OPERATIONS
MANUAL

Text Editor
Z 80 Global Assembler
Linker

SD SYSTEMS

POST OFFICE BOX 28810
DALLAS, TEXAS 75226

SD SYSTEMS

3017 LINCOLN
GARLAND, TEXAS 75041

June 12, 1979

ADDENDUM

Z80 GLOBAL PACKAGE

A change has been made to the assembler that makes it no longer necessary to have "-$" after labels in the relative jump instructions (JR) and the "DJNZ" instruction. Thus the following instructions are now assembled the same (Version 3.1):

JR TAG-$ and JR TAG

DJNZ TAG-$ and DJNZ TAG

All other configurations of operands for the relative jumps are assembled in the same way as in Version 3.0 (See paragraph 4 on page 32 Test Editor, Z80 Global Assembler, Linker Manual.)

800-527-3460 — 214-271-4667 — TELEX 73-0151

February 5, 1979

ADDENDUM

Z80 GLOBAL PACKAGE

The Assembler (ZASM) and Linker (LINK) have been revised (Version 3.0) to allow the options to be entered on the command line. This makes it possible to batch up several "assemblies" and "Links" to be done without operator interaction.

ASSEMBLER OPTIONS

To specify options for an assembly enter the file name, a space, and a slash (/), and then required options (Page 26).

Example: A> ZASM FILE.ASM /CL (CR)

This would cause the Z80 source program file named "FILE.ASM" to be assembled with the 'C' (cross reference table) and 'L' (listing on printer) options.

Note that if the slash (/) is omitted, the "options" will be requested as specified on Page 25, which prevents continuous batch operation. Therefore, if no options are required, enter the slash (/) on the command line with no options following.

LINKER OPTIONS

Linker options are entered on the command following the last file name in the list of files to be linked.

Example: A> LINK MAIN,SUB1,SUB2,SUB3 /CU A=1000 (CR)

This example would link the four programs listed with the 'C' (cross reference table), 'U' (undefined symbol list) and 'A' (link start address=1000 (hexadecimal) options (Page 37).

If the slash (/) is omitted, the options will be requested as shown in the example on Page 39.

The 'A' option is used to specify where the first relocatable module will be positioned. If one or more absolute modules precede the first relocatable module, they will be positioned at their "ORG" address while the relocatable module will be positioned starting at the address (hexadecimal) specified using the 'A' option. Note that if more options are entered after "A=hhhh", a comma (,) must be entered to separate the address specification and the following option.

TABLE OF CONTENTS

1.0 INTRODUCTION

The Text Editor assists the user in origination and modification of assembly language source programs and English text documentation. The Editor resides on a 32K system diskette. It permits random access editing of ASCII diskette files. The Editor is designed for usage with any CP/M compatible disk operating system (DOS) using a Z80 microprocessor. (CP/M is a registered trademark of Digital Research, Pacific Grove, California).

The Text Editor permits random access editing of ASCII diskette files on a line basis or character basis. Whole lines and character strings embedded within lines can be easily accessed, changed, deleted, or added to an existing or new diskette file. The size of the file to be edited is limited only by diskette capacity. All I/O operations to the diskette are transparent to the user.

The Editor is resident on diskette. When loaded, it starts at RAM address 100H. Editor buffers and variables are placed in RAM between the top of the Editor and the bottom of the Operating System. All I/O is done with the console device and the disk.

2.0 DEFINITIONS

SOURCE - ASCII characters comprising a Z80 assembly language program or some other text.

FILE - a diskette file which contains the SOURCE.

LINE - a single source statement which ends with a carriage return.

LINE POINTER - the position in the source where the next action of the Editor will be initiated.

CURRENT LINE - the line in the source pointed to by the LINE POINTER.

LINE NUMBER - the decimal number of a line, beginning at one (0001) for the first line in a file and increasing sequentially for each line. The maximum line number allowed is 9999 (decimal). Line numbers are assigned dynamically as editing of the file progresses. This means that when lines are added to or deleted from a file, all lines are automatically renumbered.

INSERT - installation of one or more lines in a file immediately following the current line. Inserted lines are assigned sequentially increasing line numbers.

DELETE - removal of one or more lines from a file.

3.0 USING THE EDITOR - CONSOLE INTERACTION

All user interaction with the Editor is via the user console. The Editor issues prompts and messages to direct the user. The user responds by entering commands or data via the console keyboard. Each command or data line is terminated by a carriage return.

The following conventions are used in this manual:

(CR) stands for carriage return.
All user input is underlined.
User input which must be entered exactly as shown is in upper case letters.
User input which is variable is shown in lower case.

4.0 USING THE EDITOR - ENTERING COMMANDS

The Editor prompts for a command with an asterisk (*). The user may then enter commands via the console keyboard. Modification of the input, such as rubout, backspace, and line delete, is supported by the operating system. entered in lower case as well as upper case. Several commands may be entered on one line. Blanks and commas are ignored on input. A command line is terminated by a carriage return. A command line may have up to 80 characters in it, including the carriage return.

All commands consist of one character followed by an optional operand. The operand may be separated from the command by zero or more blanks or commas. The operand may be a decimal number in the range 0-9999. This specifies the number of lines upon which the command is to operate. Alternatively, the operand may be two decimal numbers separated by a minus sign (-). In this case, the command takes effect on lines numbered from the first number in the operand through and including the second number. If the operand is not entered, it assumes a value of one (except for the 'F' command).

EXAMPLE

V

-VIEW command with no operand. The operand value assumes the value of one.

V5 or V 5 or V,5

-one operand shown which acts on the next 5 lines in the source file.

V42-45 or V 42-45 or V,42-45

-two operands entered. The VIEW command acts on lines numbered 42 through 45.

5.0 USING THE EDITOR - FIRST STEPS

After booting up DOS, the Text Editor may be executed by the following command:

A>EDIT filename(CR)

-where filename is the name of the diskette file to be edited on the currently selected disk. The file may not have an extension of COM, BAK, or $$$.

EXAMPLE

A>EDIT MYFILE(CR)

-user selects to run the Editor to edit the file named 'MYFILE' on the currently selected disk.

If the file does NOT exist on the diskette, then the Editor outputs the following message on the console:

***NEW FILE

-Editor indicates that a new file is being created.

The Editor then enters the 'DATA MODE' and waits for lines of data to be entered by the user:

```
***DATA MODE
0001
```

-Editor prompts for data lines starting with line number 0001 (see I-INSERT command).

At the end of editing, the new file will automatically be created.

If the file does exist on the disk, then editing of that file will be done, and the Editor prompts for a command:

```
*
```

-Editor prompts for a command (see below).

At the end of editing, the original file will be renamed with an extension of 'BAK'. The file which was edited will have all the changes in it.

The following pages describe each of the Editor commands in detail.

6.0 EDITOR COMMANDS

6.1 An - ADVANCE

Format:

An

or

an

-where n is a decimal number.

This command is used to advance the line pointer (toward the end of the file) a specified number of lines. If the operand n is not entered or it is zero, then the pointer will be position to the next line in the file. The line which is accessed is printed on the console after this command.

EXAMPLE

```
*A(CR)
 -----
        -user advances to next line.
0015 ANY STATEMENT.
        -the next line with its line number is printed on
        the console.
*A5(CR)
 ------
        -user advances 5 lines from current pointer.
0020 SOME STATEMENT
        -Editor prints line number and the line.
*
        -Editor prompts for a command.
```

If the user attempts to advance the line pointer beyond the end of the file, then an end-of-file indicator message will be printed on the user console. The line pointer will be on the last line of the file.

EXAMPLE

```
*A9999(CR)
 ---------
        -user advances over a large number of lines.
0438 LAST LINE OF FILE
***EOF
        -Editor prints last line of file and end-of-file
        indicator.
*
        -Editor prompts for a command.
```

6.2 Bn - BACKUP

Format:

Bn

or

bn

-where n is a decimal number.

This command is used to backup the line pointer (toward the beginning of the file) a specified number of lines. If the operand n is zero or it is not entered, then the pointer is positioned to the previous line in the file. The line which is accessed is printed on the console.

EXAMPLE

```
*B(CR)
 -----
        -user backs up over one line in the file.
0019 A LINE OF INFORMATION
        -Editor prints the line number and the line.
*B4(CR)
 ------
        -user backs up 4 lines from current position in the file
0015 SOME LINE
        -Editor prints the line number and the line.
*
        -Editor prompts for a command.
```

If the user attempts to back up the line pointer past the start of the file, then a top-of-file indicator will be printed on the user console. The line pointer will be on line number 0001.

EXAMPLE

```
*B9999(CR)
 ---------
        -user backs up over a large number of lines.
***TOF
0001 FIRST LINE OF FILE
        -Editor prints top-of-file indicator and first
        line of the file.
*
        -Editor prompts for a command.
```

6.3 Cn /string1/string2/ - CHANGE STRING

Format:

Cn/string1/string2/

or

cn/string1/string2/

-where n indicates the number of occurrences to change, string1 represents the characters to be changed, string2 represents the substitute or new characters, and / represents a delimiter character which does not appear in either string.

This command changes the next n occurrences of character string1 to character string2 starting with the current line. Any character which does not appear in either string1 or string2 may be used as a delimiter. All three delimiters must be identical, with the exception that the last delimiter may be a carriage return. If the operand is zero or if it is not entered, then only one occurrence of string1 will be changed. In this case, only the current line will be searched in order to locate string1. If string1 is not found in the current line, then the Editor issues a warning prompt ('?') and a new command prompt (*). The line pointer will stay on the same line.

If the operand n is greater than 1, then the search for occurrences of string1 occurs in a sequential manner starting with the current line. Each line which is changed is printed on the user console. After all changes are done, the line pointer will be on the last line that was changed. If the nth occurrence of string1 is not found before the end of the file is encountered, then the last line of the file is printed by the Editor, as well as an end of file indicator (***EOF). The line pointer will be on the last line of the file.

If string2 has no characters in it, then character string1 will be deleted each time it is encountered by the change command.

EXAMPLES

```
*V(CR)
 -----
        -user views current line.
0009 THIS IS A RECORD
        -Editor prints line number and line.
*C /THIS/THAT/(CR)
 -----------------
        -user enters change command.
0009 THAT IS A RECORD
        -Editor prints it.
*C/IS/WAS(CR)
 ------------
        -note that a carriage return for the last delimiter
        is allowed.
0009 THAT WAS A RECORD
```

```
*C /WAS //(CR)
 ------------
```
-this is the method used to delete characters in a line.
```
0009 THAT A RECORD
*C 2 /T/V/(CR)
 -------------
```
-note that blanks can be inserted between the command and operand and string definition to make the command more readable.
```
0009 VHAV A RECORD
*C4/VHAV/THAT/(CR)
 -----------------
```
-this is a multiple change request which will search forward in the file starting with the current line.
```
0009 THAT A RECORD
0024 SOME TIMES THAT IS
```
-Editor prints out each line that is changed.
```
0043 LAST LINE OF FILE
***EOF
```
-Editor reached the end of the file before any more changes could be done. An end-of-file indicator message is printed. The line pointer is on the last line in the file.

```
*
```
-Editor prompts for a command.

