NOAA Eastern Region Computer Programs
and Problems NWS ERCP - No. 41

SEVERE WEATHER POTENTIAL (SPOT) PLOTFILE GENERATOR

Ken LaPenta
National Weather Service Forecast Office
Albany, New York

Scientific Services Division
Eastern Region Headquarters
July 1987

U.S. DEPARTMENT OF COMMERCE / National Oceanic and Atmospheric Administration / National Weather Service

NOAA TECHNICAL MEMORANDUM

National Weather Service, Eastern Region Computer Programs and Problems

The Eastern Region Computer Programs and Problems (ERCP) series is a subset of the Eastern Region Technical Memorandum series. It will serve as the vehicle for the transfer of information about fully documented AFOS application programs. The format ERCP - No. 1 will serve as the model for future issuances in this series.

1 An AFOS version of the Flash Flood Checklist. Cynthia M. Scott, March 1981. (PB81 211252).

2 An AFOS Applications Program to Compute Three-Hourly Stream Stages. Alan P. Blackburn, September 1981. (PB82 156886).

3 PUPPY (AFOS Hydrologic Data Reporting Program). Daniel P. Provost, December 1981. (PB82 199720).

4 Special Search Computer Program. Alan P. Blackburn, April 1982. (PB83 175455).

5 Conversion of ALEMBIC$ Workbins. Alan P. Blackburn, October 1982. (PB83 138313).

6 Real-Time Quality Control of SAOs. John A. Billet, January 1983. (PB83 166082).

7 Automated Hourly Weather Collective from HRR Data Input. Lawrence Cedrone, January 1983 (PB83 167122).

8 Decoders for FRH, FTJ and FD Products. Cynthia M. Scott, February 1983. (PB83 176057).

9 Stability Analysis Program. Hugh M. Stone, March 1983. (PB83 197947).

10 Help for AFOS Message Comp. Alan P. Blackburn, May 1983. (PB83 213561).

11 Stability and Other Parameters from the First Transmission RAOB Data. Charles D. Little, May 1983. (PB83 220475).

12 TERR, PERR, and BIGC: Three Programs to Compute Verification Statistics. Matthew R. Peroutka, August 1983. (PB84 127521).

13 Decoder for Manually Digitized Radar Observations. Matthew R. Peroutka, June 1983. (PB84 127539).

14 Slick and Quick Data Entry for AFOS Era Verification (AEV) Program. Alan P. Blackburn, December 1983. (PB84 138726).

15 MDR--Processing Manually Digitized Radar Observations. Matthew R. Peroutka, November 1983. (PB84 161462) (Revised June 1985, PB85-220580/AS)

16 RANP: Stability Analysis Program. Hugh M. Stone, February 1984.(PB84 161447)

17 ZONES. Gerald G. Rigdon, March 1984. (PB84 174325)

18 Automated Analysis of Upper Air Soundings to Specify Precipitation Type. Joseph R. Bocchieri and Gerald G. Rigdon, March 1984. (PB84 174333)

(Continued on Inside Rear Cover)

NOAA Eastern Region Computer Programs and Problems NWS ERCP - No. 41

Severe Weather Potential (SPOT) Plotfile Generator

Ken LaPenta
National Weather Service Forecast Office
Albany, New York

Scientific Services Division
Eastern Region Headquarters
July 1987

Severe Weather Potential (SPOT)
Plotfile Generator

I. Introduction

Miller and Maddox (Maddox, 1973) developed a severe storm potential index (SPOT) computed from surface observations. Shulhan (1978) developed a technique for using the SPOT index to define areas of potential severe weather. Manually computing the indices required for a regional analysis is very time consuming. This AFOS applications program calculates the index values and generates a plotfile so a regional map can be created quickly.

II. Methodology and Software Structure

A. Methodology

The SPOT index as developed by Miller and Maddox is computed from surface observations according to the following equation:

$$SPOT = (T-60) + (TD-55) + 100*(30.00-P) + F(v) \quad (1)$$

where the variables are defined as follows:

T = surface temperature in degrees F
TD = surface dewpoint in degrees F
P = altimeter setting in inches
F(v)= wind speed term derived from table below (v is surface wind speed or gust if reported)

		WIND DIRECTION IN DEGREES					
		GT360 LT070	GE070 LE140	GT140 LE200	GT200 LE230	GT230 LE260	GT260 LE360
SFC TD (F)	GE60	0	v	if Ts at Sta v if not 2v	v	v	-2v
	LT60 GE55	0	v	same as above	v/2	-v	-2v
	LT55	0	v	v	0	-2v	-2v

note: Ts = thunderstorm Sta = station

The Shulhan spot advection technique for areas involves contouring SPOT values, analyzing streamlines and measuring the SPOT gradient along the streamlines over 1/2 degree of latitude. Shulhan applied this technique for 19 severe weather cases in the northeastern United States. He concluded that the technique correctly identified areas of severe storm occurrence 55% of the time. Severe storms were most likely in or near areas with positive SPOT advection of 20 or more units per 1/2 degree of latitude. Thus, a tight SPOT gradient rather than a particular value is important. Still, about half of the storms developed near the SPOT=60 isopleth. Local experience with SPOT application has shown that severe weather also occurs just ahead of a SPOT maximum which is followed by a strong decrease in SPOT values (negative gradient).

B. Software Structure

SPOT will calculate the SPOT index from the current surface observations at preselected stations. The index values are written into an AFOS plotfile along with the station wind direction and speed.

SPOT reads the stations to be plotted along with location and zoom information from an RDOS file SPOTIN.DT. This file can be created with an RDOS text editor or M:F/ on an ADM.

The first line contains a three-digit number indicating how many stations are to be plotted. SPOT can handle up to 150 stations. Use leading zeroes if the number takes less than three digits (e. g. for twenty stations use 020, not 20).

Succeeding lines contain information about the stations to be plotted, one line per station. The first 9 characters of the line are the AFOS key for the surface observation (cccSAOxxx). Character 10 is a blank. Characters 11 through 16 contain the "PSOWDT" data for each station where:

P = plot priority threshold
  = 0 display at all zoom levels
  = 1 display at 4:1 or higher
  = 2 display at 9:1 or higher
  = 3 display at 16:1 or higher

S = size of fonts (use 0 for normal size text)

O = orientation (use 0; this parameter doesn't affect GDM display)

W = special fonts (use 0; no difference with this plot)

D = wind group plotting convention (use 1)

T = type of report (use 2)

The next 8 characters are the X and Y pixel coordinates (4 digits each) for the station. Leading zeros should be used to fill all four

positions for each value. These values, as well as those for PSOWDT, can be taken from any B02 surface plotfile (such as NMCPLTP0A). The coordinates can also be taken from the station directory file STDIR. You may want to adjust the zoom values to account for station density on the final graphic. Figure 1 shows a sample SPOTIN.DT.

After reading the station information from SPOTIN.DT, SPOT calls the AIRXX subroutine written by Rich Thomas to decode each surface observation. The program checks to make sure that each observations is lessI than an hour old. Specials can't be used because there is no temperature or dewpoint. If SPOT encounters a special, it'll look at up to three previous versions to find a record or record special observation.

The program makes several quick checks to make sure the data is fairly reasonable. If the data is bad or an observation is missing or too old, no SPOT index is calculated and nothing is written to the plotfile for that station.

When all the SPOT values have been calculated they are written to the output file NMCPLTSA0 along with the wind directions and speeds. (If gusts are reported, they are used rather than the average speed.) Using the routine FSTORE, the file is stored in the AFOS data base as product NMCPLTSA0. PMOD can then be used to plot the values on a regional background. Figure 2 shows a finished graphic on background B31. Contours and streamlines for the Shulhan technique have been drawn by hand.

The following subroutines are used by SPOT:

CURJTIME (written by Jack May) - figures the current Julian time from the station clock and returns the value to the main program in variable CURJT.

KEYJTIME (written by Jack May) - retrieves the Julian time in minutes since midnight January 1st for a given product from DATAKEY0.

AIRXX (written by Rich Thomas) - accepts an array TDATA containing an unpacked SA0 and decodes the observation. Decoded ob is returned to the program in array IBUF.

ANDEQ (written by Rich Thomas) - a logical IF (if A=B and C=D then IV1=IV2).

ANDGO (written by Rich Thomas) - a logical statement (if A=B and C=D then GO TO statement ISTN).

ORGO (written by Rich Thomas) - a logical statement (if A=B or C=D then GO TO statement ISTN).

NUMBR (written by Rich Thomas) - determines if A is a number, returns to statement C if it is. If not, returns to statement B.

III. Cautions and Restrictions

Shulhan's SPOT index technique was developed for the northeast part of the country and studies indicated it showed skill in forecasting severe weather there. It may be applicable to other parts of the country.

The SPOT values are dependent on the correctness of the observations. AIRXX will accurately decode just about all SAO's. Several quality control checks have been incorporated into the main program to further examine the validity of various elements. The checks are rudimentary and may not catch all errors. For example, mistyped data in a SAO that isn't too far in error will escape detection.

Specials can't be used to calculate the SPOT index. When the program encounters a special, it'll check up to three previous versions of the product for a record or record special. If more than three specials have been sent in a hour before the program is run, no SPOT index will be calculated for that particular station.

The program flags Canadian stations for only three nodes: WUL, YYZ and WHX. If the program encounters an SAO from these nodes, Celsius temperature and dewpoint are converted to Fahrenheit. Other Canadian observations will not be converted.

IV. References

Maddox, R. A., A Severe Thunderstorm Surface Potential Index (SPOT), Eighth Conference on Severe Local Storms, 1973, Denver, CO.

