

MODIS

SCIENCE DATA SUPPORT TEAM

PRESENTATION

April 24, 1992

AGENDA

	<u>Page</u>
1. MODIS SDST Tracking List	1
2. MODIS Airborne Simulator (MAS)	6
3. Coding Recommendations	16
4. Software and Data Management Plan	22

DRAFT

MODIS SDST Tracking List

This MODIS SDST Tracking List is intended to keep track of individual items of work to be done. The Tracking List items are generally more specific, near-term, and detailed than the Deliverables listed in Appendix A, and they supplement the Action Items carried in the weekly SDST Presentation Handouts. Often there will be overlap between Action Items, Deliverables, and the Tracking List items.

911115.01 Develop a Glossary/Data Dictionary for MODIS starting from PDS. (Define lat, lon, chlorophyll, pixel, coordinate systems, etc.)

911115.02 Plan for transferring MAS data to the Version 0 DAAC.

911122 At Level-1B, keep geolocation based upon position and attitude. Append anchor points, DEM anchor points, and any other geolocation information we generate.

911206 Standardize on the use of Cadre, MicroSoft Project, MicroSoft Word, Excel, Windows 3.1 and Postscript.

920103.01 See what Frank McGarry, Code 530, has in the way of software engineering tools that we could experiment with.

920103.02 Understand and apply Total Quality Management (TQM).

920103.03 Adopt a file naming convention, state what it is, and use it consistently.

920110.01 Handout has preliminary draft of TLCF Plan. Use this as starting point for draft due in June.

920124.01 Assure that the Budget, FY 1992 Work Plan, schedule, and the Project Plan are all consistent.

920131.01 Come up with a set of Organizational Ethics.

920221.01 Find out if anyone is planning to deliver code from IBM computers. (Mike King's cloud code is one case.) (J. P. Muller has code in C++.)

920226.01 Emphasize Level-2 Processing shell design activity (in C).

DRAFT

- 920228.01 Develop an approach to changing Data Product Prioritization.
- ~~920228.02 Provide comments on Jim Ormsby's Glossary of Definitions.~~
- 920228.03 Check with John Barker on the calibration algorithms after the SBRC PDR (Oct 1992). (Who from the SDST will go to this PDR?)
- 920313.01 Review the Code 420 Software Management Plan.
- ~~920320.01 Get Data Product Attributes charts ready for TMs to review and update.~~
- 920320.02 Prepare a list of SDST deliverables.
- 920320.03 Develop Milestones for generating an MAS proto-processing system.
- ~~920320.04 Adjust the SDST schedule to be compatible with Bredeson's schedule.~~
- 920320.05 Get Frank McGarry to come and talk to the SDST about tools they use.
- 920324.01 Develop a plan for generating a work plan going down to the two-week level.
- 920324.02 Identify risks in porting code.
- ~~920327.01 Include algorithm descriptions in packets for TMs.~~
- 920327.02 Ted Meyer and Stan Scott in Code 423 will take over Rich Bredeson's duties.
- 920327.03 Develop an MAS schedule covering 2 or 3 years. Estimate manpower and computer resource requirements for MAS processing. Identify ancillary data.
- 920403.04 Determine launch date for MODIS PM. Work up schedule for PM algorithms, if there are any which are different from AM. (Also consider AM-PM algorithms which will require data from both MODIS AM and PM.)
- 920403.01 Consider possible ways of generating test data sets which will thoroughly test algorithms. (Packet simulator, scan cube simulator, science data sets, etc.)

DRAFT

- 920403.02 Break the SDST schedule down to the next level of detail.
- 920403.03 Put together a list of tools that are available (low priority).
- 920403.04 Have a configuration management system in place when we have code to control.
- 920403.05 Find out if anyone has netCDF working on the Mac (for support of Mike King).
- 920403.06 Get SLIP working on an LTP IRIS or a SUN (Ed Masuoka).
- 920403.07 Install FORTRAN-LINT on an LTP SGI computer with X11 rev 4 windowing (Shahin Samadi).
- 920403.08 Coordinate all work through Will Webster and/or Al Fleig.
- 920403.09 Find out if we can pass by reference rather than by value (SGI Fortran).
- 920403.10 Let TMs and others know how to get in-situ data for FIRE and other MAS experiments.
- 920407.01 ~~In the coding recommendations, include a statement that we are looking into QA Fortran, Fortran Lint, etc., and we suggest that TMs use these in the development of their code.~~
- 920407.02 When we get code, compare load map list of variables, called functions, etc. with internal documentation.
- 920410.01 Lloyd Carpenter talk with Al Fleig about the list of risks.
- 920410.02 Adopt a set of documentation procedures (including identification, version, date, author, etc.) for use on all SDST documents (guidelines, schedules, plans, etc.).

