	SP).HL	CB24	SLA	H	CB6B	BIT	5,E
E48405	CALL	PO,NN	CB25	SLA	L	CB6C	BIT	5,H
E5	PUSH	HL	CB26	SLA	(HL)	CB6D	BIT	5,L
E620	AND	N	CB27	SLA	A	CB6E	BIT	5,(HL)
E7	RST	20H	CB28	SRA	B	CB6F	BIT	5,A
E8	RET	PE	CB29	SRA	C	CB70	BIT	6,B
E9	JP	(HL)	CB2A	SRA	D	CB71	BIT	6,C
EA8405	JP	PE,NN	CB2B	SRA	E	CB72	BIT	6,D
EB	EX	DE,HL	CB2C	SRA	H	CB73	BIT	6,E
EC8405	CALL	PE,NN	CB2D	SRA	L	CB74	BIT	6,H
EE20	XOR	N	CB2E	SRA	(HL)	CB75	BIT	6,L
EF	RST	28H	CB2F	SRA	A	CB76	BIT	6,(HL)
F0	RET	P	CB38	SRL	B	CB77	BIT	6,A
F1	POP	AF	CB39	SRL	C	CB78	BIT	7,B
F28405	JP	P,NN	CB3A	SRL	D	CB79	BIT	7,C