6.4 Dn - DELETE

Format:

Dn

or

dn

or

Dn-m

or

dn-m

-where n and m are decimal numbers.

This command is used to delete, or remove, the specified lines from the file. If the operand is not entered or is zero, then only the current line is deleted. Note that line numbers are assigned dynamically as editing progresses. This means that lines in a file essentially get renumbered each time one or more lines are deleted from the file.

EXAMPLE

```
*D(CR)
 -----
        -user deletes the current line from the file.
        The line pointer is on the next line in the file.
*D4(CR)
 ------
        -user selects to delete 4 lines starting with
        the current line from the file.
*D4-15(CR)
 ---------
        -user deletes lines numbered 0004 through and including
        0015 from the current file.
*
        -Editor prompts for a command.
```

6.5 En - EXCHANGE

Format:

```
        En
or
        en
or
        En-m
or
        en-m
                -where n and m are decimal numbers.
```

This command exchanges the specified lines with new lines to be inserted via the DATA MODE. It is exactly equivalent to the command sequence:

```
        Dn          -delete lines
        B           -backup one line
        I           -go to DATA MODE
```

6.6 Fn - PRINT FLAG

Format:

```
        Fn
or
        fn
                -where n=0 will inhibit printing after all but
                the V-VIEW command, and n not = 0 will allow
                printing after all change or access commands.
```

The Editor normally prints on the console device any lines which are accessed or changed. Thus, the following commands print out a line: An, Bn, Cn, Ln, Sn, Vn. In order to reduce the print out time on a slower device (such as a teletype), this command can be used to inhibit print out on all of the commands except V-VIEW.

6.7 G file - GET FILE COMMAND

Format:

G filename

or

g filename

-where filename is the name of a file on the selected disk.

This command is used to obtain lines from a given file on disk and insert them in sequence following the current line. All lines in the file requested are read. The file which is read is not altered in any way. This command can be used with the P - PUT command to move blocks of text around within a file being edited (See P - PUT command for example).

6.8 I - INSERT COMMAND

Format:

```
I
```

or

```
i
```

This command is used to insert data lines into the file being edited or to build new files. The inserted lines always FOLLOW the current line. After the command is entered, the Editor responds with the message:

```
***DATA MODE
```

The user then enters data lines ending with carriage returns. The Editor prompts with the line number for each line to be inserted. To terminate the insertions, the user enters a single carriage return. Note that blank lines must be entered as 'space carriage return' because a single carriage return terminates the DATA MODE. After the user terminates the DATA MODE, the Editor prompts for a new command (*). Lines can be inserted before the first line of a file by using the T - INSERT AT TOP command. Note that line numbers are assigned dynamically while editing progresses. This means that the lines of a file essentially get renumbered whenever new lines are inserted.

EXAMPLE

```
*I(CR)
-----
        -user selects data mode to insert lines into the file
        being edited.
***DATA MODE
        -Editor responds with message
0004 THIS IS AN INSERTED LINE.(CR)
     ----------------------------
        -the line number being entered is printed by the
        Editor.  The user then enters the line of data.
0005 (CR)
     ----
        -user terminates DATA MODE with a carriage return.
*
        -Editor prompts for another command.
```

Note that modification of entered data lines can be done while they are being typed just as in the DOS system. Inserted lines can be up to 128 characters long.

6.9 Ln - GO TO LINE NUMBER n

Format:

```
Ln
```

or

```
ln
```

This command positions the line pointer to line number n. If the operand is zero or it is not entered, then line number 0001 is accessed. Any line number can be accessed from any position in the file. The line which is accessed is printed on the console. If the line number cannot be found because it is larger than the last line number in the file, then the pointer will be positioned at the last line in the file and an end-of-file indicator message will be printed.

EXAMPLE

```
*L10(CR)
 -------
        -user accesses line number 0010.
0010 THIS IS A LINE OF DATA
        -line number 0010 is printed with its line number.
*L2001(CR)
 ----------
        -user selects line number 2001.
0943 LAST LINE OF FILE
        -Editor prints last line of file.
***EOF
        -Editor prints an end-of-file indication.
*L1(CR)
 ------
        -user selects line number 1.
***TOF
0001 FIRST LINE OF FILE
        -Editor responds with top of file indicator
        and first line of file and its line number.
*
        -Editor prompts for a command.
```

6.10 Pn file - PUT

Format:

Pn filename

or

pn filename

or

Pn-m filename

or

pn-m filename

-where n and m are decimal numbers and filename is the name of a file on the selected disk.

This command is used to output one or more lines to a file on disk. This can be used to break up a given source module. It can also be used with the G - GET command to move blocks of text around in a file being edited. If the operand is not entered or it is zero, then only one line will be output. Lines of text which are output by the PUT command are not deleted. They may be deleted via the D - DELETE command after the PUT command is used. The filename specified must not be the same as the current file being edited. If the file already exists on disk, it is erased before any lines are output to it. After the PUT command is used, the file output to the disk remains on the disk.

EXAMPLE

```
*P25-30 TEMP(CR)
 ---------------
        -user outputs lines 25 through 30 to a file
        called TEMP.  The space between the number and
        the filename is not required.
*D25-30(CR)
 ----------
        -user deletes lines 25 through 30 from file
        being edited.
*L1(CR)
 ------
        -user accesses line number one.
*G TEMP(CR)
 ----------
        -user reads lines from file TEMP and places them
        after line number one.  This effectively moves
        lines 25-30 to just after line 1.  The space
        between the command and the filename is not
        required.
*
        -Editor prompts for a command
```

6.11 Q - QUIT

Format:

Q

or

q

This command returns control to the Operating System. The original file will be backed up on the same primary filename with a secondary filename of BAK. All of the editing will be saved in the file under the original file name. QUIT can be done at any time during the course of editing.

6.12 Sn /string/ - SEARCH FOR STRING

Format:

Sn /string/

or

sn /string/

-where n is the number of occurrences to be found, string represents any set of characters which is to be searched for, and / represents a delimiter character which does not appear in the string.

This command searches the file, starting with the NEXT line, for n occurrences of the character string between the delimiters. Each line which contains the string will be printed on the console. The pointer is positioned on the line of the nth occurrence of the string. If the nth occurrence of the string cannot be found before the end of the file being edited, then the Editor issues an end-of-file indicator (***EOF). This command always searches forward (toward increasing line numbers) in the file.

Any character which does not exist in the string to be searched for may be used as a delimiter. The second delimiter may be a carriage return. If the operand n is zero or it is not entered, then only the first occurrence of the string will be sought.

EXAMPLE

```
*S /ORD/(CR)
 -----------
        -user selects to search forward in the file, beginning
        with the next line, for the string 'ORD'.  Only the
        first occurrence of the string is sought.  The blank
        between the command and the string is not required.
0021 SOME ORDERLY DATA
        -Editor prints the line number and line when the
        string is found.  The line pointer is on line 0021.
*S10/9AH/(CR)
 ------------
        -user selects to search for and view the next
        10 occurrences of the string '9AH'.
***EOF
        -Editor encountered the end of the file and found
        no occurrences of the string.  The end-of-file
        indicator is printed.
*
        -Editor prompts for a command.
```

6.13 T - INSERT AT TOP

Format:

```
T
```

or

```
t
```

This command inserts data lines at the top (start) of the file BEFORE the first line in the file. See the I-INSERT command for proper usage.

6.14 Vn - VIEW

Format: Vn or vn or Vn-m or vn-m -where n and m are decimal numbers.

This command prints the specified lines on the console device. The line pointer is updated to the last line printed. If the operand n is zero or is not entered, then only the current line is printed.

EXAMPLE

```
*V(CR)
 -----
        -user views current line on the console.
0009 THIS IS A LINE
        -Editor prints line number and line.
*V3(CR)
 ------
        -user views current line plus two more.
0009 THIS IS A LINE
0010 THIS IS NEXT LINE
0011 THIS IS ANOTHER LINE
        -Editor prints 3 lines on the console. The
        line pointer now points to line 0011.
*V3-4(CR)
 --------
        -user selects to view lines 3 through 4.
0003 SOME LINE OF DATA
0004 NEXT LINE OF DATA
        -Editor prints lines 3 through 4.
*
        -Editor prompts for a command.
```

7.0 EDITING LARGE FILES

Editing of large files is no different than editing small files. All commands are fully functional. However, diskette access may be required for certain operations and a slight delay may be apparent before the Editor responds.

8.0 EDITOR MESSAGES

If the user enters an unrecognizable file name, a syntax error will be indicated and the Editor will return to DOS. If the user enters an unrecognizable command, then the Editor will print a question mark and another command prompt:

EXAMPLE

```
*R20(CR)
-------
?*
```

All I/O errors to and from disk result in appropriate error messages. The original file should be backed up on another disk before using the Editor.

The Editor prompts with several other messages as editing progresses:

***NEW FILE - indicates that a new file is being created rather than editing of an already existing file.

***DATA MODE - indicates that lines of data are to be entered rather than Editor commands.

***TOF - indicates that the top of file (beginning of file) has been encountered.

***EOF - indicates that the end of file has been encountered.

***END OF EDITING - indicates that the Editor has successfully completed. Control is then returned to the DOS Operating System.

***END OF WINDOW. USE 'ADVANCE' TO SEE NEXT LINE - occurs only with the VIEW command. Follow the directions.

9.0 SAMPLE EDITING SESSION

The user is urged to follow the steps given here to become acquainted with the Editor.

```
>EDIT NEWFILE(CR)
 ---------------
        - user selects to run the Editor to create a new file.
SD SYSTEMS EDITOR V1.0
***NEW FILE
        -Editor indicates that a new file is being created.
***DATA MODE
        - Editor prompts for data lines to be input from
          the console device.  User begins keying in a
          program.
0001 ; A SIMPLE SAMPLE PROGRAM(CR)
     ------------------------------
0002  LD A,(LAB1)(CR)
     ----------------
0003  LD E,0(CR)
     -----------
0004  CALL SUB1 ;SOME COMMENT(CR)
     ----------------------------
0005 LOOP LD (HL),0 ;STUFF ZEROS(CR)
     -------------------------------
0006  INC HL(CR)
     -----------
0007  DNZ LOOP-$ ;LOOP FOR ALL(CR)
     ----------------------------
0008  END(CR)
     --------
0009 (CR)
     ----
        -user terminates DATA MODE.
*B99V20(CR)
 ----------
        -user backs up to beginning of file and
         views all lines.
.
.
.
***EOF
        -Editor indicates end of file encountered.
*L7(CR)
 ------
0007  DNZ LOOP-$ ;LOOP FOR ALL
        -user views line 7 and observes an error.
*C /DN/DJN/(CR)
 --------------
        -user modifies the line.
0007  DJNZ LOOP-$ ;LOOP FOR ALL
        -Editor prints the changed line
*Q(CR)
 -----
        -user terminates the editing session.
```

10.0 EDITOR COMMAND SUMMARY

```
An          advance n lines
Bn          backup n lines
Cn/s1/s2/ change n occurrences of s1 to s2
Dn          delete n lines
En          exchange n lines
Fn          turn on or turn off print flag
G file      get file and insert into current file
I           insert lines of data
Ln          go to line number n
Pn file     put n lines out to file
Q           quit, save all editing and return to DOS
Sn/s1/      search for n occurrences of s1
T           insert lines at top of file before first line
Vn          view n lines on the console
```

SD SYSTEMS RELOCATING Z80 ASSEMBLER VERSION 3.3
OPERATIONS MANUAL

1.0 INTRODUCTION

The SD SYSTEMS Z80 ASSEMBLER is provided on a standard CP/M compatible diskette. It provides the means for assembling Z80 programs. The Assembler (ZASM) reads standard Z80 source language (Mostek and Zilog definition) and outputs an assembly listing and object code on disk. The object code is in industry standard hexadecimal format extended for relocatable and linkable programs. The Assembler supports conditional assembly, a printed symbol table, and a printed cross reference table. The Assembler can assemble any length program limited only by the symbol table size which is based on available memory and available disk space. Typically over 300 symbols are allowed in one assembly.