DRAFT

Appendix A: MODIS SDST Deliverables

Deliverable		Due Date
1	EOS Deliverables	
1.1	MODIS Software and Data Management Plan	Jun 1992
1.2	MODIS Team Leader Computing Facility Plan	Jun 1992
2	MODIS SDST Deliverables	
2.1	Schedule	Apr 1992
2.2	Project Plan	?
2.3	Executive Information Summary	?
2.4	Instantaneous Field of View (IFOV) Plots	?
3	MODIS Airborne Simulator (MAS)	
3.1	Version 1 Software	(V1.0) Nov 1991
3.2	Data Users Guide	(V1.0) Jan 1992
3.3	Data Catalog	Jun 1992
3.4	Version 1 Metadata	Jun 1992
3.5	Level-1 Processed Data	Jan 1992
4	MODIS Level-1 System	
4.1	Level-1 Software Development Plan	?
4.2	B Version, Level-1 System	
4.2.1	Level-1A PDR Report	Jan 1993
4.2.2	Level-1B PDR Report	Apr 1993
4.2.3	Level-1A & -1B CDR Report	Jan 1994
4.2.4	Level-1A & -1B Test Plan	Oct 1994
4.2.5	Level-1A & -1B Users Guide	?
4.2.6	Level-1A & -1B System Description	?
4.2.7	Level-1A & -1B System Delivery	Jul 1995
4.3	V1, Level-1 System	
4.3.1	Level-1A & -1B CDR Report	Jan 1995
4.3.2	Level-1A & -1B Test Plan	Oct 1995
4.3.3	Level-1A & -1B Users Guide	?
4.3.4	Level-1A & -1B System Description	?
4.3.5	Level-1A & -1B System Delivery	Jul 1996
4.4	V1, Level-1 System	
4.4.1	Level-1A & -1B CDR Report	Jan 1996
4.4.2	Level-1A & -1B Test Plan	Oct 1996
4.4.3	Level-1A & -1B Users Guide	?
4.4.4	Level-1A & -1B System Description	?
4.4.5	Level-1A & -1B System Delivery	Jul 1997

DRAFT

5	MODIS Level-2/3 System	
5.1	Coding Guidelines for MODIS TM Science Algorithms	
5.1.1	First Draft	Apr 1992
5.1.2	Second Draft	Jan 1993
5.1.3	B	Jul 1993
5.1.4	V1	Jul 1994
5.1.5	V2	Jul 1995
5.2	MODIS Level-2 Processing Shell	
5.2.1	SRR	Apr 1993
5.2.2	PDR	Oct 1993
5.2.3	CDR	Apr 1994
5.2.4	B	Jan 1995
5.2.5	V1	Jan 1996
5.2.6	V2	Jan 1997
5.3	TM Algorithm Integration Plan	?
5.4	TM Algorithm Test Plan	?
5.5	SDST System Delivery to PGS	
5.5.1	Beta	Jul 1995
5.5.2	V1	Jul 1996
5.5.3	V2	Jul 1997
5.5.4	First Post Launch	Oct 1998

MODIS Airborne Simulator status (Liam Gumley)

Progress up to 23 April 1992

(1) MAS data usage by Dorothy Hall's group

Dorothy Hall called me early this week and mentioned a problem her group seemed to be having with the MAS Level-1 data. Apparently the image data from channel 2 (31-OCT-91) was saturating at a value of 255 over snow covered regions. Also they could not see any data in channels 7 through 12. They asked if I could help identify the problem.