OPCODE	SOURCE CODE		OPCODE	SOURCE CODE		OPCODE	SOURCE CODE	
CB7A	BIT	7,D	CBB9	RES	7,C	CBF8	SET	7,H
CB7B	BIT	7,E	CBBA	RES	7,D	CBF9	SET	7,C
CB7C	BIT	7,H	CBBB	RES	7,E	CBFA	SET	7,D
CB7D	BIT	7,L	CBBC	RES	7,H	CBFB	SET	7,E
CB7E	BIT	7,(HL)	CBBD	RES	7,L	CBFC	SET	7,H
CB7F	BIT	7,A	CBBE	RES	7,(HL)	CBFD	SET	7,L
CB80	RES	0,B	CBBF	RES	7,A	CBFE	SET	7,(HL)
CB81	RES	0,C	CBC0	SET	0,B	CBFF	SET	7,A
CB82	RES	0,D	CBC1	SET	0,C	DD09	ADD	IX,BC
CB83	RES	0,E	CBC2	SET	0,D	DD19	ADD	IX,DE
CB84	RES	0,H	CBC3	SET	0,E	DD218405	LD	IX,NN
CB85	RES	0,L	CBC4	SET	0,H	DD228405	LD	(NN),IX
CB86	RES	0,(HL)	CBC5	SET	0,L	DD23	INC	IX
CB87	RES	0,A	CBC6	SET	0,(HL)	DD29	ADD	IX,IX
CB88	RES	1,B	CBC7	SET	0,A	DD2A8405	LD	IX,(NN)
CB89	RES	1,C	CBC8	SET	1,B	DD2B	DEC	IX
CB8A	RES	1,D	CBC9	SET	1,C	DD3405	INC	(IX+IND)
CB8B	RES	1,E	CBCA	SET	1,D	DD3505	DEC	(IX+IND)
CB8C	RES	1,H	CBCB	SET	1,E	DD360520	LD	(IX+IND),N
CB8D	RES	1,L	CBCC	SET	1,H	DD39	ADD	IX,SP
CB8E	RES	1,(HL)	CBCD	SET	1,L	DD4605	LD	B,(IX+IND)
CB8F	RES	1,A	CBCE	SET	1,(HL)	DD4E05	LD	C,(IX+IND)
CB90	RES	2,B	CBCF	SET	1,A	DD5605	LD	D,(IX+IND)
CB91	RES	2,C	CBD0	SET	2,B	DD5E05	LD	E.(IX+IND)
CB92	RES	2,D	CBD1	SET	2,C	DD6605	LD	H,(IX+IND)
CB93	RES	2,E	CBD2	SET	2,D	DD6E05	LD	L,(IX+IND)
CB94	RES	2,H	CBD3	SET	2,E	DD7005	LD	(IX+IND),B
CB95	RES	2,L	CBD4	SET	2,H	DD7105	LD	(IX+IND),C
CB96	RES	2,(HL)	CBD5	SET	2,L	DD7205	LD	(IX+IND),D
CB97	RES	2,A	CBD6	SET	2,(HL)	DD7305	LD	(IX+IND),E
CB98	RES	3,B	CBD7	SET	2,A	DD7405	LD	(IX+IND),H
CB99	RES	3,C	CBD8	SET	3,B	DD7505	LD	(IX+IND),L
CB9A	RES	3,D	CBD9	SET	3,C	DD7705	LD	(IX+IND),A
CB9B	RES	3,E	CBDA	SET	3,D	DD7E05	LD	A,(IX+IND)
CB9C	RES	3,H	CBDB	SET	3,E	DD8605	ADD	A,(IX+IND)
CB9D	RES	3,L	CBDC	SET	3,H	DD8E05	ADC	A,(IX+IND)
CB9E	RES	3,(HL)	CBDD	SET	3,L	DD9605	SUB	(IX+IND)
CB9F	RES	3,A	CBDE	SET	3,(HL)	DD9E05	SBC	A,(IX+IND)
CBA0	RES	4,B	CBDF	SET	3,A	DDA605	AND	(IX+IND)
CBA1	RES	4,C	CBE0	SET	4,B	DDAE05	XOR	(IX+IND)
CBA2	RES	4,D	CBE1	SET	4,C	DDB605	OR	(IX+IND)
CBA3	RES	4,E	CBE2	SET	4,D	DDBE05	CP	(IX+IND)
CBA4	RES	4,H	CBE3	SET	4,E	DDE1	POP	IX
CBA5	RES	4,L	CBE4	SET	4,H	DDE3	EX	(SP),IX
CBA6	RES	4,(HL)	CBE5	SET	4,L	DDE5	PUSH	IX
CBA7	RES	4,A	CBE6	SET	4,(HL)	DDE9	JP	(IX)
CBA8	RES	5,B	CBE7	SET	4,A	DDF9	LD	SP,IX
CBA9	RES	5,C	CBE8	SET	5,B	DDCB0506	RLC	(IX+IND)
CBAA	RES	5,D	CBE9	SET	5,C	DDCB050E	RRC	(IX+IND)
CBAB	RES	5,E	CBEA	SET	5,D	DDCB0516	RL	(IX+IND)
CBAC	RES	5,H	CBEB	SET	5,E	DDCB051E	RR(IX+IND)	
CBAD	RES	5,L	CBEC	SET	5,H	DDCB0526	SLA	(IX+IND)
CBAE	RES	5,(HL)	CBED	SET	5,L	DDCB052E	SRA	(IX+IND)
CBAF	RES	5,A	CBEE	SET	5,(HL)	DDCB053E	SRL	(IX+IND)
CBB0	RES	6,B	CBEF	SET	5,A	DDCB0546	BIT	0,(IX+IND)
CBB1	RES	6,C	CBF0	SET	6,B	DDCB054E	BIT	1,(IX+IND)
CBB2	RES	6,D	CBF1	SET	6,C	DDCB0556	BIT	2,(IX+IND)
CBB3	RES	6,E	CBF2	SET	6,D	DDCB055E	BIT	3,(IX+IND)
CBB4	RES	6,H	CBF3	SET	6,E	DDCB0566	BIT	4,(IX+1ND)
CBB5	RES	6,L	CBF4	SET	6,H	DDCB056E	BIT	5,(IX+IND)
CBB6	RES	6,(HL)	CBF5	SET	6,L	DDCB0576	BIT	6,(IX+IND)
CBB7	RES	6,A	CBF6	SET	6,(HL)	DDCB057E	BIT	7,(IX+IND)
CBB8	RES	7,B	CBF7	SET	6,A	DDCB0556	RES	0,(IX+IND)

