

# ZEN 80

Z80 Assembly Language Programming System  
for the MSX MICRO-COMPUTER

(C) Copyright 1.984 Avalon Software

(C) Copyright 1.984 Avalon Software

ISBN: 07457-0002-0  
Published by: Kuma Computes Ltd.  
12 Horseshoe Park  
Pangbourne  
Berks RGB 7JW  
Telex 849462  
Tel 07357 4335

Remastered by: Aitor Lekerika  
April, 15th, 2.007  
Contact me: [msx-assembly@hotmail.com](mailto:msx-assembly@hotmail.com)

## ADVICE

---

No part of this manual or programming system may be reproduced by any means without prior written permission of the author or the publisher.

This programming system is supplied in the belief that it operates as specified, but **Kuma Computers Ltd.** (the company shall not be liable in any circumstances whatsoever for any direct or indirect loss or damage to property incurred or suffered by the customer or any other person as a result of any fault or defect in goods or services supplied by the company and in no circumstances shall the Company be liable for consequential damage or loss of profits (whether or not possibility thereof was separately advised to it or reasonably foreseeable) suffered as a result of any such fault or defect or which otherwise arises from the use or performance of such goods or services.

## INTRODUCTION

---

Thank for buying this copy of ZEN. If you have any questions about ZEN then please feel free to write to *Avalon Software* , every enquiry receives a reply. All high level languages have performance limitations, when you need the manisue in speed and flexibitily the answer lies in Assembly Language proگرامing. ZEN provides you with the tools to generate or analyse Z80 Assembly Language programs.

## STARTING UP

---

Unlike BASIC, which is permanently available in ROM, you need to load ZEN from cassette. It is stored on the cassette as a binary file designed to run at address &HAB80. BASIC usually assumes that it has the whole of memory to itself. To ensure that BASIC and ZEN coexist peacefully it is necessary to change the top of memory before loading from cassette. The loading procedure is therefore as follows:

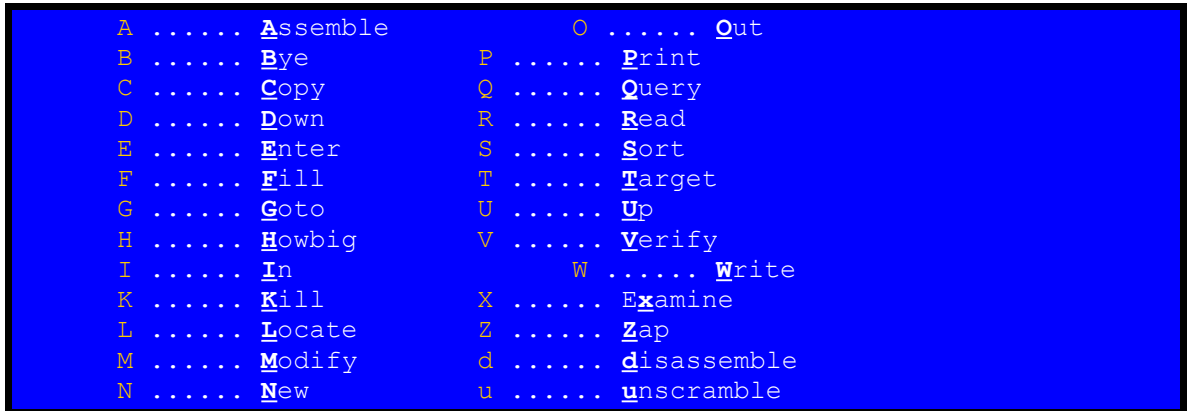
```
CLEAR 200, &H9FFF (RETURN)
BLOAD"ZEN", R      (RETURN)
```

BASIC will then load ZEN into memory and transfer control automatically when it is finished.

**COMMAND LEVEL**

---

Whenever the prompt ZEN> is displayed you are at Command Level, you may execute any of the following commands:

A screenshot of a terminal window with a blue background and white text. It displays a list of commands for the ZEN system, arranged in two columns. Each command is represented by a letter followed by six dots and the full name of the command. The first column contains commands from A to N, and the second column contains commands from O to u. The letters are in a monospaced font, and the command names are in a slightly larger, bolded font.

