A
QC
874.3
U68
no.23

NOAA Western Region Computer Programs and Problems NWS WRCP - No. 23

# ACCESSING THE AFOS DATABASE

Salt Lake City, Utah
January 1981

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

## PREFACE

This Western Region publication series is considered as a subset of our Technical Memorandum series. This series will be devoted exclusively to the exchange of information on and documentation of computer programs and related subjects. This series was initiated because it did not seem appropriate to publish computer program papers as Technical Memoranda; yet, we wanted to share this type of information with all Western Region forecasters in a systematic way. Another reason was our concern that in the developing AFOS-era there will be unnecessary and wasteful duplication of effort in writing computer programs in National Weather Service (NWS). Documentation and exchange of ideas and programs envisioned in this series hopefully will reduce such duplication. We also believe that by publishing the programming work of our forecasters, we will stimulate others to use these programs or develop their own programs to take advantage of the computing capabilities AFOS makes available.

We solicit computer-oriented papers and computer programs from forecasters for us to publish in this series. Simple and short programs should not be prejudged as unsuitable.

The great potential of the AFOS-era is strongly related to local computer facilities permitting meteorologists to practice in a more scientific environment. It is our hope that this new series will help in developing this potential into reality.

NOAA WESTERN REGION COMPUTER PROGRAMS AND PROBLEMS NWS WRCP

1 Standardized Format for Computer Series.
2 AFOS Crop and Soil Information Report Programs. Ken Mielke, July 1979.
3 Decoder for Significant Level Transmissions of Raobs. John A. Jannuzzi, August 1979.
4 Precipitable Water Estimate. Elizabeth Morse, October 1979.
5 Utah Recreational Temperature Program. Kenneth M. Labas, November 1979.
6 Normal Maximum/Minimum Temperature Program for Montana. Kenneth Mielke, December 1979.
7 Plotting of Ocean Wave Energy Spectral Data. John R. Zimmerman, December 1979.
8 Raob Plot and Analysis Routines. John Jannuzzi, January 1980.
9 The SWAB Program. Morris S. Webb, Jr., April 1980. (PB80-196041)
10 Flash-Flood Procedure. Donald P. Laurine and Ralph C. Hatch, April 1980. (PB80-298658)
11 Program to Forecast Probability of Summer Stratus in Seattle Using the Durst Objective Method. John Zimmerman, May 1980.
12 Probability of Sequences of Wet and Dry Days. Hazen H. Bedke, June 1980. (PB80-223340)
13 Automated Montana Hourly Weather Roundup. Joe L. Johnston, July 1980. (PB81-102576)
14 Lightning Activity Levels. Mark A. Mollner, July 1980. (PB81-108300)
15 Two Fortran Applications of Wind-Driven Ekman Water Transport Theory: Upwelling Index and Storm Tide. Kent S. Short, July 1980. (PB81-102568)
16 AFOS System Local Database Save and Rebuild Procedures or A Master Doomsday Program. Brian W. Finke, July 1980. (PB81-108342)
17 AFOS/RDOS Translator Subroutine. Morris S. Webb, Jr., August 1980. (PB81-108334)
18 AFOS Graphics Creation from Fortran. Alexander E. MacDonald, August 1980.
19 DATAKEYØ Repair Program. Paul D. Tolleson, August 1980. (PB81-102543)
20 Contiguous File Transfer from the DPCM to the DCM. Paul D. Tolleson, Sept. 1980. (PB81-128035)
21 Freezing Level Program. Kenneth B. Mielke, September 1980. (PB81-128043)
22 Radar Boresighting Verification Program. Thomas E. Adler, November 1980.

A
QC
874.3
U68
no. 23

NOAA Western Region Computer Programs and Problems NWS WRCP - No. 23

# ACCESSING THE AFOS DATABASE

Matthew Peroutka

National Weather Service Forecast Office
Great Falls, Montana
January 1981

UNITED STATES
DEPARTMENT OF COMMERCE
Philip M. Klutznick, Secretary

NATIONAL OCEANIC AND
ATMOSPHERIC ADMINISTRATION
Richard A. Frank, Administrator

National Weather
Service
Richard E. Hallgren, Director

SILVER SPRING
CENTER
MAR 2
N.O.A.A.
U. S. Dept. of Commerce

81 1125

This Technical Memorandum has been reviewed and is approved for publication by Scientific Services Division, Western Region.