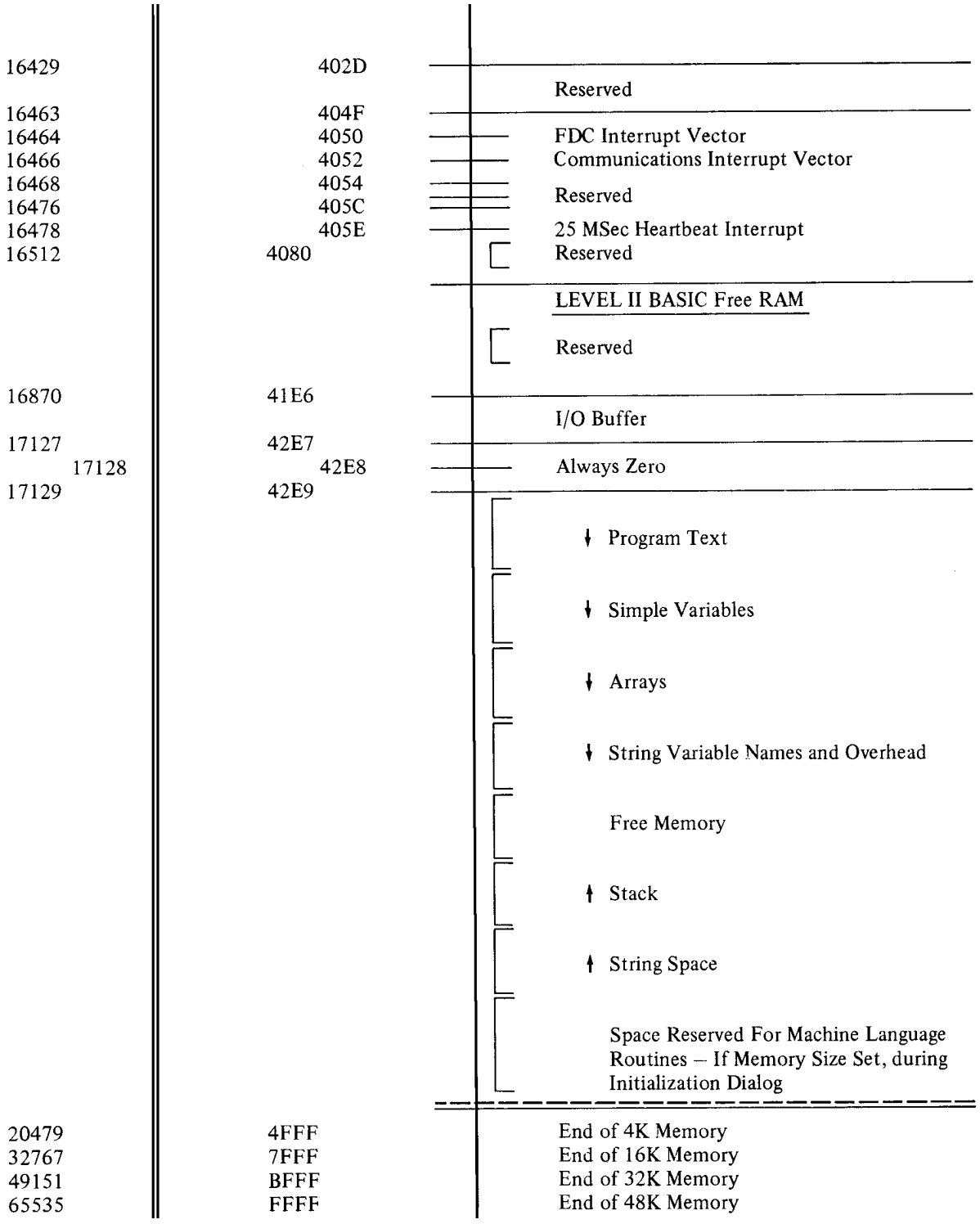
```
OPCODE    SOURCE CODE
DDCB055E  RES   1,(IX+IND)
DDCB0596  RES   2,(IX+IND)
DDCB059E  RES   3,(IX+IND)
DDCB05A6  RES   4,(IX+1ND)
DDCB05AE  RES   5,(IX+IND)
DDCB05B6  RES   6,(IX+IND)
DDCB05BE  RES   7,(IX+IND)
DDCB05C6  SET   0,(IX+IND)
DDCB05CE  SET   1,(IX+IND)
DDCB05D6  SET   2,(IX+IND)
DDCB05DE  SET   3,(IX+IND)
DDCB05E6  SET   4,(IX+1ND)
DDCB05EE  SET   5,(IX+IND)
DDCB05F6  SET   6,(IX+IND)
DDCB05FE  SET   7,(IX+IND)
ED40      IN    H,(C)
ED41      OUT   (C),B
ED42      SBC   HL,BC
ED438405  LD    (NN),BC
ED44      NEG
ED45      RETN
ED46      IM    0
ED47      LD    I,A
ED45      IN    C,(C)
ED49      OUT   (C),C
ED4A      ADC   HL,BC
ED4B8405  LD    BC,(NN)
ED4D      RETI
ED50      IN    D,(C)
ED51      OUT   (C),D
ED52      SBC   HL,DE
ED538405  LD    (NN),DE
ED56      IM    1
ED57      LD    A,I
ED58      IN    E,(C)
ED59      OUT   (C),E
ED5A      ADC   HL,DE
ED5B8405  LD    DE,(NN)
ED5E      IM    2
ED60      IN    H,(C)
ED61      OUT   (C),H
ED62      SBC   HL,HL
ED67      RRD
ED65      IN    L,(C)
ED69      OUT   (C),L
ED6A      ADC   HL,HL
ED6F      RLD
ED72      SBC   HL,SP
ED738405  LD    (NN),SP
ED78      IN    A,(C)
ED79      OUT   (C),A
ED7A      ADC   HL,SP
ED7B8405  LD    SP,(NN)
EDA0      LDI
EDA1      CPI
EDA2      INI
EDA3      OUTI
EDA5      LDD
EDA9      CPD
EDAA      IND
EDAB      OUTD
EDB0      LDIR
EDB1      CPIR
EDB2      INIR
EDB3      OTIR
EDB8      LDDR
EDB9      CPDR
EDBA      INDR
EDBB      OTDR
FD09      ADD   IY,BC
FD19      ADD   IY,DE
FD215405  LD    IY,NN
FD225405  LD    (NN),IY
FD23      INC   IY
FD29      ADD   IY,IY
FD2A8405  LD    IY,(NN)
FD2B      DEC   IY
FD3405    INC   (IY+IND)
