NAS - SYS
ASSEMBLER

D000 - DFFF

TABLE OF CONTENTS

APPENDICES

Preface

This manual is laid out in two complementary parts.

Section 1 to 3 describe the ZEAP package informally and are designed to be read in order.

The appendices following provide a useful reference section, and define all the elements of ZEAP formally, directing the user to the appropriate section in the first half of the manual where more information and examples are to be found.

Those familiar with the workings of computer assemblers and BASIC-type line editors may find it easier to read the appendices first, although this is not recommended to those who do not fully understand the terms used.

The reader should not be dismayed, however, ZEAP is easy to use and yet powerfull enough for his requirements.

If you require information or guidance on the Z80 Assembly Language itself, you are advised to consult the Mostek or Zilog Z80 Assembly Language manual. Other publications which may prove helpful include

The Z80 Microcomputer Handbool by Wiliam Barden
(Published by Howard W Sams & Co, Inc)

Z80 Instruction Handbook by Nat Wadsworth
(Published by Scientific Computer Consultants Inc.)

Z80 Programming for Logic Design by Adam Osborne et al
(published by Osborne & Associates Inc.)

Z80 Assembly Language Language Programming by Lance A Leventhal
(Published by Osborne & Associates Inc.)

Z80 Microprocessor -Programming & Interfacing by E A Nichols et al
(Published by Howard W Sams & Co, Inc)

Z80 Software Gourmet Guide & Cookbook
(Published by Scelbi Publications Inc.)

Practical Microcomputer Programming -The Z80 by W J Weller
(Northern Technology Books)

Notation

The following notation is used in this manual

ccccH	hexadecimal
(x)	x is optional
(x)...	x is optional and may be repeated indefinitely

1. Introduction

ZEAP 2.0 (Z80 Editor-Assembler Package version 2.0) is a memory resident text editor and symbolic assembler designed for use with the NASCOM 1 and NASCOM 2 microcomputers, running under NAS-SYS.

The same facilities are provided for users of the NASBUG family of monitors by ZEAP 1.2. The differences are explained in Appendix J.

The assembler translates mnemonic codes as defined in the Z80 microcode language into executable machine instructions, allowing user control over memory allocation, and symbolic names for MPU registers and instruction or data addresses. It incoporates comprehensive syntax checking and error message generation, and allows object code to be generated on cassette tape or stored directly in memory.

The editor allows for entry, examination, correction and permanent storage of source programs which are held in memory during editing and assembly.

The memory resident nature of ZEAP allows entry, assembly, testing, correction ane re-assembly of source programs without the necessity of using cassette tape at any stage, since editor, assembler, source program and object program may reside in memory simultaneously. This makes ZEAP very easy and quick to use.

1.1 Aims of ZEAP

ZEAP was produced with the intention of providing a compact editor-assembler package for the NASCOM 1 microcomputer. The following requirements were laid down during the design of the package:

* Minimum memory requirements
* Minimum extra hardware requirements
* Maximum compatibility with existing assemblers
* Ability to edit, assemble, execute and then re-edit the program with the minimum use of external storage (e.g. cassette tape)
* Ability to store source programs on cassette tape and then re-load them at a later stage
* Ability to store more than one source program at a time in memory
* Maximum use of monitor subroutines
* Ability to drive an ASCII terminal attached to the UART
* Ability to generate the monitor's "L" or "R" commands.

The result is an editor/assembler package requiring 8K bytes of user RAM, of which ZEAP uses 4K program space and 256 bytes data storage, leaving a minimum 4K byte spare for source programs and object code.

The ZEAP editor provides the following functions:

* Fully dynamic source buffer allocation
* Insertion, deletion and replacement of lines
* String searching
* Automatic line number generation for block entry of source programs
* Complete resequencing of source program line numbers
* Loading and dumping of source programs to and from cassette tape
* Listing of selected source program lines on the screen or on an ASCII terminal
* Page mode of operation
* Control of display/listing speed
* Recovery of source program in case of accident cold-start
* Automatic checksum for easy detection of hardware faults or user program malfunction

The ZEAP assember provides the following functions:

* Full range of options including control of source listing, object generation and error processing
* Numbered error messages pin-pointing the cause of the error
* Object generation onto cassette tape, or directly to memory
* Direct execution of object program from ZEAP itself
* Listing of lines containing errors
* Formatted source listing on the screen or on an ASCII terminal
* Optional Sorted Symbol Table

The editor, assembler, source program and optional object program may all reside in memory at the same time, enabling maximum ease of entry, assembly, testing, correction and re-assembly of source programs with minimum use of external storage.

The assembler source code follows closely that defined in the ZILOG assembler, the differences being noted in section 1.2.

Editor operation is described in section 2, while the assembler's function is defined in section 3.

1.2 Comparison with the ZILOG Assembler

The operation of the ZEAP assembler is very similar in most respects to the ZILOG Z80 assembler. The following differences should be noted, however:

- Expressions may contain only the operators "+" and "-", and no parathetical grouping is allowed. Expressions may be enclosed in parenthesis to respresent memory addresses. Evaluation is from left to right. A leading "-" is allowed.

- Hexadecimal numbers are denoted by a suffix "H", and the first hexadecimal digit must be 0-9. Alternatively, a prefix "#" may be used, in which case the "H" must not be present, and the first digit may be any of 0-F (this facility is provided for compatibility with previous versions of ZEAP). The two representations may be used interchangeably at all the times. The default number base is decimal, Octal and binary numbers are not supported.

- Labels must begin is the first column of the source line, directly after the single space following the sequence number. Only one label is permitted on a line. The use of a ":" suffix to indicate a label is not supported. Statements without labels must leave the first column blank, except to comments, which may begin in the first column with a ";".

- The following assembler directives (pseudo-ops) are not supported

```
MACRO
ENDM
COND
ENDC
DEFL
END
```

- Two additional assembler directives are provided

```
ENT
SKIP
```

  See Appendix D for more details.

- A single ASCII character code may be included in an expression by preceding it with a double sign, e.g. "A = 41H. This facility replaces the DEFB 's' assembler directive.

- Fields and/or expressions may be separated by one or more spaces and/or commas. The space and the comma are syntactilly equivalent in all contexts within the assembly language.

- Two additional instruction code mnemonics are provides. These are

```
RCAL label
SCAL code
```

These mnemonics codes generate RST instructions to the relative call and subroutine call routines provided in NAS-SYS. The operand of an RCAL instruction has the same definition as that for a relative jump instruction (see Appendix E, option 20). The operand of an SCAL instruction is the subroutine code for the subroutine to be invoked (see and NAS-SYS documentation). For example

```
    RCAL XXX
    .
    .
    .
XXX PUSH AF     ; SUBROUTINE
```

and

```
IN  EQU  62H
    .
    .
    .
    SCAL IN     ; CALL IN
```

1.3 Machine requirements

ZEAP occupies just under 4K bytes and requires 256 byte workspace. Thus a minimum of 8K bytes of memory is required.

With a cassette recorder the user can store source programs on cassette tape for reloding at a later time. The assembler can output object code to tape which can be subsequently loaded using the L or R commands provided in NAS-SYS.

ZEAP contains routines to drive an ASCII terminal attached to the UART for hard copy or source listings. However, this item is entirely optional and ZEAP will functions perfectly without it.

The minimum system is

A working NASCOM 1 or NASCOM 2
A television or monitor
A minimum of 8K bytes of memory
A cassette recorder

2. The ZEAP editor

The ZEAP editor provides the means by which source programs my be entered, examined and altered by the user.

2.1 Editor operation

After ZEAP has been loaded, control is passed to the editor as described in APPENDIX A.

The editor prompt, a flashing cursor, will be displayed on the bottom line of the VDU indicating that the ZEAP editor is ready to accept editor commands.

The editor is a line editor in which source lines are identified by line numbers (sequence numbers), each line of source code being identified with a unique number. Powerful context editing capabilities are available through the NAS-SYS cursor control functions.

A sequence number may be any decimal number from 1 to 9999. Leading zeros may be omitted. The sequence number is always followed by a single space to separate it from the actual source line, e.g.

```
  1000 SAMPLE LINE
```

The actual line is "SAMPLE LINE". The source line itself may of cause contain leading spaces, e.g.

```
  2000   ANOTHER LINE
```

The space after "2000" is the separator, but the next two spaces are part of the source line, which is thus " ANOTHER LINE".

A line of source code may be entered by typing a sequence number, followed by a space, followed by the source line, followed by the ENTER key. The editor stores the line of source code in memory and prompts for the next editor command.

The source program is sorted automatically in ascending sequence number order. Thus

```
  20 THIS IS THE THIRD LINE
  10 THIS IS THE FIRST LINE
  12 THIS IS THE SECOND LINE
```

would cause the lines to be stored in the order indicated.

Typing a sequence number directly followed by ENTER causes that line to be deleted. Thus

```
  12
```

would cause line 12 to be deleted.

Typing the sequence number of a line which alredy exists followed by new source code. Thus

```
  20 THIS IS NOW THE SECOND LINE
```

would cause line 20 to be replaced with the indicated text.

Thus all requirements for inserting, deleting and changing lines of source code are provided by the above techniques.

In addition the the above facilities, there are a number of commands for examing and manipulating the source program. These commands have been implemented with single letter mnemonic codes, and are described below in section 2.2.

All source lines are stored in an area of memory called the EDIT BUFFER. All editor commands operate on the information contained in this Buffer. The size of the source program is limited only by the amount of memory available.

At all times during ZEAP operation, except during assembly, the top line of the screen contains the following display

```
     ZEAP m.n/cccc  Free=pppp  End=qqqq  Ent=rrrr
```

where

m.n   is the version number (e.g. 2.0)

cccc is the copy number

pppp is the address of the first unused location after the EDIT BUFFER

qqqq is one more than the address of the last byte of the object program generated by the last assembly

rrrr is the adress of the entry point of the object program generated by the last assembly (if any)

The value pppp includes the area used for the symbol value table, which resides in memory just above the EDIT BUFFER. ZEAP ensures that this value does not exceed 5000H (the limit with a 16K extention), or the value set in the E command for the last cold-start (see Appendix A). This value is also used as the default origin setting for the assembler.

Any time before the ENTER key is depressed, a line may be edited using all the screen facilities available under NAS-SYS.

At any time when ZEAP is in process of displaying information (e.g. when listing or assembling the source program) the user may interrupt the process by depressing the ESC key (or SHIFT + ENTER keys). ZEAP will immediately abandon its current processing and display the editor prompt to indicate that it is ready to process editor commands.

Any other key may be used under the same circumstances to temporarily hold the execution of ZEAP so that the contents of the screen can be examined at length. When the user wishes to resume execution, depressing any key except ESC will restart ZEAP where it left off, and processing will continue.

Error messages from the ZEAP editor are of the form

```
  ERROR nn
```

where nn is the error number. An explanation of ZEAP error codes is given in Appendix B. The common editor message is

```
  ERROR 99
```

meaning that the last line of user input was illegal or unrecognisable as an editor command or line of source code.

If the first character of an input line is blank, the line is ignored by the editor.

2.2 ZEAP editor commands

The following discussion is independent of any knowledge of the Z80 assembly language, and therefore the source lines shown are not suitable for assembly by the ZEAP assembler.

"V" Suppose the following lines are entered

```
  20 LINE 2
  10 LINE 1
  30 LINE 3
```

The user can examine the contents of part or all of the Edit Buffer using the "V" editor command. ("V" is a mnemonic for VDU List). Thus

```
  V 10 10
  0010 LINE 1
  V 10 20
  0010 LINE 1
  0020 LINE 2
  V 20
  0020 LINE 2
  0030 LINE 3
  V
  0010 LINE 1
  0020 LINE 2
  0030 LINE 3
```

Also note

```
  V 5 15
  0010 LINE 1
  V 1 9
  V 20 10
  V 1000
```

The last three commands cause no display.

In summary

V m n	Display lines m to n inclusive
V m	Display lines from m to the end of the buffer
V	Display the entire contents of the source buffer

The space following "V" is optional, but if both m and n are specified, they must be separated by one or more spaces.

"U" When a source program has been entered by the user using the ZEAP editor, is is useful to be able to store all or part of it on cassette tape. This is archieved by the "U" editor command ("U" is a mnemonic for UART List). Its syntax is the same as that of the "V" command. Its operation is identical except that each line displayed is also output to the UART in a format which allows the line to be reloaded subsequently by the editor. Thus

```
U
0010 LINE 1
0020 LINE 2
0030 LINE 3
```

would cause those line displayed to be stored on an attached cassette recorder.

There is no identifiable Load command provided with ZEAP. Loading of source programs stored on tape using the "U" editor command is performed simply by switching the cassette recorder on while the editor prompt is displayed. ZEAP scans both the keyboard and the UART input during editor operation, and so source lines input from tape will be interpreted as if they had been entered manually. Thus playing back the above tape when the ZEAP editor prompt is displayed would cause the following display

```
0010 LINE 1
0020 LINE 2
0030 LINE 3
```

and the three lines would be entered into the edit buffer as if they had been typed on the keyboard.

If the user attaches an ASCII terminal (teletype or equivalent) to the UART, the "U" editor command can be used to obtain hard copy of all or part of the source program. Thus, with an attached ASCII terminal

```
U 10 20
0010 LINE 1
0020 LINE 2
```

and the two lines displayed are printed on the terminal. For each character printed, bit 7, the parity bit, is always zero.

"I" The ZEAP editor provides a convenient facility for the manual entry of blocks of source code, namely the "I" editor command ("I" is a mnemonic for Auto Input). If the user enters

```
I 40
```

the editor responds

```
0040
```

and any input up to the ENTER key is interpreted as line 40. Suppose the following is typed

```
0040 LINE 4           (ENTER pressed)
0050
```

After ENTER is depressed the editor encrements the sequence number by 10 and displays the new sequence number, ready for the entry of the next line of code, and so on

```
  0050 LINE 5
  0060 LINE 6
  0070
```

Note that the necessary space following the sequence number is inserted by ZEAP, so that the user need not type it.

It is possible to edit the sequence number using the cursor control keys. Thus now typing three cursor-lefts, followed by 95, followed a space results in the display

```
  0095
```

and then line 95 could be entered

```
  0095 LINE 7
  0105
```

Note that the increment of 10 is applied to the sequence number of the last line entered, and not of the last line displayed by ZEAP.

Exit from Auto Input mode (which is the name given to the above behaviour) is achieved by typing ESC (SHIFT ENTER) which deletes the current line and causes the usual editor prompt to be displayed.

If the number after the "I" is omitted, the editor displays

```
  0010
```

initially.

If a second number is typed after the "I", it is used as the sequence number increment. It must be less than 100. Thus

```
  I 100 3
  0100          (ENTER pressed)
  0103          (ENTER pressed)
                (ESC pressed)
```

So in summary

```
  I             Enter Auto Input mode at line 10 with increments of 10
  I s           Enter Auto Input mode at line s with increments of 10
  I s i         Enter Auto Input mode at line s with increments of i
```

"X" Deleting a block of source code is made easier by the "X" editor command ("X" is a mnemonic for eXpunge). "X" must always be followed by two numbers, separated by a space, which are the sequence numbers of the first and the last lines to be deleted. All lines between and including these lines are deleted. Thus

```
  V
  0010 LINE 1
  0020 LINE 2
```

```
0030 LINE 3
0040 LINE 4
0050 LINE 5
0060 LINE 6
0095 LINE 7
X 36 70
V
0010 LINE 1
0020 LINE 2
0030 LINE 3
0095 LINE 7
X 95
    ERROR 99
X 95 95
V
0010 LINE 1
0020 LINE 2
0030 LINE 3
```

Note that an attempt to use X with only one line number produced an error message.

To delete the entire edit buffer, the user should enter

```
X 1 9999
```

This command does the job of a NEW or CLEAR utility in similar editors.

In summary

```
X m n         Delete lines m to n inclusive
```

"Z" The "Z" editor command merely presents a source line for editing using the NAS-SYS cursor control facilities. Typing "Z" followed by a line number will display that line on the screen and position the cursor over the first character of the source line. The line number may also be edited, so that a line can be moved to another place in the source program. Note that the original line will not be automatically deleted. - this must be done by the user. If no line number is given, the last line output by the editor is presented for editing. Thus, after a FIND has been performed, only "Z" beed be entered to edit the displayed line.