Janet Chen from Dorothy's group reloaded the first flight track from the 31-OCT-91 MAS flight and I copied it back onto LTPIRIS2. After examination of subset images, it appeared that data from all the channels was as expected. Although channel 2 was saturating, the value was about 21.6 radiance units (2160 as stored). Data was contained in channels 7 through 12 however in channels 7-9 the data was extremely noisy. The problem appeared to be that when the data was read into EASI/PACE, the 16 bit values were converted to 8 bit values. However there was no easy way to tell what the original 16 bit radiance values were.

For this reason I wrote a short program called *read-cdf* which allows the user to dump individual pixel radiance data from a specified channel and scanline within an MAS Level-1 file. This is a useful utility for any MAS data user so it has been uploaded to the MAS FTP site. Sample output from this program is shown overleaf, along with a code listing.

(2) Black body data averaging during Level-1 calibration processing

During the Science Team Meeting, Paul Menzel recommended that along track averaging of the MAS black body counts be done to lessen line-to-line calibration variation (i.e. striping) in the calibrated IR channels. Consequently some effort has been directed at characterizing the black body noise and investigating smoothing approaches.

Since all Level-0 blackbody data has been retained in the Level-1 datasets, there was no need to go back to the Level-0 data for testing smoothing approaches (although that may be desirable for operational smoothing). I wrote a short program called *read-bb* which allows the user to dump the black body data contained within a MAS Level-1 file. Sample output from this program is shown overleaf.

As a first step, I examined the black body data from the 05-DEC-91 FIRE flight. To verify the source of the noise in the black body counts, I plotted the black body counts and temperatures separately (shown overleaf). These clearly show that the black body count variation is caused by single sample noise in the scanner. The temperatures varied by less than 0.05C for the region examined.

Considering the small variation seen in the black body temperatures, it may be necessary to go to larger smoothing windows (e.g. up to 50 scan lines) rather than the 9 line smoothing shown. A common approach must be agreed upon and then used for all Level-1 processing, rather than correcting each individual image until it looks 'good enough'.

(3) MAS integration and testing at Ames

The latest information is that the MAS will return to Ames on May 15th for integration and testing, prior to the ASTEX deployment at the beginning of June. Mike King has requested that Tom Arnold and myself go out to Ames during this period to coincide with Chris Moeller,

who will help in the instrument check-out and advise Tom and myself on MAS Quick View System (QVS) procedures for use during ASTEX.

