27
1/26/77
25 cap.
JTS

UCID- 17375

Lawrence Livermore Laboratory

AMPS AUTOMATED MESSAGE PROCESSING SYSTEM, LER 76-101701A

W. Benge/J. Spann

December 3, 1976

This is an informal report intended primarily for internal or limited external distribution. The opinions and conclusions stated are those of the author and may or may not be those of the laboratory.

Prepared for U.S. Energy Research & Development Administration under contract No. W-7405-Eng-48.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

REFER QUESTIONS TO: W. Bengé J. Spann	LAWRENCE LIVERMORE LABORATORY	ELECTRONICS ENGINEERING REPORT LIVERMORE, CALIFORNIA	UNIVERSITY OF CALIFORNIA	LER 76-1017-01
W. Bengé & J. Spann	AMPS AUTOMATED MESSAGE PROCESSING SYSTEM			DATE 1/28/77 PAGE 1 OF 39 REV. A*

TABLE OF CONTENTS

<u>CONTENTS</u>	<u>Page</u>
1.0 INTRODUCTION	2
1.1 WHY AMPS	2
1.2 SYSTEM OPERATION AND HARDWARE WITHOUT AMPS	2
1.3 AMP SYSTEM OPERATION AND HARDWARE	2
2.0 SOFTWARE	3
3.0 USERS MANUAL	5
4.0 HARDWARE	8
APPENDIX A PROGRAM LISTINGS	10
APPENDIX B PROGRAM FLOWCHARTS	29

NOTICE
This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Energy Research and Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

ABSTRACT

This report describes the operation and use of the LLL AUTOMATED MESSAGE PROCESSING SYSTEM (AMPS), which is used for off-line preparation of messages for SACNET (Secure Automatic Communications Network). The preparation of teletype messages before AMPS required typing on a teletype to create a paper tape in ASCII code compatible with SACNET. AMPS uses a MCS-80 microprocessor, CRT with keyboard, and a teletype to create the SACNET messages. Through processor control and CRT edit capabilities through-put of the system was increased about 50%.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

1.0 INTRODUCTION

The Automatic Message Processing System (AMPS) has become a part of the SACNET system for producing input in text form.

SACNET (Secure Automatic Communications Network) is a system designed for the electrical transmission of messages and computer produced data, both classified and unclassified. The system provides for a rapid exchange of information between ERDA offices and contractors.

1.1 Why AMPS

The preparation of teletype messages before AMPS required typing on a teletypewriter keyboard to create a paper tape in ASCII code compatible with SACNET. This is a relatively slow process. Although message preparation is only one of several functions performed by the SACNET communications operators, it involves more than 50% of the total work load.

1.2 System Operation and Hardware Without AMPS

System Hardware:

Model 35 Teletypewriter

System Operation:

A message is typed by the originator. It is then delivered to the communications center where it is retyped to produce a paper tape for transmission over the SACNET lines. Refer to Figure 1 for System Flow.

1.3 AMP System Operation and Hardware

System Hardware:

CRT Display Terminal and Keyboard (Lear Siegler Inc. Model ADM-1)

MCS-80 Microprocessor (Developed by LLL)

High Speed Paper Tape Reader (Remex)

Model-35 Teletypewriter (Teletype Corp.)

System Operation:

A message is typed by the originator. It is then delivered to the communications center where it is retyped on the Lear Siegler ADM-1 Terminal as input. The speed at which a message can be retyped is greatly improved. Most of the speed increase is due to the extremely powerful editing functions of the ADM-1 Terminal.

The ADM-1 Terminal message is then sent, via MCS-80 control, to the ASR-35 Teletype which produces a paper tape for transmission on SACNET. Another feature of the AMPS system is the ability to re-read a previously punched tape for purposes of editing the message. There are also standard message formats called SXR's. These SXR's are on paper tape and can be read into the ADM-1 via operator control. As a preformatted line is read into and displayed on the ADM-1, the operator simply fills in the appropriate information and then requests the next preformatted line to be displayed. Refer to Figure 2 for AMPS System Flow.

2.0 SOFTWARE

The software for AMPS was written in a new high level language called PLM. PLM is a high level language designed specifically for the Intel 8-bit microprocessors. Detailed information on the PLM language may be obtained from the PLM User's Manual. See the reference section of this report.

The following is a brief description of the AMPS software and its sub-routines. For greater detail of program operation and flow, refer to Appendix A and Appendix B.

2.1 MAIN

This is the main control section of the software. Transfer of control from this routine is determined by the input device selection made by the operator at the ADM-1 Display Terminal.

The operator selects one of two input devices. Device 1 is the High Speed Paper Tape Reader (HSR). Device 3 is the Display Terminal Keyboard (Key). Once the device selection is made, transfer of control is then given to the proper subroutine which handles input from the selected device.

2.2 CLIN

This routine clears the input buffer to zeros.

2.3 CLOUT

This routine clears the output buffer to zeros.

2.4 READ

This routine reads data from the input device determined by MAIN.

2.5 SEND

This routine sends a string of characters to a pre-selected device when it is called. SEND is called with arguments in the calling statement. The first argument provides the address of the data to be sent and the second argument provides the length of the data.

2.6 HSR

This routine reads one character from the High Speed Paper Tape Reader.

2.7 PUNCH

This routine controls the output of data to the teletype punch. The routine gets its punch data from the output buffer. There are appropriate counters and flags in other routines to tell this routine when it is finished or has data to punch.

2.8 KENTRY

This routine allows the operator to enter data from the Display Terminal Keyboard. After initial control flags and counters are set, most of the control will come from the Edit and Monitor routine (MEDIT).

2.9 READIN

This routine inputs data from the High Speed Paper Tape Reader. Its function is to control the calls to HSR, to load the input buffers and character counters, and to call a routine for displaying the input buffer content (BUFFDIS). The call to BUFFDIS is made upon receipt of a form-feed character or rubout on the input paper tape.

2.10 BUFFDIS

This routine displays data on the ADM-1 Display Terminal.

2.11 MEDIT

This routine is used to receive and echo characters sent by the Display Terminal Keyboard. When the routine is not monitoring keyboard activity, it is attempting to punch data. If there is no data for punching, it monitors keyboard activity. If the routine receives a send command character from the keyboard (which is an indication that a control command character will be coming next), all punch action stops while the routine waits for one of the following control characters:

C = Send displayed text to the ASR 35 tape punch. Do not punch end of page format.

T = Read a line from HSR and display it.

E = Send displayed text to the punch and punch end of tape format.

P = Send displayed text to the punch and punch end of page format.

Each of the control functions are handled by the MEDIT routine. When the operator sends a page of information, MEDIT handles the page format requirement of the SACNET message and outputs the page. MEDIT also handles the End Of Message format required by SACNET.

3.0 USERS MANUAL

The ADM-1 Display Terminal is the control point for the AMPS operator. The Display Terminal Keyboard is used to enter system control commands and has a number of very powerful text editing commands.

When the system is ready to accept a message, instructions for the operator will be printed on the display.

3.1 Select Input Device HSR=1, KEY=3

The operator depresses one of the suggested numeric keys to assign the input device for the controlling software. Selecting Device 1 causes paper tape to be read from the High Speed Reader. Selection Device 3 causes the Display Terminal screen to clear and the system to wait for text to be entered from the keyboard. The data input is displayed on the Display Terminal.

3.2 Control Functions

Control functions are sent to the microprocessor and are acted on by the microprocessor.

3.2.1 These functions are performed by depressing a key or a sequence of keys on the Display Terminal Keyboard.

<u>Key Sequence</u>	<u>Function</u>
Send C	Send displayed text to the punch. Do not punch end of page format. This instruction acts as a continue.
Send T	Read a line from the High Speed Paper Tape Reader and display it.
Send P	Send a full page of text to the output buffers and begin punching a paper tape of the page.
Send E	Output the remaining text. Message ended.

3.3 Edit Functions

Edit functions are not sent to the microprocessor and are an internal operation of the Display Terminal.

These functions are performed by depressing a key or a sequence of keys on the Display Terminal Keyboard.

<u>Key Sequence</u>	<u>Function</u>
ESC Q	Causes the character under the cursor and all characters to the right of the cursor to be shifted one position to the right.
ESC W	Causes the character under the cursor to be deleted. All characters to the right of the cursor on the line shift left to fill the space left by the deletion.
ESC T	Deletes the remainder of the line from cursor to end of line.
ESC R	Deletes the entire line on which the cursor is located. All lines below the cursor move up one line.
ESC E	Causes the line on which the cursor is located and all lines below the cursor to shift down one line. Used to insert a new line.

3.3.2 Only one page can be output at a time. If the operator is finished editing the second page, and attempts to output that page before the first page is finished punching, no other commands will be accepted from the keyboard until the first page is finished and the second page begins to punch.

3.3.3 When a Send E command has been processed by the system and the last page has finished punching, leader for the next job is automatically punched onto the tape and a Select Input Device message is displayed.

