8008 CROSS-ASSEMBLER

FOR USE ON DIGITAL EQUIPMENT CORPORATION'S
RESOURCE TIME-SHARING SYSTEM, RSTS.

INSTRUCTION SET: (SAME AS FOR INTEL'S INTELLEC-8 ASSEMBLER)

HLT,NOP,RLC,RRC,RAL,RAR,RET,RC,RNC,RZ,RNZ,RM,RP,RPE,RPO,RST,IN,OUT
,MOV,
ADD,ADC,SUB,SBB,ANA,XRA,ORA,CMP,INR,DCR,MVI,ADI,ACI,SUI,SBI,ANI,XR
I,
ORI,CPI,JMP,JC,JNC,JZ,JNZ,JP,JM,JPE,JPO,CALL,CC,CNC,CZ,CNZ,CP,CM,C
PE,CPO

PSEUDO-INSTRUCTIONS:

LXI	FORMAT:	LXI REG,EXP
	FUNCTION:	ASSEMBLES AS 2 MVI INSTRUCTIONS TO LOAD THE HEX VALUE OF "EXP", LOW-ORDER BYTE FIRST, INTO 2 CONSECUTIVE REGISTERS BEGINNING WITH "REG".
WORD	FORMAT:	WORD EXP
	FUNCTION:	ASSEMBLE 2 BYTES ACCORDING TO THE HEX VALUE OF "EXP", LOW-ORDER BYTE FIRST.
BYTE	FORMAT:	BYTE EXP
	FUNCTION:	ASSEMBLE 1 BYTE ACCORDING TO THE HEX VALUE OF "EXP" ("EXP"<100 HEX).
ASCII	FORMAT:	ASCII "STRING"
	FUNCTION:	ASSEMBLE THE ASCII VALUE OF EACH CHARACTER IN THE "STRING". DELIMITERS OTHER THAN QUOTES MAY BE USED, BUT THE CHARACTER USED AS A DELIMITER MAY

```
NOT
                                    APPEAR IMBEDDED IN THE STRING.

ORG             FORMAT:             ORG     EXP
                FUNCTION:           SET THE LOCATION COUNTER TO THE VA
LUE
                                    OF "EXP".

END             FORMAT:             END
                FUNCTION:           END CURRENT ASSEMBLY PASS. "END" M
UST
                                    APPEAR AS THE LAST LINE OF ANY PRO
GRAM.

SET             FORMAT:             SET     SYMBOL=EXP
                FUNCTION:           SET THE VALUE OF "SYMBOL" TO THE V
ALUE
                                    OF "EXP". NO SPACES SHOULD APPEAR
                                    BEFORE OR AFTER THE = SIGN.


EXPRESSIONS - "EXP" MAY BE ANY ONE OF THE FOLLOWING:
        1, A SYMBOL OF UP TO 6 ALPHANUMERIC CHARACTERS WHOSE VALUE
 IS
           DEFINED BY A STATEMENT LABEL OR SET PSEUDO-INSTRUCTION.
        2, A POSITIVE INTEGER HEXADECIMAL NUMBER PRECEDED BY A # S
IGN.
        3. A POSITIVE INTEGER DECIMAL NUMBER PRECEDED BY A % SIGN.
        4. A DOT (.) WHICH IS THE SYMBOL FOR THE CURRENT LOCATION.
        5. ANY OF THE ABOVE PRECEDED BY A UNARY MINUS SIGN.
        6. A STRING OF ANY OF THE ABOVE SEPARATED BY + AND - SIGNS
           WITH NO IMBEDDED SPACES, TABS, BRACKETS OR OTHER SEPARA
TING
           CHARACTERS.


LINE SYNTAX - A LINE OF SOURCE CODING MAY CONSIST OF ANY OF 4 COMP
ONENTS
        ARRANGED AS FOLLOWS:

        LABEL:  MNEMONIC        OPERAND         ; COMMENT

THE "LABEL" HAS THE EFFECT OF ASSIGNING THE CURRENT VALUE OF THE
LOCATION COUNTER TO THE SYMBOL "LABEL". SPACES AND TABS MAY BE USE
D
FREELY BETWEEN COMPONENTS OF A LINE, BUT MAY BE IMBEDDED ONLY IN
THE COMMENT. TWO-REGISTER OPERANDS SUCH AS A,M OR REGISTER,EXPRESS
ION
```

OPERANDS SUCH AS H,EXP MUST CONTAIN NO IMBEDDED SPACES OR TABS.