Shulhan, I. P., SPOT Index Advection - An Indication of Severe Storm Potential, Aerospace Sciences Technical Attachment M-12, September 1978.

V.

ERCP #41
July 1987

SPOT (Severe Weather Potential) Index Plot

PART A: INFORMATION AND INSTALLATION

PROGRAM NAME: SPOT

AAL ID:
REVISION NO.: 1.00

PURPOSE: Creates a B02 plotfile containing the SPOT indexes and observed winds for selected sites. This can be plotted on a regional background using PMOD.

PROGRAM INFORMATION:

Development Programmer:
Ken LaPenta
Location: WSFO ALB
Phone: (FTS) 562-8586
Language: FORTRAN IV/5.57
Save File Creation Date(s): 06/02/87

Maintenance Programmer:
Ken LaPenta
Location: WSFO ALB
Phone: (FTS) 562-8586
Type: Standard

Running Time: Depends on number of stations used (82 stations take 2-3 min)
Disk Space:

Program	62 RDOS blocks
Data	around 1 block for 20 stations

PROGRAM REQUIREMENTS

Program Files:

Name	DP Location	Comments
SPOT.SV	APPL1	link from SYSZ

Data Files:

Name	DP Location	R/W	Comments
SPOTIN.DT	APPL1	R	link from SYSZ
NMCPLTSAO	SYSZ	W	temporary output file

AFOS Products:

ID	Action	Comments
NMCPLTSAO	Store	Output plotfile

LOAD LINE

```
RLDR/P SPOT CURJTIME KEYJTIME AIRXX ANDEQ ANDGO ORGO NUMBR °
SPOTREV <BG UTIL FORT>.LB SPOT.LS/L
```

PROGRAM INSTALLATION

1. Move SPOT.SV to APPL1 and create a link to it in SYSZ.

2. Create the file SPOTIN.DT with M:F/ or a text editor. SPOTIN.DT's first line contains the number of stations to plot (as a three-digit number). Each following line contains the AFOS key of the station's observation, a space, and then the PSOWDT, and B02 X-coordinate and Y-coordinate all together. The last three items can be taken from any B02 plotfile (such as NMCPLTP0A).

3. Make sure that NMCPLTSAO is in the database. You should also choose a graphic for the SPOT output graphic and put that in the database, if necessary. Use the KEY: command to set the appropriate map background.

4. Write a macro to run SPOT and then create the graphic:

```
SPOT
PMOD SAO NAxx.PF/T SFC.PM/O
GENUTF XPLOT yyy
```

where xx is the number of the regional background you want to use and yyy is the output graphic (set to background xx). You can also plot SAO on B02 using PMOD with NA.PF/T or REGPLOT.

SPOT (Severe Weather Potential) Index Plot

PART B: EXECUTION AND ERROR CONDITIONS

PROGRAM NAME: SPOT.SV

AAL ID:
REVISION NO.: 1.00

PROGRAM EXECUTION:

The SPOT program creates a plotfile in NMCPLTSAO. PMOD and GENUTF must then be used to plot it. The most convenient way to produce the graphic is a macro that runs all three programs:

```
SPOT
PMOD SAO NAxx.PF/T SFC.PM/0
GENUTF XPLOT yyy
```

where xx is the number of the regional background you want to use and yyy is the output graphic (set to background xx). You can also plot SAO on B02 using PMOD with NA.PF/T or REGPLOT.

ERROR CONDITIONS

Fatal error messages are written to the Dasher. If an observation is old or cannot be decoded, no SPOT index will be calculated for that station.

Messages from ADM	Meaning
None	

Dasher Messages	Meaning
ERROR IN DATE/TIME	error in calling subroutines DATE or TIME
ERROR IN KSRCF	can't find observation in database
ERROR IN RDBKF	can't read observation from database
ERROR IN WRS	can't write to file NMCPLTSAO
ERROR IN FSTOR	can't store NMCPLTSAO in database

VI. Figures

```
082
ALBSAOALB 01001214000984
ALBSAOBGM 01001212620894
ALBSAOBTV 11001213961156
ALBSAOGFL 11011213961044
ALBSAOMPV 01021214421144
ALBSAOMSS 01001212721168
ALBSAOPBG 01001213701166
ALBSAOPOU 01011214220878
ALBSAORME 11001212701000
ALBSAOSLK 01001213281128
ALBSAOSWF 11011214120862
ALBSAOUCA 11001212800992
BOSSAOBAF 11021214900952
BOSSAOBOL 01031215000932
BOSSAOBOR 11011214900850
BOSSAOBOS 11001216001000
BOSSAOCEF 11041215020960
BOSSAODXR 11041214600862
BOSSAOFMH 11011216580956
BOSSAOGON 11011215600890
BOSSAOHVN 11011215040864
BOSSAOHYA 11011216740962
BOSSAOORH 01031215[illegible]
BOSSAOPVO [illegible]
BUF[illegible]
          [illegible]216201216
[illegible]GK 01011216641284
FWMSAOCON 01011215421076
FWMSAOLEB 01011214761098
FWMSAOMHT 01001215541052
FWMSAOPSM 01001215921080
WULSAOWOH 01031212801280
WULSAOWXF 11001214141230
WULSAOYHU 11001213481248
WULSAOYQB 01001214401404
WULSAOYSC 01021214641274
WULSAOYUL 01001213281238
YYZSAOYGK 11001211721072
YYZSAOYOW 01001212081190
YYZSAOYQA 01001209761104
YYZSAOYTR 11001211121048
YYZSAOYWA 01001210881222
YYZSAOYXU 01001208820898
YYZSAOYYB 01001209441234
YYZSAOYYZ 01001209780978
```

Figure 1. Sample SPOTIN.DT (shortened for publication)

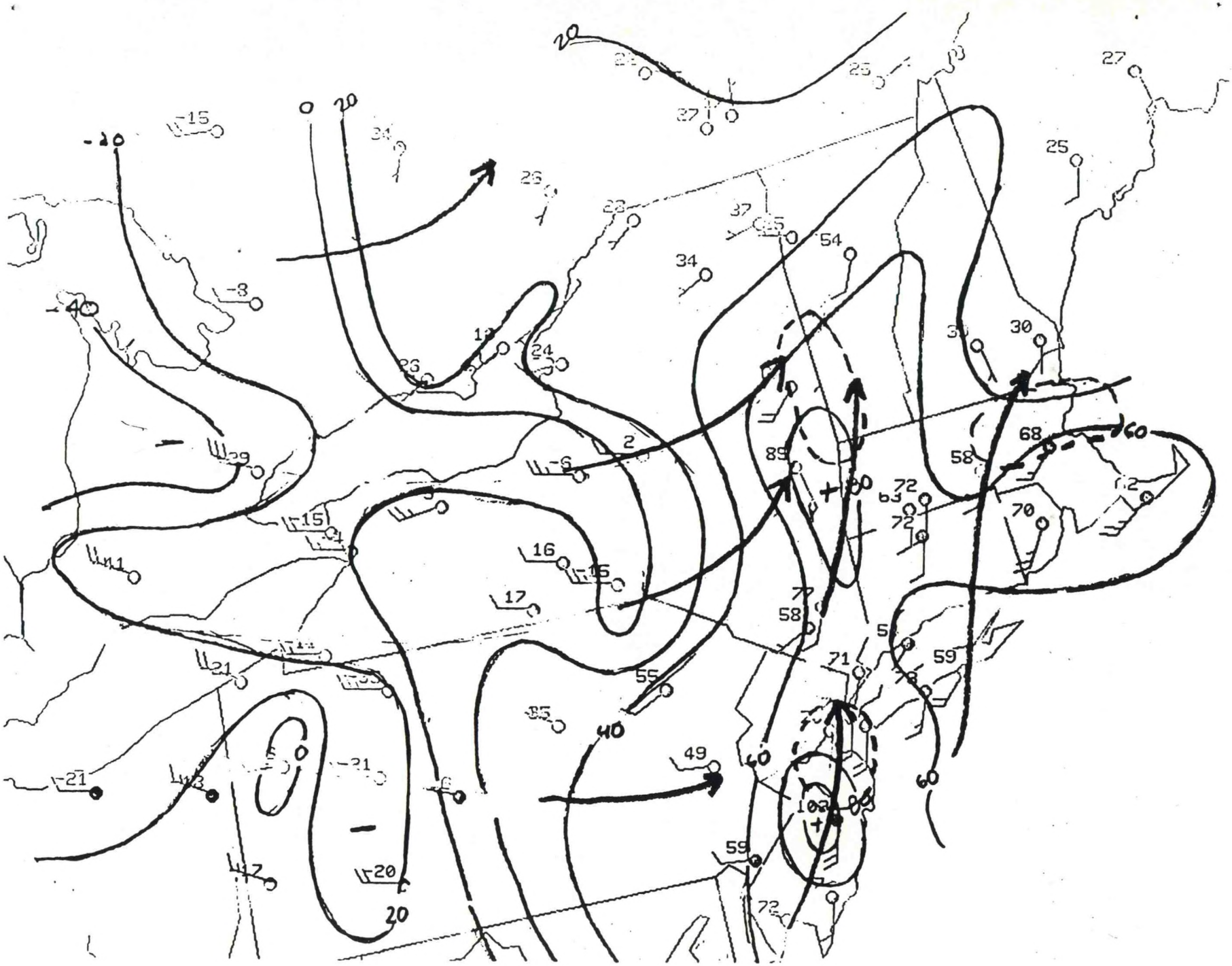

Figure 2. SPOT Graphic (contours and streamlines drawn by hand)