In summary

```
Z y           edit line y
Z             edit last line displayed
```

"F" The "F" editor command ("F" is a mnemonic for Find) enables the user to find the first and thereafter subsequent occurrences of any string which will fit on one line in the source program. Thus

```
25 ABC
```

```
55 ABCDEF
V
0010 LINE 1
0020 LINE2
0025 ABC
0030 LINE 3
0055 ABCDEF
0070 LINE 4
F/ABCD/
0055 ABCDEF
```

In this example the string "ABCD" is found in line 55, which is displayed. The "/" character is used as a delimiter. Any non-blank character may be used.

```
0055 ABCDEF
F+ABC+
0025 ABC
F
0055 ABCDEF
FT
0025 ABC
```

The command "F" above causes the next occurrence of the last mentioned string to be found. The command "FT" (a mnemonic for Find from the Top) causes the search to be restarted from the beginnning of the Edit Buffer. If no occurrence of the string is found, the editor merely prompts for the next line of input.

After an "F" editor command has been issued and executed, the user may continue to do successive "F" operations on the same string by typing ENTER only. This step-repeat feature is similar to the "S" command step-repeat provided in the NAS-SYS monitor. Each ENTER typed will find the next ocurrence of the string in the source program.

In summary

F/string/	finds first occurrence of "string"
F	finds next occurrence of last "string"
FT	finds first occurrence of last "string"
ENTER	finds next occurrence of last "string"

"/" The "/" editor command may be used to search for a string from a specified line in the edit buffer. The argument is a single line number of the first line to be searched. Thus

```
F/LINE/
0010 LINE 1
/ 40
0070 LINE 4
/ 20
0020 LINE 2
/
0010 LINE 1
```

ZEAP searches for the last string specified in an "F/string/" editor command from the line specified.

In summary

/ y	find first occurrence of the last "string" after line y
/	find first occurrence of "string"

Note that a "/" on its own has the same effect as an "FT" command, described above.

"R" The "R" editor command ("R" is a mnemonic for Resequence) allows the entire source program to be renumbered. Thus

```
V
0010 LINE 1
0020 LINE 2
0025 ABC
0030 LINE 3
0055 ABCDEF
0070 LINE 4
R 100
V
0100 LINE 1
0110 LINE 2
0120 ABC
0130 LINE 3
0140 ABCDEF
0150 LINE 4
```

Only the order of the source lines is maintained. The first line is given the line number entered after the "R", and subsequent lines are numbered sequentially in increments of 10. The arguments are the same as for the "I" editor command.

In summary

R	Resequence from sequence number 10 in increments of 10
R s	Resequence from sequence number s in increments of 10
R s i	Resequence from sequence number s in increments of i

"H" The "H" editor command sets the page size for page mode operation. Normally ZEAP displays information on the screen continuously until the output is finished or until the user holds or interrupts the output (see 2.1). By typing

```
H 10
```

the user can instruct ZEAP to pause every 10 lines of output so that the contents of the screen can be read. Any key can be depressed to resume output.

For example, entering

```
  H 4
  V
```

displays

```
  0100 LINE 1
  0110 LINE 2
  0120 ABC
  0130 LINE 3
```

and the display pauses. Hitting any key except ESC causes the display to continue

```
  0140 ABCDEF
  0150 LINE 4
```

and the editor prompt is displayed again.

If no page size is specified, a default of 15 is assumed (the size of the NASCOM screen). To remove the page mode of operation, typing

```
  H 0
```

will signal ZEAP to return to continuous output.

In summary

H p	Set page size to p
H	Set page size to 15
H 0	Restore to continuous output

"J" The "J" editor command sets the delay at the end of each line of output to the VDU, and therefore controls the display speed. Entering

```
  J 120
```

sets an end of line delay of about 1 second (at 2 MHz). Entering

```
  J
```

resets the delay to zero.

In summary

J p	Set end of line delay proportional to p ('p' to be < 256)
J	Reset delay to zero

"K" The "K" editor command is identical to the "J" command, except that it controls the delay for output to the UART (during a "U" command or an assembly with the TTY option set).

"Y" The "Y" editor command allows the EDIT BUFFER to be recovered after an accidental cold start under certain circumstances. Its format is

```
  Y
```

and it must be entered before any lines of source program are entered after a cold start. Provided the original information in the EDIT BUFFER has not been corrupted, the source program will be recovered in full.

The command may not work if the origin for the source buffer has been changed since the last cold start (see Appendix A).

"P" The "P" editor command allows object code generated by the assembler under the MEMORY option to be placed at a physical address different from the logical address of the assembly, to facilitate generation of ROM based programs. A single hexadecimal argument must be supplied (the default is zero) which specifies the amount to be added to the logical address to obtain the physical address where the object code is to be stored. Thus

```
  P 4000
```

will cause the following program to be placed physically at location 4000H.

```
        ORG     0
        JP      START
XX      DEFS    30
        etc.
```

Note that the object code is only stored in memory if the MEMORY assembler option is on. Object code stored in memory with any non-zero offset is unsuitable for direct execution. It must first be moved to the logical address of the assembly.

"Q" The "Q" editor command is provided for compatibility with earlier versions of ZEAP, and has been superceded by the "J" and "K" commands. It allows both the rate at which information is displayed on the screen, and the pause at the end of a line of listing sent to the UART, to be controlled. The format is

```
  Q ccdd
```

where ccdd is a 4 digit hexadecimal number (with no space between cc and dd), and cc is the delay to be inserted at the end of each line displayed on the VDU, and dd the delay to be inserted after a carriage return when either the "U" editor command or the "TTY" assembly option is in operation. A value of 0 signifies no delay. A value of 1 signifies a delay of about seven and a half milliseconds, and so on.

"N" The "N" editor command returns control to NAS-SYS ("N" is a mnemonic NAS-SYS). The NAS-SYS working storage and reflections are restored. ZEAP can be re-entered by following the procedure described in APPENDIX A, at which point the editor prompt will be displayed.

```
"O"     See section 3.
"+"
"-"
"A"
"W"
"G"
```

A formal account of the editor commands is given in APPENDIX C.

3. The ZEAP Assembler

The ZEAP assembler translates the source program, entered by the user into the Edit Buffer using the ZEAP editor, into executable Z80 microcode instructions which may be stored in memory for immediate execution, or on tape for subsequent use.

The assembler scans the source program twice, outputting error messages on the first pass, and a full assembler listing and the object code on the second. Various assembler options control the precise activities on each pass.

During assembly, the top line of the screen is blanked.

3.1 Assembler Operation

"A" The assembler is entered from the editor by using the editor command "A" ("A" is a mnemonic for Assemble). Since the portion of the Edit Buffer to be assembled can be selected in the same way as for the "V" and "U" editor commands, it is possible to store several source programs in the Edit Buffer simulaneously, provided that each occupies a continuous block in the Edit Buffer, i.e. prgrams do not overlap.

Suppose a complete program is stored in the Edit Buffer in lines 2000 to 2999. The command

```
A 2000 2999
```

will cause assembly of this program. If only one program is stored, simply entering

```
A
```

will assemble all lines in the source program. Similarly

```
A 5000
```

would assemble from line 5000 to the end of the edit buffer.

During assembly the top line of the CRT is used as a temporary working area, but is restored to the usual ZEAP header on return to the editor.

When the assembly is complete, and all output is finished, control is returned to the ZEAP editor, and the editor prompt is displayed ready for the next command.

In summary

```
A m n        assembles from line m to n inclusive
A m          assembles from line m to the end of the Edit Buffer
A            assembles the entire Edit Buffer contents
```

"W" The "W" editor command is identical in operation to the "A" editor command, except that only those lines containing errors are output in the assembly listing.

"G" The "G" editor command ("G" is a mnemonic for Go) causes control to be passed to the object program produced in the last assembly, provided that an ENT assembler directive was present in the source program. If the source program has been altered since the last assembly, the object program may have been overwritten, so that the "G" command must be used with caution.

3.2 Expressions

Wherever the form "exp" is encountered an expression involving label symbols and/or constants is expected. The occurrence of register and/or label symbols must be in accordance with the semantics laid down in APPENDIX F. Such an expression is always evaluated using 16 bit integer two's complement arithmetic. Expressions may be formed using the following elements

label symbol

a symbolic name of one to six characters, starting with a letter or "$" and thereafter consisting of letters, "$"s and/or numbers, which appears in the label field of a source program. The value of the symbol is that associated with it by its appearance in the label field of some source statements (see section 3.3).

decimal integer constant

a decimal number between 0 and 65535. Larger numbers will be truncated to 16 bits.

hexadecimal integer constants

one to four (five with a leading zero) hexadecimal digits (0-9, A-F), the first of which is a decimal digit (0-9), followed by an "H", or a "#" sign followed by one to four hexadecimal digits (0-9, A-F), interpreted as an unsigned hexadecimal number. Larger numbers will be truncated to 16 bits.

ASCII code value

a double quote character followed by a single character, whose ASCII code value is used (bit 7=0).

Location counter

the symbol "$" which represents the value of the location counter at the beginning of assembly of the current line (or current expression in the case of a DEFB or DEFW assembler direction; see section 3.5). This is the address at which the current instruction (or expression) is being assembled.

Any number of elements of the above kind may be combined with "+" and "-" signs to make the expression. A leading "-" sign is allowed. No parenthetical grouping is allowed. Expressions may be enclosed entirely in parentheses to represent memory addresses, in accordance with the semantics defined in APPENDIX F.

Here are some examples

```
        TABLE+3
        START-$
        80H+"A-1
        END-BEG+1
        -273
        "Z-"A+1
        BIMUNZ+BIMUNZ
```

Expressions may not contain embedded blanks or commans. A missing operator is interpreted as a "+". A missing operand is interpreted as a zero. For example

```
        12ABC   is interpreted as       12+ABC
        3+-4    is interpreted as       3+0\-4
```

3.3 Source Statement Syntax

Each line of a source program must be one of the following

(i) a Z80 instruction
(ii) a ZEAP assembler directive
(iii) a comment

The first character of a source line is the character directly after the single space following the sequence number. The last character of a source line is the last non-blank character typed before the ENTER key is pressed.

3.3.1 Labels

If the source line is type (i) or (ii), an optional label may be present. The label must be a symbolic name of one to six characters starting in the first column, the first character being a letter or "$", and subsequent characters being letters, "$"'s or numbers with no embedded spaces. Examples are

```
  START
  END
  TABLE
  L1
  P3B
  $OUT
  A$
```

The following symbols are improperly formed

```
  113P
  P4:
  LP Q
```

A label must be followed by one or more spaces and/or commas. If present, it must start in the first column of the source statement (i.e. the first character of the label must be the first character of the source line). If no label is present the first character of the source line must be a space or a comma, unless the statement is type (iii), a comment.

In case (i), the label is given the value of the location counter prior to the assembly of the rest of the statement, i.e. its value is the address at which the statement is assembled. In this way the location of any instruction or sequence of instructions can be represented symbolically and referred to elsewhere in the program, e.g. in a JP or CALL instruction.

In case (iii), the label is given the value as defined in section 3.5 and Appendix D. In this way the address of a data table or literal string for display on the screen can be represented symbolically and referred to elsewhere in the program, e.g. in a LD HL,exp instruction.

Each label defined must be unique within the program being assembled. Each label symbol referrenced in the program must appear in the label field of some source statement. The following symbolic names are reserved by ZEAP for registers and condition codes

```
  A, B, C, D, E, H, I, L, M, P, R, Z
  AF, BC, DE, HL, IX, IY, NC, NZ, PE, PO, SP
```

3.3.2 Instruction Format

Each source statement of type (i) or (ii) consists of up to four fields which are

(optional)	label field
	instruction mnemonic or assembler directive
(optional)	operand field
(optional)	comment field

Each field must be separated from the next by one or more spaces and/or commas. If the first character of the source line is a space or a comma, no label is assumed to be present. If the first character of the source line is ";" the line is assumed to be a comment (see section 3.3.3 below).

The instruction mnemonic or assembler directive must be present. It may be any mnemonic listed in Appendix F, or any directive mnemonic documented in section 3.5 and Appendix F, or an RCAL or SCAL instruction as described in section 1.2.

The operand field may or may not be present according to the syntax of the statement. In case (i) it must follow the definition given for the appropriate instruction in Appendix F. In case (ii), it must follow the definition given for the appropriate assembler directive in section 3.5 and Appendix D. If the field contains more than one operand, each operand must be separated from the next by one or more spaces and/or commas.

The comment field is optional. It must begin with ";" and ends at the end of the line. Any characters after the ";" are ignored by the assembler, except that they are reproduced literally in the assembly listing. The ";" may follow directly after the preceding field, with no intervening spaces or commas.

Although the assembler interprets the entire operand field, only the first 17 characters of the operand and/or comment fields are displayed on the assembly listing on the screen. For this reason it is suggested that the full assembly listing is complete. The operand field itself will rarely if ever need to be longer than 17 characters.

3.3.3 Comment Lines

A comment line must begin with a ";", and all characters thereafter will be ignored by the assembler, except that tey will appear on the assembly listing. The first 29 Characters will be displayed on the assembly listing on the screen.

3.4 Assembler Options

"O" The "O" editor command allows various options to be set which define the output required from the assembler ("O" is a mnemonic for Options). The "O" may be followed by a single hexadecimal mask defining which options are ON and which are OFF. This mask is obtained by adding up the option code of those options desired ON. Thus

```
O 1A
```

would set assembler options MEMORY, TAPE and PASS 2 on, and all other options OFF (1A = 10 + 08 + 02 Hex). If no number follows the "O" all assembler options are set to the default values (i.e. all off).

In summary

```
O x         set assembler options from mask x
O           set all assembler options OFF
```

"+" The "+" editor command is identical to the "O" editor command, except that each option specified is turned ON and all other options remain at their previous settings. Thus

```
+ C0
```

would switch the symbols and format options ON, and leave all other options as they were. If no hexadecimal number is supplied, there is no effect.

"-" The "-" editor command is identical to the "O" editor command, except that each options specified is turned OFF and all other options remain at their previous settings. Thus

```
- 2
```

would switch the MEMORY option OFF, and leave all other options as they were. If no hexadecimal number is supplied, there is no effect.

Appendix E contains a full account of each assembler option.

3.5 Assembler Directives

The eight assembler directives supported by ZEAP give the user the ability to control the generation of object code addresses, to generate tables or literal strings, to split up the output, and to denote an entry point for the program.

DEFB, DEFW, DEFM all cause the generation of object code for one or more bytes, words (double-bytes) and ASCII characters respectively.

EQU allows the direct assignment of an expression value to a symbolic name.

ORG and DEFS alter the assembly address ("$") so that assembler programs may be assembles at any address, and to allow for space for storage of intermediate results and other variable information.

SKIP leaves a single blank line in the assembly listing.

ENT specifies the entry point of the program.

A full account of the assembler directives is given in Appendix D.

3.6 Assembly Listing

The assembly listing is formatted with a single header for the listing, and a header for the symbol table (if required).

A line of assembly listing takes the following form

```
  aaaa cccccccc ssss bbbbbb mmmm pppppppppppppppppp
```

The explanation of the fields is as follows

aaaa — 4 digit hexadecimal address of the instruction being assembled except in a DEFB, DEFW or DEFM assembler directive, where it is the address of the first byte of code generated.

cccccccc — 2 to 8 hexadecimal digits representing the object code for the instruction, or the value of the expression in an EQU or DEFS assembler directive, or the first 4 bytes generated from a DEFB, DEFW or DEFM assembler directive, in which case subsequent bytes are displayed an successive lines, up to 4 per line.

ssss — 4 digits sequence number of the current source line

bbbbbb — 1 to 6 character label of the current source line. If no label is present, this field id left blank.

mmmm — 2 to 4 character instruction mnemonic or assembler directive.

ppp.... — Operand and comment field directly from source line.

If the source line is a comment (first character ";"), field aaaa and cccccccc are left blank, and the comment is copied directly after the sequence number.

If the line contains an error, field ccccccc will contain

```
  ERROR nn
```

and no object code is generated, A TRUNCATION error or a LABEL NOT FOUND error is reported on the following line, but the object generation is not suppressed.

Since the assembler formats the listing, there is no need to tablulate source programs. The fields of each source statement will be correctly formatted by the assembler. For example the source line

```
  0040 BIM LD A,1
```

would appear in the assembly listing as

```
  aaaa 3E 01     0040 BIM    LD   A,1
```

where aaaa is the current value of the location counter ("$").

3.7 Object Generation

3.7.1 Tape Object

If the TAPE assembler option is on, object code is output through the UART to an attached cassette recorder. When object code is being generared for re-loading with the "L" command (see option 40 in Appendix E) any block in which the number of bytes generated is not exact multiple of eight is padded out with random data. Provided the object code is generated in strict address order this will cause no trouble to the user.