3.4 HSR Input

A message in the SACNET Format or SXR Format can be read by the High Speed Paper Tape Reader of AMPS.

HSR Procedures:

3.4.1 Operator mounts the leader of the paper tape in the tape reader.

3.4.2 The operator then selects input device HSR by depressing the number 1 on the Display Terminal Keyboard.

- 3.4.3 The first page of text will be displayed on the Display Terminal and the tape reader will stop. Editing may now be performed on the displayed text. After editing is complete, the displayed text is sent to the punch using the SEND P or SEND C command. As the previously displayed text is being punched, the next page of text is automatically read and displayed. Editing may now be performed on the second page of text during the punching of the first page. Should the operator complete the editing of the second page before the punching of the first page is complete and then attempt to send the displayed page to the punch, nothing will happen. Due to the memory size, one page must complete punching before the next page can be accepted by the punch routine. At this point the program is in an idle state, awaiting a completion flag from the punch routine. When the punch completion flag is received, the routine sends the displayed page to the punch, and reads a page from the HSR. It continues in this mode until the last page is sent to the punch, using the Send E command. After punching of last page, the screen is cleared and the operator directives are displayed for a fresh start.

3.5 Key Input

The Key Input allows the operator to create a message from the keyboard of the Display terminal.

Key Input Procedures:

- 3.5.1 The operator selects the key input device by depressing the number 3 on the keyboard.
- 3.5.2 The display clears and is now ready to perform any requested edit or control function.
- 3.5.3 Operation in the key-mode is now identical to the HSR mode with two exceptions. The first exception is that when a page is sent to the punch, instead of input from the paper tape reader, the screen is cleared and the system waits for keyboard text to be input. The other exception is the Send T function. When performing the Send T function on the keyboard, one line of prepunched text will be read from an SXR tape. Information needed by the instruction line of the SXR is to be filled in from the keyboard. Then the operator performs the Send T function and again another line of text is read from the SXR and displayed. The operation continues in this fashion until the Send E function is sent.

4.0 HARDWARE

The major hardware items are listed in Section 1.3.

4.1 AMPS SYSTEM HARDWARE PRINT LIST

The following drawing packages cover the AMPS system design.

4.1.1 Major Item Prints

LEA75-1952-01-SA: MCS-80 Amps System

LEA75-1952-05-AC: Module Map, AMPS

LEA75-1952-07-SA: Display and Data Terminal Interface to
AMPS Computer

4.1.2 Module Prints

LEA72-9068-23: 8080 Microprocessor Board

LEA72-9068-24: 8080 Control Board

LEA68-9068-98: 1k RAM Board

LEA72-9068-31: 1k RAM Board

LEA68-9068-96: 512 PROM Board

LEA72-9068-30: 512 PROM Board

LEA72-9068-25: 8080 I/O Device Decoder

LEA68-9068-99: UART

LEA72-9068-28: Remex Paper Tape Reader

LEA68-9068-02: Breadboard

LEA75-2111-01: Schmidt Trigger Inverters

LEA72-9068-22: 3-channel Tri-state Buffer

FIGURE 1
BLOCK DIAGRAM OF AMPS SYSTEM

APPENDIX A

AMPS

PROGRAM LISTINGS

8080 PLM1 VERS 3.0

```

00001 1  /* AMPS -- AUTOMATIC MESSAGE PROCESSING SYSTEM */
00002 1  /* WRITTEN BY WALTER L. BERGE */
00003 1  /* MARCH 20, 1975 */
00004 1  /* PROGRAM RUNS ON AN MCS8080 MICRO PROCESSOR */
00005 1  /* PROGRAM ORIGIN IS PAGE 11 */
00006 1  10000Q;
00007 1  DECLARE FIRSTRUN BYTE;
00008 1  DECLARE (TA,ESCW) BYTE;
00009 1  TA=324Q;
00010 1  ESCW=327Q;
00011 1  DECLARE CTRLV BYTE;
00012 1  CTRLV=333Q;
00013 1  DECLARE BELL BYTE;
00014 1  DECLARE (MORE, LASTPG, KEYRUN, PUNSTAT, NEWCOUNT) BYTE;
00015 1  DECLARE (FXXC, XLAG, TRUEC, LINES) BYTE;
00016 1  DECLARE (MYCHAR, KCHAR, RUBFND, NIL) BYTE;
00017 1  DECLARE (LEAV, EAV, TIMEF, LAG, SAVED, XCC, LINEFLAG) BYTE;
00018 1  DECLARE (RET, LF, FORM, RUBOUT) BYTE;
00019 1  DECLARE IBUFF(1800) BYTE;
00020 1  DECLARE OBUFF(1800) BYTE;
00021 1  DECLARE (J, LCOUNT, PCOUNT, INSERTS, ESC) BYTE;
00022 1  DECLARE (INS, DEL, PAG, EOD, OUTA, YPOS, XPOS, M) BYTE;
00023 1  DECLARE (RBIT, SBIT, RPORT, SPORT) BYTE;
00024 1  DECLARE (LASTCHAR, TT, P, I, K, L, T, LASTK) ADDRESS;
00025 1  DECLARE (NEWK, STK) ADDRESS;
00026 1  DECLARE MESS1 DATA('SELECT INPUT DEVICE, HSR=1,');
00027 1  DECLARE MESS2 DATA('KEY=3, 215Q, 212Q');
00028 1  DECLARE CRLF DATA(215Q, 212Q);
00029 1  DECLARE RSOFF DATA('RUN SW OFF ');
00030 1  DECLARE INST DATA(233Q, 305Q);
00031 1  DECLARE DLT DATA(233Q, 322Q);
00032 1  DECLARE CURPUSS DATA(233Q, 275Q);
00033 1  DECLARE LINEPR0T DATA(233Q, 266Q);
00034 1  DECLARE CLEAR DATA(233Q, 252Q);
00035 1  DECLARE (CONTP, SXPAGE) BYTE;
00036 1  CONTP=303Q;
00037 1  RET=215Q; LF=12Q; FORM=14Q; RUBOUT=377Q; ESC=233Q; INS=311Q; NIL=0;
00038 1  DEL=304Q; PAG=320Q; EOD=305Q; T=0; INSERTS=0; K=0; LASTK=0; PCOUNT=0;
00039 1  BELL=207Q;
00040 1  /* */
00041 1  /* */
00042 1  /* */
00043 1  /* */
00044 1  CLIN:PROCEDURE;
00045 2  DO I=0 TO 1800;
00046 2  IBUFF(I)=0;
00047 3  END;
00048 2  RETURN;
00049 2  END CLIN;
00050 1  /* */
00051 1  /* */
00052 1  /* */
00053 1  /* */
00054 1  CL0UT:PROCEDURE;
00055 2  DO I=0 TO 1800;
00056 2  OBUFF(I)=0;
00057 3  END;
00058 2  RETURN;

```

```

00059 2      END CLOUT;
00060 1      /*
00061 1      /*
00062 1      /*
00063 1      /*
00064 1      /*
00065 1      /*
00066 1      /*
00067 1      /*
00068 1      PUNCH: PROCEDURE;
00069 2      IF PUNSTAT=0 THEN GO TO RETPUN;
00070 2      XLAG=INPUT(3);
00071 2      FXXC=SHR(XLAG,2);
00072 2      IF NOT CARRY THEN GO TO RETPUN;
00073 2      OUTPUT(2)=GBUFF(P);
00074 2      P=P+1;
00075 2      IF P>11 THEN GO TO RESET; /* GBUFF ENPTY */
00076 2      GO TO RETPUN;
00077 2      RESET: PUNSTAT=0;
00078 2      RETPUN: RETURN;
00079 2      END PUNCH;
00080 1      /*
00081 1      /*
00082 1      /*
00083 1      /*
00084 1      READ: PROCEDURE;
00085 2      DECLARE (XFLAG,CC) BYTE;
00086 2      A1A: XFLAG=INPUT(3);
00087 2      IF RBIT=5 THEN GO TO C1;
00088 2      IF RBIT=6 THEN GO TO C2;
00089 2      IF RBIT=7 THEN GO TO C3;
00090 2      CC=SHR(XFLAG,8);
00091 2      GO TO C9;
00092 2      C1: CC=SHR(XFLAG,5);
00093 2      GO TO C9;
00094 2      C2: CC=SHR(XFLAG,6);
00095 2      GO TO C9;
00096 2      C3: CC=SHR(XFLAG,7);
00097 2      C9: IF NOT CARRY THEN GO TO A1A;
00098 2      IF RPORT=6 THEN GO TO C10;
00099 2      MYCHAR=INPUT(1);
00100 2      GO TO C20;
00101 2      C10: MYCHAR=INPUT(6);
00102 2      C20: RETURN;
00103 2      END READ;
00104 1      /*
00105 1      /*
00106 1      /*
00107 1      /*
00108 1      SEND: PROCEDURE(LIST,XSIZE);
00109 2      DECLARE (XSIZE,CH,FLAG) BYTE;
00110 2      DECLARE LIST(84) ADDRESS;
00111 2      DECLARE (C,B,ICHAR BASED LIST) BYTE;
00112 2      DECLARE RETN LABEL;
00113 2      /* READ THE FLAG. WHEN READY - SEND A CHARACTER DO TO FINI */
00114 2      DO C=0 TO XSIZE-1;
00115 2      ICHAR(C)=ICHAR(C) AND 177Q;
00116 2      ICHAR(C)=ICHAR(C) OR 200Q;
00117 2      A1A: FLAG=INPUT(3);
00118 2      IF SBIT=2 THEN GO TO D1; /* GOING OUT TTY */