A .....	<b>A</b> ssemble	O .....	<b>O</b> ut
B .....	<b>B</b> ye	P .....	<b>P</b> rint
C .....	<b>C</b> opy	Q .....	<b>Q</b> uery
D .....	<b>D</b> own	R .....	<b>R</b> ead
E .....	<b>E</b> nter	S .....	<b>S</b> ort
F .....	<b>F</b> ill	T .....	<b>T</b> arget
G .....	<b>G</b> oto	U .....	<b>U</b> p
H .....	<b>H</b> owbig	V .....	<b>V</b> erify
I .....	<b>I</b> n	W .....	<b>W</b> rite
K .....	<b>K</b> ill	X .....	<b>E</b> xamine
L .....	<b>L</b> ocate	Z .....	<b>Z</b> ap
M .....	<b>M</b> odify	d .....	<b>d</b> isassemble
N .....	<b>N</b> ew	u .....	<b>u</b> nscramble

To select a given command type the first letter of it's name, followed by parameter if relevant, and then press the RETURN key. The BS key can be used to BackSpace while DEL will turn the cassette motor on or off. The usage of command loop parameters is explained in greater detail in the next section, wic examines each command in depth. If ZEN doesn't understand anything you've typed in it will display the error message HUH? The default command, just pressing RETURN on it's own, will clear the screen and set it to wide screen text mode. This is the preferred screen mode within ZEN.

**NEW**

---

This command lets you modify the current line of the text file.

The line is displayed with the cursor at the right position. Change the line and press RETURN to restore the new line to the text file.



**OUT**

---

This command will output a data value to the I/O port specified by the command parameter.

You will be prompted for the data parameter.

**PRINT**

---

This command display a number of lines from text file on the screen.

The number of lines is specified by the command parameter, for example P9 (RETURN) would display nine lines. The default command parameter is one.

The display commences with the current line and the lasta line displayed becomes the new current line.

## QUERY

---

This command displays sixty-four bytes of memory in hex and ASCII.

The command parameter specifies the start address, for example QA000H (RETURN) would display the start of ZEN.

If you supply no address parameter the display begins from where it last finished.

## **READ**

---

This command reads a file from cassette into memory, the command parameter specifies the type of file.

### **R (RETURN)**

---

Will read a text file and append it to the end of any text already existing in memory. You will be prompted for a filename. A filename may be from zero to six characters long, if the filename is ZEO characters long the ZEN will load the first text file it finds on the cassette. If the text file reaches ZEN's top of memory limit then reading terminates and error message MEMORI is displayed.

### **RB (RETURN)**

---

Will read a binary file into memory. You will be prompted for an address to LOAD the file at. If you supply an address parameter then the file will be loaded commencing at that address. If you default, by just pressing RETURN, then the file will be loaded at the address defined in the file header. The execution address of the file is placed in the User Program Counter for later execution if required.

Note that the CTRL-STOP key can be used to abort any I/O operation in progress but has no other action within ZEN.

## **GOTO**

---

This command loads the Z80 registers with the User loage and transfer control to the address specified in the command parameter.

For example, GO (RETURN) would perform a complete system cold start.

If no command parameter is supplied the control is transferred to te address in the User Program Counter. You will then be prompted for a breakpoint address. If you respond with a valid address parameter then a breakpoint is set.

A breakpoint is a way of stopping a running program. A Z80 CALL instruction is placed in the program to return control to ZEN trap handler. The trap handler will save all the Z80 registers in the User Image area and restore the code under the breakpoint before returning to the time of the breakpoint.

You can continue execution by using the B (RETURN) command as the Program Counter is saved as part of the trap process. Application programs may terminate with a simple jump back to the ZEN ENTRY or REENTRY points, the only difference between theses two points is that REENTRY saves the Z80 registers while ENTRY doesn't.

Note that ZEN tries to perform as little initialization as possible upon entry to keep everything "ware".

## HOWBIG

---

This command display, in hexadecimal, the start and end addresses of the text file and the top of memory.

ZEN will allow the text file to grow up to this top limit but no futher. You can change this limit if required (see ZEN listing, the LIMIT constant).

## IN

---

This command will display, in hexadecimal and binary, the data read from the I/O port specified by the command parameter.

For example, 98H (RETURN) would read te printer status port.

## **KILL**

---

This command erases the text file, as with the NEW statement in BASIC.

It is possible to recover an accidentally KILLED file as ZEN just makes the EOF pointer equal to the SOF pointer, the actual text will still be in memory. Find the address of the last text character, this will be an ASCII Carriage Return code (&H90). Increase this by one and use the MODIFY command to restore the EOF pointer (see ZEN listing, EOFP).



## **LOCATE**

---

This command is used to search the text file for a particular string of characters.

The characters string forms the command parameter. for example `LBIT 7, A (RETURN)` would find the first occurrence of the string `BIT 7, A` in the text file. The text file is searched from the line after the current line. If the string is found then taht line is made the current line. If the search fails you are at end of file. There are no restrictions on the contents of the parameter string.