Object code can be produced in one of two formats, suitable for loading under NAS-SYS using the "L" command and the "R" command, respectively. Assembler option 40 chooses between the two formats (see Appendix E).

The tape LED is used by ZEAP in the same way as it is by NAS-SYS, and may be used as a direct or indirect indication to start the cassette recorder as described in the NASCOM documentation.

3.7.2 Memory Object

If the MEMORY assembler option is on, object code is assembled direct to memory. Object instructions and data are written as they are assembled to the appropriate memory address.

ZEAP will not allow you to overwrite the edit buffer, ZEAP's data area, or ZEAP itself. An error message will be generated if this is attempted. It is thus impossible to corrupt ZEAP from within itself. The NAS-SYS data area is, however, unprotected, so that care must be taken not to overwrite it.

If no ORG assembler directive appears in the source program, assembly will begin at the first available byte of RAM not beeing used by ZEAP, as displayed at the top of the screen, but the user should bear in mind that the object program may overflow available memory with no warning.

A program so assembled may be executed by entering NAS-SYS using the ZEAP "N" editor command and executing the object code using NAS-SYS "E" command, or by using the "G" editor command in conjunction with the ENT assembler directive (see section 3.1). The object program should set the stack pointer to a free area of memory if the stack is to be used, so that ZEAP's own stack does not overflow.

If the object program works incorrectly it may be necessary to re-load ZEAP from tape, and enter the source program again. For this reason it is recommended that the source program is saved on tape before testing an object program, in case valuable data is lost and has to typed in again.

Appendix A

ZEAP Operation

ZEAP should be loaded in accordance with the procedure specified in the documentation provided with the ZEAP tape/EPROM. EPROM execution addresses are D000 and D003 respectively, with the text buffer at 1000. The description below refers to tape operations, but is the same for the EPROM version with relevant address changes.

ZEAP occupies memory from 1000H to just under 2000H (4K bytes). The area from 0F00H to 0FFFH is used for data storage and stack space. In default, the edit buffer resides directly above ZEAP, and extends continuously upward through memory.

There are two points in ZEAP. Executing at 1000H produces a cold start, and causes ZEAP to initialise its working storage and buffer space, empting the edit buffer abd setting all assembler options OFF. Executing at 1003H produces a warm start. The edit buffer is preserved and the data is not initialised. A warm start should only be performed if the data area and the edit buffer are intact.

One or two parameters may optionally be specified when a cold starting ZEAP. Entering

```
  E 1000 3000
```

sets the limit of the edit buffer to 3000H. That is, ZEAP will not write into memory above that address. The default limit is 5000H. Also, entering

```
  E 1000 4000 3000
```

sets the limit to 4000H, and the beginning of the edit buffer at 3000H, so that ZEAP will only write into the area between 3000H and 3FFFH inclusive. The default edit buffer origin is 2000H.

It is desired to use ZEAP with an edit buffer which has been read in from tape outside ZEAP control, for example by the NAS-SYS tape read function, ZEAP should be entered as follows.

```
  E 1000
or
  E 1000 <top>
or
  E 1000 <top> <buffer>
```

Now leave ZEAP via the 'N' command, read the tape under NAS-SYS, and make a warm start back to ZEAP. ZEAP will now use the edit buffer that has been loaded. See also Appendix G.

Appendix B

ZEAP Error Codes

ERROR 00 CORE FULL

The source line just entered would cause an overflow of the edit buffer. No change is made to the edit buffer.

ERROR 01 RESEQUENCE OVERFLOW

During the execution of a RESEQUENCE editor command the line number became greater than 9999. The source file is resequnced starting with line 1 in steps of 1.

ERROR 02 AUTO INPUT OVERFLOW

In AUTO-INPUT mode the line number became grater than 9999. AUTO-INPUT mode is abandoned.

ERROR 03 NON-EXISTENT LINE

An attempt was made to edit a non-existent line with the "Z" editor command.

ERROR 04 ILLEGAL GO

A "G" editor command was issued but the following conditions were not satisfied

MEMORY option ON
offset is zero
ENT directive in source program

ERROR 05 RECOVERY IMPOSSIBLE

The "Y" editor command was entered with the edit buffer not empty. The edit buffer is unchanged, and the recovery of the previous contents is impossible.

ERROR 06 ZERO INCREMENT

The increment for an "I" or an "R" editor command was zero. The command is ignored.

ERROR 10 UNRECOGNISABLE STATEMENT

A label is more than 6 characters, or a mnemonic is more than 4 characters or omitted. The statement is ignored.

ERROR 20 UNKNOWN MNEMONIC

The op-code field contains an unrecognisable mnemonic. The statement is ignored.

ERROR 21 CONTEXT ERROR

The combination of op-code and operand types encountered is illegal or the mnemonic is too short. The statement is ignored.

ERROR 22 INDEX REGISTER ERROR

IX or IY is used where only HL is permitted, or in a JP (IX) or JP (IY) instruction, the displacement is non-zero. The statement is ignored.

ERROR 23 TRUNCATION ERROR

An 8 bit operand is greater than 255 or less than -128 or an index register displacement value is greater than 127 or less than -128, or a relative branch offset is greater than 129 or less than -126, or a bit number in a BIT, SET, RES instruction is greater than 7 or less than 0, or an address in an RST instruction is illegal, or the mode in an IM instruction is not 0, 1, or 2. The value in question is truncated and the assembly of the statement continues.

ERROR 24 TOO MAY REGISTERS

A register symbol appears in an assembler directive operand, or more than one register appears in an instruction operand. The statement is ignored.

ERROR 25 REGISTER MSISMATCHED

The combination of first and second types is illegal. The statement is ignored.

ERROR 26 ILLEGAL CHARACTER

The operand field contains a character whose meaning is unassigned in the syntax of the assembly language. The statement is ignored.

ERROR 27 ILLEGAL OPERAND

The combination of a register and a label or constant in this context is illegal. The statement is ignored.

ERROR 28 PARENTHESIS ERROR

A left parenthesis occurs in an assembler directive operand, or more than one left parenthesis occurs in an instruction operand. The statement is ignored.

ERROR 30 LABEL NOT FOUND

A symbol in an expression does not occur in the label field of any statement in the source code. The label is treated as having value zero and assembly continues.

ERROR 31 LABEL REDEFINED

The symbol in the label field has previously appeared in a label field, or is a register name. The statement is ignored.

ERROR 40 DIRECTIVE ERROR

In an assembler directive, too few or too many operands appear. The statement is ignored.

ERROR 41 ILLEGAL FORWARD REFERENCE

A label symbol in an EQU, ORG or DEFS assembler directive is defined after the directive is encountered. The statemet is ignored.

ERROR 50 ERRORS IN ASSEMBLY

There were errors flagged in the preceding assembly.

ERROR 60 MEMORY OBJECT CODE OVERWRITTEN

a byte of object code generated by the present source line would overwrite ZEAP or the edit buffer if it were to be stored. This and subsequent bytes generated by the assembler are not stored in memory. This message appears once only, indicating the position of the first such byte.

ERROR 61 OVERWRITE ERROR

An ERROR 60 was flagged in the preceeding assembly.

ERROR 90 CHECKSUM ERROR

Part of ZEAP has been corrupted due to hardware errors or user tampering. If ZEAP is not reloaded, unpredictable errord may occur.

ERROR 99 ILLEGAL COMMAND

An unrecognisable editor command or an ill-formed source code line was entered. The input line is ignored.

Appendix C

ZEAP Editor Commands

The following symbols are used. All numbers are decimal unless otherwise stated.

y	sequence number (i.e. source line number)
m	first sequence number to which command is applied (default 1)
n	last sequence number to which command is applied (default 9999)
s	starting sequence number (default 10)
i	increment (default 10)
x	hexadecimal option mask
h	hexadecimal number
p	decimal count

Numbers are separated from the command letter and from each other by one or more spaces.

If n is explicitly specifies then m must be also. If i is explicitly specified the s must be also.

A m n ASSEMBLE SOURCE PROGRAM (ASSEMBLE)

Causes assembly of the indicated portion of the source program. See section 3 for more details.

F/string/
F
FT FIND STRING (FIND)

Searches for a specifies string in the edit buffer, and displays it if found.

The form "F/string/" is used to search from the beginning of the edit buffer for a character string of any length. The "/" represents a delimiter character, which may be any character, except space, but which must follow directly after the "F". If the second delimiter is omitted the command is treated as an "FT" command (described below). If the string is not found the ZEAP editor prompts for the next command.

The form "F" is used to search for the string specified in the most recent "F/string/" command, starting from the line after the line in which the last ocurrence of the string was found, instead of the from the beginning of the edit buffer. Otherwise it is identical to the "F/string/" command described above.

The form "FT" is used to search for the string specified in the most recent "F/string/" command, starting from the beginning of the edit buffer. Otherwise it is identical to the "F/string/" command described above.

/ y FIND STRING FROM LINE (FIND FROM)

Searches for the last "string" given in an "F/string/" editor command, starting the search from line y. If no line number is specified, the search starts from the beginning of the edit buffer.

G GO TO OBJECT PROGRAM (GO)

Executes the object program at the addess of the last ENT assembler directive encountered in the last assembly, provided

there was an ENT assembler directive present
the offset is zero (set by the "P" editor command)
the MEMORY assembler option is ON

H p SET PAGE MODE (PAGE)

Sets the page size (i.e. the number of lines displayed before pausing) to p. If p is zero, restores continuous output. If p is omitted, it defaults to 15.

I s i ENTER AUTO-INPUT MODE (AUTO-INPUT)

Causes the ZEAP editor to enter AUTO-INPUT mode. The number s is displayed, followed by a space. The user may then enter a line of source code terminated by the ENTER key, whereupon that line of code is entered into the edit buffer, i is added to s, and the new sequence number is displayed. The user may continue to enter source code as long as the sequence number remains less than 10000.

Exit from AUTO-INPUT mode is achieved by entering ESC (SHIFT ENTER). The editor then prompts for the next command.

J p SET VDU DELAY (VDU DELAY)

Sets the end of line delay for VDU output proportional to p. If p is omitted, it defaults to zero (no delay).

K p SET UART DELAY (UART DELAY)

Sets the end on line delay for output from the "U" editor command and under the TTY assembly option proportional to p. If p is omitted, it defaults to zero.

N RETURN TO NAS-SYS (NAS-SYS)

Causes ZEAP to return to NAS-SYS. The NAS-SYS registers and reflections are restored.

Provided the area of memory used by ZEAP is unchanged during NAS-SYS operation, ZEAP may be re-entered with the edit buffer intact, in accordance with the procedure described in Appendix A.

O x SET ASSEMBLER OPTIONS (OPTIONS)

Sets assembler options ON or OFF as specified by the hexadecimal number x. The options and their hexadecimal codes are as follows. See section 3.4 for more details.

+ 01	SUPPRESS SOURCE LISTING	(NOLIST)
+ 02	OBJECT CODE TO MEMORY	(MEMORY)
+ 04	SOURCE LISTING TO TTY	(TTY)
+ 08	OBJECT CODE TO TAPE	(TAPE)
+ 10	FORCE SECOND PASS	(PASS2)
+ 20	ADJUST RELATIVE JUMP OFFSETS	(REL)
+ 40	OBJECT TAPE FORMAT	(FORMAT)
+ 80	LIST SYMBOL TABLE	(SYMBOLS)

Initially all options are OFF.

+ x SET ASSEMBLER OPTIONS ON (ON)

The options specified by the hexadecimal number x are switched ON. All other options are unaffected.

- x SET MEMORY OPTIONS OFF (OFF)

The options specified by the hexadecimal number x are switched OFF. All other options are unaffected.

P h SET MEMORY OFFSET (OFFSET)

Set to h the number to be added to the logical assembly address to optain the physical location of the object code in memory when the MEMORY assembler option is ON.

Q h SET END OF LINE DELAYS (DELAYS)

Sets the VDU end of line delay to cc hey and the UART end of line delay to dd hex (see "J" and "K" command above), where h = ccdd.

R s i RESEQUENCE SOURCE CODE (RESEQUENCE)

Renumbers all the statements in the edit buffer so that the first line is given the number s, and subsequent lines s+i, s+2i, etc. as for the "I" editor command.

U m n LISTING TO UART (SAVE)

Causes the indicated portion of the source program to be output to the UART, and simultaneously displayed on the screen.

The output though the UART is formatted to drive either a cassette tape recorder, so that any portion of the source program may be stored permanently and loaded subsequently by ZEAP, or an ASCII terminal to obtain a hard copy listing of any portion of the source program.

V m n LISTING TO VDU (LIST)

Causes the indicated portion of the source program to be displayed on the screen.

W m n LIST ERRORS (ERRORS)

Assembles the indicated portion of the source program forcing the PASS 2 and NOLIST assembler options ON, so that a full list of errors is obtained.

X m n BLOCK DELETE (DELETE)

Causes all source lines numbered m to n inclusive to be deleted. Both m and n must be specified.

Y RECOVER EDIT BUFFER (RECOVER)

After a cold start, recovers previous contents of the edit buffer if no source lines have been entered.

Z y EDIT SOURCE LINE (EDIT)

Displays line y and opens for edit.

Appendix D

Assembler Directives

```
        label   EQU     exp (; comment)                 EQUATE SYMBOL
```

The label is given the value of the 16 bit expression on the operand field. All symbols appearing in the expression must have been previously defined. No object code is generated. The label may not be redefined.

```
        label   ORG     exp (; comment)                 SET ORIGIN
```

The location counter ($) is given the value of the 16 bit expression in the operand field. All symbols appearing in the expression must have been previously defined. No object code is generated. Assembly continues at the new origin. If a label is present, it is given the value of the expression.

```
        (label) DEFS    exp (; comment)                 DEFINE SPACE
```

The location counter ($) is increased by the value of the 16 bit ecxpression in the operand field. All symbols appearing in the expression must have been previously defined. No object code is generated. Assembly continues after a block of memory of length exp. If a label is present, it is given the original value of the location counter ($).

```
        (label) DEFB    exp (,exp)... (; comment)       DEFINE BYTE
```

For each 16 bit expression one byte of code is generated with the value of the expression. Expressions may contain forward references. If a label is present, it is given the value of the address of the first byte of code generated.

```
        (label) DEFW    exp (,exp)... (; comment)       DEFINE WORD
```

For each 16 bit expression two bytes of code are generated with the value of that expression, the low order 8 bits occupying the first byte and the high order 8 bits the second. Expression may contain forward references. If a label is present, it is given the value at the address of the first byte of code generated.

```
        (label) DEFM    /string/ (; comment)            DEFINE MESSAGE
```

The "/" may be any character except blank or comma. For each character after the first delimiter until the second delimiter or the end of the line is encountered, one byte of code is generated having the value of the ASCII code for that character, with bit 7 zero. Any characters may appear between the delimiters. Characters after the second occurence of the delimiter are ignored. If a label is present, it is given the value of the address of the first byte of code generated.

```
        SKIP                                            SKIP LINE
```

Leaves a blank line in the assembly listing.

ENT ENTRY POINT

Marks the entry point of the program. This assembler directive should preceed the first instruction of the program to be executed. The value of the location counter ("$") at the ENT in the last assembly is displayed on the top line during ZEAP operation, and is used as the execution address for the "G" editor command.

Appendix E

Assembler Options

Assembler options are set by he "O", "+" and "-" editor commands. All assembler options are switched OFF at a cold start.

Assembler options are selected by adding the hexadecimal option codes of the desired assembler options together. Thus the TTY and PASS 2 assembler options would be selected as ON by entering the command "+ 14".

+01 SUPPRESS SOURCE LISTING (NOLIST)

During the second pass, no source listing will displayed on the screen. Lines containing errors will, however, still displayed.

+02 OBJECT CODE TO MEMORY (MEMORY)

During the second pass, the object code will be assembled directly into memory.

+04 SOURCE LISTING TO TTY (TTY)

During the second pass, any source listing will be listed on an ASCII terminal attached to the UART. If the NOLIST assembler option is ON, only those lines containing errors will be listed. The output from the UART is not suitable for storage on cassette tape. Each character sent to the terminal has bit 7, the parity bit set to zero.

This assembler option may not used in conjunction with the TAPE assembler option described below.

+08 OBJECT CODE TO TAPE (TAPE)

During the second pass, the object code will be dumped to a cassette tape recorder attachted to the UART.

This assembler option may not used in conjunction with the TTY assembler option, described above.

+10 FORCE SECOND PASS (PASS2)

Normally if errors are detected during the first pass, the second pass is suppressed. If this assembler option in ON, however, the second pass will be executed regardless.