```



```

00119 3      CH=SHR(FLAG,6); /* SHIFT IN FLAG */
00120 3      IF NOT CARRY THEN GO TO A1A; /* NOT READY */
00121 3      OUTPUT(6)=ICHAR(C); /* READY SEND CHAR */
00122 3      IF ICHAR(C)=212Q THEN GO TO RETN; /*ALL DONE*/
00123 3      GO TO TR;
00124 3      D1:CH=SHR(FLAG,2);
00125 3      IF NOT CARRY THEN GO TO A1A;
00126 3      OUTPUT(2)=ICHAR(C);
00127 3      TR:END; /* END OF DO LOOP */
00128 2      RETN:RETURN;
00129 2      END SEND;
00130 1      /*
00131 1      /*
00132 1      /*
00133 1      /*
00134 1      HSR:PROCEDURE; /* ROUTINE TO READ HIGH SPEED READER */
00135 2      DECLARE (FCHAR,X) BYTE;
00136 2      OUTPUT(5)=0; /* POKE HSR */
00137 2      RFLAG:FCHAR=INPUT(3); /* READ FLAG */
00138 2      X=SHR(FCHAR,3); /* CHECK 3RD BIT */
00139 2      IF NOT CARRY THEN GO TO RFLAG; /* READER NOT READY */
00140 2      KCHAR=INPUT(5); /* READ A CHARACTER */
00141 2      RETURN;
00142 2      END HSR; /* END OF HSR ROUTINE */
00143 1      /*
00144 1      /*
00145 1      XXFORM:PROCEDURE;
00146 2      CALL SEND(.ESC,LENGTH(ESC));
00147 2      CALL SEND(.ESCW,LENGTH(ESCW));
00148 2      CALL SEND(.ESC,LENGTH(ESC));
00149 2      CALL SEND(.ESCW,LENGTH(ESCW));
00150 2      XFORM:CALL HSR; /* READ A CHARACTER FROM HSR */
00151 2      IF KCHAR=377Q THEN GO TO XFORM;
00152 2      IF KCHAR=0 THEN GO TO XFORM;
00153 2      YFORM:CALL SEND(.KCHAR,LENGTH(KCHAR));
00154 2      CALL HSR; /* READ ANOTHER CHARACTER */
00155 2      IF KCHAR=377Q THEN GO TO YYFORM;
00156 2      IF KCHAR=0 THEN GO TO YYFORM;
00157 2      GO TO YFORM;
00158 2      YYFORM:RETURN;
00159 2      END XXFORM;
00160 1      /*
00161 1      /*
00162 1      MEDIT:PROCEDURE; /* ROUTINE TO MONITOR OPERATOR EDITING */
00163 2      RUBFND=0;
00164 2      T=0;
00165 2      RBIT=5;SBIT=6;RPORT=6;SPORT=6;
00166 2      BACKUP:XLAG=INPUT(3);
00167 2      FXXC=SHR(XLAG,5);
00168 2      IF CARRY THEN GO TO BACKUP1;
00169 2      CALL PUNCH;
00170 2      GO TO BACKUP;
00171 2      BACKUP1:MYCHAR=INPUT(6);
00172 2      IF MYCHAR=ESC THEN GO TO FIND; /* IF MYCHAR = */
00173 2      /* ESC. THE OPERATOR HAS */
00174 2      /* HIT THE SEND KEY. NOW */
00175 2      /* GO FIND WHAT FUNCTION */
00176 2      /* KEY SHE HITS NEXT. */
00177 2      ECHO:CALL SEND(.MYCHAR,LENGTH(MYCHAR)); /*SEND CHAR TO */
00178 2      /* DISPLAY */

```

```

00179 2      GO TO BACKUP; /* READ ANOTHER CHARACTER */
00180      FIND: CALL READ; /* CATCH EXTRA ZERO BEING SENT BY SEND KEY */
00181 2      CALL READ; /* READ THE EDIT FUNCTION CHAR */
00182 2      IF MYCHAR=TA THEN GO TO XTape;
00183 2      IF MYCHAR=CONTP THEN GO TO SSPAGE;
00184 2      IF MYCHAR=PAG THEN GO TO SPAGE; /* WANTS TO SEND A PAGE */
00185 2      IF MYCHAR=EOG THEN GO TO SETFND; /* SEND LAST PAGE */
00186 2      GO TO BACKUP; /* UNKNOWN FUNCTION REQUEST */
00187 2      /* IF LAST PAGE FLAG IS NOT ON GO DO NORMAL INSERT AT INSERT1 */
00188      SETFND: RUBFND=1;
00189      GO TO SPAGE;
00190 2      SSPAGE: SXPAGE=1;
00191 2      SPAGE: NEWCOUNT=LCOUNT-1;
00192 2      YPUN: IF PUNSTAT<>1 THEN GO TO XPUN;
00193 2      CALL PUNCH;
00194 2      GO TO YPUN;
00195 2      XPUN: IF RUBFND=1 THEN GO TO ZPUN;
00196 2      ZPUN: PUNSTAT=1;
00197 2      YPOS=237Q; /* INITIALIZE Y LINE */
00198 2      XPOS=157Q; /* SET X END OF LINE */
00199 2      ASTER: YPOS=YPOS+1; /* BUMP Y FOR PROPER LINE POSITION */
00200 2      CALL SEND(CURPUSS,LENGTH(CURPUSS)); /* SEND */
00201 2      /* LEADING MESSAGE TO POSITION CUR */
00202 2      CALL SEND(YPOS,LENGTH(YPOS)); /* SEND Y POSITION */
00203 2      CALL SEND(XPOS,LENGTH(XPOS)); /* SEND X POSITION */
00204 2      CALL SEND(LINEPROT,LENGTH(LINEPROT)); /* SEND */
00205 2      /* A LINE PROTECTED */
00206 2      OREAD: CALL READ; /* READ A CHARACTER BEING OUTPUTTED */
00207 2      IF MYCHAR=CTRLV THEN GO TO ASTER;
00208 2      IF MYCHAR=RE THEN GO TO FINLINE; /* FINISHED LINE */
00209 2      OBUFF(T)=MYCHAR; /* STORE CHARACTER IN OBUFF */
00210 2      T=T+1;
00211 2      GO TO OREAD; /* READ ANOTHER CHARACTER */
00212 2      FINLINE: T=T-1;
00213 2      FINX: IF OBUFF(T)<>240Q THEN GO TO PACKED; /* WORD */
00214 2      /* IS PACKED GO ADD A RETURN */
00215 2      T=T-1; /* DECREMENT OBUFF INDEX */
00216 2      GO TO FINX; /* CONTINUE TO PACK */
00217 2      PACKED: T=T+1; /* INDEX TO END OF WORD */
00218 2      OBUFF(T)=RET; /* LOAD RETURN AT END OF LINE */
00219 2      T=T+1;
00220 2      OBUFF(T)=RET; /* PUT IN SECOND RETURN */
00221 2      T=T+1;
00222 2      OBUFF(T)=LF; /* PUT IN LF */
00223 2      T=T+1;
00224 2      GO TO ASTER1;
00225 2      ASTER: IF RUBFND=1 THEN GO TO FND1;
00226 2      IF SXPAGE=1 THEN GO TO EDATA; /* CONTINUE */
00227 2      OBUFF(T)=LF;
00228 2      T=T+1;
00229 2      OBUFF(T)=LF;
00230 2      T=T+1;
00231 2      OBUFF(T)=LF;
00232 2      T=T+1;
00233 2      OBUFF(T)=FORM;
00234 2      T=T+1;
00235 2      DO I=0 TO 9;
00236 2      OBUFF(T)=RET;
00237 2      T=T+1;
00238 3      END;

