AD-A217 999

DTIC FILE COPY

CHEMICAL
RESEARCH,
DEVELOPMENT &
ENGINEERING
CENTER

CRDEC-TR-88090

SAPLOT: A PROGRAM FOR THE ANALYSIS OF DATA COLLECTED WITH A LECROY 3500SA SIGNAL AVERAGER

DTIC
ELECTE
FEB 14 1990
S D
D

by Alan P. Force, Ph.D.

RESEARCH DIRECTORATE

April 1988

DISTRIBUTION STATEMENT A
Approved for public release
Distribution Unlimited

U.S. ARMY
ARMAMENT
MUNITIONS
CHEMICAL COMMAND

Aberdeen Proving Ground, Maryland 21010-5423

02 12 119

Disclaimer

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorizing documents.

Distribution Statement

Approved for public release; distribution is unlimited.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CRDEC-TR-88090	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION CRDEC 6b. OFFICE SYMBOL (*If applicable*) SMCCR-RSL	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (*City, State, and ZIP Code*) Aberdeen Proving Ground, MD 21010-5423	7b. ADDRESS (*City, State, and ZIP Code*)
8a. NAME OF FUNDING / SPONSORING ORGANIZATION CRDEC 8b. OFFICE SYMBOL (*If applicable*) SMCCR-RSL	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (*City, State, and ZIP Code*) Aberdeen Proving Ground, MD 21010-5423	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO: PROJECT NO: 1C162706 TASK NO.: A553C WORK UNIT ACCESSION NO.:

11. TITLE (*Include Security Classification*)
SAPLOT: A Program for the Analysis of Data Collected with a LeCroy 3500SA Signal Averager

12. PERSONAL AUTHOR(S)
Force, Alan P., Ph.D.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT
Technical	FROM 87 Mar TO 88 Jan	1988 April	35

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP
15	06	03

18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
LeCroy 3500,
Signal averager,
Plotting routines. (EG)

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

A computer program for the analysis of data collected with a LeCroy 3500SA signal averager has been developed. In addition to plotting routines, the program includes a conversion routine to transform data from the three-byte integer format used for disk storage to a FORTRAN compatible format. Disk copies of the program are available upon request for other groups at CRDEC working with a LeCroy 3500SA.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL SANDRA J. JOHNSON	22b. TELEPHONE (*Include Area Code*) (301) 671-2914 22c. OFFICE SYMBOL SMCCR-SPS-T

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

PREFACE

The work described in this report was authorized under Project No. 1C162706A553C. This work was started in March 1987 and completed in January 1988.

The use of trade names or manufacturers' names in this report does not constitute an official endorsement of any commercial products. This report may not be cited for purposes of advertisement.

Reproduction of this document in whole or in part is prohibited except with permission of the Commander, U.S. Army Chemical Research, Development and Engineering Center, ATTN: SMCCR-SPS-T, Aberdeen Proving Ground, Maryland 21010-5423. However, the Defense Technical Information Center and the National Technical Information Service are authorized to reproduce the document for U.S. Government purposes.

This report has been approved for release to the public.

Accesion For
NTIS CRA&I
DTIC TAB
Unannounced
Justification
By
Distribution /
Availability Codes
Dist | Avail and / or Special
A-1

DTIC COPY INSPECTED 5

BLANK

CONTENTS

BLANK

SAPLOT: A PROGRAM FOR THE ANALYSIS OF DATA COLLECTED WITH A LECROY 3500SA SIGNAL AVERAGER

1. *INTRODUCTION*

The short, bright pulses that can be produced at high repetition rates by modern lasers permit the long-range analysis of trace atmospheric species via Differential Absorption/Differential Scattering Lidar (DIAL/DISC) techniques. In the laboratory, nascent species with lifetimes in the nanosecond time regime and rapid molecular and atomic transitions can be detected and measured. To obtain accurate measurements, a fast data acquisition system with the ability to average sub-microsecond transient signals, such as the LeCroy 3500SA signal averager in our laboratory, is required.

The 3500SA signal averager acquires signals through plug-in CAMAC (Computer Automated Measurement And Control, IEEE-583) modules. The LeCroy data acquisition software (SA and SA16) controls transient recorder modules with sampling rates as fast as 200 MHz (5–nanosecond sample interval), can average 256 to 8192 (0.25 to 8k) point waveforms at a rate of 700k points per second (i.e., a 1k waveform at 700 Hz), and can store 65536 full scale waveforms in a separate 24-bit by 8k histogram averaging memory.

Included among the data analysis features are routines for archiving data, Fast Fourier Transforms, background subtract, and peak search and area computations. The system can also be programmed in FORTRAN and MACRO Assembler * under the CP/M operating system, allowing the user to customize data acquisition and analysis. Since data acquired with the SA and SA16 software are stored in a three-byte integer format, a conversion routine is required to transform the data into a FORTRAN-compatible format for user-developed analysis routines. A listing of the FORTRAN program SAPLOT.OVL (Signal Averager PLOT.OVerLay), which performs the format conversion and facilitates custom data analysis, is included at the end of this report. Disk copies of the program that are ready to run are also available upon request from other research groups using a LeCroy 3500SA at CRDEC.

2. *SUBPROGRAMS*

2.1 *SAPLOT*

The CP/M operating system uses an area of system memory, located at address 005CH, called the Transient File Control Block (TFCB)** for random access file operations. The TFCB contains the drive code, name, type, and size of a file being accessed. When a program is entered for execution at the system prompt, the operating system scans the remainder of the line following the program name and writes the information given there into the TFCB. SAPLOT calls a MACRO subroutine called GETIT to read the TFCB, open the designated file, and read 128-byte data blocks into memory. GETIT, a subroutine of the MACRO program GETIT.MAC, was developed from a set of routines found in the file copy program COPY.ASM published in Ken Barbier's book on

---

* Microsoft FORTRAN-80 and MACRO-80.

** Digital Research Corporation, *CP/M Operating System Manual*, 1983.

CP/M assembly language programming.* If the file named in the TFCB can not be found and opened on the designated disk, GETIT prints an error message and returns an error code to SAPLOT. Program execution is then transfered to the GETIT subroutine NXTFIL (NeXT FILe), which requests a new file name.

The SA software allows the user to store up to a page of comments as a data header. Once the file is read into the memory space BUFFER that was set aside by the program, GETIT prints the file on the screen until the "control Z" header terminator is reached. It then stores the address of the third data point (the first two points are not true data points) and returns control to SAPLOT. The three bytes of each data point are then read individually from memory by the GETIT subroutine TRNSB (TRaNslation SuBroutine), converted to Floating Point format, and compared to values in memory to determine the minimum and maximum values in the file. The library call HMFWR (Histogram Memory Floating point WRite)** is then used to store the data in the histogram memory for subsequent analysis.

Program memory in the LeCroy 3500 is normally limited to 64k by the 8-bit Intel 8085A micro-processor. To maximize the available program memory for data analysis, an overlay subprogram is used. When the program SAPLT2 is called by SAPLOT through the OVLAY call, data not protected is destroyed as the new program is read into memory. Information shared by the two programs must be stored in either the histogram memory or be listed in a BLOCK DATA subroutine and in a COMMON/SHARED statement in each FORTRAN program (see the *System 3500 Operator's Manual* and the listing for COMBLK.FOR at the end of this report). This process frees most of the memory used by SAPLOT and GETIT for the analysis routines. The only variables passed in the program memory are the values for the smallest data point in the file, the difference between it and the largest point, and the number of points in the file. Execution of an overlay program terminates when a RETURN statement is reached. The calling program is then read back into memory and resumes execution at the statement after the OVLAY call. In SAPLOT, NXTFIL is called, and the name of the next file to be analysed is requested.