Any Z80 based system which is running 32K CP/M compatible disk operating system (DOS) can use the Z80 Assembler.

2.0 COMMAND SUMMARY

```
ZASM file.ext / options (CR)
        - executes assembler to assemble a file
        - object output is on file.OBJ
        - listing output is on file.PRN

OPTIONS
-------
C - print cross reference table
K - no listing output
L - direct assembly listing out to listing device (no .PRN file
    is created)
N - no object output
P - pass 2 only
R - reset symbol table for pass 2 only operation
S - print symbol table
T - direct assembly listing out to console device
```

3.0 DEFINITIONS

In this manual, the following symbols are used:

- (CR) means carriage return.
- all user input is underlined.
- user input which is all upper case must be entered exactly as shown.
- user input which is lower case is variable.

SOURCE MODULE - the user's source program. Each source module is assembled into one object module by the Assembler. The end of a source module is defined by an 'END' statement or CP/M end of file code (1AH) on input.

OBJECT MODULE - the object output of the Assembler for one source module. The object module contains linking information, address and relocating information, machine code, and checksum information for use by the SD SYSTEMS Linker. The object module is in ASCII. The object module is output to a disk file with extension OBJ. The SD SYSTEMS Linker must then be used to link and relocate one or more object modules into a module loadable by the DOS. See the SD SYSTEMS Linkder Operations Manual for more details.

LOAD MODULE - the absolute machine code of one complete program. The load module is defined on disk as an absolute object file with extension HEX. The file may be loaded by the DOS loader. It is created by the SD SYSTEMS Linker from one or more relocatable object modules (secondary file name OBJ) which were created by the Z80 Assembler.

LOCAL SYMBOL - a symbol in a source module which appears in the label field of a source statement.

INTERNAL SYMBOL - a symbol in a source (and object) module which is to be made known to all other modules which are linked with it by the Linker. An internal symbol is also called global, defined, public, or common. Internal symbols are defined by the GLOBAL pseudo-op. An internal symbol must appear in the label field of the same source module. Internal symbols are assumed to be addresses, not constants, and they will be relocated when linked by the Linker.

EXTERNAL SYMBOL - a symbol which is used in a source (and object) module but which is not a local symbol (does not appear in the label field of a statement). External symbols are defined by the GLOBAL pseudo-op. External symbols may not appear in an expression which uses operators. An external symbol is a reference to a symbol that exists and is defined as internal in another program module.

GLOBAL DEFINITION - both internal and external symbols are defined as GLOBAL in a source module. The Assembler determines which are internal and which are external.

POSITION INDEPENDENT - a program which can be placed anywhere in memory. It does not require relocating information in the object module.

ABSOLUTE - a program which has no relocating information in the object module. An absolute program which is not position independent can be loaded only in one place in memory in order to work properly.

RELOCATABLE - a program which has extra information in the object module which allows the Linker to place the program anywhere in memory.

LINKABLE - a program which has extra information in the object module which defines internal and external symbols. The Linker uses the information to connect, resolve, or link, external references to internal symbols.

4.0 USING THE ASSEMBLER

The SD SYSTEMS Z80 ASSEMBLER is resident on a CP/M compatible system diskette. The user first prepares his source module using the SD SYSTEMS Editor. To use the Z80 Assembler, enter the following command:

A>ZASM file.ext / options (CR)

where 'file' is the primary file name and 'ext' is the secondary file name of the file to be assembled. Also where '/' allows the batching of several assemblies without operator interaction and 'options' are those described in paragraph 4.1.

Example: A>ZASM FILE.ASM /CL (CR)

This would cause the Z80 source program file named "FILE.ASM" to be assembled with the 'C' (cross reference table) and 'L' (listing on printer) options.

The object output of the Assembler is sent to the disk on file.OBJ, and the listing output is sent to the disk on file.PRN unless T or L options are specified. One or more object files from the Assembler may be linked and relocated by using the SD SYSTEMS Linker, which produces an absolute object file with extension HEX. The absolute object file may then be loaded via the DOS loader, and the listing file may be printed using XFER. Note that if the (/) is omitted, the "options" will be requested as specified below, which prevents continuous batch operation. Therefore, if no options are required, enter the slash (/) on the command line with no options following.

SD SYSTEMS Z80 ASSEMBLER V3.3. OPTIONS?

Options are described in paragraph 4.1. If no options are to be entered, the user enters 'carriage return'. The Assembler makes two passes over the source file. At the end of the first pass the following message is printed on the user console:

PASS 1 DONE

At the end of the assembly, the Assembler prints the total number of errors (in decimal) found:

ERRORS=nnnn

Control is then returned to the DOS console processor (A>).

4.1 ASSEMBLER OPTIONS

When the Assembler outputs the message:

OPTIONS?

the user may enter any of the following codes, terminated with a carriage return:

- C - cross reference table - prints a cross reference table of all the symbols at the end of the assembly listing.
- K - no listing - this suppresses the assembly listing. All errors are output to the user console for this option.
- L - list to listing device - this option directs the assembly listing out to the listing device rather than to a disk file.
- N - no object output - this suppresses object output from the Assembler.
- P - pass 2 only - this option selects and runs only pass 2 of the Assembler. The symbol table is left intact from a previous run of the Assembler.
- R - reset the symbol table - clears the symbol table of all previous symbol references. This operation is automatically done for pass 1. It is used primariy for single pass operation (see paragraph 5.1).
- S - symbol table - prints a symbol table at the end of the assembly listing.
- T - list to console device - this option directs the assembly listing out to the console device rather than to a disk file.

4.2 ERROR MESSAGES

Any error which is found is denoted in the assembly listing. A message is printed immediately after the statement in error. All messages are self-explanatory.

```
EXAMPLE          H2:  LC  A,B
                 ***** ERROR ***** BAD OPCODE
```

Certain errors abort the Assembler when they are encountered. Abort error messages are output only to the user console. Control is immediately returned to the DOS console processor (A>). Abort errors may occur during pass 1 or pass 2.

4.3 OBJECT OUTPUT

The object output from the Assembler is put on diskette to the same primary file name as the source input file, with a secondary file name of 'OBJ'. One or more object modules may be linked and relocated by the SD SYSTEMS Linker to produce an absolute object file with a secondary file name of 'HEX'. This file may then be loaded by the DOS loader.

4.4 ASSEMBLY LISTING OUTPUT

The assembly listing is put on diskette to the same primary file name as the source input file, with a secondary file name of 'PRN'. The user may insert tab characters in the source to obtain columns in the assembly listing. The value of each equated symbol will be printed with a pointer (>) next to it. The statement number and page number are printed in decimal. Assembler directives (see paragraph 6.4.1.) do not appear in the assembly listing, but they are assigned statement numbers. If the no listing option is selected, errors will be output to the user console. Any addresses which are relocatable will have a prime (') printed next to them.

5.0 ADVANCED OPERATIONS

5.1 PASS 2 OPERATION (SINGLE PASS OPERATION)

The Z80 Assembler can be used as a single pass assembler under the following restrictions:

1. No forward symbol references are allowed.
2. The NAME pseudo-op is not allowed.
3. A cross reference table is not selected.

The Assembler will correctly assemble Z80 programs under the above restrictions using the pass 2 only option ('P'). This is useful for assembling data tables and certain types of programs. The Assembler symbol table should be reset to assure proper operation in this mode by using the 'R' option.

5.2 ASSEMBLING SEVERAL SOURCE MODULES TOGETHER

Several source modules may be assembled together to form one object module. The 'INCLUDE' pseudo-op may be used any number of times in one module to properly sequence a set of source modules.

EXAMPLE

```
NAME MYFILE     ;name of final object module
INCLUDE FILE1
INCLUDE FILE2
INCLUDE FILE3
END
```

- the object module named 'MYFILE' will be built by the Assembler from FILE1 + FILE2 + FILE3.

The assembler has a seperate counter for included files. The assembler listing indicates included statements by placing a "+" in front of the line number.

6.0 Z80 ASSEMBLY LANGUAGE

An assembly language program (source module) consists of labels, opcodes, pseudo-ops, and comments in a sequence which defines the user's program. The assembly language conventions are described in the following paragraphs.

6.1 DELIMITERS

Labels, opcodes, operands, and pseudo-ops must be separated from each other by one or more commas, spaces, or tab characters (ASCII 09H). The label may be separated from the opcode by a colon, only, if desired.

6.2 LABELS

A label is composed of one or more characters. If more than 6 characters are used for the label, only the first 6 are recognized by the Assembler. The characters in the label cannot include ' ( ) * + - / , = < > . : ; or space. In addition, the first character cannot be a number (0-9). A label can start in any column if immediately followed by a colon (:). It does not require a colon if started in column one.

```
EXAMPLE  allowed      not allowed
         -------      -----------
         LAB          9LAB        ;STARTS WITH ILLEGAL CHARACTER
         L923         L)AB        ;CONTAINS ILLEGAL CHARACTER
         $25          L:ABC       ;CONTAINS ILLEGAL CHARACTER
```

6.3 OPCODES

The full set of Z80 opcodes is documented in the 'Z80 PROGRAMMING MANUAL' (which is available from SD SYSTEMS).

6.4 PSEUDO-OPS

Pseudo-ops are used to define assembly time parameters. Pseudo-ops appear like Z80 opcodes in the source module. Several pseudo-ops require a label. The following pseudo-ops are recognized by the Assembler:

DEFB n,n,n...- define byte - defines the contents of successive bytes to be the expressions n.

label DEFL nn - define label - sets the value of the label to the expression nn; may be repeated in the program with different values for the same label. At any point in the program, the label assumes the last previously defined value.

DEFM 'aa'- define message - defines the contents of successive bytes of memory to be the ASCII equivalent code of characters within quotes. Up to 63 characters may be in one message. Quote characters in the message may be defined by two successive quote characters ('').

DEFS nn- define storage - reserves nn bytes of memory starting at the current program counter, where nn is an expression. When loaded, these bytes will contain what was previously in memory. This pseudo-op cannot be used at the start or at the end of a program to reserve storage.

DEFW nn,nn,nn... - define word - defines the contents of successive two -byte words to be the value of expressions nn. The least significant byte is located at the current program counter address, and the most significant byte follows it.

END- end statement - defines the last line of the program. The 'END' statement is not required.