OPERATING INSTRUCTIONS:

THE ASSEMBLER REQUESTS DEVICE:FILE NAMES FOR THE SOURCE PROGRAM, HEX
OUTPUT AND ASSEMBLY LISTING OUTPUT. THE HEX OUTPUT IS A FILE IN
INTELLEC-8 HEXADECIMAL TAPE FORMAT AND MAY BE OUTPUT TO PAPER TAPE
FOR SUBSEQUENT LOADING USING AN APPROPRIATE READ-IN ROUTINE.
TWO PASSES ARE MADE OF THE SOURCE PROGRAM, THE FIRST TO EVALUATE SYMBOLS
AND THE SECOND TO PERFORM ASSEMBLY AND PRODUCE THE LISTING.

ERRORS:

ERRORS IN THE SOURCE PROGRAM ARE LISTED ON THE USERS'S TERMINAL BY
PAGE AND LINE NUMBER AND ARE ALSO FLAGGED ON THE ASSEMBLY LISTING
USING THE FOLLOWING ERROR CODES:

A ARGUMENT OUTSIDE RANGE ALLOWED FOR THE INSTRUCTION BEING USED.
I UNRECOGNIZED INSTRUCTION.
N NOT A VALID HEXADECIMAL OR DECIMAL NUMBER.
R INVALID REGISTER OPERATION OR REGISTER NOT SPECIFIED CORRECTLY.
S STRING DELIMITER MISSING.
U UNDEFINED SYMBOL OR SYMBOL LONGER THAN 6 CHARACTERS. A *U*
     ERROR IN AN ORG INSTRUCTION ON THE FIRST PASS BUT NOT THE SECOND
     IS AN INDICATION OF A PHASE ERROR BETWEEN FIRST AND SECOND PASS.

NOTE: PAGE 0 ERRORS ARE ERRORS ENCOUNTERED ON THE FIRST PASS.
      A "SYMBOL TABLE OVERFLOW" MESSAGE INDICATES THAT THE NUMBER
      OF SYMBOLS DEFINED IN THE SOURCE PROGRAM HAS EXCEEDED THE
      DIMENSIONS OF THE S% AND S$ ARRAYS IN THE ASSEMBLER.

SAMPLE ASSEMBLY:

```
 1                        ; PARTIAL ASSEMBLY OF INTELLEC 8 MONITOR
 2                                ORG   #3B00         ; LOCATE IN TOP 5 ROMS IN 16K
 3
 4    3B00    2E3B3614            LXI   H,VERS
 5    3B04    1609                MVI   C,LVER
```

```
 6    3B06    CF            VER0:   MOV   B,M
 7    3B07    46CB3C                CALL  INCHL
 8    3B0A    46473C                CALL  TO
 9    3B0D    11                    DCR   C
10    3B0E    48063B                JNZ   VER0
11    3B11    44DE3B                JMP   START         ; GO TO EN
TRY POINT
12
13    3B14    0D            VERS:   BYTE  CR
14    3B15    0A                    BYTE  LF
15    3B16    56455220
16            312E30                ASCII 'VER 1.0'
17            0009                  SET   LVER=.-VERS
```

CONSTRUCTION OF THE CROSS-ASSEMBLER:

LINES	
10-50	INITIALISATION
80-	START OF FIRST PASS (EVALUATION OF SYMBOLS)
100-170	BRANCH ACCORDING TO INSTRUCTION TYPE
190-200	START OF SECOND PASS
210-260	ASSEMBLY OF FIRST BYTE OF AN INSTRUCTION
270	ASSEMBLY OF SECOND OR THIRD BYTES IF NEEDED
280	FINISH OF SECOND PASS, PRINT-OUT OF LINE TO LISTIN G
310-370	BRANCH ACCORDING TO INSTRUCTION TYPE (SECOND PASS)
400-440	GENERATION AND PRINT-OUT OF ALPHABETICAL SYMBOL TA BLE
1000-1050	SUBROUTINE TO ANALYSE INPUT LINE
1100-1120	PAGE AND LINE FORMAT SUBROUTINE FOR LISTING
1500	PRELIMINARY ARGUMENT EVALUATION FOR SET INSTRUCTIO N
2000-2120	SUBROUTINE TO EVALUATE AN EXPRESSION
3000	SUBROUTINE TO RETURN ASCII STRING FOR ASCII INSTRU CTION
4000	SUBROUTINE TO OUTPUT A WORD AS TWO BYTES, LOW BYTE FIRST
5000-5060	SUBROUTINE TO OUTPUT A BYTE
5500	*A* ERROR SUBROUTINE
6000	PRELIMINARY UPDATE OF CHECKSUM AND BYTE COUNT FOR 6005
6005-6010	SUBROUTINE TO CONVERT AN INTEGER TO A 2 BYTE HEX S TRING
7000	SUBROUTINE TO LEFT JUSTIFY THE STRING L$
8000-8010	SUBROUTINE TO EVALUATE A REGISTER NUMBER
9000-9020	ERROR HANDLING ROUTINE
9500-9511	DATA FOR LOOK-UP TABLES