FD3505    DEC   (IY+IND)
FD360520  LD    (IY+IND),N
FD39      ADD   IY,SP
FD4605    LD    B,(IY+IND)
FD4E05    LD    C,(IY+IND)
FD5605    LD    D,(IY+IND)
FD5E05    LD    E,(IY+IND)
FD6605    LD    H,(IY+IND)
FD6E05    LD    L,(IY+IND)
FD7005    LD    (IY+IND),B
FD7105    LD    (IY+IND),C
FD7205    LD    (IY+IND),D
FD7305    LD    (IY+IND),E
FD7405    LD    (IY+IND),H
FD7505    LD    (IY+IND),L
FD7705    LD    (IY+IND),A
FD7E05    LD    A,(IY+IND)
FD8605    ADD   A,(IY+IND)
FD8E05    ADC   A,(IY+IND)
FD9605    SUB   (IY+IND)
FD9E05    SBC   A,(IY+IND)
FDA605    AND   (IY+IND)
FDAE05    XOR   (IY+IND)
FDB605    OR    (IY+IND)
FDBE05    CP    (IY+IND)
FDE1      POP   IY
FDE3      EX    (SP),IY
FDE5      PUSH  IY
FDE9      JP    (IY)
FDF9      LD    SP,IY
FDCB0506  RLC   (IY+IND)
FDCB050E  RRC   (IY+IND)
FDCB0516  RL    (IY+IND)
FDCB051E  RR    (IY+IND)
FDCB0526  SLA   (IY+IND)
FDCB052E  SRA   (IY+IND)
FDCB053E  SRL   (IY+IND)
FDCB0546  BIT   0,(IY+IND)
FDCB054E  BIT   1,(IY+IND)
FDCB0556  BIT   2,(IY+IND)
FDCB055E  BIT   3,(IY+IND)
FDCB0566  BIT   4,(IY+IND)
FDCB056E  BIT   5,(IY+IND)
FDCB0576  BIT   6,(IY+IND)
FDCB057E  BIT   7,(IY+IND)
FDCB0586  RES   0,(IY+IND)
FDCB055E  RES   1,(IY+IND)
FDCB0596  RES   2,(IY+IND)
FDCB059E  RES   3,(IY+IND)
FDCB05A6  RES   4,(IY+IND)
FDCB05AE  RES   5,(IY+IND)
FDCB05B6  RES   6,(IY+IND)
FDCB05BE  RES   7,(IY+IND)
FDCB05C6  SET   0,(IY+IND)
FDCB05CE  SET   1,(IY+IND)
FDCB05D6  SET   2,(IY+IND)
FDCB05DE  SET   3,(IY+IND)
FDCB05E6  SET   4,(IY+IND)
FDCB05EE  SET   5,(IY+IND)
FDCB05F6  SET   6,(IY+IND)
FDCB05FE  SET   7,(IY+IND)
NN        DEFS  2
IND       EQU   5
M         EQU   10H
N         EQU   20H
DIS       EQU   30H
          END
```