+20 ADJUST RELATIVE JUMP OFFSETS (REL)

Different standards in implementing the JR and similar instructions have been adopted by different manufactures. The assembler normally expects the operand of a relative jump instruction to be the address (or label) to which the jump is to be made. If the REL option in ON, however, the operand will be interpreted as the offset from the start of the jump instruction to the destination.

The convention adopted must be fixed throughout the whole program.

+40 OBJECT TAPE FORMAT (FORMAT)

Object code produced under the TAPE assembler option may be in one of two formats, suitable for loading under NAS-SYS under the "L" or "R" commands, respectively.

When this option is OFF, object code is written in eight byte blocks suitable for loading useing the "L" command.

When this option is ON, object code records of up to sixty for bytes long are written, all but the last of which are numbered "FF", suitable for loading using the "R" command.

+80 LIST SYMMBOL TABLE (SYMBOLS)

An alphabetically sorted symbol table appears at the end of the assembly listing. Each line take the following form

```
ccccH ssss llllll      ccccH ssss llllll
```

where

```
  cccc    is the hexadecimal value of the symbol
  ssss    is the line number at which the symbol is defined
  llllll  is the symbol
```

Appendix F

Z80 Opcode Listing

In Alphabetic Mnemonic Order

8E	ADC A,(HL)
DD 8E 56	ADC A,(IX+N)
FD 8E 56	ADC A,(IY+N)
8F	ADC A,A
88	ADC A,B
89	ADC A,C
8A	ADC A,D
8B	ADC A,E
8C	ADC A,H
8D	ADC A,L
CE 56	ADC A,N
ED 4A	ADC HL,BC
ED 5A	ADC HL,DE
ED 6A	ADC HL,HL
ED 7A	ADC HL,SP
86	ADD A,(HL)
DD 86 56	ADD A,(IX+N)
FD 86 56	ADD A,(IY+N)
87	ADD A,A
80	ADD A,B
81	ADD A,C
82	ADD A,D
83	ADD A,E
84	ADD A,H
85	ADD A,L
C6 56	ADD A,N
09	ADD HL,BC
19	ADD HL,DE
29	ADD HL,HL
39	ADD HL,SP
DD 09	ADD IX,BC
DD 19	ADD IX,DE
DD 29	ADD IX,IX
DD 39	ADD IX,SP
FD 09	ADD IY,BC
FD 19	ADD IY,DE
FD 29	ADD IY,IY
FD 39	ADD IY,SP
A6	AND (HL)
DD A6 56	AND (IX+N)
FD A6 56	AND (IY+N)
A7	AND A
A0	AND B
A1	AND C
A2	AND D
A3	AND E
A4	AND H
A5	AND L
E6 56	AND N
CB 46	BIT 0,(HL)
DD CB 56 46	BIT 0,(IX+N)
FD CB 56 46	BIT 0,(IY+N)
CB 47	BIT 0,A
CB 40	BIT 0,B
CB 41	BIT 0,C
CB 42	BIT 0,D
CB 43	BIT 0,E
CB 44	BIT 0,H
CB 45	BIT 0,L
CB 4E	BIT 1,(HL)
DD CB 56 4E	BIT 1,(IX+N)
FD CB 56 4E	BIT 1,(IY+N)
CB 4F	BIT 1,A
CB 48	BIT 1,B
CB 49	BIT 1,C
CB 4A	BIT 1,D
CB 4B	BIT 1,E
CB 4C	BIT 1,H
CB 4D	BIT 1,L
CB 56	BIT 2,(HL)
DD CB 56 56	BIT 2,(IX+N)
FD CB 56 56	BIT 2,(IY+N)
CB 57	BIT 2,A
CB 50	BIT 2,B
CB 51	BIT 2,C
CB 52	BIT 2,D
CB 53	BIT 2,E
CB 54	BIT 2,H
CB 55	BIT 2,L
CB 5E	BIT 3,(HL)
DD CB 56 5E	BIT 3,(IX+N)
FD CB 56 5E	BIT 3,(IY+N)
CB 5F	BIT 3,A
CB 58	BIT 3,B
CB 59	BIT 3,C
CB 5A	BIT 3,D
CB 5B	BIT 3,E
CB 5C	BIT 3,H
CB 5D	BIT 3,L
CB 66	BIT 4,(HL)
DD CB 56 66	BIT 4,(IX+N)
FD CB 56 66	BIT 4,(IY+N)
CB 67	BIT 4,A
CB 60	BIT 4,B
CB 61	BIT 4,C
CB 62	BIT 4,D
CB 63	BIT 4,E
CB 64	BIT 4,H
CB 65	BIT 4,L
CB 6E	BIT 5,(HL)
DD CB 56 6E	BIT 5,(IX+N)
FD CB 56 6E	BIT 5,(IY+N)
CB 6F	BIT 5,A
CB 68	BIT 5,B

CB 69	BIT 5,C
CB 6A	BIT 5,D
CB 6B	BIT 5,E
CB 6C	BIT 5,H
CB 6D	BIT 5,L
CB 76	BIT 6,(HL)
DD CB 56 76	BIT 6,(IX+N)
FD CB 56 76	BIT 6,(IY+N)
CB 77	BIT 6,A
CB 70	BIT 6,B
CB 71	BIT 6,C
CB 72	BIT 6,D
CB 73	BIT 6,E
CB 74	BIT 6,H
CB 75	BIT 6,L
CB 7E	BIT 7,(HL)
DD CB 56 7E	BIT 7,(IX+N)
FD CB 56 7E	BIT 7,(IY+N)
CB 7F	BIT 7,A
CB 78	BIT 7,B
CB 79	BIT 7,C
CB 7A	BIT 7,D
CB 7B	BIT 7,E
CB 7C	BIT 7,H
CB 7D	BIT 7,L
DC 1234	CALL C,NN
FC 1234	CALL M,NN
D4 1234	CALL NC,NN
D4 1234	CALL NC,NN
CD 1234	CALL NN
C4 1234	CALL NZ,NN
F4 1234	CALL P,NN
EC 1234	CALL PE,NN
E4 1234	CALL PO,NN
CC 1234	CALL Z,NN
3F	CCF
BE	CP (HL)
DD BE 56	CP (IX+N)
FD BE 56	CP (IY+N)
BF	CP A
B8	CP B
B9	CP C
BA	CP D
BB	CP E
BC	CP H
BD	CP L
FE 56	CP N
ED A9	CPD
ED B9	CPDR
ED A1	CPI
ED B1	CPIR
2F	CPL
27	DAA
35	DEC (HL)
DD 35 56	DEC (IX+N)
FD 35 56	DEC (IY+N)
3D	DEC A
05	DEC B
0B	DEC BC
0D	DEC C
15	DEC D
1B	DEC DE
1D	DEC E
25	DEC H
2B	DEC HL
DD 2B	DEC IX
FD 2B	DEC IY
2D	DEC L
3B	DEC SP
F3	DI
10 00	DJNZ $+2
FB	EI
E3	EX (SP),HL
DD E3	EX (SP),IX
FD E3	EX (SP),IY
08	EX AF,AF'
EB	EX DE,HL
D9	EXX
76	HALT
ED 46	IM 0
ED 56	IM 1
ED 5E	IM 2
ED 78	IN A,(C)
DB 56	IN A,(N)
ED 40	IN B,(C)
ED 48	IN C,(C)
ED 50	IN D,(C)
ED 58	IN E,(C)
ED 60	IN H,(C)
ED 68	IN L,(C)
34	INC (HL)
DD 34 56	INC (IX+N)
FD 34 56	INC (IY+N)
3C	INC A
04	INC B
03	INC BC
0C	INC C
14	INC D
13	INC DE
1C	INC E
24	INC H
23	INC HL
DD 23	INC IX
FD 23	INC IY

2C	INC L	22 1234	LD (NN),HL
33	INC SP	DD 22 1234	LD (NN),IX
ED AA	IND	FD 22 1234	LD (NN),IY
ED BA	INDR	ED 73 1234	LD (NN),SP
ED A2	INI	0A	LD A,(BC)
ED B2	INIR	1A	LD A,(DE)
C3 119F	JP $+3	7E	LD A,(HL)
E9	JP (HL)	DD 7E 56	LD A,(IX+N)
DD E9	JP (IX)	FD 7E 56	LD A,(IY+N)
FD E9	JP (IY)	3A 1234	LD A,(NN)
DA 11A7	JP C,$+3	7F	LD A,A
FA 11AA	JP M,$+3	78	LD A,B
D2 11AD	JP NC,$+3	79	LD A,C
C2 11B0	JP NZ,$+3	7A	LD A,D
F2 11B3	JP P,$+3	7B	LD A,E
EA 11B6	JP PE,$+3	7C	LD A,H
E2 11B9	JP PO,$+3	ED 57	LD A,I
CA 11BC	JP Z,$+3	7D	LD A,L
18 00	JR $+2	3E 56	LD A,N
38 00	JR C,$+2	46	LD B,(HL)
30 00	JR NC,$+2	DD 46 56	LD B,(IX+N)
20 00	JR NZ,$+2	FD 46 56	LD B,(IY+N)
28 00	JR Z,$+2	47	LD B,A
02	LD (BC),A	40	LD B,B
12	LD (DE),A	41	LD B,C
77	LD (HL),A	42	LD B,D
70	LD (HL),B	43	LD B,E
71	LD (HL),C	44	LD B,H
72	LD (HL),D	45	LD B,L
73	LD (HL),E	06 56	LD B,N
74	LD (HL),H	ED 4B 1234	LD BC,(NN)
75	LD (HL),L	01 1234	LD BC,NN
36 56	LD (HL),N	4E	LD C,(HL)
DD 77 56	LD (IX+N),A	DD 4E 56	LD C,(IX+N)
DD 70 56	LD (IX+N),B	FD 4E 56	LD C,(IY+N)
DD 71 56	LD (IX+N),C	4F	LD C,A
DD 72 56	LD (IX+N),D	48	LD C,B
DD 73 56	LD (IX+N),E	49	LD C,C
DD 74 56	LD (IX+N),H	4A	LD C,D
DD 75 56	LD (IX+N),L	4B	LD C,E
DD 36 56 56	LD (IX+N),N	4C	LD C,H
FD 77 56	LD (IY+N),A	4D	LD C,L
FD 70 56	LD (IY+N),B	0E 56	LD C,N
FD 71 56	LD (IY+N),C	56	LD D,(HL)
FD 72 56	LD (IY+N),D	DD 56 56	LD D,(IX+N)
FD 73 56	LD (IY+N),E	FD 56 56	LD D,(IY+N)
FD 74 56	LD (IY+N),H	57	LD D,A
FD 75 56	LD (IY+N),L	50	LD D,B
FD 36 56 56	LD (IY+N),N	51	LD D,C
32 1234	LD (NN),A	52	LD D,D
ED 43 1234	LD (NN),BC	53	LD D,E
ED 53 1234	LD (NN),DE	54	LD D,H

55	LD D,L
16 56	LD D,N
ED 5B 1234	LD DE,(NN)
11 1234	LD DE,NN
5E	LD E,(HL)
DD 5E 56	LD E,(IX+N)
FD 5E 56	LD E,(IY+N)
5F	LD E,A
58	LD E,B
59	LD E,C
5A	LD E,D
5B	LD E,E
5C	LD E,H
5D	LD E,L
1E 56	LD E,N
66	LD H,(HL)
DD 66 56	LD H,(IX+N)
FD 66 56	LD H,(IY+N)
67	LD H,A
60	LD H,B
61	LD H,C
62	LD H,D
63	LD H,E
64	LD H,H
65	LD H,L
26 56	LD H,N
2A 1234	LD HL,(NN)
21 1234	LD HL,NN
ED 47	LD I,A
DD 2A 1234	LD IX,(NN)
DD 21 1234	LD IX,NN
FD 2A 1234	LD IY,(NN)
FD 21 1234	LD IY,NN
6E	LD L,(HL)
DD 6E 56	LD L,(IX+N)
FD 6E 56	LD L,(IY+N)
6F	LD L,A
68	LD L,B
69	LD L,C
6A	LD L,D
6B	LD L,E
6C	LD L,H
6D	LD L,L
2E 56	LD L,N
ED 7B 1234	LD SP,(NN)
F9	LD SP,HL
DD F9	LD SP,IX
FD F9	LD SP,IY
31 1234	LD SP,NN
ED A8	LDD
ED B8	LDDR
ED A0	LDI
ED B0	LDIR
ED 44	NEG
00	NOP
B6	OR (HL)
DD B6 56	OR (IX+N)
FD B6 56	OR (IY+N)
B7	OR A
B0	OR B
B1	OR C
B2	OR D
B3	OR E
B4	OR H
B5	OR L
F6 56	OR N
ED BB	OTDR
ED B3	OTIR
ED 79	OUT (C),A
ED 41	OUT (C),B
ED 49	OUT (C),C
ED 51	OUT (C),D
ED 59	OUT (C),E
ED 61	OUT (C),H
ED 69	OUT (C),L
D3 56	OUT (N),A
ED AB	OUTD
ED A3	OUTI
F1	POP AF
C1	POP BC
D1	POP DE
E1	POP HL
DD E1	POP IX
FD E1	POP IY
F5	PUSH AF
C5	PUSH BC
D5	PUSH DE
E5	PUSH HL
DD E5	PUSH IX
FD E5	PUSH IY
CB 86	RES 0,(HL)
DD CB 56 86	RES 0,(IX+N)
FD CB 56 86	RES 0,(IY+N)
CB 87	RES 0,A
CB 80	RES 0,B
CB 81	RES 0,C
CB 82	RES 0,D
CB 83	RES 0,E
CB 84	RES 0,H
CB 85	RES 0,L
CB 8E	RES 1,(HL)
DD CB 56 8E	RES 1,(IX+N)
FD CB 56 8E	RES 1,(IY+N)
CB 8F	RES 1,A

CB 88	RES 1,B
CB 89	RES 1,C
CB 8A	RES 1,D
CB 8B	RES 1,E
CB 8C	RES 1,H
CB 8D	RES 1,L
CB 96	RES 2,(HL)
DD CB 56 96	RES 2,(IX+N)
FD CB 56 96	RES 2,(IY+N)
CB 97	RES 2,A
CB 90	RES 2,B
CB 91	RES 2,C
CB 92	RES 2,D
CB 93	RES 2,E
CB 94	RES 2,H
CB 95	RES 2,L
CB 9E	RES 3,(HL)
DD CB 56 9E	RES 3,(IX+N)
FD CB 56 9E	RES 3,(IY+N)
CB 9F	RES 3,A
CB 98	RES 3,B
CB 99	RES 3,C
CB 9A	RES 3,D
CB 9B	RES 3,E
CB 9C	RES 3,H
CB 9D	RES 3,L
CB A6	RES 4,(HL)
DD CB 56 A6	RES 4,(IX+N)
FD CB 56 A6	RES 4,(IY+N)
CB A7	RES 4,A
CB A0	RES 4,B
CB A1	RES 4,C
CB A2	RES 4,D
CB A3	RES 4,E
CB A4	RES 4,H
CB A5	RES 4,L
CB AE	RES 5,(HL)
DD CB 56 AE	RES 5,(IX+N)
FD CB 56 AE	RES 5,(IY+N)
CB AF	RES 5,A
CB A8	RES 5,B
CB A9	RES 5,C
CB AA	RES 5,D
CB AB	RES 5,E
CB AC	RES 5,H
CB AD	RES 5,L
CB B6	RES 6,(HL)
DD CB 56 B6	RES 6,(IX+N)
FD CB 56 B6	RES 6,(IY+N)
CB B7	RES 6,A
CB B0	RES 6,B
CB B1	RES 6,C
CB B2	RES 6,D
CB B3	RES 6,E
CB B4	RES 6,H
CB B5	RES 6,L
CB BE	RES 7,(HL)
DD CB 56 BE	RES 7,(IX+N)
FD CB 56 BE	RES 7,(IY+N)
CB BF	RES 7,A
CB B8	RES 7,B
CB B9	RES 7,C
CB BA	RES 7,D
CB BB	RES 7,E
CB BC	RES 7,H
CB BD	RES 7,L
C9	RET
D8	RET C
F8	RET M
D0	RET NC
C0	RET NZ
F0	RET P
E8	RET PE
E0	RET PO
C8	RET Z
ED 4D	RETI
ED 45	RETN
CB 16	RL (HL)
CB 17	RL A
CB 10	RL B
CB 11	RL C
CB 12	RL D
CB 13	RL E
CB 14	RL H
CB 15	RL L
DD CB 56 16	RL (IX+N)
FD CB 56 16	RL (IY+N)
17	RLA
CB 06	RLC (HL)
DD CB 56 06	RLC (IX+N)
FD CB 56 06	RLC (IY+N)
CB 07	RLC A
CB 00	RLC B
CB 01	RLC C
CB 02	RLC D
CB 03	RLC E
CB 04	RLC H
CB 05	RLC L
07	RLCA
ED 6F	RLD
CB 1E	RR (HL)
CB 1F	RR A
CB 18	RR B
CB 19	RR C