2.2 SAPLT2

The program SAPLT2 has been developed to produce plots of SA data on the computer screen and on a Hewlett-Packard 7475A Graphics Plotter. It should also work with other plotters that have an RS-232-C port and run the Hewlett-Packard Graphics Language (HPGL). Screen plots are limited by the system to 512 points of data with a vertical resolution of 1 part in 245. This leaves room on the screen for an x-axis and headers. A cursor is placed on the first point on the screen along with headers that give the cursor position in terms of the transient recorder channel number and the relative amplitude. Cursor movement is controlled by the FORTRAN subroutine CRSR and the MACRO subroutine CURSOR. The statement

KEY=INP(253)

returns a value of 15 when there is no input from the system keypad or when the RESET key is depressed when the statement above is reached in the program. Values of 0 to 14 are returned for input from the other keys (Figure 1). Using the value returned in KEY in a FORTRAN IF statement allows the program to branch without stopping to print an input request and wait for the input.

---

* Ken Barbier, *CP/M Assembly Language Programming*, Prentice-Hall Spectrum Books, 1983.

** See the LeCroy *System 3500 Operator's Manual*

LEFT CRSR 7	← ↓ 6	↑ → 5	RIGHT CRSR 4
DISP MODE 10	MRKR / SCAN 9	DF ROI / DL ROI 8	2ND FUNCT 3
V. MAX / V. MIN 13	M. GRP / SG DIS 12	ROI / DEL DIS 11	ACQ / OUTPUT 2
START / CL DAT 14	RESET 15	STOP 0	CONT / CL ROI 1

Figure 1. The System 3500 Keypad and the Value Returned for Each Key by the Statement KEY=INP(253)

The FORTRAN subroutine PLTTR is used to produce plots of from 2 to 8192 points on the graphics plotter with a vertical resolution of 1 part in 10,000. PLTTR includes routines to determine the divisions of the x-axis from the number of points being plotted, to draw the x-axis, and to determine the vertical scale of the plot from the values of the points in the file. Plot labels and comments are produced by MACRO subroutines found in HP.MAC. These routines read ASCII strings from the terminal and send the instructions required to print the label to the plotter.

The interface between the graphics plotter and the LeCroy 3500 requires a cable that connects pins 1, 2, 3, and 7 of the system printer port to pins 1, 3, 2, and 7 of the plotter. The plotter switch settings should be set for no parity checking and a baud rate of 4800 with 1 stop bit. The paper size may be set according to user requirements.

3. RUNNING THE PROGRAM

3.1 Data Retrieval

To run the program for data analysis, the user first types

SAPLOT d:filename.typ,

where d is the disk drive in which the file will be found, and typ is the file type (normally, typ=SAD). If no file name is entered or if the file is not found on the designated disk, an error message is printed, and a new file name is requested by the program.

When SAPLOT begins to execute, the user is first asked to enter the number of points in the file; typing a <RETURN> enters the default value of 4096 (4k). The message

TRANSLATION OF 3500SA DATA FOR PLOT

is printed on the screen to indicate that control has been transfered to GETIT. The data header is then printed, followed by the message

FILE IN MEMORY, PLEASE WAIT

to show that the file has been read into memory successfully and that the format conversion and data storage routines are running. The minimum and maximum values of data points in the file and the difference between them are printed when the conversion is completed. SAPLOT then completes its execution by calling the overlay subprogram SAPLT2, which is then read from the program disk into memory.

3.2 Plotting Data

A request for the size and units of the sample interval used in the data acquisition is made after SAPLT2 has begun to execute. A short menu is printed giving the user the choice of plotting on the screen or the graphics plotter, exiting the program, or returning to SAPLOT for a new file (SAPLT2 returns to this point in the program after each plot). A request for either type of plot results in a request for the channel number of the first point to be plotted and a request for the number of points in the plot. A <RETURN> at the first request gives a plot of the entire file.

3.2.1 Screen Plots

A plot initialization call clears the screen for the screen plot. The program then draws tick marks at 50-point intervals on the x-axis and labels the axis end points. The first 512 data points of the plot are displayed, a vertical cursor is drawn on the first point, and the coordinates of the cursor are printed on the top of the screen and labeled.

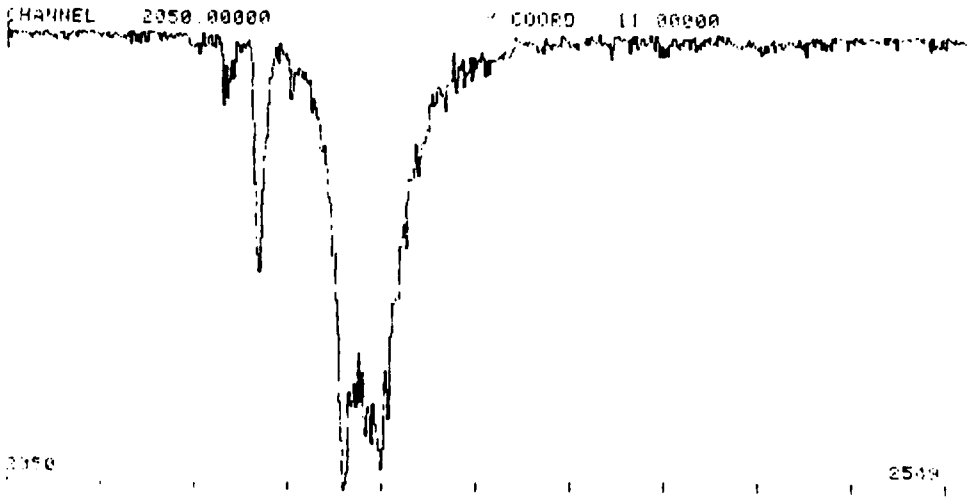

Figure 2. A Sample of a Screen Dump

As noted above, the keypad is used to control cursor movement and to update the cursor coordinate display. The up arrow/right arrow (↑ →) and RIGHT CRSR keys move the cursor to the right in steps of 1 and 5 points each; holding the key down results in continous cursor motion. The left arrow/down arrow (← ↓) and LEFT CRSR keys perform the same functions to the left. The displays for the channel number and y-coordinate of the cursor are updated as it moves from point to point.

The other keypad keys are used for the folowing functions:
2ND FUNCT calls a screen dump (Figure 2). This produces a copy of the screen display on the system dot matrix printer.
V.MAX/V.MIN allows the user to expand or compress the vertical scale of the plot.
CONT/CL ROI calculates and displays the elapsed time from the first data point in the file to the cursor position.
START/CL DAT replots the data, since the moving cursor can remove lines that connect the data points.
STOP serves two functions. It will cause an exit from the plotting routine if it is pressed while the plot is being produced, and it causes the next 512-point section of the data to be plotted when a <RETURN> is entered after the plot is complete.
The rest of the keys are ignored by SAPLT2, but can be programmed for other functions.

3.2.2 Using the Graphics Plotter

An example of a plot produced on the graphics plotter is given in Figure 3. The x-axis, its units label, and the plot are all produced automatically after the user has indicated which part of the file is to be plotted. The labels for the end points of the x-axis are displayed on the screen and must be entered on the keyboard to be printed on the plot. This allows a user to modify the format of the plot according to his needs. A provision is made so that multiline comments can be placed at the top of the plot. The comments may be entered after the data plot is complete.