ENDIF- end of conditional assembly - re-enables assembly of subsequent statements after an IF pseudo-op.

label EQU nn- equate - sets the value of a label to the expression nn; can occur only once for any label.

GLOBAL symbol - define global symbol - any symbol which is to be made known among several separately assembled modules must appear in this type of statement. The Assembler determines if the symbol is internal (defined as a label in the program), or external (used in the program but not defined as a label).

IF nn- conditional assembly - if the expression nn is true (non-zero), the IF pseudo-op is ignored. If the expression is false (zero), the assembly of subsequent statements is disabled until an ENDIF pseudo-op. IF statements cannot be nested.

INCLUDE file.ext - include source statements from another file - allows source statements from another input file to be included within the body of the given program. If the file cannot be opened properly, then assembly is aborted. The source module to be included must not end with an END pseudo-op (otherwise, assembly would be terminated). The INCLUDE pseudo-op cannot be nested.

NAME symbol- module name - this pseudo-op defines the name of the program (source and object). The name is placed in the heading of the assembly listing and in the first record of the object output. The module name defaults to 6 blanks.

PSECT op - program section - may appear only once at the start of a source module. This pseudo-op defines the program module attributes for the following operands:

REL - relocatable program (default)
ABS - absolute program. No relocating information is generated in the object module. The module will be linked where it is origined.

ORG nn- origin - sets the program counter to the value of the expression nn. If more than one ORG statement is used in a source module, then the expression nn is a given ORG statement must be greater than a previous ORG statement.

6.4.1 ASSEMBLER DIRECTIVES

Assembler directives are pseudo-ops which are designed to format the assembly listing.

```
EJECT    - eject a page of assembly listing.
LIST     - turn assembly listing on (default).
NLIST        - turn assembly listing off.
TITLE s      - place title of characters 's' at top
               of each page of assembly listing.  s can
               be up to 32 characters long.
```

6.5 OPERANDS

There may be zero, one, or more operands in a statement depending on the opcode or pseudo-op used. Operands in the Assembler may take the following forms:

A GENERIC OPERAND, such as the letter 'A', which stands for the accumulator. The following are Z80 generic operands:

```
A - Accumulator
B - B register
C - C register
D - D register
E - E register
F - F register
H - H register
L - L register

AF - AF register pair
AF' - AF' register pair
BC - BC register pair
DE - DE register pair
HL - HL register pair

SP - stack pointer register
$ - program counter

I - I register
R - refresh register

IX - IX index register
IY - IY index register
```

NZ - not zero
Z - zero
NC - not carry
C - carry
PO - parity odd/not overflow
PE - parity even/overflow
P - sign positive
M - sign negative

A CONSTANT. The constant must be in the range 0 thru 0FFFFH. It can be in the following forms:

DECIMAL	- (default); any number can be denoted as decimal by following it with the letter D. Eg: 35, 249D
HEXADECIMAL	- must begin with a number (0-9) and end with the letter H. Eg: 0AF1H
OCTAL	- must end with the letter Q or O. Eg: 377Q, 277O
BINARY	- must end with the letter B. Eg: 0110110B
ASCII	- letters enclose in quote marks will be converted to their ASCII equivalent value. Eg: 'A' = 41H

A LABEL which appears elsewhere in the program. Note that labels cannot be defined by labels which have not yet appeared in the program:

EXAMPLE	allowed	not allowed
	I EQU 7	L EQU H
	H EQU I	H EQU I
	L EQU H	I EQU 7

AN EXPRESSION. The Assembler accepts a wide range of expressions in the operand field of a statement. All expressions are evaluated left to right constrained by the hierarchies shown below. Parentheses may be used to ensure correct expression evaluation.

operation	operator	hierarchy
equal to	= or .EQ	0
signed less than	<	0
signed greater than	>	0
signed less than or equal to	<= or =<	0
signed greater than or equal to	>= or =>	0
not equal	>< or <> or .NE.	0
unsigned less than	.LT.	0
unsigned greater than	.GT.	0
unsigned less than or equal to	.LE.	0
unsigned greater than or equal	.GE.	0
reset overflow	.RES.	0

unary plus	+	1
unary minus	-	1
logical NOT (one's complement)	.NOT.	1
multiplication	*	2
division	/	2
addition	+	3
subtraction	-	3
logical AND	.AND.	4
logical OR	.OR.	4
logical XOR	.XOR.	4
logical shift right	.SHR.	4
logical shift left	.SHL.	4

All operands and expressions are converted to 16-bit values. The only exception to this is when expressions take the form:

'character string 1'='character string 2'

In this case, character string 1 and character string 2 are compared character by character for a match. If they do not match, then the value of the expression is false. If they have the same length and match, then the value of the expression is true (0FFFFH).

The reset operator (.RES.) unconditionally resets any overflow error in an operand expression. The shift operators shift their first argument right or left by the number of bit positions given in their second argument. Zeros are shifted into the vacated bit positions. The negative (2's complement) of an expression may be formed by preceding it with a minus sign. The one's complement of an expression may be formed by preceding it with the .NOT. operator.

The symbol $ is used to represent the value of the program counter of the current instruction.

```
EXAMPLE         SYM1 EQU $
                          will equate SYM to current value of
                          program counter.
```

All versions after (and including) V3.1 have been enhanced making it no longer necessary to have "-$" after labels in the relative jump instructions (JR) and the "DJNZ" instruction. Thus the following instructions are assembled the same.

JR TAG-$ and JR TAG

DJNZ TAG-$ and DJNZ TAG

Note that enclosing an expression wholly in parentheses indicates a memory address. The contents of the memory address equivalent to the expression value will be used as the operand value.

The allowed range of an expression depends on the context of its use. For example, the limits on a relative jump instruction are -126 and +129 bytes.

6.6 COMMENTS

A comment is defined as any characters following a semicolon (;) in a line. A semicolon in quotes in an operand is treated as an expression rather than a comment starter. Comments are ignored by the Assembler but they are printed in the assembly listing. Comments can begin in any column.

6.7 ABSOLUTE MODULE RULES

The pseudo-op 'PSECT ABS' defines a module to be absolute. The program will be loaded in the exact addresses at which it is assembled. This is useful for defining constants, a common block of global symbols, or a software driver whose position must be known. This method can be used to define a list of global constants as follows:

EXAMPLE

```
            PSECT   ABS     ;ABSOLUTE ASSEMBLY
            GLOBAL  AA
AA          EQU     0E3H
            GLOBAL  AX
AX          EQU     0AF3H
            END
```

6.8 RELOCATABLE MODULE RULES

1. Programs default to relocatable if the 'PSECT ABS' statement is not used or if 'PSECT REL' is used.

2. Only those values which are 16-bit address values will be relocated. 16-bit constants will not be relocated.

EXAMPLE

```
AA          EQU   0A13H       ;ABSOLUTE VALUE
            LD    A,(AA)      ;AA NOT RELOCATED
AR          EQU   $           ;RELOCATABLE VALUE
            LD    HL,(AR)     ;AR WILL BE RELOCATED UPON LINKING
```

3. Relocatable quantities may not be used as 8-bit operands. This restriction exists because only 16-bit operands are relocated by the SD SYSTEMS Linker.

EXAMPLE

```
LAB         EQU   $           ;RELOCATABLE VALUE
            DEFB  LAB         ;NOT ALLOWED
            LD    A,(IX+LAB)  ;NOT ALLOWED
            LD    A,(LAB)     ;ALLOWED
            LD    HL,LAB      ;ALLOWED
```

4. Labels equated to labels which are constants will be treated as constants. Labels equated to labels which are relocatable addresses will be relocated.

EXAMPLE

```
        B8      EQU  20H          ;CONSTANT
        C8      EQU  B8           ;CONSTANT
                LD   A,(C8)       ;C8 WILL NOT BE RELOCATED
        AR      EQU  $            ;RELOCATABLE ADDRESS
        BR      EQU  AR           ;RELOCATABLE
                LD   A,(BR)       ;BR WILL BE RELOCATED
```

5. External symbols in a relocatable program are marked relocatable, except for the first usage. The code for external symbols is actually a backward link list through the object code.

6.9 GLOBAL SYMBOL HANDLING

A global symbol is a symbol which is known by more than on module. A global symbol has its value defined in one module. It can be used by that module and by any other module which is linked with it by the SD SYSTEMS Linker. A global symbol is defined as such by the GLOBAL pseudo -op.

An internal symbol is one which is defined as global and also appears as a label in the same program. The symbol value is thus defined for all programs which use that symbol. An external symbol is one which is defined as global but does NOT appear as a label in the same program.

EXAMPLE

```
                GLOBAL  SYM1      ;DEFINE GLOBAL SYMBOL
                CALL SYM1
                .
                .
                .
                END
                     - SYM1 is an external symbol
```

EXAMPLE

```
                GLOBAL  SYM1      ;DEFINE GLOBAL SYMBOL
        SYM1    EQU  $
                LD   A,(SYM1)
                .
                .
                .
                END
                     - SYM1 is an internal symbol.  Its value
                     is the address of the LD instruction.
```

If these two programs were assembled and then linked by the SD SYSTEMS Linker, then all global symbol references from the first program would be 'resolved'. This means that each address in which an external symbol was used would be modified to the value of the corresponding internal symbol. The linked programs would be equivalent (using our example) to one program written as follows.

EXAMPLE

```
                CALL SYM1
                .
                .
                .
        SYM1    EQU  $
                LD   A,(SYM1)
                .
                .
                .
                END
```

Global symbols are used to allow large programs to be broken up into smaller modules. The smaller modules are used to ease programming, facilitate changes, or allow programming by different members of the same team.

6.9.1 GLOBAL SYMBOL RULES

1. An external symbol cannot appear in an expression which uses operators.

EXAMPLE

```
                GLOBAL  SYM1            ;EXTERNAL SYMBOL
                CALL    SYM1            ;OK
                LD      HL,(SYM1+2)     ;NOT ALLOWED
```

2. An external symbol is always considered to be a 16-bit address. Therefore, an external symbol cannot appear in an instruction requiring an 8-bit operand.

EXAMPLE

```
                GLOBAL  SYM1    ;EXTERNAL SYMBOL
                CALL    SYM1    ;OK
                LD      A,SYM1  ;NOT ALLOWED
```

3. An external symbol cannot appear in the operand field of an EQU or DEFL statement.

4. An internal symbol is always marked relocatable in a relocatable assembly. This point is important because an internal symbol will always be relocated even though it looks like a constant. To define constant internal symbols, create an absolute assembly via the PSECT ABS pseudo-op.

5. For a set of modules to be linked together, no duplication of internal symbol names is allowed. That is, an internal symbol can be defined only once in a set of modules to be linked together.

7.0 TECHNICAL INFORMATION

The Assembler is resident on a CP/M compatible system diskette and, when loaded, starts at location 100H. Assembler variables are placed in memory at the top of the Assembler. The symbol table is placed in RAM starting at the end of the Assembler and ending at the starting address of DOS. Typically, more than 300 symbols are allowed per program.

