XZ80

USERS MANUAL

SIERRA DIGITAL SYSTEMS

XZ80

USER MANUAL FOR THE
Z80 X8 SERIES CROSS-ASSEMBLER ON THE PDP8-E.

JULY 1977

SIERRA DIGITAL SYSTEMS
1440 WESTFIELD AVE.
RENO, NEVADA 89509
702-329-9548

ALTHOUGH THE INFORMATION IN THIS MANUAL HAS BEEN CHECKED FOR ACCURACY, NO RESPONSIBILITY IS ASSUMED FOR ERRORS. THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

TABLE OF CONTENTS: SECTION #

****

INTRODUCTION 1.0.0

OPERATION. 2.0.0
LOADING THE CROSS-ASSEMBLER. 2.1.0
CALLING SEQUENCE. 2.2.0
INPUT/OUTPUT FILE EXTENSIONS. 2.3.0
RUN-TIME OPTIONS. 2.4.0

ASSEMBLER CHARACTER SET. 3.0.0

STATEMENT FORMAT. 4.0.0
CODING CONVENTIONS. 4.1.0
LABELS. 4.2.0
OPERATORS. 4.3.0
OPERANDS. 4.4.0
TERMS AND EXPRESSIONS. 4.5.0

NUMERIC CONSTANTS. 5.0.0
CONSTANTS WITH RADIX INDICATORS. 5.1.0
CONSTANTS WITH ASCII INDICATORS. 5.2.0

SYMBOLS. 6.0.0
PERMANENT SYMBOLS. 6.1.0
USER DEFINED SYMBOLS. 6.2.0
LOCAL SYMBOLS. 6.3.0

CURRENT LOCATION COUNTER. 7.0.0

ARITHMETIC OPERATOR SET. 8.0.0
UNARY OPERATORS. 8.1.0
BYTE ACCESS OPERATORS (^L AND ^M). 8.1.2
THE COMPLEMENT OPERATOR (^C). 8.1.3
? OPERATOR. 8.1.4
BINARY OPERATORS. 8.2.0

PSEUDO-OPERATORS. 9.0.0
ASSIGNMENT PSEUDO-OPS. 9.1.0
.EQU. 9.1.1
.SET. 9.1.2
.DINST. 9.1.3
.ORG. 9.1.4
DEFAULT RADIX PSEUDO-OPS. 9.2.0

****

TABLE OF CONTENTS: (CONT.) SECTION #

****

DATA STORAGE PSEUDO-OPS. 9.3.0
. BYTE. 9.3.1
. DBYTE. 9.3.2
. ADDR. 9.3.3
. ZERO. 9.3.4
LISTING CONTROL DIRECTIVES. 9.4.0
. LIST. 9.4.1
. PAGE. 9.4.2
. TITLE. 9.4.3
CONDITIONAL ASSEMBLY PSEUDO-OPS. 9.5.0
. IFZERO. 9.5.1
. IFNZRO. 9.5.2
. IFDEF. 9.5.3
. IFNDEF. 9.5.4
. ENDC. 9.5.5
. END PSEUDO-OP. 9.6.0

ERROR MESSAGES. 10.0.0

MODIFICATION NOTES. 11.0.0

CROSS ASSEMBLER SPECIFICS. 12.0.0
CROSS-ASSEMBLER FILE NAMES. 12.1.0
RESERVED SYMBOLS. 12.2.0
RELATIVE ADDRESS CALCULATIONS. 12.3.0
LISTING FORMAT. 12.4.0
BINARY FILE OUTPUT. 12.5.0
ADDITIONAL ERROR MESSAGE FOR THE XZ80. . . . . 12.6.0
SAMPLE PROGRAM. 12.7.0

MICROPROCESSOR INSTRUCTION SET. 13.0.0

APPENDICES. 14.0.0
RUN-TIME OPTIONS. . . . . APPENDIX A
INDICATOR SET. . . . . APPENDIX B
PSUEDO-OPS. . . . . APPENDIX C
ERROR MESSAGES. . . . . APPENDIX D

****

1.0.0 INTRODUCTION.

THIS MANUAL DESCRIBES ONE OF THE X8 (CROSS EIGHT) SERIES OF MICRO-PROCESSOR CROSS-ASSEMBLERS SIERRA DIGITAL SYSTEMS HAS DEVELOPED FOR PDP8 USERS. THE X8 SERIES WILL HANDLE ALL OF THE POPULAR MICRO-PROCESSORS WITHIN A UNIVERSAL ASSEMBLER FORMAT. THIS COMMON BASE OF ASSEMBLER DIRECTIVES AND TECHNIQUES IS A SELECTED COMBINATION OF DESIRABLE FEATURES OBSERVED IN A SURVEY OF MANY EXISTING MINI-COMPUTER AND MICROPROCESSOR ASSEMBLERS. THE INSTRUCTION MNEMONICS AND ASSOCIATED SYNTAX OF EACH PARTICULAR MICROPROCESSOR ARE RETAINED UNCHANGED.

THIS MANUAL DESCRIBES THE USAGE OF ONE OF THE MICROPROCESSOR CROSS-ASSEMBLERS FROM THE SIERRA DIGITAL X8 SERIES. IN ORDER TO SIMPLIFY THE LEARNING PROCESS FOR INDIVIDUALS USING MORE THAN ONE CROSS-ASSEMBLER FROM THE SERIES, THIS MANUAL HAS BEEN DIVIDED INTO TWO MAJOR PARTS. SECTIONS 1 THROUGH 11 DOCUMENT THE UNIVERSAL ASSEMBLER FORMAT AS IT APPLIES TO ALL CROSS-ASSEMBLERS IN THE SERIES. THESE SECTIONS WILL BE IDENTICAL IN EVERY CROSS-ASSEMBLER MANUAL. SECTION 12 PRESENTS INFORMATION ON APPLICATION OF THE UNIVERSAL ASSEMBLER FORMAT TO THE SPECIFIC MICROPROCESSOR CROSS-ASSEMBLER. SECTION 13 PRESENTS A SUMMARY OF THE MNEMONIC INSTRUCTION CODES ASSIGNED BY THE MICROPROCESSOR VENDOR AND RECONIZED BY THE CROSS-ASSEMBLER. NO ATTEMPT HAS BEEN MADE TO DESCRIBE THE OPERATION OF THE MICROPROCESSOR ITSELF. SUCH INFORMATION MUST BE OBTAINED FROM THE MICROPROCESSOR VENDOR OR OTHER SOURCES. SECTION 14, THE APPEN-DICES, CONTAINS SUMMARY TABLES FOR QUICK REFERENCE ONCE THE USER GAINS EXPERTISE IN USING THE CROSS-ASSEMBLER.

WE AT SIERRA DIGITAL LOOK FORWARD TO DEVELOPING MORE ASSEMBLERS IN OUR X8 SERIES TO PROVIDE YOU, THE USER, WITH THE MEANS OF PIONEERING THE NEW WORLD OF MICROPROCESSORS.

2.0.0 OPERATION.

SIERRA DIGITAL'S CROSS-ASSEMBLER IS AN 8K, TWO PASS ASSEMBLER WHICH RUNS UNDER THE OS/8 OPERATING SYSTEM. THE CROSS-ASSEMBLER IS CODED IN PDP/8 ASSEMBLY LANGUAGE (PAL8) TO GIVE FAST EXECUTION TIMES. (LESS THAN 30 SECONDS FOR A NORMAL 4K BYTE PROGRAM IS TYPICAL).

PASS 1 READS THE INPUT FILES AND SETS UP THE SYMBOL TABLES. PASS 2 THEN GENERATES THE OUTPUT FILE IN THE BINARY (OBJECT) FORMAT OF THE PARTICULAR MICROPROCESSOR. THE OUTPUT FILE CAN BE CHANGED TO BNPF FORMAT THROUGH USE OF THE /B RUN-TIME OPTION.

A THIRD ASSEMBLY PASS IS DONE WHEN A LISTING OUTPUT FILE IS SPECI-FIED. WHEN NO BINARY FILE IS SPECIFIED, THE ASSEMBLER GOES DIRECTLY TO THE PASS 3 LISTING.

THE CROSS-ASSEMBLER IS NOT RESTARTABLE. IF AN ATTEMPT IS MADE TO RESTART THE ASSEMBLER WITH A .ST COMMAND, THE KEYBOARD MONITOR RETURNS A "NO!!" .

TYPING CTRL/C WILL HALT ASSEMBLY AND CAUSE AN IMMEDIATE EXIT TO THE KEYBOARD MONITOR.

TYPING CTRL/O AT THE KEYBOARD DURING ASSEMBLY WILL SUPPRESS THE LISTING OF ERROR MESSAGES TO THE CONSOLE DURING PASSES 1 AND 2. THE OUTPUT FILE WILL STILL SHOW THE ERROR MESSAGES IMMEDIATELY BEFORE THE LINE THAT IS IN ERROR.

2.1.0 LOADING AND SAVING THE CROSS-ASSEMBLER.

THE CROSS-ASSEMLER IS PROVIDED IN BINARY FORMAT ON PAPER TAPE OR IN BOTH BINARY AND IMAGE FORMATS ON FILE-STRUCTURED MEDIA.

TO LOAD THE ASSEMBLER FROM PAPER TAPE AND SAVE IT, PLACE THE TAPE IN THE READER AND CALL THE ABSOLUTE LOADER:

```
.R ABSLDR
*PTR:$

.SAVE SYS:XNAME
```

FROM FILE STRUCTURED MEDIA, THE IMAGE FORMAT PROGRAM MAY BE COPIED DIRECTLY TO THE SYSTEM DEVICE OR THE BINARY FORMAT FILE MAY BE LOADED WITH THE ABSOLUTE LOADER. MODIFICATIONS TO THE IMAGE FILE, SUCH AS INVERTING THE SENSE OF A RUN-TIME OPTION, MAY BE IMPLEMENTED ACCORDING TO THE NOTES IN SECTION # 11.0.0 .

2.2.0 CALLING SEQUENCE.

ONCE LOADED AND SAVED, THE CROSS-ASSEMBLER IS CALLED FROM THE SYSTEM DEVICE BY TYPING:

```
.R XNAME
```

THE ASSEMBLER CALLS THE COMMAND DECODER WHICH RESPONDS WITH AN ASTERISK IN THE LEFT HAND MARGIN. THE USER MAY THEN TYPE IN THE INPUT AND OUTPUT FILE SPECIFICATIONS AND RUN-TIME OPTIONS:

```
*DEV:BIN,DEV:LIST<DEV:IN1,...DEV:IN9/OPT
```

THE FIRST OUTPUT FILE IS THE MICROPROCESSOR BINARY OBJECT FILE WRITTEN IN THE FORMAT SPECIFIED BY THE VENDOR OF THE PARTICULAR MICROPROCESSOR. (SEE SECTION 12.0.0 FOR THE FORMAT SPECIFICATIONS).

THE SECOND OUTPUT FILE IS THE OPTIONAL LISTING. WHEN ONLY THE FIRST OUTPUT FILE IS SPECIFIED, THE ASSEMBLER ASSUMES THAT IT WILL BE THE BINARY OUTPUT FILE AND THE LISTING IS OMITTED.

THE FOLLOWING EXAMPLE SPECIFIES FILE "IN1" TO BE READ FROM DECTAPE 0 AND THE BINARY (OBJECT) FILE TO BE OUTPUT TO THE PAPER TAPE PUNCH WITH NO LISTING:

```
.R XNAME
*PTP:<DTA0:IN1
```

THIS EXAMPLE SPECIFIES 2 FILES AS THE SOURCE INPUT (FROM THE DSK: DEVICE) WITH ONLY THE PASS 3 LISTING BEING OUTPUT TO THE LINE PRINTER:

```
.R XNAME
*,LPT:<IN1,IN2
```

UP TO NINE INPUT FILES CAN BE SPECIFIED AS ONE PROGRAM WHERE THE LAST FILE IS TERMINATED WITH AN .END STATEMENT.

2.3.0 INPUT/OUTPUT FILE EXTENSIONS.

IF THE EXTENSION TO AN INPUT FILE NAME IS OMITTED, THE ASSEMBLER ASSUMES THE .MS EXTENSION. IF THERE IS NO FILE WITH THAT NAME AND AN .MS EXTENSION, THE ASSEMBLER ASSUMES THE NULL EXTENSION. UNLESS EXTENSIONS ARE SPECIFIED, THE .MB AND .LS EXTENSIONS ARE ADDED TO THE OUTPUT BINARY AND LISTING FILES.

```
.MB     - MICROPROCESSOR BINARY OUTPUT FILE EXTENSION.
.LS     - OUTPUT LISTING FILE EXTENSION.
.MS     - MICROPROCESSOR SOURCE FILE EXTENSION.
```

2.4.0 RUN-TIME OPTIONS.

TABLE #1 DESCRIBES THE OPTIONS WHICH MAY BE SPECIFED AT RUN-TIME IN THE INPUT LINE TO THE COMMAND DECODER.

IF ONE OR MORE OF THESE OPTIONS IS CONTINUALLY CALLED, THE USER SHOULD CONSIDER MODIFYING THE ASSEMBLER TO INVERT THE SENSE OF THE OPTION. THE MODIFICATION NOTES IN SECTION #11.0.0 EXPLAIN HOW THIS MAY BE DONE. FOR EXAMPLE, A USER WHO PREFERS TO OUTPUT FILES IN BNPF FORMAT RATHER THAN BINARY CAN INVERT THE SENSE OF THE /B OPTION. THEN THE BINARY FILES ARE NORMALLY WRITTEN IN BNPF FORMAT. USE OF THE /B OPTION THEN CAUSES THE OUTPUT FILE TO BE WRITTEN IN THE STANDARD MICROPROCESSOR BINARY CODE. SPACE IS PROVIDED IN TABLE #1 TO CHECK OFF WHICH OPTIONS HAVE BEEN INVERTED FOR YOUR REFERENCE.

TABLE #1. RUN-TIME OPTIONS. #2. 4. 0

OPTION	MEANING	INVERT?
/B	THE BINARY OUTPUT FILE IS WRITTEN IN BNPF FORMAT. INSTEAD OF IN THE MICROPROCESSOR VENDOR'S STANDARD BINARY FORMAT. FOR THE BNPF FORMAT, THE BINARY OUTPUT IS CONVERTED TO ASCII TEXT WHERE "B" INDICATES THE BEGINNNING OF A BYTE, "F" INDICATES THE END OF A BYTE, "P" INDICATES A 1 BIT AND "N" INDICATES A 0 BIT. FOUR BYTES , SEPARATED BY SPACES, ARE WRITTEN PER LINE. THE ADDRESS OF THE FIRST BYTE IS GIVEN IN SIX DIGIT OCTAL AT THE BEINNING OF THE LINE. LEADING ZEROES IN THE ADDRESS ARE CONVERTED TO SPACES. EACH LINE IS PRECEDED BY 2 SPACES. LEADER CONSISTS OF 100 NULL CHARACTERS WITH 20 RUBOUTS IMMEDIATELY PRECEEDING AND FOLLOWING THE ASCII TEXT. EXAMPLE: THE FOLLOWING CODE IS SHOWN REWRITTEN IN BNPF FORMAT. . ORG 100 . BYTE 27, C7, AF, D7, FF, 72, 0, D0 100 BNNPNNPPPF BPPNNNPPPF BPNPNPPPPF BPPNPNPPPF 104 BPPPPPPPPF BNPPPNNPNF BNNNNNNNNF BPPNPNNNNF	-------
/E	INHIBIT ERROR MESSAGES TO THE CONSOLE. NORMALLY ERROR MESSAGES ARE OUTPUT TO THE CONSOLE DURING ASSEMBLY PASSES 1 AND 2. SINCE ERROR MESSAGES ARE INCLUDED IN THE LISTING, USERS WITH SLOW CONSOLE DEVICES SUCH AS TTY'S CAN SPEED ASSEMBLY TIME WITH THIS OPTION. ALSO, IF THE BINARY FILE IS TO BE OUTPUT TO THE CONSOLE DEVICE, THE ERROR MESSAGES AND BINARY OUTPUT LINES WILL BE INTERMIXED. THE /E OPTION WILL INHIBIT ALL BUT FATAL ERROR MESSAGES SO THAT ONLY THE BINARY FILE IS OUTPUT.	-------