## **MODIFY**

---

This command allows you to examine and alter memory contents.

The start address is specified by the command parameter. For example MD000H (RETURN) would cause the command to start at &HD000.

If you supply no address parameter then the command continues from where it last finished. The byte at the address is displayed in hex and ZEN prompts for a data parameter from you.

If you supply a parameter then it is stored at that address, if you default ZEN just steps onto the next address. To return to command level type a full stop.

## ASSEMBLE

---

The function of the assembler is to read a series of assembly language statements and produce the corresponding Z80 machine code and listing.

The ZEN editing commands are used to create a text file in memory, usually called the source file, which is the input to assembler. Output of the machine code file, usually called the object file, is controlled by the LOAD operator (see under PSEUDO-OPS).

The listing output is specified by you in response to the OPTION> prompt from the assembler. You may specify V (RETURN), E (RETURN), P (RETURN) or RETURN for video, external, printer or null list output respectively. The text file is read beginning at the start of file and stopping when the END operator is found.

**BYE**

---

This command gives a warm return to BASIC, any BASIC programs in memory are unaffected.

If you then want to return to ZEN without reloading you can use the USR statement, for example, `DEFUSR =&HA000: A=USR(0)`.

Note that some form of dummy argument is required by BASIC even though it is meaningless. You can shuttle between ZEN and BASIC whenever you like without affecting any files or data in memory.

## COPY

---

This command saves a block of memory.

You will be prompted for START>, STOP> and DESTINATION> parameters. Within ZEN's command structures a numeric parameter may be a decimal, hexadecimal or octal number.

Hex numbers are "H" postfixed and octal are "O" postfixed. So if you wanted to save the block of memory from &H0200 to &H02FF up to &HC000 you type 200H (RETURN), 2FFH (RETURN) and C000H (RETURN).

## DOWN

---

This command moves the editor current line down by the number of lines specified in the command parameter.

For example D37 (RETURN) moves down thirty-seven lines.

The editor in ZEN is line orientated as in BASIC but does not use explicit line numbers, instead you use various commands to move around the text file until you reach the required position. You then use the ENTER or ZAP commands to insert or delete lines of text.

If the down command bumps into the end of file then the message EOF will be displayed.

**ENTER**

---

This command enters lines of text into the text files. ZEN will display the current line number, type in your line of text then press (RETURN).

This process will repeat until you type a full stop as the first character on the line, this returns you to command level. Your text is placed in the file at the current line, the old current and following lines are moved downward towards EOF.

Note that although line numbers are often displayed by ZEN these are dynamically computed and not stored in the text file.

**FILL**

---

This command fills a block of memory, from START> to STOP> inclusive with DATA> value.

You will be prompted for all three parameters.



## **SORT**

---

This command will sort and display the symbol table produced during the last assembly.

You will be prompted for an output option. Your possible responses are the same as for the Assembler list output. The output of this command is generated a page at a time as with list output.

You can restrict the sort process to symbols beginning with a particular letter by entering that letter as a command parameter.

For example, SB (RETURN) would only produce the symbols beginning with the letter "B".

Note that symbols are only sorted on the first letter and not the whole name.

## TARGET

---

This command will move you to any line in the text file and make it current line.

The command parameter specifies the line number, for example T1435 (RETURN) would move you to line one thousand four hundred and thirty-five.

The default command parameter, T (RETURN), moves you to start of file.

**UP**

---

This command moves you up the text file by the number of lines specified in the command parameter.

The default parameter is one.

## VERIFY

---

This command verifies a file which has just been written to asset, the command parameter specifies the type of file.

This command must be used IMMEDIATELY after a write to work correctly. This is because MSX files have no checksum associated with this. ZEN will retain an internal checksum after a write and compare this with the one computed during the verify command.

### **v** (RETURN)

Will verify a text file, no data is actually read into memory.

### **vB** (RETURN)

Will verify a binary file, no data is actually read into memory.

## WRITE

---

This command writes an area of memory to a cassette, the command parameter specifies the type of file

### w (RETURN)

Will write all the text in memory as an ASCII text file. You will be prompted for a filename as described in the READ command. ZEN text files are standard MSX text files (as generated by the BASIC SAVE command) with a CR, LF between lines and a CTRL-Z end of file mark.

### wB (RETURN)

Will write an area of memory as a binary file. You will be prompted for START>, STOP> and EXEC> addresses. The start and top addresses define the area of memory (inclusive) to be written. The load address defines the address at which the file will be loaded back in at its start address. The exec address defines the file's execution (entry) address.

**EXAMINE**

---

This command displays the Z80 registers saved in the User Image.

