National Aeronautics and Space Administration
Goddard Earth Science
Data Information and Services Center (GES DISC)

README Document for the Nimbus-6 SCAMS Scanning Microwave Spectrometer Level 2 Data Product

SCAMSN6L2

Last Revised 08/29/2014

Goddard Earth Sciences Data and Information Services Center (GES DISC)
http://disc.gsfc.nasa.gov
NASA Goddard Space Flight Center
Code 610.2
Greenbelt, MD 20771 USA

Prepared By:

James E. Johnson	08/29/2014
Name GES DISC GSFC Code 610.2	Date

Reviewed By:

Name	mm/dd/yyyy
Name GSFC Code xxx	Date
Name	mm/dd/yyyy
Name GSFC Code xxx	Date

Goddard Space Flight Center
Greenbelt, Maryland

Revision History

Revision Date	***Changes***	***Author***
08/29/2014	Original	James E. Johnson

Table of Contents

1. Introduction 5
1.1 Data Product Description 5
1.1.1 The Scanning Microwave Spectrometer 5
1.1.2 Nimbus-6 Overview 5
1.2 Algorithm Background 6
1.3 Data Disclaimer 6
2. Data Organization 7
2.1 File Naming Convention 7
2.2 File Format and Structure 7
2.3 Key Science Data Fields 9
3. Data Contents 11
3.1 Data Record 11
3.2 Metadata 13
4. Reading the Data 15
5. Data Services 30
5.1 Reverb 30
5.2 FTP 30
6. More Information 31
6.1 Web Resources 31
6.2 Point of Contact 31
6.3 References 31
7. Acknowledgements 32
Acronyms 32

1. Introduction

This document provides basic information on using the Nimbus-6 Scanning Microwave Spectrometer (SCAMS) Level-2 data product.

1.1 Data Product Description

The Scanning Microwave Spectrometer (SCAMS) Level 2 data product contains measurements of 1) brightness temperatures from five channels (22.235 to 55.45 GHz); 2) retrieved water vapor amount and cloud water content over calm oceans; 3) inverted mean layer temperatures for 1000-500 mbar, 500-250 mbar, and 250-100 mbar; and 4) derived temperatures for 14 standard pressure levels from 1000 to 100 mbar. Data files are in a binary record oriented structure. Each file typically contains one full orbit of data. Spatial coverage is global. The data are available from June 15, 1975 (day of year 166) through May 31, 1976 (day of year 152).

This product was previously available from the NASA National Space Science Data Center (NSSDC) under the name SCAMS Output File of Water Vapor and Temperature (SOTA) with the identifier ESAD-00093 (previously 75-052A-10A).

1.1.1 The Scanning Microwave Spectrometer

The Scanning Microwave Spectrometer (SCAMS) was a five channel radiometer flown on the Nimbus-6 spacecraft and launched on June 12, 1975. SCAMS was designed to retrieve tropospheric temperature profiles, and the abundance, over oceans, of water vapor and liquid water in the atmosphere for use in numerical weather forecasting. The instrument measured microwave radiation emitted at frequencies of 22.235, 31.65, 52.85, 53.85 and 55.45 GHz. The SCAMS mission was a follow-on to the successful microwave spectrometer experiment (NEMS) flown on the previous Nimbus-5 satellite.

The swath width of the SCAMS instrument is about 2400 km, with a spatial resolution of about 145 km at nadir to about 330 km at the scan edges. The scanning mechanism rotates once every 16 seconds. The Nimbus-6 SCAMS was operational from June 15, 1975 until the instrument ceased functioning on May 31, 1976, due to jamming of the scan mechanism.

The principal investigator for the SCAMS experiment was Dr. David H. Staelin.

1.1.2 Nimbus-6 Overview

The Nimbus-6 satellite was successfully launched on June 12, 1975. The spacecraft included nine experiments: (1) a Temperature-Humidity Infrared Radiometer (THIR) for measuring day and night surface and cloud top temperatures, as well as the water vapor content of the upper

atmosphere, (2) a High-Resolution Infrared Radiation Sounder (HIRS) for determining vertical temperature profiles, and the distribution of water vapor in the atmosphere, (3) the Scanning Microwave Spectrometer (SCAMS) for obtaining vertical profiles of temperature in the troposphere and abundances of liquid water and water vapor, (4) an Electrically Scanning Microwave Radiometer (ESMR) for determining liquid water content of clouds, the distribution and variation of sea ice cover, and land surfaces characteristics, (5) the Earth Radiation Budget (ERB) experiment for accurate measurements of radiation from the sun and earth, (6) a Limb Radiance Inversion Radiometer (LRIR) for determining the vertical distribution of temperature, ozone and water vapor in the stratosphere and lower mesosphere, (7) a Pressure Modulator Radiometer (PMR) for measuring the temperature structure of the upper stratosphere and mesosphere, (8) the Tropical Wind Energy Conversion and Reference Level Experiment (TWERLE) for determining upper atmospheric winds in the tropics, pressure gradients, and provide a reference level in-coordination with in-situ balloon measurements and (9) a Tracking and Data Relay Experiment (T&DRE) for demonstrating data communication from a low-orbiting spacecraft through a synchronous spacecraft to a ground telemetry station.

The orbit of the satellite can be characterized by the following:

- circular orbit at 1100 km
- inclination of 100 degrees
- period of an orbit is about 107.3 minutes
- orbits cross the equator at 26 degrees of longitude separation
- sun-synchronous

1.2 Algorithm Background

The Nimbus-6 SCAMS data were generated from the spacecraft telemetry, attitude and orbital data. The data were originally processed on IBM 360/75 computers using a 32-bit architecture. Brightness temperatures from the various microwave channels were used to derive water vapor and liquid water abundances over oceans (H_2O channels 1 and 2) temperature profiles (O_2 band channels 3, 4 and 5). Further information on the SCAMS data processing can be found in the Nimbus-6 Users' Guide Section 4.

1.3 Data Disclaimer

The data should be used care and one should first read the Nimbus-6 User’s Guide, section 4 describing the SCAMS experiment. Users should cite this data product in their research.

