MITS

"Creative Electronics"

SOFTWARE AGREEMENT

This software is copywrited and the property of MITS, Inc., 6328 Linn Avenue, N.E., Albuquerque, New Mexico, and has been supplied by MITS to you. This software is furnished subject to the following restrictions: it shall not be reproduced or copied without express written permission of MITS, Inc.

To do any of the above without approval by MITS, Inc. will make you liable and open for MITS, Inc. to take legal action against you.

This agreement shall be considered accepted and binding upon your receipt of this and any software.

MITS PROGRAMMING SYSTEM 1
(REV 2.2)

TABLE OF CONTENTS

I. GLOSSARY

II. SYSTEM MONITOR

- MONITOR COMMANDS
- PROGRAM MONITOR CALLS

III. ASSEMBLER OPERATIONS

- INTRODUCTION
- OPTIONS

IV. ASSEMBLY PROGRAMMING

- CONSTANTS
- STATEMENT STRUCTURE
- STATEMENT OPTIONS
- PROGRAMMING TRICKS
- EXAMPLE PROGRAM

V. EDITOR

APPENDIX:

- A. ABSOLUTE LOAD TAPE FORMAT
- B. ASSEMBLY MEMORY MAP
- C. ERROR CODES
- D. I/O PORT ASSIGNMENTS
- E. LOADING THE MONITOR
- F. MISCELANEOUS

I. GLOSSARY

MACHINE INSTRUCTION - BINARY BYTE(S) THAT EXECUTE TO PERFORM A DEFINED COMPUTER FUNCTION.

ASSEMBLY (SOURCE) CODE - SYMBOLIC LABELS, OPCODES, AND OPERANDS THAT ARE ORDERED IN SUCCESSION TO DEFINE A LOGICAL PROCEDURE WHICH CAN BE ASSEMBLED TO PRODUCE EXECUTABLE MACHINE INSTRUCTIONS.

OPCODES - DEFINED SYMBOLS THAT ASSEMBLE DIRECTLY AS 1 TO 3 BYTES OF MACHINE INSTRUCTION. THE SYMBOLS ARE MEANINGFUL DESCRIPTORS OF THE MACHINE FUNCTION TO BE PERFORMED DURING PROGRAM EXECUTION.

LABEL - A USER DEFINED SYMBOL THAT CORRESPONDS TO THE STORAGE ADDRESS OF THE FOLLOWING OPCODE. LABELS ARE USED TO DEFINE POINTS FOR TRANSFER OF PROGRAM EXECUTION WHICH NORMALLY PROCEEDS IN A SEQUENTIAL MANNER.

OPERANDS - SYMBOLIC REFERENCES TO REGISTERS, LABELS OR CONSTANTS THAT ARE USED TO COMPLETELY DEFINE THE FUNCTION SPECIFIED BY THE OPCODE. NONE, ONE, OR TWO OPERANDS ARE TYPICAL FOR MOST OPCODES.

PSEUDO-OPCODES - DEFINED SYMBOLS THAT DIRECT THE ASSEMBLY OF THE SOURCE CODE. THEY CAN ALLOCATE MEMORY (PROGRAM VARIABLE AND DATA STORAGE), DEFINE CONSTANTS OR AFFECT CONTROL OF THE ASSEMBLY PROCEDURE.

EXECUTION STORAGE - THE PHYSICAL MEMORY SPACE WHERE AN ASSEMBLED PROGRAM CAN EXECUTE. ABSOLUTE ASSEMBLY GENERATES MACHINE INSTRUCTIONS THAT WILL ONLY EXECUTE CORRECTLY IN MEMORY SPACE THAT WAS DEFINED DURING ASSEMBLY.

PROGRAM STORAGE - THE PHYSICAL MEMORY WHERE PROGRAM MACHINE CODE IS STORED. THE PROGRAM WILL NOT NECESSARILY EXECUTE CORRECTLY AT THIS LOCATION UNLESS ASSEMBLY OR RELOCATABLE LOADING DEFINED THE PROGRAM STORAGE TO BE THE SAME AS EXECUTION STORAGE.

DEBUG - THE PROCESS OF TESTING A PROGRAM TO REMOVE LOGIC ERRORS(BUGS) BY ANALYZING ITS EXECUTION(PERFORMANCE).

PATCH - A PROGRAM CHANGE BY INSERTION OF OVERLAYS TO FIX ERRORS WHILE DEBUGGING A PROGRAM.

II. SYSTEM MONITOR

THE SYSTEM MONITOR HAS 6 BASIC FUNCTIONS. USING FEATURES OF THE MONITOR DEVICE INDEPENDENT I/O CAN BE DONE, PROGRAMS IN ABSOLUTE FORM CAN BE LOADED OR DELETED, AND ABSOLUTE OR SOURCE FILES CAN BE SEARCHED FOR ON ANY DEVICE. ALL MONITOR COMMANDS ARE ARE READ ON SYMBOLIC DEVICE CTB.

MONITOR COMMAND FORMAT

1. EXECUTE A PROGRAM THAT HAS BEEN READ IN

   THE MONITOR SIGNALS THAT IT IS READY FOR A COMMAND BY PRINTING 2 BLANKS AND A QUESTION MARK

   ```
   ?EDT
   ```

   AFTER THE MACHINE PRINTS THE QUESTION MARK, THE THREE CHARACTER NAME OF THE PROGRAM TO BE EXECUTED IS TYPED. THE EXAMPLE SHOWN IF FOLLOWED BY A CARRIGE RETURN WOULD HAVE CAUSED THE EXECUTION OF EDT IF IT WAS IN MEMORY.

2. EXECUTION TIME OPTIONS

   USER PROGRAMS CAN BE PASSED EXECUTION OPTIONS BY THE MONITOR IF ENCLOSED IN PARENTHESIS.

   ```
   ?EDT(R)
   ```

   WHEN THE PROGRAM REQUESTED IS BRANCHED TO THE D&E REGISTER PAIR CONTAIN THE ADDRESS OF THE FIRST CHARACTER FOLLOWING THE MONITOR COMMAND. IN THE ABOVE EXAMPLE D&E WOULD POINT AT THE OPEN PARENTHESIS. THE B&C REGISTER PAIR CONTAINS THE ADDRESS OF THE MONITOR STATUS WORD, WHICH CONTAINS THE LENGTH OF THE COMMAND LINE. IN THE ABOVE EXAMPLE THE STATUS WORD CONTAINS THE NUMBER 6.

3. LOADING ABSOLUTE PROGRAM TAPES

   IF A PROGRAM NEEDS TO BE LOADED INTO MEMORY AND EXECUTED, A SYMBOLIC DEVICE NAME MUST BE SPECIFIED IN THE COMMAND.

   ```
   ?EDT,CTR(R)
   ```

   THIS COMMAND SPECIFIES THAT THE PROGRAM NAMED EDT IS TO BE LOADED FROM THE PHYSICAL DEVICE THAT CTR IS OPEN TO. THE EXECUTION OPTIONS CAN STILL BE ADDED TO THE END OF A COMMAND AS SHOWN IN THE EXAMPLE. WHEN PROGRAMS ARE LOADED IN THIS MANNER THE NAME AND START ADDRESS ARE SAVED IN THE PROGRAM TABLE(PTL).

4. FILE SEARCHING

   IF INSTEAD OF FOLLOWING A LOAD COMMAND WITH POSSIBLE EXECUTION OPTIONS THE SYMBOLIC DEVICE NAME IS FOLLOWED BY A COMMA AND A TYPE DESIGNATOR

   ```
   S - ASCII FILE
   A - ABSOLUTE FILE
   ```

   THE SPECIFIED FILE WOULD BE SEARCHED FOR ON THE PHYSICAL DEVICE THAT THE SYMBOLIC DEVICE NAME IS OPEN TO. AFTER FINDING THE FILE IT IS SKIPPED AND CONTROL IS RETURNED TO THE MONITOR. THE MAIN USE OF THIS COMMAND IS TO SEARCH FOR THE LAST FILE ON AN AUDIO CASSETTE IN ORDER TO WRITE OUT A NEW ONE. ALL FILES ON THE THE CASSETTE SHOULD BE OF THE SAME TYPE AS THE FILE YOU ARE SEARCHING FOR.

   ```
   ?EDT,CTB,A
   ```

   THIS EXAMPLE WOULD SEARCH FOR THE ABSOLUTE FILE "EDT" ON THE PHYSICAL DEVICE "CTB" IS OPENED TO. AFTER FINDING THE FILE, IT WOULD BE SKIPPED AND THEN CONTROL RETURNED TO THE MONITOR.

UTILITY PROGRAMS

THE MONITOR INCLUDES 3 UTILITY PROGRAMS:
CLR-DELETES A PROGRAM NAME FROM THE PTL.
OPN-OPENS A SYMBOLIC DEVICE NAME TO A PHYSICAL DEVICE.
CLS-CLOSES OR DISCONNECTS A SYMBOLIC NAME FROM A HARDWARE DEVICE.

1. OPN

THIS PROGRAM IS USED TO ASSIGN DIFFERENT
PHYSICAL DEVICES TO A SYMBOLIC NAME
MAKING PROGRAMS DEVICE INDEPENDENT.
?OPN CTB,TY
THIS COMMAND OPENS THE SYMBOLIC NAME
"CTB" TO THE TELETYPE. IN THIS TYPE OF
OPN THE MODE DEFAULT WOULD BE ASSUMED.
THESE ARE:
NO ECHO - DON'T ECHO INPUT
ABSOLUTE - ALL 8 BITS OF EVERY
READ ARE TRANSMITTED.
TABS ARE WRITTEN UNCHANGED
AND ARE ECHOED AS 1 SPACE.
THE OPTIONS ARE:
E - ECHO ALL INPUT ON TTY
A - ASCII READ, ONLY 7 LEAST
SIGNIFIGANT BITS TRANSMITTED
AND LINE FORMAT RECOGNIZED.
SEE APPENDEX F.
T - TAB CONTROL. SPACES ARE
PRINTED TO FORCE CURSOR INTO A
COLUMN THAT IS AN EVEN MULTIPLE
OF 8 FROM THE LEFT MARGIN.
THE TAB CHARACTER IS CONTROL I(11Q).
THE OPTIONS ARE SPECIFIED AT THE END
OF THE COMMAND SEPERATED BY COMMAS.
THE ORDER THAT THEY OCCUR IS
IRRELEVENT.
IF THE SYMBOLIC NAME IS ALREADY OPEN WHEN
ANOTHER COMMAND TO OPEN IT IS GIVEN, THE
SYMBOLIC NAME IS REOPENED ACCORDING TO THE
NEW COMMAND.