The top line shows the main registers and the lower line the Z80 alternate registers set.

## ZAP

---

This command removes a number of lines from the text file as specified by the command parameter.

For example, Z108 (RETURN) would remove one hundred and eight lines, commencing with the current line.

The default parameter is one.

## DISASSEMBLE

---

This command performs a symbolic disassembly on an area of memory and generates a text file or listing as output.

You will be prompted for the START> and STOP> addresses inclusive of the area you will to disassemble. You will then be asked the address which the program runs at>.

Sometimes you may have a program in memory at a different location to its usual run-time location, the disassembler can relocate any addresses and labels in its output to reflect this. If you default to request for the run-time start address then ZEN assumes that the program is at its normal run-time location. If you supply an actual address parameter then the output file will reflect this run-time address. You will then be asked repeatedly, for the START> and STOP> addresses inclusive of any data areas within the disassembly region. These areas which will not be decoded as instructions but as data bytes.

To terminate this process type in a stop address of zero. There is a maximum of sixty-four separate data areas, if you exceed this number ZEN will generate the error message FULL. You will now be asked for an output OPTION>.

You may specify V (RETURN), P (RETURN) or E (RETURN) for listings to the video, printer or external devices respectively. If you default then ZEN will generate a text file grows up to the top of memory limit during disassembly then the error message MEMORY is issued and disassembly terminates.

The only other error condition possible during disassembly is for the symbol table to fill up in which case the error message FULL is issued.

Note that the disassembler uses the same symbol table as the assembler and so destroys any symbols there. This is only of relevance if you wish to perform a later SORT operation.

Any illegal opcodes encountered during disassembly are treated as data statements.

Labels of the form Lnnnn (where nnnn is an address) will be generated at the appropriate positions if possible.



## **UNSCRAMBLE**

---

This command is a simplified version of the disassembler.

It will disassemble eight Z80 instructions beginning at the address specified by the command parameter.

For example, u0 (RETURN) will disassemble the start of the BASIC ROM.

If you default on the address parameter then the command continues from where it last finished.

Any illegal opcodes encountered are displayed as data bytes. ZEN will try to make an intelligent guess about how to display eight bit numeric operands. Numbers less than ten are displayed as single digit decimals. Numbers from &H41 to &H5A and &H61 to &H7A are displayed as ASCII literal characters. Other numbers are displayed as hex values with a leading zero if necessary.

## LIST OUTPUT

---

The commands Assemble, Sort and disassemble all generate large quantities of output to the video, printer or external devices. With these commands the output will be generated a page at a time with a short pause between each page. Pressing any key will stop output at the end of the page, to restart press any key except "Q". This key will force the command to QUIT and return to the command loop.

The printer and external devices are assumed to be eighty characters wide by sixty-six lines long, for example, a typical printer. If you have something different then you will need to modify ZEN. You can change the page length by modifying the PAGE procedure (see ZEN listing). You can change the various field widths by modifying the group of constants COMWIDTH/SYMWIDTH. The first byte of each of these constants pairs defines an external/printer device field width, the second defines a video device field width. You may also change the number of symbols per line procedure during a SORT, as there is a switch in the code specifically for this purpose.

The external/printer devices are presumed to respond to the ASCII control characters Forefeed (&H0C), Carriage Return (&H0D) and Linefeed (&H0A). ZEN issues a Forefeed followed by sixty-two lines of text for each page, each line being terminated by Carriage Return, Linefeed. The external device printer is set up to output to the MSX RS-232 device, while the printer driver outputs to the Centronics port. The drivers handle EPSON FX-89 type printers as they stand. If you have something unusual there is space in the driver to insert patches, to filter Linefeeds for example.

The video device is assumed to be thirty-seven characters wide but this can be changed, as for the printer and external devices, if an eighty character device becomes available. Note that line numbers are not generated on the video device for Assemble/Disassembler listings because of this reduced width. The symbol, operand and comment fields of a Z80 statement may be of indefinite length. If necessary ZEN will truncate these fields to fit into the required format.

## THE SYMBOL TABLE

---

The symbol table is the area of memory used by ZEN to store symbols during Assemble/Disassemble. It is situated between ZEN and the text file. If you wish to increase it's size it is only necessary to change the star of file pointer to the requiered hex value, here's how:

- (1) KILL the text file
- (2) Use MODIFY to change SOFP
- (3) KILL the text file again to copu SOFP into EOFP and CURRENT
- (4) Perform an ASSEMBLE to shut down the symbol table
- (5) Use WB to wrte the new verion to cassette.

Note that ZEN is a completely "soft" program, any changes you make wil be reflected in the new version.

