NOAA Eastern Region Computer Programs
and Problems NWS ERCP - No. 25

FTASUM: AVIATION FORECAST SUMMARIES

Matthew R. Peroutka
National Weather Service Forecast Office
Cleveland, Ohio

Scientific Services Division
Eastern Region Headquarters
August 1984

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)

16 RANP: Stability Analysis Program. Hugh M. Stone, February 1984.(PB84 16144

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)

NOAA EASTERN REGION COMPUTER PROGRAMS AND PROBLEMS - No. 25

FTASUM: AVIATION FORECAST SUMMARIES

Matthew R. Peroutka
National Weather Service Forecast Office
Cleveland, Ohio

Scientific Services Division
Eastern Region Headquarters
August 1984

FTASUM: AVIATION FORECAST SUMMARIES

Matthew R. Peroutka
National Weather Service Forecast Office
Cleveland, Ohio

I. General Information

A. Summary

Aviation terminal forecasts are generally logged by hand onto Weather Service Form D-7 along with their associated surface observations. In past years, the purpose of this log was to help the forecaster "keep up with the weather." Maintaining this sort of log is time-consuming and cumbersome, and the log does not reflect the entire aviation picture. (Special observations and amended forecasts are usually omitted.)

Some forecasters would like to eliminate the chore of manually logging hourly observations. The AFOS system provides other ways to monitor the weather. Console procedures, the alarm/alert features, and the terminal forecast monitoring program (MONITR) are only three of the available alternatives. These techniques, however, do not provide a record which can be reviewed easily at a later date.

This paper describes a program which uses the AFOS database to produce a complete log sheet. All forecasts and observations which occur during a given time window are printed in chronologic order. Each site can specify three time windows and the time of day during which each window is used. The site also specifies a list of one to sixteen nodes. Forecasts and observations are printed for every FTA product in the database associated with the nodes on the list.

B. Environment

The program was written in Data General's FORTRAN IV, and it runs in the background partition of the Eclipse S23Ø minicomputer. FTASUM gets its input from the local database, command line switches, and a single data file, FTASUM.DA. The system printer-plotter module (device $LPT) is used for output, but this can be altered by running a simple macro.

II. Application

A. Complete Program Description

FTASUM first reads the command line switches and the data file FTASUM.DA. This determines a start time and an end time in minutes since January 1st. The data file also provides the list of nodes to summarize.

The subroutine FTSKY generates product keys for each forecast and observation which must be summarized. A second subroutine, VNUMS, then examines the available versions of each product and compares them with the beginning and ending times. VNUMS then returns the version numbers which should be printed. The heart of the program is a subroutine called FTSTA. This routine compares the creation time of each version and prints the FT's and SA's in chronologic order.

If a new FTA or SAO product arrives in the database while the subroutine FTSTA is executing, some confusion may develop. This may cause one product to print twice while another is skipped. The new product must arrive during the twenty seconds it takes to execute FTSTA, so this problem should be quite rare.

B. Machine Requirements

FTASUM.SV requires about 32K words of memory and occupies about 25 RDOS blocks of disk space. One data file must also reside on the disk (FTASUM.DA), but it takes up only one block. A second program, LDFTASUM.SV, sets up this data file. It requires less than 10K words of memory to run. This setup program requires 20 blocks of disk space, but it is only run once to generate the data file.

Output from the program is routed directly to the printer-plotter module (device $LPT) since this is the most efficient way to use the system resources. Unfortunately, the database searches are time-consuming, and this makes the execution time rather long. About 30 to 40 seconds of run time is required for each station summarized. The printer is tied up during the entire operation. Any attempts to print from an ADM generate a "PRINTER OPEN ERROR."

Since the $LPT device is merely a link into the asynchronous line multiplexor, output can be routed to a file by executing the following macro:

```
DELETE TEMP
UNLINK $LPT
LINK $LPT TEMP
FTASUM.SV
UNLINK $LPT
LINK $LPT QTY:##
PRINT TEMP
```

where ## is the multiplexor channel number.

If this macro is used, the printer is available from foreground during the program run, but it is still tied up once the file begins to print.

Although this technique eases the printer conflict, it does so by putting more strain on the disk. Since disk contention can be a severe problem in an AFOS system, it is usually better to send the data directly to the printer.

If, after repeated attempts, the program cannot access the printer-plotter, the program sends an error message to the initiating console. If all goes well, a "JOB COMPLETE" message is sent when the program is done.

C. Database

One file, FTASUM.DA, contains the input data which drives the program. Table 1 contains the format for this file. All terminal forecasts which should be summarized must exist in the database along with their verifying observations.