CB 1A	RR D
CB 1B	RR E
CB 1C	RR H
CB 1D	RR L
DD CB 56 1E	RR (IX+N)
FD CB 56 1E	RR (IY+N)
1F	RRA
CB 0E	RRC (HL)
DD CB 56 0E	RRC (IX+N)
FD CB 56 0E	RRC (IY+N)
CB 0F	RRC A
CB 08	RRC B
CB 09	RRC C
CB 0A	RRC D
CB 0B	RRC E
CB 0C	RRC H
CB 0D	RRC L
0F	RRCA
ED 67	RRD
C7	RST 0
D7	RST 10H
DF	RST 18H
E7	RST 20H
EF	RST 28H
F7	RST 30H
FF	RST 38H
CF	RST 8H
9E	SBC (HL)
9F	SBC A
DD 9E 56	SBC A,(IX+N)
FD 9E 56	SBC A,(IY+N)
DE 56	SBC A,N
98	SBC B
99	SBC C
9A	SBC D
9B	SBC E
9C	SBC H
ED 42	SBC HL,BC
ED 52	SBC HL,DE
ED 62	SBC HL,HL
ED 72	SBC HL,SP
9D	SBC L
37	SCF
CB C6	SET 0,(HL)
DD CB 56 C6	SET 0,(IX+N)
FD CB 56 C6	SET 0,(IY+N)
CB C7	SET 0,A
CB C0	SET 0,B
CB C1	SET 0,C
CB C2	SET 0,D
CB C3	SET 0,E
CB C4	SET 0,H
CB C5	SET 0,L
CB CE	SET 1,(HL)
DD CB 56 CE	SET 1,(IX+N)
FD CB 56 CE	SET 1,(IY+N)
CB CF	SET 1,A
CB C8	SET 1,B
CB C9	SET 1,C
CB CA	SET 1,D
CB CB	SET 1,E
CB CC	SET 1,H
CB CD	SET 1,L
CB D6	SET 2,(HL)
DD CB 56 D6	SET 2,(IX+N)
FD CB 56 D6	SET 2,(IY+N)
CB D7	SET 2,A
CB D0	SET 2,B
CB D1	SET 2,C
CB D2	SET 2,D
CB D3	SET 2,E
CB D4	SET 2,H
CB D5	SET 2,L
CB DE	SET 3,(HL)
DD CB 56 DE	SET 3,(IX+N)
FD CB 56 DE	SET 3,(IY+N)
CB DF	SET 3,A
CB D8	SET 3,B
CB D9	SET 3,C
CB DA	SET 3,D
CB DB	SET 3,E
CB DC	SET 3,H
CB DD	SET 3,L
CB E6	SET 4,(HL)
DD CB 56 E6	SET 4,(IX+N)
FD CB 56 E6	SET 4,(IY+N)
CB E7	SET 4,A
CB E0	SET 4,B
CB E1	SET 4,C
CB E2	SET 4,D
CB E3	SET 4,E
CB E4	SET 4,H
CB E5	SET 4,L
CB EE	SET 5,(HL)
DD CB 56 EE	SET 5,(IX+N)
FD CB 56 EE	SET 5,(IY+N)
CB EF	SET 5,A
CB E8	SET 5,B
CB E9	SET 5,C
CB EA	SET 5,D
CB EB	SET 5,E
CB EC	SET 5,H
CB ED	SET 5,L
CB F6	SET 6,(HL)

DD CB 56 F6	SET 6,(IX+N)
FD CB 56 F6	SET 6,(IY+N)
CB F7	SET 6,A
CB F0	SET 6,B
CB F1	SET 6,C
CB F2	SET 6,D
CB F3	SET 6,E
CB F4	SET 6,H
CB F5	SET 6,L
CB FE	SET 7,(HL)
DD CB 56 FE	SET 7,(IX+N)
FD CB 56 FE	SET 7,(IY+N)
CB FF	SET 7,A
CB F8	SET 7,B
CB F9	SET 7,C
CB FA	SET 7,D
CB FB	SET 7,E
CB FC	SET 7,H
CB FD	SET 7,L
CB 26	SLA (HL)
DD CB 56 26	SLA (IX+N)
FD CB 56 26	SLA (IY+N)
CB 27	SLA A
CB 20	SLA B
CB 21	SLA C
CB 22	SLA D
CB 23	SLA E
CB 24	SLA H
CB 25	SLA L
CB 2E	SRA (HL)
DD CB 56 2E	SRA (IX+N)
FD CB 56 2E	SRA (IY+N)
CB 2F	SRA A
CB 28	SRA B
CB 29	SRA C
CB 2A	SRA D
CB 2B	SRA E
CB 2C	SRA H
CB 2D	SRA L
CB 3E	SRL (HL)
CB 3F	SRL A
CB 38	SRL B
CB 39	SRL C
CB 3A	SRL D
CB 3B	SRL E
CB 3C	SRL H
CB 3D	SRL L
96	SUB (HL)
DD 96 56	SUB (IX+N)
FD 96 56	SUB (IY+N)
97	SUB A
90	SUB B
91	SUB C
92	SUB D
93	SUB E
94	SUB H
95	SUB L
D6 56	SUB N
AE	XOR (HL)
DD AE 56	XOR (IX+N)
FD AE 56	XOR (IY+N)
AF	XOR A
A8	XOR B
A9	XOR C
AA	XOR D
AB	XOR E
AC	XOR H
AD	XOR L
EE 56	XOR N

In Numeric Order

00	NOP
01 1234	LD BC,NN
02	LD (BC),A
03	INC BC
04	INC B
05	DEC B
06 56	LD B,N
07	RLCA
08	EX AF,AF'
09	ADD HL,BC
0A	LD A,(BC)
0B	DEC BC
0C	INC C
0D	DEC C
0E 56	LD C,N
0F	RRCA
10 00	DJNZ $+2
11 1234	LD DE,NN
12	LD (DE),A
13	INC DE
14	INC D
15	DEC D
16 56	LD D,N
17	RLA
18 00	JR $+2
19	ADD HL,DE
1A	LD A,(DE)
1B	DEC DE
1C	INC E
1D	DEC E
1E 56	LD E,N
1F	RRA
20 00	JR NZ,$+2
21 1234	LD HL,NN
22 1234	LD (NN),HL
23	INC HL
24	INC H
25	DEC H
26 56	LD H,N
27	DAA
28 00	JR Z,$+2
29	ADD HL,HL
2A 1234	LD HL,(NN)
2B	DEC HL
2C	INC L
2D	DEC L
2E 56	LD L,N
2F	CPL
30 00	JR NC,$+2
31 1234	LD SP,NN
32 1234	LD (NN),A
33	INC SP
34	INC (HL)
35	DEC (HL)
36 56	LD (HL),N
37	SCF
38 00	JR C,$+2
39	ADD HL,SP
3A 1234	LD A,(NN)
3B	DEC SP
3C	INC A
3D	DEC A
3E 56	LD A,N
3F	CCF
40	LD B,B
41	LD B,C
42	LD B,D
43	LD B,E
44	LD B,H
45	LD B,L
46	LD B,(HL)
47	LD B,A
48	LD C,B
49	LD C,C
4A	LD C,D
4B	LD C,E
4C	LD C,H
4D	LD C,L
4E	LD C,(HL)
4F	LD C,A
50	LD D,B
51	LD D,C
52	LD D,D
53	LD D,E
54	LD D,H
55	LD D,L
56	LD D,(HL)
57	LD D,A
58	LD E,B
59	LD E,C
5A	LD E,D
5B	LD E,E
5C	LD E,H
5D	LD E,L
5E	LD E,(HL)
5F	LD E,A
60	LD H,B
61	LD H,C
62	LD H,D
63	LD H,E
64	LD H,H
65	LD H,L
66	LD H,(HL)
67	LD H,A

68	LD L,B
69	LD L,C
6A	LD L,D
6B	LD L,E
6C	LD L,H
6D	LD L,L
6E	LD L,(HL)
6F	LD L,A
70	LD (HL),B
71	LD (HL),C
72	LD (HL),D
73	LD (HL),E
74	LD (HL),H
75	LD (HL),L
76	HALT
77	LD (HL),A
78	LD A,B
79	LD A,C
7A	LD A,D
7B	LD A,E
7C	LD A,H
7D	LD A,L
7E	LD A,(HL)
7F	LD A,A
80	ADD A,B
81	ADD A,C
82	ADD A,D
83	ADD A,E
84	ADD A,H
85	ADD A,L
86	ADD A,(HL)
87	ADD A,A
88	ADC A,B
89	ADC A,C
8A	ADC A,D
8B	ADC A,E
8C	ADC A,H
8D	ADC A,L
8E	ADC A,(HL)
8F	ADC A,A
90	SUB B
91	SUB C
92	SUB D
93	SUB E
94	SUB H
95	SUB L
96	SUB (HL)
97	SUB A
98	SBC B
99	SBC C
9A	SBC D
9B	SBC E
9C	SBC H
9D	SBC L
9E	SBC (HL)
9F	SBC A
A0	AND B
A1	AND C
A2	AND D
A3	AND E
A4	AND H
A5	AND L
A6	AND (HL)
A7	AND A
A8	XOR B
A9	XOR C
AA	XOR D
AB	XOR E
AC	XOR H
AD	XOR L
AE	XOR (HL)
AF	XOR A
B0	OR B
B1	OR C
B2	OR D
B3	OR E
B4	OR H
B5	OR L
B6	OR (HL)
B7	OR A
B8	CP B
B9	CP C
BA	CP D
BB	CP E
BC	CP H
BD	CP L
BE	CP (HL)
BF	CP A
C0	RET NZ
C1	POP BC
C2 10E3	JP NZ,$+3
C3 10E6	JP $+3
C4 1234	CALL NZ,NN
C5	PUSH BC
C6 56	ADD A,N
C7	RST 0
C8	RET Z
C9	RET
CA 10F2	JP Z,$+3
CB 00	RLC B
CB 01	RLC C
CB 02	RLC D
CB 03	RLC E
CB 04	RLC H

CB 05	RLC L
CB 06	RLC (HL)
CB 07	RLC A
CB 08	RRC B
CB 09	RRC C
CB 0A	RRC D
CB 0B	RRC E
CB 0C	RRC H
CB 0D	RRC L
CB 0E	RRC (HL)
CB 0F	RRC A
CB 10	RL B
CB 11	RL C
CB 12	RL D
CB 13	RL E
CB 14	RL H
CB 15	RL L
CB 16	RL (HL)
CB 17	RL A
CB 18	RR B
CB 19	RR C
CB 1A	RR D
CB 1B	RR E
CB 1C	RR H
CB 1D	RR L
CB 1E	RR (HL)
CB 1F	RR A
CB 20	SLA B
CB 21	SLA C
CB 22	SLA D
CB 23	SLA E
CB 24	SLA H
CB 25	SLA L
CB 26	SLA (HL)
CB 27	SLA A
CB 28	SRA B
CB 29	SRA C
CB 2A	SRA D
CB 2B	SRA E
CB 2C	SRA H
CB 2D	SRA L
CB 2E	SRA (HL)
CB 2F	SRA A
CB 38	SRL B
CB 39	SRL C
CB 3A	SRL D
CB 3B	SRL E
CB 3C	SRL H
CB 3D	SRL L
CB 3E	SRL (HL)
CB 3F	SRL A
CB 40	BIT 0,B
CB 41	BIT 0,C
CB 42	BIT 0,D
CB 43	BIT 0,E
CB 44	BIT 0,H
CB 45	BIT 0,L
CB 46	BIT 0,(HL)
CB 47	BIT 0,A
CB 48	BIT 1,B
CB 49	BIT 1,C
CB 4A	BIT 1,D
CB 4B	BIT 1,E
CB 4C	BIT 1,H
CB 4D	BIT 1,L
CB 4E	BIT 1,(HL)
CB 4F	BIT 1,A
CB 50	BIT 2,B
CB 51	BIT 2,C
CB 52	BIT 2,D
CB 53	BIT 2,E
CB 54	BIT 2,H
CB 55	BIT 2,L
CB 56	BIT 2,(HL)
CB 57	BIT 2,A
CB 58	BIT 3,B
CB 59	BIT 3,C
CB 5A	BIT 3,D
CB 5B	BIT 3,E
CB 5C	BIT 3,H
CB 5D	BIT 3,L
CB 5E	BIT 3,(HL)
CB 5F	BIT 3,A
CB 60	BIT 4,B
CB 61	BIT 4,C
CB 62	BIT 4,D
CB 63	BIT 4,E
CB 64	BIT 4,H
CB 65	BIT 4,L
CB 66	BIT 4,(HL)
CB 67	BIT 4,A
CB 68	BIT 5,B
CB 69	BIT 5,C
CB 6A	BIT 5,D
CB 6B	BIT 5,E
CB 6C	BIT 5,H
CB 6D	BIT 5,L
CB 6E	BIT 5,(HL)
CB 6F	BIT 5,A
CB 70	BIT 6,B
CB 71	BIT 6,C
CB 72	BIT 6,D
CB 73	BIT 6,E
CB 74	BIT 6,H

CB 75	BIT 6,L
CB 76	BIT 6,(HL)
CB 77	BIT 6,A
CB 78	BIT 7,B
CB 79	BIT 7,C
CB 7A	BIT 7,D
CB 7B	BIT 7,E
CB 7C	BIT 7,H
CB 7D	BIT 7,L
CB 7E	BIT 7,(HL)
CB 7F	BIT 7,A
CB 80	RES 0,B
CB 81	RES 0,C
CB 82	RES 0,D
CB 83	RES 0,E
CB 84	RES 0,H
CB 85	RES 0,L
CB 86	RES 0,(HL)
CB 87	RES 0,A
CB 88	RES 1,B
CB 89	RES 1,C
CB 8A	RES 1,D
CB 8B	RES 1,E
CB 8C	RES 1,H
CB 8D	RES 1,L
CB 8E	RES 1,(HL)
CB 8F	RES 1,A
CB 90	RES 2,B
CB 91	RES 2,C
CB 92	RES 2,D
CB 93	RES 2,E
CB 94	RES 2,H
CB 95	RES 2,L
CB 96	RES 2,(HL)
CB 97	RES 2,A
CB 98	RES 3,B
CB 99	RES 3,C
CB 9A	RES 3,D
CB 9B	RES 3,E
CB 9C	RES 3,H
CB 9D	RES 3,L
CB 9E	RES 3,(HL)
CB 9F	RES 3,A
CB A0	RES 4,B
CB A1	RES 4,C
CB A2	RES 4,D
CB A3	RES 4,E
CB A4	RES 4,H
CB A5	RES 4,L
CB A6	RES 4,(HL)
CB A7	RES 4,A
CB A8	RES 5,B
CB A9	RES 5,C
CB AA	RES 5,D
CB AB	RES 5,E
CB AC	RES 5,H
CB AD	RES 5,L
CB AE	RES 5,(HL)
CB AF	RES 5,A
CB B0	RES 6,B
CB B1	RES 6,C
CB B2	RES 6,D
CB B3	RES 6,E
CB B4	RES 6,H
CB B5	RES 6,L
CB B6	RES 6,(HL)
CB B7	RES 6,A
CB B8	RES 7,B
CB B9	RES 7,C
CB BA	RES 7,D
CB BB	RES 7,E
CB BC	RES 7,H
CB BD	RES 7,L
CB BE	RES 7,(HL)
CB BF	RES 7,A
CB C0	SET 0,B
CB C1	SET 0,C
CB C2	SET 0,D
CB C3	SET 0,E
CB C4	SET 0,H
CB C5	SET 0,L
CB C6	SET 0,(HL)
CB C7	SET 0,A
CB C8	SET 1,B
CB C9	SET 1,C
CB CA	SET 1,D
CB CB	SET 1,E
CB CC	SET 1,H
CB CD	SET 1,L
CB CE	SET 1,(HL)
CB CF	SET 1,A
CB D0	SET 2,B
CB D1	SET 2,C
CB D2	SET 2,D
CB D3	SET 2,E
CB D4	SET 2,H
CB D5	SET 2,L
CB D6	SET 2,(HL)
CB D7	SET 2,A
CB D8	SET 3,B
CB D9	SET 3,C
CB DA	SET 3,D
CB DB	SET 3,E
CB DC	SET 3,H