## ASSEMBLER SYNTAX

---

ZEN expects assembly language statements to be constructed according to the syntax defined in the Zilog Z80 Assembly Language Programming Manual. ZEN deviates from standard in one instance in that it expects EX AF, AF rather than EX AF, AF'. The section following this one contains an alphabetically sorted listing of the entire Z80 instruction set. Each assembly language statement may be divided into a maximum of four logical fields, they are:

### **LABEL**

---

A label is a way of marking a statement so that other statements can refer to it. Line numbers serve the same purpose in BASIC, you would use GOTO 240, for example. Assembly Language allows you to use a symbolic name for a label. When you declare the label it must be postfixed with a colon ":" so that the assembler knows that it's a label. A label must begin with a letter but may contain letters or digits after that. ZEN allows labels of any length with all characters being significant. The register and condition-code names may not be used as symbols as these are reserved identifiers. Any attempt to do so will result in an error message.

### **OPERATOR**

---

There are sixty-seven operators in the Z80 Assembly Language. In addition ZEN supports seven PSEUDO-OPS, they are:

### **END**

---

This Pseudo-Op terminates assembly, it MUST be used.

### **DS / DEFS**

---

Define Storage skips over the number of object locations specified by the operand.

### **DW / DEFW**

---

Define Word places the operand in the object file in reverse order as required by the Z80 word instructions.

### **DB / DEFB**

---

Define Byte(s) places the operand(s) in the object file at successive locations. Operands are delimited by commas, each operand may be an expression with value less than 256 or may be a literal string. Literal strings may be of any length but cannot form part of an expression.

**EQU**

---

Equate assigns the value of the operand to a symbolic identifier. Any symbolic identifiers used in the operand expression must already be known to the assembler. This "no forward reference" rule is designed to prevent circular referencing.

**ORG**

---

Origin defines the start address of the object file. This Pseudo-Op can be used as often as needed to produce sections of code at different addresses. The "no forward reference" rule applies to the operand.

**LOAD**

---

Commences loading code into memory at the operand address. Use of a subsequent ORG Pseudo-Op will turn this process off, you are explicitly required to re-establish the loading process.

**OPERANDS**

---

The number of operands in a statement depends upon the operator. There are niladic, monadic and dyadic operators in the Z80 instructions set. These take zero, one and two operands respectively. There are three classes of operands:

## Registers

A, B, C, D, E, H, L, I, R, HL, DE, AF, IX, IY, SP

## Condition-codes

NZ, Z, NC, C, PO, PE, P, M

## Numeric expressions

A numeric expression is composed of one or more of the following elements delimited by the infix math operators:

- A decimal, hex or octal number. Decimal is the default base with hex numbers being "h" postfix and octal "o" postfix. Numbers must begin with a digit, a leading zero will be needed with some hex numbers.
- A literal character enclosed in single or double quotes.
- The "\$" character. This variable will be used to use the associated value in evaluating the expression.

The infix math operators are:

+	Addition
-	Substraction
*	Multiplication
/	Division
&	Logical AND
·	Logical OR

Expressions are evaluated STRICTLY LEFT TO RIGHT with no precedence ordering. Arithmetic is sixteen bit unsigned interger and overflow will be ignored.

### COMMENTS

---

Comments are ignored by the assembler. They begin with a semi-colon ";" and are terminated by the end of the line.

## ASSEMBLER ERROR HANDLING

---

If the assembler finds a syntax error the following will happen:

- (1) Assembly terminates.
- (2) An error message is displayed.
- (3) The offending line is displayed and is made the editor current line.
- (4) The command loop is re-entered.

You can now correct the error and re-assemble. It is impossible to make a syntax error which will damage ZEN or anything in memory. The error messages are:

### UNDEFINED

---

You have used a undeclared symbol

### SYMBOL

---

You have declared a zero length symbol or have forgotten the symbol needed with an EQU Pseudo-Op.

### RESERVED

---

You have tried to use a reserved word for a symbol.

### FULL

---

The symbol table is full.

### DOUBLE SYMBOL

---

You have declared the same symbol more than once.

### EOF

---

You have forgotten END and have hit end of file.

### HUH ?

---

The assembler is completely baffled.

### OPERAND

---

You have done something wrong with an operand, this covers a multitude of signs. Most types of syntax error will come under this heading as well as errors of magnitude. These occur when you try to offset too far with a relative jump or indexing instruction.





**END OF FILE**

If you see errors in this document, please, contact me at:

[msx-assembler@hotmail.com](mailto:msx-assembler@hotmail.com)

I'll try to fix the errors immediatly and you have a peace of heaven in this document. Thank you!.