III. Procedures

A. Installation

If the time switches are not used in the command line, the program checks the system clock and chooses one of three time windows to summarize.

These three windows must be selected before installing the program. The windows will usually reflect the periods between routine terminal forecast issuances (i.e., 1ØZ - 15Z, 15Z - 22Z, and 22Z - 1ØZ). Three "valid times" must also be selected. Window one will then be used between valid time one and valid time two, window two will be used between valid time two and valid time three, etc.

Finally, a list of nodes to summarize should also be compiled. Most forecast offices will need only one node, but the program will accept up to 16 nodes.

Table 1: File Format for FTASUM.DA

Word	Contents
Ø	Time to start using first window.
1	Time to start using second window.
2	Time to start using third window.
3	Start time of first window.
4	End time of first window.
5	Start time of second window.
6	End time of second window.
7	Start time of third window.
8	End time of third window.
9-10	Station I.D. of first node.
11-12	Station I.D. of second node.
"	"
"	" (Zero after last.)
"	"
39-40	Station I.D. of sixteenth node.

The program, LDFTASUM.SV takes all of this information in a conversational style at the Dasher and creates the FTASUM.DA file. Since LDFTASUM.SV is only needed once, most stations can run it from a floppy diskette. A sample conversation is included in the Appendix. If changes are required in FTASUM.DA, the LDFTASUM program can be run again. If the system manager is handy with the octal editor, this data file can be manipulated directly. A map of this file is contained in Table 1.

B. Initiation

Once the time windows and node list have been set in the file FTASUM.DA, the program can be started at an ADM by typing

RUN:FTASUM

If a specific time window is desired, it can be entered in the run command using /B and /E switches.

RUN:FTASUM 12/B 22/E

will summarize the terminal forecasts and observations between 12ØØ and 22ØØ GMT.

C. Ouput

Table 2 shows a sample of the program's output. The terminal forecasts and observations may occasionally appear out of sequence. This is because the program sorts the products according to their creation date-time polynomials. (This is the time a product is actually stored in the originator's database. Products which reach the AFOS system through the SMCC Gateway are considered as originating at the SMCC.) Discrepancies can occur between the time of creation and the valid time of a product.

D. Caution

This program requires a lot of database access. Many problems can occur which will cut the summary short.

Some problems are possible during the first hours of the new year since the program computes the product creation time in minutes since January first.

The state of the AFOS database does not constitute a conclusive record of the terminal forecasts issued or information received by an aviation forecaster.

Table 2: Sample Output from FTASUM.SV

```
TOL 201010 C70 BKN 1812 CHC C20 OVC 3SW-. 17Z 70 SCT 3110 CHC C70 BKN.
    23Z CLR 2712. 04Z VFR..

 TOL SA 1250  250-SCT 12 284/-1/-9/2512/031/
 TOL SA 1350 40 SCT E70 BKN 10 294/-1/-9/2511/034

TOL 201515 C80 BKN 2812 BKN OCNL SCT CHC 40 SCT. 23Z CLR 2710.
    09Z VFR..

 TOL SA 1450 40 SCT 15 305/0/-9/2812/037/230 1500
 TOL SA 1550 E40 BKN 12 315/2/-8/2714/040

TOL FT AMD 1 281615 1615Z C40 BKN 2715 BKN OCNL SCT. 23Z CLR 2710. 09Z
    VFR..

 TOL SA 1650 40 SCT 12SW- 319/3/-7/2412/041/S848
-------------------------------------------------------------------------------

YNG 201010 CLR 2212. 11Z C70 BKN 2315 CHC C20 OVC 3SW-. 19Z C70 BKN
    3118 CHC C20 OVC 3SW-. 00Z CLR 2712. 04Z VFR..

 YNG SA 1252 E60 OVC 10 283/0/-12/2310/026
 YNG SP 1345 M23 OVC 21/2SW- 2212/026
 YNG SA 1350 M23 OVC 21/2SW- 282/1/-9/2111/026

YNG 201515 C20 OVC 3SW- 2212 CHC 20 SCT C45 BKN. 17Z C80 BKN 2812
    CHC C30 OVC 4SW-. 00Z CLR 2710. 09Z VFR..

 YNG SP 1428 M28 OVC 4SW- 2213/027
 YNG SA 1450 M28 OVC 5SW- 289/2/-9/2310/028/BINOVC/ 30700 15/
 YNG SP 1525 M32 BKN 5SW- 2411/029
 YNG SA 1550 M32 BKN 5SW- 293/4/-7/2515/029
 YNG SA 1650 M30 OVC 7SW- 296/6/-6/2614/030/BINOVC

YNG FT AMD 1 201715 1715Z C30 BKN 2615 OCNL C20 OVC 3SW- CHC 30 SCT.
    20Z 30 SCT 2915 CHC C30 BKN 4SW-. 00Z CLR 2710. 09Z VFR..

 YNG SP 1721 M25 OVC 8SW- 2712/030/BINOVC
-------------------------------------------------------------------------------

ZZV 201010 CLR 1812. 19Z CLR 3115. 00Z CLR 2710. 04Z VFR..

 ZZV SA 1153 CLR 15 310/-5/-16/1708/037 /707 93 90404
 ZZV SA 1255 CLR 15 315/-5/-15/1506/038
 ZZV SA 1357 CLR 15 327/-1/-14/1807/040/FEW CI NW

ZZV 201515 CLR CHC 50 SCT. 09Z VFR..

 ZZV SA 1459 CLR 15 325/6/12/2312/041/ 214
 ZZV SA 1556 25 SCT 250 -BKN 15 329/10/-5/2512G17/043
 ZZV SA 1659 25 SCT 250 -BKN 15 328/11/-4/2612/043
-------------------------------------------------------------------------------
```