```

```

00239 2      DO I=0 TO 9;
00240 2          OBUFF(T)=377Q;
00241 3          T=T+1;
00242 3      END;
00243 2      GO TO EDATA;
00244 2      /* END OF A MESSAGE FORMAT */
00245 2      FND1:T=T-3; /* BACKUP */
00246 2      OBUFF(T)=377Q;
00247 2      T=T+1;
00248 2      OBUFF(T)=RET;
00249 2      T=T+1;
00250 2      OBUFF(T)=RET;
00251 2      T=T+1;
00252 2      DO I=0 TO 7;
00253 2          OBUFF(T)=LF; /* 8 LINE FEEDS */
00254 3          T=T+1;
00255 3      END;
00256 2      DO I=0 TO 3;
00257 2          OBUFF(T)=316Q; /* 4 NS */
00258 3          T=T+1;
00259 3      END;
00260 2      DO I=0 TO 64;
00261 2          OBUFF(T)=377Q; /* 12 RUBOUTS */
00262 3          T=T+1;
00263 3      END;
00264 2      EDATA:P=0;
00265 2      SXPAGE=0; /* RESET CONTINUE */
00266 2      TT=T-1; /* TT IS END OF OUTPUT BUFFER */
00267 2      PUNAG:CALL PUNCH;
00268 2      IF RUBFND=0 THEN GO TO YCLEAR;
00269 2      IF PUNSTAT=0 THEN GO TO YCLEAR;
00270 2      GO TO PUNAG;
00271 2      XTAPE:CALL XXFORM;
00272 2      GO TO BACKUP;
00273 2      YCLEAR:CALL SEND(.CLEAR,LENGTH(CLEAR));
00274 2      CALL READ; /* DUMMY READ */
00275 2      SPUN:RETURN;
00276 2      END MEDIT; /* END OF MEDIT ROUTINE */
00277 1      /*
00278 1      /*
00279 1      /*
00280 1      /*
00281 1      /* ENTRY IS FROM TUBE KEYBOARD ONLY */
00282 1      KENTRY:PROCEDURE;
00283 2          XXXX:CALL SEND(.CLEAR,LENGTH(CLEAR));
00284 2          CALL MEDIT;
00285 2          IF RUBFND=1 THEN GO TO RETENT;
00286 2          GO TO XXXX;
00287 2          RETENT:RETURN;
00288 2          END KENTRY;
00289 1      /*
00290 1      /*
00291 1      /*
00292 1      /*
00293 1      BUFFDIS:PROCEDURE; /* ROUTINE TO DISPLAY INPUT BUFFER */
00294 2          SBIT=6;
00295 2          SPORT=6; /* SET OUTPUT BIT AND PORT VALUE */
00296 2          BACK2:K=0;
00297 2          BACK:IF IBUFF(K)=RUBOUT THEN GO TO ENDBUFF;
00298 2          IF IBUFF(K)=RET THEN GO TO SRET;

```

```

00299 2      /* IS CHARACTER AN INDENTED ONE */
00300 2      IF IBUFF(K)=274Q THEN IBUFF(K)=240Q;
00301 2      OUTA=IBUFF(K);
00302 2      CALL SEND(.OUTA,LENGTH(OUTA)); /* SEND TO TUBE */
00303 2  BUMPER:K=K+1;
00304 2      GO TO BACK;
00305 2      SRET:OUTA=IBUFF(K); /* FOUND A RETURN */
00306 2      CALL SEND(.OUTA,LENGTH(OUTA)); /* SEND IT */
00307 2      CALL SEND(.LF,LENGTH(LF));
00308 2      K=K+1;
00309 2      GO TO BACK;
00310 2  ENDBUFF:CALL MEDIT;
00311 2      RETURN;
00312 2  END BUFFDIS;

00313 1  /*
00314 1  /*
00315 1  /*
00316 1  /*
00317 1  READIN:PROCEDURE; /* PROCEDURE - LOAD INPUT BUFFER FROM HSR */
00318 2      CREAD:I=0;
00319 2      AREAD:CALL HSR; /* READ HIGH SPEED READER */
00320 2      BREAD:IF KCHAR=RET THEN GO TO RETSTOR;
00321 2      IF KCHAR=12Q THEN GO TO AREAD;
00322 2      IF KCHAR=FORM THEN GO TO DREAD;
00323 2      IF KCHAR=RUBOUT THEN GO TO RUBSTOR;
00324 2      IBUFF(I)=KCHAR; /* STORE KCHAR IN INPUT BUFFER */
00325 2      I=I+1; /* BUMP INDEX */
00326 2      GO TO AREAD; /* READ ANOTHER CHARACTER */
00327 2  RETSTOR:IBUFF(I)=RET;
00328 2      CALL HSR; /* READ SECOND RETURN */
00329 2      I=I+1;
00330 2      GO TO AREAD;
00331 2  DREAD:DO K=0 TO 19;
00332 2      CALL HSR;
00333 2      END;
00334 2      IBUFF(I)=RUBOUT;
00335 2      CALL BUFFDIS;
00336 2      GO TO CREAD;
00337 2  RUBSTOR:IBUFF(I)=RET;
00338 2      I=I+1;
00339 2      IBUFF(I)=RUBOUT;
00340 2      CALL BUFFDIS; /* ROUTINE TO DISPLAY DATA */
00341 2      IF RUBEND=1 THEN GO TO RETX;
00342 2      CALL MEDIT;
00343 2  RETX:RETURN;
00344 2      END READIN; /* END OF ROUTINE */

00345 1  /*
00346 1  /*
00347 1  /*
00348 1  /*
00349 1  /*
00350 1  /*
00351 1  /*
00352 1  /*
00353 1  BEGIN:SBIT=2;
00354 1      SPORT=2;
00355 1      FIRSTRUN=1;
00356 1      PUNSTAT=0;
00357 1      PCOUNT=0;
00358 1  SXPAGE=0;

```

```

00359 1 RUBFND=0;
00360 1 KEYRUN=0;
00361 1 LASTPG=0; /* INITIALIZE LAST PAGE FLAG TO ZERO */
00362 1 MORE=0; /* RESET BUFFER EMPTY ,FULL FLAG */
00363 1 LCOUNT=0;
00364 1 DO I=0 TO 50;
00365 1 CALL SEND(.NIL,LENGTH(NIL));
00366 2 END;
00367 1 /* SEND LEADER */
00368 1 CALL CLIN; /* CLEAR INPUT BUFFER */
00369 1 CALL CLOUT; /* CLEAR OUTPUT BUFFER */
00370 1 SBIT=6;
00371 1 SPORT=6;
00372 1 RBIT=5;
00373 1 RPORT=6;
00374 1 CALL SEND(.CLEAR,LENGTH(CLEAR));
00375 1 CALL SEND(.MESS1,LENGTH(MESS1));
00376 1 CALL SEND(.MESS2,LENGTH(MESS2));
00377 1 CALL READ;
00378 1 MYCHAR=MYCHAR-260Q;
00379 1 IF MYCHAR=1 THEN GO TO HR0UT;
00380 1 /* INPUT IS FROM KEYBOARD */
00381 1 CALL KENTRY;
00382 1 GO TO BEGIN;
00383 1 /* INPUT DEVICE IS OCR */
00384 1 /* SEE IF RUN SWITCH IS ON */
00385 1 /* HIGH SPEED READER ROUTINE AND ENTRY */
00386 1 HR0UT: CALL SEND(.CLEAR,LENGTH(CLEAR));
00387 1 CALL READIN;
00388 1 GO TO BEGIN;
00389 1 EOF
NO PROGRAM ERRORS

```

```

S00295 HR0UT
S00293 260
S00291 50
S00289 BEGIN
S00288 RETX
S00285 19
S00283 RUBSTOR
S00281 DREAD
S00278 RETSTOR
S00276 BREAD
S00275 AREAD
S00274 CREAD
S00272 READIN
S00271 BUMPER
S00269 274
S00268 SRET
S00266 ENDBUFF
S00264 BACK
S00263 BACK2
S00261 BUFFDIS
S00260 RETENT
S00258 XXXX
S00256 KENTRY
S00255 SPUN
S00253 YCLEAR
S00251 PUNAG
S00249 64

```

S00247 316
S00239 9
S00237 EDATA
S00235 FND1
S00233 PACKED
S00231 240
S00230 FINX
S00229 FINLINE
S00227 ASTER
S00225 OREAD
S00224 ASTER1
S00223 157
S00222 237
S00221 ZPUN
S00219 XPUN
S00217 YPUN
S00216 SETFND
S00214 SPAGE
S00212 SSPAGE
S00210 XTAPE
S00208 ECHO
S00207 FIND
S00205 BACKUP
S00203 BACKUP
S00201 MEDIT
S00199 YYFORM
S00197 YFORM
S00194 XFORM
S00192 XXFORM
S00190 RFLAG
S00189 X
S00188 FCHAR
S00186 HSR
S00184 TR
S00181 D1
S00179 A1A
S00178 200
S00177 177
S00174 RETN
S00173 ICHAR
S00172 B
S00171 C
S00170 FLAG
S00169 CH
S00167 XSIZE
S00166 LIST
S00165 SEND
S00164 C20
S00163 C10
S00160 C9
S00159 8
S00158 C3
S00156 7
S00155 C2
S00153 6
S00152 C1
S00150 5
S00149 A1A
S00148 CC
S00147 XFLAG