VII. Source Code

```
C     SPOT                                              REVISION 1.00
C     OCT 1985              KEN LAPENTA                 WSFO ALB FTS 8-562-6586
C     DG FORTRAN IV/5.20  DG ECLIPSE 230                RDOS REV 7.20
C     RLDR SPOT CURJTIME KEYJTIME AIRXX ANDEQ ANDGO ORGO NUMBR
C     BG.LB UTIL.LB FORT.LB AFOSE.LB
C     SUBROUTINES
C         AIRXX - DEVELOPED BY RICH THOMAS - DECODES SURFACE
C                 OBS.  CALLS ON SUBROUTINES ANDEQ ANGO ORGO NUMBR
C         CURJTIME AND KEYJTIME - DEVELOPED BY JACK MAY - RETURN
C                 CURRENT JULIAN TIME AND JULIAN TIME OF A AFOS
C                 PRODUCT
C
C     PURPOSE - TO CREATE AN AFOS PLOTFILE (NMCPLTSAO) OF
C     SPOT INDEX VALUES THAT WILL BE PLOOTTED USING REGPLOT
C
C
      DIMENSION D(150),V(150),KEYREC(25),
     1ID(150),IA(8,150),IH(12),IG(6),IZ(2),I9(2),IHH(12,150)
       INTEGER X(150),X1(150),Y(150),Y1(150),IBUF(60),TDATA(256),
     1IDATA(128),SPOT(150),NOW(3),TODAY(3),CAN,
     2JBUF(48),KBUF(24),KEY(5),IU(15),IV(16),IUU(8),IVV(8)
      COMMON/SPOT/IHDR(10),IHDS(19),IHDU(22),IHDV(9),IEND(4)
       DATA IHDR/"NMCPLTSA0000",177777K,177777K,2400K,142600K/
       DATA IHDS/"19,290,00000Z,SHULHAN  SPOT  INDEX ;",6412K/
      DATA IHDU/"NMCPLTSA0001020002048153628501425009750168 8"/
      DATA IHDV/"119,250,00000Z,RUN"/
      DATA IEND/5105K,"ND",142400K,203K/
C READ IN INPUT DATA ON STATIONS FROM SPOTIN.DT
       CALL GCHN(ICH,IER)  ;OPEN CHANNEL FOR READING IN STATION DATA
       CALL OPEN(ICH,"SPOTIN.DT",1,IER)
       READ (ICH,10) NNN  ;NNN=NUMBER OF STATIONS TO BE READ IN
   10 FORMAT (I3)
      DO 20 I=1,NNN
      READ (ICH,11) ((IA(J,I),J=1,8),X(I),X1(I),Y(I),Y1(I))
   11 FORMAT (8A2,4I2)
   20 CONTINUE
       CALL CLOSE(ICH,IER)  ;CLOSE CHANNEL TO SPOTIN.DT
       DO 90 I=1,NNN
      N3=0
      DO 25 J=6,8
      IG(J-5)=IA(J,I)
   25 CONTINUE
      CALL UNPACK(IG,6,IH)
      DO 30 J=1,12
      IHH(J,I)=IH(J)
   30 CONTINUE
       CAN=0  ;FLAG FOR CANADIAN DATA
      N2=-1
       DO 55 I1=1,5
       KEY(I1)=IA(I1,I)
   55 CONTINUE
C CHECK TO SEE IF OBSERVATIONS IS LESS THAN 1 HOUR OLD
      CALL KSRCF(KEY,KEYREC,IER)
      IF(IER.NE.1) GO TO 501
       CALL DATE(TODAY,IER)
      IF (IER.NE.1) GO TO 500
       CALL TIME(NOW,IER)
       IF (IER.NE.1) GO TO 500
       CALL CURJTIME(TODAY,NOW(1),NOW(2),NOW(3),CURJT)
       CALL KEYJTIME(KEY,PRODJT,IER)
```

```
      TDIFF=CURJT-PRODJT  ;CHECK TO SEE IF SFC OB IS OLD
       IF (TDIFF.GT.60) GO TO 80
C READS OBSERVATIONS
      CALL RDBKF(0,IDATA,IER)  ;READ SFC OB
       IF(IER.NE.1) GO TO 502
   56  CALL UNPACK(IDATA,256,TDATA)
       IF (IA(I,I).EQ.'WU'.OR.IA(1,I).EQ.'YY') CAN=1
      IF (IA(1,I).EQ.'WH') CAN=1
       CALL AIRXX(TDATA,IBUF,CAN,IER)  ;DECODE SFC OB
       IF(IER.NE.1) GO TO 70
       IF (IBUF(6).EQ.3) GO TO 70  ;CHECK FOR SPECIALS
        IGUST=0
       W=0  ; EQUALS 1 IF TSTM AT STATION
       T=IBUF(35)  ;TEMP
       TD=IBUF(36)  ;DEWPOINT
       P=IBUF(41)  ;ALTIMETER SETTING
       D(I)=IBUF(38)  ;WIND DIRECTION
       V(I)=IBUF(39)  ;WIND SPEED
      IGUST=IBUF(40)
      IF (IBUF(40).GT.V(I)) V(I)=IBUF(40)
      IF (V(I).GT.85) V(I)=85
C  CHECKS FOR BAD SURFACE DATA
      IF (D(I).LT.0) GO TO 80
       IF (D(I).GT.360) GO TO 80
      IF (V(I).LT.0) GO TO 80
      IF (T.LT.-30) GO TO 80
      IF (TD.LT.-30) GO TO 80
       DO 60 I2=24,33
       IF (IBUF(I2).EQ.84) W=1  ;TSTM AT STATION
   60 CONTINUE
       P=P/100
C FIND  VF  WHICH IS USED TO CALCULATED THE SPOT INDEX
C VF  IS A FUNCTION OF D, V, W, AND TD
       IF (D(I).GE.70) GO TO 201
       VF=0.0
       GO TO 250
  201 IF (D(I).GT.140) GO TO 202
       VF=V(I)
       GO TO 250
  202 IF (D(I).GT.200) GO TO 203
       IF (TD.GE.55.AND.W.EQ.1) VF=V(I)
       IF (TD.GE.55.AND.W.EQ.0) VF=2*V(I)
       IF (TD.LT.55) VF=V(I)
       GO TO 250
  203 IF (D(I).GT.230) GO TO 204
      IF (TD.LT.55) VF=0
      IF (TD.GE.55.AND.TD.LT.60) VF=V(I)/2
      IF (TD.GE.60) VF=V(I)
      GO TO 250
  204 IF (D(I).GT.260) GO TO 205
      IF (TD.GE.55.AND.TD.LT.60) VF=-1*V(I)
      IF (TD.GE.60) VF=V(I)
      IF (TD.LT.55) VF=-2*V(I)
      GO TO 250
  205 VF=-2*V(I)
  250 CONTINUE
C CALCULATE SPOT INDEX
      XSPOT=(T-60)+(TD-55)+100*(30.00-P)+VF
      SPOT(I)=XSPOT+.5  ;ROUND OFF TO WHOLE NUMBR
       IF (SPOT(I).LT.0) SPOT(I)=SPOT(I)-1
```

```
      IF (SPOT(I).LT.-300) GO TO 80
      GO TO 90
   70 IF (N3.EQ.3) GO TO 80
      CALL PRVRF(IER)  ;IF OBSERVATION IS A SPECIAL PREV VERSION CHECKED
      IF(IER.NE.1) GO TO 80
      CALL RDBKF(0,IDATA,IER)
      IF (IER.NE.1) GO TO 502
      PRODJT=ISHFT(IDATA(9),-8)*16384.+IAND(IDATA(9),377K)*128.+
     1ISHFT(IDATA(10),-8)
      IF (CURJT.GT.PRODJT+60) GO TO 80
      N3=N3+1
      GO TO 56
   80 SPOT(I)=999
   90 CONTINUE
      CALL GCHN(ICH,IER)  ;GET CHANNEL FOR WRITING OUT DATA
      CALL DFILW("NMCPLTSAO",IER)
      CALL CRAND("NMCPLTSAO",IER)
C  CREATE PLOTFILE TO BE USED IN GENERATING GRAPHIC
       CALL OPENN(ICH,"NMCPLTSAO",IER)
C  DETERMINE DATE/TIME INFORMATION FOR PLOTFILE HEADERS
      TODAY(3)=TODAY(3)-1900
      IU(5)=INT(TODAY(1)/10.)
      IU(6)=TODAY(1)-IU(5)*10
      IU(3)=INT(TODAY(2)/10.)
      IU(4)=TODAY(2)-IU(3)*10
      IU(7)=INT(TODAY(3)/10.)
      IU(8)=TODAY(3)-10*IU(7)
      IU(1)=INT(NOW(1)/10.)
      IU(2)=NOW(1)-10*IU(1)
      IU(9)=0
      IU(10)=0
      IU(11)=0
      IU(12)=0
      IU(13)=15K
      IU(14)=12K
      IU(15)=61K
C  CHANGE NUMERICAL VALUES TO ASCII EQUIVALENTS
      DO 126 I=1,12
      IF (IU(I).EQ.0) IU(I)=60K
      IF (IU(I).EQ.1) IU(I)=61K
      IF (IU(I).EQ.2) IU(I)=62K
      IF (IU(I).EQ.3) IU(I)=63K
      IF (IU(I).EQ.4) IU(I)=64K
      IF (IU(I).EQ.5) IU(I)=65K
      IF (IU(I).EQ.6) IU(I)=66K
      IF (IU(I).EQ.7) IU(I)=67K
      IF (IU(I).EQ.8) IU(I)=70K
      IF (IU(I).EQ.9) IU(I)=71K