IV. Program Listings

```
C     FTASUM.SV                  AVIATION FORECAST AND OBSERVATION SUMMARY
      DIMENSION NODES(16,2), KYFT(5), KYSA(5)
      COMMON /FTS/ ISPCR(37)
      DATA ISPCR /"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~",
     +"~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~",6415K/
C
C      THIS PROGRAM PRINTS A CHRONOLOGIC SUMMARY OF FTA AND SAO PRODUCTS FOR A
C      LIST OF NODES ON $LPT.  THE TIME WINDOW PRINTED IS DETERMINED BY A SET
C      OF WINDOWS IN FTASUM.DA.  LOCAL SWITCHES /B AND /E IN THE COMMAND LINE
C      CAN RESET THE BEGINNING AND ENDING TIMES.
C
C      RLDR FTASUM FTSOP FTSKY FTSTA VNUMS FTSPT GVER JDATE MASK BG.LB UTIL.LB
C      FORT.LB
C
      CALL FTSOP(BTIM,ETIM,NODES)          ; READ SWITCHES AND DATA FILE.
      DO 100 I = 1, 5                      ; TRY TO OPEN PRINTER.
      CALL GCHN(IC,IER)
      CALL OPENE(IC,"$LPT",0,IER)
      IF (IER.EQ.1) GOTO 200
      CALL WAIT(5,2,IER)
  100 CONTINUE
      CALL FORKE("FTASUM","PPM TROUBLE",IER)    ; BOMB OUT!
      GOTO 900
  200 CALL FTSKY(KYFT,KYSA,NODES,IER)
      IF (IER.NE.1) GOTO 900
      CALL VNUMS(KYFT,BTIM,ETIM,IBF,IEF,IER)
      IF (IER.NE.1) GOTO 200
      CALL VNUMS(KYSA,BTIM,ETIM,IBS,IES,IER)
      IF (IER.NE.1) GOTO 200
      CALL FTSTA(IC,KYFT,KYSA,IBF,IEF,IBS,IES,IER)
      CALL WRL(IC,ISPCR,N,IER)
      GOTO 200
  900 CALL FORK("FTASUM",IER)
      CALL KLOSE(IC,IER)
      CALL EXIT
      END
```

```
      SUBROUTINE FTSOP(BTIM,ETIM,NODES)
      LOGICAL BTEST
      DIMENSION IBF(256), IDT(3), NODES(16,2), IAR(2)
C
C      THIS SUBROUTINE READS THE NODE LIST, SHIFT TIMES, AND WINDOW TIMES FROM
C      THE FTASUM.DA FILE.  IF THE /B OR /E SWITCHES ARE INVOKED, THEY ARE
C      READ AND ANALYZED.
C
      CALL GCHN(IC,IER)                          ; READ DATA FROM FILE.
      CALL OPENR(IC,"FTASUM.DA",0,IER)
      CALL RDB(IC,IBF,0,1,IER)
      CALL KLOSE(IC,IER)
      CALL DATE(IDT,IER)                         ; GET CURRENT DATE AND TIME.
      CALL FGTIM(IHR,I,J)
      ISF = 3                                    ; DETERMINE CURRENT SHIFT.
      IF (IHR.GT.IBF(1).AND.IHR.LE.IBF(2)) ISF = 1
      IF (IHR.GT.IBF(2).AND.IHR.LE.IBF(3)) ISF = 2
      I = (ISF + 1)*2                            ; DETERMINE WINDOW TIMES.
      IB = IBF(I)
      IE = IBF(I+1)
      DO 100 I = 1, 16                           ; LOAD NODE LIST.
       J = 8 + I*2
       NODES(I,1) = IBF(J)
  100  NODES(I,2) = IBF(J+1)
      CALL GCHN(IC,IER)                          ; CHECK RUN LINE OPTION SWITCHES.
      CALL OPENR(IC,"COM.CM",0,IER)
      IF (IER.NE.1) GOTO 300
      CALL RDL(IC,IBF,N,IER)
      IF (IER.NE.1) GOTO 300
      NBY = 4
      CALL RDS(IC,IAR,NBY,IER)
      IF (IER.NE.1) GOTO 300
  200 CALL RDL(IC,IBF,N,IER)
      IF (IER.NE.1) GOTO 300
      NBY = 4
      CALL RDS(IC,IAR,NBY,IER)
      IF (IER.NE.1) GOTO 300