TABLE #1. RUN-TIME OPTIONS. (CONT.) #2.4.0

OPTION	MEANING	INVERT?
/H	INHIBIT HEADINGS AND PAGINATION. NORMALLY, THE ASSEMBLER AUTOMATICALLY PAGES THE OUTPUT, ADDING A HEADER TO THE TOP OF THE PAGE. USE OF THE /H OPTION WILL ELIMINATE THE HEADING AND THE PAGINATION.	-------
/J	LIST UNASSEMBLED STATEMENTS AND CONDITIONAL ASSEMBLY PSEUDO-OPS. STATEMENTS WHICH DO NOT GET ASSEMBLED DUE TO CONDITIONAL ASSEMBLY PSEUDO-OPS ARE NORMALLY NOT LISTED. NEITHER ARE THE CONDITIONAL PSEUDO-OPS THEMSELVES. USE OF THE /J OPTION WILL ADD THESE STATEMENTS TO THE LISTING.	-------
/K	EXPAND SYMBOL TABLE STORAGE INTO EXTRA CORE. NORMALLY MOST OF FIELD 1 IS USED FOR BOTH LOCAL AND NORMAL USER SYMBOL STORAGE. USE OF THE /K OPTIONS EXPANDS CORE USAGE TO 12K WHERE THE LOCAL SYMBOL TABLE RESIDES IN FIELD 2 AND THE REGULAR SYMBOL TABLE RESIDES IN FIELD 1.	-------
/L	OUTPUT LEADER IN BINARY FILE FOR .ORG STATEMENTS THIS OPTION MAY BE USED TO PHYSICALLY SEPARATE DISCONTINUOUS SECTIONS OF THE BINARY OUTPUT ON A PAPER TAPE.	-------
/O	OUTPUT LISTING WITH BINARY CODE IN OCTAL FORMAT. THE GENERATED BINARY CODE IS NORMALLY PRINTED IN HEXADECIMAL AT THE LEFT OF THE PROGRAM STATEMENTS IN THE LISTING FILE. THE /O OPTION WILL CAUSE THE BINARY CODE TO BE LISTED IN OCTAL INSTEAD OF HEXADECIMAL.	-------
/N	LIST ONLY THE SYMBOL TABLE. THE THIRD PASS LISTING NORMALLY CONSISTS OF THE STATEMENT LISTING PLUS THE USER SYMBOL TABLE LISTING. THE /N OPTION CAUSES ONLY THE SYMBOL TABLE TO BE LISTED.	-------
/P	INCLUDE NORMALLY UNLISTED PSEUDO-OPS IN THE LISTING SOME PSEUDO-OPS WILL NOT BE LISTED BY PASS 3 UNLESS THE /P OPTION IS USED.	-------
/S	OMIT THE SYMBOL TABLE FROM LISTING. ONLY THE PROGRAM STATEMENTS ARE LISTED WITH THIS OPTION.	-------

TABLE #1. RUN-TIME OPTIONS. (CONT.) #2.4.0

OPTION	MEANING	INVERT?
/T	REPLACE FORM/FEED WITH 3 CR/LF'S. WHEN LISTING TO A DEVICE SUCH AS A TTY WHICH DOES NOT HAVE A FORM/FEED CONTROL, USE OF THE /T OPTION WILL REPLACE THE FORM/FEED WITH 3 BLANK LINES .	-------
/W	INHIBIT WARNING MESSAGES. WHEN WARNING MESSAGES CAN BE SAFELY IGNORED, THIS OPTION WILL PREVENT THEM FROM BEING OUTPUT.	-------
/0 TO /9	USER FLAGS, USED WITH THE ? OPERATOR, SEE SECTION # 8.1.4 .	

3.0.0 ASSEMBLER CHARACTER SET.

THE FOLLOWING CHARACTERS ARE LEGAL SOURCE CODE CHARACTERS:

1) ALPHABETICS A-Z, UPPER CASE ASCII
2) NUMERICS 0-9
3) THE SPECIAL CHARACTERS LISTED BELOW.

*	MULTIPLICATION
/	DIVISION
&	BOOLEAN AND
!	INCLUSIVE OR
+	ADDITION
-	SUBTRACTION
[ ]	PRECEDENCE INDICATORS
^	UNIVERSAL UNARY OPERATOR (UPARROW). USED WITH: ^C - COMPLEMENT (UPARROW C) ^B - BINARY RADIX INDICATOR (UPARROW B) ^D - DECIMAL RADIX INDICATOR (UPARROW D) ^H - HEXADECIMAL RADIX INDICATOR (UPARROW H) ^O - OCTAL RADIX INDICATOR (UPARROW O) ^L - LEAST SIGNIFICANT BYTE ACCESS OPERATOR ^M - MOST SIGNIFICANT BYTE ACCESS OPERATOR
;	COMMENT INDICATOR
" OR '	ASCII INDICATOR
?	USER FLAG OPERATOR
.	CURRENT LOCATION COUNTER (PERIOD)

THE CARRIAGE RETURN CHARACTER IS RECOGNIZED AS THE TERMINATOR FOR EACH SOURCE LINE. THE LINE-FEED, RUBOUT, FORM-FEED, AND NULL CHARACTERS ARE IGNORED BY THE ASSEMBLER. FORM-FEED CHARACTERS OCCURING IN THE SOURCE HAVE NO AFFECT ON THE LISTING. ALL ASCII CHARACTERS MAY BE USED IN THE COMMENT FIELD OF A STATEMENT.

4.0.0 STATEMENT FORMAT.

STATEMENTS ARE WRITTEN IN THE GENERAL FORM:

```
        LABEL   OPERATOR        OPERAND         ;COMMENT
```

LABELS MUST START IN COLUMN 1. THEY MAY BE DIRECTLY FOLLOWED WITH AN OPTIONAL COMMA IF DESIRED. THE MODIFICATION NOTES EXPLAIN HOW TO REPLACE THE COMMA WITH ANOTHER DELIMITER SUCH AS A COLON.

OPERATORS MUST BE SEPARATED FROM THE LABEL WITH AT LEAST ONE SPACE OR TAB. WHEN NO LABEL IS PRESENT, THE OPERATOR MAY BEGIN IN ANY COLUMN BEYOND COLUMN 1.

THE OPERAND (IF ANY) MUST BE SEPARATED FROM THE OPERATOR WITH AT LEAST ONE SPACE OR TAB.

THE COMMENT (IF ANY) MUST BE SEPARATED FROM THE OPERAND (OR OPERATOR IF THERE IS NO OPERAND BY A SEMICOLON (;).

AN INPUT LINE MAY BE UP TO 127 CHARACTERS LONG (NOT INCLUDING THE CARRIAGE RETURN). WHEN THE INPUT LINES ARE OUTPUT TO THE LISTING FILE, ANY CHARACTERS AFTER THE 72D COLUMN ARE WRITTEN ON THE NEXT LINE(S) BEGINNING AT THE 25TH COLUMN OF THE FIRST SOURCE LINE (NORMAL COMMENT COLUMN). SEE THE MODIFICATION NOTES IN SECTION #11.0.0 TO ADJUST FOR NARROWER OR WIDER PAGE OUTPUT. THE CARRIAGE RETURN IS A TERMINATOR FOR BOTH THE STATEMENT AND THE LINE. ONLY ONE STATEMENT IS ALLOWED PER 127 CHARACTER LINE.

4.1.0 CODING CONVENTIONS:

ALTHOUGH THE ASSEMBLER WILL ACCEPT PROGRAMS WRITTEN IN FREE FORMAT, THE USE OF TABS MAKES FOR MORE READABLE CODE. TAB STOPS ARE SET EVERY 8 CHARACTERS IN THE LINE SO THAT THE USE OF THE TAB KEY SIMPLIFIES INPUT. GENERALLY:

LABELS	OCCUPY THE FIRST TAB FIELD,	COLUMNS 1 THROUGH 8
OPERATORS	OCCUPY THE SECOND TAB FIELD,	COLUMNS 9 THROUGH 16.
OPERANDS	OCCUPY THE THIRD TAB FIELD,	COLUMNS 17 THROUGH 24.
COMMENTS	OCCUPY THE REMAINING FIELDS,	COLUMNS 25 THROUGH 127.

4.2.0 LABELS.

A LABEL IS A SYMBOL WHICH PRECEDES THE OPERATOR AND MUST FOLLOW THE SYMBOL NAMING CONVENTIONS DESCRIBED IN SECTION # 6.2.0 . IN ALL BUT THE SYMBOL DEFINITION PSEUDO-OPS, (.EQU, .SET, .DINST) THE LABEL IS A LOCATION TAG AND IS EQUAL TO THE VALUE OF THE CURRENT LOCATION COUNTER.

EXAMPLE:

```
      2  1                .ORG     201
      0  6     LABEL1      .EQU      6          ;LABEL1=6
 201  1        LABEL2      .BYTE     1          ;LABEL2=LOCATION TAG=201
```

NOTE THAT A JUMP TO LABEL1 WILL TRANSFER TO ADDRESS 6 WHILE A JUMP TO LABEL2 GOES TO ADDRESS 201.

A LABEL LACKING BOTH AN OPERATOR AND OPERAND IS SET EQUAL TO THE VALUE OF THE NEXT ADDRESS TO BE ASSEMBLED. IF USED AT THE BEGINNING OF THE PROGRAM, IT IS SET EQUAL TO THE VALUE OF THE FIRST ADDRESS. WHEN A SOLITARY LABEL IS FOLLOWED BY AN .ORG STATEMENT, IT RETAINS THE ORIGINAL VALUE ASSIGNED BEFORE THE ORIGIN CHANGE.

4.3.0 OPERATORS.

AN OPERATOR IS A MNEMONIC WHICH INDICATES THE ACTION TO BE PERFORMED AND IS EITHER A PSEUDO-OP OR ONE OF THE MICROPROCESSOR INSTRUCTIONS. PSEUDO-OPS ARE DESCRIBED IN SECTION #9.0.0. THE MICROPROCESSOR INSTRUCTION SET IS DESCRIBED IN SECTION #13.0.0 . THESE OPERATORS SHOULD NOT BE CONFUSED WITH ARITHMETIC OPERATORS USED IN OPERAND EXPRESSIONS.

4.4.0 OPERANDS.

AN OPERAND REPRESENTS THE PART OF THE INSTRUCTION WHICH IS TO BE ACTED ON. IT CAN BE A TERM OR AN EXPRESSION.

THE .BYTE, .DBYTE, AND .ADDR PSEUDO-OPS CAN HAVE MULTIPLE OPERANDS.

REFER TO THE EXPLANATION OF EACH OPERATOR FOR THE PROPER OPERAND FORMAT.

IT SHOULD BE NOTED THAT OPERAND EXPRESSIONS ARE EVALUATED TO A SINGLE NUMERICAL VALUE BY THE ASSEMBLER. BINARY CODE IS NOT GENERATED TO MAKE THE MICROPROCESSOR EVALUATE THE EXPRESSION.

4.5.0 TERMS AND EXPRESSIONS.

A TERM IS A SINGLE VALUE, A CONSTANT OR SYMBOL. THE CURRENT LOCATION COUNTER (REPRESENTED BY A PERIOD) IS CONSIDERED A TERM.

TERMS ARE COMBINED WITH OPERAND ARITHMETIC OPERATORS TO FORM EXPRESSIONS.

EXAMPLE: IN THE INSTRUCTION BELOW THE OPERAND IS AN EXPRESSION WHICH HAS TWO ARITHMETIC OPERATORS AND THREE TERMS.

```
        SYMBOL .EQU     1+NEW * 15
```

16 BIT INTEGER ARITHMETIC IS USED TO EVALUATE EXPRESSIONS.

5.0.0 NUMERIC CONSTANTS.

A CONSTANT IS A NUMERIC VALUE REPRESENTED BY A STRING OF DIGITS. THE DEFAULT RADIX OR TEMPORARY RADIX INDICATORS IDENTIFY THE RADIX OF THE CONSTANT. A CONSTANT WITHOUT ANY TEMPORARY RADIX INDICATOR IS CONSIDERED TO BE IN THE DEFAULT RADIX, WHICH IS INITIALLY HEXADECIMAL.

EXAMPLE: THE HEXADECIMAL NUMBER 16 (22 IN BASE 10) IS STORED IN "VALUE" :

```
    0  16           VALUE   .EQU    16
```

THE MAXIMUM VALUE FOR A CONSTANT IS 65535 (BASE 10 UNSIGNED).

THE MINIMUM VALUE FOR A CONSTANT IS -32768 (BASE 10 SIGNED).

5.1.0 CONSTANTS WITH RADIX INDICATORS.

CONSTANTS IN A BASE DIFFERENT FROM THAT OF THE DEFAULT RADIX CAN BE SPECIFIED THROUGH USE OF THE TEMPORARY RADIX INDICATORS. THESE INDICATORS ARE VERY USEFUL FOR ENTERING INDIVIDUAL CONSTANTS. HOWEVER, IF A LARGE GROUP OF VALUES IN ANOTHER RADIX MUST BE ENTERED, IT IS MORE CONVENIENT TO CHANGE THE DEFAULT RADIX USING THE PSUEDO-OPS DESCRIBED IN SECTION # 9.2.0 .

THE TEMPORARY RADIX INDICATORS ARE:

^B	BINARY
^D	DECIMAL
^H	HEXADECIMAL
^O	OCTAL

THE ^ IS THE UPARROW CHARACTER (UNIVERSAL UNARY OPERATOR).

A HEXADECIMAL CONSTANT WHICH DOES NOT BEGIN WITH A NUMBER SHOULD BE WRITTEN WITH A LEADING ZERO TO DISTINGUISH IT FROM FROM A SYMBOL. A RADIX INDICATOR PRECEDING A SYMBOL IS IGNORED.

EXAMPLE: THE FIRST STATEMENT IS VALID, THE SECOND IS NOT.

```
VALUE   .EQU    ^H0A302         ;VALUE=A302, BASE 16
VALUE   .EQU    ^HA302          ;VALUE = SYMBOL A302
```

SINCE THE SYMBOL A302 MAY NOT EXIST, THE SECOND STATEMENT WILL PROBABLY CAUSE AN UNDEFINED SYMBOL ERROR. TEMPORARY RADIX INDICATORS AFFECT THE NEXT DIGIT STRING IN THE EXPRESSION UNLESS A SYMBOL NAME OR BINARY OPERATOR OCCURS FIRST. IN THAT CASE, THE TEMPORARY RADIX INDICATOR WOULD BE IGNORED. NO ERROR MESSAGE IS GIVEN.

5.2.0 CONSTANTS WITH ASCII INDICATORS.

THE " AND ' INDICATORS ARE USED TO FORM THE 7 BIT ASCII VALUE OF A CHARACTER. THERE ARE FOUR ACCEPTABLE WAYS TO WRITE THE INDICATORS:

"A" OR "A OR 'A' OR 'A ALL EQUAL 41 (BASE 16).

NOTE THAT THE CLOSING QUOTE IS OPTIONAL, BUT IF USED IT MUST MATCH THE OPENING QUOTE. ONLY ONE CHARACTER CAN FOLLOW THE INDICATOR.

THE " IS SPECIALLY HANDLED IN THE .BYTE PSEUDO-OP WHERE IT IS USED TO INPUT TEXT STRINGS. SEE SECTION # 9.3.1 .

6.0.0 SYMBOLS.

THE WORD "SYMBOL" IS USED HERE AS A GENERAL TERM FOR ANY MNEMONIC WHICH IS TO HAVE A VALUE. THIS IS IN CONTRAST TO AN OPERATOR, WHICH IS A MNEMONIC WHICH SPECIFIES A PROCESS.

A LABEL IS A SYMBOL THAT PRECEDES AN OPERATOR IN THE STATEMENT. IF THE LABEL IS USED TO STORE THE VALUE OF THE CURRENT LOCATION COUNTER , IT IS CALLED A LOCATION TAG.

6.1.0 PERMANENT SYMBOLS.

PERMANENT SYMBOLS ARE THE CROSS-ASSEMBLER PSEUDO-OPS AND MICROPROCESSOR OPERATORS. IF NECESSARY, THE .DINST STATEMENT CAN BE USED TO RENAME A MICROPROCESSOR OPERATOR. THE CROSS-ASSEMBLER PSEUDO-OPS CANNOT BE USED IN A .DINST INSTRUCTION. THE TABLES IN THE APPENDICES SUMMARIZE THE PERMANENT SYMBOL SET.

6.2.0 USER DEFINED SYMBOLS.

THESE SYMBOLS CAN BE LOCATION TAGS OR REPRESENT A VALUE.

A SYMBOL IS A STRING OF FROM ONE TO SIX ALPHANUMERIC CHARACTERS DELIMITED BY A NON-ALPHANUMERIC CHARACTER. USER-DEFINED SYMBOLS MUST CONFORM TO THE FOLLOWING RULES:

1) THE CHARACTERS MUST BE LEGAL ALPHA-NUMERICS. (A-Z OR 0-9)
2) THE FIRST CHARACTER MUST BE ALPHABETIC (A-Z).
3) ONLY THE FIRST SIX CHARACTERS ARE USED, ANY OTHERS ARE IGNORED. SYMBOLS ARE STORED IN THE SYMBOL TABLE AND REFERENCED ONLY BY THE FIRST SIX CHARACTERS.
4) A USER-DEFINED SYMBOL CANNOT HAVE THE SAME NAME AS ANY OF THE PERMANENT SYMBOL NAMES. AS THE PERIOD IS CONSIDERED AS PART OF THE ASSEMBLER PSEUDO-OP NAME, A USER-DEFINED SYMBOL WHICH IS IDENTICAL EXCEPT FOR THE LEADING PERIOD IS LEGAL.