  126 CONTINUE
      CALL PACK(IU,15,IUU)
      IV(1)=40K
      IV(4)=132K
      IV(5)=40K
      IV(8)=40K
      IV(11)=40K
      IV(2)=IU(1)
      IV(3)=IU(2)
      IV(6)=IU(5)
      IV(7)=IU(6)
      IV(9)=IU(3)
```

```
      IV(10)=IU(4)
      IV(12)=IU(7)
      IV(13)=IU(8)
      IV(14)=73K
      IV(15)=15K
      IV(16)=12K
C  WRITE PLOTFILE HEADERS
      CALL PACK(IV,16,IVV)
       CALL WRS(ICH,IHDR,20,IER)
      CALL WRS(ICH,IHDU,43,IER)
      CALL WRS(ICH,IUU,15,IER)
       CALL WRS(ICH,IHDS,38,IER)
      CALL WRS(ICH,IHDV,18,IER)
      CALL WRS(ICH,IVV,16,IER)
      DO 400 I=1,NNN
       IF (SPOT(I).GT.900) GO TO 400  ;DON/T PLOT MISSING DATA
       IHOLD=0
       IHOLD1=0
C  GENERATE DATA LINE FOR CURRENT SFC OB
      DO 120 J=11,16
      JBUF(J)=IHH(J-10,I)
  120 CONTINUE
      JBUF(18)=54K
      JBUF(17)=54K
      CALL UNPACK(IA(4,I),2,IZ)
      CALL UNPACK(IA(5,I),2,I9)
      JBUF(19)=IZ(1)
      JBUF(20)=IZ(2)
      JBUF(21)=I9(1)
      JBUF(22)=54K
      JBUF(23)=8
      JBUF(24)=54K
C  WRITE WIND INFO TO PLOTFILE
      JBUF(25)=INT(D(I)/100.)
      ITMP=D(I)-JBUF(25)*100
      JBUF(26)=INT(ITMP/10.)
      JBUF(27)=0
      JBUF(28)=INT(V(I)/10.)
      JBUF(29)=V(I)-JBUF(28)*10
      JBUF(30)=54K
      JBUF(31)=54K
      DO 121 J=36,44
      JBUF(J)=54K
  121 CONTINUE
       JBUF(45)=73K
       IF (SPOT(I).LT.0) IHOLD1=1
       IF (IHOLD1.EQ.1) SPOT(I)=SPOT(I)*N2
       JBUF(1)=INT(X(I)/10.)  ;X COORDINATE
       JBUF(2)=X(I)-JBUF(1)*10
       JBUF(3)=INT(X1(I)/10.)
       JBUF(4)=X1(I)-JBUF(3)*10
       JBUF(5)=54K
      JBUF(6)=INT(Y(I)/10.)  ;Y COORDINATE
      JBUF(7)=Y(I)-10*JBUF(6)
      JBUF(8)=INT(Y1(I)/10.)
      JBUF(9)=Y1(I)-JBUF(8)*10
      JBUF(10)=54K
      JBUF(47)=15K  ; CARRIAGE RETURN/LF
      JBUF(48)=12K
      DO 130 M1=1,2
```

```
      M=M1+(M1-1)*4
      IF (JBUF(M).EQ.0) JBUF(M)=40K
      IF (JBUF(M+1).EQ.0.AND.JBUF(M).EQ.40K) JBUF(M+1)=40K
      IF (JBUF(M+2).EQ.0.AND.JBUF(M+1).EQ.40K) JBUF(M+2)=40K
  130 CONTINUE
      JBUF(32)=INT(SPOT(I)/1000.)  ;SPOT INDEX VALUE
      ITMP=SPOT(I)-JBUF(32)*1000
      JBUF(33)=INT(ITMP/100.)
      ITMP=ITMP-JBUF(33)*100
      JBUF(34)=INT(ITMP/10.)
      JBUF(35)=ITMP-JBUF(34)*10
      IF (JBUF(32).EQ.0) JBUF(32)=40K
      IF (JBUF(32).EQ.40K.AND.JBUF(33).EQ.0) JBUF(33)=40K
      IF (JBUF(32).EQ.40K.AND.JBUF(33).EQ.40K) IHOLD=1
      IF (IHOLD.EQ.1.AND.JBUF(34).EQ.0) JBUF(34)=40K
      IF (SPOT(I).LT.10) IHOLD2=34
      IF (SPOT(I).LT.100.AND.SPOT(I).GE.10) IHOLD2=33
      IF (SPOT(I).LT.1000.AND.SPOT(I).GE.100) IHOLD2=32
      IF (IHOLD1.EQ.1) JBUF(IHOLD2)=55K
      IF (IHOLD1.EQ.1) SPOT(I)=SPOT(I)*N2
      DO 390 J=1,48  ;CHNGE NUMERICAL VALUES TO ASCII EQUIVALENTS
      IF(JBUF(J).EQ.0) JBUF(J)=60K
      IF(JBUF(J).EQ.1) JBUF(J)=61K
      IF(JBUF(J).EQ.2) JBUF(J)=62K
      IF(JBUF(J).EQ.3) JBUF(J)=63K
      IF(JBUF(J).EQ.4) JBUF(J)=64K
      IF(JBUF(J).EQ.5) JBUF(J)=65K
      IF(JBUF(J).EQ.6) JBUF(J)=66K
      IF(JBUF(J).EQ.7) JBUF(J)=67K
      IF(JBUF(J).EQ.8) JBUF(J)=70K
      IF(JBUF(J).EQ.9) JBUF(J)=71K
  390 CONTINUE
      JBUF(46)=40K
      CALL PACK(JBUF,48,KBUF)
      CALL WRS(ICH,KBUF,48,IER)  ;WRITE DATA LINE FOR CURRENT SFC OB
      IF(IER.NE.1) GO TO 504
  400 CONTINUE
      CALL WRS(ICH,IEND,8,IER)  ;WRITE ENDING
      IF (IER.NE.1) GO TO 504
      CALL RESET
      CALL FSTOR('NMCPLTSAO',0,IER)  ;STORE FILE AS AFOS PRODUCT
      IF (IER.NE.1) GO TO 505
      CALL FORK('SPOT',IER)
      GO TO 600
C  ERROR RETURNS
  500 TYPE 'ERROR IN DATE/TIME'
      GO TO 600
  501 TYPE 'ERROR IN KSRCF'
      GO TO 600
  502 TYPE 'ERROR IN RDBKF'
      GO TO 600
  504 TYPE 'ERROR IN WRS'
      GO TO 600
  505 TYPE 'ERROR IN FSTOR'
  600 CONTINUE
      END
```

```
      SUBROUTINE CURJTIME(NDATE,IHOUR,MIN,ISEC,CURJT)
C CURJTIME FIGURES THE CURRENT JULIAN MINUTE FROM THE STATION CLOCK.
C CURJT IS THE VARIABLE PASSED BACK TO THE PROGRAM
C
      DIMENSION NDATE(3)
C
      MONTH=NDATE(1)
      IDATE=NDATE(2)
      IYEAR=NDATE(3)
C
C FIGURE LEAP YEAR
      LYEAR=IYEAR-(4*(IFIX(IYEAR/4.)))          ; IF LYEAR=0, LEAPYEAR
C
      CURJT=0
      IF(MONTH.GE.2) CURJT=CURJT+44640.         ;ADD JAN MINUES
      IF(MONTH.GE.3) CURJT=CURJT+40320.         ;ADD FEB MINUES
      IF(LYEAR.EQ.0.AND.MONTH.GT.2) CURJT=CURJT+1440 ;ADD LEAP YEAR MINUT
      IF(MONTH.GE.4) CURJT=CURJT+44640.         ;ADD MAR MINUTES
      IF(MONTH.GE.5) CURJT=CURJT+43200.         ;ADD APR MINUTES
      IF(MONTH.GE.6) CURJT=CURJT+44640.         ;ADD MAY MINUTES
      IF(MONTH.GE.7) CURJT=CURJT+43200.         ;ADD JUN MINUTES
      IF(MONTH.GE.8) CURJT=CURJT+44640.         ;ADD JUL MINUTES
      IF(MONTH.GE.9) CURJT=CURJT+44640.           ;ADD AUG MINUTES
      IF(MONTH.GE.10) CURJT=CURJT+43200.        ;ADD SEP MINUTES
      IF(MONTH.GE.11) CURJT=CURJT+44640.        ;ADD OCT MINUTES
      IF(MONTH.GE.12) CURJT=CURJT+43200.          ;ADD NOV MINUTES
C
C
      CURJT=CURJ+(IDATE-1)*1440.                  ;ADD DAYS SINCE LAST M1
      CURJT=CURJT+(IHOUR*60.)                   ;# OF HRS PAST MIDN
      CURJT=CURJT+FLOAT(MIN)                    ;ADD NUMBER OF MIN
C
      RETURN
      END
```

```
      SUBROUTINE KEYJTIME(KEY,PRODJT,IER)
C
C THIS SUBROUTINE RETRIEVES THE JULIAN TIME (MINUTES SINCE MIDNIGHT
C JANUARY 1ST) FROM DATAKEY0 OF THE PRODUCT DEFINED IN THE KEY.
C
C JACK MAY/ WSFO CLEVELAND/
C
      DIMENSION KEY(5),KREC(20)
      INTEGER UNKREC(40),A0,A1,A2
C
C PUT KEY RECORD INTO VARIABLE ARRAY KREC AND UNPACK INTO ARRAY UNKREC
      CALL KSRCF(KEY,KREC,IER)
      IF (IER.NE.1) GO TO 900
      CALL UNPACK(KREC,40,UNREC)
C
C JULIAN TIME NOW CONTAINED IN THREE WORDS OF UNKREC (WORDS 19,20,21)
C BELOW THEY ARE DEFINED AS A0,A1, AND A2
C
C JULIAN TIME = A0(2**14)+A1(2**7)+A2
C
      A0=UNKREC(19)
      A1=UNKREC(20)
      A2=UNKREC(21)
C
      PRODJT=0