2. Data Organization

The Nimbus-6 Scanning Microwave Spectrometer Level-2 data product spans the time period from June 15, 1975 to May 31, 1976. Each file typically contains about one orbit worth of data.

2.1 File Naming Convention

The data product files are named according to the following convention:

<Platform>-<Instrument>_<Date>_<Orbit>_<Tape>.<Suffix>

where:

- Platform = name of the platform or satellite (always Nimbus6)
- Instrument = name of the instrument (always SCAMS)
- Date = Data start date and time in UTC in format <YYYY>m<MMDD>t<hhmmss> where
    1. YYYY = 4 digit year (1975 or 1976)
    2. MM = 2 digit month (01-12)
    3. DD = 2 digit day of month (01-31)
    4. hh = 2 digit hour of day (01-23)
    5. mm = 2 digit minute of hour (01-59)
    6. dd = 2 digit second of minute (01-59)
- Orbit = the start orbit number when the data were collected (5 digits preceded by the letter 'o')
- Tape = the id of the tape the file is from (e.g. DS1)
- Suffix = the file format (always TAP)

```
File name example: Nimbus6-SCAMS_1975m0615t214155_o00049_DS1.TAP
```

2.2 File Format and Structure

The data are stored as they were originally written in IBM binary (big-endian) record oriented structured files, hereafter referred to as TAP. The files were written on the original 9-track tapes using a blocked FORTRAN format with a size that is an increment of 1400 byte size records. The typical block contains three data recors and has a block size of 4200 bytes.

Figure 1 below shows a schematic of the SCAMS Level-2 file structure. Please note that this represents the optimal file structure with a 4-byte word marker indicating the block size of 4200, followed by a block of this size, and ending with another 4-byte word marker with the same as the begin block size. Each block contains three 1400 byte records. The file will contain multiple blocks.

Figure 1: SCAMS Level-2 file structure

Offset	Length	Object
0	4	Begin $Block_1$ Marker (value = 4200)
4	4200	$Block_1$ { $Record_1$ (size = 1400 bytes); $Record_2$ (size = 1400 bytes); $Record_3$ (size = 1400 bytes) }
4204	4	End $Block_1$ Marker (value = 4200)
4208	4	Begin $Block_2$ Marker (value = 4200)
4212	4200	$Block_2$ { $Record_1$ (size = 1400 bytes); $Record_2$ (size = 1400 bytes); $Record_3$ (size = 1400 bytes) }
8412	4	End $Block_2$ Marker (value = 4200)
		⋮
(n-1)x4208	4	Begin $Block_n$ Marker (value = 4200)
(n-1)x4208 + 4	4200	$Block_n$ { $Record_1$ (size = 1400 bytes); $Record_2$ (size = 1400 bytes); $Record_3$ (size = 1400 bytes) }
(n-1)x4208 + 4204	4	End $Block_n$ Marker (value = 4200)

where n = block number

Note that this represents the ideal block structure for SCAMS. There are many cases where the files deviate from this structure.

Case 1: A block does not always contain three records. Some blocks may contain two or just a single record. Block sizes will be 2800 or 1400 respectively.

Case 2: Block size is not an increment of the record size. There are examples where the blocks contain additional 4-byte block size markers and even additional 4 byte record size markers. These blocks are typically 4216 in size. These additional markers are usually byte-swapped.

Case 3: Sometimes the block size is given as 4200 indicating three records. Be careful and analyze the first 4 bytes of the record as these may actually be additional 4-byte block size and record size markers embedded in the record. This results in an offset to each record with the last record missing 16 bytes.

Other issues:

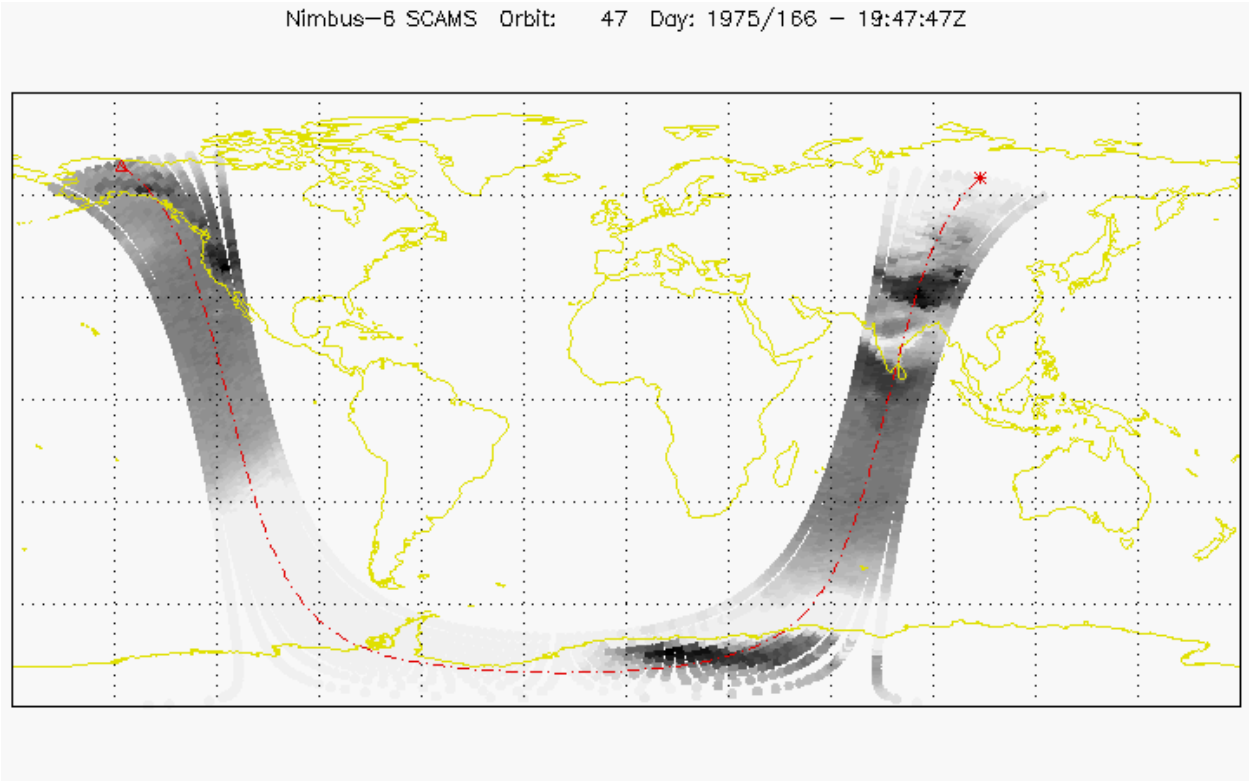
- Some files are missing the last end block marker.
- Some files the last record is incomplete (less than 1400 bytes).
- The list below shows files with incorrectly labelled records that are from previous orbits.