2. CLS

CLOSING A SYMBOLIC NAME REMOVES
THE NAME FROM THE I/O TABLE MAKING
THE NAME UNAVAILABLE UNTIL IT IS
REOPENED.
?CLS CTB
IN THIS EXAMPLE "CTB" IS NOW CLOSED
MEANING THAT NO PROGRAM SHOULD BE
EXECUTED THAT READS OR WRITES ON
"CTB". IF AN I/O OPERATION IS ATTEMPTED
THE PROGRAM WILL ABORT, CAUSING THE
MONITOR TO PRINT AN ERROR MESSAGE.

3. CLR

THIS COMMAND DELETES A PROGRAM NAME
FROM THE PTL.
?CLR EDT
AFTER GIVING THE ABOVE COMMAND
?EDT
IS ILLEGAL.
THE PROGRAM MUST BE RELOADED IN ORDER
TO BE EXECUTED.

PROGRAM MONITOR CALLS

THE FOLLOWING SECTION DECRIBES HOW A USER WRITTEN PROGRAM CAN USE FEATURES OF THE MONITOR TO FREE HIM OF THE NEED TO WRITE I/O HANDLERS FOR EACH PROGRAM HE WRITES.

BEFORE ANY CALL TO THE MONITOR IS PREFORMED THE B&C REGISTER PAIR MUST CONTAIN THE ADDRESS OF A MONITOR CONTROL BLOCK. A MONITOR CONTROL BLOCK IS USED TO SPECIFY THE OPERATION TO BE PERFORMED, SYMBOLIC DEVICE TO USE, ETC. THE MONITOR IS CALLED BY EXECUTING A RST 6. ALL REGISTERS ARE RESTORED BEFORE RETURNING TO THE CALLING PROGRAM, AND THE MONITOR CONTROL BLOCK IS LEFT UNCHANGED.

1. READ

```
                LXI     B,RDPKT ;LOAD B&C WITH ADDR OF THE
                                ;MONITOR CONTROL BLOCK
                RST     6       ;CALL THE MONITOR
- DONT PUT MONITOR CONTROL BLOCK NEXT IN YOUR PROGRAM -
RDPKT:  DB      20Q             ;OPERATION CODE FOR READ
        DB      "CTR"           ;SYMBOLIC DEVICE NAME
        DW      INBUF           ;ADDRESS OF THE BEGINNING OF THE
                                ;INPUT BUFFER
        DW      80              ;MAXIMUM NUMBER OF CHARACTERS
                                ;TO BE READ IN
        DW      STAT            ;ADDRESS OF STATUS WORD
        DW      END             ;END OF FILE RETURN ADDRESS
```

AFTER THE READ HAS BEEN COMPLETED THE STATUS WORD CONTAINS THE NUMBER OF BYTES READ IN.

2. WRITE

```
                LXI     B,WRPKT ;LOAD B&C WITH ADDRESS OF THE
                                ;MONITOR CONTROL BLOCK
                RST     6       ;CALL THE MONITOR

WRPKT:  DB      22Q             ;OPERATION CODE FOR WRITE
        DB      "CTR"           ;SYMBOLIC DEVIE NAME
        DW      OUTBUF          ;ADDRESS OF THE OUTPUT BUFFER
        DW      80              ;NUMBER OF BYTES TO WRITE OUT
        DW      STAT            ;ADDRESS OF STATUS WORD
```

THE STATUS WORD CONTAINS THE NUMBER OF BYTES OUTPUT WHEN THE MONITOR RETURNS.

3. OPEN

```
                LXI     B,OPNPKT ;LOAD B&C WITH ADDRESS OF THE
                                 ;MONITOR CONTROL BLOCK
                RST     6        ;CALL THE MONITOR

OPNPKT: DB      63Q              ;OPERATION CODE FOR OPEN
        DB      "CTB"            ;SYMBOLIC DEVICE NAMED TO BE
                                 ;OPENED
        DB      "TY"             ;PHYSICAL DEVICE TO BE OPENED TO
        DB      XXX              ;HARDWARE CONTROL BYTE
```

THE HARDWARE CONTROL BYTE SPECIFIES ECHO CONTROL AND AND ASCII OR ABSOLUTE READ MODE.
BIT 1 = 1 FOR ASCII READ MODE, 0 FOR ABSOLUTE
BIT 2 = 1 FOR INPUT ECHO, 0 FOR NO ECHO

4. CLOSE

```
                LXI     B,CLSPKT ;LOAD B&C WITH THE ADDRESS OF
                                 ;THE MONITOR CONTROL BLOCK
                RST     6        ;CALL THE MONITOR

CLSPKT: DB      62Q              ;OPERATION CODE FOR CLOSE
        DB      "CTB"            ;SYMBOLIC NAME TO BE CLOSED
```

5. ERROR

```
                LXI     B,ERRPKT ;LOAD B&C WITH ADDRESS OF THE
                                 ;MONITOR CONTROL BLOCK
                RST     6        ;CALL THE MONITOR

ERRPKT: DB      60Q              ;OPERATION CODE FOR ERROR
                                 ;HANDLING ROUTINE
        DB      "X"              ;ONE CHARACTER TO BE OUTPUT AS
                                 ;ERROR MESSAGE
```

THE CHARACTER SPECIFIED FOLLOWED BY A # SIGN WILL BE ECHOED BY THE MONITOR INSTEAD OF THE NEXT 2 CHARACTERS THAT WOULD NORMALLY BE ECHOED.

6. PASS PROGRAM NAME

```
        LXI     B,PASPKT ;LOAD B&C WITH ADDRESS OF THE
                         ;MONITOR CONTROL BLOCK
        RST     6        ;CALL THE MONITOR

PASPKT: DB      61Q      ;OPERATION CODE FOR PASS PROGRAM
                         ;NAME ROUTINE
        DB      "PRG"    ;3 CHARACTER PROGRAM NAME
        DW      PRG      ;START ADDRESS OF PROGRAM
```

THE 5 BYTES OF NAME AND ADDRESS ARE COPIED INTO THE PTL.

7. FIND ASCII FILE

```
        LXI     B,FFPKT  ;LOAD B&C WITH ADDRESS OF THE
                         ;MONITOR CONTROL BLOCK
        RST     6        ;CALL THE MONITOR

FFPKT:  DB      65Q      ;OPERATION CODE FOR A FIND FILE
        DB      "CTB"    ;SYMBOLIC DEVICE TO SEARCH ON
        DB      "FIL"    ;FILE TO BE SEARCHED FOR
```

THIS CALL CAUSES THE MONITOR TO SEARCH FOR THE NAMED PROGRAM ON THE PHYSICAL DEVICE THE SYMBOLIC DEVICE NAME IS OPEN TO AND RETURN TO THE CALLING PROGRAM AS SOON AS IT IS FOUND.

8. RETURNING TO THE MONITOR

WHEN A PROGRAM HAS FINISHED ITS JOB AND WISHES TO RETURN TO THE MONITOR, A BRANCH TO LOCATION 0 WILL ACCOMPLISH THIS. THE STACK POINTER IS AUTOMATICALLY RELOADED SO THAT ANYTHING LEFT ON THE STACK IS DELETED. THIS ALLOWS THE USE OF A RST 0 INSTRUCTION TO RETURN TO THE MONITOR.

NOTE - 30 OCTAL BYTES OF STACK HAVE BEEN ALLOCATED FOR USER PROGRAM USE. IF MORE STACK SPACE IS NEEDED, A LXI SP INSTRUCTION WILL BE NEEDED IN YOUR PROGRAM TO SET UP ITS OWN STACK.

III. ASSEMBLER OPERATION

INTRODUCTION

TYPICAL ASSEMBLERS PROCESS SOURCE CODE BY READING THE SAME SOURCE CODE 2,3, OR 4 TIMES TO PRODUCE A LOAD TAPE THAT MUST THEN BE LOADED BEFORE EXECUTING THE PROGRAM. AN ASSEMBLER THAT REQUIRES THAT TYPE OF PROCEDURE IS EXTREMELY CUMBERSOME FOR USERS WITH PAPER TAPE OR CASSETTE MAGNETIC TAPE INPUT. OFF-LINE STORAGE IS ALWAYS REQUIRED FOR ASSEMBLIES OF THIS TYPE. FURTHER, HIGH SPEED STORAGE IS DESIREABLE DUE TO THE EXTENSIVE I-O REQUIRED DURING PROCESSING.

THE MITS LOADING ASSEMBLER WAS DESIGNED TO PROCESS SOURCE CODE DIRECTLY INTO MEMORY FOR IMMEDIATE EXECUTION OR TO PRODUCE AN ABSOLUTE LOAD TAPE FOR LATER EXECUTION IN THE SPACE OCCUPIED BY THE ASSEMBLER. THE SOURCE CODE IS PROCESSED ONLY ONCE, THEREBY PRODUCING EXECUTABLE CODE IN A MINIMUM AMOUNT OF TIME. SIGNIFICANT IMPROVEMENT IN PROGRAM DEVELOPMENT TIME IS ACHIEVED, ESPECIALLY FOR USERS WITH PROGRAM INPUT RATES UNDER 100 CHARACTERS/SECOND. FURTHERMORE, THE ASSEMBLER IS STILL RESIDENT IN MEMORY WITH THE USER PROGRAM, SO IT CAN BE USED TO PATCH PROGRAM ERRORS DURING DEBUGGING. SINCE THE PATCHES CAN BE ENTERED IN SYMBOLIC SOURCE CODE AND LABELS CAN STILL BE ASSIGNED TO CORRECT EXECUTION SEQUENCES FEWER ERRORS ARE INTRODUCED IN THE DEBUG PROCESS. THUS, COMPLETE PROGRAMS CAN BE INPUT AND DEVELOPED WITH INPUT FROM ONLY AN ASCII KEYBOARD AND A MINIMAL AMOUNT OF MEMORY(THE MONITOR AND ASSEMBLER REQUIRE APPROXIMATELY 5K BYTES).

THE LOADING ASSEMBLER ALLOWS DIRECT LOADING OF EXECUTABLE PROGRAMS TO ANY UNUSED MEMORY SPACE OR INDIRECT LOADING VIA OFF-LINE LOAD TAPE TO ANY PLACE IN MEMORY. PROGRAM MODULES(PARTS) CAN BE DEVELOPED BY ASSEMBLY TO EXECUTION MEMORY FOR DEBUGGING, THEN ASSEMBLED WITH ALL SOURCE ERRORS CORRECTED FOR OFF-LINE LOADING TO OTHER PROGRAM SPACE. THE MODULES CAN BE LINKED BY ASSEMBLY USING THE PRESERVED SYMBOLS FROM PREVIOUS ASSEMBLIES OR BY DEFINING NAMES FOR REFERENCED LOCATIONS TO OTHER ASSEMBLED MODULES PRIOR TO EACH ASSEMBLY.