6.3.0 LOCAL SYMBOLS.

OFTEN, WHEN PROGRAMMING SHORT SECTIONS OF CODE WHICH INVOLVE NUMEROUS JUMP OR BRANCHING INSTRUCTIONS, THE USER FINDS IT DIFFICULT TO CREATE MEANINGFUL LABELS THAT WILL NOT CONFLICT WITH OTHER SYMBOLS IN THE PROGRAM. IN CASES LIKE THIS, LOCAL SYMBOLS CAN BE USED INSTEAD OF REGULAR SYMBOLS.

LOCAL SYMBOLS HAVE THE FORMAT "$N" WHERE "N" IS A DECIMAL INTEGER FROM 0-255 INCLUSIVE.

LOCAL SYMBOLS MUST BE DEFINED AND REFERENCED WITHIN LOCAL SYMBOL BLOCKS. LOCAL SYMBOL BLOCKS ARE SECTIONS OF THE PROGRAM THAT START ON A STATEMENT HAVING A REGULAR SYMBOL USED AS A LOCATION TAG AND END ON THE STATEMENT JUST BEFORE THE OCCURANCE OF THE NEXT REGULAR SYMBOL LOCATION TAG. NOTE THAT LABELS FOR THE .EQU, .DINST AND .SET PSEUDO-OPS ARE NOT LOCATION TAGS AND DO NOT DELIMIT LOCAL SYMBOL BLOCKS.

THERE IS NO EFFECTIVE LIMIT TO THE SIZE OF A LOCAL SYMBOL BLOCK.

THE SAME LOCAL SYMBOL CAN BE DEFINED AND USED IN AN UNLIMITED NUMBER OF LOCAL SYMBOL BLOCKS.

EXAMPLE:

```
TAG1    .BYTE   "TEXT"  ;SYMBOL BLOCK BEGINS
$1      .EQU    VALUE   ;DEFINE LOCAL $1
$2      .EQU    -1      ;DEFINE LOCAL $2
VALU1   .EQU    $1-$2   ;CALCULATE NEW VALUE
TAG2    .BYTE   "TEXT"  ;NEW SYMBOL BLOCK
$1      .EQU    VALU1   ;DEFINE LOCAL $1
$2      .EQU    -2      ;DEFINE LOCAL $2
VALU2   .EQU    $1*$2   ;CALCULATE NEW VALUE.
TAG3    .BYTE   "TEXT"  ;ENDS SECOND BLOCK
```

7.0.0 CURRENT LOCATION COUNTER.

THE CURRENT LOCATION COUNTER IS INDICATED BY A PERIOD. IT REPRESENTS THE ADDRESS OF THE NEXT BYTE TO BE ASSEMBLED.

THE CURRENT LOCATION COUNTER CANNOT BE USED IN THE LABEL FIELD.

AT THE BEGINNING OF THE SOURCE INPUT THE CURRENT LOCATION COUNTER IS SET TO ZERO. IT CAN BE REASSIGNED THROUGH USE OF THE .ORG PSEUDO-OP.

EXAMPLE:

```
      0  60                    .ORG        60     ;INITIAL ADDRESS
      0   0        VALUE       .EQU         0     ;NO EFFECT ON .
     60  22        TAG         .BYTE       22     ;  .  = 60 (BASE 8)
      1  00                    .ORG       100     ; REASSIGN COUNTER
    100  10        TAG1        .BYTE       10     ; .= 100
```

LOCATION TAGS ARE ALWAYS SET EQUAL TO THE VALUE OF THE CURRENT LOCATION COUNTER WHEN THEY ARE ASSEMBLED. IN THE EXAMPLE ABOVE, THE LOCATION TAG "TAG" = 60.

THE CURRENT LOCATION COUNTER IS AUTOMATICALLY UPDATED IN THE ASSEMBLER AS SOON AS THE CURRENT INSTRUCTION IS ASSEMBLED. NOTE THAT IN THE MULTI-OPERAND DATA STORAGE PSEUDO-OPS, (.BYTE, .DBYTE, AND .ADDR) THE LOCATION COUNTER IS CHANGING AS THE OPERANDS ARE ASSEMBLED.

EXAMPLE: THE LOCATION COUNTER IS USED AS AN OPERAND 3 TIMES IN AN .ADDR PSEUDO-OP.

```
      0 20              .ORG      20
   20 20  0             .ADDR     .,.,.
   22 22  0
   24 24  0
   20 20  0
```

THE CURRENT LOCATION COUNTER USES THE FULL ADDRESS RANGE OF THE MICROPROCESSOR.

8.0.0 THE ARITHMETIC OPERATOR SET.

THERE ARE TWO TYPES OF ARITHMETIC OPERATORS: UNARY AND BINARY OPERATORS.

UNARY OPERATORS ACT ON ONLY ONE ITEM, THE TERM OR EXPRESSION FOLLOWING THEM.

BINARY OPERATORS ACT ON TWO ITEMS: THE TERM OR EXPRESSION PRECEEDING THEM AND THE TERM OR EXPRESION FOLLOWING THEM.

8.1.0 UNARY OPERATORS.

THE + (PLUS) AND - (MINUS) UNARY OPERATORS ASSIGN A POSITIVE OR NEGATIVE SIGN TO THE EXPRESSION FOLLOWING THEM. AN EXPRESSION IS ASSUMED TO BE POSITIVE IF NOT OTHERWISE SPECIFIED.

8.1.2 BYTE ACCESS OPERATORS.

THE ^L AND ^M (WHERE ^ IS THE UPARROW CHARACTER) ARE UNARY OPERATORS WHICH PROVIDE ACCESS TO THE LEAST AND MOST SIGNIFICANT 8 BIT BYTES OF THE VALUE OF AN EXPRESSION OR TERM.

EXAMPLE: TO SET "VALUE" EQUAL TO THE MOST SIGNIFICANT BYTE OF 3B61 (BASE 16), THE STATEMENT BELOW IS USED.

```
VALUE   .SET    ^M3B61          ;VALUE = 003B
```

THIS NEXT STATEMENT TAKES THE LEAST SIGNIFICANT BYTE.

```
VALUE   .SET    ^L3B61          ;VALUE = 0061
```

BYTE ACCESS OPERATORS MAY BE COMBINED WITH THE OTHER UNARY OPERATORS AND THE RADIX INDICATORS.

8.1.3 THE COMPLEMENT OPERATOR.

THE ^C (UPARROW C) IS A LOGICAL UNARY OPERATOR WHICH COMPLEMENTS THE EXPRESSION FOLLOWING IT.

EXAMPLE:

```
VALUE   .EQU    ^C7241  ;VALUE = 8DBE
```

THE COMPLEMENT OPERATOR CAN BE COMBINED WITH THE OTHER UNARY OPERATORS AND THE RADIX INDICATORS.

8.1.4. ? OPERATOR.

THIS IS THE USER FLAG OPERATOR, A UNARY OPERATOR USED IN CONJUNCTION WITH THE COMMAND DECODER USER FLAG OPTIONS (/0 TO /9). IT HAS THE FORM ?EXPRESSION AND MAY BE USED IN OPERANDS LIKE ANY OTHER TERM. THE RESULTING VALUE OF THE QUESTION MARK OPERATOR EQUALS 1 IF THE VALUE OF ITS EXPRESSION MATCHES A USER FLAG THAT WAS SPECIFIED TO THE COMMAND DECODER AT RUN-TIME. OTHERWISE IT EQUALS 0. THIS OPERATOR IS USEFUL FOR CONTROLLING CONDITIONAL ASSEMBLY AND LISTING PARAMETERS WITHOUT HAVING TO MODIFY THE SOURCE FILE.

EXAMPLE: THE /2 OPTION WAS SPECIFIED TO THE COMMAND DECODER AT RUN-TIME.

```
. R XNAME
*BIN,LOUT<SOURCE/2
```

THE SOURCE FILE CONTAINS THE FOLLOWING .LIST STATEMENTS:

```
.
.LIST       ?2-1
.
.
.LIST        1
.
```

AT THE FIRST .LIST STATEMENT, THE ?2 TERM EQUALS 1 SINCE /2 WAS SPECIFED AT RUN-TIME. THE OPERAND (?2-1) EQUALS ZERO. THEREFORE LISTING IS INHIBITED UNTIL THE SECOND .LIST INSTRUCTION. AS THE OPERAND VALUE OF THIS STATEMENT IS 1, LISTING IS ENABLED AGAIN. NOTE THAT IF THE /2 OPTION WAS NOT SPECIFIED, THE INSTRUCTIONS AFTER THE FIRST .LIST WOULD BE INCLUDED IN THE "LOUT" FILE LISTING.

8.2.0 BINARY OPERATORS.

SIX SPECIAL CHARACTERS ARE USED TO PERFORM THE FOLLOWING BINARY OPERATIONS:

*	MULTIPLICATION
/	DIVISION
&	BOOLEAN AND
!	INCLUSIVE OR
+	ADDITION
-	SUBTRACTION

THE UNARY OPERATORS TAKE PRECEDENCE OVER THE BINARY OPERATORS DURING ASSEMBLY. THE * AND / OPERATORS ARE EXECUTED NEXT, THEN THE OTHER BINARY OPERATORS FROM LEFT TO RIGHT. BRACKETS, [AND], ARE USED TO CHANGE THE ORDER OF PRECEDENCE WHEN NECESSARY. A [IS A SHIFT/K ON TTY KEYBOARDS, AND A] IS A SHIFT/M.

EXAMPLE: IN THE STATEMENT BELOW THE OPERAND EXPRESSION IS EVALUATED IN THIS ORDER: [A* [-B]] + [[2/D] * [^C [^B101]]]

```
VALUE  .EQU    A*-B+2/D*^C^B101
```

ADDITION AND SUBRACTION ARE ACCOMPLISHED BY TWO'S COMPLEMENT 16 BIT ARITHMETIC. NO CHECKS FOR OVERFLOW ARE MADE.

MULTIPLICATION IS ACCOMPLISHED BY REPEATED ADDITION. NO CHECKS FOR SIGN OR OVERFLOW ARE MADE.

DIVISION IS ACCOMPLISHED BY REPEATED SUBRACTION. THE QUOTIENT IS THE NUMBER OF SUBTRACTIONS PERFORMED. THE REMAINDER IS NOT SAVED. NO CHECKS ARE MADE FOR SIGN. DIVISION BY ZERO RESULTS IN ZERO.

THE BOOLEAN AND FUNCTION (&) IS A BIT BY BIT LOGICAL AND OF TWO NUMBERS:

THE BOOLEAN INCLUSIVE OR (!) IS A BIT BY BIT LOGICAL OR OF TWO NUMBERS.

9.0.0 PSEUDO-OPERATORS.

PSEUDO-OPERATORS ARE INSTRUCTIONS TO THE ASSEMBLER WHICH ALLOW GREATER FLEXIBILTIY IN PROGRAMMING.

A SUMMARY OF THE PSEUDO-OPS AND THEIR FUNCTIONS IS GIVEN IN THE APPENDIX.

9.1.0 ASSIGNMENT PSEUDO-OPS.

ASSIGNMENT PSEUDO-OPS ARE USED TO DEFINE VALUES, INPUT ASCII TEXT AND REASSIGN THE LOCATION COUNTER.

9.1.1 .EQU PSEUDO-OP.

THE .EQU IS USED TO ASSIGN A VALUE TO A SYMBOL. THIS SYMBOL VALUE CANNOT BE CHANGED ONCE DEFINED. .EQU IS USEFUL FOR ASSIGNING NAMES TO LOCATIONS WHICH ARE NOT LOADED BY THE OBJECT CODE.

EXAMPLE:

```
NAME1    .EQU    300*6
```

9.1.2 .SET PSEUDO-OP.

THE .SET IS USED EXACTLY LIKE THE .EQU EXCEPT THAT THE SYMBOL CAN BE REDEFINED WITH ANOTHER .SET AT ANY POINT IN THE PROGRAM:

EXAMPLE: THE FOLLOWING IS PERFECTLY LEGAL FOR A .SET BUT NOT AN .EQU.

```
NAME1    .SET    300*6
NAME1    .SET    22
```

NOTE THAT IT IS GOOD PRACTICE TO USE THE .EQU FOR ASSIGNMENTS RATHER THAN THE .SET EXCEPT (OF COURSE) WHERE THERE IS A SPECIFIC NEED TO REDEFINE A VALUE. THIS HELPS PREVENT THE ACCIDENTAL REDEFINITION OF A VALUE IN A PROGRAM.

9.1.3 .DINST PSEUDO-OP.

THE .DINST IS USED TO GIVE A MICROPROCESSOR OPERATOR ANOTHER NAME. THE ORIGINAL OPERATOR NAME WILL STILL BE VALID. NOTE THAT THE ASSEMBLER PSEUDO-OPS CANNOT BE RENAMED.

EXAMPLE: THE MICROPROCESSOR INSTRUCTION "OPR" IS DEFINED AS "NEWOP". ANY FURTHER REFERENCES TO "NEWOP" IN THE PROGRAM WILL BE TREATED ACCORDING TO THE DEFINITION OF "OPR".

```
NEWOP   .DINST  OPR
```

"NEWOP" IS DEFINED TO BE THE EQUIVALENT TO THE MICROPROCESSOR INSTRUCTION "OPR" AND IS ADDED TO THE OPERATOR SET FOR THE REMAINDER OF THE ASSEMBLY.

REFERENCES TO USER DEFINED OPERATORS ARE NOT ALLOWED TO PRECEDE THEIR .DINST STATEMENT.

ASSEMBLER PSEUDO-OPS CANNOT BE USED IN EITHER THE LABEL OR OPERAND FIELDS OF ANY STATEMENT AND THEREFORE CANNOT BE DEFINED WITH THE .DINST STATEMENT.

LOCAL SYMBOLS CANNOT BE USED IN THE OPERATOR FIELDS, THEREFORE THEY SHOULD NOT BE USED WITH A .DINST STATEMENT.

9.1.4 .ORG PSEUDO-OP.

THE .ORG REASSIGNS THE LOCATION COUNTER.

THE LOCATION COUNTER WILL BE 0 AT THE START OF THE SOURCE INPUT.

THE .ORG OPERAND CANNOT BE FORWARD REFERENCED, (REFERRED TO A LABEL DEFINED FURTHER ON IN THE PROGRAM) AND CANNOT HAVE A LABEL.

9.2.0 DEFAULT RADIX PSEUDO-OPS.

INITIALLY, THE DEFAULT RADIX IS SET TO HEXADECIMAL SO THAT CONSTANTS ARE READ IN AS BASE 16 VALUES. (SEE MODIFICATION NOTES IF ANOTHER INITIAL DEFAULT RADIX IS DESIRED.)

AT ANY POINT IN THE PROGRAM, THE DEFAULT RADIX CAN BE REASSIGNED THROUGH USE OF THESE PSEUDO-OPS:

```
.BIN            ;BINARY RADIX
.DECM           ;DECIMAL RADIX
.HEX            ;HEXADECIMAL RADIX
.OCT            ;OCTAL RADIX
```

THE DEFAULT RADIX PSEUDO-OPS CANNOT HAVE AN OPERAND OR A LABEL.

ADDITIONALLY, THE RADIX OF INDIVIDUAL CONSTANTS CAN BE SPECIFIED BY THE USE OF THE ^B, ^D, ^H AND ^O INDICATORS. SEE SECTION # 5.1.0 . THESE INDICATORS DO NOT CHANGE THE DEFAULT RADIX.

9.3.0 DATA STORAGE PSEUDO-OPS.

THREE PSEUDO-OPS CAN BE USED TO STORE DATA. THEIR FORMAT IS:

```
      LABEL      PSEUDO-OP     OPERAND,OPERAND,....     ;COMMENT
```

THE PSEUDO-OPS CAN HAVE AS MANY OPERANDS AS WILL FIT ON ONE 127 CHARACTER LINE.

EACH OPERAND CAN BE A SYMBOL,CONSTANT, OR EXPRESSION. COMMAS SEPARATE THE OPERANDS.

THE DOUBLE QUOTE (") CHARACTER IS USED DIFFERENTLY IN THE .BYTE COMMAND, BUT THE SINGLE QUOTE (') RETAINS ITS NORMAL FUNCTION.

9.3.1 .BYTE PSEUDO-OP.