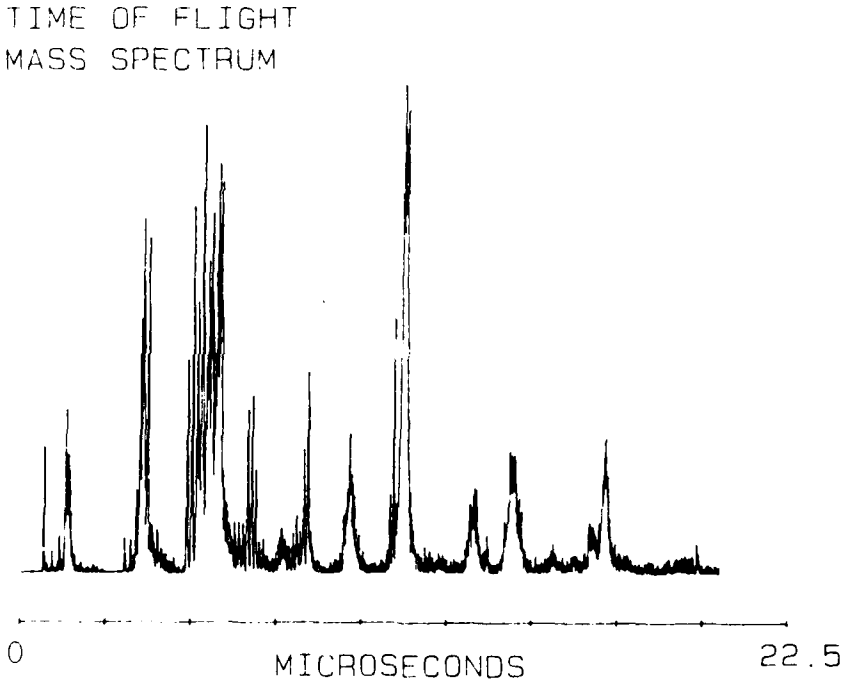

Figure 3. A Plot of a Time-of-Flight Mass Spectrum Produced on the HP Graphics Plotter

4. *PROGRAM MODIFICATION*

SAPLOT, SAPLT2, and the subroutines called by these programs can be modified to suit the needs of the user. A working knowledge of FORTRAN and the MACRO assembler is required. It should be noted that one of the FORTRAN library calls, HMFRD or HMIRD (Histogram Memory Floating point or Integer ReaD), is required to access data from the histogram memory if a new FORTRAN subroutine is written.

Once the modifications have been made, the program must be compiled, linked, and loaded. This can be done for SAPLOT and SAPLT2 by running the batch file SAPLOT.SUB (any new or modified MACRO routines should first be assembled by typing M80 =filename). This is done by typing SUBMIT SAPLOT. SAPLOT.SUB contains all of the commands required to compile, link, and load SAPLOT and SAPLT2. A listing of SAPLOT.SUB is included with the program listing at the end of this report. The SUBMIT SAPLOT command will call each of the command lines in the batch file. When the batch file is complete, the program is ready to run. For more information on how to modify, compile, link, load, and run programs on the System 3500SA, see the LeCroy System 3500 FORTRAN Software Documentation Manuals.