Nimbus6-SCAMS_1975m0626t224255_o00446_DS2.TAP
Nimbus6-SCAMS_1975m0718t063249_o01475_DS7.TAP
Nimbus6-SCAMS_1975m0816t202952_o00880_DS5.TAP
Nimbus6-SCAMS_1975m0915t215614_o01941_DS10.TAP
Nimbus6-SCAMS_1975m0916t060014_o01947_DS10.TAP
Nimbus6-SCAMS_1975m0916t162934_o01950_DS10.TAP
Nimbus6-SCAMS_1975m1103t233958_o01943_DS10.TAP
Nimbus6-SCAMS_1975m1105t214550_o02608_DS13.TAP
Nimbus6-SCAMS_1975m1120t132515_o02265_DS11.TAP
Nimbus6-SCAMS_1976m0113t224544_o02892_DS14.TAP
Nimbus6-SCAMS_1976m0117t144840_o02941_DS14.TAP
Nimbus6-SCAMS_1976m0117t164024_o02942_DS14.TAP
Nimbus6-SCAMS_1976m0420t080421_o04591_DS22.TAP
Nimbus6-SCAMS_1976m0429t063549_o04406_DS21.TAP
Nimbus6-SCAMS_1976m0429t081829_o04410_DS21.TAP

2.3 Key Science Data Fields

The primary science data fields in this data product are the vertical profile of temperature in degrees Kelvin from 1000 to 10 mbar, and integrated abundances of liquid H_2O and water vapor over oceans in units of mm.

Figure 2: Example showing one orbit swath of Temperature (Kelvin) at 850 mbar from SCAMS (darkest values = 270 K; lightest values = 320 Kelvin)



3. Data Contents

The granularity is typically one orbit or roughly 100 minutes of data per data file. Some data files will contain multiple orbits that include orbits found in previous data files (see list in section 2-2). Please check the time stamps in the data records carefully as misplaced data records are identified with the same orbit number.

3.1 Data Record

Note: There is a difference in the table below compared to table 4-8 found in the Nimbus-6 User's Guide at the data fields or parameters "Playback Orbit", "Reference Orbit" and "Spares". It should be "Playback Orbit", "Spare", and "Reference Orbit" as indicated below.

Table 3-1: Data fields within a record or major frame

Field Name	Count	Type	Units	Scale Factor
Day	1	I*2	-	1
Minutes	1	I*2	-	1
Altitude of Spacecraft	1	I*2	km	1
Latitude of Spacecraft	1	R*4	°N	1
Longitude of Spacecraft	1	R*4	°E	1
Data Missing Flag	1	L*1	-	1
Ascending Flag	1	L*1	-	1
Lost Frames Since Last Frame	1	I*2	-	1
Pitch Error	4	I*2	deg	32
Roll Error	4	I*2	deg	32
Digital A Data	160	I*2	-	1
Playback Orbit	1	I*2	-	1
Spare	1	I*2	-	-
Reference Orbit (format = YYDDDHH)	1	I*4	-	-
Housekeeping Temps	12	R*4	°K	1

T_A channel 1	13	I*2	°K	32
T_A channel 2	13	I*2	°K	32
T_A channel 3	13	I*2	°K	32
T_A channel 4	13	I*2	°K	32
T_A channel 5	13	I*2	°K	32
Surface Elevation of Observation	13	I*2	km	32
Latitude of Observation	13	I*2	°N	32
Longitude of Observation	13	I*2	°E	32
T_B channel 1	13	I*2	°K	32
T_B channel 2	13	I*2	°K	32
T_B channel 3	13	I*2	°K	32
T_B channel 4	13	I*2	°K	32
T_B channel 5	13	I*2	°K	32
Surface Reflectivity	13	I*2	%	32
H2O Vapor, Integrated	13	I*2	mm	32
H2O Liquid, Integrated	13	I*2	mm	32
1000 – 500 mb Geopotential Thickness	13	I*2	Dm	32
500 – 250 mb Geopotential Thickness	13	I*2	Dm	32
250 – 100 mb Geopotential Thickness	13	I*2	Dm	32
1000 mb Temperature	13	I*2	°K	32
850 mb Temperature	13	I*2	°K	32
700 mb Temperature	13	I*2	°K	32
500 mb Temperature	13	I*2	°K	32
400 mb Temperature	13	I*2	°K	32
300 mb Temperature	13	I*2	°K	32
250 mb Temperature	13	I*2	°K	32

200 mb Temperature	13	I*2	°K	32
150 mb Temperature	13	I*2	°K	32
100 mb Temperature	13	I*2	°K	32
70 mb Temperature	13	I*2	°K	32
50 mb Temperature	13	I*2	°K	32
30 mb Temperature	13	I*2	°K	32
10 mb Temperature	13	I*2	°K	32
Spares	52	I*2	-	-
Flags	13	I*2	-	-

3.2 Metadata

The metadata are contained in a separate XML formatted file having the same name as the data file with .xml appended to it.

Table 3-2: Metadata attributes associated with the data file.