THE .BYTE PSEUDO-OP STORES DATA IN SINGLE BYTES OF MEMORY. NUMERICAL BYTE VALUES CAN RANGE FROM -128 TO +255 (DECIMAL). NORMALLY, DOUBLE QUOTES AND SINGLE QUOTES ARE TREATED IDENTICALLY AND ARE USED TO FORM THE ASCII VALUE OF A SINGLE CHARACTER. HOWEVER, IN THE .BYTE PSEUDO-OP , THE DOUBLE QUOTE IS USED TO INDICATE TEXT STRINGS. DATA IS STORED SEQUENTIALLY AS IT IS PROCESSED, LEFT TO RIGHT. A TEXT STRING MUST BE CLOSED WITH A DOUBLE QUOTE.

EXAMPLE: THE ASCII VALUES OF THE TEXT ABC IS STORED:

```
      2  00      .ORG    200
200 41           .BYTE   "ABC",0,'B
201 42
202 43
203  0
204 42
```

THESE STATEMENTS WOULD BE INVALID:

```
          .BYTE   'ABC'    ;THE ' IS NOT FOR TEXT STRINGS
          .BYTE   "ABC     ;TEXT MUST END WITH A "
```

9.3.2 .DBYTE PSEUDO-OP.

THE .DBYTE IS SIMILAR TO THE .BYTE EXCEPT THAT IT STORES DOUBLE BYTE QUANTITIES. IT DOES NOT ACCEPT TEXT STRINGS. THE THE MOST SIGNIFICANT BYTE IS STORED FIRST, THEN THE LEAST SIGNIFICANT BYTE.

9.3.3 .ADDR PSEUDO-OP.

THE .ADDR PSEUDO-OP IS THE SAME AS THE .DBYTE PSEUDO-OP EXCEPT THAT THE LEAST SIGNIFICANT BYTE IS STORED FIRST. MANY MICROPROCESSORS USE THIS REVERSED FORMAT FOR ADDRESSES. FOR EXAMPLE:

```
      2 00     .ORG    200
 200  1 32     .DBYTE  ^H3132  ;HEX CONSTANT
 202 32 31     .ADDR   ^H3132  ;REVERSED BYTES
```

9.3.4 .ZERO PSEUDO-OP.

THE .ZERO PSEUDO-OP RESERVES THE NUMBER OF BYTES INDICATED BY THE OPERAND AND SETS THEM TO ZERO.

EXAMPLE: 16 ADDRESSES, 1 TO 10 (BASE 16) ARE ZEROED.

```
        0  1            .ORG      1
     1  0               .ZERO     10
    11 10               .BYTE     10
```

ONLY THE FIRST BYTE WILL BE PRINTED IN THE LISTING. THE LOCATION COUNTER IS ADVANCED. THE OPERAND OF .ZERO CANNOT BE FORWARD REFERENCED, (REFERED TO A LABEL DEFINED FURTHER ON IN THE PROGRAM).

9.4.0 LISTING CONTROL DIRECTIVES.

THROUGH USE OF THE .LIST, .PAGE AND .TITLE PSEUDO-OPS, PLUS SEVERAL RUN-TIME OPTIONS, THE SOURCE PROGRAM CAN BE LISTED IN VARIOUS WAYS AT ASSEMBLY TIME.

NORMALLY, THE ASSEMBLER AUTOMATICALLY PAGES THE OUTPUT, ADDING A HEADER AT THE TOP OF THE PAGE. (NOTE THAT PAGE NUMBERS REPRESENT THE LISTING PAGE NUMBERS, NOT INPUT FILE PAGES.)

NOT ALL PSEUDO-OPS ARE LISTED IN THE OUTPUT. THE CONDITIONAL ASSEMBLY AND LISTING CONTROL PSEUDO-OPS ARE NOT LISTED UNLESS THE /P OPTION IS SPECIFED. SEE RUN-TIME OPTIONS # 2.4.0 .

NORMALLY THE STATEMENTS WHICH ARE NOT ASSEMBLED DUE TO CONDITIONAL ASSEMBLY ARE NOT LISTED. USE OF THE /J COMMAND DECODER OPTION WILL ENABLE LISTING OF THESE STATEMENTS PLUS THE NORMALLY UNLISTED CONDITONAL ASSEMBLY PSUEDO-OPS.

THE PAGINATION AND HEADING CAN BE SUPPRESSED THROUGH USE OF THE /H COMMAND DECODER OPTION.

IF THE OUTPUT DEVICE IS ONE WHICH DOES NOT PAGE ON A FORM FEED (A TTY), THE /T DECODER OPTION CAN BE USED TO CHANGE THE FORM FEED (WHICH NORMALLY STARTS A NEW PAGE) TO 3 CARRIAGE RETURN/LINE FEEDS SO THAT PAGES WILL BE SEPARATED BY 3 BLANK LINES IN THE LISTING.

WARNING MESSAGES ARE NORMALLY OUTPUT TO BOTH THE TERMINAL AND THE SOURCE LISTING. TO INHIBIT THESE MESSAGES, THE /W DECODER OPTION IS USED.

9.4.1 .LIST PSEUDO-OP.

A LIST FLAG IS USED DURING ASSEMBLY TO INDICATE WHETHER OR NOT THE STATEMENTS ARE TO BE LISTED. INITIALLY, THE FLAG IS ON AND STAYS ON UNLESS A .LIST PSEUDO-OP IS ENCOUNTERED.

A .LIST PSEUDO-OP CAN BE USED WITH OR WITHOUT AN OPERAND. A LABEL CANNOT BE USED WITH THE .LIST PSEUDO-OP.

WHEN A .LIST PSEUDO-OP WITHOUT AN OPERAND IS ENCOUNTERED, THE LIST FLAG IS INVERTED.

EXAMPLE:

```
                                        ;LIST FLAG INITIALLY ON
            .ORG        200             ;LISTED
VALUE       .SET        1               ;LISTED
            .LIST                       ;LIST FLAG OFF
VALU2       .SET        70              ;NOT LISTED
            .LIST                       ;LIST FLAG BACK ON
```

NOTE THAT UNLESS THE /P OPTION IS USED, THE .LIST OPERATOR ITSELF WILL NOT BE LISTED.

WHEN A .LIST PSEUDO-OP WITH AN OPERAND IS ENCOUNTERED, THEN LISTING IS INHIBITED IF THE OPERAND IS EQUAL TO ZERO. (THE LIST FLAG IS SET OFF). IF THE OPERAND IS NOT ZERO, LISTING IS ENABLED. (THE LIST FLAG IS SET ON).

9.4.2 .PAGE PSEUDO-OP.

INSERTING A .PAGE PSEUDO-OP IN THE PROGRAM WILL NORMALLY START A NEW PAGE BEGINNING WITH THE NEXT LINE. (THE .PAGE STATEMENT ITSELF IS NOT NORMALLY LISTED.) IF THE /P COMMAND DECODER OPTION IS USED, THE .PAGE STATEMENT WILL BE THE FIRST LINE OF THE NEW PAGE.

THE /H COMMAND DECODER OPTION INHIBITS THE .PAGE PSEUDO-OP.

THE .PAGE PSEUDO-OP CAN HAVE NO LABEL OR OPERAND.

9.4.3 .TITLE PSEUDO-OP.

THE .TITLE IS USED TO REPLACE THE HEADING WITH UP TO 32 CHARACTERS OF TEXT. ITS FORMAT IS:

```
                    .TITLE  HEADING OF 32 CHARACTERS
```

THE FIRST CHARACTER AFTER THE .TITLE IS THE PSEUDO-OP DELIMITER WHICH CANNOT BE AN ALPHA-NUMERIC CHARACTER . THE DELIMITER IS CONSIDERED THE FIRST CHARACTER OF THE 32 CHARACTER GROUP AND WILL BE PRINTED OUT. ANY TEXT AFTER 32 CHARACTERS WILL BE IGNORED. TABS CAN BE USED IN THE HEADING.

THE /H COMMAND DECODER OPTION INHIBITS THE .TITLE PSEUDO-OP.

THE /P COMMAND DECODER ENABLES THE LISTING OF THE .TITLE PSEUDO-OP.

A SEMICOLON DOES NOT DELIMIT THE HEADING TEXT. COMMENTS CAN BE MADE ONLY AFTER THE 32 CHARACTER HEADING GROUP.

WHEN PLACED AT THE BEGINNING OF THE PROGRAM, THE .TITLE PSEUDO-OP WILL SET THE HEADING FOR THE FIRST PAGE. THE .TITLE MUST APPEAR BEFORE THE FIRST LINE TO BE LISTED.

EXAMPLE: THE FOLLOWING STATEMENTS WILL CAUSE THE HEADING OF THE FIRST PAGE TO BE "*MAIN PROGRAM".

```
        .TITLE*MAIN PROGRAM
VALUE   .EQU    1
        .LIST   VALUE
```

9.5.0 CONDITIONAL ASSEMBLY PSUEDO-OPERATORS.

THE .IFZERO, .IFNZRO, .IFDEF AND .IFNDEF OPERATORS ARE USED TO PROVIDE FOR THE CONDITIONAL ASSEMBLY IN A PROGRAM, SO THAT GROUPS OF STATEMENTS CAN BE ADDED (OR OMITTED) DURING THE ASSEMBLY PROCESS. EACH IS DESCRIBED INDIVIDUALLY IN THE SECTIONS THAT FOLLOW. ALL HAVE THE GENERAL FORM:

```
        PSEUDO-OP       OPERAND       ;COMMENT
```

EACH OPERAND MUST MEET THE CONDITIONS OF ITS PSEUDO-OP IN ORDER FOR THE STATEMENTS THAT FOLLOW IT TO BE ASSEMBLED. IF THE CONDITIONS ARE NOT MET; THESE STATEMENTS ARE OMITTED. THE .ENDC PSEUDO-OP INDICATES THE END OF THE GROUP OF STATEMENTS WHICH ARE AFFECTED. EACH CONDITIONAL PSEUDO-OP MUST HAVE ONE .ENDC STATEMENT.

CONDITIONAL PSEUDO-OPS CANNOT HAVE LABELS.

CONDITIONAL PSEUDO-OPS CAN BE NESTED UP TO 4095 LEVELS.

EXAMPLE:

```
VALUE1      .EQU        0          ;DEFINE VALUE1
            .IFZERO     VALUE1     ;VALUE1 = 0 ?  - YES.
            .BYTE       "TEXT"     ;ASSEMBLED.
            .IFDEF      VALUE2     ;VALUE2 DEFINED? - NO.
            .BYTE       "TEXT"     ;OMITTED.
            .ENDC                  ;END OF INNER CONDITIONAL
DOC         .EQU        17         ;ASSEMBLED.
            .ENDC                  ;END OF OUTER CONDITIONAL
            .
            .
```

THE CONDITIONAL PSEUDO-OPS ARE NOT INCLUDED IN THE ASSEMBLY LISTING UNLESS THE /P OR /J COMMAND DECODER OPTION IS SPECIFIED.

ONE CONDITIONAL CAN INHIBIT ANOTHER.

EXAMPLE: THREE DIFFERENT RESULTS CAN OCCUR IN THE FOLLOWING TYPE OF CONDITIONAL NESTING:

```
            CONDITIONAL 1
            .                      ;STATEMENT GROUP 1.
            CONDITIONAL 2
            .                      ;STATEMENT GROUP 2.
            .
            .ENDC                  ;END CONDITIONAL 2.
            .                      ;STATEMENT GROUP 3.
            .ENDC                  ;END CONDITIONAL 1.
```

IF BOTH CONDITIONALS ARE MET, ALL THE STATEMENTS, GROUPS 1 THROUGH 3, WILL BE ASSEMBLED.

IF CONDITIONAL 2 IS NOT MET, BUT CONDITONAL 1 IS MET, THEN GROUP 1 AND GROUP 3 WILL BE ASSEMBLED. GROUP 2 IS NOT ASSEMBLED.

IF CONDITIONAL 1 IS NOT MET, CONDITIONAL 2 IS IGNORED AND GROUPS 1 THROUGH 3 WILL NOT BE ASSEMBLED.

9.5.1 . IFZERO PSEUDO-OP.

IF THE OPERAND OF THE . IFZERO IS:

EQUAL TO ZERO	- ASSEMBLY IS UNAFFECTED.
NOT EQUAL TO ZERO	- STATEMENTS TO NEXT . ENDC ARE OMITTED.

THE OPERAND CANNOT BE FORWARD REFERENCED.

9.5.2 . IFNZRO PSEUD-OP.

IF THE OPERAND OF THE . IFNZRO IS:

EQUAL TO ZERO	- STATEMENTS TO NEXT . ENDC ARE OMITTED.
NOT EQUAL TO ZERO	- ASSEMBLY IS UNAFFECTED.

THE OPERAND CANNOT BE FORWARD REFERENCED.

9.5.3 . IFDEF PSEUDO-OP.

IF THE SYMBOL OPERAND OF THE . IFDEF IS:

DEFINED	- ASSEMBLY IS UNAFFECTED.
NOT DEFINED	- STATEMENTS TO NEXT . ENDC ARE OMITTED.

NOTE THAT . IFDEF WILL ACCEPT ONLY A SINGLE SYMBOL NAME AS THE OPERAND.

A SYMBOL IS CONSIDERED TO BE DEFINED IF IT HAS BEEN USED IN THE LABEL FIELD OF A STATEMENT PRECEEDING THE CONDITIONAL PSEUDO-OP.

9.5.4 . IFNDEF PSEUDO-OP.

IF THE SYMBOL OPERAND OF THE . IFNDEF IS:

DEFINED	- STATEMENTS TO NEXT . ENDC ARE OMITTED.
NOT DEFINED	- ASSEMBLY IS UNAFFECTED.

NOTE THAT ONLY A SINGLE SYMBOL NAME IS ALLOWED AS THE OPERAND.

A SYMBOL IS CONSIDERED TO BE DEFINED IF IT HAS BEEN USED IN THE LABEL FIELD OF A STATEMENT PRECEEDING THE CONDITIONAL PSEUDO-OP.

9.5.5 . ENDC PSEUDO-OP.

THIS PSEUDO-OP INDICATES THE END OF A CONDITONAL ASSEMBLY GROUP.

EVERY CONDITIONAL PSUEDO-OP MUST BE PAIRED WITH A . ENDC.

9.6.0 . END PSEUDO-OP.

THIS INDICATES THE END OF THE SOURCE PROGRAM. IT CANNOT HAVE EITHER A LABEL OR AN OPERAND. A WARNING MESSAGE WILL OCCUR IF THE . END STATEMENT IS LEFT OFF.

#10.0.0 ERROR MESSAGES AND WARNINGS.

BOTH PASS #1 AND PASS #2 CAN GENERATE ERROR MESSAGES. THESE ARE PRINTED ON THE CONSOLE DEVICE AS THEY OCCUR. IF A LISTING IS SPECIFIED, PASS 3 WILL LIST THE ERROR MESSAGE ABOVE THE LINE IN WHICH THE ERROR OCCURS.

ERROR MESSAGES WHICH ARE SENT TO THE CONSOLE HAVE THE FORM:

E: XX AT LABEL+N

WHERE "N" IS A DECIMAL NUMBER OF LINES BEYOND THE STATEMENT WHICH CONTAINED THE GIVEN LABEL. IF NO LABEL WAS GIVEN, "N" IS THE NUMBER OF LINES FROM THE BEGINNING LINE OF THE PROGRAM.

IF THE BINARY OUTPUT FILE IS SENT TO THE CONSOLE, AND ERROR MESSAGES OCCUR, THE OUTPUT FILE LINES AND ERROR MESSAGES WILL BE INTERMIXED. USE OF THE /E OPTION WILL INHIBIT THE ERROR MESSAGES TO THE CONSOLE SO THAT ONLY THE BINARY FILE IS OUTPUT. THIS IS USEFUL WHEN A USER WOULD LIKE TO TRY OUT CERTAIN PARTS OF A PROGRAM AND IS NOT YET CONCERNED WITH OTHER PARTS KNOWN TO HAVE ERRORS.

INDIVIDUAL ERROR MESSAGES ARE EXPLAINED IN TABLE #2 WHICH DIVIDES THE MESSAGES INTO THREE TYPES:

1) FATAL ERRORS- THESE ERRORS CAUSE THE IMMEDIATE EXIT TO THE OS/8 MONITOR. THE CURRENT OUTPUT FILE IS NOT CLOSED. /E WILL NOT INHIBIT FATAL ERROR MESSAGES. FATAL ERROR MESSAGES ARE ALWAYS SENT TO THE CONSOLE DEVICE.

2) WARNING MESSAGES INDICATE MINOR PROGRAM PROBLEMS. ASSEMBLY IS NOT HALTED. GOOD PROGRAMMING PRACTICES WILL ELIMINATE ALL WARNING MESSAGES.