S00145 READ
S00144 RESET
S00142 .
S00140 2
S00139 3
S00138 RETPUN
S00135 PUNCH
S00131 CLOUT
S00129 1800
S00126 CLIN
S00125 207
S00124 320
S00123 304
S00122 0
S00121 311
S00120 377
S00119 14
S00118 12
S00117 303
S00116 SXPAGE
S00115 CONTP
S00114 252
S00113 CLEAR
S00111 266
S00110 LINEPR0T
S00108 275
S00107 CURPUSS
S00105 322
S00104 DLT
S00102 305
S00101 233
S00100 INST
S00098 'RUN SW OFF'
S00097 RS0FF
S00095 CRLF
S00093 212
S00092 215
S00091 'KEY=3'
S00090 MESS2
S00088 'SELECT INPUT DEVICE, HSR=1,'
S00087 MESS1
S00085 STK
S00084 NEWK
S00083 LASTK
S00082 T
S00081 L
S00080 K
S00079 I
S00078 P
S00077 TT
S00076 LASTCHAR
S00075 SPORT
S00074 RPORT
S00073 SBIT
S00072 RBIT
S00071 M
S00070 XPOS
S00069 YPOS
S00068 OUTA
S00067 EOD

S00066 PAG
S00065 DEL
S00064 INS
S00063 ESC
S00062 INSERTS
S00061 PCOUNT
S00060 LCOUNT
S00059 J
S00058 OBUFF
S00057 IBUFF
S00056 RUBOUT
S00055 FORM
S00054 LF
S00053 RET
S00052 LINEFLAG
S00051 XCC
S00050 SVEL
S00049 LAG
S00048 TIMEF
S00047 EAV
S00046 LEAV
S00045 NIL
S00044 RUBFND
S00043 KCHAR
S00042 MYCHAR
S00041 LINES
S00040 TRUEC
S00039 XLAG
S00038 FXXC
S00037 NEWCOLNT
S00036 PUNSTAT
S00035 KEYRUN
S00034 LASTPG
S00033 MORE
S00032 BELL
S00031 333
S00030 CTRLV
S00029 327
S00028 324
S00027 ESCW
S00026 TA
S00025 FIRSTRUN
S00024 10000
S00022 DEC
S00021 DOUBLE
S00020 MOVE
S00019 LAST
S00018 LENGTH
S00017 OUTPUT
S00016 INPUT
S00015 LOW
S00014 HIGH
S00013 TIME
S00012 SCR
S00011 SCL
S00010 SHR
S00009 SHL
S00008 ROR
S00007 ROL
S00006 STACKPTR

S00005 MEMORY
S00004 PARITY
S00003 SIGN
S00002 ZERO
S00001 CARRY

PAGE 11

8080 PLM2 VERS 3.0

1=0J0003Q	7=010003Q	10=010010Q	11=010013Q	13=010016Q	26=010021Q
27=010054Q	28=010064Q	29=010066Q	30=010101Q	31=010103Q	32=010105Q
33=010107Q	34=010111Q	35=010113Q	37=010120Q	38=010146Q	39=010217Q
40=010223Q	44=010230Q	45=010233Q	46=010262Q	47=010273Q	48=010314Q
49=010315Q	55=010325Q	56=010344Q	57=010355Q	58=010376Q	59=010377Q
69=010405Q	70=010413Q	71=010412Q	72=010421Q	73=010431Q	74=010441Q
75=010450Q	76=010474Q	78=010501Q	79=010502Q	86=010504Q	87=010510Q
88=010523Q	89=010531Q	90=010537Q	91=010547Q	92=010554Q	93=010567Q
94=010574Q	95=010607Q	96=010614Q	97=010627Q	98=010637Q	99=010650Q
100=010647Q	101=010656Q	102=010660Q	103=010665Q	109=010676Q	114=010702Q
115=010716Q	116=010743Q	117=010775Q	118=011001Q	119=011020Q	120=011030Q
121=011040Q	122=011047Q	123=011074Q	124=011103Q	125=011105Q	126=011113Q
127=011122Q	128=011133Q	129=011134Q	137=011137Q	138=011141Q	139=011153Q
140=011163Q	141=011162Q	142=011167Q	146=011172Q	147=011177Q	148=011207Q
149=011217Q	150=011227Q	151=011232Q	152=011243Q	153=011251Q	154=011256Q
155=011261Q	156=011272Q	157=011300Q	158=011301Q	164=011306Q	165=011316Q
166=011330Q	167=011335Q	168=011352Q	169=011362Q	170=011365Q	171=011367Q
172=011372Q	173=011404Q	177=011401Q	178=011411Q	179=011414Q	180=011417Q
181=011422Q	182=011427Q	183=011435Q	184=011450Q	185=011463Q	186=011476Q
189=011503Q	190=011506Q	191=011511Q	192=011516Q	193=011535Q	194=011540Q
195=011544Q	196=011552Q	197=011547Q	198=011553Q	199=011560Q	200=011570Q
201=011577Q	202=011602Q	203=011607Q	204=011617Q	205=011627Q	206=011632Q
207=011637Q	208=011645Q	209=011657Q	210=011667Q	211=011702Q	212=011707Q
213=011714Q	214=011741Q	215=011736Q	216=011743Q	217=011750Q	218=011755Q
219=011772Q	220=012005Q	221=012022Q	222=012035Q	223=012052Q	224=012065Q
225=012072Q	226=012105Q	227=012115Q	228=012124Q	229=012137Q	230=012154Q
231=012167Q	232=012204Q	233=012217Q	234=012234Q	235=012247Q	236=012277Q
237=012311Q	238=012324Q	239=012346Q	240=012372Q	241=012404Q	242=012412Q
243=012434Q	244=012437Q	245=012453Q	246=012457Q	247=012470Q	248=012476Q
249=012513Q	250=012526Q	251=012543Q	252=012556Q	253=012606Q	254=012620Q
255=012633Q	256=012655Q	257=012701Q	258=012713Q	259=012721Q	260=012743Q
261=012767Q	262=013001Q	263=013007Q	264=013031Q	265=013041Q	266=013045Q
267=013052Q	268=013057Q	269=013073Q	270=013103Q	271=013106Q	272=013111Q
273=013121Q	274=013124Q	275=013125Q	283=013135Q	284=013140Q	285=013144Q
286=013152Q	287=013153Q	295=013160Q	296=013164Q	297=013173Q	298=013220Q
299=013242Q	300=013237Q	301=013272Q	302=013306Q	303=013322Q	304=013327Q
305=013334Q	306=013347Q	307=013363Q	308=013373Q	309=013400Q	310=013405Q
311=013410Q	312=013411Q	319=01342'Q	320=013424Q	321=013436Q	322=013446Q
323=013455Q	324=013466Q	325=013476Q	326=013511Q	327=013516Q	328=013531Q
329=013542Q	330=013547Q	331=013554Q	332=013602Q	333=013605Q	334=013625Q
335=013637Q	336=013650Q	337=013653Q	338=013666Q	339=013701Q	340=013716Q
341=013727Q	342=013741Q	343=013742Q	354=013747Q	355=013753Q	356=013760Q
357=013765Q	358=013772Q	359=013776Q	360=014003Q	361=014007Q	362=014012Q
363=014017Q	364=014024Q	365=014051Q	366=014061Q	367=014101Q	368=014104Q
369=014107Q	371=014113Q	372=014117Q	373=014123Q	374=014127Q	375=014137Q
376=014147Q	377=014157Q	378=014162Q	379=014165Q	380=014176Q	381=014173Q
382=014176Q	383=014201Q	386=014211Q	387=014214Q	388=014217Q	
S00001 16372Q	S00002 16372Q	S00003 16372Q	S00004 16372Q		
S00005 26000Q	S00006 16372Q	S00023 16372Q	S00025 16372Q		
S00026 16373Q	S00027 16374Q	S00030 16375Q	S00032 16376Q		
S00033 16377Q	S00034 16400Q	S00035 16401Q	S00036 16402Q		
S00037 16403Q	S00038 16404Q	S00039 16405Q	S00040 16406Q		
S00041 16407Q	S00042 16410Q	S00043 16411Q	S00044 16412Q		
S00045 16413Q	S00046 16414Q	S00047 16415Q	S00048 16416Q		
S00049 16417Q	S00050 16420Q	S00051 16421Q	S00052 16422Q		
S00053 16423Q	S00054 16424Q	S00055 16425Q	S00056 16426Q		
S00057 16427Q	S00058 22037Q	S00059 25447Q	S00060 25450Q		
S00061 25451Q	S00062 25452Q	S00063 25453Q	S00064 25454Q		
S00065 25455Q	S00066 25456Q	S00067 25457Q	S00068 25460Q		

```

S00069 25461Q S00070 25462Q S00071 25463Q S00072 25464Q
S00073 25465Q S00074 25466Q S00075 25467Q S00076 25470Q
S00077 25472Q S00078 25474Q S00079 25476Q S00080 25500Q
S00081 25502Q S00082 25504Q S00083 25506Q S00084 25510Q
S00085 25512Q S00115 25514Q S00116 25515Q S00147 25516Q
S00148 25517Q S00166 25520Q S00167 25771Q S00169 25772Q
S00170 25773Q S00171 25774Q S00172 25775Q S00188 25776Q
S00189 25777Q
STACK SIZE = 12 BYTES
010000Q LXI SP 16372Q LXI H 16373Q MOV MI 324Q INR L
010011Q MOV MI 327Q INR L MOV MI 333Q JMP 10113Q
010021Q 123Q 105Q 114Q 105Q 103Q 124Q 040Q 111Q 116Q 120Q 125Q 124Q 040Q 104Q
010037Q 105Q 126Q 111Q 103Q 105Q 054Q 040Q 110Q 123Q 122Q 075Q 061Q 054Q
010054Q 04CQ 113Q 105Q 131Q 075Q 063Q 215Q 212Q
010064Q 212Q 212Q
010066Q 122Q 125Q 116Q 040Q 123Q 127Q 040Q 117Q 106Q 106Q 040Q
010101Q 233Q 305Q
010103Q 233Q 322Q
010105Q 233Q 275Q
010107Q 233Q 266Q
010111Q 233Q 252Q
010113Q LXI H 25514Q MOV MI 303Q LXI H 16423Q MOV MI 215Q
010125Q INR L MOV MI 12Q INR L MOV MI 14Q
010133Q INR L MOV MI 377Q LXI H 25453Q MOV MI 233Q
010143Q INR L MOV MI 311Q LXI H 16413Q MOV MI 0Q
010153Q LXI H 25455Q MOV MI 304Q INR L MOV MI 320Q
010163Q INR L MOV MI 305Q MOV LI 104Q MOV MI 0Q
010172Q INX H MOV MI 0Q MOV LI 52Q MOV MI 0Q
010201Q MOV LI 100Q MOV MI 0Q INX H MOV MI 0Q
010210Q MOV LI 106Q MOV MI 0Q INX H MOV MI 0Q
010217Q MOV LI 51Q MOV MI 0Q LXI H 16376Q MOV MI 207Q
010230Q JMP 13742Q LXI H 25476Q MOV MI 0Q INX H
010241Q MOV MI 0Q MOV AI 10Q MOV BI 7Q LXI H 25476Q
010252Q SUB M INR L MOV CA MOV AB
010256Q SBC M JC 10314Q DCR L MOV CM
010264Q INR L MOV BM MOV H LXI H 16427Q DAD B
010272Q MOV MI 0Q LXI H 25476Q MOV CM INR L
010301Q MOV BM LXI H 1Q DAD B SHLD 25476Q
010311Q JMP 10243Q RET LXI H 25476Q MOV MI 0Q
010322Q INX H MOV MI 0Q MOV AI 10Q MOV BI 7Q
010331Q LXI H 25476Q SUB M INR L MOV CA
010337Q MOV AB SBC M JC 10376Q DCR L
010345Q MOV CM INR L MOV BM LXI H 22037Q
010353Q DAD B MOV MI 0Q LXI H 25476Q MOV CM
010362Q INR L MOV BM LXI H 1Q DAD B
010370Q SHLD 25476Q JMP 10325Q RET LXI H 16402Q
010402Q MOV AM SUB I 0Q JZ 10501Q IN
010412Q MOV LI 5Q MOV MA RAR
010417Q ORA A RAR MOV MA
010423Q JNC 10501Q LXI H 25474Q MOV CM INR L
010433Q MOV BM LXI H 22037Q DAD B MOV AM
010441Q OUT 2Q LHL INX H SHLD 25474Q
010452Q LXI H 25472Q MOV AM MOV BM
010460Q INR L SUB M MOV CA
010464Q MOV AB SBC M JC 10474Q JMP 10501Q
010474Q LXI H 16402Q MOV MI 0Q RET IN 3Q
010504Q LXI H 25516Q MOV MA MOV LI 64Q MOV AM
010513Q SUB I JZ 10554Q MOV AM SUB I 6Q
010523Q JZ 10574Q MOV AM SUB I 7Q JZ 10614Q
010534Q MOV CI 10Q MOV LI 116Q MOV AM ORA A