      IF (.NOT.BTEST(IAR(1),14)) GOTO 210; /B SWITCH FOR BEGINING TIME.
       IB = ISHFT(IBF(1),-8) - 48
       IF (N.EQ.3.OR.N.EQ.5) IB = IB*10 + IAND(IBF(1),377K) - 48
       GOTO 200
  210 IF (.NOT.BTEST(IAR(1),11)) GOTO 300; /E SWITCH FOR ENDING TIME.
       IE = ISHFT(IBF(1),-8) - 48
       IF (N.EQ.3.OR.N.EQ.5) IE = IE*10 + IAND(IBF(1),377K) - 48
       GOTO 200
  300 CALL KLOSE(IC,IER)
      IF (IB.LT.IE) GOTO 500
       IF (IHR.LE.12) GOTO 400                   ; CHECK FOR DAY CROSSOVER.
        IE = IE + 24
        GOTO 500
  400   IB = IB - 24
  500 JD = JDATE(IDT(1),IDT(2),IDT(3)) - 1       ; CONVERT TO JULIAN MINUTES.
      BTIM = JD*1440. + IB*60.
      ETIM = JD*1440. + IE*60.
      RETURN
      END
```

```
      SUBROUTINE FTSKY(KYFT,KYSA,NODES,IRER)
      DIMENSION KYFT(5), KYSA(5), NODES(16,2), KREC(20)
      COMMON /FTN/ NNODE
      DATA NNODE /0/
C
C      THIS SUBROUTINE CREATES THE FTA AND SAO KEYS.  IRER RETURNS 1 FOR
C      SUCCESS.
C
      IRER = 1
      IF (NNODE.NE.0) GOTO 200
  100 NNODE = NNODE + 1
      IF (NODES(NNODE,1).EQ.0) GOTO 800    ; MOVE TO A NEW NODE.
      KYFT(1) = NODES(NNODE,1)             ; NO MORE NODES.
      KYFT(2) = IOR(IAND(NODES(NNODE,2),177400K),"<0>F")
                                           ; SEARCH FOR 1ST FTA ON THIS NODE.
      KYFT(3) = "TA"
      KYFT(4) = "00"
      KYFT(5) = "0"
  200 CALL KSRCF(KYFT,KREC,IER)
      CALL NXKF(KREC,IER)                  ; SEARCH FOR CURRENT KEY.
      IF (IER.NE.1) GOTO 100               ; GET NEXT KEY.
      IF (IAND(KREC(2),377K).NE."<0>F".OR.KREC(3).NE."TA") GOTO 100
      DO 300 I = 1, 4
  300 KYFT(I) = KREC(I)                    ; SUCCESS!  LOAD ARRAYS.
      KYFT(5) = IAND(KREC(5),177400K)
      KYSA(1) = KYFT(1)
      KYSA(2) = IOR(IAND(KYFT(2),177400K),"<0>S")
                                           ; PREPARE SAO KEY.
      KYSA(3) = "AO"
      KYSA(4) = KYFT(4)
      KYSA(5) = KYFT(5)
      GOTO 900
  800 IRER = -1
  900 RETURN
      END
```

```
      SUBROUTINE VNUMS(KEY,BTIM,ETIM,IBV,IEV,IRER)
      DIMENSION KEY(5), KREC(20), IBF(128)
      PTIM(I,J) = ISHFT(I,-8)*16384. + IAND(I,377K)*128. + ISHFT(J,-8)
C
C      THIS SUBROUTINE SEARCHES THE AVAILABLE VERSIONS OF THE PRODUCT KEY FOR
C      VERSIONS WHICH FALL WITH THE TIME WINDOW (BTIM,ETIM).  THE OLDEST (IBV)
C      AND LATEST (IEV) VERSION NUMBERS ARE RETURNED.  IRER HAS A VALUE OF 1
C      IF THE CALL SUCCEEDS.
C
      IRER = 1
      CALL KSRCF(KEY,KREC,IER)                    ; OPEN PRODUCT.
      IF (IER.NE.1) GOTO 800
      NUMV = IAND(KREC(9),177K)                   ; NUMBER OF VERSIONS STORED.
      CALL RDBKF(0,IBF,IER)
      IF (IER.NE.1) GOTO 800
      IEV = 0
  100 IF (PTIM(IBF(9),IBF(10)).LE.ETIM) GOTO 200   ; LATEST VERSION IN WINDOW.
       IEV = IEV + 1
       IF (IEV.GT.NUMV) GOTO 800
       CALL PRVRF(IER)
       IF (IER.NE.1) GOTO 800
       CALL RDBKF(0,IBF,IER)