3) NON-FATAL ERRORS - THE OCCURANCE OF A NON-FATAL ERROR WILL NOT HALT ASSEMBLY. THE ASSEMBLER ATTEMPTS TO DO AS MUCH OF THE LINE AS POSSIBLE. FOR EXAMPLE, IF THE OPERAND CANNOT BE EVALUATED, IT GIVES IT A VALUE OF ZERO, WRITES THE ERROR MESSAGE AND CONTINUES.

TABLE #2. #10.0.0

****

**** FATAL ERRORS ****

E:DF — DEVICE FULL:
FILE #N THERE IS NOT ENOUGH ROOM LEFT ON THE OUTPUT DEVICE FOR THE FILE. "N" INDICATES WHICH OF THE TWO OUTPUT FILES WAS IN ERROR.

E:LT — LOCAL SYMBOL TABLE OVERFLOW:
THIS ERROR OCCURS ONLY IF THE /K OPTION IS IN USE. CONVERSION OF SOME OF THE LOCAL SYMBOLS TO REGULAR SYMBOL NAMES WILL USUALLY SOLVE THIS PROBLEM. SEE THE NOTES ON THE /K RUN-TIME OPTION.

E:OE — OPEN ERROR IN OUTPUT FILE:
FILE #N AN ATTEMPT WAS MADE TO OPEN AN OUTPUT FILE ON AN INPUT-ONLY DEVIDE. (PTR:, CDR:, ETC.) "N" INDICATES WHICH ONE OF THE TWO POSSIBLE OUTPUT FILES WAS IN ERROR.

E:PE — PHASE ERROR:
A LOCATION TAG HAS A DIFFERENT ADDRESS IN ONE PASS THAN IT HAD IN THE PREVIOUS PASS.

E:RE — READ ERROR:
FILE #N AN ERROR HAS OCCURRED WHILE READING FROM AN INPUT FILE DEVICE. "N" INDICATES WHICH ONE OF THE NINE POSSIBLE INPUT FILES HAD THE ERROR.

E:ST — SYMBOL TABLE OVERFLOW:
THE PROGRAM IS TOO LARGE. WHERE CONVENIENT, DIVIDE IT AND ASSEMBLE EACH PART SEPARATELY. ALSO REFER TO THE NOTES ON THE /K RUN-TIME OPTION.

E:WE — WRITE ERROR:
FILE #N AN ERROR HAS OCCURRED WHILE WRITING TO AN OUTPUT FILE DEVICE. "N" INDICATES WHICH ONE OF THE TWO OUTPUT FILES HAD THE ERROR.

**** WARNING MESSAGES ****

W:EF — NO .END STATEMENT:
THE LAST INPUT FILE MUST HAVE AN .END STATEMENT. THE ASSEMBLER PROCEEDS AS IF AN .END WERE PRESENT.

W:UC — ASSEMBLY WAS CONDITIONALLY INHIBITED AT THE END OF THE PROGRAM: EACH CONDITIONAL ASSEMBLY PSEUDO-OP MUST BE PAIRED WITH AN .ENDC STATEMENT.

****

****

**** NON-FATAL ERRORS ****

E: BN – BAD NESTING OF BRACKETS:
EACH OPEN BRACKET MUST BE PAIRED WITH A CLOSED BRACKET.

E: DR – DIGIT OUTSIDE OF RADIX:
THE CONSTANT CONTAINS A DIGIT NOT RECOGNIZED UNDER THE SPECIFIED RADIX. FOR EXAMPLE, THE DIGIT "2" IS NOT RECOGNIZED IN BINARY RADIX. THE CONSTANT WILL BE EVALUATED AS IF THAT DIGIT WERE ZERO.

E: IL – ILLEGAL LABEL FIELD:
THE LABEL MAY NOT BE IN THE PROPER SYMBOL FORMAT, SEE SECTION #6. 2. 0 . ALSO, SOME PSEUDO-OPS CANNOT HAVE LABELS.

E: IO – ILLEGAL OPERAND VALUE:
REFER TO THE SECTION ON THE STATEMENT'S OPERATOR TO DETERMINE THE ALLOWABLE OPERAND TERMS.

E: LO – LINE INPUT OVERFLOW:
ONLY 127 CHARACTERS, NOT INCLUDING THE CARRIAGE RETURN AND LINE FEED, ARE ALLOWED IN AN INPUT LINE.

E: LS – LOCAL SYMBOL SYNTAX ERROR:
THE CORRECT FORMAT FOR A LOCAL SYMBOL IS $N WHERE "N" IS A DECIMAL NUMBER FROM 0 TO 255.

E: ML – MULTIPLE LABEL DEFINITION:
THE SAME LABEL HAS A DIFFERENT VALUE AND IS USED WITH AN OPERATOR OTHER THAN A . SET PSEUDO-OP.

E: MO – MISSING OR ILLEGAL MNEMONIC IN OPERATOR FIELD:

E: OC – OPERAND TOO COMPLEX:
TOO MANY TERMS AND OPERATORS EXIST IN THE OPERAND. DIVIDE THE EXPRESSION USING THE . SET COMMAND.

EXAMPLE: THE FIRST EXPRESSION IS DIVIDED INTO THE TWO STATEMENTS FOLLOWING IT.

```
WORD        . EQU       [ EXPR1 ] + [ EXPR2 ]

TEMP        . SET       [ EXPR1 ]
WORD        . EQU       TEMP + [ EXPR2 ]
```

E: OM – OPERAND MISSING.

****

****

E:OS - OPERAND SYNTAX ERROR.

E:PS - ILLEGAL PERMANENT SYMBOL USAGE IN OPERAND:
REFER TO THE APPENDICES TABLES TO SEE WHICH NAMES ARE USED IN THE ASSEMBLER AND MICROPROCESSOR INSTRUCTION SETS AND RENAME YOUR SYMBOL SO THAT IT WILL NOT CONFLICT.

E:TL - LABEL DEFINED TOO LATE:
ONLY ONE LEVEL OF FORWARD REFERENCING IS ALLOWED.

E:US - UNDEFINED SYMBOL:

****

NOTE: REFER TO SECTION #12.0.0 FOR ADDITIONAL ERROR MESSAGES WHICH ARE SPECIFIC TO THE TYPE OF MICROPROCESSOR BEING USED.

#11.0.0 MODIFICATION NOTES.

VARIOUS MODIFICATIONS CAN BE MADE TO THE ASSEMBLER FOR GREATER OPERATING CONVENIENCE. BEFORE MAKING ANY CHANGES, THE USER SHOULD READ THE DESCRIPTION OF EACH OPTION CAREFULLY. NO CHECKS ON PATCH VALIDITY ARE MADE. ALSO KEEP A RECORD OF ALL CHANGES SO THAT THE STATUS OF THE CROSS-ASSEMBLER IS ALWAYS KNOWN.

MODIFICATIONS ARE MADE BY PATCHING LOCATIONS IN THE IMAGE (.SV) FILE USING ODT. REFER TO THE OS/8 MANUAL FOR A DETAILED EXPLANATION OF ODT OPERATION.

THE EXAMPLE BELOW SHOWS AN ODT PATCH BEING MADE TO FILE "XNAME.SV" WHERE THE CONTENT OF LOCATION 10107 IS CHANGED FROM 3 TO 2.

```
.GET SYS:XNAME
.ODT
10107/0003 2
^C
.SA SYS:XNAME
```

#11.1.0 CHANGING THE DEFAULT INPUT FILE EXTENSION (.MS).

PATCH LOCATION 10100 TO CONTAIN THE NEW 2 CHARACTER 6 BIT ASCII EXTENSION.

#11.2.0 CHANGING THE DEFAULT BINARY OUTPUT FILE EXTENSION (.MB)

PATCH LOCATION 10101 TO CONTAIN THE NEW 2 CHARACTER 6 BIT ASCII EXTENSION.

#11.3.0 CHANGING THE DEFAULT LISTING OUTPUT FILE EXTENSION (.LS).

PATCH LOCATION 10102 TO CONTAIN THE NEW 2 CHARACTER 6 BIT ASCII EXTENSION.

#11.4.0 CHANGING THE BASE YEAR DATE.

IN OS/8 ONLY 3 BITS ARE PROVIDED TO INDICATE THE CURRENT YEAR. THIS ALLOWS ONLY NUMBERS FROM 0 TO 7 WHICH MUST BE ADDED TO A BASE YEAR TO FORM THE ACTUAL YEAR NUMBER. IN 1978 AND AT ADDITIONAL 8 YEAR INTERVALS THE BASE YEAR MUST BE CHANGED TO PROVIDE THE PROPER DATE PRINTOUT. TO DO THIS, PATCH LOCATION 10104 TO CONTAIN THE TWO CHARACTER 6 BIT ASCII REPRESENTATION OF THE TWO LEAST SIGNIFICANT DIGITS OF THE YEAR.

BASE YEAR:	PATCH TO LOCATION 10104 (IN OCTAL).
1978	6770
1986	7066
1994	7164
2002	6062

SHOULD THIS PROGRAM SURVIVE UNTIL THE YEAR 2000 THE TWO MOST SIGNIFICANT DIGITS MAY BE CHANGED BY PATCHING LOCATION 10103 TO CONTAIN 6260.

#11.5.0 CHANGING THE DEFAULT RADIX. (HEXADECIMAL)

INITIALLY THE DEFAULT RADIX IS SET TO HEXADECIMAL. THIS MAY BE MODIFIED TO BINARY, OCTAL, OR DECIMAL BY PATCHING LOCATION 10105 FROM THE FOLLOWING TABLE.

RADIX:	PATCH LOCATION 10105 TO:
OCTAL	1
HEXADECIMAL	2
DECIMAL	3
BINARY	4

#11.6.0 GENERATING 8 BIT ASCII CHARACTERS WITHIN THE BINARY PROGRAM.

THE ASCII CHARACTERS GENERATED AS OPERANDS WITH THE QUOTE CHARACTERS ARE SEVEN BIT REPRESENTATIONS TYPICAL OF MOST MICROPROCESSOR SYSTEMS. TO GENERATE EIGHT BIT ASCII WITH THE EIGHTH BIT ALWAYS SET (AS IS DONE IN SOME PDP8 SOFTWARE), PATCH LOCATION 10106 TO CONTAIN 377. (ORIGINAL CONTENT WAS 177).

#11.7.0 RUNNING UNDER OS8 VERSION 2.

THE CROSS-ASSEMBLER IS SET UP TO USE THE OS/8 VERSION 3 METHOD FOR CORE SIZE DETERMINATION. IN OS/8 V3 THE CORE SIZE IS CONTAINED IN A MONITOR LOCATION. IN PREVIOUS VERSIONS, THE CORE SIZE MUST BE DETERMINED BY ACCESSING EACH FIELD OF MEMORY TO SEE IF IT EXISTS ON THE SYSTEM. THEREFORE , TO RUN THE CROSS-ASSEMLER UNDER VERSION 2, PATCH LOCATION 10107 TO CONTAIN 2. (ORIGINAL CONTENT WAS 3).

#11.8.0 CHANGING THE NUMBER OF LINES PER PAGE. (6)

THE NORMAL NUMBER OF LINES PER PAGE IS SET AT 66. 6 OF THE 66 LINES ARE USED BY THE ASSEMBLER FOR THE HEADING AND MARGIN. TO ALTER THE NUMBER OF LINES ON A PAGE, PATCH LOCATION 10110 TO BE THE TOTAL POSITIVE LINES PER PAGE INCLUDING HEADING AND MARGIN.

#11.9.0 CHANGING THE NUMBER OF CHARACTERS PER LINE. (72)

THE TOTAL NUMBER OF CHARACTERS PRINTED ON ONE LINE (EXCLUDING CARRIAGE RETURN AND LINE FEED) IS SET AT 72 (BASE 10). TO MODIFY THIS COUNT, PATCH LOCATION 10111 TO CONTAIN THE POSITIVE NUMBER OF CHARACTERS TO BE PRINTED ON A LINE (EXCLUDING THE CR AND LF).

#11.10.0 INITIAL FORM/FEED CONTROL.

SOME LINE PRINTER HANDLERS WHEN FIRST INITIALIZED WILL ISSUE AN AUTOMATIC FORM FEED. TO AVOID EJECTING AN ADDITIONAL PAGE EACH TIME THE ASSEMBLER IS CALLED, THE FIRST FORM FEED FROM THE HEADING HAS BEEN SUPPRESSED. TO REENABLE THIS FIRST FORM FEED, PATCH LOCATION 10112 WITH 214 (BASE 8).

#11.11.0 CHANGING LABEL DELIMINATOR (,).

TO PROVIDE COMPATIBILITY WITH OTHER ASSEMBLER FORMATS AN OPTIONAL LABEL DELIMITER WILL BE ACCEPTED. NORMALLY, THIS DELIMITER IS A COMMA, BUT IT CAN BE MODIFIED TO ANY OTHER NON-ALPHANUMERIC CHARACTER (EXCEPT THE SEMICOLON OR CARRIAGE RETURN). TO MODIFY THE DELIMITING CHARACTER PATCH LOCATION 10113 WITH THE 8 BIT ASCII VALUE FOR THE CHARACTER.

#11.12.0 CHANGING FROM 8 BIT TO 7 BIT ASCII IN THE OUTPUT FILES.

ALL ASCII OUTPUT TO THE BINARY (OBJECT) AND LISTING FILES IS IN 8 BIT ASCII FORMAT. TO OUTPUT 7 BIT ASCII FORMAT PATCH LOCATION 10114 TO CONTAIN 177. (ORIGINAL CONTENT WAS 377).

#11.13.0 CHANGING THE SENSE OF THE RUN-TIME OPTIONS.

EACH SLASH OPTION (EXCEPT /0 TO /9) MAY HAVE ITS SENSE INVERTED BY PATCHING THE LOCATIONS SHOWN IN THE FOLLOWING TABLE WITH THE DESCRIBED VALUE.

OPTION:	LOCATION:	STANDARD:	INVERTED:
/B	10116	7650	7640
/E	10117	7640	7650
/H	10120	7650	7640
/J	10121	7650	7640
/K	10122	7650	7640
/L	10123	0	1
/N	10124	7650	7640
/O	10125	7650	7640
/P	10126	7640	7650
/S	10127	7650	7640
/T	10130	7650	7640
/W	10131	7650	7640

#12.0.0 8080 CROSS-ASSEMBLER SPECIFICS.

THE FIRST ELEVEN SECTIONS OF THIS MANUAL HAVE PRESENTED SIERRA DIGITAL'S UNIVERSAL ASSEMBLER FORMAT AS IT IS APPLIED TO ALL CROSS-ASSEMBLERS IN THE X8 SERIES. THIS SECTION PRESENTS ADDITIONAL INFORMATION ON THE APPLICATION OF THE UNIVERSAL ASSEMBLER FORMAT TO A SPECIFIC CROSS-ASSEMBLER FOR THE Z80 MICROPROCESSOR. THE Z80-MICROPROCESSOR WAS DESIGNED BY ZILOG, INC., 10460 BUBB ROAD, CUPERTINO, CALIFORNIA 95014 AND IS SECOND SOURCED BY MOSTEK, 1215 WEST CROSBY ROAD, CAROLLTON, TEXAS 75006. NO ATTEMPTS WILL BE MADE IN THIS MANUAL TO EXPLAIN THE OPERATION OF THE MICROPROCESSOR. EXCELLENT MANUALS COVERING THE OPERATION AND PROGRAMMING OF THE MICROPROCESSORS ARE AVAILABLE FROM THEIR MANUFACTURERS. SECTION #13 PRESENTS A SUMMARY OF THE INSTRUCTION MNUEMONIC CODES AND OPERANDS DEFINED BY ZILOG AND RECOGNIZED BY OUR CROSS-ASSEMBLER.

#12.1.0 CROSS-ASSEMBLER FILE NAMES.

THE CROSS-ASSEMBLER IS PROVIDED ON FILE STRUCTURED MEDIA UNDER THE NAMES:

XZ80.SV	- FOR THE OS/8 SAVE IMAGE FILE.
XZ80.BN	- FOR THE OS/8 BINARY FORMAT FILE.

IT IS SUGGESTED THAT THE SAME NAMING CONVENTIONS BE USED WHEN LOADING THE CROSS-ASSEMBLER FROM PAPER TAPE.

#12.2.0 RESERVED SYMBOLS

THE FOLLOWING SPECIAL SYMBOLS ARE RESERVED FOR USE TO DESIGNATE REGISTERS AND CONDITION CODES. THESE NAMES CANNOT BE USED AS USER DEFINED SYMBOLS AND WILL ONLY BE RECOGNIZED WHEN USED AS DEMONSTRATED IN THE LISTING OF SECTION 13.