L. W. Snellman, Chief
Scientific Services Division
Western Region Headquarters
Salt Lake City, Utah

CONTENTS


| | | Page |
|---|---|---|
| I. | General Information . . . . . | 1 |
| II. | Application . . . . . | 1 |
| III. | Procedures . . . . . | 2 |
| IV. | Program Listings . . . . . | 4 |

# ACCESSING THE AFOS DATABASE

Matthew Peroutka
National Weather Service Forecast Office
Great Falls, Montana

## I. General Information

One of the more evasive problems associated with AFOS background programming is the user/program interface. Since most background programs cannot prompt the user, input must be contained in the command line or entered into a preformat. Using a preformat forces the user to go through a few rather time-consuming (and disk-consuming) operations. These are:

1. Have the user store the input data in a suitable product.

2. Have the user (or a procedure) issue a SAVE: command to deposit the data in a FORTRAN-formatted RDOS file on DPØ.

3. Have the user (or the procedure) wait for the SAVE: command to finish its business, and then

4. Have the user (or the procedure) issue the RUN: command to start execution of the background program.

This process is cumbersome, slows execution time, and tends to fill the disk with redundant data. As more and more applications programs vie for space on the disk, it will be necessary to find more ways to efficiently utilize available space.

The subroutines described in this paper will allow a moderately skilled FORTRAN programmer to access the database in a simple and direct manner. They use some of the routines from BG.LB, the AFOS Background Library.

## II. Application

### A. Program Purpose and Function:

The system of subroutines described in this paper should make it possible for a moderately skilled FORTRAN programmer to access the database one line at a time and extract the letters or numbers needed.

### B. Derivation:

The system uses two arrays in a common block structure. The first array acts as a buffer. When the subroutine AFREAD is first called, the first block of data in the product is loaded into this array. AFREAD simply scans through this block, character-by-character, until a terminator sequence (OCTAL 15'S followed by OCTAL 12'S) is read.

The second array is the current line of the product. Functions INTCVT, FLTCVT, and subroutine CHRCVT all operate upon this array to extract the necessary numerical values from the ASCII characters.

C. Machine Requirements

Since these are subroutines and not fully loaded programs, it is difficult to estimate the amount of machine code required. It is safe to say, however, that including these subroutines in a program will not add more than 10K to the program size.

It is almost impossible to estimate the effect on execution time. Since these routines were written in FORTRAN and then compiled, it is reasonable to expect their machine-language counterparts (FORTRAN read and write statements) to be more efficient. However, the use of these subroutines will definitely be faster than issuing a SAVE: command, pausing, and executing a background program.

D. Comments and Restrictions

As mentioned before, the subroutine uses a labeled common block. Its name is QARDQ. If the programmer wishes to use a labeled common block in the calling program, the programmer's common block should have a different name.

There are many different formats used in the AFOS database. The author has written these routines to handle as many different formats as he could find. That does not preclude, however, the possibility of new formats finding their way into the database. Special care was taken to ensure that the subroutines could read virtually anything produced by the ADM message composition mode (since this is where most program input will originate). So, as a general rule--if the subroutines have trouble reading a product, call it up on an ADM, EDIT the product (E:), and store it back into database.

The subroutines require the files BG.LB, UTIL.LB, and FORT.LB to successfully reload.

## III. Procedures

At this point, it will be necessary to deviate from the prescribed format to describe each subroutine and its function.

### A. AFREAD

AFREAD is the "workhorse" of the system. It will open a new product for reading, read a line from the current product, or "rewind" the current product (initialize the product so the next line read will be the first line). The calling sequences are as follows:

```
CALL AFREAD(1,IPROD,$IERR)
```

This CALL will open a product for reading. The name of the product (CCCNNNXXX) is stored in IPROD (an array or a literal string). If AFREAD cannot open the desired product for any reason, program control will return to line number IERR. IPROD should contain at least nine letters to guarantee success. If XXX is only two letters, e.g., a state abbreviation, pad it with a space or two. If a new product is needed, simply repeat this call with a different product name.

```
Example:  CALL AFREAD(1,"GTFLFPGTF",$8ØØ)
```

Open GTFLFPGTF for reading. If the product cannot be opened, return control to line 8ØØ.

```
CALL AFREAD(2,IOUT,$IEND,$IERR)
```

This CALL will load the next line of the current product into the integer array IOUT, two ASCII characters per word. IOUT must have a dimension of at least 4Ø. If the last line of the current product has already been read, program control will return to statement number IEND. If some other problem occurs, control will return to statement number IERR.

```
EXAMPLE:  DIMENSION IOUT(4Ø)
          CALL AFREAD(2,IOUT,$8ØØ,$85Ø)
```

After this CALL, the array IOUT will contain the current line of the product. IOUT will always contain 8Ø characters, no matter how short the current line. The extra characters will be blanks. If the last line of the current product has already been read, control will return to line number 8ØØ. If some other problem occurs, control will return to line number 85Ø.

```
CALL AFREAD(3,DUMMY,$IERR)
```

This CALL is used to "rewind" the current product. The second argument is not used in this case. If the subroutine returns successfully, the next read will read the first line of the current product. If an error results, program control will return to line number IERR.

```
Example:  CALL AFREAD(3,X,$8ØØ)
```

Rewind the current product. If there is a problem, return to line 8ØØ.