Sample output from *read-cdf*

```
Enter MAS Level-1B netCDF file name : 31oct91-01.cdf
Enter channel, record : 2,2100
Scanline HHMMSSSS  BB1T  BB2T Gain Slope      Intercept    Lat    Lon
  13001 19265600 -22.22 -2.99 0.50 9.190000E-02 0.000000E+00 38.89 -120.14
Enter pixel1, pixel2 : 200,300
  17.55  15.53  14.06  19.21  21.41  18.84  16.45  16.63  19.02  21.78
  19.39   9.65   5.24   4.69   5.05   7.44   5.97   8.36  11.30  11.49
  10.02   8.36   4.87   4.50   4.50   5.24   4.69   4.14   4.69   6.16
   5.97   4.87  11.86  13.14  10.75  14.24  13.33  19.57  22.33  19.76
  17.74  16.27  15.71  10.94  14.06  15.35  18.10  10.20   2.48   4.50
   8.55  18.29  14.98  12.41  11.30  19.39  20.31  17.92  19.21  19.57
  20.13  15.53  13.69  20.49  18.29  18.84  15.71  19.57  18.47  21.60
  21.78  21.60  21.78  21.60  21.60  21.60  21.60  21.60  21.60  21.60
  20.86  20.13  20.49  16.08  12.41   5.97  10.38  16.27   8.73  10.57
  17.74   9.47  15.53  23.07  21.41  21.60  21.60  20.49  19.02  21.23
   21.96
Enter pixel1, pixel2 : -1 -1
Enter channel, record : 7,2100
Scanline HHMMSSSS  BB1T  BB2T Gain Slope      Intercept    Lat    Lon
  13001 19265600 -22.22 -2.99 1.00 7.595765E-03 5.489122E-02 38.89 -120.14
Enter pixel1, pixel2 : 200,300
  0.49   0.40   0.39   0.39   0.39   0.38   0.28   0.34   0.28   0.26
  0.21   0.09   0.09   0.05   0.05   0.05   0.05   0.05   0.05   0.05
  0.12   0.05   0.05   0.05   0.16   0.26   0.36   0.45   0.44   0.51
  0.56   0.45   0.51   0.58   0.59   0.59   0.63   0.72   0.79   0.78
  0.75   0.73   0.76   0.75   0.78   0.65   0.62   0.67   0.69   0.50
  0.47   0.56   0.42   0.24   0.34   0.43   0.48   0.43   0.37   0.37
  0.22   0.28   0.31   0.21   0.25   0.25   0.31   0.31   0.33   0.35
  0.49   0.44   0.47   0.53   0.58   0.57   0.68   0.71   0.60   0.56
  0.55   0.64   0.69   0.78   0.76   0.80   0.77   0.84   0.78   0.82
  0.81   0.83   0.76   0.76   0.62   0.52   0.41   0.38   0.40   0.43
   0.34
Enter pixel1, pixel2 : -1 -1
Enter channel, record : 12,2100
Scanline HHMMSSSS  BB1T  BB2T Gain Slope      Intercept    Lat    Lon
  13001 19265600 -22.22 -2.99 1.00 1.454648E-01 2.890211E+01 38.89 -120.14
Enter pixel1, pixel2 : 200,300
  79.23  82.43  81.41  81.27  80.11  82.29  81.56  82.14  80.69  82.00
  78.21  73.71  76.32  75.01  71.96  73.71  74.72  78.51  80.83  79.81
  80.69  77.20  74.43  76.03  77.05  77.63  78.36  82.58  82.00  81.41
  88.83  84.76  82.72  80.69  80.40  83.16  85.78  84.62  83.45  83.16
  81.41  80.83  81.27  85.49  83.89  82.29  82.58  87.38  90.14  82.00
  84.76  83.89  82.87  81.71  83.45  85.49  86.80  84.18  86.07  84.32
  84.47  86.36  88.25  85.92  84.76  87.67  87.09  89.27  86.22  84.91
  86.80  84.76  83.60  83.02  83.74  84.76  87.23  83.16  82.58  82.58
  83.02  83.16  84.91  86.22  86.36  88.25  85.34  85.05  85.34  85.49
  86.36  85.49  84.62  83.31  83.60  82.43  82.29  82.00  85.63  85.05
   83.74
```

```

#####
c PROGRAM READ-CDF
c
c Purpose: To read selected contents of a MAS Level-1B
c netCDF flight track file, and allow the user to
c view calibrated radiances for individual pixels.
c
c Revised: 21-APR-1992
c
c Author: Liam Gumley, RDC
c
c Language: FORTRAN-77
c
c Updates:
c
c Notes:
c This code has been compiled and executed successfully on
c Silicon Graphics Iris (Irix 3.3.2) and DEC VAX (VMS 5.3)
c platforms. The netCDF library used was version 2.02.
c
c To compile on the Iris:
c
c % f77 -vms_cc -o read-cdf read-cdf.f libnetcdf.a libsun.a
c assuming the files netcdf.inc, libnetcdf.a, libsun.a are
c in the current directory.
c
c To compile on the VAX:
c
c $ fortran read-cdf.for
c $ link mesdump,netcdf/lib
c
c assuming the files netcdf.inc, netcdf.olb are in the current
c directory.
c
c-----  

c include 'netcdf.inc'
c
c Integer cdfid, ndims, nvars, natts, recdim, rcode, dimsiz,
c & mindex( 2 )
c
c Integer headid, anchid, scanid, rateid, timeid, dateid, bb1tid,
c & bb2tid, gainid, slopid, intrid, platid, plonid, senzid, senaid,
c & solzid, solaid, alatid,alonid, ahedid, aaltid, dataid, framid,
c & thumid, bendid, rollid, bb1cid, bb2cid, apitid
c
c Character filnam*72, dimnam*72
c
c Integer*2 rate1, bb1t1, bb2t1, gain1
c Integer*4 scan1, time1, date1
c Real*4 alat1, alon1, slop1, intr1
c
c Integer record, channel, start( 3 ), count( 3 )
c
c Integer*2 dataline( 716 ), pixel1, pixel2, pixel
c
c-----  

c set error options (verbose messages, exit on error)
c call ncopt( ncverbos + ncfatal )
c open netcdf file