SYMBOL	MEANING
A	ACCUMULATOR
AF	ACCUMULATOR AND FLAGS
B	REGISTER B
BC	REGISTER PAIR B AND C.
C	REGISTER C OR CARRY CONDITION.
D	REGISTER D
DE	REGISTER PAIR D AND E
E	REGISTER E
H	REGISTER H
HL	REGISTER PAIR H AND L
I	INTERRUPT VECTOR REGISTER
IX	INDEX REGISTER X
IY	INDEX REGISTER Y
L	REGISTER L
M	MINUS CONDITION
NC	NO CARRY CONDITION
NZ	NOT ZERO CONDITION
P	PLUS CONDITION
PE	PARITY EVEN CONDITION
PO	PARITY ODD CONDITION
R	REFRESH REGISTER
Z	ZERO CONDITION

NOTE: THERE IS NO CONFLICT BETWEEN THE '.SET' PSEUDO-OP AND THE 'SET' MICROPROCESSOR INSTRUCTION BECAUSE OF THE LEADING PERIOD ON THE PSEUDO-OP.

#12.3.0 RELATIVE ADDRESS CALCULATIONS:

THE RELATIVE ADDRESS INSTRUCTIONS 'JR' AND 'DJNZ' ALLOW A JUMP WITHIN THE RANGE OF -126 TO +129 BYTES FROM THE ADDRESS OF THE INSTRUCTION'S OP-CODE BYTE. THE CROSS-ASSEMBLER ALWAYS SUBTRACTS THE ADDRES OF THE LOCATION FOLLOWING THE RELATIVE ADDRESSING INSTRUCTION FROM THE OPERAND VALUE TO FORM THE VALUE STORED IN THE INSTRUCTION.

#12.4.0 LISTING FORMAT.

THE LISTING FILE IS OUTPUT WITH THE OBJECT CODE PRINTED TO THE LEFT OF THE SOURCE CODE LINES. AS EACH MICROPROCESSOR INSTRUCTION MAY CODE INTO ONE, TWO, OR THREE BYTES, ROOM IS PROVIDED FOR THREE COLUMNS OF GENERATED OBJECT CODE PLUS A COLUMN FOR THE ADDRESS. THE ADDRESS AND OBJECT CODE ARE NORMALLY PRINTED IN HEXADECIMAL BUT THIS MAY BE CHANGED TO OCTAL WITH THE /O COMMAND DECODER OPTION. SOURCE LINES WHICH EXCEED THE PRINTOUT LIMIT WILL CONTINUED AT COLUMN 25 (STANDARD COMMENT TAB STOP) OF THE SOURCE PRINTOUT POSITION. TABS OCCURING IN THE SOURCE PROGRAM ARE CONVERTED TO THE PROPER NUMBER OF BLANK CHARACTERS BY THE ASSEMBLER. THIS IS DONE BY THE ASSEMBLER RATHER THAN THE DEVICE HANDLER OR DEVICE BECAUSE THE BEGINNING OF THE SOURCE PRINTOUT DOES NOT OCCUR ON A STANDARD TAB STOP.

#12.5.0 BINARY FILE OUTPUT:

THE OBJECT (BINARY) OUTPUT IS COMPATIBLE WITH THE INTEL HEXADECIMAL OBJECT CODE FORMAT. THE OUTPUT FILE CONSISTS OF ASCII TEST REPRESENTING HEXADECIMAL NUMBERS IN THE FOLLOWING FORMAT:

LEADER STRINGS OF 100 NULL CHARACTERS PRECEED AND FOLLOW THE OBJECT OUTPUT. EACH LINE BEGINS WITH A COLON AND IS FOLLOWED BY A TWO HEX DIGIT BYTE COUNT, A FOUR HEX DIGIT ADDRESS, A TWO HEX DIGIT RECORD TYPE (ALWAYS 0), UP TO 16 BYTES OF DATA (EACH 2 HEX DIGITS), AND A TWO HEX DIGIT CHECKSUM.

EXAMPLE:

:CCAAAATTDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDSS

WHERE:

CC	IS THE TWO HEXADECIMAL DIGIT COUNT FOR THE NUMBER OF DATA BYTES (REPRESENTED BY PAIRS OF D'S) IN THE LINE. A COUNT OF ZERO INDICATES THE TERMINATION OF THE OBJECT OUTPUT. (:00)
AAAA	IS THE HEXADECIMAL ADDRESS FOR STORING THE FIRST DATA BYTE. EACH ADDITIONAL DATA BYTE IS TO BE STORED IN SEQUENTIAL ADDRESSES. THE ADDRESS IS PRESENTED WITH ITS MOST SIGNIFICANT BYTE FIRST.
TT	IS THE TWO HEXADECIMAL RECORD TYPE. THIS INDICATOR IS CURRENTLY UNUSED AND ASSIGNED A VALUE OF 00.
DD	REPRESENTS TWO HEXADECIMAL DIGITS FOR A BYTE OF OBJECT (BINARY) CODE. UP TO 16 BYTES MAY BE OUTPUT ON ONE LINE.
SS	IS THE TWO HEXADECIMAL DIGIT CHECKSUM OF THE LINE. ALL EIGHT BIT BYTES IN THE LINE AFTER THE RECORD MARK (':') ARE SUMMED. THE LEAST SIGNIFICANT BYTE OF THE NEGATIVE OF THIS VALUE IS THE CHECKSUM. THUS, IF ALL BYTES IN THE LINE ARE ADDED TOGETHER WITH CARRYS IGNORED, AND THIS SUM IS ADDED TO THE CHECKSUM, THE RESULT WILL BE ZERO.

THE BINARY OUTPUT FILE CAN BE CHANGED TO BNPF FORMAT THROUGH THE USE OF THE /B RUN-TIME OPTION. SECTION #2.4.0 DESCRIBES THE BNPF OUTPUT.

#12.6.0 ADDITIONAL ERROR MESSAGE FOR THE Z80:

STANDARD ERROR:

E:JR RELATIVE JUMP ADDRESS OUT OF RANGE.
THE OPERAND ADDRESS WAS OUT OF THE RANGE FROM THE REQUIRED -126 TO +129 (DECIMAL) BYTES FROM THE FIRST BYTE OF THE RELATIVE ADDRESSING INSTRUCTION.

```
.R XZ80
*TTY:,TTY:<SAZ80/J/P/1
  E:MO AT POSITN+  21
  E:MO AT POSITN+  21

:10100000DB00CB7F28FAE60F0F4FDB01CB7F200DF3
:10101000CD2A103E80D300AFD300FF18E3E67FCD8A
:10102000241018EFCD3310B612C9CD3310477E2FE0
:10103000A012C947E678810F0F0FC6805F3E00CE31
:1010400030577BE607C64C6F26101AC9010204080B
:0410500010204080AC
:000000
```

```
SAMPLE ROUTINE                    JUL  1, 1977    XZ80--V1A   PAGE    1

                                  .TITLE SAMPLE ROUTINE
                        ;         THIS ROUTINE READS IN STATUS INFORMATION AND
                        ;           UPDATES SIXTEEN DIFFERENT 128 BIT FLAG TABLES
                        ;           IN RAM.
          0   0         IPORT1    .EQU    0
          0   1         IPORT2    .EQU    1
          0   0         OPORT1    .EQU    0
         30  80         TABLES    .EQU    3080      ;BASE OF FLAG TABLES
         10   0                   .ORG    1000
  1000   DB   0         LOOP      IN      A,(IPORT1) ;GET READY FLAG AND TABLE #
  1002   CB  7F                   BIT     7,A
  1004   28  FA                   JR      Z,LOOP    ;WAIT UNTIL DATA READY
  1006   E6   F                   AND     0F
  1008    F                       RRCA
  1009   4F                       LD      C,A       ;SAVE SHIFTED TABLE NUMBER
  100A   DB   1                   IN      A,(IPORT2) ;GET BIT POSITION NUMBER
  100C   CB  7F                   BIT     7,A
  100E   20   D                   JR      NZ,$2     ;MOST SIG BIT MEANS SET BIT ON
  1010   CD  2A  10               CALL    CLEARB
  1013   3E  80         $1        LD      A,80      ;STROBE ACKNOWLEDGE LINE
  1015   D3   0                   OUT     (OPORT1),A
  1017   AF                       XOR     A
  1018   D3   0                   OUT     (OPORT1),A
                                  .IFNZRO ?1        ;USER FLAG 1 IS SELECTED FOR
  101A   FF                       RST     38        ;DEBUGGING.   A BREAKPOINT CALL
                                  .ENDC             ;(RST 38) IS INSERTED FOR
                                  .IFZERO ?1        ;DEBUGGING AND A 'NOP' IS
                                  NOP               ;INSERTED FOR NORMAL OPERATION
                                  .ENDC
  101B   18  E3                   JR      LOOP      ;LOOP BACK FOR ANOTHER TRY
  101D   E6  7F         $2        AND     ^B 01111111      ;MASK OUT SIGN
  101F   CD  24  10               CALL    SETB      ;SET BIT INTO TABLE
  1022   18  EF                   JR      $1
```

```
                        .PAGE
                ;       SUBROUTINES TO SET AND CLEAR BITS IN A TABLE
1024 CD 33 10 SETB      CALL    POSITN  ;POSITION THE POINTERS
1027 B6                 OR      (HL)    ;OR IN THE DECODED BIT
1028 12                 LD      (DE),A  ;STORE RESULT BACK
1029 C9                 RET
102A CD 33 10 CLEARB    CALL    POSITN  ;POSITION THE POINTERS
102D 47                 LD      B,A     ;SAVE PREVIOUS BYTE TEMPORARILY
102E 7E                 LD      A,(HL)  ;GET DECODED BIT
102F 2F                 CPL
1030 A0                 AND     B       ;MASK OUT SELECTED BIT
1031 12                 LD      (DE),A  ;STORE BACK RESULT
1032 C9                 RET
                ;       ROUTINE TO POSITION THE TABLE BYTE POINTER
                ;          AND DECODE THE BIT POSITION.
1033 47         POSITN  LD      B,A     ;SAVE TEMPORARILY
1034 E6 78              AND     ^O 170  ;MASK FOR BYTE NUMBER IN TABLE
1036 81                 ADD     A,C     ;COMBINE WITH TABLE NUMBER
1037  F                 RRCA            ;AND FORM BYTE ADDRESS
1038  F                 RRCA
1039  F                 RRCA
103A C6 80              ADD     A,^L TABLES
103C 5F                 LD      E,A     ;SET UP ADDRESS IN D,E
103D 3E  0              LD      A,0
103F CE 30              ADC     A,^M TABLES
1041 57                 LD      D,A
1042 78                 LD      A,B     ;DECODE BIT NUMBER WITHIN BYTE
1043 E6  7              AND     7
1045 C6 4C              ADD     A,^L $1 ;FORM LOOKUP TABLE ADDRESS
1047 6F                 LD      L,A     ;FOR DECODED BIT
1048 26 10              LD      H,^M $1
104A 1A                 LD      A,(DE)  ;GET TABLE BYTE
104B C9                 RET
                        .BIN            ;TABLE IS IN BINARY
104C  1         $1      .BYTE   1,10,100,1000
104D  2
104E  4
104F  8
1050 10                 .BYTE   10000,100000,1000000,10000000
1051 20
1052 40
1053 80
******  E:MD
                        JUNK            ;SAMPLE ERROR
                        .END
```



```
  102A CLEARB       0 IPORT1      1 IPORT2    1000 LOOP
     0 OPORT1    1033 POSITN    1024 SETB     3080 TABLES
```