The following routines are used to manipulate the ASCII characters in the current line. The first two are functions for integer and real conversion, and the last one is a subroutine to move the characters.

```
FUNCTION INTCVT(IBGN,N)
```

This function will scan the current line and extract integer values from ASCII characters. The scan starts at character IBGN, and N characters are scanned. If the function returns successfully, INTCVT contains the desired value. Blank spaces between numbers are considered zeros; leading or trailing blanks are ignored. Integers cannot be larger than 32,767 or smaller than -32,767.

Example: I = INTCVT(5,5)

Look at characters five through ten and convert those numbers into an integer value. Put that integer value into the Variable I.

FUNCTION FLTCVT(IBGN,N)

This function will scan the current line and extract floating point values from the ASCII characters. The scan starts with character number IBGN, and N characters are scanned. If the function returns successfully, FLTCVT contains the desired value. Blank spaces between numbers are considered zeros. Leading or trailing blanks are ignored. Numbers must be in floating point notation, not exponential, i.e., -47.55 is acceptable, but -.4755 EØ2 is not.

Example: X = FLTCVT(5,5)

Look at characters five through ten and convert those characters into a real value. Put that real value into the Variable X.

SUBROUTINE CHRCVT(IBGN,N,IOUT)

This subroutine will start with character IBGN and transfer N characters from the current line into the integer array IOUT. IOUT will have two characters in each word.

```
Example:  DIMENSION IOUT(3)
          CALL CHRCVT(5,5,IOUT)
```

Transfer characters five through ten from the current line into the array IOUT, two characters per word.