```

```

10      write( *, 10 )
format( *, 'Enter MAS Level-1B netCDF file name : ' $ )
15      read( *, 15 ) filnam
format( a72 )

cdfid = ncopen( filnam, ncnowrit, rcode )

c-----  

c inquire about contents of netcdf file
call ncinq( cdfid, ndims, nvars, natts, recdim, rcode )

c get id's for desired data items
headid = ncvid( cdfid, 'DataSetHeader', rcode )
anchid = ncvid( cdfid, 'AnchorPtIndex', rcode )
framid = ncvid( cdfid, 'DataFrameStatus', rcode )
scanid = ncvid( cdfid, 'ScanLineCounter', rcode )
thumid = ncvid( cdfid, 'ThumbwheelSwitches', rcode )
rateid = ncvid( cdfid, 'ScanRate', rcode )
timeid = ncvid( cdfid, 'GMTTime', rcode )
bendid = ncvid( cdfid, 'S-BendIndicator', rcode )
rollid = ncvid( cdfid, 'AircraftRollCount', rcode )
dateid = ncvid( cdfid, 'Year&DayOfYear', rcode )
bb1tid = ncvid( cdfid, 'Blk8dy1Temperature', rcode )
bb2tid = ncvid( cdfid, 'Blk8dy2Temperature', rcode )
gainid = ncvid( cdfid, 'AmplifierGain', rcode )
bb1cid = ncvid( cdfid, 'Blk8dy1Counts', rcode )
bb2cid = ncvid( cdfid, 'Blk8dy2Counts', rcode )
slopid = ncvid( cdfid, 'Calibrationslope', rcode )
intrid = ncvid( cdfid, 'CalibrationIntercept', rcode )
platid = ncvid( cdfid, 'PixelLatitude', rcode )
plonid = ncvid( cdfid, 'PixelLongitude', rcode )
senzid = ncvid( cdfid, 'SensorZenithAngle', rcode )
senaid = ncvid( cdfid, 'SensorAzimuthAngle', rcode )
solzid = ncvid( cdfid, 'SolarZenithAngle', rcode )
solaid = ncvid( cdfid, 'SolarAzimuthAngle', rcode )
alatid = ncvid( cdfid, 'AircraftLatitude', rcode )
alonid = ncvid( cdfid, 'AircraftLongitude', rcode )
ahedid = ncvid( cdfid, 'AircraftHeading', rcode )
apitid = ncvid( cdfid, 'AircraftPitch', rcode )
dataid = ncvid( cdfid, 'CalibratedData', rcode )

c-----  

c reset error option since we may have an old file with
c mis-spelling 'AircraftAltitude'
call ncopt( 0 )

aaltid = ncvid( cdfid, 'AircraftAltitude', rcode )
if( rcode .ne. 0 ) then
  aaltid = ncvid( cdfid, 'AircraftAltitude', rcode )
endif

call ncopt( ncverbos + ncfatal )

c-----  

c get size of unlimited dimension (time is dimension 1)
call ncinq( cdfid, 1, dimnam, dimsiz, rcode )
if( dimsiz .lt. 1 ) then
  stop 'Error: 0 data records in this file'
endif