Appendix C. TRS-80 LEVEL I Memory Map

ADDRESS DECIMAL	ADDRESS HEXADECIMAL	CONTENTS
0	0000	LEVEL I BASIC ROM
4095	0FFF	LEVEL I BASIC ROM
4096	1000	Not Used
14335	37FF	
14336	3800	Keyboard Memory
15359	3BFF	
15360	3C00	Video Monitor Memory
16383	3FFF	
RAM MEMORY BEGINS		
16384	4000	Begin Level I Scratchpad Storage
16488	4068	Pointer to Current Cursor Position (LSB)
16489	4069	Pointer to Current Cursor Position (MSB)
16490	406A	Pointer to End of RAM Memory (LSB)
16491	406B	Pointer to End of RAM Memory (LSB)
16492	406C	Pointer to Beginning of Free MEM (LSB)
16493	406D	Pointer to Beginning of Free MEM (MSB) Preset to 42001
16528	4090	I/O Status Byte
16894	41FF	End of Level I Scratchpad Storage
16895	4200	BASIC Program Storage
20479	4FFF	End of 4K Memory
32767	7FFF	End of 16K Memory
49151	BFFF	End of 32K Memory
65535	FFFF	End of 48K Memory

Appendix D. LEVEL II Memory Map

ADDRESS DECIMAL	ADDRESS HEXADECIMAL	CONTENTS
0	0000	
		LEVEL II BASIC ROM
12288	3000	
		Reserved
14302	37DE	Communication Status Address
14303	37DF	Communication Data Address
14304	37E0	Interrupt Latch Address
14305	37E1	Disk Drive Select Latch Address
14308	37E4	Cassette Select Latch Address
14312	37E8	Line Printer Address
14316	37EC	Floppy Disk Controller Address
14336	3800	
		TRS-80 Keyboard Memory
15360	3000	
		TRS-80 CRT Video Memory
16383	3FFF	
16384	4000	
		LEVEL II BASIC Fixed RAM
		Vectors (RST'S 1 Through 7)
16402	4012	
16405	4015	Keyboard Device Control Block
		DCB + 0 = DCB Type
		+ 1 = Driver Address
		+ 2 = Driver Address
		+ 3 = 0
		+ 4 = 0
		+ 5 = 0
		+ 6 = 'K'
		+ 7 = 'I'
16413	401D	Video Display Control Block
		DCB + 0 = DCB Type
		+ 1 = Driver Address (LSB)
		+ 2 = Driver Address (MSB)
		+ 3 = Cursor POS N (LSB)
		+ 4 = Cursor POS N (MSB)
		+ 5 = Cursor Character
		+ 6 = 'D'
		+ 7 = 'O'
16421	4025	Line Printer Control Block
		DCB + 0 = DCB Type
		+ 1 = Driver Address (LSB)
		+ 2 = Driver Address (MSB)
		+ 3 = Line/Page
		+ 4 = Line Counter
		+ 5 = 0
		+ 6 = 'P'
		+ 7 = 'R'