LISTING OF CROSS-ASSEMBLER:

```
1 ! 8008 CROSS ASSEMBLER. WRITTEN BY R. TAPP, APRIL 1975.
10 INPUT "SOURCE FILE"; I$ : OPEN I$ FOR INPUT AS FILE 1% :
        INPUT "HEX FILE"; O$ : OPEN O$ FOR OUTPUT AS FILE 2% :
        INPUT "LISTING FILE"; O$ : OPEN O$ FOR OUTPUT AS FILE 3%
20 O$=SYS(CHR$(6%)+CHR$(-7%)) : ON ERROR GOTO 9000 :
        PRINT #2%, CHR$(0%); FOR I%=1% TO 50%
30 DIM S%(200),M$(63),Q$(32),S$(200)
40 B$="SYMBOL" : M$="CODES" : R$="ABCDEHLM" : S$=B$+":" : X$="XX"
:
        Y$=SPACE$(43%)
45 DEF FNL$=S$+" "+M$+" "+LEFT(L$,INSTR(N%+2%,L$," ")-1%) :
        DEF FNP$=" PAGE"+NUM$(P%)+"LINE"+NUM$(Q%)+">"+FNL$+" "+C$
50 READ M$(I%) FOR I%=0% TO 62% :
        FOR I%=0% TO 32% : READ A% : Q$(I%)=CHR$(A%) : NEXT I%
80 GOSUB 1000 : IF S%<>0% THEN S$(K%)=LEFT(S$,S%-1%) :
        S%(K%)=L% : K%=K%+1%
100 L%=L%+N% : ON C% GOTO 80,110,120,130,140,150,160,170,80
110 L%=L%+2%
120 L%=L%+1%
130 L%=L%+1% : GOTO 80
140 GOSUB 3000 : L%=L%+N% : GOTO 80
150 GOSUB 2000 : L%=A% : GOTO 80
160 CLOSE 1% : OPEN I$ FOR INPUT AS FILE 1% : L%=0% : GOTO 190
170 GOSUB 1500 : S$(K%)=LEFT(L$,M%-1%) : S%(K%)=A% : K%=K%+1% : GO
TO 80
190 GOSUB 1100 : GOSUB 1000 : IF C%>5% THEN PRINT #3%, SPACE$(8%);
        ELSE A%=L% : GOSUB 8100 : PRINT #3%, "    ";
200 ON C% GOTO 210,310,320,330,340,350,360,370,280
210 IF M%>14% THEN 220 ELSE A%=ASCII(Q$(M%)) : GOTO 270
220 IF M%>17% THEN 230 ELSE GOSUB 2000 : IF M%=15% THEN
        IF A%<0% OR A%>7% THEN GOSUB 5500 ELSE A%=A%*8%+5%
        ELSE IF A%<0% OR A%>31% THEN GOSUB 5500 ELSE A%=A%+A%+65%
225 GOTO 270
230 IF M%>26% THEN 250 ELSE IF M%>18% THEN R%=(M%-19%)*8%+128%
        ELSE GOSUB 8000 : R%=A%*8%+192%
240 GOSUB 8000 : A%=R%+A% : IF A%=255% THEN GOSUB 8010
245 GOTO 270
250 IF M%>28% THEN 260 ELSE GOSUB 8000 :
        IF A%=0% OR A%=7% THEN GOSUB 8010
255 A%=A%*8%+M%-27% : GOTO 270
260 IF M%>37% THEN A%=ASCII(Q$(M%-23%)) ELSE IF M%>29% THEN
        A%=(M%-30%)*8%+4% ELSE GOSUB 8000 : A%=A%*8%+6%
270 GOSUB 5000 : IF N%=1% THEN 280 ELSE GOSUB 2000 :
        IF N%=2% THEN GOSUB 5000 ELSE GOSUB 4000
```

```
280 PRINT #3%,, : PRINT #3%, FNL$, UNLESS ASCII(M$)<33% :
        PRINT #3%, C$ : GOTO 190
310 GOSUB 8000 : R%=A%-1% : GOSUB 2000 : CHANGE CVT%$(A%) TO A% :
        FOR M%=1% TO 2% : A%=(R%+M%)*8%+6% : GOSUB 5000 :
        A%=A%(M%) : GOSUB 5000 : NEXT M% : GOTO 280
320 GOSUB 2000 : GOSUB 4000 : GOTO 280
330 GOSUB 2000 : GOSUB 5000 : GOTO 280
340 GOSUB 3000 : C%=0% : FOR M%=1% TO N% : IF C%=4% THEN C%=0% :
        PRINT #3% : GOSUB 1100 : PRINT #3%, SPACE$(8%);
345 A%=ASCII(RIGHT(A$,M%)) : GOSUB 5000 : C%=C%+1% : NEXT M% : GOT
O 280
350 GOSUB 2000 : L%=A% : GOSUB 5050 UNLESS X%=0% : GOTO 280