       IF (IER.NE.1) GOTO 800
       GOTO 100
  200 IBV = IEV                                   ; OLDEST VERSION IN WINDOW.
  300 IF (PTIM(IBF(9),IBF(10)).LE.BTIM) GOTO 900
      IBV = IBV + 1
      IF (IBV.GT.NUMV) GOTO 400
      CALL PRVRF(IER)
      IF (IER.NE.1) GOTO 800
      CALL RDBKF(0,IBF,IER)
      IF (IER.NE.1) GOTO 800
      GOTO 300
  400 IBV = NUMV
      GOTO 900
  800 IRER = -1
  900 RETURN
      END
```

```
      SUBROUTINE FTSTA(IC,KYFT,KYSA,IBVF,IEVF,IBVS,IEVS,IRER)
      DIMENSION KYFT(5), KYSA(5), IBF(128)
      PTIM(I,J) = ISHFT(I,-8)*16384. + IAND(I,377K)*128. + ISHFT(J,-8)
C
C      THIS SUBROUTINE PRINTS THE DESIRED VERSIONS OF THE FT'S (KYFT) AND SA'S
C      (KYSA) IN ORDER ON CHANNEL IC.  IRER RETURNS 1 IF SUCCESSFUL.
C
      IRER = 1
      IFT = IBVF
      ISA = IBVS
      CALL GVER(KYFT,IBVF,IBF,IER)          ; GET OLDEST FT AND SA.
      IF (IER.NE.1) GOTO 800
      TFT = PTIM(IBF(9),IBF(10))
      CALL GVER(KYSA,IBVS,IBF,IER)
      IF (IER.NE.1) GOTO 800
      TSA = PTIM(IBF(9),IBF(10))
  100 IF (TSA.GE.TFT) GOTO 300              ; COMPARE TIMES.
      CALL GVER(KYSA,ISA,IBF,IER)           ; PRINT SA.
      IF (IER.NE.1) GOTO 800
      CALL FTSPT(IC,IBF,0,IER)
      IF (IER.NE.1) GOTO 800
      IF (ISA.LE.IEVS) GOTO 200
      ISA = ISA - 1                         ; GET NEXT SA.
      CALL GVER(KYSA,ISA,IBF,IER)
      IF (IER.NE.1) GOTO 800
      TSA = PTIM(IBF(9),IBF(10))
      GOTO 100
  200 CALL GVER(KYFT,IFT,IBF,IER)           ; LAST SA GONE.  PRINT REST OF FT'S.
      IF (IER.NE.1) GOTO 800
      CALL FTSPT(IC,IBF,1,IER)
      IF (IER.NE.1) GOTO 800
      IF (IFT.LE.IEVF) GOTO 900
      IFT = IFT - 1
      GOTO 200
  300 CALL GVER(KYFT,IFT,IBF,IER)           ; PRINT FT.
      IF (IER.NE.1) GOTO 800
      CALL FTSPT(IC,IBF,1,IER)
      IF (IER.NE.1) GOTO 800
      IF (IFT.LE.IEVF) GOTO 400
      IFT = IFT - 1
      CALL GVER(KYFT,IFT,IBF,IER)
      IF (IER.NE.1) GOTO 800
      TFT = PTIM(IBF(9),IBF(10))
      TFT = PTIM(IBF(9),IBF(10))
      GOTO 100
  400 CALL GVER(KYSA,ISA,IBF,IER)           ; LAST FT.  PRINT REST OF SA'S.
      IF (IER.NE.1) GOTO 800
      CALL FTSPT(IC,IBF,0,IER)
      IF (IER.NE.1) GOTO 800
      IF (ISA.LE.IEVS) GOTO 900
      ISA = ISA - 1
      GOTO 400
  800 IRER = -1
  900 RETURN
      END
```

```
      SUBROUTINE FTSPT(ICHN,IBF,ISP,IRER)
      DIMENSION IBF(128), IBFT(256), IBFU(1008), LINE(41)
C
C      THIS SUBROUTINE PRINTS UP TO FOUR BLOCKS OF ANY AFOS PRODUCT ON RDOS
C      CHANNEL ICHN.  AFOS BLOCK ZERO OF THE PRODUCT MUST RESIDE IN IBF.  IF
C      ISP IS 1, BLANK LINES ARE PRINTED BEFORE AND AFTER THE PRODUCT.  IRER
C      RETURNS 1 FOR SUCCESS.
C
      IRER  = 1
      DO 200 I = 1, 4                          ; LOAD THE BLOCKS INTO IBFU.
       CALL UNPACK(IBF,256,IBFT)
       LAST = I*252
       DO 100 J = 5, 256
  100  IBFU(LAST-256+J) = IBFT(J)
       CALL NXBKF(IBF,IER)