```

010542Q	RAR		DCR C		JNZ	10541Q	INR L	
010550Q	MOV MA		JMP	10631Q	MOV CI	5Q	MOV LI	116Q
010560Q	MOV AM		ORA A		RAR		DCR C	
010564Q	JNZ	10561Q	INR L		MOV MA		JMP	10631Q
010574Q	MOV CI	6Q	MOV LI	116Q	MOV AM		ORA A	
010602Q	RAR		DCR C		JNZ	10601Q	INR L	
010610Q	MOV MA		JMP	10631Q	MOV CI	7Q	MOV LI	116Q
010620Q	MOV AM		ORA A		RAR		DCR C	
010624Q	JNZ	10621Q	INR L		MOV MA		JNC	10502Q
010634Q	LXI H	25466Q	MOV AM		SUB I	6Q	JZ	10656Q
010645Q	IN	1Q	LXI H	16410Q	MOV MA		JMP	10664Q
010656Q	IN	6Q	LXI H	16410Q	MOV MA		RET	
010665Q	LXI H	25520Q	MOV MC		INX H		MOV MB	
010673Q	MOV LI	371Q	MOV ME		MOV LI	374Q	MOV MI	0Q
010702Q	LXI H	25771Q	MOV CM		DCR C		MOV AC	
010710Q	MOV LI	374Q	SUB M		JC	11133Q	MOV CM	
010717Q	MOV BI	0Q	LHLD	25520Q	DAD B		PUSH H	
010726Q	LXI H	25774Q	MOV CM		MOV BI	0Q	LHLD H	25520Q
010737Q	DAD B		MOV AM		ANA I	177Q	POP H	
010744Q	MOV MA		LXI H	25774Q	MOV CM		MOV BI	0Q
010753Q	LHLD	25520Q	DAD B		PUSH H		LXI H	25774Q
010763Q	MOV CM		MOV BI	0Q	LHLD	25520Q	DAD B	
010772Q	MOV AM		ORA I	2C0Q	POP H		MOV MA	
010777Q	IN	3Q	LXI H	25773Q	MOV MA		MOV LI	65Q
011007Q	MOV AM		SUB I	2Q	JZ	11074Q	MOV CI	6Q
011017Q	MOV LI	373Q	MOV AM		ORA A		RAR	
011024Q	DCR C		JNZ	11022Q	DCR L		MOV MA	
011032Q	JNC	10777Q	MOV LI	374Q	MOV CM		MOV BI	0Q
011042Q	LHLD	25520Q	DAD B		MOV AM		OUT	6Q
011051Q	LXI H	25774Q	MOV CM		MOV BI	0Q	LHLD	25520Q
011062Q	DAD B		MOV AM		SUB I	212Q	JZ	11133Q
011071Q	JMP	11124Q	MOV LI	373Q	MOV AM		ORA A	
011100Q	RAR		ORA A		RAR		DCR L	
011104Q	MOV MA		JNC	10777Q	MOV LI	374Q	MOV CM	
011113Q	MOV BI	0Q	LHLD	25520Q	DAD B		MOV AM	
011122Q	OUT	2Q	LXI H	25774Q	INR M		JNZ	10702Q
011133Q	RET		XRA A		OUT	5Q	IN	3Q
011141Q	LXI H	25776Q	MOV MA		ORA A		RAR	
011147Q	ORA A		RAR		ORA A		RAR	
011153Q	INR L		MOV MA		JNC	11137Q	IN	5Q
011162Q	LXI H	16411Q	MOV MA		RET		LXI B	25453Q
011172Q	MOV EI	1Q	CALL	10665Q	LXI B	16374Q	MOV EI	1Q
011204Q	CALL	10665Q	LXI B	25453Q	MOV EI	1Q	CALL	10665Q
011217Q	LXI B	16374Q	MOV EI	1Q	CALL	10665Q	CALL	11134Q
011232Q	MOV AM		SUB I	377Q	JZ	11227Q	MOV AM	
011241Q	SUB I	0Q	JZ	11227Q	LXI B	16411Q	MOV EI	1Q
011253Q	CALL	10665Q	CALL	11134Q	MOV AM		SUB I	377Q
011264Q	JZ	11300Q	MOV AM		SUB I	0Q	JZ	11300Q
011275Q	JMP	11246Q	RET		LXI H	16412Q	MOV MI	0Q
011306Q	LXI H	25504Q	MOV MI	0Q	INX H		MOV MI	0Q
011316Q	MOV LI	64Q	MOV MI	5Q	INR L		MOV MI	6Q
011325Q	INR L		MOV MI	6Q	INR L		MOV MI	6Q
011333Q	IN	3Q	LXI H	16405Q	MOV MA		MOV CI	5Q
011343Q	MOV AM		ORA A		RAR		DCR C	
011347Q	JNZ	11344Q	DCR L		MOV MA		JC	11365Q
011357Q	CALL	10377Q	JMP	11333Q	IN	6Q	MOV LI	10Q
011371Q	MOV MA		LXI H	25453Q	SUB M		JZ	11414Q
011401Q	LXI B	16410Q	MOV EI	1Q	CALL	10665Q	JMP	11333Q
011414Q	CALL	10502Q	CALL	10502Q	MOV AM		DCR H	
011424Q	MOV LI	373Q	SUB M		JZ	13103Q	INR H	