ASSEMBLER OPTIONS

THE ASSEMBLER WAS DESIGNED AS A MODULE OF THE MITS OPERATING SYSTEM TO BE LOADED BY THE SYSTEM MONITOR. THE ASSEMBLER IS LOADED FROM THE INPUT DEVICE SPECIFIED(I.E. ASM,CTB LOADS THE ABSOLUTE TAPE FROM THE DEVICE CTB IS OPEN TO). EXECUTION OF THE ASSEMBLER WILL BEGIN WHEN AN END OF TAPE IS DETECTED. LATER ASSEMBLY(S) CAN BE ENTERED WITHOUT RELOADING THE PROGRAM AND THE USER CAN SELECT ENTRY OPTIONS AS FOLLOWS:

ASM(I,A,S,P)

WHERE: P=PRESERVE SYMBOLS ENTERED DURING PREVIOUS ASSEMBLY(S).
USED FOR SYMBOLIC PATCHES & PROGRAM ADDITIONS

S=SYMBOL TABLE LISTING WANTED AT END.
ALL DEFINED NAMES & LABEL SYMBOLS WITH
CORRESPONDING PROGRAM ADDRESSES
AND THE NEXT PROGRAM ADDRESS($) ARE OUTPUT

I=INVERSE ASSEMBLY LISTING WANTED AT END.
OCTAL PROGRAM ADDRESSES,STORED BYTES, AND OPCODE
EQUIVALENCES. IF SENSE SWITCH 14 IS RAISED NO
SYMBOLIC OPCODES ARE PRINTED.

A=ABSOLUTE TAPE DUMP WANTED AT END.
BINARY OUTPUT FOR MONITOR LOADING

ALL OUTPUT BEGINS AT THE FIRST STORAGE ADDRESS THAT WAS DEFINED WHEN BEGINNING ASSEMBLY AND CONTINUES TO THE CURRENT ADDRESS. PROGRAM ADDRESSES ARE USED FOR THE ABSOLUTE TAPE DUMP, INVERSE AND DEFINED SYMBOL LISTINGS.
WARNING*** IF ASSEMBLY IS BEGUN WITHOUT THE P OPTION THE SYMBOL TABLE IS CLEARED AND IS NOT RECOVERABLE.

SOURCE PROGRAM INPUT-OUTPUT

SOURCE PROGRAMS ARE INPUT FROM A MONITOR-DEFINED DEVICE CALLED "INP" AND ALL SELECTED OUTPUT IS TO DEVICE "LST". INPUT CAN BE SELECTED FROM A SOURCE FILE THAT WAS CREATED BY THE EDITOR. THE FILES CAN BE ASSEMBLED IN ANY SEQUENCE SELECTED.

FILE OPCODE

THE FILE INPUT OPCODE FORCES SOURCE INPUT TO BE READ FROM SYMBOLIC DEVICE "MAG". IF A FILE NAME IS GIVEN AS AN OPERAND, A FILE WITH THAT HEADER IS SEARCHED FOR BEFORE PROCESSING ANY INPUT. THE SOURCE FILE MUST END WITH A CONTROL Z OR EOA OPCODE SO THAT CONTROL IS RESTORED TO THE MONITOR AT THE END OF FILE. ASSEMBLY CAN BE CONTINUED BY ENTERING THE ASSEMBLER WITH THE P-OPTION.

EXAMPLE: FILE TWO INPUT FILE "TWO" FROM CASSETTE

END OPCODE

ALL OF THE ENTRY OPTIONS IN THE GROUP(S,I,A) ARE PERFORMED EACH TIME AN END PSEUDO-OPCODE IS ENCOUNTERED. THE END STATEMENT WILL ALSO PRODUCE A LISTING OF ALL UNDEFINED SYMBOL NAMES WITH PROGRAM STORAGE LOCATIONS THAT REFERENCE THE SYMBOL. THE ENTRY OPTIONS AND END STATEMENT PROVIDE INFORMATION TO ASSIST THE DEBUGGING TASK AND TO DOCUMENT THE PROGRAM VIA A LOAD MAP(EXECUTION MEMORY) AND MACHINE CODE/OPCODE LISTING.
IF THE A-OPTION WAS SELECTED THE FIRST 3-LETTERS OF THE OPERAND DEFINE THE PROGRAM NAME WHEN LOADED BY THE MONITOR. UP TO 77 CHARACTERS FOLLOWING CAN BE USED TO DOCUMENT THE PROGRAM(REVISION, DATE, ETC.)

EOA OPCODE

THE ASSEMBLER WILL RETURN CONTROL TO THE MONITOR WHEN AN END OF ASSEMBLY(EOA) PSEUDO-OPCODE IS ENCOUNTERED.
THE MONITOR PRINTS A PROMPT TO INDICATE IT IS IN CONTROL.

MEMORY ALLOCATION

THE USER MUST UNDERSTAND THE WAY THAT MEMORY IS USED DURING THE ASSEMBLY PROCESS TO AVOID ERRORS AND TO USE AVAILABLE MEMORY IN AN EFFICIENT WAY. THE DIAGRAM IN APPENDIX B ILLUSTRATES THE RELATIVE STORAGE USED DURING ASSEMBLY.
THE USER MUST ESTIMATE THE SYMBOL SPACE NEEDED FOR EACH ASSEMBLY BEFORE DEFINING THE FIRST STORAGE LOCATION. IT SHOULD BE APPARANT THAT SHORT SYMBOLS AND FEW LABELS OR NAMES WILL INCREASE THE SPACE AVAILABLE FOR USER PROGRAM STORAGE.

ORR & ORG OPCODES

THE ASSEMBLER CHECKS FOR SYMBOL TABLE OVERFLOW INTO USER PROGRAM SPACE AND FOR USER PROGRAM MEMORY OVERFLOW(TYPICALLY 8K). ASSEMBLY WILL ABORT AND RETURN CONTROL TO THE MONITOR IF EITHER OCCURS. THE USER MUST ALLOCATE PROGRAM/EXECUTION SPACE PRIOR TO ASSEMBLY BY USING THE ORR AND ORG PSEUDO OPCODES TO CONTROL THE STORAGE AND PROGRAM ADDRESS SPACE RESPECTIVELY.
THE PROGRAM ADDRESS AND STORAGE SPACE POINTERS ARE INITIALIZED TO ADDRESS 0 UNLESS THE PRESERVE(P OPTION) IS SELECTED FOR ASSEMBLER ENTRY. THE ORG OPCODE ALTERS BOTH ADDRESSES TO THE OPERAND SPECIFIED VALUE FOR PROGRAM EXECUTION AND MUST BE ENTERED BEFORE ANY OTHER INSTRUCTION. IF THE PROGRAM IS TO BE ASSEMBLED FOR OFF-LINE LOADING(IT USES SPACE OCCUPIED BY THE MONITOR OR ASSEMBLER) THE THE ORR OPCODE MUST FOLLOW THE ORG TO SPECIFY THE PROGRAM STORAGE ADDRESS. THE PROGRAM STORAGE ADDRESS SPECIFIED BY THE ORG OPERAND FOR ON-LINE ASSEMBLY BE IN HIGH ENOUGH MEMORY TO LEAVE ROOM FOR PROGRAM SYMBOL TABLE GENERATION. THE ASSEMBLER BUILDS THE SYMBOL TABLE UPWARD TOWARD PROGRAM STORAGE. SEE APPENDIX B MEMORY ALLOCATION DURING ASSEMBLY.

DS OPCODE

STORAGE IS ALLOCATED WITH THE DS OPCODE. THE VALUES OF BYTES IN THE SPACE ARE NOT CHANGED AND SHOULD NORMALLY BE PRESET DURING EXECUTION PRIOR TO USE. THE OPERAND MUST BE A DEFINED SYMBOL(EQU OR SET TO A CONSTANT) OR A CONSTANT VALUE. A LABEL SYMBOL DEFINES AN ADDRESS WHICH SHOULD NOT GENERALLY BE USED FOR A STORAGE OPERAND.
A DS OPCODE IS GENERALLY PRECEDED BY A LABEL USED TO REFERENCE THE STORAGE ALLOCATED DURING PROGRAM EXECUTION.(I.E. LABEL: DS 100)

DW OPCODE

AN ADDRESS WORD OR TWO BYTE QUANTITY IS PRESET(ASSIGNED DURING ASSEMBLY) BY USING THE DW OPCODE. THE 2-BYTE VALUE IS STORED WITH THE LEAST SIGNIFICANT BYTE IN THE FIRST MEMORY ADDRESS AND THE MOST SIGNIFICANT BYTE IN THE NEXT HIGHER MEMORY LOCATION. THIS FEATURE IS THE SAME AS ALL 2-BYTE OPERANDS FOR MACHINE OPCODES(I.E. JMP,LXI,ETC.) THIS ARRANGEMENT IS CONVENIENT BECAUSE IT ALLOWS BYTE REFERENCES TO THE LEAST SIGNIFICANT BYTE USING THE SAME LABEL AS A WORD REFERENCE. MULTIPLE OPERANDS ARE ALLOWED AND MUST BE SEPARATED WITH A SPACE OR COMMA.

DB OPCODE

ALL BYTE CONSTANTS ARE DEFINED BY USING THE DB OPCODE. MULTIPLE OPERANDS CAN BE USED. ALL OPERANDS DEFINE ONE BYTE OF STORAGE EXCEPT STRING OR LITERAL CONSTANTS WHICH ARE STORED AS ONE ASCII CHARACTER PER BYTE. EACH OPERAND MUST BE SEPARATED BY A SPACE OR COMMA DELIMITER.

DC OPCODE

THE DEFINE CHARACTER PSUEDO OP IS USED TO DEFINE LITERAL CONSTANTS OF DETERMINABLE LENGTH. ALL CHARACTERS EXCEPT THE LAST HAVE THEIR HIGH ORDER BIT MASKED TO ZERO, BUT THE LAST CHARACTER HAS IT SET TO ONE. THE LAST CHARACTER CAN THEN BE FOUND BY SEARCHING FOR A CHARACTER WITH ITS HIGH ORDER BIT ON.

EQU OPCODE

A SYMBOL CAN BE DEFINED PRIOR TO USE BY ASSIGNING IT A VALUE EQUAL TO A SPECIFIED CONSTANT OR ANOTHER LABEL THAT HAS ALREADY BEEN DEFINED. A SYMBOLIC NAME(NOT A LABEL) IS DEFINED BY USING THE EQU OPCODE AND A DEFINED OPERAND. THE EQU OPCODE CANNOT BE USED TO CHANGE THE VALUE ASSIGNED TO A NAME. REFER TO THE SET OPCODE.
NOTE: ERRORS ARE SOMETIMES SET AS NAME SYMBOLS THAT ARE UNDEFINED. SINCE OPERAND VALUES ARE ASSIGNED TO THE FIRST UNDEFINED NAME, A VALUE CAN BE ASSIGNED TO THE WRONG NAME AFTER AN ERROR.