       IF (IER.NE.1) GOTO 300
  200 CONTINUE
  300 IF (ISP.EQ.1) CALL WRL(ICHN,"<40><15>",N,IER)   ; BLANK LINE.
      IF (ISP.EQ.1.AND.IER.NE.1) GOTO 800
      IB = MASK("<0><305><0><200>",2,IBFU,1,50); FIND START OF DATA.
      IF (IB.EQ.-1) GOTO 800
      IB = MASK(5015K,1,IBFU,IB,50)          ; FIND START OF DATA.
      IF (IB.EQ.-1) GOTO 800
      IB = MASK(20203K,1,IBFU,IB,100)
      IF (IB.EQ.-1) GOTO 800
      LAST = MASK(203K,1,IBFU,IB,LAST)       ; FIND END OF TEXT.
      IF (LAST.EQ.-1) GOTO 800
  400 IE = MASK(5015K,1,IBFU,IB,LAST)        ; FIND END OF LINE.
      IF (IE.EQ.-1) IE = LAST
      IBFU(IE) = 15K                         ; SINGLE CR ENDS OUTPUT LINE.
      N = IE - IB + 1                        ; LENGTH OF LINE.
      IF (N.GT.81) GOTO 800
      CALL PACK(IBFU(IB),N,LINE)             ; LOAD OUTPUT ARRAY.
      CALL WRS(ICHN,LINE,N,IER)              ; WRITE LINE.
      IF (IER.NE.1) GOTO 800
      IF (IE.EQ.LAST) GOTO 500               ; FIND START OF NEXT LINE.
      IB = MASK(20203K,1,IBFU,IE,LAST)
      IF (IB.EQ.-1) GOTO 800
      IF (IB.NE.LAST) GOTO 400
  500 IF (ISP.EQ.1) CALL WRL(ICHN,"<40><15>",N,IER)   ; BLANK LINE
      IF (IER.NE.1) GOTO 800
      GOTO 900                               ; END OF LOOP.
  800 IRER = -1                              ; END WITH ERROR.
  900 RETURN
      END
```

```
      SUBROUTINE GVER(KEY,IVR,IBF,IER)
      DIMENSION KEY(5), KREC(20), IBF(128)
C
C      THIS SUBROUTINE GETS THE (IVR)TH PREVIOUS VERSION OF THE PRODUCT KEY.
C      IER RETURNS 1 IF THE CALL IS SUCCESSFUL.
C
      IER = 1
      CALL KSRCF(KEY,KREC,IER)              ; GET PRODUCT.
      IF (IER.NE.1) GOTO 800
      IF (IVR.EQ.0) GOTO 200
      DO 100 I = 1, IVR                     ; CALL PREVIOUS VERSIONS.
       CALL PRVRF(IER)
       IF (IER.NE.1) GOTO 800
  100 CONTINUE
  200 CALL RDBKF(0,IBF,IER)
      IF (IER.NE.1) GOTO 800
      GOTO 900
  800 IER = -1
  900 RETURN
      END
```

```
      FUNCTION MASK(MSK,LMSK,IUP,IBGN,ISTOP)
      DIMENSION MSK(1), IUP(1)
C
C     THIS FUNCTION SEARCHES IUP (AN UNPACKED ARRAY OF ASCII CHARACTERS) FROM
C     IBGN TO ISTOP FOR A STRING WHICH MATCHES THE MASK MSK.  MSK IS LMSK
C     WORDS IN LENGTH AND HAS THE FOLLOWING CHARACTERISTICS:
C
C          1.  IF THE LEFT BYTE OF A WORD IS 0, THE RIGHT BYTE NEEDS AN
C              EXACT MATCH.
C          2.  IF THE LEFT BYTE IS NON-ZERO, THEN A MATCH MUST BE GREATER
C              THAN OR EQUAL TO THE LEFT BYTE, AND LESS THEN OR EQUAL TO THE
C              RIGHT BYTE.  "09" OR "AZ", E. G.
C          3.  IF A WORD EQUALS -1, IT WILL MATCH ANY BYTE (A WILD CARD).
C
C     THE FUNCTION RETURNS THE LOCATION IN IUP OF THE BEGINNING OF THE
C     SUBSTRING.  MASK RETURNS -1 IF THE SEARCH FAILS.
C
      LIMIT = ISTOP - LMSK + 1
      DO 600 MASK = IBGN, LIMIT
       DO 500 I = 1, LMSK
        M = MSK(I)
