ZEN

Z80 Assembly Language Programming System

for

the

MSX MICRO-COMPUTER

ISBN 07457-0002-0

Published by:- Kuma Computers Ltd.,
12 Horseshoe Park,
Pangbourne,
Berks RG8 7JW

Telex 849462 Tel 07357 4335

M9I ZEN Reference Manual

Introduction

Thank you 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 maximum in speed and flexibility the answer lies in Assembly Language programming. 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 A000H. 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:

(1) Type clear 200,&H9FFF(RETURN)

(2) Type bload"ZEN",R(RETURN)

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

Command level

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

A ..... Assemble	O ..... Out
B ..... Bye	P ..... Print
C ..... Copy	Q ..... Query
D ..... Down	R ..... Read
E ..... Enter	S ..... Sort
F ..... Fill	T ..... Target
G ..... Goto	U ..... Up
H ..... Howbig	V ..... Verify
I ..... In	W ..... Write
K ..... Kill	X ..... Xamine
L ..... Locate	Z ..... Zap
M ..... Modify	d ..... disassemble
N ..... New	u ..... unscramble

To select a given command type in the first letter of it's name, followed by a 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, which 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 rightmost 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 displays a number of lines from the 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 last 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 QA8000H(RETURN) would display the start of ZEN. If you supply no address parameter then 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 zero characters long then 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 the error message MEMORY 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 Image and transfers control to the address specified in the command parameter. For example G0(RETURN) would perform a complete system cold start. If no command parameter is supplied then control is transferred to the 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 at that address. If you default, by just pressing (RETURN), then no 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 the 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 ZEN command loop. You can thus examine the state of the Z80 at the time of the breakpoint. You can continue execution by using the G(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 these two points is that REENTRY saves the Z80 registers while ENTRY doesn't. Note that ZEN tries to perform as little initialisation as possible upon entry to keep everything 'warm'.

Howbig This command displays, 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 further. 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 I98H(RETURN) would read the 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 (0DH). 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 character 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 that 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 D000H. 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 to 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 the 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. The null output option is much the fastest mode (the assembler is peripheral-bound) and should be used until all syntax errors are corrected. 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 re-loading you can use the USR statement, i.e. 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 moves a block of memory. You will be prompted for START>, STOP> and DESTINATION> parameters. Within ZEN's command structure 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 move the block of memory from 200H to 2FFH up to C000H you would 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 default command parameter is one so D(RETURN) moves down one line. 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 file. 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 downwards 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 a 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 the 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 the 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 cassette, 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 them. 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>, LOAD>, and EXEC> addresses. The start and stop addresses define the area of memory (inclusive) to be written. The load address defines the address at which the file will load back in when READ or BLOADed, if you default then the file will be loaded back in at its start address. The exec address defines the file's execution (entry) address.

Xamine 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 register set.

Zap This command removes a number of lines from the text file as specified by the command parameter. For example Z100(RETURN) would remove one hundred and eight lines, commencing with the current line. The default command 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 wish 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 it's usual run-time location, the disassembler can relocate any addresses and labels in it's output to reflect this. If you default to the request for the run-time start address then ZEN assumes that the program is at it's 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 are 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 F(RETURN) for listings to the video, printer or external devices. If you default then ZEN will generate a text file and add it to the end of any text already in memory. If the 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 which 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 from 41H to 5AH and 61H to 7AH are displayed as ASCII literal characters. Other numbers are displayed as hex values with a leading zero if necessary.

MSX ZEN Reference Manual

Further Information

List Output

The commands Assemble, Sort and disassemble can 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 i.e 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 constant 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 produced 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 Formfeed (0CH), Carriage Return (0DH) and Linefeed (0AH). ZEN issues a Formfeed followed by sixty-two lines of text for each page, each line being terminated by Carriage Return, Linefeed. The external device driver is set up to output to the MSX RS232 device, while the printer driver outputs to the Centronics port. The drivers handle EPSON FX-80 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 Assembler/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 Assembly/disassembly. It is situated between ZEN and the text file. If you wish to increase it's size it is only necessary to change the start-of-file pointer to the required new value, here's how: (1) KILL the text file (2) Use MODIFY to change SOFP (3) KILL the text file again to copy SOFP into EOFP and CURRENT (4) Perform an ASSEMBLE to shut down the symbol table (5) Use WB to write the new version to cassette. Note that ZEN is a completely 'soft' program, any changes you make will 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 the 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:

(1) Label
(2) Operator
(3) Operands
(4) Comment

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

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

DB or 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 instruction set. These take zero, one and two operands respectively. There are three classes of operand:

(1) Registers (A,B,C,D,E,H,L,I,R,HL,DE,BC,AF,IX,IY,SP)

(2) Condition-codes (NZ,Z,NC,C,PO,PE,P,M)

(3) Numeric expressions

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

(1) A decimal, hex or octal number. Decimal is the default base with hex numbers being 'H' postfixed and octal 'O' postfixed. Numbers must begin with a digit, a leading zero will be needed with some hex numbers.

(2) A literal character enclosed in single or double quotes.

(3) The $ character. This variable mimics the program counter of the run-time program.

(4) A symbolic name. The assembler will use the associated value in evaluating the expression.

The infix math operators are:

+ addition
- subtraction
* multiplication
/ division
& logical AND
. logical OR

Expressions are evaluated STRICTLY LEFT TO RIGHT with no precedence ordering. Arithmetic is sixteen bit unsigned integer 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-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 an 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.

ORG! You have forgotten ORG.

HUH? The assembler is completely baffled.

OPERAND You have done something wrong with an operand, this covers a multitude of sins! 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.