```

```

c-----  

c      get data items for desired record and channel  

20    continue  

        write( *, '( " Enter channel, record : "' $ "' )'  

        read( *, * ) channel, record  

c      scan line number  

        call ncvgt1( cdfid, scanid, record, scan1, rcode )  

c      time (HHMMSSS)  

        call ncvgt1( cdfid, tim eid, record, time1, rcode )  

c      black body 1 and 2 temperatures for desired channel  

c      ( degrees C x 100 )  

        mindex( 1 ) = channel  

        mindex( 2 ) = record  

        call ncvgt1( cdfid, bb1tid, mindex, bb1t1, rcode )  

        call ncvgt1( cdfid, bb2tid, mindex, bb2t1, rcode )  

c      gain for desired channel ( x 1000 )  

        call ncvgt1( cdfid, gainid, mindex, gain1, rcode )  

c      calibration slope for desired channel  

c      ( same units as calibrated radiance )  

        call ncvgt1( cdfid, slopid, mindex, slop1, rcode )  

c      calibration intercept for desired channel  

c      ( same units as calibrated radiance )  

        call ncvgt1( cdfid, intrid, mindex, intr1, rcode )  

c      nadir longitude (degrees, East positive, Greenwich=0)  

        call ncvgt1( cdfid, alonid, record, alon1, rcode )  

c      nadir latitude (degrees, North positive)  

        call ncvgt1( cdfid, alatid, record, alat1, rcode )  

c      calibrated data for this channel ( x 100, nearest integer )  

c      (milliWatts / square centimeter / steradian / micron (VIS/NIR) )  

c      (milliWatts / square centimeter / steradian / wavenumber (IR) )  

        start( 1 ) = 1  

        start( 2 ) = channel  

        start( 3 ) = record  

        count( 1 ) = 716  

        count( 2 ) = 1  

        count( 3 ) = 1  

        call ncvgt( cdfid, dataid, start, count, dataline, rcode )  

c-----  

c      write formatted output  

        write( *, '( " Scanline HHMMSSS  BB1T  BB2T Gain",  

&   " Slope      Intercept   Lat   Lon"' )'  


```

READ-CDF.F 4-23-92 11:44a

```

&   1x, 1pe13.6, 1x, 1pe13.6, 0pf7.2, 1x, f7.2 )' )  

&   scan1, tim e1, 0.01 * real( bb1t1 ), 0.01 * real( bb2t1 ),  

&   0.001 * real( gain1 ), slop1, intr1, alat1, alon1  

40    continue  

        write( *, '( " Enter pixel1, pixel2 : "' $ "' )'  

        read( *, * ) pixel1, pixel2  

c      check if we should go back and read another record  

        if( pixel1 .lt. 0 .or. pixel2 .lt. 0 ) go to 20  

        write( *, '( 10( 1x, 16 ) )' )  

&   ( dataline( pixel ), pixel = pixel1, pixel2 )  

        go to 40  

c-----  

        end  

c#####

```

Sample output from read-bb

Enter MAS Level-1B netCDF file name : 31oct91-01.cdf

Enter channel number to unpack : 2

Enter start record, end record, step : 2100,2110,1

Record	Scanline	HHMMSSSS	BB1T	BB2T	BB1C	BB2C	Slope	Intercept
2100	13001	19265600	-22.22	-2.99	5	4	9.190000E-02	0.000000E+00
2101	13002	19265600	-22.22	-2.99	5	6	9.190000E-02	0.000000E+00
2102	13003	19265600	-22.22	-2.99	5	5	9.190000E-02	0.000000E+00
2103	13004	19265600	-22.22	-2.99	5	5	9.190000E-02	0.000000E+00
2104	13005	19265600	-22.22	-2.99	5	4	9.190000E-02	0.000000E+00
2105	13006	19265700	-22.22	-2.99	5	6	9.190000E-02	0.000000E+00
2106	13007	19265700	-22.22	-2.99	5	5	9.190000E-02	0.000000E+00
2107	13008	19265700	-22.22	-3.00	5	4	9.190000E-02	0.000000E+00
2108	13009	19265700	-22.22	-3.00	5	7	9.190000E-02	0.000000E+00
2109	13010	19265700	-22.22	-2.99	5	6	9.190000E-02	0.000000E+00
2110	13011	19265700	-22.22	-2.99	5	4	9.190000E-02	0.000000E+00