ERRORS: 1

```
                ;       THIS SECTION IS AN ALPHABETICAL LISTING OF THE
                ;       Z80 INSTRUCTION SET WITH ALL POSSIBLE OPERAND
                ;       TYPE VARIATIONS.
                ;
                ;       ADD WITH CARRY INSTRUCTIONS
 0 8E                   ADC     A,(HL)
 1 DD 8E  5             ADC     A,(IX+INDEX)
 4 FD 8E  5             ADC     A,(IY+INDEX)
 7 8F                   ADC     A,A
 8 88                   ADC     A,B
 9 89                   ADC     A,C
 A 8A                   ADC     A,D
 B 8B                   ADC     A,E
 C 8C                   ADC     A,H
 D 8D                   ADC     A,L
 E CE FF                ADC     A,BYT
10 ED 4A                ADC     HL,BC
12 ED 5A                ADC     HL,DE
14 ED 6A                ADC     HL,HL
16 ED 7A                ADC     HL,SP
                ;
                ;       ADD WITHOUT CARRY INSTRUCTIONS
18 86                   ADD     A,(HL)
19 DD 86  5             ADD     A,(IX+INDEX)
1C FD 86  5             ADD     A,(IY+INDEX)
1F 87                   ADD     A,A
20 80                   ADD     A,B
21 81                   ADD     A,C
22 82                   ADD     A,D
23 83                   ADD     A,E
24 84                   ADD     A,H
25 85                   ADD     A,L
26 C6 FF                ADD     A,BYT
28  9                   ADD     HL,BC
29 19                   ADD     HL,DE
2A 29                   ADD     HL,HL
2B 39                   ADD     HL,SP
2C DD  9                ADD     IX,BC
2E DD 19                ADD     IX,DE
30 DD 29                ADD     IX,IX
32 DD 39                ADD     IX,SP
34 FD  9                ADD     IY,BC
36 FD 19                ADD     IY,DE
38 FD 29                ADD     IY,IY
3A FD 39                ADD     IY,SP
3C A6                   AND     (HL)
                ;
                ;       LOGICAL AND INSTRUCTIONS
3D DD A6  5             AND     (IX+INDEX)
40 FD A6  5             AND     (IY+INDEX)
43 A7                   AND     A
44 A0                   AND     B
45 A1                   AND     C
46 A2                   AND     D
47 A3                   AND     E
48 A4                   AND     H
49 A5                   AND     L
4A E6 FF                AND     BYT
```

```
                ;
                ;       TEST BIT INSTRUCTIONS
4C CB 46                BIT     0,(HL)
4E DD CB                BIT     0,(IX+INDEX)
50  5 46
52 FD CB                BIT     0,(IY+INDEX)
54  5 46
56 CB 47                BIT     0,A
58 CB 40                BIT     0,B
5A CB 41                BIT     0,C
5C CB 42                BIT     0,D
5E CB 43                BIT     0,E
60 CB 44                BIT     0,H
62 CB 45                BIT     0,L
64 CB 4E                BIT     1,(HL)
66 DD CB                BIT     1,(IX+INDEX)
68  5 4E
6A FD CB                BIT     1,(IY+INDEX)
6C  5 4E
6E CB 4F                BIT     1,A
70 CB 48                BIT     1,B
72 CB 49                BIT     1,C
74 CB 4A                BIT     1,D
76 CB 4B                BIT     1,E
78 CB 4C                BIT     1,H
7A CB 4D                BIT     1,L
7C CB 56                BIT     2,(HL)
7E DD CB                BIT     2,(IX+INDEX)
80  5 56
82 FD CB                BIT     2,(IY+INDEX)
84  5 56
86 CB 57                BIT     2,A
88 CB 50                BIT     2,B
8A CB 51                BIT     2,C
8C CB 52                BIT     2,D
8E CB 53                BIT     2,E
90 CB 54                BIT     2,H
92 CB 55                BIT     2,L
94 CB 5E                BIT     3,(HL)
96 DD CB                BIT     3,(IX+INDEX)
98  5 5E
9A FD CB                BIT     3,(IY+INDEX)
9C  5 5E
9E CB 5F                BIT     3,A
A0 CB 58                BIT     3,B
A2 CB 59                BIT     3,C
A4 CB 5A                BIT     3,D
A6 CB 5B                BIT     3,E
A8 CB 5C                BIT     3,H
AA CB 5D                BIT     3,L
AC CB 66                BIT     4,(HL)
AE DD CB                BIT     4,(IX+INDEX)
B0  5 66
B2 FD CB                BIT     4,(IY+INDEX)
B4  5 66
B6 CB 67                BIT     4,A
B8 CB 60                BIT     4,B
BA CB 61                BIT     4,C
```

```
 BC CB 62               BIT     4,D
 BE CB 63               BIT     4,E
 C0 CB 64               BIT     4,H
 C2 CB 65               BIT     4,L
 C4 CB 6E               BIT     5,(HL)
 C6 DD CB               BIT     5,(IX+INDEX)
 C8  5 6E
 CA FD CB               BIT     5,(IY+INDEX)
 CC  5 6E
 CE CB 6F               BIT     5,A
 D0 CB 68               BIT     5,B
 D2 CB 69               BIT     5,C
 D4 CB 6A               BIT     5,D
 D6 CB 6B               BIT     5,E
 D8 CB 6C               BIT     5,H
 DA CB 6D               BIT     5,L
 DC CB 76               BIT     6,(HL)
 DE DD CB               BIT     6,(IX+INDEX)
 E0  5 76
 E2 FD CB               BIT     6,(IY+INDEX)
 E4  5 76
 E6 CB 77               BIT     6,A
 E8 CB 70               BIT     6,B
 EA CB 71               BIT     6,C
 EC CB 72               BIT     6,D
 EE CB 73               BIT     6,E
 F0 CB 74               BIT     6,H
 F2 CB 75               BIT     6,L
 F4 CB 7E               BIT     7,(HL)
 F6 DD CB               BIT     7,(IX+INDEX)
 F8  5 7E
 FA FD CB               BIT     7,(IY+INDEX)
 FC  5 7E
 FE CB 7F               BIT     7,A
100 CB 78               BIT     7,B
102 CB 79               BIT     7,C
104 CB 7A               BIT     7,D
106 CB 7B               BIT     7,E
108 CB 7C               BIT     7,H
10A CB 7D               BIT     7,L
                ;
                ;       CALL SUBROUTINE INSTRUCTIONS
10C DC 34 12            CALL    C,NN
10F FC 34 12            CALL    M,NN
112 D4 34 12            CALL    NC,NN
115 CD 34 12            CALL    NN
118 C4 34 12            CALL    NZ,NN
11B F4 34 12            CALL    P,NN
11E EC 34 12            CALL    PE,NN
121 E4 34 12            CALL    PO,NN
124 CC 34 12            CALL    Z,NN
                ;
127 3F                  CCF             ;COMPLEMENT CARRY FLAG
                ;
                ;       COMPARE INSTRUCTIONS
128 BE                  CP      (HL)
129 DD BE  5            CP      (IX+INDEX)
12C FD BE  5            CP      (IY+INDEX)
```

```
12F BF                  CP      A
130 B8                  CP      B
131 B9                  CP      C
132 BA                  CP      D
133 BB                  CP      E
134 BC                  CP      H
135 BD                  CP      L
136 FE FF               CP      BYT
                ;
138 ED A9               CPD             ;COMPARE AND DECREMENT
13A ED B9               CPDR            ;COMPARE, DECREMENT, AND REPEAT
13C ED A1               CPI             ;COMPARE AND INCREMENT
13E ED B1               CPIR            ;COMPARE, INCREMENT AND REPEAT
140 2F                  CPL             ;COMPLEMENT ACCUMULATOR
141 27                  DAA             ;DECIMAL ADJUST ACCUMULATOR
                ;
                ;       DECREMENT INSTRUCTIONS
142 35                  DEC     (HL)
143 DD 35  5            DEC     (IX+INDEX)
146 FD 35  5            DEC     (IY+INDEX)
149 3D                  DEC     A
14A  5                  DEC     B
14B  B                  DEC     BC
14C  D                  DEC     C
14D 15                  DEC     D
14E 1B                  DEC     DE
14F 1D                  DEC     E
150 25                  DEC     H
151 2B                  DEC     HL
152 DD 2B               DEC     IX
154 FD 2B               DEC     IY
156 2D                  DEC     L
157 3B                  DEC     SP
                ;
158 F3                  DI              ;DISABLE INTERRUPTS
159 10  0               DJNZ    LABEL1  ;DECREMENT B, JUMP RELATIVE ON
                                        NON 0
15B FB          LABEL1  EI              ;ENABLE INTERRUPTS
                ;
                ;       EXCHANGE INSTRUCTIONS
15C E3                  EX      (SP),HL
15D DD E3               EX      (SP),IX
15F FD E3               EX      (SP),IY
161  8                  EX      AF,AF'
162 EB                  EX      DE,HL
                ;
163 D9                  EXX             ;EXCHANGE REGISTER BANKS
164 76                  HALT            ;HALT THE PROCESSOR
                ;
                ;       SET INTERRUPT MODE INSTRUCTIONS
165 ED 46               IM      0
167 ED 56               IM      1
169 ED 5E               IM      2
                ;
                ;       INPUT INSTRUCTIONS
16B ED 78               IN      A,(C)
16D DB 20               IN      A,(N)
16F ED 40               IN      B,(C)
```

```
171 ED 48              IN      C,(C)
173 ED 50              IN      D,(C)
175 ED 58              IN      E,(C)
177 ED 60              IN      H,(C)
179 ED 68              IN      L,(C)
                ;
                ;      INCREMENT INSTRUCTIONS
17B 34                 INC     (HL)
17C DD 34  5           INC     (IX+INDEX)
17F FD 34  5           INC     (IY+INDEX)
182 3C                 INC     A
183  4                 INC     B
184  3                 INC     BC
185  C                 INC     C
186 14                 INC     D
187 13                 INC     DE
188 1C                 INC     E
189 24                 INC     H
18A 23                 INC     HL
18B DD 23              INC     IX
18D FD 23              INC     IY
18F 2C                 INC     L
190 33                 INC     SP
                ;
191 ED AA              IND             ;INPUT AND DECREMENT
193 ED BA              INDR            ;INPUT, DECREMENT, AND REPEAT
195 ED A2              INI             ;INPUT AND INCREMENT
197 ED B2              INIR            ;INPUT, INCREMENT, AND REPEAT
                ;
                ;      JUMP ABSOLUTE INSTRUCTIONS
199 E9                 JP      (HL)
19A DD E9              JP      (IX)
19C FD E9              JP      (IY)
19E DA 34 12           JP      C,NN
1A1 FA 34 12           JP      M,NN
1A4 D2 34 12           JP      NC,NN
1A7 C3 34 12           JP      NN
1AA C2 34 12           JP      NZ,NN
1AD F2 34 12           JP      P,NN
1B0 EA 34 12           JP      PE,NN
1B3 E2 34 12           JP      PO,NN
1B6 CA 34 12           JP      Z,NN
                ;
                ;      JUMP RELATIVE INSTRUCTIONS
1B9 38  8              JR      C,LABEL2
1BB 18  6              JR      LABEL2
1BD 30  4              JR      NC,LABEL2
1BF 20  2              JR      NZ,LABEL2
1C1 28  0              JR      Z,LABEL2
                ;
                ;      LOAD INSTRUCTIONS
1C3  2         LABEL2  LD      (BC),A
1C4 12                 LD      (DE),A
1C5 77                 LD      (HL),A
1C6 70                 LD      (HL),B
1C7 71                 LD      (HL),C
1C8 72                 LD      (HL),D
1C9 73                 LD      (HL),E
```

```
1CA 74             LD      (HL),H
1CB 75             LD      (HL),L
1CC 36 FF          LD      (HL),BYT
1CE DD 77  5       LD      (IX+INDEX),A
1D1 DD 70  5       LD      (IX+INDEX),B
1D4 DD 71  5       LD      (IX+INDEX),C
1D7 DD 72  5       LD      (IX+INDEX),D
1DA DD 73  5       LD      (IX+INDEX),E
1DD DD 74  5       LD      (IX+INDEX),H
1E0 DD 75  5       LD      (IX+INDEX),L
1E3 DD 36          LD      (IX+INDEX),BYT
1E5  5 FF
1E7 FD 77  5       LD      (IY+INDEX),A
1EA FD 70  5       LD      (IY+INDEX),B
1ED FD 71  5       LD      (IY+INDEX),C
1F0 FD 72  5       LD      (IY+INDEX),D
1F3 FD 73  5       LD      (IY+INDEX),E
1F6 FD 74  5       LD      (IY+INDEX),H
1F9 FD 75  5       LD      (IY+INDEX),L
1FC FD 36          LD      (IY+INDEX),BYT
1FE  5 FF
200 32 34 12       LD      (NN),A
203 ED 43          LD      (NN),BC
205 34 12
207 ED 53          LD      (NN),DE
209 34 12
20B 22 34 12       LD      (NN),HL
20E DD 22          LD      (NN),IX
210 34 12
212 FD 22          LD      (NN),IY
214 34 12
216 ED 73          LD      (NN),SP
218 34 12
21A  A             LD      A,(BC)
21B 1A             LD      A,(DE)
21C 7E             LD      A,(HL)
21D DD 7E  5       LD      A,(IX+INDEX)
220 FD 7E  5       LD      A,(IY+INDEX)
223 3A 34 12       LD      A,(NN)
226 7F             LD      A,A
227 78             LD      A,B
228 79             LD      A,C
229 7A             LD      A,D
22A 7B             LD      A,E
22B 7C             LD      A,H
22C ED 57          LD      A,I
22E 7D             LD      A,L
22F 3E FF          LD      A,BYT
231 46             LD      B,(HL)
232 DD 46  5       LD      B,(IX+INDEX)
235 FD 46  5       LD      B,(IY+INDEX)
238 47             LD      B,A
239 40             LD      B,B
23A 41             LD      B,C
23B 42             LD      B,D
23C 43             LD      B,E
23D 44             LD      B,H
23E 45             LD      B,L
```

```
23F  6 FF            LD      B,BYT
241 ED 4B            LD      BC,(NN)
243 34 12
245  1 34 12         LD      BC,NN
248 4E               LD      C,(HL)
249 DD 4E  5         LD      C,(IX+INDEX)
24C FD 4E  5         LD      C,(IY+INDEX)
24F 4F               LD      C,A
250 48               LD      C,B
251 49               LD      C,C
252 4A               LD      C,D
253 4B               LD      C,E
254 4C               LD      C,H
255 4D               LD      C,L
256  E FF            LD      C,BYT
258 56               LD      D,(HL)
259 DD 56  5         LD      D,(IX+INDEX)
25C FD 56  5         LD      D,(IY+INDEX)
25F 57               LD      D,A
260 50               LD      D,B
261 51               LD      D,C
262 52               LD      D,D
263 53               LD      D,E
264 54               LD      D,H
265 55               LD      D,L
266 16 FF            LD      D,BYT
268 ED 5B            LD      DE,(NN)
26A 34 12
26C 11 34 12         LD      DE,NN
26F 5E               LD      E,(HL)
270 DD 5E  5         LD      E,(IX+INDEX)
273 FD 5E  5         LD      E,(IY+INDEX)
276 5F               LD      E,A
277 58               LD      E,B
278 59               LD      E,C
279 5A               LD      E,D
27A 5B               LD      E,E
27B 5C               LD      E,H
27C 5D               LD      E,L
27D 1E FF            LD      E,BYT
27F 66               LD      H,(HL)
280 DD 66  5         LD      H,(IX+INDEX)
283 FD 66  5         LD      H,(IY+INDEX)
286 67               LD      H,A
287 60               LD      H,B
288 61               LD      H,C
289 62               LD      H,D
28A 63               LD      H,E
28B 64               LD      H,H
28C 65               LD      H,L
28D 26 FF            LD      H,BYT
28F 2A 34 12         LD      HL,(NN)
292 21 34 12         LD      HL,NN
295 ED 47            LD      I,A
297 DD 2A            LD      IX,(NN)
299 34 12
29B DD 21            LD      IX,NN
29D 34 12
```

```
29F FD 2A             LD      IY,(NN)
2A1 34 12
2A3 FD 21             LD      IY,NN
2A5 34 12
2A7 6E                LD      L,(HL)
2A8 DD 6E  5          LD      L,(IX+INDEX)
2AB FD 6E  5          LD      L,(IY+INDEX)
2AE 6F                LD      L,A
2AF 68                LD      L,B
2B0 69                LD      L,C
2B1 6A                LD      L,D
2B2 6B                LD      L,E
2B3 6C                LD      L,H
2B4 6D                LD      L,L
2B5 2E FF             LD      L,BYT
2B7 ED 7B             LD      SP,(NN)
2B9 34 12
2BB F9                LD      SP,HL
2BC DD F9             LD      SP,IX
2BE FD F9             LD      SP,IY
2C0 31 34 12          LD      SP,NN
                ;
2C3 ED A8             LDD               ;LOAD AND DECREMENT
2C5 ED B8             LDDR              ;LOAD, DECREMENT, AND REPEAT
2C7 ED A0             LDI               ;LOAD AND INCREMENT
2C9 ED B0             LDIR              ;LOAD, INCREMENT, AND REPEAT
2CB ED 44             NEG               ;NEGATE ACCUMULATOR
2CD  0                NOP               ;NO OPERATION
                ;
                ;       LOGICAL OR INSTRUCTIONS
2CE B6                OR      (HL)
2CF DD B6  5          OR      (IX+INDEX)
2D2 FD B6  5          OR      (IY+INDEX)
2D5 B7                OR      A
2D6 B0                OR      B
2D7 B1                OR      C
2D8 B2                OR      D
2D9 B3                OR      E
2DA B4                OR      H
2DB B5                OR      L
2DC F6 FF             OR      BYT
                ;
2DE ED BB             OTDR              ;OUTPUT, DECREMENT, AND REPEAT
2E0 ED B3             OTIR              ;OUTPUT, INCREMENT, AND REPEAT
                ;
                ;       OUTPUT INSTRUCTIONS
2E2 ED 79             OUT     (C),A
2E4 ED 41             OUT     (C),B
2E6 ED 49             OUT     (C),C
2E8 ED 51             OUT     (C),D
2EA ED 59             OUT     (C),E
2EC ED 61             OUT     (C),H
2EE ED 69             OUT     (C),L
2F0 D3 20             OUT     (N),A
                ;
2F2 ED AB             OUTD              ;OUTPUT AND DECREMENT
2F4 ED A3             OUTI              ;OUTPUT AND INCREMENT
                ;
```

```
                        ;       POP STACK INSTRUCTIONS
2F6 F1                          POP     AF
2F7 C1                          POP     BC
2F8 D1                          POP     DE
2F9 E1                          POP     HL
2FA DD E1                       POP     IX
2FC FD E1                       POP     IY
                        ;
                        ;       PUSH STACK INSTRUCTIONS
2FE F5                          PUSH    AF
2FF C5                          PUSH    BC
300 D5                          PUSH    DE
301 E5                          PUSH    HL
302 DD E5                       PUSH    IX
304 FD E5                       PUSH    IY
                        ;
                        ;       RESET BIT INSTRUCTIONS
306 CB 86                       RES     0,(HL)
308 DD CB                       RES     0,(IX+INDEX)
30A  5 86
30C FD CB                       RES     0,(IY+INDEX)
30E  5 86
310 CB 87                       RES     0,A
312 CB 80                       RES     0,B
314 CB 81                       RES     0,C
316 CB 82                       RES     0,D
318 CB 83                       RES     0,E
31A CB 84                       RES     0,H
31C CB 85                       RES     0,L
31E CB 8E                       RES     1,(HL)
320 DD CB                       RES     1,(IX+INDEX)
322  5 8E
324 FD CB                       RES     1,(IY+INDEX)
326  5 8E
328 CB 8F                       RES     1,A
32A CB 88                       RES     1,B
32C CB 89                       RES     1,C
32E CB 8A                       RES     1,D
330 CB 8B                       RES     1,E
332 CB 8C                       RES     1,H
334 CB 8D                       RES     1,L
336 CB 96                       RES     2,(HL)
338 DD CB                       RES     2,(IX+INDEX)
33A  5 96
33C FD CB                       RES     2,(IY+INDEX)
33E  5 96
340 CB 97                       RES     2,A
342 CB 90                       RES     2,B
344 CB 91                       RES     2,C
346 CB 92                       RES     2,D
348 CB 93                       RES     2,E
34A CB 94                       RES     2,H
34C CB 95                       RES     2,L
34E CB 9E                       RES     3,(HL)
350 DD CB                       RES     3,(IX+INDEX)
352  5 9E
354 FD CB                       RES     3,(IY+INDEX)
356  5 9E
```

```
358 CB 9F               RES     3,A
35A CB 98               RES     3,B
35C CB 99               RES     3,C
35E CB 9A               RES     3,D
360 CB 9B               RES     3,E
362 CB 9C               RES     3,H
364 CB 9D               RES     3,L
366 CB A6               RES     4,(HL)
368 DD CB               RES     4,(IX+INDEX)
36A  5 A6
36C FD CB               RES     4,(IY+INDEX)
36E  5 A6
370 CB A7               RES     4,A
372 CB A0               RES     4,B
374 CB A1               RES     4,C
376 CB A2               RES     4,D
378 CB A3               RES     4,E
37A CB A4               RES     4,H
37C CB A5               RES     4,L
37E CB AE               RES     5,(HL)
380 DD CB               RES     5,(IX+INDEX)
382  5 AE
384 FD CB               RES     5,(IY+INDEX)
386  5 AE
388 CB AF               RES     5,A
38A CB A8               RES     5,B
38C CB A9               RES     5,C
38E CB AA               RES     5,D
390 CB AB               RES     5,E
392 CB AC               RES     5,H
394 CB AD               RES     5,L
396 CB B6               RES     6,(HL)
398 DD CB               RES     6,(IX+INDEX)
39A  5 B6
39C FD CB               RES     6,(IY+INDEX)
39E  5 B6
3A0 CB B7               RES     6,A
3A2 CB B0               RES     6,B
3A4 CB B1               RES     6,C
3A6 CB B2               RES     6,D
3A8 CB B3               RES     6,E
3AA CB B4               RES     6,H
3AC CB B5               RES     6,L
3AE CB BE               RES     7,(HL)
3B0 DD CB               RES     7,(IX+INDEX)
3B2  5 BE
3B4 FD CB               RES     7,(IY+INDEX)
3B6  5 BE
3B8 CB BF               RES     7,A
3BA CB B8               RES     7,B
3BC CB B9               RES     7,C
3BE CB BA               RES     7,D
3C0 CB BB               RES     7,E
3C2 CB BC               RES     7,H
3C4 CB BD               RES     7,L
                ;
                ;       RETURN FROM SUBROUTINE INSTRUCTIONS
3C6 C9                  RET
```

```
3C7 D8              RET     C
3C8 F8              RET     M
3C9 D0              RET     NC
3CA C0              RET     NZ
3CB F0              RET     P
3CC E8              RET     PE
3CD E0              RET     PO
3CE C8              RET     Z
                ;
3CF ED 4D           RETI            ;RETURN FROM INTERUPT
3D1 ED 45           RETN            ;RETURN FROM NON MASKABLE INTER
                                    UPT
                ;
                ;       ROTATE LEFT THROUGH CARRY INSTRUCTIONS
3D3 CB 16           RL      (HL)
3D5 DD CB           RL      (IX+INDEX)
3D7  5 16
3D9 FD CB           RL      (IY+INDEX)
3DB  5 16
3DD CB 17           RL      A
3DF CB 10           RL      B
3E1 CB 11           RL      C
3E3 CB 12           RL      D
3E5 CB 13           RL      E
3E7 CB 14           RL      H
3E9 CB 15           RL      L
                ;
3EB 17              RLA             ;ROTATE ACC LEFT THROUGH CARRY
                ;
                ;       ROTATE LEFT CIRCULAR INSTRUCTIONS
3EC CB  6           RLC     (HL)
3EE DD CB           RLC     (IX+INDEX)
3F0  5  6
3F2 FD CB           RLC     (IY+INDEX)
3F4  5  6
3F6 CB  7           RLC     A
3F8 CB  0           RLC     B
3FA CB  1           RLC     C
3FC CB  2           RLC     D
3FE CB  3           RLC     E
400 CB  4           RLC     H
402 CB  5           RLC     L
                ;
404  7              RLCA            ;ROTATE ACC LEFT CIRCULAR
405 ED 6F           RLD             ;ROTATE DIGIT LEFT
                ;
                ;       ROTATE RIGHT THROUGH CARRY INSTRUCTIONS
407 CB 1E           RR      (HL)
409 DD CB           RR      (IX+INDEX)
40B  5 1E
40D FD CB           RR      (IY+INDEX)
40F  5 1E
411 CB 1F           RR      A
413 CB 18           RR      B
415 CB 19           RR      C
417 CB 1A           RR      D
419 CB 1B           RR      E
41B CB 1C           RR      H
```

```
41D CB 1D               RR      L
                ;
41F 1F                  RRA             ;ROTATE ACC RIGHT THROUGH CARRY
                ;
                ;       ROTATE RIGHT CIRCULAR INSTRUCTIONS
420 CB  E               RRC     (HL)
422 DD CB               RRC     (IX+INDEX)
424  5  E
426 FD CB               RRC     (IY+INDEX)
428  5  E
42A CB  F               RRC     A
42C CB  8               RRC     B
42E CB  9               RRC     C
430 CB  A               RRC     D
432 CB  B               RRC     E
434 CB  C               RRC     H
436 CB  D               RRC     L
                ;
438  F                  RRCA            ;ROTATE ACC RIGHT CIRCULAR
439 ED 67               RRD             ;ROTATE DIGIT RIGHT
                ;
                ;       RESTART INSTRUCTIONS
43B C7                  RST     0
43C D7                  RST     10
43D DF                  RST     18
43E E7                  RST     20
43F EF                  RST     28
440 F7                  RST     30
441 FF                  RST     38
442 CF                  RST     8
                ;
                ;       SUBTRACT WITH CARRY INSTRUCTIONS
443 9E                  SBC     A,(HL)
444 DD 9E  5            SBC     A,(IX+INDEX)
447 FD 9E  5            SBC     A,(IY+INDEX)
44A 9F                  SBC     A,A
44B 98                  SBC     A,B
44C 99                  SBC     A,C
44D 9A                  SBC     A,D
44E 9B                  SBC     A,E
44F 9C                  SBC     A,H
450 9D                  SBC     A,L
451 DE FF               SBC     A,BYT
453 ED 42               SBC     HL,BC
455 ED 52               SBC     HL,DE
457 ED 62               SBC     HL,HL
459 ED 72               SBC     HL,SP
                ;
45B 37                  SCF             ;SET CARRY FLAG
                ;
                ;       SET BIT INSTRUCTIONS
45C CB C6               SET     0,(HL)
45E DD CB               SET     0,(IX+INDEX)
460  5 C6
462 FD CB               SET     0,(IY+INDEX)
464  5 C6
466 CB C7               SET     0,A
468 CB C0               SET     0,B
```

```
46A CB C1               SET     0,C
46C CB C2               SET     0,D
46E CB C3               SET     0,E
470 CB C4               SET     0,H
472 CB C5               SET     0,L
474 CB CE               SET     1,(HL)
476 DD CB               SET     1,(IX+INDEX)
478  5 CE
47A FD CB               SET     1,(IY+INDEX)
47C  5 CE
47E CB CF               SET     1,A
480 CB C8               SET     1,B
482 CB C9               SET     1,C
484 CB CA               SET     1,D
486 CB CB               SET     1,E
488 CB CC               SET     1,H
48A CB CD               SET     1,L
48C CB D6               SET     2,(HL)
48E DD CB               SET     2,(IX+INDEX)
490  5 D6
492 FD CB               SET     2,(IY+INDEX)
494  5 D6
496 CB D7               SET     2,A
498 CB D0               SET     2,B
49A CB D1               SET     2,C
49C CB D2               SET     2,D
49E CB D3               SET     2,E
4A0 CB D4               SET     2,H
4A2 CB D5               SET     2,L
4A4 CB DE               SET     3,(HL)
4A6 DD CB               SET     3,(IX+INDEX)
4A8  5 DE
4AA FD CB               SET     3,(IY+INDEX)
4AC  5 DE
4AE CB DF               SET     3,A
4B0 CB D8               SET     3,B
4B2 CB D9               SET     3,C
4B4 CB DA               SET     3,D
4B6 CB DB               SET     3,E
4B8 CB DC               SET     3,H
4BA CB DD               SET     3,L
4BC CB E6               SET     4,(HL)
4BE DD CB               SET     4,(IX+INDEX)
4C0  5 E6
4C2 FD CB               SET     4,(IY+INDEX)
4C4  5 E6
4C6 CB E7               SET     4,A
4C8 CB E0               SET     4,B
4CA CB E1               SET     4,C
4CC CB E2               SET     4,D
4CE CB E3               SET     4,E
4D0 CB E4               SET     4,H
4D2 CB E5               SET     4,L
4D4 CB EE               SET     5,(HL)
4D6 DD CB               SET     5,(IX+INDEX)
4D8  5 EE
4DA FD CB               SET     5,(IY+INDEX)
4DC  5 EE
```

```
4DE CB EF               SET     5,A
4E0 CB E8               SET     5,B
4E2 CB E9               SET     5,C
4E4 CB EA               SET     5,D
4E6 CB EB               SET     5,E
4E8 CB EC               SET     5,H
4EA CB ED               SET     5,L
4EC CB F6               SET     6,(HL)
4EE DD CB               SET     6,(IX+INDEX)
4F0  5 F6
4F2 FD CB               SET     6,(IY+INDEX)
4F4  5 F6
4F6 CB F7               SET     6,A
4F8 CB F0               SET     6,B
4FA CB F1               SET     6,C
4FC CB F2               SET     6,D
4FE CB F3               SET     6,E
500 CB F4               SET     6,H
502 CB F5               SET     6,L
504 CB FE               SET     7,(HL)
506 DD CB               SET     7,(IX+INDEX)
508  5 FE
50A FD CB               SET     7,(IY+INDEX)
50C  5 FE
50E CB FF               SET     7,A
510 CB F8               SET     7,B
512 CB F9               SET     7,C
514 CB FA               SET     7,D
516 CB FB               SET     7,E
518 CB FC               SET     7,H
51A CB FD               SET     7,L
                ;
                ;       SHIFT LEFT ARITHMETIC INSTRUCTIONS
51C CB 26               SLA     (HL)
51E DD CB               SLA     (IX+INDEX)
520  5 26
522 FD CB               SLA     (IY+INDEX)
524  5 26
526 CB 27               SLA     A
528 CB 20               SLA     B
52A CB 21               SLA     C
52C CB 22               SLA     D
52E CB 23               SLA     E
530 CB 24               SLA     H
532 CB 25               SLA     L
                ;
                ;       SHIFT RIGHT ARITHMENTIC INSTRUCTIONS
534 CB 2E               SRA     (HL)
536 DD CB               SRA     (IX+INDEX)
538  5 2E
53A FD CB               SRA     (IY+INDEX)
53C  5 2E
53E CB 2F               SRA     A
540 CB 28               SRA     B
542 CB 29               SRA     C
544 CB 2A               SRA     D
546 CB 2B               SRA     E
548 CB 2C               SRA     H
```

```
54A CB 2D                SRA     L
                ;
                ;        SHIFT RIGHT LOGICAL INSTRUCTIONS
54C CB 3E                SRL     (HL)
54E DD CB                SRL     (IX+INDEX)
550  5 3E
552 FD CB                SRL     (IY+INDEX)
554  5 3E
556 CB 3F                SRL     A
558 CB 38                SRL     B
55A CB 39                SRL     C
55C CB 3A                SRL     D
55E CB 3B                SRL     E
560 CB 3C                SRL     H
562 CB 3D                SRL     L
                ;
                ;        SUBTRACT WITHOUT CARRY INSTRUCTIONS
564 96                   SUB     (HL)
565 DD 96  5             SUB     (IX+INDEX)
568 FD 96  5             SUB     (IY+INDEX)
56B 97                   SUB     A
56C 90                   SUB     B
56D 91                   SUB     C
56E 92                   SUB     D
56F 93                   SUB     E
570 94                   SUB     H
571 95                   SUB     L
572 D6 FF                SUB     BYT
                ;
                ;        EXCLUSIVE OR INSTRUCTIONS
574 AE                   XOR     (HL)
575 DD AE  5             XOR     (IX+INDEX)
578 FD AE  5             XOR     (IY+INDEX)
57B AF                   XOR     A
57C A8                   XOR     B
57D A9                   XOR     C
57E AA                   XOR     D
57F AB                   XOR     E
580 AC                   XOR     H
581 AD                   XOR     L
582 EE FF                XOR     BYT
                ;
    12 34       NN       .EQU    1234     ;ADDRESS VALUE (16 BITS)
     0  5       INDEX    .EQU    5        ;INDEX VALUE (0 TO 255 DEC)
     0 20       N        .EQU    20       ;EIGHT BIT VALUE (0 TO 255)
    FF FF       BYT      .EQU    -1       ;BYTE VALUE (-128 TO +255 DEC)
                         .END
```

```
FFFF BYT          5 INDEX      15B LABEL1     1C3 LABEL2
  20 N         1234 NN
```