APPENDIX

PROGRAM LISTING

BLANK

```
*
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       THIS PROGRAM RETRIEVES 3005A DATA FROM DISK
C       AND CONVERTS IT TO FLOATING POINT FORMAT
C       FOR USE IN THE PROGRAM SAPLT2.FOR
C       NOVEMBER 1987
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        PROGRAM SAPLOT
        DIMENSION POINT(1)
        COMMON/SHARED/XMIN,DIFF,NPTS
        DO 100 I=1,100
10      XMAX=0.
        XMIN=1.0E12
        WRITE(1,20)
20      FORMAT('0ENTER THE NUMBER OF POINTS IN THE FILE ')
        READ(1,30) NPTS
        IF(NPTS.EQ.0) NPTS=4096
30      FORMAT(I4)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       RETRIEVE DATA FROM DISK
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        CALL GETIT(IOK)
        IF(IOK.NE.1) GOTO 70
        WRITE(1,40)
40      FORMAT('0FILE IN MEMORY, PLEASE WAIT')
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       CONVERT DATA TO 4 BYTE FLOATING POINT
C       AND STORE IN DATA HISTOGRAM MEMORY
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        DO 50 J=3,NPTS
        CALL TRNSB(IDATA)
        BYT=FLOAT(IDATA)
        CALL TRNSB(IDATA)
        BYT=BYT+FLOAT(IDATA)*256.
        CALL TRNSB(IDATA)
        POINT(1)=BYT+FLOAT(IDATA)*65536.
        IF(POINT(1).LT.XMIN) XMIN=POINT(1)
        IF(POINT(1).GT.XMAX) XMAX=POINT(1)
        CALL HMFWR(POINT,J,1)
50      CONTINUE
        DIFF=XMAX-XMIN
        WRITE(1,60) XMIN,XMAX,DIFF
60      FORMAT('0MIN=',F12.0,' MAX=',F12.0,' DIFF=',F8.0)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       THE OVERLAY PROGRAM IS CALLED TO PLOT THE DATA
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        CALL OVLAY('SAPLOT  ',SAPLT2,'SAPLT2  ')
C
C       GET A NEW FILE NAME
C
70      CALL NXTFIL
100     CONTINUE
        END
```

```
*
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       THIS PROGRAM IS AN OVERLAY SUBROUTINE CALLED
C       BY SAPLOT.  IT PLOTS THE 3500 SA DATA TAKEN FROM
C       DISK BY GETIT.MAC.
C       17 JULY 1987 UPDATE PLOTS ON HP PLOTTER
C       CONVERTED TO PRESENT FORM NOVEMBER 1987
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        SUBROUTINE SAPLT2
        DIMENSION PT(512),IPT(512)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       THE COMMON STATEMENT IS REQUIRED FOR THE OVERLAY
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        COMMON/SHARED/KMIN,DIFF,NPTS
        DATA KSTOP/0/
100     WRITE(1,101)
101     FORMAT('0UNITS OF SAMPLE INTERVAL?',/,
     1  ' ENTER 1 FOR MICRO SECONDS',/,
     2  ' 2 FOR MILLISECONDS',/,
     3  ' RETURN FOR NANOSECONDS ')
        READ(1,210) IUNIT
        WRITE(1,110)
110     FORMAT('0ENTER SAMPLE INTERVAL - FLOATING POINT ')
        READ(1,120) SAMPL
120     FORMAT(F12.5)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       DETERMINATION OF THE POINTS TO BE PLOTTED
C       ALSO DETERMINES IF DATA IS PLOTTED ON THE SCREEN
C       OR A HARD COPY ON THE HP PLOTTER USING PLTTR
C       MAXIMUM POINTS=8192
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
200     WRITE(1,201)
201     FORMAT('0ENTER 1 TO PLOT ON SCREEN',/,
     1  ' 2 TO EXIT PROGRAM, 3 FOR A NEW FILE',/,
     2  ' RET FOR HARD COPY ')
        READ(1,210) IPLOT
210     FORMAT(I2)
        IF(IPLOT.EQ.3) RETURN
        IF(IPLOT.EQ.2) STOP
        IF(IPLOT.NE.1) CALL PLTTR(POINT,SAMPL,IUNIT,KMIN,DIFF,NPTS)
        IF(IPLOT.NE.1) GOTO 100
        CALL WHTPTS(ISTART,IFIN,NPTS)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       INITIALIZATION OF PLOT VARIABLES
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       START OF PLOT
C       PLOT DONE IN 512 POINT SECTIONS
C       INP(253) RETURNS 0 IF THE STOP KEY ON THE KEYPAD
C       IS DEPRESSED WHEN THE INSTRUCTION IS REACHED,
C       1 FOR CONTROL ROI
```

```
*
C       2 FOR ACQ/OUTPUT
C       3 FOR 2ND FUNCT
C       4 FOR RIGHT CRSR
C       5 FOR UP/RIGHT ARROW
C       6 FOR LEFT/DOWN ARROW
C       7 FOR LEFT CRSR
C       8 FOR DF ROI/DL ROI
C       9 FOR MRKR/SCAN
C       10 FOR DISP MODE
C       11 FOR ROI/DBL DS
C       12 FOR M.GRP/SG DIS
C       13 FOR V.MAX/V.MIN
C       14 FOR START/CL DAT
C       15 FOR RESET OR WHEN NO KEY IS DEPRESSED
C       THE KEYPAD ALLOWS COMMANDS TO BE INPUT
C       WITHOUT STOPPING THE PROGRAM IF NO COMMAND IS REQUIRED
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        DO 300 IX=ISTART,IFIN,512
        KEY=INP(253)
        IF(KEY.EQ.KSTOP) GOTO 310
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       RESET CURSOR TO THE START OF THE SCREEN AFTER
C       A SCREEN PLOT
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        TIME1=FLOAT(IX-1)
        IFIN2=IFIN-IX
        IF(IFIN2.GT.512) IFIN2=512
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       GET NEXT 512 POINTS FROM DATA MEMORY
C       INITIALIZE GRAPHICS AND
C       DRAW AND LABEL SCREEN AXIS
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        CALL HMFRD(PT,IX,IFIN2)
        CALL PLINIT
        DO 250 IJ=1,11
        IK=(IJ-1)*50
        CALL PLOT(IK,0,3)
        CALL PLOT(IK,5,2)
250     CONTINUE
        XXX=FLOAT(IX)
        ZZZ=XXX+499.
        CALL NUMBER(0,15,XXX,-1,5,0)
        CALL NUMBER(499,15,ZZZ,-1,5,1)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       CALCULATE VALUE TO BE PLOTTED ON THE SCREEN FOR
C       POINT 1
C       SET = MIN AND MAX VALUES FOR MIN-MAX DETERMINATION
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        IPT(1)=IFIX((PT(1)-XMIN)*245/DIFF)
        MIN=IPT(1)
```

```
*
      MAX=IPT(1)
      CALL PLOT(0.,IPT(1),3)
      CALL PLOT(0,IPT(1),2)
      DO 270 NM=2,IFIN2
      IPT(NM)=IFIX((PT(NM)-XMIN)*245/DIFF)
      CALL PLOT((NM-1),IPT(NM),1)
      IF(IPT(NM).LT.MIN) MIN=IPT(NM)
      IF(IPT(NM).GT.MAX) MAX=IPT(NM)
270   CONTINUE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     CRSR DRAWS AND MOVE THE CURSOR AND CALCULATES
C     TIME
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      CALL CRSR(IPT,TIME1,SAMPL,IX,XXX,ZZZ,MIN,MAX)
      READ(1,210) ICON
300   CONTINUE
310   CALL PLTCLS
      WRITE(1,320)
320   FORMAT('0TYPE 1 TO QUIT, RETURN TO RE-PLOT DATA')
      READ(1,210) IQUIT
      IF(IQUIT.NE.1) GOTO 200
      STOP
      END



      SUBROUTINE WHTPTS(ISTART,IFIN,NPTS)
      WRITE(1,1)
1     FORMAT('0ENTER THE CHANNEL NUMBER OF THE 1ST',/,
     1  ' POINT TO BE PLOTTED, OR',/,
     2  ' TYPE RETURN FOR THE ENTIRE FILE ')
      READ(1,4) ISTART
      IF(ISTART.NE.0) GOTO 2