16429
402D
Reserved
16463
404F
16464
4050
FDC Interrupt Vector
16466
4052
Communications Interrupt Vector
16468
4054
16476
405C
Reserved
16478
405E
25 MSec Heartbeat Interrupt
16512
4080
Reserved
LEVEL II BASIC Free RAM
Reserved
16870
41E6
I/O Buffer
17127
42E7
17128
42E8
Always Zero
17129
42E9
Program Text
Simple Variables
Arrays
String Variable Names and Overhead
Free Memory
Stack
String Space
Space Reserved For Machine Language Routines – If Memory Size Set, during Initialization Dialog
20479
4FFF
End of 4K Memory
32767
7FFF
End of 16K Memory
49151
BFFF
End of 32K Memory
65535
FFFF
End of 48K Memory

Appendix E. ROM Address and Subroutines

LEVEL I BASIC Addresses

KEYBOARD SCAN A-register contains input byte; input byte is displayed at current cursor	WAIT	CALL JR	0B40H Z,WAIT	;SCAN ;Z=1 IF KB CLEAR
DISPLAY BYTE AT CURSOR	PUSH PUSH LD RST POP POP	DE IY A,20H 10H IY DE	;MUST SAVE ; DE & IY ;BYTE TO DISPLAY ;DISPLAY BYTE ;RESTORE ; DE & IY	
TURN ON CASSETTE On board cassette is turned on via remote plug	CALL	0FE9H		
SAVE MEMORY TO CASSETTE Cassette is turned off	CALL LD LD CALL	0FE9H HL,7000H DE,7100H 0F4BH	•TURN ON CASSETTE ;START ADDRESS ;LAST+1 ADDRESS ;SAVE IT	
LOAD MEMORY FROM CASSETTE On return HL = last + 1 address Z = 0 if checksum error Z = 1 if checksum OK Cassette is turned off	CALL	0EF4H	;TURN ON & READ	
RETURN TO LEVEL I BASIC	Press JP JP	RESET 0 01C9H	 ;POWER UP ;RE-ENTRY WITH READY	
RETURN TO TBUG	Set a Breakpoint to next opcode address. JP	40B1H	;RE-ENTER TBUG	

LEVEL II BASIC Addresses

```
TURN ON CURSOR          PUSH    DE          ;MUST SAVE
    CHARACTER           PUSH    IY          ;  DE & IY
                        LD      A,0EH       ;0EH IS CURSOR BYTE
                        CALL    33H         ;DISPLAY ROUTINE
                        POP     IY          ;RESTORE
                        POP     DE          ;  DE & IY

KEYBOARD SCAN       AGN PUSH    DE          ;MUST SAVE
    A register contains byte when loop      PUSH    IY          ;  DE & IY
    falls through.      CALL    2BH         ;SCAN ROUTINE
    Byte is not displayed on screen!        OR      A           ;A=0 IF KB CLEAR
                        JR      Z,AGN       ;BRANCH IF NO BYTE
                        POP     IY          ;RESTORE
                        POP     DE          ;  DE&IY

DISPLAY BYTE            PUSH    DE          ;MUST SAVE
    AT CURSOR           PUSH    IY          ;  DE & IY
                        LD      A,20H       ;BYTE TO DISPLAY
                        CALL    33H         ;DISPLAY
                        POP     IY          ;RESTORE
                        POP     DE          ;   DE & IY

DEFINE DRIVE        ;A-REGISTER SPECIFIES CASSETTE (0 OR 1)
                        LD      A,0         ;ON BOARD
                    CASSETTE
                        CALL    0212H       ;DEFINE DRIVE

WRITE LEADER            CALL    0287H
    AND SYNC BYTE

TURN OFF                CALL    01F8H
    CASSETTE

SAVE MEMORY             LD      A,0         ;ON BOARD
    TO CASSETTE     CASSETTE
    USER must CALL 264H often   CALL    0212H       ;DEFINE DRIVE
    enough to keep up with 500 baud.    CALL    0287H       ;WRITE LEADER
    Timing is automatic.        LD      A,20H       ;BYTE TO RECORD
                                CALL    0264H       ;OUTPUT BYTE
                                CALL    01F8H       ;CASSETTE OFF

LOOK FOR LEADER         CALL    0296H
    AND SYNC BYTE
```