SET OPCODE

A NAME CAN BE CHANGED OR REASSIGNED BY USING THE SET OPCODE IN THE SAME MANNER AS THE EQU OPCODE. IT CAN ALSO BE USED TO ASSIGN A VALUE THE FIRST TIME A NAME IS DEFINED.

BEG OPCODE

THE OPERAND OF THE BEG OPCODE IN A PROGRAM SETS THE BEGIN EXECUTION ADDRESS, OUTPUT BY THE ASSEMBLER DURING AN ABSOLUTE DUMP. IF THE BEG OPCODE IS NOT FOUND A START ADDRESS OF 0 WILL BE ASSUMED, CAUSING A RETURN TO THE MONITOR WHEN THE PROGRAM IS LOADED.

RUN OPCODE

THE OPERAND OF THE RUN OPCODE IS RETURNED AS A PROGRAM NAME ALONG WITH THE ADDRESS FROM THE LATEST BEG STATEMENT TO BE ENTERED INTO THE PTL. THE ADDRESS OF THE BEG STATEMENT IS THEN BRANCHED TO. THIS OPCODE SHOULD ONLY BE USED DURING ON-LINE ASSEMBLY.

IV. ASSEMBLER PROGRAMMING

ASSEMBLY PROGRAMS INCLUDE SYMBOLIC NAMES,CONSTANTS, OPCODES AND COMMENTS IN SEQUENTIAL STATEMENTS THAT ARE CONVERTED BY THE ASSEMBLER TO PRODUCE EXECUTABLE MACHINE INSTRUCTIONS. EACH LINE OR PROGRAM STATEMENT OF SOURCE CODE MUST FOLLOW CERTAIN RULES THAT GOVERN THE ACCEPTABLE STRUCTURE OF THE PROGRAM. IF THEY ARE NOT OBSERVED, ASSEMBLY OR EXECUTION ERRORS WILL OCCUR. THIS SECTION WILL DEFINE

THE FORM THAT IS ACCEPTABLE TO THE MITS LOADING ASSEMBLER.

CHARACTER SET

THE ENTIRE 128-CHARACTER ASCII CHARACTER SET IS ACCEPTABLE BUT ALL OPCODES ARE DEFINED IN CAPITAL LETTERS. ANY COMBINATION OF CHARACTERS BEGINNING WITH A NON-NUMERIC CHARACTER CAN BE USED FOR STATEMENT LABELS OR SYMBOLIC NAMES. THE MAXIMUM LENGTH FOR THESE SYMBOLS IS 255 CHARACTERS BUT TO MINIMIZE SYMBOL TABLE LENGTH THEY SHOULD BE KEPT AS SHORT AS POSSIBLE.

CONSTANTS

CONSTANTS CAN BE USED WHENEVER AN OPERAND IS REQUIRED. ALL CONSTANTS BEGIN WITH A NUMERIC CHARACTER AND CAN END WITH AN ALPHABETIC CHARACTER THAT DEFINES THE RADIX OF CONVERSION. IF THE LAST CHARACTER IS NUMERIC THE CONVERSION DEFAULTS TO DECIMAL. LEGAL CONVERSIONS ARE AS FOLLOWS:

12340	OCTAL
5678Q	OCTAL(8 CONVERTS AS Ø)
12345D	DECIMAL
ØABCD	DECIMAL(A CONVERTS AS 1,ETC.)
Ø57EFH	HEXIDECIMAL

NOTE: VALUES ARE FIRST MASKED LEAVING ONLY THE SIGNIFICANT BINARY QUANTITIES, THUS ALPHABETIC CONVERSIONS ARE LEGAL.
BYTE VALUES ARE SET EQUAL TO THE CONVERTED VALUE USING MODULUS 256 ARITHMETIC. SIMILARLY, OVERFLOW OF 16-BIT CONSTANTS(WORDS) DURING CONVERSION IS IGNORED.

STRING OR LITERAL CONSTANTS ARE DEFINED BY ENCLOSING ALL CHARACTERS IN " SYMBOLS. THE " SYMBOL CANNOT BE DEFINED IN A STRING CONSTANT. (I.E. DB "THIS IS A MESSAGE" IS A CONVENIENT WAY TO STORE A MESSAGE FOR OUTPUT DURING PROGRAM EXECUTION)

WARNING****** ONLY ONE CHARACTER SHOULD BE USED IF A SINGLE BYTE OPERAND IS REQUIRED(I.E. MVI A,"ABC" WILL STORE 4 BYTES).

STATEMENT STRUCTURE

THE ASSEMBLY SOURCE STATEMENTS MAY INCLUDE ANY OF THE FOLLOWING IN THE ORDER GIVEN:

1. SYMBOLIC LABEL OF ANY LENGTH TERMINATED BY A COLON(:). THE SYMBOL CAN INCLUDE ANY ASCII CHARACTER EXCEPT DELIMITERS(SPACE OR COMMA) IN ANY COMBINATION INCLUDING INSTRUCTION OPCODES. THE FOLLOWING SYMBOLS ARE PREDEFINED VALUES.

   $=NEXT PROGRAM BYTE ADDRESS
   FOLLOWING ARE ONLY VALID BYTE(NOT WORD)VALUES.
   B,C,D,E,H,L,M,A=Ø,1,2,3,4,5,6,7 RESPECTIVELY
   SP AND PSW=6

2. A NAME IS THE SAME AS A LABEL EXCEPT THAT A TERMINATING COLON IS NOT USED. A NAME IS USED IN PLACE OF A LABEL AND REMAINS UNDEFINED UNTIL A DEFINING PSEUDO OPCODE IS ENCOUNTERED (I.E. EQU,SET).

3. OPCODE(S) OR PSEUDO OPCODE(S) WITH REQUIRED OPERANDS. ALL OPCODES THAT ARE DEFINED IN THE ALTAIR 8800 OPERATORS MANUAL AND ALL PSEUDO-OPCODE DEFINED IN SECTION II ARE ACCEPTABLE TO THE LOADING ASSEMBLER.

4. COMMENTS ARE USED TO DOCUMENT THE SOURCE CODE AND SHOULD BE STATED IN TERMS OF THE FUNCTION BEING PERFORMED BY THE PROGRAM. COMMENTS BEGIN WITH A SEMI-COLON(;) WHICH TERMINATES ASSEMBLY OF ALL FOLLOWING ASCII CHARACTERS ON THE LINE. LINES THAT BEGIN WITH A SEMI-COLON CONTAIN ONLY COMMENTS.

STATEMENT OPTIONS

ALL REGISTER PAIR INSTRUCTION OPERANDS CAN REFERENCE EITHER OF THE TWO 8-BIT REGISTERS IN THE PAIR.
THUS--
POP A IS THE SAME AS POP PSW
LXI L IS THE SAME AS LXI H
DCX C IS THE SAME AS DCX D

MULTIPLE INSTRUCTIONS CAN APPEAR ON THE SAME LINE OF SOURCE CODE. THIS FEATURE CAN BE USED TO MINIMIZE THE NUMBER OF CHARACTERS ON A SOURCE TAPE AND IN SOME CASES IMPROVES THE PROGRAM READABILITY. (I.E. PUSH B, POP D
RAL,RLC,RAR,RAR)

THE DELIMITERS(SPACE OR COMMA)CAN BE USE ANYWHERE IN THE LINE TO IMPROVE READABILITY.

PROGRAMMING TRICKS

THE CHOICE OF OPCODE(S) TO USE TO ACHIEVE A SPECIFIC RESULT IS GENERALLY BASED ON THE GENERALLY ACCEPTED CRITERIA THAT A PROGRAM SHOULD USE A MINIMAL AMOUNT OF SPACE AND SHOULD EXECUTE AS RAPIDLY AS POSSIBLE. OFTEN ONE CONSIDERATION IS SACRIFICED IN FAVOR OF THE OTHER BUT CERTAIN PRACTICES SHOULD ALWAYS BE AVOIDED IN FAVOR OF ANOTHER WHICH PRODUCES THE SAME RESULT AT LESS COST IN TIME OR STORAGE. THE FOLLOWING PRACTICES ARE RECOMMENDED:

1. AVOID CALL SUBROUTINE
   RET
   IN FAVOR OF
   JMP SUBROUTINE

   THE JMP STATEMENT WILL RETURN TO THE SAME PLACE WITHOUT NEED FOR THE RET THUS SAVING ONE BYTE OF PROGRAM STORAGE.

2. AVOID CPI 0
   WHICH REQUIRES TWO BYTES OF STORAGE
   IN FAVOR OF
   ORA A
   WHICH REQUIRES ONE BYTE OF STORAGE

   ALL FLAG BITS ARE AFFECTED IN THE SAME MANNER WITHOUT CHANGING THE CONTENTS OF THE A-REGISTER.

3. AVOID PUSH B,POP H
   OR SIMILAR REGISTER CONTENTS TRANSFER
   IN FAVOR OF
   MOV H,B MOV L,C
   WHICH EXECUTES IN LESS TIME.

4. IF A SERIES OF MVI 2-BYTE INSTRUCTIONS ARE USED FOLLOWED BY A JUMP TO THE SAME ADDRESS,LESS STORAGE CAN BE USED BY LXI B(DB 1) REPLACING THE JUMPS SAVING TWO BYTES.

FOR EXAMPLE:

```
MVI A,"A"                    MVI A,"A"
DB 1                         JMP ERR
MVI A,"B"                    MVI A,"B"
DB 1                         JMP ERR

 ETC.

ERR:
```

A JUMP TO ANY MVI TO SET THE "ERR" CODE WILL LOAD THE A-REG WITH THE CHARACTER TO BE OUTPUT AND WILL SKIP OVER THE REST BY EXECUTING THE DB 1 AS LXI B,XXXX WHERE XXXX IS THE TWO-BYTE MVI INSTRUCTIONS. THE CONTENTS OF BC WILL CHANGE. OTHER REGISTER PAIRS CAN BE USED SIMILARLY.

EXAMPLE PROGRAM

A SAMPLE PROGRAM AND ASSEMBLY ARE GIVEN TO ILLUSTRATE THE OPERATION OF THE ASSEMBLER AND USE OF THE MONITOR CALLS FOR OUTPUT.