      XNUM1=A0*(2.**14)
      XNUM2=A1*(2.**7)54
      XNUM3=A2
      PRODJT=XNUM1+XNUM2+XNUM3
C
      IER=1
      RETURN
  900 IER=0
      RETURN
      END
```

```
C SUBROUTINE ACCEPTS ARRAY TADATA(256) WHICH CONTAINS AN
C UNPACKED SAO CALLED FROM MAIN PROGRAM,
C SAO IS DECODED AND RETURNED IN ARRAY ABUF(60) - MIXED
C INTEGER/ASCII FORM.  CAN FLAG INDICATING SAO IS US(0)
C OR CANADIAN (1)..PROGRAMMER RICH THOMAS SXB.ISL.SDO 9/79
C REQUIRES SUBROUTINES ANDEQ ANDGO ORGO NUMBR
      SUBROUTINE AIRXX(TDATA,IBUF,CAN,IER)
      REAL LEVAP
      INTEGER V1,V2,VX,VY,P,Q,R,T,HGT(3)
      INTEGER TDATA(256),IBUF(60),CX
      INTEGER CP,CP1,CP2,CP3,CP4,CAN
      INUM(IA,IB,IC,ID)=(IA-48)*1000+(IB-48)*100+(IC-48)*10+ID-48
C     THIS FUNCTION CONVERTS ASCII TO INTEGER
C     BEGIN DECODING
      IER=0
     CHBL0=23
C     INITIALIZE ARRAY TO BE USED FOR FORMATTED DATA
      DO 80 I=1,60
   80 IBUF(I)=-99     ;  -99 DENOTES MISSING FOR INTEGERS
      DO 85 N=1,4
   85 IBUF(N)=32      ;  ASCII BLANKS
      DO 90 N=24,33
   90 IBUF(N)=32
      DO 95 N=44,49
   95 IBUF(N)=32
      DO 100 N=10,18,4
      IBUF(N)=32
      IBUF(N+1)=0
      IBUF(N+2)=0
  100 IBUF(N+3)=32
      IBUF(23)=32
      IBUF(53)=32
      IBUF(54)=32
      IBUF(55)=32
      M=0
      IC=CHBL0+21
      JC=CHBL0+31
      DO 110 CP=IC,JC      ; CP IS CHARACTER POINTER IN DATA
      IF(TDATA(CP).GT.47)GO TO 105       ;LETTER OR NUMBER  (ID)
      IF(M.GT.0)GO TO 115     ; IF M=0 NO LETTERS/NUMBERS FOUND SO FAR
      GO TO 110
  105 M=M+1        ;INCREMENT INDEX - LETTER/NUMBER FOUND
      IBUF(M)=TDATA(CP)        ;LOAD INTO IBUF
  110 CONTINUE
      IER=4        ;    IER=4     ID NOT FOUND;(OR UNKNOWN OB TYPE)
      RETURN
C     CP IS LOCATION OF BLANK AFTER ID
C     GET OB TYPE
  115 J=CP+1        ;  ADVANCE ONE CHARACTER POSITION
      K=J+8         ; SEARCH FOR OB TYPE IS LIMITED TO NEXT 8 CHARACTERS
      IBUF(6)=0       ; INITIALIZE TO 0
      DO 120 CP=J,K
      CP1=CP+1
      ITM=TDATA(CP)
      ITM2=TDATA(CP1)
      IF(ITM.EQ.83)GO TO 130     ; S- TYPE
      VX=82       ;'R'
      VY=83       ;'S'
      CALL ANDGO(ITM,VX,ITM2,VY,$125)     ; RS TYPE
      V2=4
      VY=65
      CALL ANDEQ(ITM,VX,ITM2,VY,IBUF(6),V2)  ;UNMANNED RAMOS
      IF(ITM.EQ.65)IBUF(6)=4   ;UNMANNED AMOS OR AUTOB
      IF(ITM.EQ.32)GO TO 120
      GO TO 135
  120 CONTINUE
```

```
125 IBUF(6)=2      ;RS
130 IF(ITM2.EQ.65)IBUF(6)=1     ;SA
     IF(ITM2.EQ.80)IBUF(6)=3     ;SP
     IF(ITM2.EQ.87)IBUF(6)=6     ;SW
135 IF(IBUF(6).EQ.0)IER=4       ;IER=4 UNKNOWN OB TYPE;(0R ID NOT FOUND)
     MINCP=CP1+3      ;SET MIN CP FOR RAMOS/AMOS (UNMANNED)
     IF(IBUF(6).EQ.4)GO TO 255      ;NO TIME IN UNMANNED AMOS/RAMOS OBS
     J=CP+2
     K=J+8
     DO 200 CP=J,K
     CP1=CP+1
     CP2=CP+2
     CP3=CP+3
     CALL NUMBR(TDATA(CP),$200,$210)      ;CHECK FOR A 4 NUMBER SEQUENCE
200 CONTINUE
205 IER=5       ;   IER=5  ERROR IN LOCATING TIME
     GO TO 270
210 DO 220 JCP=CP1,CP3
     IF(TDATA(JCP)-48)205,215,215    ;BETWEEN 0
215 IF(TDATA(JCP)-57)220,220,205    ; AND 9?
220 CONTINUE
    IBUF(5)=INUM(TDATA(CP),TDATA(CP1),TDATA(CP2),TDATA(CP3))   ; TIME
     IHOUR=0
     IHOUR=IBUF(5)/100
     IF(IBUF(5)-IHOUR*100.GE.45)IHOUR=IHOUR+1   ; TIME JUST BEFORE HOUR
     J=CP3+1
     K=J+8
     DO 230 CP=J,K     ;CVHECK FOR RAMOS/AMOS
     IF(TDATA(CP).NE.32)GO TO 235
230 CONTINUE
     MINCP=CP3
     GO TO 205
235 VX=82      ;'R'
     VY=65      ;'A'      RAMOS
     V1=TDATA(CP)
     CALL ORGO(V1,VX,V1,VY,$240) ;'R--' OR 'A---' ? IF SO GO TO 240
     MINCP=CP-1
     GO TO 255
240 IBUF(6)=5      ;MANNED RAMOS/AMOS TYPE
     J=CP+1
     K=J+5
     DO 245 CP=J,K
     IF(TDATA(CP).EQ.32)GO TO 250    ; FIND FIRST SPACE PAST 'RAMOS'
245 CONTINUE
     GO TO 205
250 MINCP=CP      ;SET MIN CHARACTER POINTER
255 CONTINUE
C    SEARCH FOR WIND GROUP
     GO TO 270
265 IER=6       ;    IE=6  COULDN'T FIND WIND GROUP
     CP=MINCP+1
    CX=CP+50
    JXT=24
    DO 266 JX=CP,CX
    IF (TDATA(JX).EQ.32) GO TO 266
    IF (TDATA(JX).LT.32) GO TO 267
    IF (TDATA(JX).EQ.131) GO TO 267
    IBUF(JXT)=TDATA(JX)
    JXT=JXT+1
    IF (JXT.GT.33) GO TO 267
266 CONTINUE
267 RETURN
```

```
      IF (TDATA(CP).EQ.131) GO TO 265;  END OF MESSAGE
      IF (TDATA(CP).NE.47) GO TO 270 ; SLASH ?
      IY=CP    ;HOOK FOR ALT SETTING AND RMRKS/POINTS TO SLSH AFT WND
  275 IY1=CP-1
      IY2=CP-2
      IY3=CP-3
      IY4=CP-4
      V1=TDATA(IY3)
      VX=81
      VY=71
      CALL ORGO(V1,VX,V1,VY,$285) ; G OT Q INDICATOR
      DO 280 IX=IY4,IY1
      CALL NUMBR(TDATA(IX),$282,$280)    ;CHECK IF ALL 4 ARE NO.S
  280 CONTINUE
      IA=48
      IB=48
      IBUF(39)=INUM(IA,IB,TDATA(IY2),TDATA(IY1))    ;WIND SPEED
      ID=48
      IBUF(38)=INUM(IA,TDATA(IY4),TDATA(IY3),ID)     ;WIND DIR
      GO TO 295
  282 DO 283 IX=IY4,IY1
      IF (TDATA(IX).NE.'<0>M') GO TO 270
  283 CONTINUE
      GO TO 295
  285 DO 290 IX=IY2,IY1
      CALL NUMBR(TDATA(IX),$270,$290) ;CHCK TFOR 2 NO.S AFT G OR Q
  290 CONTINUE
      IA=48
      IB=48
      IBUF(40)=INUM(IA,IB,TDATA(IY2),TDATA(IY1))  ; GUSTS
      CP=CP-3     ;RESET CP
      GO TO 275
C TEMP DEWPT SEARCHES
  295 CP=CP-5   ;CP IS AR SLASH AFT WND
      IF (TDATA(CP).EQ.69) CP=CP-1  ; 'E' FOR ESTIMATED
      M=36    ;    DEWPT DEG F
      N=43    ;    DEWPT DEG C
      CONST=5./9.
      IF (TDATA(CP).NE.47) GO TO 365;  CHECK TO BE SURE ITS A SLASH
      CP=CP-1
      IX=CP+1
      JSIGN=0
  300 IF (TDATA(CP).EQ.77) GO TO 375    ;MISSING  ..M
  305 IX=IX-1
      ITM=TDATA(IX)
      IF(ITM-48)315,310,310    ; BETWEEN 0 AND 97
  310 IF(ITM-57)305,305,315
  315 IF(ITM.EQ.45) JSIGN=1    ;NEGATIVE?
      IF(ITM.EQ.45) GO TO 305
      IF (CP.EQ.IX) GO TO 370  ;NO TEMP--UNEXPECTED CHARACTER
      IX=IX+JSIGN
      I=CP-IX
      IF (I-3) 320,320,370
  320 IF (I-1)370,325,325  ; DETERMINE HOW MANY CHARACTERS IN TEMP