      ISTART=3
      IFIN=NPTS
      RETURN
2     WRITE(1,3)
3     FORMAT('0ENTER NUMBER OF POINTS TO BE PLOTTED ')
      READ(1,4) IFIN
4     FORMAT(I4)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     1ST 2 POINTS ARE NOT USED. SINCE THEY ARE
C     NOT TRUE DATA POINTS
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IF(ISTART.LT.3) ISTART=3
      IFIN=IFIN-1+ISTART
      IF(IFIN.GT.NPTS) IFIN=NPTS-1+ISTART
      RETURN
      END
```

```
*
      SUBROUTINE CRSR(IPT,TIME1,SAMPL,IX,XXX,ZZZ,MIN.MAX)
      DIMENSION IPT(512)
      DATA KSTOP/0/,K2ND/3/,KRT5/4/.KRT/5/.KLEFT/6/,KL5/7/
      DATA KRGO/8/.KLGO/9/.MSEC/1/.MULT/13/.KPLOT/14/
C
C     SET UP CHANNEL NUMBER FOR PLOT HEADER
C
      CHAN1=TIME1+1
60    K=0
      CHAN2=CHAN1
      CALL NUM(K.IPT.IY.CHAN2)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C     SUBROUTINE RETURNS TO LINE 100 AFTER
C     EACH KEY FUNCTION IS COMPLETED
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
100   KEY=INP(253)
C
C     RIGHT CURSOR
C
      IF(KEY.NE.KRT) GOTO 200
      CALL ERASE
      IF(K.EQ.511) K=-1
      K=K+1
      CHAN2=CHAN1+K
      CALL NUM(K.IPT.IY,CHAN2)
      GOTO 100
C
C     LEFT CURSOR
C
200   IF(KEY.NE.KLEFT) GOTO 300
      CALL ERASE
      IF(K.EQ.0) K=512
      K=K-1
      CHAN2=CHAN1+K
      CALL NUM(K.IPT.IY.CHAN2)
      GOTO 100
C
C     RIGHT CURSOR X 5
C
300   IF(KEY.NE.KRT5) GOTO 400
      CALL ERASE
      IF(K.GT.506) K=506-K
      K=K+5
      CHAN2=CHAN1+K
      CALL NUM(K.IPT,IY,CHAN2)
      GOTO 100
C
C     LEFT CURSOR X 5
C
400   IF(KEY.NE.KL5) GOTO 500
```

```
*
      CALL ERASE
      IF(K.LT.5) K=K+511
      K=K-5
      CHAN2=CHAN1+K
      CALL NUM(K,IPT,IY,CHAN2)
      GOTO 100
C
C     CALCULATE TIME FROM START TO PRESENT CURSOR POSITION
C
500   IF(KEY.NE.MSEC) GOTO 600
      TIMEX=(TIME1+K)*SAMPL
      CALL WRTSYM(255,255,'TIME    ',7,0)
      CALL NUMBER(325,255,TIMEX,5,15,0)
      GOTO 100
C
C     REPLOT DATA
C
600   IF(KEY.NE.KPLOT) GOTO 700
      CALL PLOT(0,IPT(1),3)
      CALL PLOT(0,IPT(1),2)
      DO 610 I=1,511
      CALL PLOT(I,IPT(I+1),1)
610   CONTINUE
      GOTO 100
C
C     DATA IS REPLOTTED ON A DIFFERENT SCALE
C     RESOLUTION IS <= 1 PART IN 245
C
700   IF(KEY.NE.MULT) GOTO 800
      CHANG=FLOAT(MAX-MIN)
      BOTM=FLOAT(MIN)
      CALL PLINIT
      CALL PLTOLS
      WRITE(1,704) CHANG
704   FORMAT('0LARGEST PEAK IS',F5.0,' POINTS DEEP',/,
     1   ' ')
      WRITE(1,705)
705   FORMAT('CENTER SCALE FACTOR FROM 10 TO 245',/,
     1  ' RETURN=245=FULL SCALE ')
      READ(1,706) ISCAL
706   FORMAT(I4)
      SCALE=FLOAT(ISCAL)
      IF(SCALE.LT.1.) SCALE=245.
      CALL PLINIT
      DO 707 MARK=1,550,50
      MARK1=MARK-50
      CALL PLOT(MARK1,0,3)
      CALL PLOT(MARK1,5,2)
707   CONTINUE
      CALL NUMBER(0,15,XXX,-1,5,0)
```

```
*
      CALL NUMBER(500,15,ZZZ.-1.5.1)
      PT=FLOAT(IPT(1))
      IY=IFIX(((PT-BOTM)*SCALE/CHANG)+245.-SCALE)
      IPT(1)=IY
      CALL PLOT(0,IY,3)
      CALL PLOT(0,IY.2)
      DO 710 I=1.511
      PT=FLOAT(IPT(I+1))
      IY=IFIX(((PT-BOTM)*SCALE/CHANG)+245.-SCALE)
      IPT(I+1)=IY
      CALL PLOT(I.IY,1)
710   CONTINUE
      GOTO 80
800   IF(KEY.EQ.KSTOP) RETURN
C
C     HARD COPY OF SCREEN PLOT
C
      IF(KEY.EQ.K2ND) CALL SCRDMP
      GOTO 100
      END



C
C     HEADER OF THE SCREEN PLOT IS SET UP
C
      SUBROUTINE NUM(K,IPT,IY,CHAN2)
      DIMENSION IPT(512)
      IY=255-IPT(K+1)
      CALL DRAW(K.IY)
      Z=FLOAT(IY)
      CALL WRTSYM(0.255.'CHANNEL',7,0)
      CALL NUMBER(72,255.CHAN2.5.15.0)
      CALL WRTSYM(255.255.'Y COORD',7.0)
      CALL NUMBER(325,255,Z,5.15.0)
      RETURN
      END
```

```
*
C
C       ROUTINE PLOTS DATA ON AN HP PLOTTER FROM PORT 2
C       I.E., THE PRINTER PORT
C       THE RESOLUTION IS AS GOOD AS THE DATA OR 1 PART IN
C       10,000, WHICH EVER IS LESS
C
        SUBROUTINE PLTTR(POINT,SAMPL,IUNIT,XMIN,DIFF,NPTS)
        DIMENSION POINT(1)
        CALL WHTPTS(ISTART,IFIN,NPTS)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C       SET UP X AXIS AND THE # OF TICK MARKS
C       AND SET HP PLOTTER SCALE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
        ID=IFIN-ISTART
        IF(ID.GT.1999) DIV=500.
        IF(ID.LT.2000) DIV=100.
        IF(ID.LT.1000) DIV=50.
        IF(ID.LT.200) DIV=10.
        MULT=IFIX(DIV)
        ISCLO=IFIX(FLOAT(ISTART)/DIV)
        ISCLO=ISCLO*MULT-MULT
        ISCHI=IFIX(FLOAT(IFIN+1)/DIV)
        ISCHI=(ISCHI+1)*MULT+MULT
        WRITE(2,200) ISCLO,ISCHI
200     FORMAT(' SC',I4,',',I4,',-2000.,2000;SR1.5,3;')
        ISCLO=ISCLO+MULT
        ISCHI=ISCHI-MULT
        SECLO=SAMPL*FLOAT(ISCLO)
        SECHI=SAMPL*FLOAT(ISCHI)
        ITIC=(ISCHI-ISCLO)/MULT
        WRITE(2,210) ISCLO
210     FORMAT(' SP1;PA',I4,',-1000;PD;XT;')
        DO 220 I=1,ITIC
        WRITE(2,230) MULT
220     CONTINUE
230     FORMAT(' PR',I3,',0;XT;')
C
C       LABEL X AXIS
C
        WRITE(1,240) SECLO
240     FORMAT('01ST SCALE LABEL=',F10.3,',
     1  ' ENTER VALUE WHEN REQUESTED')
        WRITE(2,250) ISCLO
250     FORMAT(' PU;PA',I4,',-1000;CP-.5,-1;')
        CALL COMNT
        WRITE(1,260) SECHI
260     FORMAT('02ND SCALE LABEL=',F10.3,',
     1  ' ENTER VALUE WHEN REQUESTED')
        WRITE(2,270) ISCHI
270     FORMAT(' PA',I4,',-1000;CP-1.2,-1;')
```

```
*
341:            CALL COMNT
342:   C
343:   C        PLOT DATA ON HP
344:   C
345:            DO 400 I=ISTART,IFIN
346:            CALL HMFRD(POINT(1),I,1)
347:            Y=10000.*(1.-(POINT(1)-XMIN)/DIFF)
348:            WRITE(2,350) I,Y
349:   350      FORMAT(' PA',I4,',',F12.4,';PD;')
350:   400      CONTINUE
351:            WRITE(2,410) ISCLO
352:   410      FORMAT(' PU;PA',I4,',12000;CP-1,-1;PRO,C;')
353:   C
354:   C        ENTER COMMENT FOR TOP OF PLOT
355:   C
356:   450      CALL COMNT
357:            WRITE(1,460)
358:   460      FORMAT('0TYPE 1 FOR ANOTHER LABEL LINE ')
359:            READ(1,465) NOTHR
360:   465      FORMAT(I2)
361:            IF(NOTHR.EQ.1) WRITE(2,470)
362:   470      FORMAT(' CP;PRO,0;')
363:            IF(NOTHR.EQ.1) GOTO 450
364:   C
365:   C        PLOT & AXIS UNITS LABEL
366:   C
367:            IF(IUNIT.EQ.0) CALL TIMU0
368:            IF(IUNIT.EQ.1) CALL TIMU1
369:            IF(IUNIT.EQ.2) CALL TIMU2
370:            RETURN
371:            END
```

```
341: *
     ;GETIT.MAC - BASED ON COPY.MAC FOUND IN THE BOOK
     ;CP/M ASSEMBLY LANGUAGE PROGRAMMING
     ;BY KEN BARBIER
     ;PUBLISHED BY PRENTICE-HALL SPECTRUM BOOKS 1983