SD SYSTEMS LINKER VERSION 3.1
OPERATIONS MANUAL

1.0 INTRODUCTION

The SD SYSTEMS Linker is provided on a standard CP/M compatible diskette. The Linker (LINK) provides the means for linking object modules produced by the Z80 Assembler (ZASM). The Linker concatenates modules together and resolves global symbol references which provide communication between modules. The Linker produces a load module containing "hex" format machine code which may be read by the DOS LOAD commmand. The LOAD command reads a load module (secondary filename = HEX) and produces a memory image file (secondary filename = COM) which can be executed by the disk operating system (DOS).

2.0 COMMAND SUMMARY

In this manual, the following symbols are used:

- (CR) means carriage return.
- all user input is underlined.
- user input which is all upper case must be entered exactly as shown.
- user input which is lower case is variable.

A>LINK filename 1, filename 2,....filename N / options (CR)

- Links object input files (secondary filename=OBJ)
- Produces a LOAD module (secondary filename=HEX)
- As an option creates a cross reference file (secondary filename=CRS)

OPTIONS

C - Produces an output file containing a global cross reference table and a load map.

U - Lists all undefined global symbols

A - Is used to specify where the first relocatable module will be positioned. If one or more absolute modules precede the first relocatable module, they will be positioned at their "ORG" address while the relocateable module will be positioned starting at the address (hexadecimal) specified using the 'A' option. Note that if more options are entered after "A=hhhh, a comma (,) must be entered to seperate the address specification and the following option.

3.0 DEFINITIONS

SOURCE MODULE - the user's source program. Each source module is assembled into one object module by the Assembler.

OBJECT MODULE - the object output of the Assembler for one source module. The object module contains linking information, address and relocating information, machine code, and checksum information for use by the Linker. The object module is in ASCII. The object module is output to a disk file with extension OBJ.

LOAD MODULE - the absolute machine code of one complete program. The load module is defined on disk as an absolute object file with secondary filename of HEX. A Load module is produced by the Linker.

GLOBAL DEFINITION - both internal and external symbols are defined as GLOBAL in a source module. The Assembler determines which are internal and which are external. (See Z80 Assembler description of internal and external symbols).

ABSOLUTE - a program which has no relocating information in the object module. An absolute program which is not position independent can be loaded only in one place in memory in order to work properly.

RELOCATABLE - a program which has extra information in the object module which allows the Linker to place the program anywhere in memory.

4.0 LINKER OPERATION

During Pass 1 the Linker reads one or more object input files and places the global symbol definitions in the Linker symbol table. In PASS 2 global symbol references are resolved and an output Load file is produced. The Load file has the same primary filename as the first object input file (filename 1) and has a secondary filename of HEX. If the cross reference option is specified a cross reference file is produced. The cross reference file has the same primary filename name the first object input file (filename 1) and has a secondary filename of CRS.

In Pass 2 as each object input module is read its beginning and ending address in memory is printed on the console. The module type is also listed as either absolute or relocatable (ABS/REL). Absolute modules are always positioned at their starting address in memory as defined by the ORG pseudo-op. Relocatable modules are positioned at the next location after the end address of the previous module. If the first input module is relocatable, it is positioned by the starting link address. If the starting link address is not specified by the A option it assumes a value of 0.

It is suggested that the first object input module read by the Linker have a starting address of 100H for operation with the DOS. This starting address should also serve as the entry point for the combined Load module. A starting address of 0100H can be created either with the ORG pseudo-op or the Linker A option. The DOS loads and begins execution of RAM image files at location 0100H.

When absolute modules are being linked together, the files in the LINK command must appear in sequential order according to their starting addresses in memory. If an absolute module is encountered having a starting address lower in memory than a previous module a module sequence error message will be generated. Furthermore, if a source module contains more than one ORG statement, the address used in any given ORG statement must be greater than a previous ORG statement.

5.0 EXAMPLE OF LINK COMMAND

EXAMPLE 1. Link the relocatable object modules MAIN.OBJ, SUB1.OBJ, ,SUB2.OBJ,SUB3.OBJ together starting at 100H producing the LOAD module MAIN.HEX. Also generate a global cross referecne table and a load map in the file MAIN.CRS.

```
A>LINK MAIN,SUB1,SUB2,SUB3 /C A=100    (CR)
```

Note: With LINK (Version 3.1) Linker options can be entered on the command line following the last file name in the list of files to be linked.This makes it possible to batch up several links to be done without operator interaction. If the slash (/) is omitted, the options will be requested as shown below

```
OPTIONS? A C (CR)

ENTER STARTING LINK ADDRESS> 100  (CR)

MAIN    .OBJ
SUB1    .OBJ
SUB2    .OBJ
SUB3    .OBJ
UNDEFINED SYMBOLS 00
PASS 2
MAIN    .OBJ    REL    BEG ADDR 0100    END ADDR 0125
SUB1    .OBJ    REL    BEG ADDR 0126    END ADDR 01CD
SUB2    .OBJ    REL    BEG ADDR 01CE    END ADDR 01E8
SUB3    .OBJ    REL    BEG ADDR 01E9    END ADDR 0212

A>
```

EXAMPLE 2. Using the load module MAIN.HEX created in Example 1 and the DOS LOAD command create a memory image file and begin execution of MAIN.

```
A>LOAD MAIN.HEX   (CR)
  --------------------
A>MAIN   (CR)
  -----------
```

NOTE: Execution of MAIN has been started.

EXAMPLE 3. Using the DOS TYPE command list the global cross reference table and the load map for the modules linked in Example 1.

```
A>TYPE MAIN.CRS   (CR)
  --------------------
```

```
LOAD MAP

MAIN     .OBJ    REL    BEG ADDR 0100    END ADDR 0125
SUB1     .OBJ    REL    BEG ADDR 0126    END ADDR 01CD
SUB2     .OBJ    REL    BEG ADDR 01CE    END ADDR 01E8
SUB3     .OBJ    REL    BEG ADDR 01E9    END ADDR 0212


 GLOBAL CROSS REFERENCE TABLE

 SYMBOL ADDR  REFERENCES
 CRLF   E59C  020C 01E6
 MAIN   0100
 MODNO  01FB  01D4 01D1 012C 0129 010C 0109
 MSGBEG 013F  0101
 MSGEND 0165  011E
 MSGMAI 018A  010F
 MSGMOD 01C2  0201
 MSGSB2 0195  01D7
 MSGSB3 019B  01F2
 PRINT  01E0  01F5 013C 0132 0121 0112 0104
 PTEST  0138  01F8 01DD
 SUB1   0126  0115
 SUB123 020F
 SUB2   01CE  0118
 SUB3   01E9  011B
```