LOAD MEMORY
FROM CASSETTE
User must CALL 0235H often enough to keep up with 500 baud. User must do own checksum if desired. A-register contains byte read. The user must turn off the Cassette (CALL 01F8H) when all bytes have been read.

```
LD      A,0
CALL    0212H   ;DEFINE DRIVE
CALL    0296H   ;FIND SYNC BYTE
CALL    0235H   ;READ ONE BYTE
```

RETURN TO
LEVEL II BASIC

```
Press   RESET
JP      0       ;LIKE POWER UP
JP      1A19H   ;RE-ENTRY
```

RETURN TO TBUG
(UNDER LEVEL II BASIC)

Set a Breakpoint to next opcode address.

```
JP      43A0H   ;RE-ENTER TBUG
```

OUTPUT TO LINE PRINTER
(LEVEL II ONLY)

```
                ;PUT ASCII BYTE IN
                ;A-REGISTER AND CALL PRTOUT
                ;BUSY CONDITION TESTED FOR

PRTOUT  EXX             ;SAVE REGS.
        LD      HL,37E8H    ;LOAD LP POINTER IN HL
PRTLP8  LD      D,(HL)      ;LOAD LP STATUS BYTE
        BIT     7,D         ;IS THE PRINTER BUSY?
        JP      NZ,PRTLP8
        LD      (HL),A      ;OUTPUT BYTE TO PRINTER
        EXX
        RET
```

Appendix F Base Conversions

BINARY	HEX.	DECIMAL
00000000	00	0
00000001	01	1
00000010	02	2
00000011	03	3
00000100	04	4
00000101	05	5
00000110	06	6
00000111	07	7
00001000	08	8
00001001	09	9
00001010	0A	10
00001011	0B	11
00001100	0C	12
00001101	0D	13
00001110	0E	14
00001111	0F	15
00010000	10	16
00011000	18	24
00100000	20	32
00101000	28	40
00110000	30	48
00111000	38	56
01000000	40	64
01001000	48	72
01010000	50	80
01011000	58	88
01100000	60	96
01101000	68	104
01110000	70	112
01111000	78	120
10000000	80	128
10001000	88	136
10011000	90	144
10011000	98	152
10100000	A0	160
10101000	A8	168
10110000	B0	176
10111000	B8	184
11000000	C0	192
11001000	C8	200
11010000	D0	208
11011000	D8	216
11100000	E0	224
11101000	E8	232
11110000	F0	240
11111000	F8	248
11111100	FC	252
11111110	FE	254
11111111	FF	255

Appendix G. Where to Look for More Information

The Z-80 Microcomputer Handbook, by William Barden, Jr. Howard W. Sams & Co., Inc.

Z-80 Assembly Language Programming, A Zilog, Inc., Publication

Z-80 Programming for Logic Design, by Adam Osborne et al. Osborne & Associates, Inc. Berkeley CA

IMPORTANT NOTICE

ALL RADIO SHACK COMPUTER PROGRAMS ARE DISTRIBUTED ON AN "AS IS" BASIS WITHOUT WARRANTY

Radio Shack shall have no liability or responsibility to customer or any other person or entity with respect to any liability, loss or damage caused or alleged to be caused directly or indirectly by computer equipment or programs sold by Radio Shack, including but not limited to any interruption of service, loss of business or anticipatory profits or consequential damages resulting from the use or operation of such computer or computer programs.

NOTE: Good data processing procedure dictates that the user test the program, run and test sample sets of data, and run the system in parallel with the system previously in use for a period of time adequate to insure that results of operation of the computer or program are satisfactory.

RADIO SHACK A DIVISION OF TANDY CORPORATION

U.S.A.: FORT WORTH, TEXAS 76102

CANADA: BARRIE, ONTARIO L4M 4W5

TANDY CORPORATION

AUSTRALIA	BELGIUM	U.K.
280-316 VICTORIA ROAD RYDALMERE, N.S.W. 2116	PARC INDUSTRIEL DE NANINNE 5140 NANINNE	BILSTON ROAD WEDNESBURY WEST MIDLANDS WS10 7JN

PRINTED IN U.S.A.