     ;MODIFIED BY ALAN FORCE APRIL-JULY 1987

     ;SUBROUTINE GETIT REQUIRED FOR REFORM.FOR 23 APRIL 1987
     ;DISKEQ.LIB     21 APRIL 1987
     ;MULTI-WRITE FILE COPY PROGRAM

     ;ASCII CHARACTERS
     CR      EQU     0DH     ;CARRIAGE RETURN
     LF      EQU     0AH     ;LINE FEED
     CTRLZ   EQU     1AH     ;OPERATOR INTERRUPT
     SPAC    EQU     20H     ;ASCII SPACE

     ;CP/M BDOS FUNCTIONS
     RCONF   EQU     1       ;READ CON: INTO A REGISTER
     WCONF   EQU     2       ;WRITE A REGISTER TO CON:
     RBUFF   EQU     10      ;READ A CONSOLE LINE

     ;CP/M DISK ACCESS FUNCTIONS
     INITF   EQU     13      ;INITIALIZE BDOS FUNCTION
     OPENF   EQU     15      ;OPEN FILE FUNCTION
     CLOSF   EQU     16      ;CLOSE FILE FUNCTION
     FINDF   EQU     17      ;FIND FILE FUNCTION
     DELEF   EQU     19      ;DELETE A FILE FUNCTION
     READF   EQU     20      ;READ ONE RECORD FUNCTION
     WRITF   EQU     21      ;WRITE ONE RECORD FUNCTION
     MAKEF   EQU     22      ;CREATE FILE FUNCTION
     SDMAF   EQU     26      ;SET DMA FUNCTION

     ;CP/M ADDRESSES
     RBOOT   EQU     0       ;RE-BOOT CP/M SYSTEM
     DRIVE   EQU     4       ;CURRENT DRIVE SELECTION
     BDOS    EQU     5       ;SYSTEM CALL ENTRY
     MEMAX   EQU     7       ;MSB OF TOP OF MEMORY
     TFCB    EQU     5CH     ;TRANSIENT FILE CONTROL BLOCK
     FCBTY   EQU     TFCB+9  ;FILE TYPE IN FCB
     FCBEX   EQU     TFCB+12 ;FILE EXTENT IN FCB
     FCBS2   EQU     TFCB+14 ;SYSTEM USE IN FCB
     FCBRC   EQU     TFCB+15 ;RECORD COUNT IN FCB
     FCBCR   EQU     TFCB+32 ;CURRENT RECORD IN FCB
     TBUFF   EQU     80H     ;TRANSIENT BUBBER
     TPA     EQU     100H    ;TRANSIENT PROGRAM AREA

     ;CP/M FLAGS
     B040K   EQU     0       ;BDOS RETURN FOR ALL OK
     BDER1   EQU     1       ;BDOS RETURN 1  EOF
     BDER2   EQU     2       ;BDOS RETURN 2
     BDERR   EQU     255     ;BDOS RETURN ERROR FLAG
```

```
*
;MOD OF COPY.LIB ADDED TO GET 23 APRIL 1987

GETIT:: PUSH    H       ;SAVE ADDRESS FOR PASSED PARAMETER
        LDA     DRIVE   ;SAVE INITIAL DRIVE SELECTED
        STA     DRSAV
START1: CALL    CCRLF   ;START A NEW LINE
        LXI     H,SINON ;WITH A SIGN ON MESSAGE
        CALL    COMSG
        CALL    TWOCR
        CALL    GET     ;GET THE NAMED FILE
START2: CALL    CCRLF   ;BEGIN WRITE PORTION
DONE:   LDA     DRSAV   ;RESTORE INITIAL DRIVE
        STA     DRIVE
DONE2:: LXI     H,BUFFER        ;BUFFER ADDRESS TO HL
        LXI     D,1926  ;NEED 1926 BYTES ADDED TO BUFFER
        DAD     D       ;ADDRESS TO SKIP HEADER AND 1ST 2 POINTS
        SHLD    ADSAV   ;SAVE ADDRESS OF FIRST DATA POINT
        POP     H       ;GET ADDRESS FOR PASSED PARAMETER
        LDA     AOK
        MOV     M,A
        RET

;GET.LIB        21 APRIL 1987
;READ A FILE FROM DISK INTO "BUFFER"

GET:    LXI     H,BUFFER ;GET BUFFER START
        SHLD    NEXT    ;  ADDRESS FOR DMA
        LXI     D,TFCB  ;SEE IF THE FILE IS ON DISK
        MVI     C,OPENF ;  AND OPEN FOR READ
        CALL    BDOS
        CPI     BDERR   ;IS IT THERE?
        JNZ     GET1    ;IF YES, READ IT IN
        MVI     A,0
        STA     AOK
        CALL    TWOCR   ;IF NOT, SHOW ERROR
        CALL    SPMSG
        JB      ' CAN NOT FIND ',0
        CALL    SHOFN   ;SHOW FILE NAME
ERREX:  CALL    TWOCR   ;ERROR EXIT TO CP/M
        RET

GET1:   MVI     A,1
        STA     AOK
        XRA     A       ;ZERO RECORD COUNTER
        STA     RECCT   ;  AND READ A FILE INTO BUFFER
GET2:   LHLD    NEXT    ;SET BUFFER ADDRESS
        XCHG
        MVI     C,SDMAF
        CALL    BDOS
```

```
*
        LXI     D,TFCB  ;READ ONE RECORD INTO BUFFER
        MVI     C,READF
        CALL    BDOS
        CPI     BDAOK   ;READ OK?
        JZ      GET3    ;YES, DO MORE
        CPI     BDER1   ;MAYBE. END OF FILE?
        JZ      GETEX1  ;YES, NO PROBLEM
        CALL    REMSG   ;NO, SHOW ERROR
        JMP     ERREX   ;AND ALL DONE

GET3:   LDA     RECCT   ;COUNT THE RECORD
        INR     A
        STA     RECCT
        LHLD    NEXT    ;INCREMENT BUFFER ADDRESS BY
        LXI     D,128   ;  RECORD SIZE
        DAD     D
        SHLD    NEXT
        LDA     MEMAX   ;ROOM LEFT IN RAM?
        DCR     A       ;STOP BELOW CCP
        CMP     H       ;COMPARE MSB
        JNZ     GET2    ;CONTINUE IF NOT EQUAL
        CALL    TWOCR   ;  ELSE SHOW OUT OF MEMORY
        CALL    SPMSG
        DB      ' OUT OF MEMORY ',0
        JMP     ERREX   ;AND GIVE UP

GETEX1: LXI     H,BUFFER        ;WRITE OUT HEADER - SEE COMSG:
MSGJMP: MOV     A,M
        CPI     CTRLZ
        JZ      GETEX
        CALL    CO
        INX     H
        JMP     MSGJMP

GETEX:  CALL    CCRLF   ;NORMAL EXIT
        CALL    CPDMA   ;RESTORE CP/M DMA
        RET

;SHOFN.LIB      22 APRIL 1987
;DISPLAY FILENAME.TYP FROM TRANSIENT FCB
SHOFN:  PUSH    B       ;SAVE TEMP STORE AND INDEX
        PUSH    H
        LDA     FCBTY   ;SAVE FIRST CHAR OF TYPE
        MOV     C,A     ;  INTO TEMPORARY STORE
        XRA     A       ;FORCE 2 TERMINATORS FOR
        STA     FCBTY   ;  FILE NAME AND
        STA     FCBEX   ;  FILE TYPE
        LXI     H,TFCB  ;SHOW DISK DRIVE
        MOV     A,M
        ANI     0FH     ;LIMIT TO 4 BITS
```

```
        ORI     40H     ;CONVERT TO ASCII
        CALL    CO
        MVI     A,':'   ;SHOW THE COLON
        CALL    CO
        INX     H       ;AND SHOW THE FILE TYPE
        CALL    COMSG
        MOV     A,C
        LXI     H,FCBTY ;RESTORE TYPE
        MOV     M,A
        MVI     A,'.'   ;SHOW SEPARATOR
        CALL    CO
        CALL    COMSG   ;SHOW TYPE
        POP     H
        POP     B       ;RESTORE AND RETURN
        RET

;DISKSU.LIB     22 APRIL 1987

;DISPLAY READ ERROR MESSAGE
REMSG:  CALL    TWOCR
        CALL    SPMSG
        DB      'PERMANENT READ ERROR ',CR,LF,0
        RET

;DISPLAY WRITE ERROR MESSAGE
WEMSG:  CALL    TWOCR
        CALL    SPMSG
        DB      'PERMANENT WRITE ERROR',CR,LF,0
        RET

;DISPLAY WRITE OPEN ERROR MESSAGE
WROPN:  CALL    TWOCR
        CALL    SPMSG
        DB      'CAN NOT OPEN FOR WRITE',CR,LF,0
        RET

;RESTORE CP/M DMA ADDRESS TO THE TRANSIENT BUFFER
CPDMA:  LXI     D,TBUFF
        MVI     C,SDMAF
        CALL    BDOS
        RET

;WRTCHR.LIB     22 APRIL 1987
;THE CHARACTER IN REGISTER A IS OUTPUT TO THE CONSOLE
CO:     PUSH    B
        PUSH    D
        PUSH    H
        MVI     C,WCONF ;THE WRITE FUNCTION IS SELECTED
        MOV     E,A     ;CHARACTER TO E