## IV. Program Listings

```
      SUBROUTINE AFREAD(IOPT,IOUT,IEND,IERR)
      COMMON/QARDQ/IOUTU(8Ø),I,IBFU(257),KREC(2Ø)
      DIMENSION IOUT(4Ø),IBFP(128)
      GOTO (1ØØ,6ØØ,3ØØ),IOPT
C     ---------> IOPT = 1.  OPEN THE NEW PRODUCT. <----------
  1ØØ CALL KSRCF(IOUT,KREC,IER)
      IF (IER.NE.1) RETURN IEND
C     -----> "REWIND" THE PRODUCT.  IOPT = 1 OR 3. <------
  3ØØ CALL RDBKF(Ø,IBFP,IER)
      IF (IER.NE.1) RETURN IEND
      CALL UNPACK(IBFP,256,IBFU)
      I = 5
C      ---------------> READ A LINE.  <-------------
  6ØØ II = 1
      DO 625 J = 1,8Ø
  625  IOUTU(J) = 4ØK
  65Ø IF (I.LE.256) GOTO 7ØØ
       CALL NXBKF(IBFP,IER)
       IF (IER.NE.1) RETURN IERR
       CALL UNPACK(IBFP,256,IBFU)
      I = 5
  7ØØ IF (IBFU(I).EQ.2Ø3K) RETURN IEND
      IF (IBFU(I).EQ.15K) GOTO 8ØØ
       IF (IBFU(I).EQ.12K) GOTO 75Ø
        IF (II.GE.81) GOTO 875
         IOUTU(II) = IBFU(I)
         II = II + 1
  75Ø I = I + 1
      GOTO 65Ø
  8ØØ IF (I.LT.256) GOTO 825
       CALL NXBKF(IBFP,IER)
      IF (IER.NE.1) RETURN IERR
      CALL UNPACK(IBFP,256,IBFU)
      IBFU(4) = 15K
      I = 4
  825 IF (IBFU(I+1).EQ.12K) GOTO 85Ø
       I = I + 1
      GOTO 65Ø
  85Ø I = I + 2
  875 IF (IOPT.EQ.2) CALL PACK(IOUTU,8Ø,IOUT)
      RETURN
      END
```

```
      INTEGER FUNCTION INTCVT(IBGN,N)
C
C     THIS FUNCTION IS USED WITH THE SUBROUTINE AFREAD.  ASCII
C     CHARACTERS IN THE CURRENT LINE ARE SCANNED AND INTERPRETED
C     AS INTEGERS.  THE SCAN BEGINS WITH CHARACTER IBGN.  N
C     CHARACTERS ARE SCANNED.
C
      COMMON/QARDQ/IOUTU(8Ø)
      LOGICAL NEG
      INTCVT = Ø
      NEG = .FALSE.
      IEND = IBGN + N - 1
  1ØØ IF (IOUTU(IEND).NE.32) GOTO 2ØØ
       IF (IEND.EQ.IBGN) RETURN
       IEND = IEND - 1
       GOTO 1ØØ
  2ØØ DO 25Ø I = IBGN,IEND
       IF (IOUTU(I).NE.32) GOTO 3ØØ
  25Ø CONTINUE
      RETURN
  3ØØ IF (IOUTU(I).EQ.43) GOTO 4ØØ
       IF (IOUTU(I).NE.45) GOTO 5ØØ
       NEG = .TRUE.
  4ØØ I = I + 1
  5ØØ J = I
      DO 6ØØ I = J,IEND
       IF (IOUTU(I).EQ.32) IOUTU(I) = 48
       IF (IOUTU(I).LT.48.OR.IOUTU(I).GT.57) GOTO 8ØØ
       INTCVT = INTCVT*1Ø + IOUTU(I) - 48
  6ØØ CONTINUE
      IF (NEG) INTCVT = -INTCVT
      RETURN
  8ØØ CALL SPCHR("ERROR IN INTCVT:  ILLEGAL CHARACTER<12>",I)
      STOP
      END
```

```
      REAL FUNCTION FLTCVT(IBGN,N)
C
C     THIS FUNCTION IS USED WITH THE SUBROUTINE AFREAD.  ASCII
C     CHARACTERS IN THE CURRENT LINE ARE SCANNED AND INTERPRETED
C     AS REAL NUMBERS.  IF NO DECIMAL POINT IS DETECTED, IT IS
C     ASSUMED TO FOLLOW THE LAST NUMERAL IN THE FIELD.  THE SCAN
C     BEGINS WITH CHARACTER IBGN.  N CHARACTERS ARE SCANNED.
C
      COMMON/QARDQ/IOUTU(8Ø)
      LOGICAL NEG
      FLTCVT = Ø.
      NEG = .FALSE.
       IEND = IBGN + N - 1
 1ØØ   IF (IOUTU(IEND).NE.32) GOTO 2ØØ
        IF (IEND.EQ.IBGN) RETURN
        IEND = IEND - 1
        GOTO 1ØØ
 2ØØ  DO 25Ø I = IBGN,IEND
       IF (IOUTU(I).NE.32) GOTO 3ØØ
 25Ø  CONTINUE
      RETURN

 3ØØ   IF (IOUTU(I).EQ.43) GOTO 4ØØ
        IF (IOUTU(I).NE.45) GOTO 5ØØ
        NEG = .TRUE.
 4ØØ  I = I + 1
 5ØØ  J = I
      DO 6ØØ I = J,IEND
       IF (IOUTU(I).EQ.32) IOUTU(I) = 48
       IF (IOUTU(I).LT.48.OR.IOUTU(I).GT.57) GOTO 7ØØ
       FLTCVT = FLTCVT*1Ø + IOUTU(I) - 48
 6ØØ  CONTINUE
      IF (NEG) FLTCVT = -FLTCVT
      RETURN
 7ØØ  IF (IOUTU(I).NE.46) GOTO 8ØØ
      J = I + 1
      DIV = 1Ø.
      DO 75Ø I = J,IEND
       IF (IOUTU(I).EQ.32) IOUTU(I) = 48
       IF (IOUTU(I).LT.48.OR.IOUTU(I).GT.57) GOTO 8ØØ
       FLTCVT = FLTCVT + (IOUTU(I) - 48)/DIV
       DIV = DIV*1Ø.
 75Ø  CONTINUE
      IF (NEG) FLTCVT = -FLTCVT
      RETURN
 8ØØ  CALL SPCHR("ERROR IN INTCVT:  ILLEGAL CHARACTER",I)
      STOP
      END
```

```
      SUBROUTINE CHRCVT(IBGN,N,IOUT)
C
C      THIS SUBROUTINE IS USED WITH SUBROUTINE AFREAD.  ASCII
C      CHARACTERS IN THE CURRENT LINE ARE SCANNED AND TRANS-
C      FERRED TO THE INTEGER ARRAY IOUT, TWO CHARACTERS PER
C      WORD.  THE SCAN BEGINS WITH CHARACTER IBGN.  N CHARAC-
C      TERS ARE TRANSFERRED.

      COMMON/QARDQ/IOUTU(8Ø)
      CALL PACK(IOUTU(IBGN),N,IOUT)
      RETURN
      END
```

NOAA CENTRAL LIBRARY
CIRC QC874.3 .U68 no.23
Peroutka, Ma Accessing the AFOS databas
3 8398 0003 3998 0

# 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**