THE SAMPLE PROGRAM WILL DUMP OUT ANY SECTION OF MEMORY IN OCTAL AS SHOWN LATER IN THE EXAMPLE. THIS TYPE OF MEMORY DUMP CAN BE VERY USEFUL IN DEBUGGING PROGRAMS. IN ORDER TO USE THIS PROGRAM, CHANGE THE ADDRESSES AT LOCATIONS FIRST AND LAST TO THE ADDRESS OF THE FIRST AND LAST MEMORY LOCATION YOU WANT DUMPED.

```
        ORG     20000Q  ;SET LOCATION COUNTER
DUMP:   LHLD    FIRST   ;GET ADDRESS OF FIRST BYTE TO BE DUMPED
        XCHG            ;PUT ADDRESS IN D&E
NEWLN:  LXI     H,BUF   ;GET ADDDRESS OF OUTPUT BUFFER
        PUSH    H       ;SAVE ADDRESS
        LHLD    LAST    ;LOAD ADDRESS OF LAST BYTE TO BE DUMPED
        MOV     A,L     ;SUBTRACT LOW ORDER BYTES
        SUB     E
        MOV     A,H     ;SUBTRACT HIGH ORDER BYTE
        SBB     D
        POP     H       ;RESTORE H&L
        JC      NONE    ;JUMPS OUT IF NO MORE BYTES TO BE DUMPED
        MOV     A,D     ;START CONVERSION OF ADDRESS TO ASCII
        RAL             ;ROTATE HIGH BIT INTO C
        MVI     A,0     ;ZERO OUT REST OF A BUT DONT CHANGE FLAGS
        RAL             ;ROTATE HIGH BIT INTO LOW ORDER POSITION
        ORI     60Q     ;OR IN ASCII 0
        MOV     M,A     ;STORE IN OUTPUT BUFFER
        INX     H       ;INCREMENT BUFFER POINTER
        MOV     A,D     ;PICK UP HIGH ORDER BITE AGAIN
        RAR             ;ROTATE BITS 4,5,6 INTO LOW ORDER POSITIONS
        RAR
        RAR
        RAR
        ANI     7       ;MASK OFF ALL BITS EXCEPT LOW THREE
        ORI     60Q     ;OR IN ASCII 0
        MOV     M,A     ;STORE IN OUTPUT BUFFER
        INX     H       ;INCREMENT POINTER INTO OUTPUT BUFFER
        MOV     A,D     ;PICK UP HIGH BYTE OF ADDRESS
        RAR             ;ROTATE BITS 1,2,3 INTO LOW ORDER POSITION
        ANI     7       ;MASK OFF ALL BITS EXCEPT LOW THREE
        ORI     60Q     ;OR IN ASCII 0
        MOV     M,A     ;STORE IN OUTPUT BUFFER
        INX     H       ;INCREMENT POINTER INTO OUTPUT BUFFER
        MOV     A,D     ;PICK UP HIGH BYTE OF ADDRESS
        RAR             ;SAVE LOW BIT IN THE CARRY FLAG
        MOV     A,E     ;PICK UP LOW BYTE OF ADDRESS
        CALL    LAST3   ;CALL SUBROUTINE THAT CONVERTS 3 DIGITS
        MVI     B,8     ;LOAD B WITH NO OF BYTES TO DUMP PER LINE
NXTNUM: PUSH    H       ;SAVE H&L
        LHLD    LAST    ;LOAD ADDRESS OF LAST BYTE TO DUMP
        MOV     A,L     ;DO THE DOUBLE WORD COMPARE AGAIN
        SUB     E
        MOV     A,H
        SBB     D
        POP     H       ;RESTORE THE H&L REGISTERS
        JC      LNDN    ;JUMP TO ROUTINE TO FINISH UP IF DONE
        MVI     C,5     ;IF ANOTHER TO COME SEPERATE THEM BY 5 BLANKS
        CALL    BLANK
        XRA     A       ;SET THE CARRY FLAG TO 0
        LDAX    D       ;GET BYTE TO DUMP
        CALL    LAST3   ;CALL ROUTINE TO CONVERT 3 DIGITS
        INX     D       ;INCREMENT POINTER TO DUMP NEXT BYTE
CHKLN:  DCR     B       ;DECREMENT LINE BYTE COUNTER
        JNZ     NXTNUM  ;JUMP TO CONVERT NEXT NUMBER IF IT WILL FIT ON LINE
        LXI     B,OUT   ;GET ADDRESS OF MONITOR CONTROL BLOCK
        RST     6       ;WRITE OUT LINE
        JMP     NEWLN   ;JUMP TO WRITE OUT NEXT LINE
LNDN:   MVI     C,8     ;PAD LINE WITH 8 BLANKS FOR EACH NUMBER THAT WOULD FIT
        CALL    BLANK
        DCR     B       ;DECREMENT NUMBERS THAT COULD FIT IN LINE
        JNZ     LNDN    ;LOOP UNTIL LINE FILLED WITH BLANKS
        LXI     B,OUT   ;GET ADDRESS OF MONITOR CONTROL BLOCK
        RST     6       ;WRITE OUT LINE
NONE:   RST     0       ;RETURN TO THE MONITOR
BLANK:  MVI     A,40Q   ;PUT A ASCII BLANK IN A
BL:     MOV     M,A     ;STORE IT IN THE OUTPUT BUFFER
        INX     H       ;INCREMENT THE OUTPUT BUFFER POINTER
```

```
        DCR     C       ;DECREMENT THE NUMBER OF BLANKS TO STORE
        JNZ     BL      ;LOOP UNTIL ALL STORED
        RET             ;RETURN TO CALLER
LAST3:  PUSH    PSW     ;SAVE BYTE TO CONVERT
        RAL             ;ROTATE CARRY AND HIGH 2 BITS TO LOW ORDER POSITION
        RAL
        RAL
        ANI     7       ;MASK OFF ALL BUT LOW ORDER THREE BITS
        ORI     60Q     ;OR IN A ASCII 0
        MOV     M,A     ;STORE DIGIT IN OUTPUT BUFFER
        INX     H       ;INCREMENT THE OUTPUT BUFFER POINTER
        POP     PSW     ;POP BYTE TO CONVERT
        PUSH    PSW     ;SAVE FOR LATER ALSO
        RAR             ;ROTATE BITS 3,4,5 INTO LOW ORDER POSITION
        RAR
        RAR
        ANI     7       ;MASK OFF ALL BUT LOW THREE BITS
        ORI     60Q     ;OR IN AN ASCII 0
        MOV     M,A     ;STORE DIGIT IN OUTPUT BUFFER
        INX     H       ;INCREMENT OUTPUT BUFFER POINTER
        POP     PSW     ;POP BYTE TO CONVERT
        ANI     7       ;MASK OFF ALL BUT LOW THREE BYTES
        ORI     60Q     ;OR IN ASCII 0
        MOV     M,A     ;STORE DIGIT IN OUTPUT BUFFER
        INX     H       ;INCREMENT OUTPUT BUFFER POINTER
        RET             ;RETURN TO CALLER
OUT:    DB      22Q     ;MONITOR WRITE OPERATION CODE
        DB      "LST"   ;SYMBOLIC DEVICE TO WRITE ON
        DW      BUF     ;ADDRESS OF OUTPUT BUFFER
        DW      72      ;WRITE OUT 72 CHARATERS
        DW      STAT    ;ADDRESS OF STATUS WORD
STAT:   DW      0
FIRST:  DW      15100Q
LAST:   DW      15272Q
BUF:    DS      70
        DB      15Q     ;ASCII CARRIAGE RETURN
        DB      12Q     ;ASCII LINE FEED
        ORG     OUT     ;SET ORIGIN SO THAT CONSTANTS ARE NOT UNASSEMBLED
        BEG     DUMP    ;SETS ADDRESS OF PLACE TO START EXECUTING PROGRAM
        END     DMP
        EOA
```

CHANGE SENSE SWITCH 15 FOR DUMP

NOTE: AT THIS POINT THE PUNCH OR OUTPUT TAPE IS READIED FOR
OUTPUT OF THE PROGRAM IN ABSOLUTE BINARY FORMAT(APPENDIX A).

UNDEFINED SYMBOLS

SYMBOL TABLE

```
DUMP 020000
FIRST 020234
NEWLN 020004
BUF 020236
LAST 020232
NONE 020147
LAST3 020161
NXTNUM 020067
LNDN 020132
BLANK 020150
CHKLN 020116
OUT 020216
BL 020152
STAT 020230
020000 052 LHLD 020234
020003 353 XCHG
020004 041 LXI 020232
020007 345 PUSH
020010 052 LHLD 020236
020013 175 MOV
020014 223 SUB
020015 174 MOV
020016 232 SBB
020017 341 POP
020020 332 JC 020147
020023 172 MOV
020024 027 RAL
020025 076 MVI 000
```

```
020027 027 RAL
020030 366 ORI 060
020032 167 MOV
020033 043 INX
020034 172 MOV
020035 037 RAR
020036 037 RAR
020037 037 RAR
020040 037 RAR
020041 346 ANI 007
020043 366 ORI 060
020045 167 MOV
020046 043 INX
020047 172 MOV
020050 037 RAR
020051 346 ANI 007
020053 366 ORI 060
020055 167 MOV
020056 043 INX
020057 172 MOV
020060 037 RAR
020061 173 MOV
020062 315 CALL 020161
020065 006 MVI 010
020067 345 PUSH
020070 052 LHLD 020232
020073 175 MOV
020074 223 SUB
020075 174 MOV
020076 232 SBB
020077 341 POP
020100 332 JC 020132
020103 016 MVI 005
020105 315 CALL 020150
020110 257 XRA
020111 032 LDAX
020112 315 CALL 020161
020115 023 INX
020116 005 DCR
020117 302 JNZ 020067
020122 001 LXI 020216
020125 367 RST
020126 015 DCR
020127 303 JMP 020004
020132 016 MVI 010
020134 315 CALL 020150
020137 005 DCR
020140 302 JNZ 020132
020143 001 LXI 020216
020146 367 RST
020147 307 RST
020150 076 MVI 040
020152 167 MOV
020153 043 INX
020154 015 DCR
020155 302 JNZ 020152
020160 311 RET
020161 365 PUSH
020162 027 RAL
020163 027 RAL
020164 027 RAL
020165 346 ANI 007
020167 366 ORI 060
020171 167 MOV
020172 043 INX
020173 361 POP
020174 365 PUSH
020175 037 RAR
020176 037 RAR
020177 037 RAR
020200 346 ANI 007
020202 366 ORI 060
020204 167 MOV
020205 043 INX
020206 361 POP
020207 346 ANI 007
020211 366 ORI 060
020213 167 MOV
020214 043 INX
020215 311 RET
```

```
020216 022 STAX
020217 114 MOV

RUN DMP
015100  104 040 002 007 001 105 040 003
015110  007 001 110 040 004 007 001 114
015120  040 005 007 001 115 040 006 007
015130  002 123 120 040 006 007 003 120
015140  123 127 040 006 007 001 044 040
015150  000 000 000 000 000 000 000 000
015160  377 377 377 377 377 377 377 377
015170  377 377 377 377 377 377 377 377
015200  000 000 000 000 000 000 000 000
015210  000 022 114 123 124 272 032 016
015220  000 073 033 022 115 101 107 000
015230  000 016 000 073 033 061 104 115
015240  120 000 040 020 115 101 107 272
015250  032 110 000 073 033 335 017 065
015260  115 101 107 377 377 377 060 000
015270  043 007 122
```

V. TEXT EDITOR

THE EDITOR IS USED TO CREATE AND MODIFY SOURCE PROGRAM FILES USING THE FOUR EDITING COMMANDS. THESE NOW INCLUDE THE ALTER COMMAND, USED TO MAKE CORRECTIONS WITHIN A LINE, ELIMINATING THE NEED TO REPLACE ALL MISTYPED LINES. THE INSERT, DELETE, AND REPLACE COMMAND ARE STILL INCLUDED AND HAVE BEEN IMPROVED TO EASE THE JOB OF MODIFYING A PROGRAM.