APPENDIX A

Z80 OPCODE LISTINGS

```
 ADDR   CODE      STMT SOURCE STATEMENT

                  0002 ; PSEUDO OPS
                  0003 ;
 >0000            0004          ORG     0
                  0005          PSECT   REL
                  0006 ;
'0000   AA        0007          DEFB    0AAH
'>0001            0008 L2       DEFL    $
 >55AA            0009 L2       DEFL    55AAH
'0001   41424344  0010          DEFM    'ABCD'
 >0005            0011 NN       DEFS    2
'0007   BBAA      0012          DEFW    0AABBH
 >AABB            0013 L1       EQU     0AABBH
 >0005            0014 IND      EQU     5
 >0020            0015 N        EQU     20H
 >0030            0016 DIS      EQU     30H
                  0017          GLOBAL  NN
                  0018          IF      0
                  0019 ; SHOULD NOT BE ASSEMBLED
                  0020          LD      A,B
                  0021          ENDIF
                  0022          IF      1
                  0023 ; SHOULD BE ASSEMBLED
'0009   78        0024          LD      A,B
                  0025          ENDIF
                  0026 ; TURN LISTING OFF
                  0031 ; LISTING SHOULD BE ON
                  0032 ;
                  0033 ;
                  0034 ;
                  0035 ; Z80 OPCODES
                  0036 ;
'000B   8E        0037          ADC     A,(HL)
'000C   DD8E05    0038          ADC     A,(IX+IND)
'000F   FD8E05    0039          ADC     A,(IY+IND)
'0012   8F        0040          ADC     A,A
'0013   88        0041          ADC     A,B
'0014   89        0042          ADC     A,C
'0015   8A        0043          ADC     A,D
'0016   8B        0044          ADC     A,E
'0017   8C        0045          ADC     A,H
'0018   8D        0046          ADC     A,L
'0019   CE20      0047          ADC     A,N
'001B   ED4A      0048          ADC     HL,BC
'001D   ED5A      0049          ADC     HL,DE
'001F   ED6A      0050          ADC     HL,HL
'0021   ED7A      0051          ADC     HL,SP
                  0052 ;
'0023   86        0053          ADD     A,(HL)
'0024   DD8605    0054          ADD     A,(IX+IND)
'0027   FD8605    0055          ADD     A,(IY+IND)
'002A   87        0056          ADD     A,A
'002B   80        0057          ADD     A,B
'002C   81        0058          ADD     A,C
'002D   82        0059          ADD     A,D
'002E   83        0060          ADD     A,E
'002F   84        0061          ADD     A,H
'0030   85        0062          ADD     A,L
'0031   C620      0063          ADD     A,N
```

```
 ADDR  CODE      STMT SOURCE STATEMENT

'0033  09        0064          ADD     HL,BC
'0034  19        0065          ADD     HL,DE
'0035  29        0066          ADD     HL,HL
'0036  39        0067          ADD     HL,SP
'0037  DD09      0068          ADD     IX,BC
'0039  DD19      0069          ADD     IX,DE
'003B  DD29      0070          ADD     IX,IX
'003D  DD39      0071          ADD     IX,SP
'003F  FD09      0072          ADD     IY,BC
'0041  FD19      0073          ADD     IY,DE
'0043  FD29      0074          ADD     IY,IY
'0045  FD39      0075          ADD     IY,SP
                 0076 ;
'0047  A6        0077          AND     (HL)
'0048  DDA605    0078          AND     (IX+IND)
'004B  FDA605    0079          AND     (IY+IND)
'004E  A7        0080          AND     A
'004F  A0        0081          AND     B
'0050  A1        0082          AND     C
'0051  A2        0083          AND     D
'0052  A3        0084          AND     E
'0053  A4        0085          AND     H
'0054  A5        0086          AND     L
'0055  E620      0087          AND     N
                 0088 ;
'0057  CB46      0089          BIT     0,(HL)
'0059  DDCB0546  0090          BIT     0,(IX+IND)
'005D  FDCB0546  0091          BIT     0,(IY+IND)
'0061  CB47      0092          BIT     0,A
'0063  CB40      0093          BIT     0,B
'0065  CB41      0094          BIT     0,C
'0067  CB42      0095          BIT     0,D
'0069  CB43      0096          BIT     0,E
'006B  CB44      0097          BIT     0,H
'006D  CB45      0098          BIT     0,L
                 0099 ;
'006F  CB4E      0100          BIT     1,(HL)
'0071  DDCB054E  0101          BIT     1,(IX+IND)
'0075  FDCB054E  0102          BIT     1,(IY+IND)
'0079  CB4F      0103          BIT     1,A
'007B  CB48      0104          BIT     1,B
'007D  CB49      0105          BIT     1,C
'007F  CB4A      0106          BIT     1,D
'0081  CB4B      0107          BIT     1,E
'0083  CB4C      0108          BIT     1,H
'0085  CB4D      0109          BIT     1,L
                 0110 ;
'0087  CB56      0111          BIT     2,(HL)
'0089  DDCB0556  0112          BIT     2,(IX+IND)
'008D  FDCB0556  0113          BIT     2,(IY+IND)
'0091  CB57      0114          BIT     2,A
'0093  CB50      0115          BIT     2,B
'0095  CB51      0116          BIT     2,C
'0097  CB52      0117          BIT     2,D
'0099  CB53      0118          BIT     2,E
'009B  CB54      0119          BIT     2,H
'009D  CB55      0120          BIT     2,L
                 0121 ;
```

```
 ADDR   CODE      STMT SOURCE STATEMENT

'009F   CB5E      0122          BIT     3,(HL)
'00A1   DDCB055E  0123          BIT     3,(IX+IND)
'00A5   FDCB055E  0124          BIT     3,(IY+IND)
'00A9   CB5F      0125          BIT     3,A
'00AB   CB58      0126          BIT     3,B
'00AD   CB59      0127          BIT     3,C
'00AF   CB5A      0128          BIT     3,D
'00B1   CB5B      0129          BIT     3,E
'00B3   CB5C      0130          BIT     3,H
'00B5   CB5D      0131          BIT     3,L
                  0132 ;
'00B7   CB66      0133          BIT     4,(HL)
'00B9   DDCB0566  0134          BIT     4,(IX+IND)
'00BD   FDCB0566  0135          BIT     4,(IY+IND)
'00C1   CB67      0136          BIT     4,A
'00C3   CB60      0137          BIT     4,B
'00C5   CB61      0138          BIT     4,C
'00C7   CB62      0139          BIT     4,D
'00C9   CB63      0140          BIT     4,E
'00CB   CB64      0141          BIT     4,H
'00CD   CB65      0142          BIT     4,L
                  0143 ;
'00CF   CB6E      0144          BIT     5,(HL)
'00D1   DDCB056E  0145          BIT     5,(IX+IND)
'00D5   FDCB056E  0146          BIT     5,(IY+IND)
'00D9   CB6F      0147          BIT     5,A
'00DB   CB68      0148          BIT     5,B
'00DD   CB69      0149          BIT     5,C
'00DF   CB6A      0150          BIT     5,D
'00E1   CB6B      0151          BIT     5,E
'00E3   CB6C      0152          BIT     5,H
'00E5   CB6D      0153          BIT     5,L
                  0154 ;
'00E7   CB76      0155          BIT     6,(HL)
'00E9   DDCB0576  0156          BIT     6,(IX+IND)
'00ED   FDCB0576  0157          BIT     6,(IY+IND)
'00F1   CB77      0158          BIT     6,A
'00F3   CB70      0159          BIT     6,B
'00F5   CB71      0160          BIT     6,C
'00F7   CB72      0161          BIT     6,D
'00F9   CB73      0162          BIT     6,E
'00FB   CB74      0163          BIT     6,H
'00FD   CB75      0164          BIT     6,L
                  0165 ;
'00FF   CB7E      0166          BIT     7,(HL)
'0101   DDCB057E  0167          BIT     7,(IX+IND)
'0105   FDCB057E  0168          BIT     7,(IY+IND)
'0109   CB7F      0169          BIT     7,A
'010B   CB78      0170          BIT     7,B
'010D   CB79      0171          BIT     7,C
'010F   CB7A      0172          BIT     7,D
'0111   CB7B      0173          BIT     7,E
'0113   CB7C      0174          BIT     7,H
'0115   CB7D      0175          BIT     7,L
                  0176 ;
'0117   DC0500'   0177          CALL    C,NN
'011A   FC0500'   0178          CALL    M,NN
'011D   D40500'   0179          CALL    NC,NN
```

```
 ADDR  CODE      STMT SOURCE STATEMENT

'0120  CD0500'   0180         CALL    NN
'0123  C40500'   0181         CALL    NZ,NN
'0126  F40500'   0182         CALL    P,NN
'0129  EC0500'   0183         CALL    PE,NN
'012C  E40500'   0184         CALL    PO,NN
'012F  CC0500'   0185         CALL    Z,NN
                 0186 ;
'0132  3F        0187         CCF
                 0188 ;
'0133  BE        0189         CP      (HL)
'0134  DDBE05    0190         CP      (IX+IND)
'0137  FDBE05    0191         CP      (IY+IND)
'013A  BF        0192         CP      A
'013B  B8        0193         CP      B
'013C  B9        0194         CP      C
'013D  BA        0195         CP      D
'013E  BB        0196         CP      E
'013F  BC        0197         CP      H
'0140  BD        0198         CP      L
'0141  FE20      0199         CP      N
                 0200 ;
'0143  EDA9      0201         CPD
'0145  EDB9      0202         CPDR
'0147  EDA1      0203         CPI
'0149  EDB1      0204         CPIR
                 0205 ;
'014B  2F        0206         CPL
                 0207 ;
'014C  27        0208         DAA
                 0209 ;
'014D  35        0210         DEC     (HL)
'014E  DD3505    0211         DEC     (IX+IND)
'0151  FD3505    0212         DEC     (IY+IND)
'0154  3D        0213         DEC     A
'0155  05        0214         DEC     B
'0156  0B        0215         DEC     BC
'0157  0D        0216         DEC     C
'0158  15        0217         DEC     D
'0159  1B        0218         DEC     DE
'015A  1D        0219         DEC     E
'015B  25        0220         DEC     H
'015C  2B        0221         DEC     HL
'015D  DD2B      0222         DEC     IX
'015F  FD2B      0223         DEC     IY
'0161  2D        0224         DEC     L
'0162  3B        0225         DEC     SP
                 0226 ;
'0163  F3        0227         DI
                 0228 ;
'0164  102E      0229         DJNZ    DIS
                 0230 ;
'0166  FB        0231         EI
                 0232 ;
'0167  E3        0233         EX      (SP),HL
'0168  DDE3      0234         EX      (SP),IX
'016A  FDE3      0235         EX      (SP),IY
'016C  08        0236         EX      AF,AF'
'016D  EB        0237         EX      DE,HL
```

```
 ADDR  CODE      STMT SOURCE STATEMENT

'016E  D9        0238          EXX
                 0239 ;
'016F  76        0240          HALT
                 0241 ;
'0170  ED46      0242          IM      0
'0172  ED56      0243          IM      1
'0174  ED5E      0244          IM      2
                 0245 ;
'0176  ED78      0246          IN      A,(C)
'0178  DB20      0247          IN      A,(N)
'017A  ED40      0248          IN      B,(C)
'017C  ED48      0249          IN      C,(C)
'017E  ED50      0250          IN      D,(C)
'0180  ED58      0251          IN      E,(C)
'0182  ED70      0252          IN      F,(C)
'0184  ED60      0253          IN      H,(C)
'0186  ED68      0254          IN      L,(C)
                 0255 ;
'0188  34        0256          INC     (HL)
'0189  FD3405    0257          INC     (IY+IND)
'018C  DD3405    0258          INC     (IX+IND)
'018F  3C        0259          INC     A
'0190  04        0260          INC     B
'0191  03        0261          INC     BC
'0192  0C        0262          INC     C
'0193  14        0263          INC     D
'0194  13        0264          INC     DE
'0195  1C        0265          INC     E
'0196  24        0266          INC     H
'0197  23        0267          INC     HL
'0198  DD23      0268          INC     IX
'019A  FD23      0269          INC     IY
'019C  2C        0270          INC     L
'019D  33        0271          INC     SP
                 0272 ;
'019E  EDAA      0273          IND
'01A0  EDBA      0274          INDR
'01A2  EDA2      0275          INI
'01A4  EDB2      0276          INIR
                 0277 ;
'01A6  E9        0278          JP      (HL)
'01A7  DDE9      0279          JP      (IX)
'01A9  FDE9      0280          JP      (IY)
'01AB  DA0500'   0281          JP      C,NN
'01AE  FA0500'   0282          JP      M,NN
'01B1  D20500'   0283          JP      NC,NN