011433Q	MOV LI	10Q	MOV AM		LXI H	25514Q	SUB M	
011442Q	JZ	11506Q	LXI H	1641CQ	MOV AM		LXI H	25456Q
011454Q	SUB M		JZ	11511Q	LXI H	16410Q	MOV AM	
011464Q	LXI H	25457Q	SUB M		JZ	11476Q	JMP	11333Q
011476Q	LXI H	16412Q	MOV MI	1Q	JMP	11511Q	INR L	
011507Q	MOV MI	1Q	LXI H	25450Q	MOV CM		DCR C	
011516Q	LXI H	164J3Q	MOV MC		LXI H	16402Q	MOV CM	
011526Q	DCR C		JNZ	11540Q	CALL	10377Q	JMP	11522Q
011540Q	MOV LI	12Q	MOV CM		DCR C		JZ	11547Q
011547Q	MOV LI	2Q	MOV MI	1Q	LXI H	25461Q	MOV MI	237Q
011560Q	INR L		MOV MI	157Q	LXI H	25461Q	INR M	
011567Q	LXI B	10105Q	MOV EI	2Q	CALL	10665Q	LXI B	25461Q
011602Q	MOV EI	1Q	CALL	10665Q	LXI B	25462Q	MOV EI	1Q
011614Q	CALL	10665Q	LXI B	10107Q	MOV EI	2Q	CALL	10665Q
011627Q	CALL	10502Q	MOV AM		DCR H		MOV LI	375Q
011636Q	SUB M		JZ	12072Q	INR H		MOV LI	10Q
011645Q	MOV AM		MOV LI	23Q	SUB M		JZ	11707Q
011654Q	LXI H	25504Q	MOV CM		INR L		MOV BM	
011662Q	LXI H	22037Q	DAD B		XCHG		LXI H	16410Q
011672Q	MOV CM		MOV AC		STAX D		LHLD	25504Q
011700Q	INX H		SHLD	25504Q	JMP	11627Q	LHLD	25504Q
011712Q	DCX H		SHLD	25504Q	LXI H	25504Q	MOV CM	
011722Q	INR L		MOV BM		LXI H	22037Q	DAD B	
011730Q	MOV AM		SUB I	240Q	JNZ	11750Q	LHLD	25504Q
011741Q	DCX H		SHLD	25504Q	JMP	11716Q	LHLD	25504Q
011753Q	INX H		SHLD	25504Q	LXI H	25504Q	MOV CM	
011763Q	INR L		MOV BM		LXI H	22037Q	DAD B	
011771Q	XCHG		LXI H	16423Q	MOV CM		MOV AC	
011777Q	STAX D		LHLD	25504Q	INX H		SHLD	25504Q
012007Q	LXI H	25504Q	MOV CM		INR L		MOV BM	
012015Q	LXI H	22C37Q	DAD B		XCHG		LXI H	16423Q
012025Q	MOV CM		MOV AC		STAX D		LHLD	25504Q
012033Q	INX H		SHLD	25504Q	LXI H	25504Q	MOV CM	
012043Q	INR L		MOV BM		LXI H	22037Q	DAD B	
012051Q	XCHG		LXI H	16424Q	MOV CM		MOV AC	
012057Q	STAX D		LHLD	25504Q	INX H		SHLD	25504Q
012067Q	JMP	11563Q	INR H		MOV LI	12Q	MOV CM	
012076Q	DCR C		JZ	12437Q	LXI H	25515Q	MOV CM	
012106Q	DCR C		JZ	13031Q	MOV LI	104Q	MOV CM	
012115Q	INR L		MOV BM		LXI H	22037Q	DAD B	
012123Q	XCHG		LXI H	16424Q	MOV CM		MOV AC	
012131Q	STAX D		LHLD	25504Q	INX H		SHLD	25504Q
012141Q	LXI H	25504Q	MOV CM		INR L		MOV BM	
012147Q	LXI H	22037Q	DAD B		XCHG		LXI H	16424Q
012157Q	MOV CM		MOV AC		STAX D		LHLD	25504Q
012165Q	INX H		SHLD	25504Q	LXI H	25504Q	MOV CM	
012175Q	INR L		MOV BM		LXI H	22037Q	DAD B	
012203Q	XCHG		LXI H	16424Q	MOV CM		MOV AC	
012211Q	STAX D		LHLD	25504Q	INX H		SHLD	25504Q
012221Q	LXI H	25504Q	MOV CM		INR L		MOV BM	
012227Q	LXI H	22037Q	DAD B		XCHG		LXI H	16425Q
012237Q	MOV CM		MOV AC		STAX D		LHLD	25504Q
012245Q	INX H		SHLD	25504Q	LXI H	25476Q	MOV MI	0Q
012256Q	INX H		MOV MI	0Q	MOV AI	11Q	LXI H	25476Q
012266Q	SUB M		INR L		MOV CA		MOV AI	0Q
012273Q	SBC M		JC	12346Q	MOV LI	104Q	MOV CM	
012302Q	INR L		MOV BM		LXI H	22037Q	DAD B	
012310Q	XCHG		LXI H	16423Q	MOV CM		MOV AC	
012316Q	STAX D		LHLD	25504Q	INX H		SHLD	25504Q
012326Q	LXI H	25476Q	MOV CM		INR L		MOV BM	

012334Q	LXI H	1Q	DAD B		SHLD	25476Q	JMP	12261Q
012346Q	DCR L		MOV MI	0Q	INX H		MOV MI	0Q
012354Q	MOV AI	11Q	LXI H	25476Q	SUB M		INR L	
012363Q	MOV CA		MOV AI	0Q	SBC M		JC	12434Q
012372Q	MOV LI	104Q	MOV CM		INR L		MOV BM	
012377Q	LXI H	22037Q	DAD B		MOV MI	377Q	LHLD	25504Q
012410Q	INX H		SHLD	25504Q	LXI H	25476Q	MOV CM	
012420Q	INR L		MOV BM		LXI H	1Q	DAD B	
012426Q	SHLD	25476Q	JMP	12354Q	JMP	13031Q	LXI H	25504Q
012442Q	MOV AM		INR L		MOV BM		SUB I	3Q
012447Q	MOV CA		MOV AB		SBC I	0Q	DCR L	
012454Q	MOV MC		INX H		MOV MA		DCR L	
012460Q	MOV CM		INR L		MOV BM		LXI H	22037Q
012466Q	DAD B		MOV MI	377Q	LHLD	25504Q	INX H	
012475Q	SHLD	25504Q	LXI H	25504Q	MOV CM		INR L	
012505Q	MOV BM		LXI H	22037Q	DAD B		XCHG	
012513Q	LXI H	16423Q	MOV CM		MOV AC		STAX D	
012521Q	LHLD	25504Q	INX H		SHLD	25504Q	LXI H	25504Q
012533Q	MOV CM		INR L		MOV BM		LXI H	22037Q
012541Q	DAD B		XCHG		LXI H	16423Q	MOV CM	
012547Q	MOV AC		STAX D		LHLD	25504Q	INX H	
012555Q	SHLD	25504Q	LXI H	25476Q	MOV MI	0Q	INX H	
012566Q	MOV MI	0Q	MOV AI	7Q	LXI H	25476Q	SUB M	
012576Q	INR L		MOV CA		MOV AI	0Q	SBC M	
012603Q	JC	12655Q	MOV LI	104Q	MOV CM		INR L	
012612Q	MOV BM		LXI H	22037Q	DAD B		XCHG	
012620Q	LXI H	16424Q	MOV CM		MOV AC		STAX D	
012626Q	LHLD	25504Q	INX H		SHLD	25504Q	LXI H	25476Q
012640Q	MOV CM		INR L		MOV BM		LXI H	1Q
012646Q	DAD B		SHLD	25476Q	JMP	12570Q	DCR L	
012656Q	MOV MI	0Q	INX H		MOV MI	0Q	MOV AI	3Q
012665Q	LXI H	25476Q	SUB M		INR L		MOV CA	
012673Q	MOV AI	0Q	SBC M		JC	12743Q	MOV LI	104Q
012703Q	MOV CM		INR L		MOV BM		LXI H	22037Q
012711Q	DAD B		MOV MI	316Q	LHLD	25504Q	INX H	
012720Q	SHLD	25504Q	LXI H	25476Q	MOV CM		INR L	
012730Q	MOV BM		LXI H	1Q	DAD B		SHLD	25476Q
012740Q	JMP	12663Q	DCR L		MOV MI	0Q	INX H	
012747Q	MOV MI	0Q	MOV AI	100Q	LXI H	25476Q	SUB M	
012757Q	INR L		MOV CA		MOV AI	0Q	SBC M	
012764Q	JC	13031Q	MOV LI	104Q	MOV CM		INR L	
012773Q	MOV BM		LXI H	22C37Q	DAD B		MOV MI	377Q
013002Q	LHLD	25504Q	INX H		SHLD	25504Q	LXI H	25476Q
013014Q	MOV CM		INR L		MOV BM		LXI H	1Q
013022Q	DAD B		SHLD	25476Q	JMP	12751Q	LXI H	25474Q
013034Q	MOV MI	0Q	INX H		MOV MI	0Q	MOV LI	115Q
013043Q	MOV MI	0Q	LHLD	25504Q	DCX H		SHLD	25472Q
013054Q	CALL	10377Q	LXI H	16412Q	MOV AM		SUB I	0Q
013065Q	JZ	13111Q	MOV LI	2Q	MOV AM		SUB I	0Q
013075Q	JZ	13111Q	JMP	13054Q	CALL	11167Q	JMP	11333Q
013111Q	LXI B	10111Q	MOV EI	2Q	CALL	10665Q	CALL	10502Q
013124Q	RET		LXI B	10111Q	MOV EI	2Q	CALL	10665Q
013135Q	CALL	11301Q	MOV LI	12Q	MOV CM		DCR C	
013144Q	JZ	13152Q	JMP	13125Q	RET		LXI H	25465Q
013156Q	MOV MI	6Q	MOV LI	67Q	MOV MI	6Q	MOV LI	100Q
013166Q	MOV MI	0Q	INX H		MOV MI	0Q	LXI H	25500Q
013176Q	MOV CM		INR L		MOV BM		LXI H	16427Q
013204Q	DAD B		MOV AM		LXI H	16426Q	SUB M	
013212Q	JZ	13405Q	LXI H	25500Q	MOV CM		INR L	
013222Q	MOV BM		LXI H	16427Q	DAD B		MOV AM	