  325 P=48      ;SET TO '0'
      Q=48
      R=48
      GO TO (340,335,330),I
  330 I=CP-2
      P=TDATA(I)
  335 I=CP-1
      Q=TDATA(I)
  340 R=TDATA(CP)
```

```
      IA=48
      IBUF(M)=INUM(IA,P,Q,R);  CONVERT TO INTEGER TEMP/DEWPOINT
      IF (JSIGN.EQ.1) IBUF(M)=-IBUF(M)  ;NEGATIVE
      IF (CAN.EQ.0) GO TO 345 ; US DATA
      IBUF(N)=IBUF(M)    ;CANADNS..CNVRT FOR FHRENHT.
      IBUF(M)=IBUF(M)*100
      Q=IBUF(M)/5*9
      P=Q
      Q=Q/100
      R=P-Q
      IF (R.GT.50)Q=Q+1
      IBUF(M)=Q+32
      GO TO 355
  345 IF (IBUF(M).LT.0) GO TO 350  ;DEG F TO DEG C
      IBUF(N)=IFIX(((IBUF(M)-32.)*CONST)+.5)
      GO TO 355
  350 IBUF(N)=IFIX(((IBUF(M)-32.)*CONST)-.5)
  355 IF (M.EQ.35) GO TO 385
  360 IX=IX-JSIGN
      M=35     ; TEMP DEG F
      N=42     ; TEMP DEG C
      JSIGN=0
      IF (TDATA(IX).NE.47)GO TO 370     ; SHD BE A SLASH
      CP=IX-1
      GO TO 300
  365 V1=TDATA(CP)
      V2=IBUF(6)     ;CHECK FOR A BLANK BFO WIND
      VX=32     ; AND IT/S AN SP TYPR
      VY=3
      CALL ANDGO(V1,VX,V2,VY,$420)  ; GO TO OBS VIS
  370 IER=7     ; IER=7  - TEMP/PRES DECODE ERRORS
      RETURN
  375 CP=CP-1
      IF (TDATA(CP).NE.47)GO TO 380;     + SHD BE A SL
       IF(M.EQ.35)GO TO 390
       IX=CP
       GO TO 360
  380 VX=32    ;BLANK
       VY=35
       CALL ANDGO(TDATA(CP),VX,M,VY,$420)
       IF(TDATA(CP).EQ.77)GO TO 375      ;ADDITIONAL M'S
       GO TO 370
  385 IX=IX-JSIGN
       CP=IX
       ITM=TDATA(CP)
       IF(ITM.EQ.32)GO TO 420      ;SKIP PRESSURE
       IF(ITM.EQ.47)GO TO 390      ;CP AT SLASH BEFORE TEMP
       GO TO 370
C      SL PRESSURE    (CP IS AT SLASH BEFORE TEMP)
  390 J1=CP
  395 J2=CP-1
       IF(TDATA(J2).EQ.32)GO TO 400  ;LOOK FOR 1ST BLANK BEFORE SLASH
       IF(TDATA(J2).EQ.69)GO TO 400  ;EST. PRESSURE
       CP=CP-1
       GO TO 395
  400 IF(J1-CP-3)405,410,370  ;SHOULD HAVE BEEN THREE NUMBERS
  405 IF(J1.EQ.CP)GO TO 420   ;NO PRESSURE
       GO TO 370
```

```
410 J1=CP+1
    J2=CP+2
    IA=48
    IBUF(34)=INUM(IA,TDATA(CP),TDATA(J1),TDATA(J2)) ;CONVERT ASCII PRES TO INTEGER
    ITM=IBUF(34)
    IF(ITM.GE.500)IBUF(34)=ITM+9000
    IF(ITM.LT.500)IBUF(34)=ITM+10000
415 CP=CP-1
C     POINTER NOW AT SPACE BEFORE PRESS (OR TEMP IF NO PRES)
C     DECODE OBSTRUCTION TO VISION
420 IF(IBUF(6).EQ.4)GO TO 455  ;UNMANNED - SKIP TO ALT SETTING
    J1=CP
    DO 440 J2=0,9
    IF(J1.LE.MINCP)GO TO 455  ;SKIP TO ALT SETTING
    IF(TDATA(J1).EQ.32)GO TO 430
    IF(TDATA(J1).EQ.69)GO TO 430   ;EST. PRESSURE
    IF(TDATA(J1)-48)435,425,425 ; BETWEEN 0 AND 9?
425 IF(TDATA(J1)-57)444,444,435 ;NUMBER WOULD BE VSBY
430 IF(J2.EQ.0)GO TO 415
435 ITM=TDATA(J1)
    IF(TDATA(J1).EQ.86)GO TO 445  ;VRBL VSBY
    IBUF(33-J2)=ITM  ;LOAD OBSTRUCTIONS TO VISION
    J1=J1-1
440 CONTINUE
    GO TO 460
C     DECODE VSBY
444 IBUF(23)=32
    GO TO 468
445 IBUF(23)=32
    IF(TDATA(J1).EQ.86)IBUF(23)=86  ; 'V' FOR VRBL
    IF(IBUF(23).EQ.32)GO TO 468
447 J1=J1-1
    R=48   ;'0'
    P=48   ;'0'
    ITM=TDATA(J1)
    CALL NUMBR(ITM,$450,$468)
450 IF (TDATA(J1).EQ.32) GO TO 447
    GO TO 445
455 GO TO 715
460 IER=8
    J1=J1+1
461 J1=J1-1
    IF (TDATA(J1).EQ.86) GO TO 463
    IF (TDATA(J1)-48) 461,462,462
462 IF (TDATA(J1)-57) 463,463,461
463 J3=J1+10
    J2=J1+1
    JX=24
    DO 464 J4=J2,J3
    IBUF(JX)=TDATA(J4)
    JX=JX+1
464 CONTINUE
    GO TO 445
468 R=48
    Q=48
    P=48
    JB=J1   ;BEGINNING CHAR PSN
    JE=J1   ;ENDING CHAR PSN
    R=TDATA(J1)   ;RIGHT MOST NUM
    J1=J1-1
    ITM=TDATA(J1)
    CALL NUMBR(ITM,$510,$500)
500 JB=J1
```

```
      P=TDATA(J1)
      J1=J1-1
      ITM=TDATA(J1)
      CALL NUMBR(ITM,$510,$504)
  504 JB=J1
      Q=TDATA(J1)
      J1=J1-1
      ITM=TDATA(J1)
      CALL NUMBR(ITM,$510,$505)
  505 IF (ITM-67) 506,557,557
  506 IER=8
      GO TO 715
  510 IF (ITM.EQ.32) GO TO 545  ; ONE OR TWO DIG VSBY
      IF (ITM.NE.47) GO TO 544  ; FRACTIONAL VSBY
      IA=48
      IB=48
      IDN=INUM(IA,IB,P,R)        ;DENOMINATOR
      J1=J1-1
      ITM=TDATA(J1)
      CALL NUMBR(ITM,$505,$515)
  515 N=ITM-48      ;NUMERATOR
      IDN=(N*1000)/IDN      ;FRACTION/THSNDTHS
      J1=J1-1
      ITM=TDATA(J1)
      CALL NUMBR(ITM,$525,$520)
  520 N=(ITM-48)*1000
      JB=J1     ;RESET BGNNG PSN(NO SPC SEPARTG WHOLE DIGIT/FRCTN
      GO TO 540
  525 IF (ITM.NE.32) GO TO 535
      J1=J1-1
      ITM=TDATA(J1)
      CALL NUMBR(ITM,$535,$530)
  530 N=(ITM-48)*1000  ;WHOLE DIGIT-2 SPACE BFO FRCTN (SPC SEPRTR)
      JB=J1
      GO TO 540
  535 N=0
      JB=J1+1
      J1=J1+1
  540 IBUF(22)=-(N+IDN) ;STORE INTEGER VAL OF VSBY-NEG INDIC 1/1000MI
      GO TO 550
  544 J1=J1+1
  545 IBUF(22)=((Q-48)*100+(P-48)*10+(R-48))  ;CONVERT VSBY TO INTGR
  550 M=44
      DO 555 I=JB,JE
      IBUF(M)=TDATA(I)  ;LOAD IN ASCII STRING FOR VSBY
  555 M=M+1
C  CP (=J1) IS PSN OF SPC BFO VSBY OR FIRST CHAR OF VSBY
C  DECODE UP TO 3 CLD LYRS
      GO TO 558
  557 J1=J1+1
  558 M=10
      I=1
  560 IF(I.GT.3) GO TO 710  ; 1 PASS FOR EACH OF 3 CLD LYRS
  565 J1=J1-1
  570 IF (J1.LT.MINCP) GO TO 715  ; BEGINNG OF OB FOUND
      DO 575 J=1,3
  575 HGT(J)=48
      ITM=TDATA(J1)
      IF(ITM-67) 565,600,580  ; (OV)C
  580 IF(ITM-78) 710,600,585  ; (BK)N
  585 IF(ITM-82) 710,600,590  ; (CL)R
  590 IF(ITM-84) 710,600,595  ; (SC)T
  595 IF(ITM-88) 710,605,710  ; X
  600 J1=J1-2
```

```
605 IBUF(M)=TDATA(J1)
610 J1=J1-1
    IF(TDATA(J1)-45) 620,615,630
615 IBUF(M+1)=1  ;FLAG FOR - OR -X
    GO TO 610
620 IF(TDATA(J1)-32) 630,610,630
625 J1=J1-1
630 DO 655 J=1,3
    IF(TDATA(J1)-48) 660,635,635  ; BTWN 0 AND 9?
635 IF(TDATA(J1)-57) 645,645,640
640 IF(TDATA(J1).EQ.86) IBUF(M+1)=IBUF(M+1)+50; VRBL FLAG
    IF(TDATA(J1).EQ.86) GO TO 625
    GO TO 660
645 HGT(J)=TDATA(J1)
    IF(J1-MINCP) 715,660,650
650 J1=J1-1
655 CONTINUE
660 IA=48
    IBUF(M+2)=INUM(IA,HGT(3),HGT(2),HGT(1))  ;COMPUTE HGT