CB DD	SET 3,L
CB DE	SET 3,(HL)
CB DF	SET 3,A
CB E0	SET 4,B
CB E1	SET 4,C
CB E2	SET 4,D
CB E3	SET 4,E
CB E4	SET 4,H
CB E5	SET 4,L
CB E6	SET 4,(HL)
CB E7	SET 4,A
CB E8	SET 5,B
CB E9	SET 5,C
CB EA	SET 5,D
CB EB	SET 5,E
CB EC	SET 5,H
CB ED	SET 5,L
CB EE	SET 5,(HL)
CB EF	SET 5,A
CB F0	SET 6,B
CB F1	SET 6,C
CB F2	SET 6,D
CB F3	SET 6,E
CB F4	SET 6,H
CB F5	SET 6,L
CB F6	SET 6,(HL)
CB F7	SET 6,A
CB F8	SET 7,B
CB F9	SET 7,C
CB FA	SET 7,D
CB FB	SET 7,E
CB FC	SET 7,H
CB FD	SET 7,L
CB FE	SET 7,(HL)
CB FF	SET 7,A
CC 1234	CALL Z,NN
CD 1234	CALL NN
CE 56	ADC A,N
CF	RST 8H
D0	RET NC
D1	POP DE
D2 12F0	JP NC,$+3
D3 56	OUT (N),A
D4 1234	CALL NC,NN
D4 1234	CALL NC,NN
D5	PUSH DE
D6 56	SUB N
D7	RST 10H
D8	RET C
D9	EXX
DA 1301	JP C,$+3
DB 56	IN A,(N)
DC 1234	CALL C,NN
DD 09	ADD IX,BC
DD 19	ADD IX,DE
DD 21 1234	LD IX,NN
DD 22 1234	LD (NN),IX
DD 23	INC IX
DD 29	ADD IX,IX
DD 2A 1234	LD IX,(NN)
DD 2B	DEC IX
DD 34 56	INC (IX+N)
DD 35 56	DEC (IX+N)
DD 36 56 56	LD (IX+N),N
DD 39	ADD IX,SP
DD 46 56	LD B,(IX+N)
DD 4E 56	LD C,(IX+N)
DD 56 56	LD D,(IX+N)
DD 5E 56	LD E,(IX+N)
DD 66 56	LD H,(IX+N)
DD 6E 56	LD L,(IX+N)
DD 70 56	LD (IX+N),B
DD 71 56	LD (IX+N),C
DD 72 56	LD (IX+N),D
DD 73 56	LD (IX+N),E
DD 74 56	LD (IX+N),H
DD 75 56	LD (IX+N),L
DD 77 56	LD (IX+N),A
DD 7E 56	LD A,(IX+N)
DD 86 56	ADD A,(IX+N)
DD 8E 56	ADC A,(IX+N)
DD 96 56	SUB (IX+N)
DD 9E 56	SBC A,(IX+N)
DD A6 56	AND (IX+N)
DD AE 56	XOR (IX+N)
DD B6 56	OR (IX+N)
DD BE 56	CP (IX+N)
DD CB 56 06	RLC (IX+N)
DD CB 56 0E	RRC (IX+N)
DD CB 56 16	RL (IX+N)
DD CB 56 1E	RR (IX+N)
DD CB 56 26	SLA (IX+N)
DD CB 56 2E	SRA (IX+N)
DD CB 56 46	BIT 0,(IX+N)
DD CB 56 4E	BIT 1,(IX+N)
DD CB 56 56	BIT 2,(IX+N)
DD CB 56 5E	BIT 3,(IX+N)
DD CB 56 66	BIT 4,(IX+N)
DD CB 56 6E	BIT 5,(IX+N)
DD CB 56 76	BIT 6,(IX+N)
DD CB 56 7E	BIT 7,(IX+N)
DD CB 56 86	RES 0,(IX+N)
DD CB 56 8E	RES 1,(IX+N)
DD CB 56 96	RES 2,(IX+N)

DD CB 56 9E	RES 3,(IX+N)
DD CB 56 A6	RES 4,(IX+N)
DD CB 56 AE	RES 5,(IX+N)
DD CB 56 B6	RES 6,(IX+N)
DD CB 56 BE	RES 7,(IX+N)
DD CB 56 C6	SET 0,(IX+N)
DD CB 56 CE	SET 1,(IX+N)
DD CB 56 D6	SET 2,(IX+N)
DD CB 56 DE	SET 3,(IX+N)
DD CB 56 E6	SET 4,(IX+N)
DD CB 56 EE	SET 5,(IX+N)
DD CB 56 F6	SET 6,(IX+N)
DD CB 56 FE	SET 7,(IX+N)
DD E1	POP IX
DD E3	EX (SP),IX
DD E5	PUSH IX
DD E9	JP (IX)
DD F9	LD SP,IX
DE 56	SBC A,N
DF	RST 18H
E0	RET PO
E1	POP HL
E2 13F4	JP PO,$+3
E3	EX (SP),HL
E4 1234	CALL PO,NN
E5	PUSH HL
E6 56	AND N
E7	RST 20H
E8	RET PE
E9	JP (HL)
EA 1401	JP PE,$+3
EB	EX DE,HL
EC 1234	CALL PE,NN
ED 40	IN B,(C)
ED 41	OUT (C),B
ED 42	SBC HL,BC
ED 43 1234	LD (NN),BC
ED 44	NEG
ED 45	RETN
ED 46	IM 0
ED 47	LD I,A
ED 48	IN C,(C)
ED 49	OUT (C),C
ED 4A	ADC HL,BC
ED 4B 1234	LD BC,(NN)
ED 4D	RETI
ED 50	IN D,(C)
ED 51	OUT (C),D
ED 52	SBC HL,DE
ED 53 1234	LD (NN),DE
ED 56	IM 1
ED 57	LD A,I
ED 58	IN E,(C)
ED 59	OUT (C),E
ED 5A	ADC HL,DE
ED 5B 1234	LD DE,(NN)
ED 5E	IM 2
ED 60	IN H,(C)
ED 61	OUT (C),H
ED 62	SBC HL,HL
ED 67	RRD
ED 68	IN L,(C)
ED 69	OUT (C),L
ED 6A	ADC HL,HL
ED 6F	RLD
ED 72	SBC HL,SP
ED 73 1234	LD (NN),SP
ED 78	IN A,(C)
ED 79	OUT (C),A
ED 7A	ADC HL,SP
ED 7B 1234	LD SP,(NN)
ED A0	LDI
ED A1	CPI
ED A2	INI
ED A3	OUTI
ED A8	LDD
ED A9	CPD
ED AA	IND
ED AB	OUTD
ED B0	LDIR
ED B1	CPIR
ED B2	INIR
ED B3	OTIR
ED B8	LDDR
ED B9	CPDR
ED BA	INDR
ED BB	OTDR
EE 56	XOR N
EF	RST 28H
F0	RET P
F1	POP AF
F2 1485	JP P,$+3
F3	DI
F4 1234	CALL P,NN
F5	PUSH AF
F6 56	OR N
F7	RST 30H
F8	RET M
F9	LD SP,HL
FA 1492	JP M,$+3
FB	EI
FC 1234	CALL M,NN
FD 09	ADD IY,BC
FD 19	ADD IY,DE

FD 21 1234	LD IY,NN
FD 22 1234	LD (NN),IY
FD 23	INC IY
FD 29	ADD IY,IY
FD 2A 1234	LD IY,(NN)
FD 2B	DEC IY
FD 34 56	INC (IY+N)
FD 35 56	DEC (IY+N)
FD 36 56 56	LD (IY+N),N
FD 39	ADD IY,SP
FD 46 56	LD B,(IY+N)
FD 4E 56	LD C,(IY+N)
FD 56 56	LD D,(IY+N)
FD 5E 56	LD E,(IY+N)
FD 66 56	LD H,(IY+N)
FD 6E 56	LD L,(IY+N)
FD 70 56	LD (IY+N),B
FD 71 56	LD (IY+N),C
FD 72 56	LD (IY+N),D
FD 73 56	LD (IY+N),E
FD 74 56	LD (IY+N),H
FD 75 56	LD (IY+N),L
FD 77 56	LD (IY+N),A
FD 7E 56	LD A,(IY+N)
FD 86 56	ADD A,(IY+N)
FD 8E 56	ADC A,(IY+N)
FD 96 56	SUB (IY+N)
FD 9E 56	SBC A,(IY+N)
FD A6 56	AND (IY+N)
FD AE 56	XOR (IY+N)
FD B6 56	OR (IY+N)
FD BE 56	CP (IY+N)
FD CB 56 06	RLC (IY+N)
FD CB 56 0E	RRC (IY+N)
FD CB 56 16	RL (IY+N)
FD CB 56 1E	RR (IY+N)
FD CB 56 26	SLA (IY+N)
FD CB 56 2E	SRA (IY+N)
FD CB 56 46	BIT 0,(IY+N)
FD CB 56 4E	BIT 1,(IY+N)
FD CB 56 56	BIT 2,(IY+N)
FD CB 56 5E	BIT 3,(IY+N)
FD CB 56 66	BIT 4,(IY+N)
FD CB 56 6E	BIT 5,(IY+N)
FD CB 56 76	BIT 6,(IY+N)
FD CB 56 7E	BIT 7,(IY+N)
FD CB 56 86	RES 0,(IY+N)
FD CB 56 8E	RES 1,(IY+N)
FD CB 56 96	RES 2,(IY+N)
FD CB 56 9E	RES 3,(IY+N)
FD CB 56 A6	RES 4,(IY+N)
FD CB 56 AE	RES 5,(IY+N)
FD CB 56 B6	RES 6,(IY+N)
FD CB 56 BE	RES 7,(IY+N)
FD CB 56 C6	SET 0,(IY+N)
FD CB 56 CE	SET 1,(IY+N)
FD CB 56 D6	SET 2,(IY+N)
FD CB 56 DE	SET 3,(IY+N)
FD CB 56 E6	SET 4,(IY+N)
FD CB 56 EE	SET 5,(IY+N)
FD CB 56 F6	SET 6,(IY+N)
FD CB 56 FE	SET 7,(IY+N)
FD E1	POP IY
FD E3	EX (SP),IY
FD E5	PUSH IY
FD E9	JP (IY)
FD F9	LD SP,IY
FE 56	CP N
FF	RST 38H

Appendic G

ZEAP Internal Registers

The contents of a number of memory locations used by ZEAP may be of interest to the user. The user is cautioned to use these registers only as directed. Any uses other than those documented below may cause unpredictable results.

All 16 bit values are stored with the least significant 8 bits first.

BUFP 0F00H - 0F01H

This 16 bit value is the address of the edit buffer. The first two bytes of the edit buffer itself contain a 16 bit offset, which when added to the contents of BUFP, give one more than the address of the end of the edit buffer. Thus if BUFP contained 2000H and 2000H - 2001H contained 0174H, then the extend of the edit buffer would be 2000H to 2173H, and could be dumped under NAS-SYS control using

```
  W 2000 2174
```

Note that the buffer itself is position independent. That is, a buffer created in one area of memory could be copied to another and ZEAP would continue to operate correctly on it, provided the new buffer origin were specified as explained in APPENDIX A.

OUTCH 0F05H - 0F06H

This 16 bit value is the address of the external output routine. It is initially set to point to a subroutine within ZEAP which outputs a single character to the UART. The user may substitute the address of a routine which outputs the ASCII character contained in register A. All registers must be preserved through this routine, except AF. A routine for driving a high-speed parallel printer might be substituted for example. All output from the "U" editor command and under the TTY assembler option is routed through OUTCH, but output from the TAPE assembler option is directly through the UART.

Appendix H

ASCII Code Table

All values are in hexadecimal, with zero parity.

00 NUL	10 DLE	20	30 0	40 @	50 P	60 `	70 p
01 SOH	11 DC1	21 !	31 1	41 A	51 Q	61 a	71 q
02 STX	12 DC2	22 "	32 2	42 B	52 R	62 b	72 r
03 ETX	13 DC3	23 #	33 3	43 C	53 S	63 c	73 s
04 EOT	14 DC4	24 $	34 4	44 D	54 T	64 d	74 t
05 ENQ	15 NAK	25 %	35 5	45 E	55 U	65 e	75 u
06 ACK	16 SYN	26 &	36 6	46 F	56 V	66 f	76 v
07 BEL	17 ETB	27 '	37 7	47 G	57 W	67 g	77 w
08 BS	18 CAN	28 (	38 8	48 H	58 X	68 h	78 x
09 HT	19 EM	29)	39 9	49 I	59 Y	69 i	79 y
0A LF	1A SUB	2A *	3A :	4A J	5A Z	6A j	7A z
0B VT	1B ESC	2B +	3B ;	4B K	5B [	6B k	7B {
0C FF	1C FS	2C ,	3C <	4C L	5C \	6C l	7C |
0D CR	1D GS	2D -	3D =	4D M	5D]	6D m	7D }
0E SO	1E RS	2E .	3E >	4E N	5E ^	6E n	7E ~
0F SI	1F VS	2F /	3F ?	4F O	5F	6F o	7F DEL

The following control codes are used by ZEAP 1.2 and earlier NASCOM monitors.

1C	CURSOR HOME (NASBUG T4 ONLY)
1D	BACKSPACE
1E	CLEAR SCREEN
1F	NEW LINE

For details of code use by NAS-SYS, refer to the NAS-SYS manual.

Appendix I

ZEAP 2.0 Object Listing (RAM version)

```
ZEAP 2.0 (C) 1980 SIGMA ACCOUNTING & MGMENT SERVICES LTD

 LOC  0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
1000 C3 36 1B C3 78 1B 00 20 - 00 50 C3 BF 13 00 00 00
1010 00 00 FF FF FF FF 00 00 - 00 00 00 36 10 AA 10 8F
1020 12 F4 12 91 20 2C 98 3B - 00 82 2B 29 27 C0 24 A0
1030 23 22 80 00 A0 A0 56 70 - 19 55 63 19 4E 13 19 52
1040 78 19 4F 24 19 41 2D 1D - 46 A2 19 58 1E 1A 5A FA
1050 19 49 0B 1B 50 5F 19 3A - CC 1F 48 2C 19 2B 17 19
1060 2D 1D 19 57 23 1D 4A 51 - 19 4B 56 19 47 35 19 59
1070 F3 18 2F 9D 19 00 5A 45 - 41 50 20 32 2E 30 2F 41
1080 34 39 38 20 20 46 72 65 - 65 3D 20 20 20 20 20 20
1090 45 6E 64 3D 20 20 20 20 - 20 20 45 6E 74 3D 5A 5A
10A0 5A 5A 5A 5A 5A 20 67 5B - 75 00 80 00 01 4C 44 60
10B0 83 78 04 F4 3E 04 B1 0A - 04 E5 1A 04 F3 3A 14 EC
10C0 57 14 EE 5F 03 00 83 40 - 04 F4 06 03 62 F2 21 04
10D0 F3 2A 03 6A F2 31 04 E2 - F9 14 F3 7B 03 0C F2 01
10E0 14 F3 4B 03 73 E2 22 14 - 8C 43 04 E0 32 03 31 E0
10F0 02 03 65 E0 12 13 C9 A0 - 14 D2 B0 13 C4 A8 14 D2
1100 B8 13 6C E0 47 13 6E E0 - 4F 01 4A 52 F8 18 03 09
1110 F8 20 02 50 F2 C3 03 06 - F2 C2 43 E3 E9 01 50 55
1120 53 48 8F C5 02 4F 50 8F - C1 01 43 41 4C 4C F2 CD
1130 05 06 F2 C4 02 50 83 B8 - 03 F4 FE 03 CC 2F 13 C9
1140 A1 14 D2 B1 13 C4 A9 14 - D2 B9 02 43 C6 3F 81 45
1150 51 D5 00 02 58 68 E8 08 - 03 64 E2 EB 03 6B E2 E3
1160 03 D8 D9 02 C9 FB 82 4E - D4 80 01 49 4E 43 80 04
1170 04 8C 03 03 60 F5 DB 14 - E7 78 13 00 E7 40 13 C9
1180 A2 14 D2 B2 13 C4 AA 14 - D2 BA 12 4D FE 46 81 44
1190 45 46 C2 04 84 D7 05 84 - CD 03 84 D3 02 03 43 80
11A0 05 04 8C 0B 02 4A 4E 5A - F8 10 02 41 C1 27 02 C9
11B0 F3 11 53 42 43 62 8C 42 - 04 60 83 98 05 F4 DE 02
11C0 55 42 83 90 04 F4 D6 82 - 4B 49 D0 81 0A 4C 41 83
11D0 20 0A 52 41 83 28 0B 4C - 83 38 0A 45 54 7A 83 C0
11E0 02 43 C6 37 03 41 4C F4 - DF 01 52 45 D4 C9 04 86
11F0 C0 14 C9 4D 14 CE 45 0B - 53 7A 83 80 02 43 41 4C
1200 F8 D7 0A 4C 43 83 00 04 - C1 07 0B 83 10 03 C1 17
1210 13 C4 6F 0A 52 43 83 08 - 04 C1 0F 0B 83 18 03 C1
1220 1F 13 C4 67 02 53 54 FC - C7 01 4F 52 83 B0 03 F4
1230 F6 83 C7 01 02 55 54 75 - E0 D3 14 67 80 41 14 C9
1240 A3 14 C4 AB 12 54 49 D2 - B3 13 44 D2 BB 01 41 44
1250 44 60 83 80 05 F4 C6 04 - 62 8C 09 03 43 60 83 88
1260 05 F4 CE 14 62 8C 4A 02 - 4E 44 83 A0 04 F4 E6 01
1270 58 4F 52 83 A8 04 F4 EE - 09 42 49 54 7A 83 40 11
1280 4E 45 C7 44 02 4F D0 00 - 01 48 41 4C D4 76 00 80
1290 00 01 C8 26 02 CC 62 01 - C1 60 02 C6 68 01 C4 22
12A0 02 C5 64 01 C2 20 02 C3 - 30 01 DA 2C 01 C3 66 01
12B0 4E DA 2A 02 C3 2E 01 53 - D0 6A 81 A4 00 01 C5 24
12C0 01 CC 28 01 C9 6C 12 D8 - 62 32 D9 62 01 CD 38 01
12D0 D0 36 02 C5 34 02 CF 32 - 01 D2 6E 00 60 63 28 26
```