Enter channel number to unpack : 7

Enter start record, end record, step : 2100,2110,1

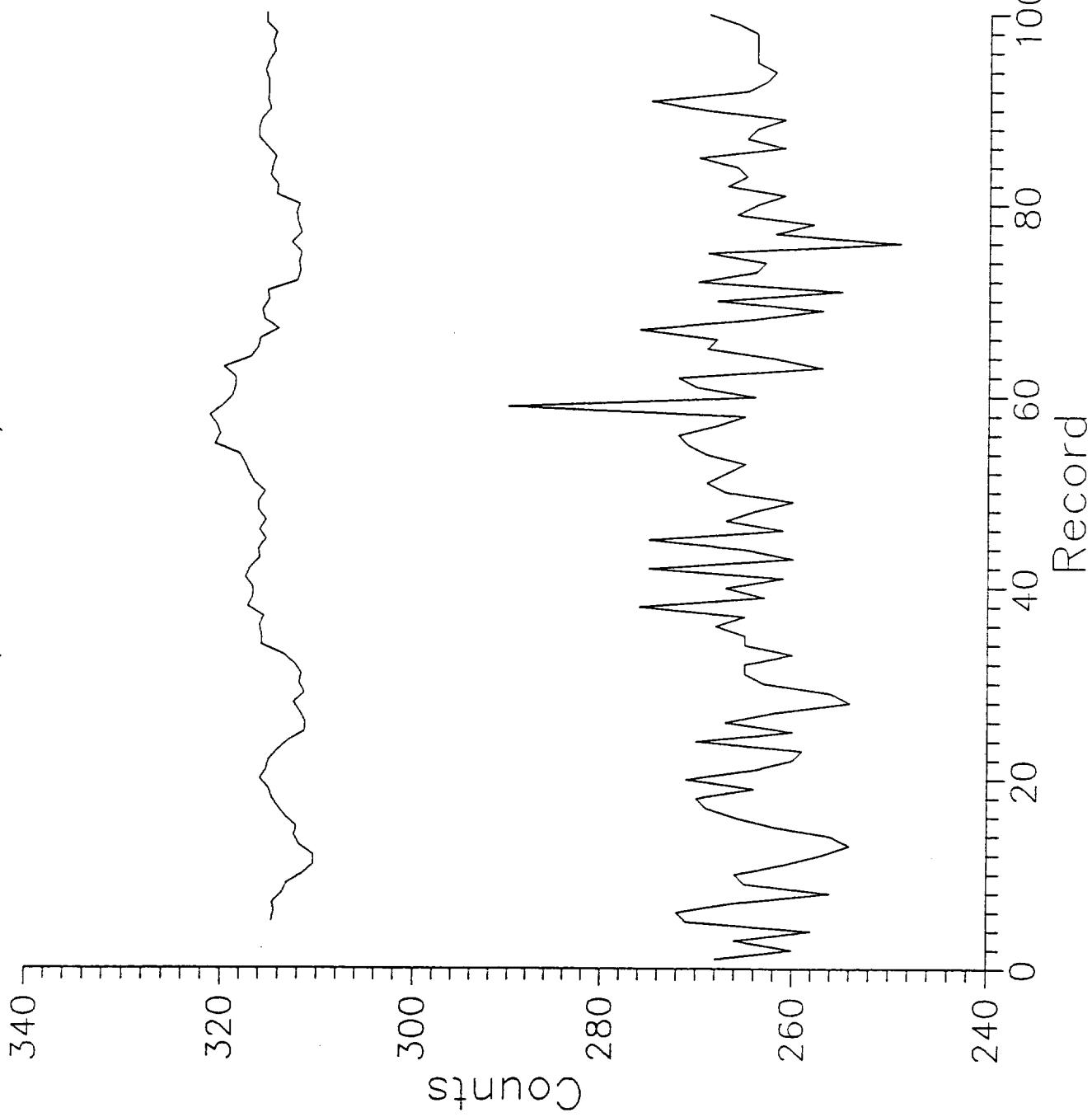
Record	Scanline	HHMMSSSS	BB1T	BB2T	BB1C	BB2C	Slope	Intercept
2100	13001	19265600	-22.22	-2.99	0	14	7.595765E-03	5.489122E-02
2101	13002	19265600	-22.22	-2.99	7	10	3.544690E-02	-1.932371E-01
2102	13003	19265600	-22.22	-2.99	17	15	0.000000E+00	0.000000E+00
2103	13004	19265600	-22.22	-2.99	1	19	5.907817E-03	4.898340E-02
2104	13005	19265600	-22.22	-2.99	3	11	1.329259E-02	1.501346E-02
2105	13006	19265700	-22.22	-2.99	2	10	1.329259E-02	2.830604E-02
2106	13007	19265700	-22.22	-2.99	0	17	6.255336E-03	5.489122E-02
2107	13008	19265700	-22.22	-3.00	0	22	4.829850E-03	5.489122E-02
2108	13009	19265700	-22.22	-3.00	0	20	5.312835E-03	5.489122E-02
2109	13010	19265700	-22.22	-2.99	1	23	4.833668E-03	5.005755E-02
2110	13011	19265700	-22.22	-2.99	1	42	2.593676E-03	5.229754E-02