665 IF(TDATA(J1)-32)670,700,675
670 IF(J1.EQ.MINCP)GO TO 715
    J1=J1+1
    GO TO 705
675 ITM=TDATA(J1)
    IF(ITM-69) 695,690,680 ; E
680 IF (ITM-77) 695,690,685 ; M
685 IF (ITM-87)695,690,695 ; W
690 IBUF(M+3)=TDATA(J1)  ; LOAD CIG INDIC
    J1=J1-1
     GO TO 665
695 I=I+1
     M=M+4     ;SET FOR NEXT COULD GROUP
     GO TO 570     ;ALREADY AT ANOTHER CHARACTER
700 IF(J1-MINCP)715,715,701     ;NO CEILING
701 J1=J1-1
     ITM=TDATA(J1)
     IF(ITM.EQ.69.OR.ITM.EQ.77.OR.ITM.EQ.87)GO TO 675
     J1=J1+1
705 M=M+4     ;SET FOR NEXT CLOUD GROUP
     IF(I.EQ.3)GO TO 715     ;AT A SPACE NOW
     I=I+1
     GO TO 560
C     GET ALTIMETER SETTING (IY WAS PSN OF SLASH AFTER WIND GROUP)
710 IER=8     ; IER=8 CLOUDS/WX/VSBY ERROR
715 IF(TDATA(IY+1)-77)720,740,835     ;CHECK FOR MISSING
720 DO 730 J=1,3
     IF(TDATA(IY+J)-48)735,725,725 ;CHECK THAT NEXT 3 CHRTCTRS NUMBERS
725 IF(TDATA(IY+J)-57)730,730,735
730 CONTINUE
     IA=48
     IBUF(41)=INUM(IA,TDATA(IY+1),TDATA(IY+2),TDATA(IY+3)) ; CALC A-STNG
     IF(IBUF(41).GT.450)IBUF(41)=IBUF(41)+2000
     IF(IBUF(41).LE.450)IBUF(41)=IBUF(41)+3000
     GO TO 740
735 IER=10   ;IER=10  ALTIMETER GROUP ERROR
     GO TO 835
740 IF(IHOUR/3*3.NE.IHOUR)GO TO 835  ;BYPASS REMARKS IF NOT 3HRLY
     IA=1
     IB=2     ;BYPASS REMARKS IF NOT SA OR RS
    CALL ORGO(IBUF(6),IA,IBUF(6),IB,$745)
     GO TO 835
```

```
C       FIND APP GROUP
  745 CP=IY+4
  750 DO 825 IY=CP,256
       ITM=TDATA(IY)
       IF(ITM.EQ.131)GO TO 835      ;END OF OB
       IF(ITM.NE.32)GO TO 825
       CALL NUMBR(TDATA(IY+1),$825,$755)
  755 ITM=TDATA(IY+2)
       CALL NUMBR(ITM,$760,$765)
  760 IA=49
       IB=47
       CALL ANDGO(TDATA(IY+1),IA,ITM,IB,$805)   ; '1/' COMBINATION
       GO TO 825
  765 ITM=TDATA(IY+3)
       CALL NUMBR(ITM,$770,$775)
  770 IF(ITM.EQ.32)GO TO 830
       IF(ITM.GT.125)GO TO 830      ;SPECIAL CHARACTER (END OF OB),
       IF(ITM.LT.20)GO TO 830       ;CR LF (END OF OB)
       IA=49
       IB=47
       CALL ANDGO(TDATA(IY+1),IA,ITM,IB,$805)   ; '1*/' SEQUENCE
       GO TO 825
  775 ITM=TDATA(IY+4)
       CALL NUMBR(ITM,$785,$780)
  780 CALL NUMBR(TDATA(IY+5),$815,$790)
  785 IF(ITM.EQ.47)GO TO 820  ; ***/ SEQUENCE
       IF(ITM-32)790,790,825
       IF(ITM-90)825,825,790
  790 IF(IBUF(50).NE.-99)GO TO 825
       IBUF(50)=TDATA(IY+1)-48   ;PRESSURE CHARACTERISTIC
       IA=48
       IB=48
       IBUF(51)=INUM(IA,IB,TDATA(IY+2),TDATA(IY+3))  ;PRES TNDNCY
       CP=IY+4
       CALL NUMBR(TDATA(CP),$750,$795)
  795 IA=48
       IB=48
       IBUF(52)=INUM(IA,IB,TDATA(CP),TDATA(CP+1))  ;PCPN
       ITM=TDATA(CP+3)
       IF(ITM.LE.57)GO TO 800  ; NUMBER
       ITM1=IBUF(52)
       IF(ITM.EQ.79)IBUF(52)=ITM1+100  ;'ONE'  INCHES OF PCPN
       ITM2=TDATA(CP+4)
       IA=84
       IB=87
       IV2=ITM1+200
       CALL ANDEQ(ITM,IA,ITM2,IB,IBUF(52),IV2)  ; 'TWO'
       IB=72
       IV2=ITM1+300
       CALL ANDEQ(ITM,IA,ITM2,IB,IBUF(52),IV2)  ; 'THREE'
       IA=70
       IB=79
       IV2=ITM1+400
       CALL ANDEQ(ITM,IA,ITM2,IB,IBUF(52),IV2)  ; 'FOUR'
       IB=73
       IV2=ITM1+500
       CALL ANDEQ(ITM,IA,ITM2,IB,IBUF(52),IV2)  ; 'FIVE'
       IF(IBUF(52).GT.99)GO TO 800
       IER=11   ; IER=11   REMARKS ERROR
       GO TO 835
  800 CP=CP+2
```

```
C       LOOK FOR CLOUD GROUP
        GO TO 750
  805 IB=47
        CALL ANDGO(TDATA(IY+3),IB,TDATA(IY+4),IB,$810)  ; '--//' SQNC
        GO TO 825
  810 IF(IBUF(53).NE.32)GO TO 825
        IBUF(53)=TDATA(IY+2)   ; LOW CLOUDS
        IBUF(54)=TDATA(IY+3)   ; MID CLOUDS
        IBUF(55)=TDATA(IY+4)   ; HIGH CLOUDS
        CP=IY+5
        GO TO 750  ;LOOK FOR MAX/MIN TEMP
  815 IF(TDATA(IY+5)-32)820,820,816
  816 IF(TDATA(IY+5)-90)817,817,820
  817 IER=11    ; IER=11  REMARKS ERROR
        GO TO 835
  820 IF(TDATA(IY+1).EQ.49)GO TO 810
  825 CONTINUE
        GO TO 835
  830 IF(IBUF(56).NE.-99)GO TO 835
        IA=48
        IB=48
        IBUF(56)=INUM(IA,IB,TDATA(IY+1),TDATA(IY+2))   ;MAX/MIN
C MIXING RATIO CALC
  835 TX=IBUF(42)
      TD=IBUF(43)
      IF(TX.LT.-60.0.OR.TX.GT.60.) GO TO 840
      IF(TD.LT.-60.0.OR.TD.GT.60.)GO TO 840
      IF (IBUF(34).LT.9000.OR.IBUF(34).GT.11000)GO TO 840
      PRES=FLOAT(IBUF(34))/10.
      LEVAP=597.3-.566*TX
      E=6.11*EXP(9.045*LEVAP*(1./273.-1./(TD+273)))
      QGKG=.622*E/(PRES-E)*1000.
      IBUF(37)=IFIX(QGKG*10.+.5)
  840 IF(IER.EQ.0) IER=1
      RETURN
      END
```

```
CCCC
C   THIS SUBROUTINE IS JUST A LOGICAL IF STATEMENT
C   IF A=B AND C=D THEN IV1=IV2
C   USED TO SAVE DISK SPACE/CORE REQUIREMENT
C
C   PROGRAMMER  RICH THOMAS SXB.ISL.SDO 7/79 (SAODECODER)
C
      SUBROUTINE ANDEQ(A,B,C,D,IV1,IV2)
      INTEGER A,B,C,D
      IF (A.EQ.B.AND.C.EQ.D) IV1=IV2
      RETURN
      END

C   THIS SUBROUTINE IS A LOGICAL IF STATEMENT
C   IF A=B AND C=D THEN GO STATEMENT # ISTN IN
C   CALLING PROGRAM.  USED TO CUT DOWN PROGRAM SIZE
C   BY SUBROUTINING AN OPERATION REPEATED MANY TIMES
C   PROGRAMMER- RICH THOMAS SXB,ISL,SDO 7/79 (SAODECODER)
      SUBROUTINE ANDGO(A,B,C,D,ISTN)
      INTEGER A,B,C,D
      IF(A.EQ.B.AND.C.EQ.D)RETURN ISTN
      RETURN
      END

C   THIS SUBROUTINE IS A LOGICAL IF STATEMENT
C   IF A=B OR C=D GO TO STATEMENT # ISTN IN CALLING PROGRAM.
C   USED TO SAVE CORE
C
C   PROGRAMMER-RICH THOMAS SXB.ISL.SDO 9/79 (SAODECODER)
C
      SUBROUTINE ORGO(A,B,C,D,ISTN)
      INTEGER A,B,C,D
      IF(A.EQ.B.OR.C.EQ.D)RETURN ISTN
      RETURN
      END

C   THIS SUBROUTINE DETERMINES IF  "A" IS A NUMBER AND RETURNS
C   TO STATEMENT #B IN CALLING PROGRAM IF NOT.  RETURNS TO
C   STATEMENT #C IN CALLING PROGRAM IF IT IS
C
C   PROGRAM.ER-RICH THOMAS  SXB.ISL.SDO 9/79 (SAODECODER)
C
      SUBROUTINE NUMBR(A,B,C)
      INTEGER A,B,C
      IF (A-48) 900,901,901
  901 IF (A-57) 902,902,900
  900 RETURN B; NOT A NUM
  902 RETURN C; 0 TO 9
      END
```