'01B4  C30500'   0284          JP      NN
'01B7  C20500'   0285          JP      NZ,NN
'01BA  F20500'   0286          JP      P,NN
'01BD  EA0500'   0287          JP      PE,NN
'01C0  E20500'   0288          JP      PO,NN
'01C3  CA0500'   0289          JP      Z,NN
                 0290 ;
'01C6  382E      0291          JR      C,DIS
'01C8  182E      0292          JR      DIS
'01CA  302E      0293          JR      NC,DIS
'01CC  202E      0294          JR      NZ,DIS
'01CE  282E      0295          JR      Z,DIS
```

```
 ADDR  CODE       STMT SOURCE STATEMENT

                  0296 ;
'01D0  02         0297          LD      (BC),A
'01D1  12         0298          LD      (DE),A
'01D2  77         0299          LD      (HL),A
'01D3  70         0300          LD      (HL),B
'01D4  71         0301          LD      (HL),C
'01D5  72         0302          LD      (HL),D
'01D6  73         0303          LD      (HL),E
'01D7  74         0304          LD      (HL),H
'01D8  75         0305          LD      (HL),L
'01D9  3620       0306          LD      (HL),N
                  0307 ;
'01DB  DD7705     0308          LD      (IX+IND),A
'01DE  DD7005     0309          LD      (IX+IND),B
'01E1  DD7105     0310          LD      (IX+IND),C
'01E4  DD7205     0311          LD      (IX+IND),D
'01E7  DD7305     0312          LD      (IX+IND),E
'01EA  DD7405     0313          LD      (IX+IND),H
'01ED  DD7505     0314          LD      (IX+IND),L
'01F0  DD360520   0315          LD      (IX+IND),N
                  0316 ;
'01F4  FD7705     0317          LD      (IY+IND),A
'01F7  FD7005     0318          LD      (IY+IND),B
'01FA  FD7105     0319          LD      (IY+IND),C
'01FD  FD7205     0320          LD      (IY+IND),D
'0200  FD7305     0321          LD      (IY+IND),E
'0203  FD7405     0322          LD      (IY+IND),H
'0206  FD7505     0323          LD      (IY+IND),L
'0209  FD360520   0324          LD      (IY+IND),N
                  0325 ;
'020D  320500'    0326          LD      (NN),A
'0210  ED430500'  0327          LD      (NN),BC
'0214  ED530500'  0328          LD      (NN),DE
'0218  220500'    0329          LD      (NN),HL
'021B  DD220500'  0330          LD      (NN),IX
'021F  FD220500'  0331          LD      (NN),IY
'0223  ED730500'  0332          LD      (NN),SP
                  0333 ;
'0227  0A         0334          LD      A,(BC)
'0228  1A         0335          LD      A,(DE)
'0229  7E         0336          LD      A,(HL)
'022A  DD7E05     0337          LD      A,(IX+IND)
'022D  FD7E05     0338          LD      A,(IY+IND)
'0230  3A0500'    0339          LD      A,(NN)
'0233  7F         0340          LD      A,A
'0234  78         0341          LD      A,B
'0235  79         0342          LD      A,C
'0236  7A         0343          LD      A,D
'0237  7B         0344          LD      A,E
'0238  7C         0345          LD      A,H
'0239  ED57       0346          LD      A,I
'023B  7D         0347          LD      A,L
'023C  3E20       0348          LD      A,N
'023E  ED5F       0349          LD      A,R
                  0350 ;
'0240  46         0351          LD      B,(HL)
'0241  DD4605     0352          LD      B,(IX+IND)
'0244  FD4605     0353          LD      B,(IY+IND)
```

```
 ADDR   CODE       STMT SOURCE STATEMENT

'0247   47         0354          LD      B,A
'0248   40         0355          LD      B,B
'0249   41         0356          LD      B,C
'024A   42         0357          LD      B,D
'024B   43         0358          LD      B,E
'024C   44         0359          LD      B,H
'024D   45         0360          LD      B,L
'024E   0620       0361          LD      B,N
                   0362 ;
'0250   ED4B0500'  0363          LD      BC,(NN)
'0254   010500'    0364          LD      BC,NN
                   0365 ;
'0257   4E         0366          LD      C,(HL)
'0258   DD4E05     0367          LD      C,(IX+IND)
'025B   FD4E05     0368          LD      C,(IY+IND)
'025E   4F         0369          LD      C,A
'025F   48         0370          LD      C,B
'0260   49         0371          LD      C,C
'0261   4A         0372          LD      C,D
'0262   4B         0373          LD      C,E
'0263   4C         0374          LD      C,H
'0264   4D         0375          LD      C,L
'0265   0E20       0376          LD      C,N
                   0377 ;
'0267   56         0378          LD      D,(HL)
'0268   DD5605     0379          LD      D,(IX+IND)
'026B   FD5605     0380          LD      D,(IY+IND)
'026E   57         0381          LD      D,A
'026F   50         0382          LD      D,B
'0270   51         0383          LD      D,C
'0271   52         0384          LD      D,D
'0272   53         0385          LD      D,E
'0273   54         0386          LD      D,H
'0274   55         0387          LD      D,L
'0275   1620       0388          LD      D,N
                   0389 ;
'0277   ED5B0500'  0390          LD      DE,(NN)
'027B   110500'    0391          LD      DE,NN
                   0392 ;
'027E   5E         0393          LD      E,(HL)
'027F   DD5E05     0394          LD      E,(IX+IND)
'0282   FD5E05     0395          LD      E,(IY+IND)
'0285   5F         0396          LD      E,A
'0286   58         0397          LD      E,B
'0287   59         0398          LD      E,C
'0288   5A         0399          LD      E,D
'0289   5B         0400          LD      E,E
'028A   5C         0401          LD      E,H
'028B   5D         0402          LD      E,L
'028C   1E20       0403          LD      E,N
                   0404 ;
'028E   66         0405          LD      H,(HL)
'028F   DD6605     0406          LD      H,(IX+IND)
'0292   FD6605     0407          LD      H,(IY+IND)
'0295   67         0408          LD      H,A
'0296   60         0409          LD      H,B
'0297   61         0410          LD      H,C
'0298   62         0411          LD      H,D
```

```
 ADDR  CODE       STMT SOURCE STATEMENT

'0299  63         0412          LD      H,E
'029A  64         0413          LD      H,H
'029B  65         0414          LD      H,L
'029C  2620       0415          LD      H,N
                  0416 ;
'029E  2A0500'    0417          LD      HL,(NN)
'02A1  210500'    0418          LD      HL,NN
                  0419 ;
'02A4  ED47       0420          LD      I,A
                  0421 ;
'02A6  DD2A0500'  0422          LD      IX,(NN)
'02AA  DD210500'  0423          LD      IX,NN
                  0424 ;
'02AE  FD2A0500'  0425          LD      IY,(NN)
'02B2  FD210500'  0426          LD      IY,NN
                  0427 ;
'02B6  6E         0428          LD      L,(HL)
'02B7  DD6E05     0429          LD      L,(IX+IND)
'02BA  FD6E05     0430          LD      L,(IY+IND)
'02BD  6F         0431          LD      L,A
'02BE  68         0432          LD      L,B
'02BF  69         0433          LD      L,C
'02C0  6A         0434          LD      L,D
'02C1  6B         0435          LD      L,E
'02C2  6C         0436          LD      L,H
'02C3  6D         0437          LD      L,L
'02C4  2E20       0438          LD      L,N
                  0439 ;
'02C6  ED4F       0440          LD      R,A
                  0441 ;
'02C8  ED7B0500'  0442          LD      SP,(NN)
'02CC  F9         0443          LD      SP,HL
'02CD  DDF9       0444          LD      SP,IX
'02CF  FDF9       0445          LD      SP,IY
'02D1  310500'    0446          LD      SP,NN
                  0447 ;
'02D4  EDA8       0448          LDD
'02D6  EDB8       0449          LDDR
'02D8  EDA0       0450          LDI
'02DA  EDB0       0451          LDIR
                  0452 ;
'02DC  ED44       0453          NEG
                  0454 ;
'02DE  00         0455          NOP
                  0456 ;
'02DF  B6         0457          OR      (HL)
'02E0  DDB605     0458          OR      (IX+IND)
'02E3  FDB605     0459          OR      (IY+IND)
'02E6  B7         0460          OR      A
'02E7  B0         0461          OR      B
'02E8  B1         0462          OR      C
'02E9  B2         0463          OR      D
'02EA  B3         0464          OR      E
'02EB  B4         0465          OR      H
'02EC  B5         0466          OR      L
'02ED  F620       0467          OR      N
                  0468 ;
'02EF  EDBB       0469          OTDR
```

```
 ADDR  CODE      STMT SOURCE STATEMENT

'02F1  EDB3      0470         OTIR
                 0471 ;
'02F3  ED79      0472         OUT     (C),A
'02F5  ED41      0473         OUT     (C),B
'02F7  ED49      0474         OUT     (C),C
'02F9  ED51      0475         OUT     (C),D
'02FB  ED59      0476         OUT     (C),E
'02FD  ED61      0477         OUT     (C),H
'02FF  ED69      0478         OUT     (C),L
'0301  D320      0479         OUT     (N),A
                 0480 ;
'0303  EDAB      0481         OUTD
'0305  EDA3      0482         OUTI
                 0483 ;
'0307  F1        0484         POP     AF
'0308  C1        0485         POP     BC
'0309  D1        0486         POP     DE
'030A  E1        0487         POP     HL
'030B  DDE1      0488         POP     IX
'030D  FDE1      0489         POP     IY
'030F  F5        0490         PUSH    AF
'0310  C5        0491         PUSH    BC
'0311  D5        0492         PUSH    DE
'0312  E5        0493         PUSH    HL
'0313  DDE5      0494         PUSH    IX
'0315  FDE5      0495         PUSH    IY
                 0496 ;
'0317  CB86      0497         RES     0,(HL)
'0319  DDCB0586  0498         RES     0,(IX+IND)
'031D  FDCB0586  0499         RES     0,(IY+IND)
'0321  CB87      0500         RES     0,A
'0323  CB80      0501         RES     0,B
'0325  CB81      0502         RES     0,C
'0327  CB82      0503         RES     0,D
'0329  CB83      0504         RES     0,E
'032B  CB84      0505         RES     0,H
'032D  CB85      0506         RES     0,L
                 0507 ;
'032F  CB8E      0508         RES     1,(HL)
'0331  DDCB058E  0509         RES     1,(IX+IND)
'0335  FDCB058E  0510         RES     1,(IY+IND)
'0339  CB8F      0511         RES     1,A
'033B  CB88      0512         RES     1,B
'033D  CB89      0513         RES     1,C
'033F  CB8A      0514         RES     1,D
'0341  CB8B      0515         RES     1,E
'0343  CB8C      0516         RES     1,H
'0345  CB8D      0517         RES     1,L
                 0518 ;
'0347  CB96      0519         RES     2,(HL)
'0349  DDCB0596  0520         RES     2,(IX+IND)
'034D  FDCB0596  0521         RES     2,(IY+IND)
'0351  CB97      0522         RES     2,A
'0353  CB90      0523         RES     2,B
'0355  CB91      0524         RES     2,C
'0357  CB92      0525         RES     2,D
'0359  CB93      0526         RES     2,E
'035B  CB94      0527         RES     2,H
```

```
 ADDR  CODE      STMT SOURCE STATEMENT

'035D  CB95      0528          RES     2,L
                 0529 ;
'035F  CB9E      0530          RES     3,(HL)
'0361  DDCB059E  0531          RES     3,(IX+IND)
'0365  FDCB059E  0532          RES     3,(IY+IND)
'0369  CB9F      0533          RES     3,A
'036B  CB98      0534          RES     3,B
'036D  CB99      0535          RES     3,C
'036F  CB9A      0536          RES     3,D
'0371  CB9B      0537          RES     3,E
'0373  CB9C      0538          RES     3,H
'0375  CB9D      0539          RES     3,L
                 0540 ;
'0377  CBA6      0541          RES     4,(HL)
'0379  DDCB05A6  0542          RES     4,(IX+IND)
'037D  FDCB05A6  0543          RES     4,(IY+IND)
'0381  CBA7      0544          RES     4,A
'0383  CBA0      0545          RES     4,B
'0385  CBA1      0546          RES     4,C
'0387  CBA2      0547          RES     4,D
'0389  CBA3      0548          RES     4,E
'038B  CBA4      0549          RES     4,H
'038D  CBA5      0550          RES     4,L
                 0551 ;
'038F  CBAE      0552          RES     5,(HL)