Name	Description
LongName	Long name of the data product.
ShortName	Short name of the data product.
VersionID	Product or collection version.
GranuleID	Granule identifier, i.e. the name of the file.
Format	File format of the data file.
CheckSumType	Type of checksum used.
CheckSumValue	The value of the calculated checksum.
SizeBytesDataGranule	Size of the file or granule in bytes.
InsertDateTime	Date and time when the granule was inserted into the archive. The format for date is YYYY-MM-DD and time is hh-mm-ss.
RangeBeginningDate	Begin date when the data was collected in YYYY-MM-DD format.

RangeBeginningTime	Begin time of the date when the data was collected in hh-mm-ss format.
RangeEndingDate	End date when the data was collected in YYYY-MM-DD format.
RangeEndingTime	End time of the date when the data was collected in hh-mm-ss format.
PlatformShortName	Short name or acronym of the platform or satellite
InstrumentShortName	Short name or acronym of the instrument
SensorShortName	Short name or acronym of the sensor
GPolygon: PointLatitude	Latitudes of the polygon (rectangle) points that represent the satellite coverage. Each point is identified by its latitude and longitude pair.
GPolygon: PointLongitude	Longitudes of the polygon (rectangle) points that represent the satellite coverage. Each point is identified by its latitude and longitude pair.
Orbit	Satellite orbit number.
ElapsedMinTime	Duration in minutes of data collected during an orbit.