        CALL    BDOS    ;OUTPUT BY CP/M
        POP     H
        POP     D
        POP     B
        RET
```

```
*
;CROUT.LIB      22 APRIL 1987
;A CARRIAGE RETURN AND LINE FEED ARE SENT TO THE CONSOLE
TWOCR:  CALL    CCRLF   ;GIVES TWO LINES
CCRLF:  MVI     A,CR    ;TO BE PRINTED
        CALL    CO
        MVI     A,LF
        JMP     CO      ;LEAVE OUT IF DIRECTLY ABOVE CO

;STRNGO.LIB     22 APRIL 1987
;A MESSAGE PPOINTED  TO BY HL IS SENT TO THE CONSOLE
COMSG:  MOV     A,M     ;GET CHARACTER FROM MEMORY
        ORA     A       ;CHECKS FOR 0 TERMINATOR
        RZ              ; RETURNS IF CHARACTER IS 0
        CALL    CO      ;ELSE OUTPUT THE CHARACTER
        INX     H       ;POINT TO THE NEXT CHARACTER
        JMP     COMSG   ;AND CONTINUE

;STAKMSG.LIB    22 APRIL 1987
;SP POINTS TO THE NEXT INSTRUCTION OF THE MAIN PROGRAM
;WHEN A CALL IS MADE.  IF THIS IS A DB PSEUDO-OP WHICH CONTAINS
;THE LINE TO BE PRINTED IT CAN BE USED AS THE ADDRESS
SPMSG:  XTHL            ;EXCHANGE HL AND TOS
        XRA     A       ;CLEAR FLAGS AND ACCUMULATOR
        ADD     M       ;GET ONE CHARACTER - 0 FLAG SET IF 0
        INX     H       ;POINT TO THE NEXT CHARACTER
        XTHL            ;RESTORE STACK FOR
        RZ              ;RETURN IF DONE
        CALL    CO      ;ELSE DISPLAY THE CHARACTER
        JMP     SPMSG   ;AND CHECK THE NEXT ONE

;TRANSB IS USED TO PLACE A DATA POINT IN THE MEMORY
;LOCATION DESIGNATED BY REFORM FOR THE DATA POINT
TRNSB:: XCHG            ;STORE DATA RETURN ADDRESS IN DE PAIR
        LHLD    ADSAV   ;LOAD ADDRESS OF NEXT DATA POINT IN HL
        MOV     A,M     ;MOVE POINT FROM MEMORY TO A
        INX     H       ;INCREMENT DATA INDEX
        SHLD    ADSAV   ;STORE MEMORY INDEX FOR NEXT POINT
        XCHG            ;MOVE DATA RETURN ADDRESS BACK TO HL
        MOV     M,A     ;MOVE POINT TO MEMORY TO BE PASSED
        RET

        ;NXTFIL READS A FILENAME AND PLACES IT IN THE FCB
NXTFIL:: CALL   SPMSG   ;GET THE FILENAME
        DB      CR,LF,'ENTER FILENAME',CR,LF,0
        LXI     H,FCBSTR+1      ;POINT AT CHAR COUNT
        MVI     M,0     ;ZERO CHAR COUNT
        INX     H       ;POINT AT LINE LENGTH
        MVI     M,14    ; & SET TO 14
        XCHG            ;ADDRESS TO DE FOR BDOS
        MVI     C,RBUFF ;READ BUFFER FUNCTION
```

```
        CALL    BDOS    ;READ FILENAME
        LXI     H,FCBSTR+1      ;POINT AT CHAR COUNT
        MOV     E,M     ; & PLACE IN LSB OF DE
        MVI     D,0     ;ZERO MSB OF DE
        DAD     D       ;ADD TO FCBSTR ADDRESS
        INX     H       ;POINT AT END OF STRING
        MVI     M,'!'   ;INSERT TERMINATOR

        ;CHECK FOR DRIVE DESIGNATOR
        LXI     H,FCBSTR+3      ;CHECK 2ND CHAR OF FILENAME
        MOV     A,M
        DCX     H       ;POINT AT 1ST CHAR
        CPI     ':'     ;2ND CHAR A : ?
        JNZ     NODSK   ;NO
        MOV     A,M     ;GET DRIVE
        CPI     'A'     ;DRIVE A?
        JNZ     BDSK    ; NO, B
        MVI     A,1     ;CODE FOR A
        JMP     DSK2
BDSK:   MVI     A,2     ;CODE FOR B
DSK2:   INX     H
        INX     H       ;POINT AT 1ST CHAR OF FILENAME
        JMP     NODSK2
NODSK:  MVI     A,0     ;DEFAULT DISK
NODSK2: LXI     D,TFCB
        XCHG            ;POINT AT TFCB
        MOV     M,A     ;DESIGNATE DRIVE
        INX     H       ;POINT AT F1 OF TFCB
        XCHG            ;BACK TO FILENAME IN FCBSTR

        ;PUT FILENAME IN FCB
        MVI     C,8     ;SET COUNTER - 8 CHAR IN FILENAME REQ.
DSK3:   MOV     A,M     ;GET NEXT CHAR
        INX     H       ;POINT AT NEXT CHAR
        CPI     '.'     ;UP TO FILE TYPE?
        JZ      PADIT   ;YES, FILL IN SPACES
        CPI     '!'     ;END OF FILENAME?
        JZ      PADIT   ;YES, FILL IN SPACES
        XCHG    ;NEITHER, POINT AT BYTE IN FCB
        MOV     M,A     ;CHAR INTO FCB F1-F8
        INX     H       ;POINT AT NEXT BYTE OF FCB
        XCHG    ;POINT BACK TO FILENAME
        DCR     C       ;DECRIMINT COUNTER
        JNZ     DSK3    ;GET ANOTHER CHAR
        MOV     A,M     ;NEXT CHAR A '.'?
        CPI     '.'     ;
        JNZ     PAD2    ;NO, FILL TYPE WITH SPACES

PUTTYP: INX     H       ;POINT AT 1ST CHAR OF TYPE
        LXI     D,FCBTY ;POINT DE AT TYPE IN FCB
```

```
*
PUT2:   MOV     A,M     ;GET CHAR
        CPI     '!'     ;END OF FILENAME?
        JZ      ZEROS   ;YES. TYPE COMPLETE
        INX     H       ;NEXT CHAR
        XCHG    ;POINT AT FCB
        MOV     M,A     ;CHAR INTO T1-T3 OF FCB
        INX     H       ;NEXT BYTE OF FCB
        XCHG
        JMP     PUT2

        ;PUT SPACES INTO BLANKS OF F1-F8
PADIT:  XCHG    ;POINT AT NEXT BYTE OF FCB
BLNKS:  MVI     M,SPAC
        INX     H       ;NEXT BYTE
        DCR     C       ;COUNTER -1
        JNZ     BLNKS
        XCHG    ;POINT AT TYPE
        DCX     H       ;'.' PRESENT?
        MOV     A,M
        CPI     '.'
        JZ      PUTTYP
        ;NO TYPE GIVEN - PAD WITH SPACES
PAD2:   LXI     H,FCBTY
        MVI     C,3     ;SET UP COUNTER
PAD3:   MVI     M,SPAC
        INX     H
        DCR     C
        JNZ     PAD3

        ;ZERO FCB12-15 & 32
ZEROS:  LXI     H,FCBEX ;POINT AT FCB12
        MVI     C,4
ZERO1:  MVI     M,0
        DCR     C
        JNZ     ZERO1
        LXI     H,FCBCR ;POINT AT FCB32
        MVI     M,0
        RET

;RAM.LIB        22 APRIL        1987
;RAM VARIABLES AND BUFFERS
INBUF:  DS      80      ;LINE INPUT BUFFER
DRSAV:  DS      1       ;CURRENT DRIVE AT ENTRY
RECCT:  DS      1       ;TOTAL RECORDS READ/WRITE
CTSAV:  DS      1       ;SAVE LOCATION FOR COUNT
NEXT:   DS      2       ;NEXT DMA ADDRESS
SPSAV:  DS      2       ;SAVE RETURN ADDRESS FROM SP
ADSAV:  DS      2       ;SAVE ADDRESS OF NEXT DATA POINT
FCBSTR: DS      17      ;STORE FOR FILENAME
AOK:    DS      1       ;STORE FOR FILE FOUND FLAG
SINON:  DB      'TRANSLATION OF 8500/84 DATA FOR PLOT',CR,LF,0

;FROM HERE TO EOP IS BUFFER SPACE
BUFFER: DS      26498   ;USE THIS BUFFER FOR 8K PROGRAM
        END
```

```
*
        ;CURSOR.MAC 20 MAY 1987
        ;UPDATED 9 JULY 1987 TO INCLUDE HP PLOTTER ROUTINES
        ;DISPLAY PROCESSOR ADDRESSES
BASE    EQU     0F0H    ;STANDARD BASE ADDRESS
ACK     EQU     BASE+5  ;ACKNOWLEDGE INPUT PORT
LO      EQU     BASE+6  ;LOW BYTE OUTPUT PORT
HI      EQU     BASE+7  ;HI BYTE OUTPUT PORT
ICTL    EQU     0FEH    ;INTERRUPT CONTROLLER PORT

        ;PROGRAM CONSTANTS
LENGTH  EQU     20
DCODE   EQU     11011010B       ;CODE TO DRAW VERT LINE
ECODE   EQU     11010010B       ;CODE TO ERASE VERT LINE