LOGICAL I/O DEVICES USED BY THE EDITOR

BEFORE RUNNING THE EDITOR, ALL LOGICAL I/O DEVICES USED BY IT SHOULD BE OPEN. THEY ARE LISTED BELOW ALONG WITH MODE INFORMATION NEEDED FOR PROPER OPERATION.

MAG - FILE I/O DEVICE NAME
- A - ASCII READ MODE SHOULD BE SPECIFIED
- T - TABS SHOULD NOT BE SPECIFIED
- E - CAN BE SPEIFIED IF THE USER WANTS A LISTING TABS WILL NOT BE EXPANDED

ALT - ALTER COMMAND I/O
- A - ASCII MODE SHOULD NOT BE SCECIFIED
- T - TABS SHOULD BE SPECIFIED
- E - ECHOING SHOULD NOT BE SPECIFIED

BUFFER AREA

THE FIRST 2 K OF MEMORY FOLLOWING THE EDITOR IS ALLOCATED AS A BUFFER TO STORE THE LINES OF TEXT THAT YOU ARE EDITING. IF THE SIZE OR LOCATION OF THIS BUFFER AREA NEED TO BE CHANGED, TWO ADDRESSES WITHIN THE EDITOR MUST BE CHANGED. STARTING AT LOCATION 3724Q IS THE ADDRESS OF THE BEGINNING OF THE BUFFER AND THE ADDRESS OF THE END OF THE BUFFER STARTS AT LOCATION 4311Q.

LOADING THE EDITOR

OPEN A LOGICAL DEVICE NAME TO THE AC OR TY DEPENDING ON WHETHER YOUR COPY OF THE EDITOR IS ON AUDIO CASSETTE OR PAPER TAPE. THE EDITORS FILE NAME IS EDT AND IS LOADED BY TYPING EDT,LOGICAL DEVICE NAME FOLLOWED BY A CARRIAGE RETURN. WHEN THE EDITOR IS FINISHED LOADING IT TYPES A PROMPT.
EXAMPLE.

```
?OPN TTY,TY
?EDT,TTY
(TURN ON PAPER TAPE READER)
START INPUT
*       (THE ASTRISK IT PRINTED WHENEVER THE EDITOR IS READY FOR A COMMAND)
```

ONCE THE EDITOR HAS BEEN LOADED INTO YOUR MACHINE, IT CAN BE EXECUTED AGAIN BY TYPING EDT CARRIAGE RETURN.
EXAMPLE.

```
?EDT
START INPUT
*
```

IF YOU WOULD LIKE TO CONTINUE EDITING LINES LEFT IN THE EDITORS BUFFER AREA WHEN YOU LAST EXITED THE EDITOR, USE THE R EXECUTION OPTION.
EXAMPLE.

```
?EDT(R)
*
```

START INPUT IS NOT PRINTED IN THIS CASE AND THE BUFFER IS NOT REINITIALIZED. THIS FEATURE IS ESPECIALLY USEFUL WHEN ASSEMBLING DIRECTLY FROM THE EDITORS BUFFER.

EDITOR COMMANDS

I [LINE NUMBER] INSERT COMMAND

THE INSERT COMMAND CAUSES THE EDITOR TO ENTER THE INSERT MODE AT THE LINE SPECIFIED. AFTER ALL LINES TO BE ENTERED HAVE BEEN TYPED, TYPE A CONTROL Z TO RETURN TO THE COMMAND LEVEL OF THE EDITOR. IF NO LINE NUMBER IS TYPED ALL LINES ARE INSERTED BEFORE THE FIRST LINE.

D RANGE DELETE COMMAND

DELETES ALL LINES IN THE SPECIFIED RANGE.

R RANGE REPLACE COMMAND

DELETES THE LINES IN THE RANGE AND ENTERS THE INSERT MODE

P [RANGE] PRINT COMMAND

PRINTS ALL LINES WITHIN THE RANGE OR ALL LINES IN THE BUFFER IF NO RANGE IS GIVEN.

F[STRING][(ESCAPE)[LINE NUMBER]] STRING SEARCH COMMAND

THE FIND COMMAND SEARCHES THE BUFFER AREA STARTING AT THE GIVEN LINE NUMBER, PRINTING THE FIRST LINE IT FINDS THE STRING IN. IF NO STRING IS GIVEN, THE STRING FROM THE LAST FIND COMMAND ISSUED IS SEARCHED FOR. IF NO LINE NUMBER IS TYPED, THE EDITOR STARTS SEARCHING AT THE CURRENT LINE PLUS 1 (IE .+1). THE ESCAPE IS OPTIONAL WHEN NOT TYPING A LINE NUMBER.

S SAVE FILE COMMAND

SAVE COMMAND PRINTS
FILENAME=
AN OPTIONAL 3 CHARACTER FILE NAME IS TYPED FOLLOWED BY A CARRIAGE RETURN. THE EDITOR RESPONDS BY TYPING
CHANGE SENSE SWITCH 15
AS SOON AS THIS MESSAGE HAS FINISHED PRINTING TURN ON THE DEVICE THE FILE IS TO BE DUMPED ON. CHANGE THE POINTION OF SENSE SWITCH 15 TO INDICATE THE DEVICE IS READY. WHEN ALL LINES HAVE BEEN DUMPED, THE EDITOR RETURNS TO THE MONITOR. WHEN A FILE NAME IS GIVEN, A HEADER BLOCK IS WRITTEN CONTAINING THE FILE NAME. IF NO FILE NAME WAS TYPED, NO HEADER BLOCK IS OUTPUT.

L LOAD FILE COMMAND

THE LOAD COMMAND PRINTS
FILENAME=

AN OPTIONAL 3 CHARACTER FILE NAME IS TYPED, FOLLOWED BY A CARRIAGE RETURN. IF A FILE NAME IS TYPED, A HEADER BLOCK CONTAINING THE PROPER FILE NAME IS SEARCHED FOR AND THE FILE FOLLOWING IT IS LOADED INTO THE BUFFER. IF NO FILE NAME IS TYPED, ALL LINES ARE LOADED UNTIL AN END OF FILE IS READ. THIS COMMAND READS FILES FROM SYMBOLIC DEVICE MAG.

$ BACKUP COMMAND

IF AN ESCAPE IS TYPED TO THE EDITOR A DOLLAR SIGN IS ECHOED AND AND THE CURRENT LINE MINUS 1(IE .-1) IS TYPED.

LINE FEED NEXT LINE COMMAND

IF A LINE FEED IS TYPED TO THE EDITOR THE CURRENT LINE PLUS 1(IE.+1) IS PRINTED.

E EXIT COMMAND

CAUSES THE EDITOR BUFFER READ POINTER TO BE RESET TO THE BEGINNING OF THE BUFFER, AND RETURNS TO THE MONITOR.

A LINE NUMBER ALTER COMMAND

THE ALTER COMMAND PUTS THE EDITOR INTO ALTER MODE, ALLOWING THE PROGRAMMER TO CHANGE LINES WITHOUT REPLACING THEM. THE FOLLOWING COMMAND CHARACTERS ARE RECOGNIZED BUT NOT ECHOED, AND ALL COMMANDS CAN BE PREFIXED BY A REPETITION FACTOR OF UP TO SIX DIGITS. THIS REPETITION FACTOR IS REFERED TO AS "N" IN THE FOLLOWING DESCRIPTION, AND IS ASSUMED TO BE ONE(1) IF NOT GIVEN.

ALTER MODE COMMANDS

D - DELETES THE NEXT N CHARACTERS IN LINE. IF THERE WERE N CHARACTERS IN THE LINE TO BE DELETED A SLASH(/) IS PRINTED FOLLOWED BY THE LAST CHARACTER THE COMMAND DELETED.
EXAMPLE.
THE CURRENT LINE IS
BLOB: MOV A,B
*A. GIVE ALTER COMMAND
YOU TYPE 3D
THE EDITOR TYPES /O BECAUSE THE O IS THE LAST CHARACTER DELETED IF 333D HAD BEEN TYPED ALL CHARACTERS IN THE LINE WOULD HAVE BEEN DELETED AND NOTHING WOULD BE TYPED BY THE EDITOR.

I - INSERTS ALL CHARACTERS TYPED AFTER THE I INTO THE LINE AT THE CURRENT CURRENT PLACE IN THE LINE. ALL CHARACTERS ARE ECHOED AND TYPING AN ESCAPE WILL GET YOU BACK TO THE ALTER MODE.

R - DELETES THE NEXT N CHARACTERS IN THE LINE AND ENTERS THE INSERT MODE.

S - TYPING A S FOLLOWED BY ANY CHARACTER WILL CAUSE A SEARCH FOR THE N'TH OCCURANCE OF THAT CHARACTER.

BLANK - TYPING A BLANK WILL CAUSE THE NEXT N CHARACTERS IN THE OLD LINE TO BE COPIED INTO THE NEW LINE AND BE PRINTED OUT.

CR - TYPING A CARRIAGE RETURN WILL PRINT OUT THE REST OF THE OLD LINE, INSERTING THE CHARACTERS AT THE SAME TIME INTO THE NEW LINE. THE OLD LINE LINE WILL THE BE REPLACED BY THE NEW ONE AND CONTROL IS RETURNED TO THE COMMAND LEVEL OF THE EDITOR.

Q - CAUSES CONTROL TO RETURN TO THE COMMAND LEVEL OF THE EDITOR WITHOUT REPLACING THE OLD LINE. THIS COMMAND IS USED TO ABORT AN ALTER DURING WHICH YOU MADE A BAD MISTAKE.

RANGE AND LINE NUMBER SPECIFICATIONS

WHEN A RANGE IS CALLED FOR BY AN INSTRUCTION, THE FOLLOWING SYNTAX IS REQUIRED.
LINE NUMBER,[LINE NUMBER]
THE SQUARE BRACKETS USED IN THE DESCRIPTION SIGNIFY THE SPECIFICATION WITHIN THEM IS OPTIONAL.