        IF (M.EQ.-1) GOTO 500              ;WILD CARD.
        ML = ISHFT(M,-8)
        MR = IAND(M,377K)
        L = IUP(MASK+I-1)
        IF (ML.NE.0) GOTO 200              ;NEED AN EXACT MATCH.
         IF (MR.EQ.L) GOTO 500
         GOTO 600
  200   IF (L.LT.ML.OR.L.GT.MR) GOTO 600   ;RANGE FOR MATCH.
  500  CONTINUE
       GOTO 700
  600 CONTINUE
      MASK = -1
  700 RETURN                               ;UNSUCCESSFUL SEARCH.
      END
```

```
      INTEGER FUNCTION JDATE(MONTH,IDAY,IYEAR)
C
C      THIS FUNCTION WILL GENERATE A JULIAN DATE FOR THE MONTH, DAY, AND
C      YEAR INPUT.  IF THERE IS A PROBLEM WITH THE INPUT DATA, THE FUNC-
C      TION WILL RETURN A VALUE OF ZERO.
C
      COMMON /QJULQ/ MLIST(12)
      DATA MLIST/31,28,31,30,31,30,31,31,30,31,30,31/
      JDATE = 0                                  ;CHECK INPUT DATA.
      IF (MONTH.LT.1.OR.MONTH.GT.12) RETURN
      IF (IYEAR.LT.1) RETURN
      IF (IDAY.LT.1) RETURN
      MLIST(2) = 28
      I = IYEAR/4*4                              ;IS THIS A LEAP YEAR?
      IF (I.EQ.IYEAR) MLIST(2) = 29
      I = IYEAR/100*100
      IF (I.EQ.IYEAR) MLIST(2) = 28
      I = IYEAR/400*400
      IF (I.EQ.IYEAR) MLIST(2) = 29
      IF (IDAY.GT.MLIST(MONTH)) RETURN
      DO 100 I = 1,MONTH                         ;COMPUTE JULIAN DATE.
  100 JDATE = JDATE + MLIST(I)
      JDATE = JDATE - MLIST(MONTH) + IDAY
      RETURN
      END
```

```
C             LDFTASUM.SV                SET UP FTASUM.DA FILE.
      DIMENSION IBF(256)
C
C      THIS PROGRAM CONDUCTS A CONVERSATION AT THE SYSTEM CONSOLE, AND USES
C      THE DATA TO LOAD THE FTASUM.DA FILE WITH DEFAULT VALUES.
C
C      RLDR LDFTASUM <UTIL FORT>.LB
C
      CALL CRAND("FTASUM.DA",IER)          ; SEE IF THE FILE EXISTS.
      IF (IER.EQ.1) GOTO 100
      TYPE "FTASUM.DA ALREADY EXISTS."

      WRITE (10,900)
      READ  (11,910) I
      IF (ISHFT(I,-8).EQ.116K) CALL EXIT
  100 TYPE "AT WHAT TIME DO THE THREE SHIFTS BEGIN?"   ; LOAD SHIFT TIMES.
      ACCEPT IBF(1), IBF(2), IBF(3)
      DO 200 I = 1, 3                      ; LOAD WINDOW TIMES.
      WRITE (10,920) I
      J = (I+1)*2
      J1 = J + 1
      ACCEPT IBF(J), IBF(J1)
  200 CONTINUE
      WRITE (10,930)                       ; LOAD NODE IDENTIFIERS.
      ACCEPT N
      DO 300 I = 1, N
      WRITE (10,940) I
      J = 8 + (2*I)
      J1 = J + 1
      READ  (11,950) IBF(J), IBF(J1)
  300 CONTINUE
      IBF(10+2*N) = 0
      CALL GCHN(IC,IER)                    ; OPEN FILE AND WRITE DATA.
      CALL OPENE(IC,"FTASUM.DA",0,IER)
      CALL WRB(IC,IBF,0,1,IER)
      IF (IER.NE.1) TYPE "CAN'T WRITE TO FILE!"
      CALL KLOSE(IC,IER)
      CALL EXIT
  900 FORMAT(" DO YOU WANT TO REPLACE IT?  ",Z)
  910 FORMAT(A1)
  920 FORMAT(" WINDOW TIMES FOR SHIFT ",I1,"?  ",Z)
  930 FORMAT(" HOW MANY NODES WILL BE PROCESSED?  ",Z)
  940 FORMAT(" I. D. FOR NODE ",I2,"?  ",Z)
  950 FORMAT(A2,A1)
      END
```