PCODE   EQU     10101000B       ;CODE TO PLOT POINT



        ;ERASE ROUTINE - ERASES OLD CURSOR AND REPLACES POINT
        ; ON SPECTRUM
ERASE:: MVI     A,ECODE ;GET ERASE CODE AND STORE FOR OUTPUT
        STA     HICOD1
        CALL    OODOUT
        MVI     A,PCODE ;GET CODE TO PLOT PT & STORE
        STA     HICOD1
        XRA     A       ;ZERO A FOR 2ND OPCODE
        STA     HICOD2
        LDA     YCOORD  ;GET Y COORDINATE
        ADI     10      ; & ADJUST OFFSET
        STA     YCOORD
        JMP     CODOUT

        ;DRAW ROUTINE - DRAWS NEW CURSOR AFTER STORING
        ;X AND Y COORDINATES SENT FROM FORTRAN PROGRAM
DRAW::  MVI     A DCODE ;GET DRAW CODE FOR CURSOR
        STA     HICOD1
        MOV     A,M     ;GET X FROM MEMORY
        STA     XCOORD
        INX     H       ;POINT AT YB
        MOV     A,M     ;GET YB
        ANI     1       ;ZERO ALL BUT LSB
        STA     XHIBIT
        DCX     H
        XCHG            ;POINT AT Y
        MOV     A,M     ;GET Y
        SUI     10      ;ADJUST OFFSET OF CURSOR
        STA     YCOORD
        XCHG
        MVI     A,LENGTH        ;GET CURSOR LENGTH
        STA     HICOD2
```

```
*
        ;CODOUT - PUTS VARIABLES IN B-C REGISTER FOR OUTPUT
CODOUT: LDA     XHIBIT  ;GET X8
        MOV     B,A
        LDA     HICOD1  ;GET OPCODE
        ADD     B       ;AND PUT X8 IN FOR OUTPUT
        MOV     B,A     ;MOVE TO B FOR OUTPUT
        LDA     XCOORD  ;GET X
        MOV     C,A     ;MOVE TO C FOR OUTPUT
        CALL    SEND
        LDA     HICOD2
        MOV     B,A
        LDA     YCOORD
        MOV     C,A


        ;SEND ROUTINE OUTPUTS TO DISPLAY PROCESSOR
SEND:   MVI     A,10    ;PREPARE FOR INTERRUPT
        OUT     ICTL    ; CONTROLLER FOR POLLING
WAIT:   IN      ICTL    ;READ INTERRUPT STATUS
        ANI     8       ;CHECK THE INT 3 BIT
        JZ      WAIT    ;LOOP IF NOT SET
        IN      ACK     ;TURN OFF THE INTERRUPT
        MOV     A,C     ;SEND THE LO BYTE
        OUT     LO
        MOV     A,B     ;SEND THE HI BYTE
        OUT     HI
        RET



        ;VARIABLES
MOVNUM: DS      1       ;INCREMENT FACTOR
XCOORD: DS      1       ;X7-X0 OF X COORDINATE
XHIBIT: DS      1       ;LSB IS X8 OF XCOORD
YCOORD: DS      1       ;Y7-Y0 OF YCOORDINATE
HICOD1: DS      1       ;HI BYTE OF 1ST OPCODE
LOCOD1: DS      1       ;LOW BYTE OF 1ST OPCODE
HICOD2: DS      1       ;HI BYTE OF 2ND OPCODE
LOCOD2: DS      1       ;LOW BYTE OF 2ND OPCODE

        END
```

```
*^D
;HP.MAC - USED TO PLOT ON HP7475A
RCONF   EQU     1       ;READ A CHARACTER
CR      EQU     0DH
LF      EQU     0AH
WCONF   EQU     2       ;CONSOLE WRITE FUNCTION
HPBYT   EQU     5       ;OUTPUT FUNCTION SENDS OUT RS232
BDOS    EQU     5       ;CPM OUTPUT FUNCTION
RBOOT   EQU     0       ;REBOOTS SYSTEM
RBUFF   EQU     10      ;FUNCTION TO READ CONSOLE BUFFER
ETX     EQU     3       ;LABEL TERMINATOR



;ENTER AND PLOT LABEL
COMNT:: CALL    SPMSG
        DB      CR,LF,'ENTER LABEL-40 CHARACTER MAX!',CR,LF,0
        LXI     H,MCHAR+1
        MVI     M,0     ;ZERO CHAR COUNT
        DCX     H       ;POINT AT LINE LENGTH
        MVI     M,80    ;AND SET TO 80
        XCHG            ;MCHAR ADDRESS TO DE
        MVI     C,RBUFF ;READ BUFFER FUNCTION
        CALL    BDOS    ;READ A LINE
        LXI     H,MCHAR+1 ;POINT AT CHAR COUNT
        MOV     E,M     ; & PUT IN LSB OF DE
        MVI     D,0     ;0 MSB OF DE
        DAD     D       ;ADD TO ADDRESS IN HL
        INX     H       ;POINT AT THE END OF STRING
        MVI     M,0     ;INSERT TERMINATOR

;SEND A LABEL TO THE HP
HPMSG:  CALL    HPINS
        DB      'LB'
        LXI     H,MCHAR+2       ;POINT AT THE START OF THE LABEL
HPOUT:  MOV     A,M     ;GET CHAR
        ORA     A       ;SEE IF IT IS 0
        JZ      LAST    ;AND FINISH IF 0
        CALL    CTOHP   ;OR SEND CHAR TO HP
        INX     H       ;POINT AT THE NEXT CHAR
        JMP     HPOUT

LAST:   CALL    HPINS   ;FINISH LABEL AND STORE THE PEN
        DB      ETX,'!'
        RET

;INSTRUCTION TO HP PLOTTER
HPINS:  XTHL            ;EXCHANGE HL AN TOS
        XRA     A       ;CLEAR FLAGS AND ACCUMULATOR
        MOV     A,M     ;GET A CHARACTER - 0 FLAG SET IF 0
        CPI     '!'
```

```
M
        INX     H       ;POINT TO NEXT CHAR
        XTHL            ;RESTORE STAK FOR RETURN
        RZ              ;RETURN IF DONE
        CALL    CTOHP   ;SEND CHARACTER TO HP PLOTTER
        JMP     HPINS

;CHARACTER TO HP PLOTTER
CTOHP:  PUSH    B
        PUSH    D
        PUSH    H
        MVI     C,HPBYT
        MOV     E,A     ;CHAR TO E FOR OUTPUT
        CALL    BDOS    ;OUTPUT TO CP/M
        POP     H
        POP     D
        POP     B
        RET

;THIS PRINTS THE FIRST LINE IN THE PROGRAM AFTER THE CALL
;ON THE SCREEN
SPMSG:  XTHL    ;EXCHANGE HL AND TOS
        XRA     A       ;CLEAR FLAGS AND ACCUMULATOR
        ADD     M       ;GET A CHARACTER
        INX     H       ;POINT AT THE NEXT CHARACTER
        XTHL            ;RESET TOS AND HL
        RZ              ;RETURN IF DONE
        CALL    CO
        JMP     SPMSG

;THIS ROUTINE SENDS THE ETX CHARACTER
PRET::  CALL    HPINS
        DB      ETX '!;'
        RET

;THE CHARACTER IN REGISTER A IS SENT TO THE SCREEN
CO:     PUSH    B
        PUSH    D
        PUSH    H
        MVI     C,WCONF ;SELECT WRITE FUNCTION
        MOV     E,A     ;CHAR TO E
        CALL    BDOS
        POP     H
        POP     D
        POP     B
        RET

MASSU:: CALL    HPINS
        DB      'SR1.5,3:SC0,10.-2.13;PA5.-1;CP-5.-1.3;'
        DB      'LBMASS UNITS'.ETX,'SP:PA10,12 SR:!'
        RET
```

```
*
TIMUC:: CALL    HPINS
        DB      'SR1.5,3;SC0,10,-2,12;PA5,-1;CP-5.5,-1.3;'
        DB      'LBNANOSECONDS',ETX,'SP;PA10,12;SR:!'
        RET

TIMU1:: CALL    HPINS
        DB      'SR1.5,3;SC0,10,-2,12;PA5,-1;CP-6,-1.3;'
        DB      'LBMICROSECONDS',ETX,'SP;PA10,12;SR:!'
        RET

TIMU2:: CALL    HPINS
        DB      'SR1.5,3;SC0,10,-2,12;PA5,-1;CP-6,-1.3;'
        DB      'LBMILLISECONDS',ETX,'SP;PA10,12;SR:!'
        RET

MCHAR:  DS      33      ;START OF LABEL BUFFER

        END



*
        BLOCK DATA COMBLK
        COMMON/SHARED/XMIN,DIFF,NPTS
        END



*
FBO =SAPLT2
LBO SAPLT2/N,COMBLK,B:OVLAY,CURSOR,HP,B:HMUTIL1B,B:PLOTL326,B:AFU,FORLIB/B,SAPLT2/Y/E
B:FIXADR SAPLOT SAPLT2
FBO =SAPLOT
ERA SAPLOT.PGR
LBO SAPLOT/N,COMBLK,B:OVLAY,GETIT,B:HMUTIL1B,B:AFU,FORLIB/B,SAPLOT/E
ERA SAPLOT.REL
ERA SAPLT2.REL
```