'0391  DDCB05AE  0553          RES     5,(IX+IND)
'0395  FDCB05AE  0554          RES     5,(IY+IND)
'0399  CBAF      0555          RES     5,A
'039B  CBA8      0556          RES     5,B
'039D  CBA9      0557          RES     5,C
'039F  CBAA      0558          RES     5,D
'03A1  CBAB      0559          RES     5,E
'03A3  CBAC      0560          RES     5,H
'03A5  CBAD      0561          RES     5,L
                 0562 ;
'03A7  CBB6      0563          RES     6,(HL)
'03A9  DDCB05B6  0564          RES     6,(IX+IND)
'03AD  FDCB05B6  0565          RES     6,(IY+IND)
'03B1  CBB7      0566          RES     6,A
'03B3  CBB0      0567          RES     6,B
'03B5  CBB1      0568          RES     6,C
'03B7  CBB2      0569          RES     6,D
'03B9  CBB3      0570          RES     6,E
'03BB  CBB4      0571          RES     6,H
'03BD  CBB5      0572          RES     6,L
                 0573 ;
'03BF  CBBE      0574          RES     7,(HL)
'03C1  DDCB05BE  0575          RES     7,(IX+IND)
'03C5  FDCB05BE  0576          RES     7,(IY+IND)
'03C9  CBBF      0577          RES     7,A
'03CB  CBB8      0578          RES     7,B
'03CD  CBB9      0579          RES     7,C
'03CF  CBBA      0580          RES     7,D
'03D1  CBBB      0581          RES     7,E
'03D3  CBBC      0582          RES     7,H
'03D5  CBBD      0583          RES     7,L
                 0584 ;
'03D7  C9        0585          RET
```

```
 ADDR   CODE       STMT SOURCE STATEMENT

'03D8   D8         0586         RET     C
'03D9   F8         0587         RET     M
'03DA   D0         0588         RET     NC
'03DB   C0         0589         RET     NZ
'03DC   F0         0590         RET     P
'03DD   E8         0591         RET     PE
'03DE   E0         0592         RET     PO
'03DF   C8         0593         RET     Z
                   0594 ;
'03E0   ED4D       0595         RETI
'03E2   ED45       0596         RETN
                   0597 ;
'03E4   CB16       0598         RL      (HL)
'03E6   DDCB0516   0599         RL      (IX+IND)
'03EA   FDCB0516   0600         RL      (IY+IND)
'03EE   CB17       0601         RL      A
'03F0   CB10       0602         RL      B
'03F2   CB11       0603         RL      C
'03F4   CB12       0604         RL      D
'03F6   CB13       0605         RL      E
'03F8   CB14       0606         RL      H
'03FA   CB15       0607         RL      L
                   0608 ;
'03FC   17         0609         RLA
                   0610 ;
'03FD   CB06       0611         RLC     (HL)
'03FF   DDCB0506   0612         RLC     (IX+IND)
'0403   FDCB0506   0613         RLC     (IY+IND)
'0407   CB07       0614         RLC     A
'0409   CB00       0615         RLC     B
'040B   CB01       0616         RLC     C
'040D   CB02       0617         RLC     D
'040F   CB03       0618         RLC     E
'0411   CB04       0619         RLC     H
'0413   CB05       0620         RLC     L
                   0621 ;
'0415   07         0622         RLCA
                   0623 ;
'0416   ED6F       0624         RLD
                   0625 ;
'0418   CB1E       0626         RR      (HL)
'041A   DDCB051E   0627         RR      (IX+IND)
'041E   FDCB051E   0628         RR      (IY+IND)
'0422   CB1F       0629         RR      A
'0424   CB18       0630         RR      B
'0426   CB19       0631         RR      C
'0428   CB1A       0632         RR      D
'042A   CB1B       0633         RR      E
'042C   CB1C       0634         RR      H
'042E   CB1D       0635         RR      L
                   0636 ;
'0430   1F         0637         RRA
                   0638 ;
'0431   CB0E       0639         RRC     (HL)
'0433   DDCB050E   0640         RRC     (IX+IND)
'0437   FDCB050E   0641         RRC     (IY+IND)
'043B   CB0F       0642         RRC     A
'043D   CB08       0643         RRC     B
```

```
 ADDR  CODE      STMT SOURCE STATEMENT

'043F  CB09      0644          RRC     C
'0441  CB0A      0645          RRC     D
'0443  CB0B      0646          RRC     E
'0445  CB0C      0647          RRC     H
'0447  CB0D      0648          RRC     L
                 0649 ;
'0449  0F        0650          RRCA
                 0651 ;
'044A  ED67      0652          RRD
                 0653 ;
'044C  C7        0654          RST     0
'044D  CF        0655          RST     08H
'044E  D7        0656          RST     10H
'044F  DF        0657          RST     18H
'0450  E7        0658          RST     20H
'0451  EF        0659          RST     28H
'0452  F7        0660          RST     30H
'0453  FF        0661          RST     38H
                 0662 ;
'0454  9E        0663          SBC     A,(HL)
'0455  DD9E05    0664          SBC     A,(IX+IND)
'0458  FD9E05    0665          SBC     A,(IY+IND)
'045B  9F        0666          SBC     A,A
'045C  98        0667          SBC     A,B
'045D  99        0668          SBC     A,C
'045E  9A        0669          SBC     A,D
'045F  9B        0670          SBC     A,E
'0460  9C        0671          SBC     A,H
'0461  9D        0672          SBC     A,L
'0462  DE20      0673          SBC     A,N
                 0674 ;
'0464  ED42      0675          SBC     HL,BC
'0466  ED52      0676          SBC     HL,DE
'0468  ED62      0677          SBC     HL,HL
'046A  ED72      0678          SBC     HL,SP
                 0679 ;
'046C  37        0680          SCF
                 0681 ;
'046D  CBC6      0682          SET     0,(HL)
'046F  DDCB05C6  0683          SET     0,(IX+IND)
'0473  FDCB05C6  0684          SET     0,(IY+IND)
'0477  CBC7      0685          SET     0,A
'0479  CBC0      0686          SET     0,B
'047B  CBC1      0687          SET     0,C
'047D  CBC2      0688          SET     0,D
'047F  CBC3      0689          SET     0,E
'0481  CBC4      0690          SET     0,H
'0483  CBC5      0691          SET     0,L
                 0692 ;
'0485  CBCE      0693          SET     1,(HL)
'0487  DDCB05CE  0694          SET     1,(IX+IND)
'048B  FDCB05CE  0695          SET     1,(IY+IND)
'048F  CBCF      0696          SET     1,A
'0491  CBC8      0697          SET     1,B
'0493  CBC9      0698          SET     1,C
'0495  CBCA      0699          SET     1,D
'0497  CBCB      0700          SET     1,E
'0499  CBCC      0701          SET     1,H
```

```
 ADDR   CODE       STMT SOURCE STATEMENT

'049B   CBCD       0702          SET     1,L
                   0703 ;
'049D   CBD6       0704          SET     2,(HL)
'049F   DDCB05D6   0705          SET     2,(IX+IND)
'04A3   FDCB05D6   0706          SET     2,(IY+IND)
'04A7   CBD7       0707          SET     2,A
'04A9   CBD0       0708          SET     2,B
'04AB   CBD1       0709          SET     2,C
'04AD   CBD2       0710          SET     2,D
'04AF   CBD3       0711          SET     2,E
'04B1   CBD4       0712          SET     2,H
'04B3   CBD5       0713          SET     2,L
                   0714 ;
'04B5   CBDE       0715          SET     3,(HL)
'04B7   DDCB05DE   0716          SET     3,(IX+IND)
'04BB   FDCB05DE   0717          SET     3,(IY+IND)
'04BF   CBDF       0718          SET     3,A
'04C1   CBD8       0719          SET     3,B
'04C3   CBD9       0720          SET     3,C
'04C5   CBDA       0721          SET     3,D
'04C7   CBDB       0722          SET     3,E
'04C9   CBDC       0723          SET     3,H
'04CB   CBDD       0724          SET     3,L
                   0725 ;
'04CD   CBE6       0726          SET     4,(HL)
'04CF   DDCB05E6   0727          SET     4,(IX+IND)
'04D3   FDCB05E6   0728          SET     4,(IY+IND)
'04D7   CBE7       0729          SET     4,A
'04D9   CBE0       0730          SET     4,B
'04DB   CBE1       0731          SET     4,C
'04DD   CBE2       0732          SET     4,D
'04DF   CBE3       0733          SET     4,E
'04E1   CBE4       0734          SET     4,H
'04E3   CBE5       0735          SET     4,L
                   0736 ;
'04E5   CBEE       0737          SET     5,(HL)
'04E7   DDCB05EE   0738          SET     5,(IX+IND)
'04EB   FDCB05EE   0739          SET     5,(IY+IND)
'04EF   CBEF       0740          SET     5,A
'04F1   CBE8       0741          SET     5,B
'04F3   CBE9       0742          SET     5,C
'04F5   CBEA       0743          SET     5,D
'04F7   CBEB       0744          SET     5,E
'04F9   CBEC       0745          SET     5,H
'04FB   CBED       0746          SET     5,L
                   0747 ;
'04FD   CBF6       0748          SET     6,(HL)
'04FF   DDCB05F6   0749          SET     6,(IX+IND)
'0503   FDCB05F6   0750          SET     6,(IY+IND)
'0507   CBF7       0751          SET     6,A
'0509   CBF0       0752          SET     6,B
'050B   CBF1       0753          SET     6,C
'050D   CBF2       0754          SET     6,D
'050F   CBF3       0755          SET     6,E
'0511   CBF4       0756          SET     6,H
'0513   CBF5       0757          SET     6,L
                   0758 ;
'0515   CBFE       0759          SET     7,(HL)
```

```
 ADDR  CODE      STMT SOURCE STATEMENT

'0517  DDCB05FE  0760          SET     7,(IX+IND)
'051B  FDCB05FE  0761          SET     7,(IY+IND)
'051F  CBFF      0762          SET     7,A
'0521  CBF8      0763          SET     7,B
'0523  CBF9      0764          SET     7,C
'0525  CBFA      0765          SET     7,D
'0527  CBFB      0766          SET     7,E
'0529  CBFC      0767          SET     7,H
'052B  CBFD      0768          SET     7,L
                 0769 ;
'052D  CB26      0770          SLA     (HL)
'052F  DDCB0526  0771          SLA     (IX+IND)
'0533  FDCB0526  0772          SLA     (IY+IND)
'0537  CB27      0773          SLA     A
'0539  CB20      0774          SLA     B
'053B  CB21      0775          SLA     C
'053D  CB22      0776          SLA     D
'053F  CB23      0777          SLA     E
'0541  CB24      0778          SLA     H
'0543  CB25      0779          SLA     L
                 0780 ;
'0545  CB2E      0781          SRA     (HL)
'0547  DDCB052E  0782          SRA     (IX+IND)
'054B  FDCB052E  0783          SRA     (IY+IND)
'054F  CB2F      0784          SRA     A
'0551  CB28      0785          SRA     B
'0553  CB29      0786          SRA     C
'0555  CB2A      0787          SRA     D
'0557  CB2B      0788          SRA     E
'0559  CB2C      0789          SRA     H
'055B  CB2D      0790          SRA     L
                 0791 ;
'055D  CB3E      0792          SRL     (HL)
'055F  DDCB053E  0793          SRL     (IX+IND)
'0563  FDCB053E  0794          SRL     (IY+IND)
'0567  CB3F      0795          SRL     A
'0569  CB38      0796          SRL     B
'056B  CB39      0797          SRL     C
'056D  CB3A      0798          SRL     D
'056F  CB3B      0799          SRL     E
'0571  CB3C      0800          SRL     H
'0573  CB3D      0801          SRL     L
                 0802 ;
'0575  96        0803          SUB     (HL)
'0576  DD9605    0804          SUB     (IX+IND)
'0579  FD9605    0805          SUB     (IY+IND)
'057C  97        0806          SUB     A
'057D  90        0807          SUB     B
'057E  91        0808          SUB     C
'057F  92        0809          SUB     D
'0580  93        0810          SUB     E
'0581  94        0811          SUB     H
'0582  95        0812          SUB     L
'0583  D620      0813          SUB     N
                 0814 ;
'0585  AE        0815          XOR     (HL)
'0586  DDAE05    0816          XOR     (IX+IND)
'0589  FDAE05    0817          XOR     (IY+IND)
```

```
 ADDR   CODE      STMT SOURCE STATEMENT

'058C   AF        0818          XOR     A
'058D   A8        0819          XOR     B
'058E   A9        0820          XOR     C
'058F   AA        0821          XOR     D
'0590   AB        0822          XOR     E
'0591   AC        0823          XOR     H
'0592   AD        0824          XOR     L
'0593   EE20      0825          XOR     N
                  0826 ;
                  0827          END

ERRORS=0000
```