4. Reading the Data

The data are written in a proprietary binary record-oriented format. We provide the following sample FORTRAN code snippet for reading the data:

```
C-----------------------------------------------------------------------
C ^NAME: READ_SCAMS
C
C ^DESCRIPTION:
C     This program opens and reads a Nimbus-6 SCAMS level-2 data file
C     and prints the contents of the file to the screen. Data files
C     consist of blocks of data typically containing three records,
C     though this isn't always the case. See the Nimbus-6 User's Guide
C     Section 4 for the SCAMS file specification.
C
C ^MAJOR VARIABLES:
C     FNAME  - name of input file
C     BLOCK  - buffer for data block typically has three data records
C     BUFF   - buffer for holding temporary 4-byte word
C     WORD   - integer 4-byte word
C     BLKSIZ - size of block in bytes
C     IOS    - I/O status number
C
C ^NOTES:
C     Some blocks contain extra words, some blocks are short, and some
C     records contain extra words while some are short. Often records
C     end prematurely, either they are not complete 1400 size or are
C     missing the end word block size.
C
C     Compile: gfortran -o READ_SCAMS.EXE READ_SCAMS.FOR
C
C ^ORGANIZATION: NASA/GSFC, Code 610.2
C
C ^AUTHOR:  James Johnson
C
C ^ADDRESS: james.johnson@nasa.gov
C
C ^CREATED: Aug. 29, 2014
C-----------------------------------------------------------------------


      CHARACTER          FNAME*1024
      CHARACTER          BLOCK(4200+16)! Buffer = 4200 add 16 for 4 extra words
      CHARACTER          BUFF(4)       ! Buffer to hold 4-byte word
      INTEGER*4          WORD          ! 4-byte word
      INTEGER*4          BLKSIZ        ! Block size header
      EQUIVALENCE        (BUFF, WORD)


C     Get the name of the input data file to read
      PRINT *, 'Enter the name of the input file:'
      READ (5,'(A)') FNAME
```

```
C     Open the specified input file
      OPEN (UNIT=1, FILE=FNAME, STATUS='OLD', ACCESS='DIRECT',
     &      FORM='UNFORMATTED', RECL=1, ERR=99, IOSTAT=IOS)

C     Initialize N (block number) and IOFF (byte offset)
      N=1
      IOFF=0

C     Loop through the file reading all blocks of data
      DO

C       Read the first 4-byte word or block size header
        DO I=1,4
          READ (1, REC=I+IOFF, ERR=91, IOSTAT=IOS) BUFF(I)
        END DO
        IOFF = I+IOFF-1
        BLKSIZ = WORD

C       Next read the block of data
        DO I=1,BLKSIZ
          READ (1, REC=I+IOFF, ERR=92, IOSTAT=IOS) BLOCK(I)
        END DO
        IOFF=I+IOFF-1
        CALL RECBLK(BLOCK, BLKSIZ, N)

C       Finally read the last 4-byte word (should match first block size header)
        DO I=1,4
          READ (1, REC=I+IOFF, ERR=93, IOSTAT=IOS) BUFF(I)
        END DO
        IOFF=I+IOFF-1
        IF (BLKSIZ .NE. WORD) THEN
          PRINT '("BLKSIZ: ",I10," != ",I10)', BLKSIZ, WORD
        ENDIF

        N=N+1

      END DO

C     Close the input file
      CLOSE(1)
      GOTO 100

   91 PRINT '("ERROR: READ FIRST WORD, IOSTAT: ",I5)', IOS
      GOTO 100
   92 PRINT '("ERROR: READ BLOCK ",I4,", IOSTAT: ",I5)', N, IOS
      GOTO 100
   93 PRINT '("ERROR: READ LAST WORD, IOSTAT = ",I5)', IOS
      GOTO 100
   99 PRINT '("ERROR: OPEN IOSTAT =",I5)', IOS

  100 STOP
      END
```

```
C-----------------------------------------------------------------------
C ^SUBROUTINE: RECBLK
C
C     This subroutine will extract records from the block of data
C-----------------------------------------------------------------------

      SUBROUTINE RECBLK(BLOCK, BLKSIZ, N)

      INTEGER*4          BLKSIZ        ! Size of block
      CHARACTER          BLOCK(4200+16)! Buffer = 4200 add 16 for 4 extra words

      CHARACTER          DREC(1400+4)  ! Data record
      CHARACTER          BUFF(4)       ! Buffer to hold 4-byte word
      CHARACTER          TEMP(4)       ! Temporary data buffer
      INTEGER*4          WORD          ! 4-byte word
      INTEGER*4          RECSIZ /1400/ ! Size of data record
      EQUIVALENCE        (BUFF, WORD)


      PRINT '("**************")'
      PRINT '("BLOCK  =",X,I5)', N

C     Get number of records in block of data
      NRECS = BLKSIZ / RECSIZ

C     Check if the block has an extra padded word = blksiz
      IOFF = 0
      BUFF(1:4) = BLOCK(1:4)
      IF (WORD .EQ. BLKSIZ) THEN
        IOFF = 4                      ! skip 4 bytes
      ELSE
C     Sometimes a block has a corrupted word as 2 byte-swapped short ints
        TEMP(1:4) = BUFF(1:4)
        BUFF(1) = TEMP(2)
        BUFF(2) = TEMP(1)
        BUFF(3) = TEMP(4)
        BUFF(4) = TEMP(3)
        IF (WORD .EQ. BLKSIZ) THEN
          IOFF = 4                    ! skip 4 bytes
        END IF
      END IF

      DO I=1,NRECS
        DREC(1:1400) = BLOCK((I-1)*RECSIZ+I*IOFF+IOFF+1:
     &                       I*(RECSIZ+IOFF)+IOFF)
        PRINT '("**************")'
        PRINT '("RECORD =",X,I5)', I
        PRINT '("**************")'
        CALL PRDREC(DREC(1:1400))
      END DO

      RETURN
      END
```

```
C-----------------------------------------------------------------------
C ^SUBROUTINE: PRDREC
C
C     This subroutine will print data records to the screen
C-----------------------------------------------------------------------

      SUBROUTINE PRDREC(DREC)

      CHARACTER          DREC*1400,    ! Data record size, 1400 bytes
     &                   TMP*4         ! Temporary buffer
      INTEGER*2          DAY,          ! Day number of observation
     &                   MINS,         ! Minutes since start of day
     &                   SECS,         ! Seconds of minute
     &                   SATALT        ! Satellite altitude in km
      REAL*4             SATLAT,       ! Satellite latitude in deg_N  ; IBM real
     &                   SATLON        ! Satellite longitude in deg_E ; IBM real
      LOGICAL*1          DATFLG,       ! Data missing somewhere flag
     &                   ASCFLG        ! Ascending flag
      INTEGER*2          LOST,         ! Lost frames since last frame
     &                   PITERR(4),    ! Pitch error in deg.          ; scale=32
     &                   ROLERR(4),    ! Roll error in deg.           ; scale=32
     &                   DIGITA(160),  ! Digital A data
     &                   ORBIT,        ! Orbit number
     &                   SPARE1        ! Spare #1
      INTEGER*4          REFORB        ! Reference orbit (YYDDDHH)
      REAL*4             HKEEPT(12)    ! Housekeeping temp. in K      ; IBM real
      INTEGER*2          TACH1(13),    ! Temperature A channel 1 in K ; scale=32
     &                   TACH2(13),    ! Temperature A channel 2 in K ; scale=32
     &                   TACH3(13),    ! Temperature A channel 3 in K ; scale=32
     &                   TACH4(13),    ! Temperature A channel 4 in K ; scale=32
     &                   TACH5(13),    ! Temperature A channel 5 in K ; scale=32
     &                   ELEV(13),     ! Surface elevation in km      ; scale=32
     &                   LAT(13),      ! Latitude in deg_N            ; scale=32
     &                   LON(13),      ! Longitude in deg_E           ; scale=32
     &                   TBCH1(13),    ! Temperature B channel 1 in K ; scale=32
     &                   TBCH2(13),    ! Temperature B channel 2 in K ; scale=32
     &                   TBCH3(13),    ! Temperature B channel 3 in K ; scale=32
     &                   TBCH4(13),    ! Temperature B channel 4 in K ; scale=32
     &                   TBCH5(13),    ! Temperature B channel 5 in K ; scale=32
     &                   REFLEC(13),   ! Surface reflectivity in %    ; scale=32
     &                   H2OVAP(13),   ! H2O vapor in mm              ; scale=32
     &                   H2OLIQ(13),   ! H2O liquid in mm             ; scale=32
     &                   GPT1K(13),    ! Geop.Thickness 1000-500 in Dm; scale=32
     &                   GPT500(13),   ! Geop.Thickness  500-250 in Dm; scale=32
     &                   GPT250(13),   ! Geop.Thickness  250-100 in Dm; scale=32