ZEAP 2.0 (C) 1980 SIGMA ACCOUNTING & MGMENT SERVICES LTD

```
 LOC  0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
12E0 24 22 66 20 38 36 34 32 - 66 2E 2C 2A 6A 62 64 30
12F0 68 62 64 30 38 DC 12 08 - DC 12 38 E4 12 34 E8 12
1300 44 EC 12 44 F0 12 D9 E1 - D1 E3 47 14 CB 7E 20 77
1310 23 7E E6 7F FE 70 38 32 - 4F A8 0F FE 08 30 62 FE
1320 05 38 73 08 FD 7E F5 B7 - 20 1B 08 FE 06 FD 7E F4
1330 38 0B 28 0C FE 03 3D 30 - 0C 3C 28 01 3C 07 07 07
1340 FD 77 F3 E6 C7 C4 B5 13 - 18 4F FE 20 30 30 E5 2A
1350 1B 0F 85 6F 30 01 24 78 - 08 7E E6 0F 4F 7E 08 23
1360 46 23 66 68 06 00 ED B1 - E1 47 20 15 08 CB 21 D6
1370 10 30 FA CB 39 FD 7E F3 - B1 FD 77 F3 18 1B B8 28
1380 18 CB 7E 23 28 FB 2B 23 - 23 7E 5F E6 07 BA CA 10
1390 13 23 30 ED D9 C9 FD 77 - F2 E3 D5 E5 D9 B7 C9 D9
13A0 E1 D1 E3 7E FE 80 23 56 - D5 D9 D1 7A C9 7A F6 7F
13B0 A3 07 9F 92 C8 FD CB 00 - 4E C8 FD CB F6 F6 C9 DF
13C0 6F C9 44 4D 2A 85 0F B7 - ED 52 19 38 0B ED 42 30
13D0 03 B7 ED 62 19 22 85 0F - EB CD 35 14 EB B7 ED 42
13E0 C5 E3 C1 C5 ED B0 CD 42 - 14 C1 C9 21 00 00 1A D6
13F0 30 D8 FE 0A D0 D5 54 5D - 29 29 19 29 16 00 5F 19
1400 D1 13 18 EA 2A 0C 0C 3A - 0B 0C FE 02 3E 10 20 03
1410 3A 0E 0C 32 84 0F B7 3E - 06 CA 19 1B 7C B5 C0 3A
1420 84 0F 85 27 6F 7C CE 00 - 27 67 C9 23 23 AF 47 4F
1430 ED B1 3D BE C9 E5 2A 00 - 0F 5E 23 56 2B EB 19 EB
1440 E1 C9 E5 D5 2A 00 0F EB - B7 ED 52 EB 18 12 E5 2A
1450 00 0F 23 23 5E 23 56 E1 - C9 E5 D5 2A 00 0F 23 23
1460 73 23 72 D1 E1 C9 CD 35 - 14 EB CD 4E 14 19 C9 DF
1470 5F 3E FF B7 C8 C5 47 AF - FF CD 08 1D 10 F9 C1 C9
1480 3E 0D CD 19 1D 3E 0A CD - 19 1D 3A 08 0F 18 E4 3E
1490 20 36 A0 2B AE E6 7F 28 - F6 E5 AF 2B AE 28 09 3E
14A0 A0 AE 20 F6 36 20 18 F2 - E1 C9 EF 5A 45 41 50 20
14B0 5A 38 30 20 41 73 73 65 - 6D 62 6C 65 72 20 2D 20
14C0 00 C9 C5 2B E5 ED 5B 19 - 0F D5 16 00 D5 CD 4C 16
14D0 CB 71 28 0D CD 06 13 30 - F4 CD 35 14 2A 89 0F 18
14E0 1C CD 9F 13 38 F3 CB 7B - 28 39 ED 5B 0C 0F 18 33
14F0 CD 55 15 38 0D 13 13 CD - 26 15 28 19 C1 C1 C5 C5
1500 18 EE E1 CD 4C 16 CB 71 - 20 F9 11 00 00 FD CB F6
1510 DE 37 9F 18 0E DD E5 E3 - 23 B7 ED 52 E1 EB 2B 56
1520 2B 5E E1 C1 C1 C9 44 4D - E1 E3 C5 E3 E3 CD 4C 16
1530 E3 CB 71 28 10 23 BE 28 - F3 F5 CD 4D 16 F1 CB 71
1540 20 0C B1 18 09 CD 4C 16 - AF CB 71 28 01 91 E3 C1
1550 E3 E5 60 69 C9 CD 2D 14 - D5 CD 38 16 D1 D8 23 CD
1560 4C 16 30 F1 2B B7 C9 CD - 9C 1C CD AA 14 EF 53 79
1570 6D 62 6F 6C 20 54 61 62 - 6C 65 00 CD 9C 1C CD 9C
1580 1C 11 A4 10 21 9D 10 E5 - E5 2A 89 0F CD 55 15 38
1590 18 E5 D5 CD 26 15 C1 C1 - 30 F2 E1 E5 C5 C5 CD 26
15A0 15 E1 C1 30 E7 F1 C5 18 - E3 E1 C1 B7 ED 42 09 CA
15B0 9C 1C E5 E5 23 CD C2 14 - E1 28 20 E5 EB DF 66 EF
```

```
ZEAP 2.0 (C) 1980 SIGMA ACCOUNTING & MGMENT SERVICES LTD

 LOC  0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F

15C0 08 48 20 00 E1 2B CD 0C - 16 06 07 CD 82 16 21 5E
15D0 F4 19 11 A2 0B D4 21 16 - DC 9C 1C D1 18 A6 2A 00
15E0 0F 23 23 CD 2B 14 C8 5E - 23 56 2B E5 21 8F 18 7E
15F0 34 BE 00 35 2A 0C 0C B7 - ED 52 E1 3F D0 C8 18 E3
1600 23 B7 3E A0 12 C8 2B CD - 49 16 18 F4 CD 38 16 D8
1610 E5 EB DF 66 22 12 0F E1 - 23 CD 4C 16 CB 7F 11 8B
1620 0F E5 2A 29 0C EB 22 29 - 0C E1 C9 11 FA 0B 06 70
1630 3E 20 1B 12 10 FC 18 E9 - 7E C6 01 9F D8 5E 23 56
1640 2B E5 2A 0E 0C ED 52 E1 - C9 7E 12 13 23 7E FE 30
1650 38 13 FE 41 38 0A 0E C0 - FE 5B D8 0E 98 FE A0 C8
1660 0E E0 FE 3A D8 E5 E6 7F - 32 2D 0F 21 1D 0F 4E 23
1670 CB 7E 20 FA BE 20 F8 E1 - CB 61 C0 37 C9 CB 59 C0
1680 06 05 CB 61 20 07 CD 49 - 16 10 F7 04 C9 13 10 FD
1690 CD 4D 16 CB 41 C8 CD 4C - 16 18 F8 7C A5 3C C8 CD
16A0 21 16 DF 66 C9 06 00 E5 - FD 66 F6 2E 70 E3 11 00
16B0 00 CD 4D 16 FE 28 20 2E - E3 CB 54 20 78 CB D4 2C
16C0 CB 98 E3 CD 4C 16 38 1E - E3 FD 74 F6 CB 60 20 0A
16D0 CB 48 20 0D ED 53 F2 0F - 18 07 7B CD B1 13 FD 73
16E0 F7 45 E1 C3 90 16 ED 53 - 82 0F CB 69 28 38 FE 22
16F0 20 0F CD 4C 16 16 00 5F - FE A0 20 27 1E 20 2B 18
1700 22 EB FE 23 28 15 D5 DF - 64 1A FE 48 20 07 E1 2A
1710 21 0C 13 18 0C D1 CD EB - 13 18 06 13 DF 64 2A 21
1720 0C EB 2B E3 18 58 CB 71 - 28 74 CD C2 14 2B E3 20
1730 3E CB 48 3E 24 20 75 CB - 50 20 71 CB C8 7A FE 62
1740 20 21 7B E6 30 F5 B0 47 - 0F 0F 0F 0F A5 E6 01 B4
1750 67 F1 CB 7C 20 06 CB FC - B4 67 18 07 AC E6 30 3E
1760 25 20 49 7D E6 01 B2 6F - ED 5B 82 0F C3 C0 16 30
1770 0D FD 7E F6 E6 08 B4 67 - CB 40 3E 41 20 2E E5 CB
1780 58 2A 82 0F 20 03 19 18 - 03 B7 ED 52 EB E1 CB 48
1790 28 DA CB 60 3E 27 28 14 - CB 45 28 10 18 CE CB D8
17A0 E3 FE 2D CA C2 16 CB 49 - 20 C2 3E 26 C3 19 1B D9
17B0 2A 0C 0F FD CB 00 4E CA - 4C 18 FD CB 01 5E 28 32
17C0 ED 4B 7C 0F ED 5B 7E 0F - FD CB 00 5E 28 03 10 18
17D0 04 FD CB 01 5E CD 57 18 - 41 11 2F 0F 2A 0C 0F 22
17E0 80 0F FD CB 00 DE 18 E0 - 12 13 ED 43 7C 0F ED 53
17F0 7E 0F 2A 0C 0F E5 FD CB - 01 4E 28 2B ED 5B 0A 0F
1800 19 FD CB 00 76 20 20 01 - 00 0F 11 00 10 CD DE 18
1810 01 00 10 11 00 20 CD DE - 18 ED 4B 00 0F E5 CD 66
1820 14 EB E1 CD DE 18 77 FD - CB 00 46 20 1E 2A 29 0C
1830 11 69 F4 19 30 13 F5 CD - 9C 1C 11 D0 0B 06 46 CD
1840 30 16 11 8F 0B CD 21 16 - F1 DF 68 E1 23 22 0C 0F
1850 D9 C9 ED 4B 7C 0F BF C5 - F5 79 90 CA DB 18 21 1E
1860 1D 22 78 0C 2A 80 0F FD - CB 01 76 20 23 0E 00 DF
1870 66 21 2F 0F 06 08 7E DF - 67 23 EF 20 00 10 F7 79
1880 DF 68 EF 0D 00 F1 F5 20 - 4C EF 2E 0D 00 18 46 00
1890 47 AF CD 1E 1D EF FF FF - FF FF 00 7D CD 1E 1D 7C
```

ZEAP 2.0 (C) 1980 SIGMA ACCOUNTING & MGMENT SERVICES LTD

```
 LOC  0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
18A0 CD 1E 1D 78 CD 1E 1D F1 - F5 3E FF 20 01 AF 4F CD
18B0 1E 1D 7D 84 80 81 CD 1E - 1D 21 2F 0F 0E 00 7E 23
18C0 F5 81 4F F1 CD 1E 1D 10 - F5 79 CD 1E 1D 06 0A AF
18D0 CD 1E 1D 10 FA 21 F4 1C - 22 78 0C F1 C1 C9 E5 B7
18E0 ED 52 E1 D0 E5 B7 ED 42 - E1 D8 FD CB 00 F6 3E 60
18F0 C3 19 1B 2A 00 0F 23 23 - 23 23 23 7E FE FF 3E 05
1900 C2 1D 1B 2A 75 0F 22 00 - 0F 11 6F 0F 01 06 00 EB
1910 ED B0 C9 DF 77 DF 5B 7D - FD B6 01 18 08 7D 2F FD
1920 A6 01 18 01 7D FD 77 01 - 32 07 0F C9 7B 20 02 3E
1930 0F 32 14 0F C9 2A 0A 0F - 7C B5 20 10 FD CB 01 4E
1940 28 0A 2A 0E 0F 23 7C B5 - 28 02 2B E9 3E 04 C3 19
1950 1B 7B 32 09 0F C9 7B 32 - 08 0F C9 22 08 0F C9 22
1960 0A 0F C9 FD CB 00 D6 CD - 6F 14 CD 80 14 CD 71 14
1970 CD DE 15 CD 93 1C 18 FB - CD 04 14 EB 2A 00 0F 23
1980 23 23 CD 2C 14 C8 73 23 - 72 EB CD 1F 14 EB 30 F2
1990 3E 01 32 84 0F CD 1D 1B - 11 01 00 18 DF CD DE 15
19A0 18 2B 2A 29 0C 11 C1 FF - 19 7E EB 2A 85 0F FE 20
19B0 28 1B 21 41 00 19 01 43 - 00 ED B9 EB 11 2E 0F ED
19C0 B0 1B 3E A0 12 2A 00 0F - 23 23 23 23 23 22 85 0F
19D0 7E 3C C8 23 23 E5 11 2F - 0F 1A 13 FE A0 28 0C BE
19E0 23 28 F6 E1 7E 23 B7 20 - EC 18 E2 E1 CD 2D 14 EB
19F0 2A 85 0F ED 53 85 0F C3 - 93 1C 20 06 2A 10 0F 22
1A00 0C 0C CD DE 15 3E 03 D2 - 19 1B CD 93 1C AF 32 FE
1A10 0F EF 13 12 12 12 12 12 - 00 DF 63 C3 3C 1A FE 02
1A20 C2 17 1B CD DE 15 E5 2B - CD 4E 14 CD 55 15 38 05
1A30 1B 1B 23 18 F6 CD 59 14 - D1 C3 C2 13 1A FE 20 CA
1A40 17 1B D5 CD EB 13 C6 10 - C2 17 1B EB D1 B7 ED 52
1A50 01 FB FF 09 DA 17 1B DF - 64 2A 21 0C 7C B5 CA 17
1A60 1B 22 0C 0C 22 10 0F 21 - 2B 00 19 7E B7 28 05 36
1A70 20 23 18 F7 2B CD 8F 14 - E5 B7 ED 52 E5 23 38 02
1A80 23 23 E5 F5 E5 EB CD 4C - 16 CD 4E 14 30 02 13 13
1A90 D5 CD DE 15 D1 30 0B E5 - 23 CD 4C 16 30 02 1B 1B
1AA0 E1 37 D5 54 5D DC 2B 14 - C1 E3 09 19 E3 EB E3 ED
1AB0 52 D5 CD 35 14 19 EB 2A - 02 0F AF ED 52 DA 19 1B
1AC0 50 59 CD 59 14 E1 D1 CD - C2 13 F1 38 27 E1 E5 19
1AD0 EB CD 42 14 03 ED B8 AF - 12 1B E1 C1 E3 ED B8 21
1AE0 22 0C ED A8 ED A8 C1 2A - 85 0F B7 ED 52 19 38 04
1AF0 09 22 85 0F FD CB 00 66 - 28 0E 2A 0C 0C CD 1F 14
1B00 3E 02 DA 19 1B 22 87 0F - C3 CC 1B CD 04 14 22 87
1B10 0F FD CB 00 E6 18 F1 3E - 99 2A 78 0F E5 11 8F 0B
1B20 CD 21 16 5F EF 45 72 72 - 6F 72 20 00 7B DF 68 FD
1B30 CB 00 FE C3 9C 1C 2A 00 - 0F 22 75 0F 11 6F 0F 01
1B40 06 00 ED B0 21 06 10 11 - 00 0F 01 30 00 ED B0 3A
1B50 0B 0C FE 02 38 0E 28 06 - 2A 10 0C 22 00 0F 2A 0E
1B60 0C 22 02 0F 2A 00 0F 36 - 06 23 AF 77 23 77 23 77
1B70 23 77 23 3D 77 22 85 0F - 31 FE 0F 11 8A 0B CD 21
```