LINE NUMBERS

THREE TYPES OF INE NUMBERS ARE NOW RECOGNIZED BY THE EDITOR. THESE ARE AS FOLLOWS.

NUMBER(N)	THE N'TH LINE IN THE BUFFER.
.[+ OR - NUMBER]	RELATIVE ABOUT THE CURRENT LINE.
*[- NUMBER]	RELATIVE ABOUT THE LAST LINE IN THE BUFFER

EXAMPLE.

```
*P*
```
PRINTS THE LAST LINE IN THE BUFFER.
```
*P10
```
PRINTS THE TENTH LINE IN THE BUFFER.
```
*P.+10
```
ASSUMING THIS LINE WAS EXECUTED AFTER THE P10 COMMAND, LINE 20 WOULD BE PRINTED.

APPENDIX A

ABSOLUTE LOAD TAPE FORMAT

BEGIN/NAME RECORD

BYTE 1	170 OCTAL	BEGIN SYNC
BYTES 2&3	0-177777 OCTAL	BEGIN EXECUTION ADDRESS
BYTES 4-6	NAME	PROGRAM NAME FOR MONITOR TASK
BYTES 7-	COMMENTS	END STATEMENT DOCUMENTATION (I.E.PROGRAM NO,REVISION,AND DATE)

PROGRAM LOAD RECORD

BYTE 1	74 OCTAL	LOAD SYNC BYTE
BYTE 2	0-377 OCTAL	NO. OF LOAD BYTES(N)
BYTE 3	L.S. BYTE	LOAD ADDRESS
BYTE 4	M.S. BYTE	" "
	DATA BYTES	
BYTE N+5	*	CHECKSUM BYTE

* THE CHECKSUM IS GENERATED BY ADDING W/O CARRY ALL BYTES EXCEPT THE FIRST TWO.

END-OF-FILE RECORD

BYTE 1	32 OCTAL	PAPER/AUDIO CASSETTE EOF

APPENDIX B

ASSEMBLY MEMORY MAP

MEMORY BLOCKS	BOUNDARIES
	TOP OF MEMORY
USER SPACE	FIRST WORD OF PROGRAM STORAGE
SYMBOL TABLE (CHAR. LENGTH +4 BYTES/SYMBOL)	SYMBOLS ARE ADDED TO HIGHER MEMORY
TEMPORARY ASSEMBLER VARIABLES	FIRST WORD OF ASSEMBLER RAM STORAGE
ASSEMBLER PROGRAM STORAGE (VERSION 1)	FIRST WORD OF ASSEMBLER (3700Q)
MONITOR TABLES & HANDLERS	
RESTART TRAPS	BOTTOM OF MEMORY

RECOMENDED MEMORY LAYOUT WITH EDITOR AND ASSEMBLER

MEMORY BLOCKS	BOUNDARIES
	TOP OF MEMORY
EDITOR BUFFER AREA	FIRST WORD OF BUFFER
USER AREA	
SYMBOL TABLE	
ASSEMBLER (VERSION 2)	FIRST WORD OF ASSEMBLER (7350Q)
	DEFAULT START OF EDIT BUFFER
EDITOR	EDITOR STARTS AT 3700Q
MONITOR	BOTTOM OF MEMORY

APPENDIX C

ERROR CODES ARE THE FIRST TWO CHARACTERS ON THE LINE FOLLOWING OCCURANCE OF AN ERROR. THE CHARACTERS REPLACE THE INPUT CHARACTERS THAT ARE NORMALLY ECHOED ON A TTY OR COMPTER TERMINAL.

ASSEMBLER ERRORS

I# ILLEGAL OPERAND
UNDEFINED BYTE SYMBOL
STRING NOT ALLOWED
NAME VALUE MUST BE DEFINED
ORR OR ORG MUST BE DEFINED VALUE

O# MEMORY OVERFLOW
PROGRAM SPACE NOT LARGE ENOUGH

Q# SYMBOL TABLE OVERFLOW
PROGRAM STORAGE SHOULD BEGIN AT HIGHER MEMORY ADDRESS

S# SYMBOL UNDEFINED

D# DOUBLE DEFINED SYMBOL

N# NO NAME DEFINED

B# NO ORIGIN SPECIFIED

MONITOR ERRORS

F# I/O TABLE FULL
USE THE CLS UTILITY PROGRAM TO CLOSE AN UNUSED SYMBOLIC NAME FREEING SPACE IN THE TABLE TO OPEN THE NAME YOU NEED.

H# HARDWARE DEVICE UNDEFINED
AN ATTEMPT WAS MADE TO OPEN A SYMBOLIC NAME TO A NONEXISTANT HARDWARE DEVICE.

N# I/O DEVICE NAME NOT OPEN
BEFORE TRYING TO READ FROM A SYMBOLIC DEVICE, THAT DEVICE SHOULD BE OPENED. SEE OPEN COMMAND UNDER MONITOR UTILITY PROGRAMS.

V# ILLEGAL OPERATION
A ILLEGAL OPERATION CODE WAS GIVEN IN A MONITOR CONTROL BLOCK.

S# SYNTAX ERROR IN COMMAND LINE

A# ATTEMPT TO STORE OVER MONITOR
NO PROGRAM CAN LOAD BEFORE 3700Q.

U# PROGRAM NAME NOT IN PTL

P# PROGRAM TABLE(PTL) FULL
USE THE CLR UTILITY TO CLEAR THE NAME OF A PROGRAM THAT IN'NT NEEDED ANY LONGER.

D# NAME ALREADY IN PTL
USE THE CLR UTILITY TO REMOVE THE PROGRAM NAME FROM THE PTL.

M# MEMORY MALFUNCTION - MEMORY NOT WORKING OR NONEXISTANT
AFTER STORING INTO MEMORY THE STORED BYTE DID NOT COMPARE EXACTLY WITH THE VALUE STORED. THIS IS CHECK WHEN THE MONITOR LOADS A PROGRAM.

L# LOAD ERROR. A CHECKSUM ERROR OCCURED WHILE LOADING A PROGRAM.

APPENDIX D.

THE I/O SPECIFICATIONS GIVEN BELOW SHOW ONLY THE CONTROL I/O PORT NUMBER BECAUSE THE MONITOR ALWAYS ASSUMES THE DATA PORT IS THE NEXT HIGHEST NUMBERED PORT FROM THE CONTROL. IF THE USER NEEDS TO CHANGE ANY OF THE PORT ASSIGNMENTS THE I/O TABLE FORMAT IS GIVEN BELOW.

```
        DB      "PD"    ;2 CHARACTER NAME OF PHYSICAL DEVICE
        DW      TTHND   ;ADDRESS OF HANDLER USED FOR THIS DEVICE
        DB      0       ;READ - WRITE CONTROL PORT NUMBER
        DB      0       ;ECHO CONTROL PORT NUMBER
        DW      TABLOC  ;ADDRESS OF WORD CONTAINING TAB COLUMN INFORMATION
                        ;FOR DEVICE.
```

THIS TABLE STARTS AT LOCATION 105Q.

PHYSICAL DEVICE NAMES

TY - TELETYPE
READ,WRITE - CONTROL PORT 0
ECHO - CONTROL PORT 0

AC - AUDIO CASSETTE
READ,WRITE - CONTROL PORT 6
ECHO - CONTROL PORT 0

EB - EDIT BUFFER
READS IN THE ASCII MODE FROM THE EDITORS BUFFER. IF A READ OCCURS AND THERE ARE NO MORE LINES LEFT IN THE BUFFER THE HANDLER EXITS TO THE MONITOR.
ECHO - CONTROL PORT 0

ADDING HANDLERS TO THE MONITOR

AFTER THE ENTRIES FOR THE FOUR DEVICES LISTED ABOVE THERE IS SPACE FOR TWO MORE DEVICE NAMES. THE NEW DEVICE NAME, ITS HANDLERS START LOCATION, AND I/O PORTS TO USE SHOULD BE PATCHED IN AT THE FIRST AVAILABLE LOCATION IN THIS TABLE.

WHEN THE MONITOR BRANCHES TO A HANDLER THE B&C REGISTER PAIR CONTAIN THE ADDRESS OF THE READ WRITE I/O PORT IN THE TABLE SHOWN ABOVE. THE D&E REGISTER PAIR CONTAIN THE ADDRESS OF THE FOLLOWING TABLE.

```
        DB      0       ;I/O MODE CONTROL BYTE (SET IN OPEN)
        THE REST OF THE TABLE CONTAINS A COPY OF THE
        MONITOR CONTROL BLOCK FROM THE CALL.
```

APPENDIX E.

WHEN THE ALTAIR IS FIRST TURNED ON, THERE IS RANDOM GARBAGE IN ITS MEMORY. THE MONITOR IS SUPPLIED ON A PAPER TAPE OR AUDIO CASSETTE. SOMEHOW THE INFORMATION ON THE PAPER TAPE OR CASSETTE MUST BE TRANSFERED INTO THE COMPUTER. PROGRAMS THAT PERFORM THIS TYPE OF INFORMATION TRANSFER ARE CALLED LOADERS.

SINCE INITIALLY THERE IS NOTHING OF USE IN MEMORY, YOU MUST TOGGLE IN, USING THE SWITCHES ON THE FRONT PANEL, A 20 INSTRUCTION BOOTSTRAP LOADER. THIS LOADER WILL THEN LOAD THE MONITOR. SO, TO LOAD THE MONITOR, FOLLOW THESE STEPS:

1) TURN THE ALTAIR ON.

2) RAISE THE STOP SWITCH AND RESET SWITCH SIMULTANEOUSLY.

3) TURN YOUR TERMINAL (USUALLY A TELETYPE) TO LINE.

BECAUSE THE INSTRUCTIONS MUST BE TOGGLED IN VIA THE SWITCHES ON THE FRONT PANEL, IT IS RATHER INCONVENIENT TO SPECIFY THE POSITIONS OF EACH SWITCH AS UP OR DOWN. THEREFORE, THE SWITCHES ARE ARRANGED IN GROUPS OF 3 AS INDICATED BY THE BROKEN LINES BELOW SWITCHES 0-15. TO SPECIFY THE POSITIONS OF EACH SWITCH WE USE THE NUMBERS 0-7 AS SHOWN BELOW:

	SWITCHES		
LEFTMOST	MIDDLE	RIGHTMOST	NUMBER
DOWN	DOWN	DOWN	0
DOWN	DOWN	UP	1
DOWN	UP	DOWN	2
DOWN	UP	UP	3
UP	DOWN	DOWN	4
UP	DOWN	UP	5
UP	UP	DOWN	6
UP	UP	UP	7

SO, TO PUT THE OCTAL NUMBER 315 IN SWITCHES 0-7, THE SWITCHES WOULD HAVE THE FOLLOWING POSITIONS:

			SWITCHES					
7	6	/	5	4	3	/	2	1	0
UP	UP	/	DOWN	DOWN	UP	/	UP	DOWN	UP
	3			1				5	

NOTE THAT SWITCHES 8-15 WERE NOT USED. SWITCHES 0-7 CORRESPOND TO THE SWITCHES LABELED DATA ON THE FRONT PANEL. A MEMORY ADDRESS USES ALL 16 SWITCHES.