360 GOSUB 5050 UNLESS X%=0% : PRINT #2%, ":0000000000" : GOTO 280
370 GOSUB 1500 : GOSUB 8100 : GOTO 280
400 FOR I%=0% TO K%-2% : FOR J%=I%+1% TO K%-1% : FOR L%=1% TO 6% :
        C%=ASCII(RIGHT(S$(J%),L%)) : S%=ASCII(RIGHT(S$(I%),L%)) :
        IF C%<S% THEN 420 ELSE IF C%>S% THEN 430
410 NEXT L%
420 S$=S$(J%)+"" : A%=S%(J%) : S$(J%)=S$(I%)+"" : S%(J%)=S%(I%) :
        S$(I%)=S$+"" : S%(I%)=A%
430 NEXT J% : NEXT I% : C%=0% : GOSUB 1100 : FOR M%=0% TO K%-1% :
        IF C%=4% THEN C%=0% : PRINT #3% : GOSUB 1100
440 A%=S%(M%) : R%=A% : PRINT #3%,, : PRINT #3% USING "\    \\   \
",
        S$(M%)," = #"; : GOSUB 8100 : PRINT #3%, " = %"R%; :
        C%=C%+1% : NEXT M% : PRINT #3% : GOTO 9010
1000 INPUT LINE #1%, L$ : I%=INSTR(1%,L$,CHR$(13%)) :
        J%=INSTR(1%,L$,CHR$(10%)) :
        IF I%>J% THEN I%=J% ELSE IF I%=0% THEN I%=LEN(L$)
1005 C%=INSTR(1%,L$,";") : IF C%=0% THEN C%=I%
1010 C$=MID(L$,C%,I%-C%) : LSET L$=LEFT(L$,C%-1%) :
        S%=INSTR(1%,L$,":") : LSET S$=LEFT(L$,S%) :
        LSET L$=RIGHT(L$,S%+1%)
1020 GOSUB 7000 : I%=INSTR(1%,L$,CHR$(9%))-1% :
        J%=INSTR(1%,L$," ")-1% : IF I%<J% AND I%>=0% THEN J%=I%
1025 LSET M$=LEFT(L$,J%) : LSET L$=RIGHT(L$,J%+2%)
1030 GOSUB 7000 : I%=INSTR(2%,L$,CHR$(9%)) :
        LSET L$=LEFT(L$,I%-1%) UNLESS I%=0% : A$=L$+"" :
        IF ASCII(M$)<33% THEN N%=0% : C%=9% : RETURN
1040 C%=1% : M%=0% : M%=M%+1% UNTIL LEFT(M$,J%)=M$(M%) OR M%=63% :
        IF M%<29% THEN N%=1% ELSE IF M%<38% THEN N%=2%
        ELSE IF M%<56% THEN N%=3% ELSE N%=0% : C%=M%-54% :
        IF C%=9% THEN PRINT "*I*"FNP$ : PRINT #3%, "*I*";
1050 RETURN
1100 IF Q%=0% THEN P%=P%+1% : PRINT #3%, CHR$(12%);CHR$(10%)
        "PAGE"P%,I$ : PRINT #3%
1110 Q%=Q%+1% : PRINT #3% USING "##    ",Q%; : IF Q%=50% THEN Q%=0
%
```

```
1120 RETURN
1500 M%=INSTR(1%,A$,"=") : LSET A$=RIGHT(A$,M%+1%)
2000 A%=0% : C%=ASCII(A$) : IF C%=43% OR C%=45% THEN
        LSET A$=RIGHT(A$,2%) ELSE C%=43%
2010 IF C%=45% THEN S%=1% ELSE IF C%=43% THEN S%=0% ELSE RETURN
2020 C%=48% : C%=ASCII(RIGHT(A$,I%)) FOR I%=2% UNTIL C%<47% : I%=I
%-2% :
        LSET B$=LEFT(A$,I%) : LSET A$=RIGHT(A$,I%+2%)
2030 J%=ASCII(B$) : IF J%=35% THEN 2050 ELSE IF J%=37% THEN 2080
        ELSE IF J%=46% THEN 2090 ELSE J%=0% :
        J%=J%+1% UNTIL S$(J%)=LEFT(B$,I%) OR J%=K% :
        IF J%<K% THEN J%=S%(J%) ELSE PRINT "*U*"FNP$ :
        PRINT #3%, "*U*";
2040 GOTO 2100
2050 RSET B$=MID(B$,2%,I%-1%) : CHANGE B$ TO A% : FOR I%=1% TO 2%
:
        IF A%(I%)<>32% THEN 2120
2060 NEXT I% : FOR I%=3% TO 6% : IF A%(I%)>70% THEN 2120
        ELSE IF A%(I%)>64% THEN A%(I%)=A%(I%)-55%
        ELSE IF A%(I%)>57% THEN 2120