APPENDIX: A SAMPLE RUN OF LDFTASUM.SV

```
LDFTASUM
FTASUM.DA ALREADY EXISTS
DO YOU WANT TO REPLACE IT?  YES
AT WHAT TIME DO THE THREE SHIFTS BEGIN?
10,15,22
WINDOW TIMES FOR SHIFT 1?  22,10
WINDOW TIMES FOR SHIFT 2?  10,15
WINDOW TIMES FOR SHIFT 3?  15,22
HOW MANY NODES WILL BE PROCESSED? 1
I.D. FOR NODE  1?  CLE
R
```

FTASUM: AVIATION FORECAST SUMMARIES

PART A: INFORMATION AND INSTALLATION

PROGRAM NAME: FTASUM.SV

AAL ID:
REV NO.: 1.ØØ

PURPOSE: Prints a summary of aviation terminal forecasts and their verifying observations in chronologic order. The operator can specify the time window through switches in the run command. If no times are supplied, the program chooses one of three standard windows which is specified by the site. Up to sixteen nodes can be summarized, but most offices will only want one.

PROGRAM INFORMATION:

Development Programmer:
Matthew R. Peroutka
Location: WSFO Cleveland, Ohio
Phone: (FTS) 942-4949
Language: FORTRAN IV/Rev. 5.6Ø
Date: 09/21/84
Running Time: 30 - 40 seconds per station
Disk Space:
Program 25 RDOS blocks
Data 1 RDOS block

Maintenance Programmer:
Matthew R. Peroutka
Location: WSFO Cleveland, OH
Phone: (FTS) 942-4949
Type: Normal
Revision Date:

PROGRAM REQUIREMENTS

Program Files:

Name	Comments
FTASUM.SV	actual program
LDFTASUM.SV	needed only for setup

Data Files:

Name	DP Location	R/W	Comments
FTASUM.DA	DPØ	Read	information table

AFOS Products:

ID	Action	Comments
cccFTAxxx cccSAOxxx	Input	Chosen by selecting nodes in setup program, LDFTASUM.SV

LOAD LINE

RLDR FTASUM FTSOP FTSKY VNUMS FTSTA GVER FTSPT MASK JDATE <BG UTIL FORT>.LB
RLDR FTASUM <UTIL FORT>.LB

PROGRAM INSTALLATION

1. LDFTASUM.SV can be run from a floppy as it will only be needed once. FTASUM.SV and FTASUM.DA can reside on DPØ or DPØF with links to DPØ.

2. Run LDFTASUM at the Dasher by typing, LDFTASUM. LDFTASUM will ask for the three default time windows. It will also ask for the times when each window should be used. A list of nodes in then input. See sample dialog in Appendix of ERCP #25.

FTASUM: AVIATION FORECAST SUMMARIES

PART B: EXECUTION AND ERROR CONDITIONS

PROGRAM NAME: FTASUM.SV

AAL ID:
REV. NO.: 1.ØØ

PROGRAM EXECUTION:

The basic command is RUN:FTASUM. This will send a summary for the default time period to the printer-plotter module. Local switches can be used to select a specific time period.

/B for time to begin
/E for time to end.

For example: RUN:FTASUM 2/B 1Ø/E

will summarize terminal forecasts and observations from 2ØØ GMT to 1ØØØ GMT.

An alert is sent to the originating console when the program completes.

ERROR CONDITIONS:

ADM	MEANING
PPM TROUBLE	Program can't access PPM ($LPT)

DASHER	MEANING
CAN'T WRITE TO FILE (LDFTASUM)	Problem opening or writing to FTASUM.DA

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.

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:

ENVIRONMENTAL SCIENCE INFORMATION CENTER (D822)
ENVIRONMENTAL DATA AND INFORMATION SERVICE
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
U.S. DEPARTMENT OF COMMERCE

6009 Executive Boulevard
Rockville, MD 20852