THE BOOTSTRAP LOADER IS THE FOLLOWING PROGRAM:

THE FOLLOWING BOOTSTRAP LOADER IS FOR USERS LOADING FROM PAPER TAPE AND NOT USING A REV 0 SERIAL I/O BOARD.

ADDRESS/DATA

```
000/  041
001/  175
002/  017
003/  061
004/  022
005/  000
006/  333
007/  000
010/  017
011/  330
012/  333
013/  001
014/  275
015/  310
016/  055
017/  167
020/  300
021/  351
022/  003
023/  000
```

THE FOLLOWING 21 BYTE BOOTSTRAP LOADER IS FOR USERS LOADING FROM PAPER TAPE AND USING A REV 0 SERIAL I/O BOARD ON WHICH THE UPDATE CHANGING THE FLAG BITS HAS NOT BEEN MADE. IF THE UPDATE HAS BEEN MADE, USE THE ABOVE BOOTSTRAP LOADER.

```
000/    041
001/    175
002/    017
003/    061
004/    023
005/    000
006/    333
007/    000
010/    346
011/    040
012/    310
013/    333
014/    001
015/    275
016/    310
017/    055
020/    167
021/    300
022/    351
023/    003
024/    000
```

THE FOLLOWING BOOTSTRAP LOADER IS FOR USERS WITH THE MONITOR SUPPLIED ON AN AUDIO CASSETTE.

```
000/    041
001/    175
002/    017
003/    061
004/    022
005/    000
006/    333
007/    006
010/    017
011/    330
012/    333
013/    007
014/    275
015/    310
016/    055
017/    167
020/    300
021/    351
022/    003
023/    000
```

SO, TO LOAD THE BOOTSTRAP LOADER:

4) PUT SWITCHES 0-15 IN THE DOWN POSITION.

5) RAISE EXAMINE.

6) PUT 041 (THE DATA FOR ADDRESS 000) IN SWITCHES 0-7.

7) RAISE DEPOSIT.

8) PUT THE DATA FOR THE NEXT ADDRESS IN SWITCHES 0-7 (FOR ADDRESS 001 THIS IS 175)

9) DEPRESS DEPOSIT NEXT.

10) REPEAT STEPS 8-9 UNTIL THE ENTIRE LOADER IS TOGGLED IN.

NEXT CHECK THAT THE BOOTSTRAP LOADER IS IN CORRECTLY:

11) PUT SWITCHES 0-15 IN THE DOWN POSITION.

12) RAISE EXAMINE.

13) CHECK THAT LIGHTS D0-D7 CORRESPOND WITH THE DATA THAT SHOULD BE IN ADDRESS 000. A LIGHT 'ON' MEANS THE SWITCH WAS UP; A LIGHT 'OFF' MEANS THE SWITCH WAS DOWN. SO FOR ADDRESS 000, LIGHTS D1-D4 AND D6-D7 SHOULD BE OFF, AND LIGHTS D0 AND D5

SHOULD BE ON.

IF THE CORRECT VALUE IS THERE, GO TO STEP 16.
IF THE VALUE IS WRONG, GO TO THE NEXT STEP, 14.

14) PUT THE CORRECT VALUE IN SWITCHES 0-7.

15) RAISE DEPOSIT.

16) DEPRESS EXAMINE NEXT.

17) IF YOU HAVE NOT CHECKED THE ENTIRE BOOTSTRAP LOADER, REPEAT STEPS 13-16 UNTIL YOU HAVE.

18) IF YOU FOUND A MISTAKE, GO BACK TO STEP 11 AND CHECK THE BOOTSTRAP LOADER AGAIN.

19) PUT THE TAPE OF THE MONITOR INTO THE TAPE READER. BE SURE THE TAPE IS POSITIONED AT THE BEGINNING OF THE LEADER. THE LEADER IS THE SECTION OF TAPE AT THE BEGINNING WITH 6 OUT OF THE 8 HOLES PUNCHED.
IF YOU ARE LOADING FROM AUDIO CASSETTE, PUT THE CASSETTE IN THE RECORDER. BE SURE THE TAPE IS FULLY REWOUND.

20) PUT SWITCHES 0-15 IN THE DOWN POSITION.

21) RAISE EXAMINE.

22) IF YOU HAVE CONNECTED YOUR TERMINAL TO A REV 0 SERIAL I/O BOARD ON WHICH THE UPDATE CHANGING THE FLAG BITS HAS NOT BEEN MADE, RAISE SWITCH 14.
IF YOU ARE LOADING FROM AUDIO CASSETTE, RAISE SWITCH 15.
IF YOU HAVE THE REV 0 SERIAL I/O BOARD CONNECTED TO YOUR TERMINAL AND ARE LOADING FROM CASSETTE, THEN BOTH SWITCHES 14 AND 15 SHOULD BE UP.
IF YOUR TERMINAL IS CONNECTED TO A PIO BOARD AND YOU ARE LOADING FROM AUDIO CASSETTE RAISE SENSE SWITCHES 13 AND 15.

23) TURN ON THE TAPE READER AND THEN DEPRESS RUN. BE SURE RUN IS DEPRESSED WHILE THE READER IS STILL ON THE LEADER. DO NOT DEPRESS RUN BEFORE TURNING ON THE READER, SINCE THIS MAY CAUSE THE TAPE TO BE READ INCORRECTLY.
IF YOU ARE LOADING FROM CASSETTE, THEN TURN THE CASSETTE RECORDER TO PLAY. WAIT 15 SECONDS AND THEN DEPRESS RUN.

24) IF A "C" OR AN "O" IS PRINTED ON THE TERMINAL AS THE TAPE READS IN, THE TAPE HAS BEEN MISREAD, AND YOU MUST START OVER AT STEP 11. MAKE SURE YOU CHECK THE BOOTSTRAP LOADER CAREFULLY AS IT MAY HAVE BEEN WRITTEN OVER DURING THE LOAD PROCESS.
STEP 4.

25) WHEN THE TAPE FINISHES READING, THE MONITOR SHOULD START UP AND PRINT THE NORMAL PROMPT - ? .
IF YOU ARE LOADING FROM CASSETTE, BE SURE TO REWIND THE TAPE AND TURN THE RECORDER TO STOP.

APPENDIX F.

AUDIO CASSETTE USERS

THE FOLLOWING TABLE SHOWS THE ORDER AND LENGTH OF FILES ON THE CASSETTE OF PACKAGE 1.

PROGRAM NAME	TIME FROM START OF TAPE (IN SECONDS)
MONITOR	13 - 105
ASM	120 - 225
EDT	240 - 305
AM2	320 - 425

WHEN YOU ARE ABOUT TO RECORD A NEW FILE ON A CASSETTE POSITION THE CASSETTE AFTER THE LAST FILE. WHEN USING EITHER THE EDITOR OR ASSEMBLER TO DUMP OUT A FILE, TURN THE RECORDER TO RECORD FOR A FEW SECONDS BEFORE FLIPPING SENSE SWITCH 15. A GAP OF THIS TYPE SHOULD BE INSERTED BETWEEN ALL FILES ON A CASETTE.

NOTES ON PACKAGE 1

THE PRESENT MONITOR ASSUMES THAT ALL I/O DEVICES ARE CONNECTED TO THE SAME TYPE OF I/O BOARD. IF YOU HAVE A TERMINAL HOOKED UP VIA A PIO CARD, AND WANT TO SAVE PROGRAMS ON AUDIO CASSETTE, SEE THE LAST SECTION IN THIS APPENDIX.

ASCII LINE INPUT

THE FOLLOWING DESCRIBES THE ACTION TAKEN FOR VARIOUS SPECIAL CHARACTERS.

CR - ENDS A LINE. THE MONITOR RETURNS TO THE CALLING PROGRAM WHEN TYPED. IT IS NOT COUNTED IN THE LINE LENGTH RETURNED. A LINE FEED IS ALSO WRITTEN OUT IF INPUT IS BEING ECHOED.
LF - ENDS A LINE. ONLY A LINE FEED IS ECHOED. SEE ABOVE.
ESCAPE - ENDS A LINE. $ IS ECHOED. SEE ABOVE.
OCTAL 0 - IGNORED
CONTROL A - RUBS OUT COMPLETE LINE TYPED.
RUBOUT - BACKSPACES ONE CHARACTER FOR EACH ONE TYPED.
CONTROL Z - END OF FILE, BRANCHES TO ADDRESS GIVEN IN CONTROL BLOCK.

ASSEMBLER VERSIONS

TWO COPIES OF THE ASSEMBLER ARE SUPPLIED IN PACKAGE 1. VERSION 1 LOADS STARTING AT LOCATION 3700Q AND THE SYMBOL TABLE STARTS AT 116350. VERSION 2 LOADS AT LOCATION 73500 AND THE SYMBOL TABLE STARTS AT 153050.

RUNNING VERSION 1 GIVES YOU MAXIMUM MEMORY FOR PROGRAM SPACE BUT DOES NOT ALLOW THE EDITOR TO BE RESIDENT AT THE SAME TIME. VERSION 2 LETS YOU LOAD THE EDITOR BETWEEN THE MONITOR AND THE ASSEMBLER ALLOWING YOU TO ASSEMBLE DIRECTLY FROM THE EDITORS BUFFER USING THE EDIT BUFFER READ FEATURE. IF THIS SETUP IS TO BE USED THE EDITOR MUST BE PATCHED TO MOVE THE BUFFER AREA. THE BUFFER AREA IS ASSUMED TO BE IMMEDIATELY AFTER THE EDITOR IF NOT PATCHED, AND WOULD DESTROY THE ASSEMBLER IF NOT MOVED. SEE APPENDIX B FOR RECOMMENDED MEMORY LAYOUT.

ABSOLUTE FILE NAMES

VERSION 1 - ASM
VERSION 2 - AM2

PIO BOARD MODIFICATION

IF YOU NEED TO MODIFY YOUR PIO BOARD SO THAT THE CONTROL PORT STATUS IS THE SAME AS A NEW SERIAL I/O BOARD, FOLLOW THIS PROCEDURE.

1) CUT LANDS FROM IC-D PIN 1 TO IC - L PIN 14, AND FROM IC-D PIN 4 TO IC-L PIN 12.

2) CONNECT A JUMPER FROM IC-H PIN 23 TO IC L PIN 14.

3) CONNECT A JUMPER FROM IC-G PIN 23 TO IC-L PIN 12.