ERCP #41
July 1987

SPOT (Severe Weather Potential) Index Plot

PART A: INFORMATION AND INSTALLATION

PROGRAM NAME: SPOT

AAL ID:
REVISION NO.: 1.00

PURPOSE: Creates a B02 plotfile containing the SPOT indexes and observed winds for selected sites. This can be plotted on a regional background using PMOD.

PROGRAM INFORMATION:

Development Programmer:
Ken LaPenta
Location: WSFO ALB
Phone: (FTS) 562-8586
Language: FORTRAN IV/5.57
Save File Creation Date(s): 06/02/87

Maintenance Programmer:
Ken LaPenta
Location: WSFO ALB
Phone: (FTS) 562-8586
Type: Standard

Running Time: Depends on number of stations used (82 stations take 2-3 min)
Disk Space:

Program	62 RDOS blocks
Data	around 1 block for 20 stations

PROGRAM REQUIREMENTS

Program Files:

Name	DP Location	Comments
SPOT.SV	APPL1	link from SYSZ

Data Files:

Name	DP Location	R/W	Comments
SPOTIN.DT	APPL1	R	link from SYSZ
NMCPLTSAO	SYSZ	W	temporary output file

AFOS Products:

ID	Action	Comments
NMCPLTSAO	Store	Output plotfile

LOAD LINE

```
RLDR/P SPOT CURJTIME KEYJTIME AIRXX ANDEQ ANDGO ORGO NUMBR °
SPOTREV <BG UTIL FORT>.LB SPOT.LS/L
```

PROGRAM INSTALLATION

1. Move SPOT.SV to APPL1 and create a link to it in SYSZ.

2. Create the file SPOTIN.DT with M:F/ or a text editor. SPOTIN.DT's first line contains the number of stations to plot (as a three-digit number). Each following line contains the AFOS key of the station's observation, a space, and then the PSOWDT, and B02 X-coordinate and Y-coordinate all together. The last three items can be taken from any B02 plotfile (such as NMCPLTP0A).

3. Make sure that NMCPLTSA0 is in the database. You should also choose a graphic for the SPOT output graphic and put that in the database, if necessary. Use the KEY: command to set the appropriate map background.

4. Write a macro to run SPOT and then create the graphic:

```
SPOT
PMOD SA0 NAxx.PF/T SFC.PM/0
GENUTF XPLOT yyy
```

where xx is the number of the regional background you want to use and yyy is the output graphic (set to background xx). You can also plot SA0 on B02 using PMOD with NA.PF/T or REGPLOT.

SPOT (Severe Weather Potential) Index Plot

PART B: EXECUTION AND ERROR CONDITIONS

PROGRAM NAME: SPOT.SV

AAL ID:
REVISION NO.: 1.00

PROGRAM EXECUTION:

The SPOT program creates a plotfile in NMCPLTSAO. PMOD and GENUTF must then be used to plot it. The most convenient way to produce the graphic is a macro that runs all three programs:

```
SPOT
PMOD SAO NAxx.PF/T SFC.PM/O
GENUTF XPLOT yyy
```

where xx is the number of the regional background you want to use and yyy is the output graphic (set to background xx). You can also plot SAO on B02 using PMOD with NA.PF/T or REGPLOT.

ERROR CONDITIONS

Fatal error messages are written to the Dasher. If an observation is old or cannot be decoded, no SPOT index will be calculated for that station.

Messages from ADM	Meaning
None	

Dasher Messages	Meaning
ERROR IN DATE/TIME	error in calling subroutines DATE or TIME
ERROR IN KSRCF	can't find observation in database
ERROR IN RDBKF	can't read observation from database
ERROR IN WRS	can't write to file NMCPLTSAO
ERROR IN FSTOR	can't store NMCPLTSAO in database

Eastern Region Computer Programs and Problems (Continued)

19 Verification of Asynchronous Transmissions. Lawrence Cedrone, March 1984. (PB84 189885)

20 AFOS Hurricane Plotter. Charles Little, May 1984. (PB84 199629)

21 WARN - A Warning Formatter. Gerald G. Rigdon, June 1984. (PB84 204551)

22 Plotting TDL Coastal Wind Forecasts. Paula Severe, June 1984 (Revised) (PB84 220789)

23 Severe Weather Statistics STADTS Decoder (SWX) and Plotter (SWY). Hugh M. Stone, June 1984. (PB84 213693)

24 WXR. Harold Opitz, August 1984. (PB84 23722) (Revised August 1985, PB84 100815/AS)

25 FTASUM: Aviation Forecast Summaries. Matthew Peroutka, August 1984. (PB85 112977)

26 SAOSUM: A Short Summary of Observations. Matthew Peroutka, October 1984. (PB85 120384)

27 TRAJ - Single Station Trajectory Plot. Tom Niziol, December 1984. (PB85 135002)

28 VIDTEX. Gerald G. Rigdon, February 1985. (PB85 175669/AS)

29 Isentropic Plotter. Charles D. Little, February 1985. (PB85 175651/AS)

30 CERR: An Aviation Verification Program. M. Peroutka, April 1985. (PB85 204824/AS)

31 Correlation and Regression Equation - REGRS. Hugh M. Stone, May 1985. (PB85 213353/AS)

32 Scatter Diagram and Histogram Program - SCATR. Hugh M. Stone, May 1985. (PB85 213346/AS)

33 TIMCHEK. Gerald G. Rigdon, June 1985. (PB85-221257/AS)

34 A MOS Temperature - PoP Forecast Plot. William C. Randel, October 1985. (PB86 120029/AS)

35 ROTODRAW. Thomas Niziol, November 1985 (PB86 131828/AS)

36 LAWEB: Data Processing for the Great Lakes. William C. Randel and Matthew R. Peroutka, March 1986. (PB86 176658/AS)

37 Convective Parameters & Hodograph Program - Convect. Hugh M. Stone, April 1986. (PB86-197225/AS)

38 DWXR - SHEF Product Compression Program. Harold H. Opitz, September 1986.

39 CRASHQ: Listing Products Being Transmitted At the Time of a Crash. William C. Randel, January 1987 (PB87-151890/AS)

40 AVGPLOT and AVGCLIM. Alan Blackburn, March 1987 (PB87-180626/AS)

NOAA SCIENTIFIC AND TECHNICAL PUBLICATIONS

The National Oceanic and Atmospheric Administration was established as part of the Department of Commerce on October 3, 1970. The mission responsibilities of NOAA are to assess the socioeconomic impact of natural and technological changes in the environment and to monitor and predict the state of the solid Earth, the oceans and their living resources, the atmosphere, and the space environment of the Earth.

The major components of NOAA regularly produce various types of scientific and technical information in the following kinds of publications:

PROFESSIONAL PAPERS—Important definitive research results, major techniques, and special investigations.

CONTRACT AND GRANT REPORTS—Reports prepared by contractors or grantees under NOAA sponsorship.

ATLAS—Presentation of analyzed data generally in the form of maps showing distribution of rainfall, chemical and physical conditions of oceans and atmosphere, distribution of fishes and marine mammals, ionospheric conditions, etc.

TECHNICAL SERVICE PUBLICATIONS—Reports containing data, observations, instructions, etc. A partial listing includes data serials; prediction and outlook periodicals; technical manuals, training papers, planning reports, and information serials; and miscellaneous technical publications.

TECHNICAL REPORTS—Journal quality with extensive details, mathematical developments, or data listings.

TECHNICAL MEMORANDUMS—Reports of preliminary, partial, or negative research or technology results, interim instructions, and the like.

Information on availability of NOAA publications can be obtained from:

NATIONAL TECHNICAL INFORMATION SERVICE
U. S. DEPARTMENT OF COMMERCE
5285 PORT ROYAL ROAD
SPRINGFIELD, VA 22161