     &                   T1000(13),    ! Temperature at 1000 mbar in K; scale=32
     &                   T850(13),     ! Temperature at  850 mbar in K; scale=32
     &                   T700(13),     ! Temperature at  700 mbar in K; scale=32
     &                   T500(13),     ! Temperature at  500 mbar in K; scale=32
     &                   T400(13),     ! Temperature at  400 mbar in K; scale=32
     &                   T300(13),     ! Temperature at  300 mbar in K; scale=32
     &                   T250(13),     ! Temperature at  250 mbar in K; scale=32
     &                   T200(13),     ! Temperature at  200 mbar in K; scale=32
     &                   T150(13),     ! Temperature at  150 mbar in K; scale=32
     &                   T100(13),     ! Temperature at  100 mbar in K; scale=32
     &                   T70(13),      ! Temperature at   70 mbar in K; scale=32
     &                   T50(13),      ! Temperature at   50 mbar in K; scale=32
```

```
     &                    T30(13),      ! Temperature at   30 mbar in K; scale=32
     &                    T10(13),      ! Temperature at   10 mbar in K; scale=32
     &                    SPARE2(52),   ! Spare #2
     &                    FLAGS(13)     ! Flags
      LOGICAL*1           L1TMP
      INTEGER*1           I1TMP
      INTEGER*2           I2TMP
      INTEGER*4           I4TMP
      REAL*4              R4TMP
      EQUIVALENCE  (TMP, L1TMP, I1TMP, I2TMP, I4TMP, R4TMP)


      IOFF = 0
      ISIZ = 2
      DO 11 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   11 CONTINUE
      IOFF = IOFF + ISIZ
      DAY = I2TMP
      PRINT '("DAY    =",X,I5)', DAY

      ISIZ = 2
      DO 13 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   13 CONTINUE
      IOFF = IOFF + ISIZ
      MINS = I2TMP
      PRINT '("MINS   =",X,I5)', MINS

      ISIZ = 2
      DO 15 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   15 CONTINUE
      IOFF = IOFF + ISIZ
      SECS = I2TMP
      PRINT '("SECS   =",X,I5)', SECS

      ISIZ = 2
      DO 17 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   17 CONTINUE
      IOFF = IOFF + ISIZ
      SATALT = I2TMP
      PRINT '("SATALT =",X,I5)', SATALT

      ISIZ = 4
      DO 19 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   19 CONTINUE
      IOFF = IOFF + ISIZ
      CALL IBMFLT(R4TMP, SATLAT)
      PRINT '("SATLAT =",X,G12.6)', SATLAT

      ISIZ = 4
      DO 21 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   21 CONTINUE
```

```
      IOFF = IOFF + ISIZ
      CALL IBMFLT(R4TMP, SATLON)
      PRINT '("SATLON =",X,G12.6)', SATLON

      ISIZ = 1
      TMP(1:1) = DREC(IOFF+ISIZ:IOFF+ISIZ)
      IOFF = IOFF + ISIZ
      DATFLG = L1TMP
      PRINT '("DATFLG =",X,L1)', DATFLG

      ISIZ = 1
      TMP(1:1) = DREC(IOFF+ISIZ:IOFF+ISIZ)
      IOFF = IOFF + ISIZ
      ASCFLG = L1TMP
      PRINT '("ASCFLG =",X,L1)', ASCFLG

      ISIZ = 2
      DO 23 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   23 CONTINUE
      IOFF = IOFF + ISIZ
      LOST = I2TMP
      PRINT '("LOST   =",X,I5)', LOST

      ISIZ = 2
      DO 25 J=1,4
        DO 24 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   24   CONTINUE
        IOFF = IOFF + ISIZ
        PITERR(J) = I2TMP
   25 CONTINUE
      PRINT '("PITERR =",4(X,G12.6))', (PITERR/32.0)

      ISIZ = 2
      DO 27 J=1,4
        DO 26 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   26   CONTINUE
        IOFF = IOFF + ISIZ
        ROLERR(J) = I2TMP
   27 CONTINUE
      PRINT '("ROLERR =",4(X,G12.6))', (ROLERR/32.0)

      ISIZ = 2
      DO 29 J=1,160
        DO 28 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   28   CONTINUE
        IOFF = IOFF + ISIZ
        DIGITA(J) = I2TMP
   29 CONTINUE
      PRINT '("DIGITA =",10(X,I5),/,(8X,10(X,I5)))', DIGITA

      ISIZ = 2
      DO 31 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
```

```
31 CONTINUE
   IOFF = IOFF + ISIZ
   ORBIT = I2TMP
   PRINT '("ORBIT  =",X,I5)', ORBIT

   ISIZ = 2
   DO 33 K=1,ISIZ
     TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
33 CONTINUE
   IOFF = IOFF + ISIZ
   SPARE1 = I2TMP
   PRINT '("SPARE1 =",X,I5)', SPARE1

   ISIZ = 4
   DO 35 K=1,ISIZ
     TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
35 CONTINUE
   IOFF = IOFF + ISIZ
   REFORB = I4TMP
   PRINT '("REFORB =",X,I10)', REFORB

   ISIZ = 4
   DO 37 J=1,12
     DO 36 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
36   CONTINUE
     IOFF = IOFF + ISIZ
     CALL IBMFLT(R4TMP, HKEEPT(J))
37 CONTINUE
   PRINT '("HKEEPT =",5(X,G12.6),/,(8X,5(X,G12.6)))', HKEEPT

   ISIZ = 2
   DO 39 J=1,13
     DO 38 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
38   CONTINUE
     IOFF = IOFF + ISIZ
     TACH1(J) = I2TMP
39 CONTINUE
   PRINT '("TACH1  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TACH1/32.0)

   ISIZ = 2
   DO 41 J=1,13
     DO 40 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
40   CONTINUE
     IOFF = IOFF + ISIZ
     TACH2(J) = I2TMP
41 CONTINUE
   PRINT '("TACH2  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TACH2/32.0)

   ISIZ = 2
   DO 43 J=1,13
     DO 42 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
42   CONTINUE
     IOFF = IOFF + ISIZ
```

```
      TACH3(J) = I2TMP
   43 CONTINUE
      PRINT '("TACH3  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TACH3/32.0)

      ISIZ = 2
      DO 45 J=1,13
        DO 44 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   44   CONTINUE
        IOFF = IOFF + ISIZ
        TACH4(J) = I2TMP
   45 CONTINUE
      PRINT '("TACH4  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TACH4/32.0)

      ISIZ = 2
      DO 47 J=1,13
        DO 46 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   46   CONTINUE
        IOFF = IOFF + ISIZ
        TACH5(J) = I2TMP
   47 CONTINUE
      PRINT '("TACH5  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TACH5/32.0)

      ISIZ = 2
      DO 49 J=1,13
        DO 48 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   48   CONTINUE
        IOFF = IOFF + ISIZ
        ELEV(J) = I2TMP
   49 CONTINUE
      PRINT '("ELEV   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (ELEV/32.0)

      ISIZ = 2
      DO 51 J=1,13
        DO 50 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   50   CONTINUE
        IOFF = IOFF + ISIZ
        LAT(J) = I2TMP
   51 CONTINUE
      PRINT '("LAT    =",5(X,G12.6),/,(8X,5(X,G12.6)))', (LAT/32.0)

      ISIZ = 2
      DO 53 J=1,13
        DO 52 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   52   CONTINUE
        IOFF = IOFF + ISIZ
        LON(J) = I2TMP
   53 CONTINUE
      PRINT '("LON    =",5(X,G12.6),/,(8X,5(X,G12.6)))', (LON/32.0)

      ISIZ = 2
      DO 55 J=1,13
        DO 54 K=1,ISIZ
```

```
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
54    CONTINUE
      IOFF = IOFF + ISIZ
      TBCH1(J) = I2TMP
55 CONTINUE
   PRINT '("TBCH1  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TBCH1/32.0)

   ISIZ = 2
   DO 57 J=1,13
     DO 56 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
56   CONTINUE
     IOFF = IOFF + ISIZ
     TBCH2(J) = I2TMP
57 CONTINUE
   PRINT '("TBCH2  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TBCH2/32.0)

   ISIZ = 2
   DO 59 J=1,13
     DO 58 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
58   CONTINUE
     IOFF = IOFF + ISIZ
     TBCH3(J) = I2TMP
59 CONTINUE
   PRINT '("TBCH3  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TBCH3/32.0)

   ISIZ = 2
   DO 61 J=1,13
     DO 60 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
60   CONTINUE
     IOFF = IOFF + ISIZ
     TBCH4(J) = I2TMP
61 CONTINUE
   PRINT '("TBCH4  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TBCH4/32.0)

   ISIZ = 2
   DO 63 J=1,13
     DO 62 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
62   CONTINUE
     IOFF = IOFF + ISIZ
     TBCH5(J) = I2TMP
63 CONTINUE
   PRINT '("TBCH5  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (TBCH5/32.0)

   ISIZ = 2
   DO 65 J=1,13
     DO 64 K=1,ISIZ
       TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
64   CONTINUE
     IOFF = IOFF + ISIZ
     REFLEC(J) = I2TMP