ZEAP 2.0 (C) 1980 SIGMA ACCOUNTING & MGMENT SERVICES LTD

```
 LOC  0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
1B80 16 EF 43 6F 70 79 72 69 - 67 68 74 20 28 63 29 20
1B90 53 69 67 6D 61 20 53 6F - 66 74 77 61 72 65 20 55
1BA0 6E 69 74 0D 00 AF CD B5 - 1F 32 00 0C 3A 14 0F 32
1BB0 77 0F FD 21 FE 0F FD F9 - 21 A8 10 DF 71 21 F4 1C
1BC0 22 78 0C 21 A5 1B 22 7A - 0F 22 78 0F 3A 07 0F CD
1BD0 C2 1F FD F9 AF 67 6F 22 - 0C 0C 2B 22 0E 0C 21 76
1BE0 10 11 CA 0B 01 28 00 ED - B0 21 88 1C 22 78 0C CD
1BF0 66 14 11 DE 0B CD 9B 16 - 2A 0C 0F 11 E8 0B CD 9B
1C00 16 2A 0E 0F 11 F2 0B CD - 9B 16 21 F4 1C 22 78 0C
1C10 11 8A 0B CD 21 16 FD CB - 00 66 28 05 2A 87 0F DF
1C20 66 DF 63 D5 1A 01 2B 0C - FE 20 20 06 0A FE 46 C2
1C30 A5 1B 02 01 0A 0C 13 03 - AF 02 DF 64 7E B7 28 11
1C40 03 23 7E 02 03 23 7E 02 - 21 0B 0C 34 7E FE 03 20
1C50 E9 ED 4B 2B 0C 2A 15 0F - D1 7E B7 CA 3C 1A 23 B9
1C60 28 04 23 23 18 F3 11 8A - 0B CD 21 16 5E 23 56 21
1C70 A5 1B E5 D5 2A 29 0C 11 - C1 FF 19 EB CD EB 13 EB
1C80 2A 0C 0C 3A 0B 0C B7 C9 - E5 2A 29 0C 77 23 22 29
1C90 0C E1 C9 CD 0C 16 C2 A5 - 1B CD 00 16 FD CB 00 56
1CA0 28 26 EB 21 BA 0B FD CB - 00 4E 28 03 21 CF 0B CD
1CB0 8F 14 EB CD 1E 16 11 8A - 0B 1A CD 19 1D 13 1A B7
1CC0 F2 BA 1C CD 80 14 18 1F - FD 7E 00 E6 03 FE 01 3A
1CD0 09 0F C4 73 14 3A 77 0F - D6 01 38 0B 20 03 3A 14
1CE0 0F 32 77 0F CC 0F 1D ED - 5B 12 0F ED 53 10 0F CD
1CF0 08 1D 3E 0D F5 E5 D5 C5 - FE 1F 28 02 DF 65 C1 D1
1D00 E1 FE 1B CA A5 1B F1 C9 - DF 62 D0 FE 1B 28 04 CF
1D10 FE 1B C0 2A 7A 0F E5 18 - CE CD 04 0F 18 EA CD BF
1D20 13 18 E5 FD 7E 01 E6 24 - F6 11 FD 77 01 21 80 1F
1D30 22 78 0F 21 FF FF 22 0E - 0F E1 E1 2E 01 E5 CD 66
1D40 14 22 0C 0F E5 CD B5 1D - E1 22 0C 0F E1 E5 CB 7D
1D50 28 04 CB 64 28 5C CB 5C - C4 6F 14 E1 7C E6 05 F6
1D60 02 6F E5 CD 9C 1C CB 45 - 20 19 CD AA 14 EF 53 6F
1D70 75 72 63 65 20 4C 69 73 - 74 69 6E 67 00 CD 9C 1C
1D80 CD 9C 1C 21 09 09 FD CB - 01 76 28 03 21 41 41 22
1D90 7C 0F E5 CD B5 1D FD CB - 01 7E C4 67 15 3E 61 FD
1DA0 CB 00 76 C4 1D 1B 3E 50 - FD CB 00 7E C4 1D 1B CD
1DB0 52 18 C3 A5 1B CD 35 14 - D5 DD E1 CD DE 15 2B 22
1DC0 89 0F 23 CD 2B 16 11 98 - 0B CD 21 16 CD 0C 16 C0
1DD0 F5 30 0B E5 2A 0C 0F DD - 75 00 DD 74 01 E1 FE 3B
1DE0 28 08 06 07 CD 82 16 CC - 7D 16 F5 CD 00 16 F1 D1
1DF0 E5 D5 08 F1 F5 21 00 00 - E5 E5 E5 08 3E 10 C2 19
1E00 1B 08 30 0B 21 9D 0B CD - C2 14 3E 31 D2 19 1B FE
1E10 3B CA 68 1F 11 8A 0B 2A - 0C 0F CD 9B 16 2A 17 0F
1E20 E5 16 00 D5 21 A3 0B 18 - 08 CD 06 13 3E 20 DA 19
1E30 1B CD 4C 16 38 F3 CD 93 - 16 FD CB EE 7E CA FA 1E
1E40 CD 9F 13 38 E7 F5 FE 80 - CB 59 3E 40 D2 DB 1E 20
1E50 DD F1 F5 FD CB F6 D6 01 - D0 0B FE 03 28 40 30 55
```

ZEAP 2.0 (C) 1980 SIGMA ACCOUNTING & MGMENT SERVICES LTD

```
 LOC  0  1  2  3  4  5  6  7    8  9  A  B  C  D  E  F
1E60 06 05 CD A7 16 3F 3E 40 - D2 19 1B 62 6B F1 FE 01
1E70 28 07 38 0C ED 4B 0C 0F - 09 22 0C 0F FD CB 00 9E
1E80 FD CB F8 46 28 0A FE 02 - 28 06 DD 75 00 DD 74 01
1E90 EB FE 01 11 8A 0B CC 21 - 16 DF 66 C3 68 1F 56 C5
1EA0 CD 4C 16 C1 BA 28 24 FE - A0 28 20 02 03 18 F0 CD
1EB0 AD 13 F1 30 16 C5 06 04 - CD A7 16 C1 F5 7B 02 03
1EC0 FD CB F1 46 28 E9 7A 02 - 03 18 E7 21 D0 0B B7 ED
1ED0 42 09 30 C7 7E 23 CD AF - 17 18 F3 28 89 F1 FE 81
1EE0 30 08 2A 0C 0F 22 0E 0F - 18 B1 CD 2B 16 18 AC CD
1EF0 A5 16 78 CD 06 13 3E 21 - 38 52 CB 59 28 F1 CD 9F
1F00 13 38 F3 C1 E1 CB 51 28 - 10 2B 2B FD CB 01 6E 20
1F10 08 D5 ED 5B 0C 0F ED 52 - D1 EB 7B CB 51 C4 B1 13
1F20 CB 49 C4 AD 13 79 B7 7B - 08 7A CB 41 EB E1 E5 37
1F30 F5 3F F5 08 F5 7A B0 0C - F5 3E CB CB 5B 28 01 F5
1F40 CB 65 28 1A 7C CB 73 28 - 08 B7 3E 22 C2 19 1B 18
1F50 05 C1 CB 45 F5 C5 3E DD - B5 F5 CB 63 20 EC 3E ED
1F60 CB 63 C4 AF 17 F1 30 FA - FD CB 00 46 CC 9C 1C FD
1F70 CB F6 5E 20 0F FD CB F6 - 76 3E 23 C4 1D 1B 18 09
1F80 FD CB F6 5E 3E 30 C4 1D - 1B 31 F6 0F F1 30 04 DD
1F90 23 DD 23 E1 CD 08 1D C3 - C3 1D 13 00 00 00 00 00
1FA0 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
1FB0 00 00 00 00 00 32 FE 0F - 21 BA 0B 06 10 77 23 10
1FC0 FC C9 32 FF 0F 3A 14 0F - 32 77 0F C9 EF 43 6F 6D
1FD0 6D 61 6E 64 3F 0D 00 DF - 63 01 2B 0C 1A FE 20 C8
1FE0 FE 41 38 0D FE 5B 30 09 - 02 32 0A 0C 13 DF 79 30
1FF0 03 DF 6B C9 DF 60 DF 5C - C9 00 00 00 00 00 00 00
```

The following addresses contain data particular to each copy of ZEAP 2.0

107F 1080 1081 1082 10A6 10A7 1FB4

Appendix J

ZEAP 1.2 Operation

This manual decribes the operation of ZEAP version 2.0 for use under the NAS-SYS monitor. This appendix describes the differences between ZEAP 2.0 and ZEAP 1.2, the latter for use with the NASBUG family of monitors.

The differences are as follows

* When ZEAP 1.2 is waiting for input, it prompts with a ":" at the beginning of the bottom line of the screen. It is impossible to move the cursor to the left of the ":" prompt, or off the right hand end of the line.

* There are no cursor control facilities. The BACKSPACE key may be used to correct errors in typing immediately. Line deletion is achieved by typing "!" (SHIFT "1"), not ESC. The "!" is displayed, the deleted line is scrolled up the screen, and a ":" prompt is issued ready for the next line of input. Troughout this manual read "NEW LINE" for "ENTER".

* Since there is no cursor control, some of the facilities of the NAS-SYS cursor control have been incorporated in the "Z" editor command. These facilities are documented in full here.

The limitation of line replacement as a method of correcting minor mistakes is clear from the following example:

```
:40 ILNE 4
```

To interchange the "I" and the "L" requires that the whole line be re-entered. A powerful alternative is provided in the ZEAP editor- Entering

```
:Z 40
```

causes the following two lines to displayed:

```
:0040 ILNE 4
      ^
```

ZEAP has now entered Edit mode. The arrow under the first character of the line is the cursor. The user can advance the cursor to the position where the correction ia to be made by depressing the space bar. Typing a single space gives

```
  :0040 ILNE 4
         ^                 (SPACE typed)
```

Now the offending letter "L" can be deleted by typing "<" (SHIFT ","), thus

```
  :0040 INE 4
         ^                 ("<" typed)
```

Note that all the characters to the right of the cursor have been moved up to fill the gap left by the deleted "L". Now, using the BACKSPACE key, the cursor can be positioned under the "I", before which an L is to be inserted:

```
  :0040 INE 4
        ^                  (BACKSPACE typed)
```

Now to make room for the L the ">" (SHIFT ".") is used:

```
  :0040 INE 4
       ^
```

Note that all characters above and to the right of the cursor are shifted one place right to make room for the insertion. Finally typing "L" will give

```
  :0040 LINE 4
         ^
```

The "L" is inserted at the position of the cursor, which is then advanced one place.

Now that editing is completed, the NEW LINE is pressed to signify that fact

```
  :0040 LINE 4
  :
```

The cursor arrow disappears, and the editor prompts for the next command. The new line 40 is entered just as if it had been typed manually.

The SPACE and BACKSPACE keys cause the cursor to move one place right or left respectively. It is impossible to move the cursor off the left or right hand ends of the buttom line. These keys cannot be used to enter spaces or delete characters in the line being edited as they do in normal editor operation. The ">" and "<" keys must be used for these purposes, respectively.

The ">" (insert) key causes all characters above and to the right of the cursor to be shifted one place right to allow insertion of text. Repeated depressions cause more space to be left. Characters shifted off the right hand end of the line are lost. The cursor remains where it is.

The "<" (delete) key causes the character above the cursor to be deleted and all characters to the right of the deleted character to be moved one place left to fill the gap left by the deleted character. Repeated depressions cause more characters to be deleted. Spaces enter from the right-hand end of line. The cursor remains where it is.

The NEW LINE key causes Edit mode to be terminated, and the edited line is interpreted as a line of source code entry.

The "!" key causes Edit mode to be abandoned. The edited line is ignored and the original version of it remains intact in the Edit Buffer.

Depressing any other key causes the approriate character to replace the character currently above the cursor, and the cursor is advanced one place to the right.

A space may be entered into the line being edited by depressing the "<", ">" and SPACE keys in sequence.

In Edit mode the sequence number itself can also be edited. Thus

```
  :Z40
  :0040 LINE 4
        ^
```

Typing three backspaces followed by a "7" gives

```
  :0070 LINE 4
      ^                    (BACKSPACE BACKSPACE BACKSPACE "7" typed)
```

Now typing NEW LINE gives

```
  :0070 LINE 4
  :
```

And now

```
  :V
  0010 LINE 1
  0020 LINE 2
  0030 LINE 3
  0040 LINE 4
  0070 LINE 4
  :
```

Note that the original line still exists, so

```
  :40                      (deletes line 40)
  :V
  0010 LINE 1
  0020 LINE 2
  0030 LINE 3
```

```
0070 LINE 4
:
```

In summary

Z y	edit line y

and then the following keys may be used

SPACE	cursor right
BACKSPACE	cursor left
">"	insert
"<"	delete
NEW LINE	leave Edit mode
"!"	abandon Edit mode
other	replace current character

ZEAP 2.0 REFERENCE CARD

EDITOR COMMANDS

A m n	ASSEMBLE SOURCE PROGRAM
F/string/	FIND STRING
F	FIND NEXT
FT	FIND FROM TOP
/ y	FIND STRING FROM LINE
G	GO TO OBEJECCT PROGRAM
H p	SET PAGE MODE
I s i	ENTER AUTO INPUT MODE
J p	SET VDU DELAY
K p	SET UART DELAY
N	RETURN TO MONITOR
O x	SET ASSEMBLER OPTION
+ x	SET ASSEMBLER OPTION ON
- x	SET ASSEMBLER OPTION OFF
P h	SET MONITOR OFFSET
Q h	SET END OF LINE DELAYS
R s i	RESEQUENCE SOURCE CODE
U m n	LISTING TO UART
V m n	LISTING TO VDU
W m n	LIST ERRORS
X m n	BLOCK DELETE
Y	RECOVER EDIT BUFFER
Z y	EDIT SOURCE LINE

ZEAP ERROR CODES

00	CORE FULL
01	RESEQUENCE OVERFLOW
02	AUTO INPUT OVERFLOW
03	NON-EXISTENT LINE
04	ILLEGAL GO
05	RECOVERY IMPOSSIBLE
06	ZERO INCREMENT
10	UNRECOGNISABLE STATEMENT
20	UNKNWON MNEMONIC
21	CONTEXT ERROR
22	INDEX REGISTER ERROR
23	TRUNCATION ERROR
24	TOO MANY REGISTERS
25	REGISTER MISMATCHED
26	ILLEGAL CHARACTER
27	ILLEGAL COMMAND
28	PARENTHESIS ERROR
30	LABEL NOT FOUND
31	LABEL REDEFINED
40	DIRECTIVE ERROR
41	ILLEGAL FORWARD REFERENCE
50	ERROR IN ASSEMBLY
60	MEMORY OBJECT CODE OVERWRITING
61	OVERWRITE ERROR
90	CHECKSUM ERROR
99	ILLEGAL COMMMAND

ASSEMBLER OPTIONS

+01	SUPPRESS SOURCE LISTING
+02	OBJECT CODE TO MEMORY
+04	SOURCE LISTING TO TTY
+08	OBJECT CODE TO TAPE
+10	FORCE SECOND PASS
+20	ADJUST RELATIVE JUMP OFFSETS
+40	OBJECT TAPE FORMAT
+80	LIST SYMBOL TABLE

ASSEMBLER DIRECTIVES

EQU	EQUATE SYMBOL
ORG	SET ORIGIN
DEFS	DEFINE SPACE
DEFB	DEFINE BYTE
DEFW	DEFINE WORD
DEFM	DEFINE MESSAGE
SKIP	SKIP LINE
ENT	ENTRY POINT