ERRORS: 0

APPENDIX A - RUN-TIME OPTIONS. #14.0.0

```
/B            - OUTPUT BINARY FILE IN BNPF FORMAT.
/E            - INHIBIT ERROR MESSAGES TO CONSOLE.
/H            - INHIBIT HEADINGS AND PAGINATION.
/J            - LIST UNASSEMBLED STATEMENTS AND CONDITIONAL
                ASSEMBLY PSEUDO-OPS.
/K            - EXPAND SYMBOL TABLE STORAGE INTO ADDITIONAL
                CORE.
/L            - OUTPUT LEADER (NULLS) IN BINARY FILE  FOR EACH
                .ORG STATEMENT.
/N            - LIST ONLY THE SYMBOL TABLE.
/O            - OUTPUT LISTING IN OCTAL FORMAT INSTEAD OF IN
                HEXADECIMAL.
/P            - INCLUDE NORMALLY UNLISTED PSEUDO-OPS IN THE
                LISTING.
/S            - OMIT THE SYMBOL TABLE FROM THE LISTING.
/T            - REPLACE THE FORM/FEED WITH 3 CR/LF'S.
/W            - INHIBIT WARNING MESSAGES.
/0 TO /9      - USER FLAGS, USED WITH THE ? OPERATOR.
```

APPENDIX B - INDICATOR SET.

```
*               MULTIPLICATION.
/               DIVISION.
&               BOOLEAN AND.
!               INCLUSIVE OR.
+               ADDITION.
-               SUBTRACTION.
^C              COMPLEMENT INDICATOR, (UPARROW B).
^B              BINARY RADIX INDICATOR, (UPARROW B).
^D              DECIMAL RADIX INDICATOR, (UPARROW D).
^H              HEXADECIMAL RADIX INDICATOR, (UPARROW H).
^O              OCTAL RADIX INDICATOR, (UPARROW O).
^L              LEAST SIGNIFICANT BYTE ACCESS OPERATOR,
                (UPARROW L).
^M              MOST SIGNIFICANT BYTE ACCESS OPERATOR,
                (UPARROW M).
;               COMMENT INDICATOR.
" OR '          ASCII CHARACTER INDICATOR.
?               USER FLAG OPERATOR.
.               CURRENT LOCATION COUNTER, (PERIOD).
```

****

.ADDR	DOUBLE BYTE DATA STORAGE, REVERSED FORMAT.
.BIN	CHANGES DEFAULT RADIX TO BINARY.
.BYTE	SINGLE BYTE DATA STORAGE.
.DBYTE	DOUBLE BYTE DATA STORAGE.
.DECM	CHANGES DEFAULT RADIX TO DECIMAL.
.DINST	RENAMES A MICROPROCESOR INSTRUCTION.
.END	PROGRAM TERMINATOR.
.ENDC	ENDS CONDITIONAL ASSEMBLY.
.EQU	ASSIGNS A PERMANENT VALUE TO A SYMBOL.
.HEX	CHANGES DEFAULT RADIX TO HEXADECIMAL.
.IFDEF	INCLUDE CODE TO .ENDC IF SYMBOL IS DEFINED.
.IFNDEF	INDLUDE CODE TO .ENDC IF SYMBOL IS NOT DEFINED.
.IFNZRO	INCLUDE CODE TO .ENDC IF OPERAND DOES NOT EQUAL 0.
.IFZERO	INCLUDE CODE TO .ENDC IF OPERAND EQUALS 0.
.LIST	PROVIDES SELECTIVE LISTINGS.
.OCT	CHANGES DEFAULT RADIX TO OCTAL.
.ORG	REASSIGNS THE CURRENT LOCATION COUNTER.
.PAGE	BEGINS NEW PAGE IN LISTING.
.SET	ASSIGNS A TEMPORARY VALUE TO A SYMBOL.
.TITLE	SPECIFIES HEADING.
.ZERO	ZEROS A SPECIFED NUMBER OF BYTES.

****

APPENDIX D - ERROR MESSAGES. #14.0.0

****

E:BN - BAD NESTING OF BRACKETS.
E:DF - OUTPUT FILE DEVICE FULL. (FATAL)
E:DR - DIGIT OUTSIDE OF RADIX.
E:IL - ILLEGAL LABEL FIELD.
E:IO - ILLEGAL OPERAND VALUE.
E:JR - RELATIVE JUMP ADDRESS OUT OF RANGE.
E:LO - LINE INPUT OVERFLOW.
E:LS - LOCAL SYMBOL SYNTAX ERROR.
E:LT - LOCAL SYMBOL TABLE OVERFLOW. (FATAL)
E:ML - MULTIPLE LABEL DEFINITION.
E:MO - MISSING OR ILLEGAL MNEMONIC IN OPERATOR FIELD.
E:OC - OPERAND TOO COMPLEX.
E:OE - OPEN ERROR IN OUTPUT FILE. (FATAL)
E:OM - OPERAND MISSING.
E:OS - OPERAND SYNTAX ERROR.
E:PE - PHASE ERROR, ADDRESS CONFLICT. (FATAL)
E:PS - ILLEGAL PERMANENT SYMBOL USAGE IN OPERAND.
E:RE - INPUT FILE READ ERROR. (FATAL)
E:RV - BAD REGISTER VALUE FIELD.
E:ST - SYMBOL TABLE OVERFLOW. (FATAL)
E:TL - LABEL DEFINED TOO LATE.
E:US - UNDEFINED SYMBOL.
E:WE - OUTPUT FILE WRITE ERROR. (FATAL)

W:EF - NO .END STATEMENT IN LAST FILE.
W:UC - UNINHIBITED CONDITIONAL ASSEMBLY IN EFFECT
AT ASSEMBLY END.

****