65 CONTINUE
   PRINT '("REFLEC =",5(X,G12.6),/,(8X,5(X,G12.6)))', (REFLEC/32.0)
```

```
      ISIZ = 2
      DO 67 J=1,13
        DO 66 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   66   CONTINUE
        IOFF = IOFF + ISIZ
        H2OVAP(J) = I2TMP
   67 CONTINUE
      PRINT '("H2OVAP =",5(X,G12.6),/,(8X,5(X,G12.6)))', (H2OVAP/32.0)

      ISIZ = 2
      DO 69 J=1,13
        DO 68 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   68   CONTINUE
        IOFF = IOFF + ISIZ
        H2OLIQ(J) = I2TMP
   69 CONTINUE
      PRINT '("H2OLIQ =",5(X,G12.6),/,(8X,5(X,G12.6)))', (H2OLIQ/32.0)

      ISIZ = 2
      DO 71 J=1,13
        DO 70 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   70   CONTINUE
        IOFF = IOFF + ISIZ
        GPT1K(J) = I2TMP
   71 CONTINUE
      PRINT '("GPT1K  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (GPT1K/32.0)

      ISIZ = 2
      DO 73 J=1,13
        DO 72 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   72   CONTINUE
        IOFF = IOFF + ISIZ
        GPT500(J) = I2TMP
   73 CONTINUE
      PRINT '("GPT500 =",5(X,G12.6),/,(8X,5(X,G12.6)))', (GPT500/32.0)

      ISIZ = 2
      DO 75 J=1,13
        DO 74 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   74   CONTINUE
        IOFF = IOFF + ISIZ
        GPT250(J) = I2TMP
   75 CONTINUE
      PRINT '("GPT250 =",5(X,G12.6),/,(8X,5(X,G12.6)))', (GPT250/32.0)

      ISIZ = 2
      DO 77 J=1,13
        DO 76 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
   76   CONTINUE
```

```
      IOFF = IOFF + ISIZ
      T1000(J) = I2TMP
77 CONTINUE
   PRINT '("T1000  =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T1000/32.0)

   ISIZ = 2
   DO 79 J=1,13
      DO 78 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
78    CONTINUE
      IOFF = IOFF + ISIZ
      T850(J) = I2TMP
79 CONTINUE
   PRINT '("T850   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T850/32.0)

   ISIZ = 2
   DO 81 J=1,13
      DO 80 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
80    CONTINUE
      IOFF = IOFF + ISIZ
      T700(J) = I2TMP
81 CONTINUE
   PRINT '("T700   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T700/32.0)

   ISIZ = 2
   DO 83 J=1,13
      DO 82 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
82    CONTINUE
      IOFF = IOFF + ISIZ
      T500(J) = I2TMP
83 CONTINUE
   PRINT '("T500   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T500/32.0)

   ISIZ = 2
   DO 85 J=1,13
      DO 84 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
84    CONTINUE
      IOFF = IOFF + ISIZ
      T400(J) = I2TMP
85 CONTINUE
   PRINT '("T400   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T400/32.0)

   ISIZ = 2
   DO 87 J=1,13
      DO 86 K=1,ISIZ
        TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
86    CONTINUE
      IOFF = IOFF + ISIZ
      T300(J) = I2TMP
87 CONTINUE
   PRINT '("T300   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T300/32.0)
```

```
      ISIZ = 2
      DO 89 J=1,13
        DO 88 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
88      CONTINUE
        IOFF = IOFF + ISIZ
        T250(J) = I2TMP
89    CONTINUE
      PRINT '("T250   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T250/32.0)

      ISIZ = 2
      DO 91 J=1,13
        DO 90 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
90      CONTINUE
        IOFF = IOFF + ISIZ
        T200(J) = I2TMP
91    CONTINUE
      PRINT '("T200   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T200/32.0)

      ISIZ = 2
      DO 93 J=1,13
        DO 92 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
92      CONTINUE
        IOFF = IOFF + ISIZ
        T150(J) = I2TMP
93    CONTINUE
      PRINT '("T150   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T150/32.0)

      ISIZ = 2
      DO 95 J=1,13
        DO 94 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
94      CONTINUE
        IOFF = IOFF + ISIZ
        T100(J) = I2TMP
95    CONTINUE
      PRINT '("T100   =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T100/32.0)

      ISIZ = 2
      DO 97 J=1,13
        DO 96 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
96      CONTINUE
        IOFF = IOFF + ISIZ
        T70(J) = I2TMP
97    CONTINUE
      PRINT '("T70    =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T70/32.0)
```

```
      ISIZ = 2
      DO 99 J=1,13
        DO 98 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
 98     CONTINUE
        IOFF = IOFF + ISIZ
        T50(J) = I2TMP
 99   CONTINUE
      PRINT '("T50    =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T50/32.0)

      ISIZ = 2
      DO 101 J=1,13
        DO 100 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
100     CONTINUE
        IOFF = IOFF + ISIZ
        T30(J) = I2TMP
101   CONTINUE
      PRINT '("T30    =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T30/32.0)

      ISIZ = 2
      DO 103 J=1,13
        DO 102 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
102     CONTINUE
        IOFF = IOFF + ISIZ
        T10(J) = I2TMP
103   CONTINUE
      PRINT '("T10    =",5(X,G12.6),/,(8X,5(X,G12.6)))', (T10/32.0)

      ISIZ = 2
      DO 105 J=1,52
        DO 104 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
104     CONTINUE
        IOFF = IOFF + ISIZ
        SPARE2(J) = I2TMP
105   CONTINUE
      PRINT '("SPARE2 =",10(X,I5),/,(8X,10(X,I5)))', SPARE2

      ISIZ = 2
      DO 107 J=1,13
        DO 106 K=1,ISIZ
          TMP(K:K) = DREC(IOFF+ISIZ+1-K:IOFF+ISIZ+1-K)
106     CONTINUE
        IOFF = IOFF + ISIZ
        FLAGS(J) = I2TMP
107   CONTINUE
      PRINT '("FLAGS  =",10(X,I5),/,(8X,10(X,I5)))', FLAGS

      RETURN
      END
```

```
C-----------------------------------------------------------------------
C ^SUBROUTINE: SWPBYT
C
C     This subroutine will swap the bytes of a data element
C-----------------------------------------------------------------------

      SUBROUTINE SWPBYT(DATBUF, NBYTES, SWPBUF)

      CHARACTER          DATBUF*4       ! Input data buffer
      INTEGER*4          NBYTES         ! Number of bytes in data element
      CHARACTER          SWPBUF*4       ! Output swapped buffer

      PRINT '(A4,X,A4,X,I5)', DATBUF, SWPBUF, NBYTES
      DO 10 K=1,ISIZ
        SWPBUF(K:K) = DATBUF(ISIZ+1-K:ISIZ+1-K)
   10 CONTINUE

      RETURN
      END


C-----------------------------------------------------------------------
C ^SUBROUTINE: IBMFLT
C
C     This subroutine will convert value to an IBM float
C-----------------------------------------------------------------------

      SUBROUTINE IBMFLT(IWORD, R4FLT)

      INTEGER*4      IWORD              ! input word to convert to IBM float
      INTEGER*4      IDROW              ! reverse the bits of input word
      REAL*8         A /16.0/           ! base number
      INTEGER*4      B /64/             ! exponent offset
      REAL*8         C /0.0/            ! fraction offset
      INTEGER*1      S                  ! sign flag
      INTEGER*2      E                  ! binary exponent
      REAL*8         F                  ! binary fraction
      REAL*8         M                  ! mantissa
      REAL*8         V                  ! float value
      INTEGER*4      I                  ! counter
      REAL*4         R4FLT              ! returned 4-byte float value


      S = ISHFT(IWORD, -31)

      E = 0
      DO 10 I=0,6
        E = E + IAND(ISHFT(IWORD,-24),ISHFT(1,I))
   10 END DO

      IDROW = 0
      DO 11 I=0,31
        IF (IAND(IWORD,ISHFT(1,I)) .NE. 0) THEN
          IDROW = IOR(IDROW,(ISHFT(1,31-I)))
        END IF
   11 END DO
```

```
      F = 0.0
      DO 12 I=0,31
        IF (ISHFT(IAND(ISHFT(IDROW,-8),ISHFT(1,I)),1) .NE. 0) THEN
          F = F + 1 / FLOAT(ISHFT(IAND(ISHFT(IDROW,-8),ISHFT(1,I)),1))
        END IF
   12 END DO

      M = C + F                          ! calculate the mantissa
      V = (-1)**S * M * A**(E - B)       ! calculate the float value
      IF (ABS(V) .LT. 2.0**(-149)) THEN
        V = (-1)**S * 0.0                ! avoid underflow
      END IF

      R4FLT = V
      RETURN
      END
```