013230Q	LXI H	16423Q	SUB M	JZ	13334Q	LXI H	25500Q
013242Q	MOV CM		INR L	MOV BM		LXI H	16427Q
013250Q	DAD B		MOV AM	SUB I	274Q	JNZ	13273Q
013257Q	LXI H	25500Q	MOV CM	INR L		MOV BM	
013265Q	LXI H	16427Q	DAD B	MOV MI	240Q	LXI H	25500Q
013276Q	MOV CM		INR L	MOV BM		LXI H	16427Q
013304Q	DAD B		MOV AM	LXI H	25460Q	MOV MA	
013312Q	LXI B	25460Q	MOV EI	CALL	10665Q	LHLD	25500Q
013325Q	INX H		SHLD	JMP	13173Q	LXI H	25500Q
013337Q	MOV CM		INR L	MOV BM		LXI H	16427Q
013345Q	DAD B		MOV AM	LXI H	25460Q	MOV MA	
013353Q	LXI B	25460Q	MOV EI	CALL	10665Q	LXI B	16424Q
013366Q	MOV EI	1Q	CALL	LHLD	25500Q	INX H	
013377Q	SHLD	25500Q	JMP	CALL	11301Q	RET	
013411Q	LXI H	25476Q	MOV MI	INX H		MOV MI	0Q
013421Q	CALL	11134Q	MOV AM	MOV LI	23Q	SUB M	
013430Q	JZ	13516Q	MOV LI	MOV AM		SUB I	12Q
013440Q	JZ	13421Q	MOV AM	MOV LI	25Q	SUB M	
013447Q	JZ	13554Q	MOV LI	MOV AM		MOV LI	26Q
013457Q	SUB M		JZ	LXI H	25476Q	MOV CM	
013467Q	INR L		MOV BM	LXI H	16427Q	DAD B	
013475Q	XCHG		LXI H	MOV CM		MOV AC	
013503Q	STAX D		LHLD	INX H		SHLD	25476Q
013513Q	JMP	13421Q	LXI H	MOV CM		INR L	
013523Q	MOV BM		LXI H	DAD B		XCHG	
013531Q	LXI H	16423Q	MOV CM	MOV AC		STAX D	
013537Q	CALL	11134Q	LHLD	INX H		SHLD	25476Q
013551Q	JMP	13421Q	LXI H	MOV MI	0Q	INX H	
013562Q	MOV MI	0Q	MOV AI	LXI H	25500Q	SUB M	
013572Q	INR L		MOV CA	MOV AI	0Q	SBC M	
013577Q	JC	13625Q	CALL	LXI H	25500Q	MOV CM	
013611Q	INR L		MOV BM	LXI H	1Q	DAD B	
013617Q	SHLD	25500Q	JMP	MOV LI	76Q	MOV CM	
013630Q	INR L		MOV BM	LXI H	16427Q	DAD B	
013636Q	XCHG		LXI H	MOV CM		MOV AC	
013644Q	STAX D		CALL	JMP	13411Q	LXI H	25476Q
013656Q	MOV CM		INR L	MOV BM		LXI H	16427Q
013664Q	DAD B		XCHG	LXI H	16423Q	MOV CM	
013672Q	MOV AC		STAX D	LHLD	25476Q	INX H	
013700Q	SHLD	25476Q	LXI H	MOV CM		INR L	
013710Q	MOV BM		LXI H	DAD B		XCHG	
013716Q	LXI H	16426Q	MOV CM	MOV AC		STAX D	
013724Q	CALL	13153Q	MOV LI	MOV CM		DCR C	
013733Q	JZ	13741Q	CALL	RET		LXI H	25465Q
013745Q	MOV MI	2Q	MOV LI	MOV MI	2Q	LXI H	16372Q
013756Q	MOV MI	1Q	INR H	MOV LI	2Q	MOV MI	0Q
013765Q	LXI H	25451Q	MOV MI	MOV LI	115Q	MOV MI	0Q
013776Q	LXI H	16412Q	MOV MI	MOV LI	1Q	MOV MI	0Q
014007Q	DCR L		MOV MI	DCR H		MOV LI	377Q
014015Q	MOV MI	0Q	LXI H	MOV MI	0Q	MOV LI	76Q
014026Q	MOV MI	0Q	INX H	MOV MI	0Q	MOV AI	62Q
014035Q	LXI H	25476Q	SUB M	INR L		MOV CA	
014043Q	MOV AI	0Q	SBC M	JC	14101Q	LXI B	16413Q
014054Q	MOV EI	1Q	CALL	LXI H	25476Q	MOV CM	
014065Q	INR L		MOV BM	LXI H	1Q	DAD B	
014073Q	SHLD	25476Q	JMP	CALL	10233Q	CALL	10315Q
014107Q	MOV LI	65Q	MOV MI	MOV LI	67Q	MOV MI	6Q
014117Q	MOV LI	64Q	MOV MI	MOV LI	66Q	MOV MI	6Q
014127Q	LXI B	10111Q	MOV EI	CALL	10665Q	LXI B	10021Q
014142Q	MOV EI	33Q	CALL	LXI B	10054Q	MOV EI	10Q

014154C	CAL_	10665Q	CALL	10502Q	MOV AM	SUB I	260Q
014165C	MOV MA		MOV CM		DCR C	JZ	14201Q
014173C	CAL_	13125Q	JMP	13742Q	LXI B	MOV EI	2Q
014206C	CAL_	10665Q	CALL	13411Q	JMP	EI	
014220Q	HLT						

NO PROGRAM ERRORS

APPENDIX B

AMPS

PROGRAM FLOWCHARTS

MAIN

MEDIT (1)

MEDIT (3)

KENTRY

CLIN

CLOUT

MEDIT (4)

READIN

READ

SEND

PUNCH

Notice

This computer code material was prepared as an account of work sponsored by the United States government. Neither the United States nor the United States Energy Research & Development Administration, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights.

NOTICE

Reference to a company or product name does not imply approval or recommendation of the product by the University of California or the U.S. Energy Research & Development Administration to the exclusion of others that may be suitable.

Printed in the United States of America

Available from

National Technical Information Service

U.S. Department of Commerce

5285 Port Royal Road

Springfield, VA 22161

Price: Printed Copy \$: Microfiche \$3.00

<u>Page Range</u>	<u>Domestic Price</u>	<u>Page Range</u>	<u>Domestic Price</u>
001-025	\$ 3.50	326-350	10.00
026-050	4.00	351-375	10.50
051-075	4.50	376-400	10.75
076-100	5.00	401-425	11.00
101-125	5.50	426-450	11.75
126-150	6.00	451-475	12.00
151-175	6.75	476-500	12.50
176-200	7.50	501-525	12.75
201-225	7.75	526-550	13.00
226-250	8.00	551-575	13.50
251-275	9.00	576-600	13.75
276-300	9.25	601-up	*
301-325	9.75		

*Add \$2.50 for each additional 100 page increment from 601 to 1,000 pages;
add \$4.50 for each additional 100 page increment over 1,000 pages.

Technical Information Department

LAWRENCE LIVERMORE LABORATORY

University of California | Livermore, California | 94550