        ELSE IF A%(I%)>47% THEN A%(I%)=A%(I%)-48%
        ELSE IF A%(I%)=32% THEN A%(I%)=0% ELSE 2120
2070 NEXT I% : A%(4%)=A%(3%)*16%+A%(4%) : A%(6%)=A%(5%)*16%+A%(6%)
 :
        J%=CVT$%(CHR$(A%(4%))+CHR$(A%(6%))) : GOTO 2100
2080 J%=VAL(MID(B$,2%,I%-1%)) : GOTO 2100
2090 IF N%<2% THEN J%=L% ELSE J%=L%-1%
2100 IF S% THEN A%=A%-J% ELSE A%=A%+J%
2110 GOTO 2010
2120 A%=0% : PRINT "*N*"FNP$ : PRINT #3%, "*N*"; : RETURN
3000 N%=INSTR(2%,A$,CHR$(ASCII(A$)))-2% : LSET A$=RIGHT(A$,2%) :
        RETURN UNLESS N%=-2% : N%=6% :
        PRINT "*S*"FNP$ : PRINT #3%, "*S*"; : RETURN
4000 CHANGE CVT%$(A%) TO A% : A%=A%(2%) : GOSUB 5000 : A%=A%(1%)
5000 IF X%=0% THEN X0%=L% : Y%=0% : Z%=0%
5005 IF A%<-128% OR A%>255% THEN GOSUB 5500 ELSE IF A%<0% THEN
        A%=ASCII(RIGHT(CVT%$(A%),2%))
5010 GOSUB 6000 : LSET Y$=LEFT(Y$,Y%)+X$ : PRINT #3%, X$; :
        Y%=Y%+2% : L%=L%+1% : CHANGE CVT%$(L%) TO B% :
        IF B%(2%)-B%(2%)/16%*16% THEN RETURN
5050 A%=X% : GOSUB 6000 : LSET B$=X$ : CHANGE CVT%$(X0%) TO B% :
        FOR J%=1% TO 2% : A%=B%(J%) : GOSUB 6000 : A%=J%+J% :
        LSET B$=LEFT(B$,A%)+X$ : NEXT J%
5060 A%=ASCII(RIGHT(CVT%$(Z%),2%)) : GOSUB 6000 :
        LSET Y$=":"+B$+"00"+LEFT(Y$,Y%)+X$ : PRINT #2%, Y$ :
        X%=0% : RETURN
5500 A%=0% : PRINT "*A*"FNP$ : PRINT #3%, "*A*"; : RETURN
6000 Z%=Z%-A% : X%=X%+1%
```

```
6005 A%(7%)=A%/16% : A%(8%)=A%-16%*A%(7%) : FOR I%=7% TO 8% :
        IF A%(I%)>9% THEN A%(I%)=A%(I%)+55% ELSE A%(I%)=A%(I%)+48%
6010 NEXT I% : LSET X$=CHR$(A%(7%))+CHR$(A%(8%)) : RETURN
7000 LSET L$=RIGHT(L$,2%) FOR I%=1% WHILE ASCII(L$)<33% AND I%<10%
 :
        RETURN
8000 A%=INSTR(1%,R$,LEFT(A$,1%))-1% : IF A%=-1% THEN 8010
        ELSE LSET A$=RIGHT(A$,3%) : RETURN
8010 PRINT "*R*"FNP$ : PRINT #3%, "*R*"; : RETURN
8100 CHANGE CVT%$(A%) TO A% : FOR J%=1% TO 2% : A%=A%(J%) : GOSUB
6005 :
        PRINT #3%, X$; : NEXT J% : RETURN
9000 IF ERL=2080% THEN RESUME 2120 ELSE Q%=0% :
        IF ERR=11% THEN RESUME 400
9010 PRINT #2%, CHR$(0%); FOR I%=1% TO 50% :
        PRINT #I%, CHR$(26%); FOR I%=2% TO 3% : CLOSE 1%,2%,3%
9020 IF ERL=80% OR ERL=170% THEN PRINT "SYMBOL TABLE OVERFLOW"
        ELSE IF ERR=28% THEN RESUME 9999 ELSE ON ERROR GOTO 0
9500 DATA HLT,NOP,RLC,RRC,RAL,RAR,RET,RC,RNC,RZ,RNZ,RM,RP,RPE,RPO,
        RST,IN,OUT,MOV,ADD,ADC,SUB,SBB,ANA,XRA,ORA,CMP,INR,DCR
9501 DATA MVI,ADI,ACI,SUI,SBI,ANI,XRI,ORI,CPI,JMP,JC,JNC,JZ,JNZ,
        JP,JM,JPE,JPO,CALL,CC,CNC,CZ,CNZ,CP,CM,CPE,CPO,
        LXI,WORD,BYTE,ASCII,ORG,END,SET
9510 DATA 0,192,2,10,18,26,7,35,3,43,11,51,19,59,27
9511 DATA 68,96,64,104,72,80,112,120,88,70,98,66,106,74,82,114,122
,90
9999 END
```