5. Data Services

5.1 Reverb

The GES DISC provides basic temporal and advanced (event) searches through the EOSDIS Reverb data search and download interface:

http://reverb.echo.nasa.gov

Reverb allows users the ability to search on keywords, spatial region, and time period on datasets archived and various data centers. It offers various download options that suit users with different preferences and different levels of technical skills. To search for the SCAMS data enter GES_DISC_SCAMSN6L2_V001 into the keyword field.

5.2 FTP

The Nimbus data products are available for users to download directly using anonymous FTP:

ftp://acdisc.gsfc.nasa.gov/data/s4pa/Nimbus6_SCAMS_Level2/SCAMSN6L2.001/

The data are organized in directories by year with subdirectories by day of year. README and other documentation are located under the doc directory.

6. More Information

6.1 Web Resources

For other Nimbus data products, please see the GES DISC's Nimbus heritage data web page at:
http://disc.gsfc.nasa.gov/nimbus/

To search for other related data, please visit NASA's Global Change Master Directory at:
http://gcmd.nasa.gov.

6.2 Point of Contact

Name: GES DISC Help Desk
URL: http://disc.gsfc.nasa.gov/
E-mail: gsfc-help-disc@lists.nasa.gov
Phone: 301-614-5224
Fax: 301-614-5228
Address: Goddard Earth Sciences Data and Information Services Center
Attn: Help Desk
Code 610.2
NASA Goddard Space Flight Center
Greenbelt, MD 20771, USA

6.3 References

D. H. Staelin, et al, "The Nimbus-6 User's Guide - Section 4: The Scanning Microwave Spectrometer (SCAMS) Experiment", NASA Goddard Space Flight Center, Feb. 1975, Pages 59-86

7. Acknowledgements

The Nimbus data recovery task at the GES DISC is funded by NASA's Earth Science Data and Information System program.

Acronyms

EOS: Earth Observing System

ESDIS: Earth Science and Data Information System

GES DISC: Goddard Earth Sciences Data and Information Services Center

GSFC: Goddard Space Flight Center

SCAMS: Scanning Microwave Spectrometer

L2: Level-2 Data

NASA: National Aeronautics and Space Administration

Reverb: ECHO's Next Generation Metadata and Service Discovery Tool

QA: Quality Assessment

UT: Universal Time