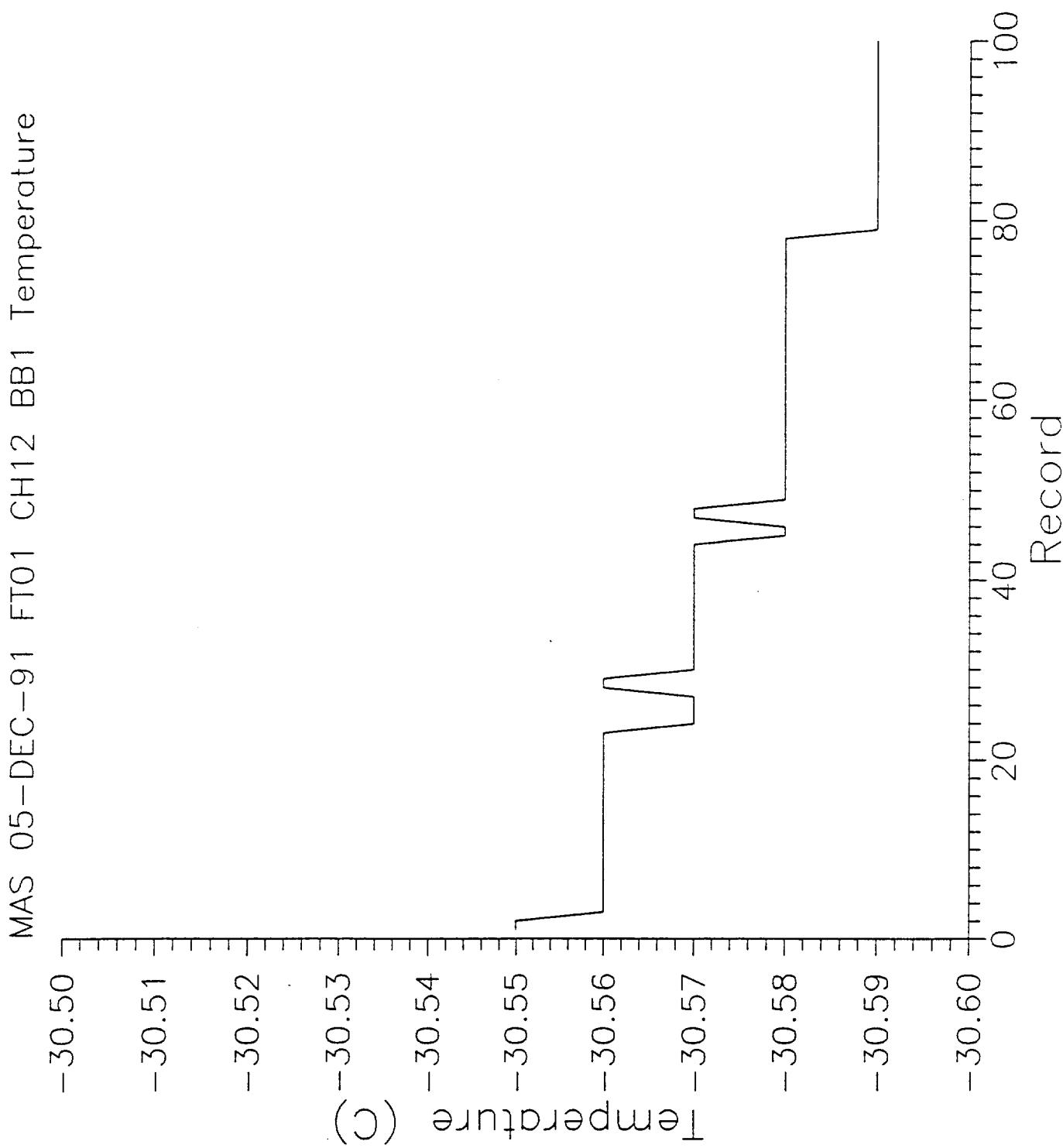
Enter channel number to unpack : 12

Enter start record, end record, step : 2100,2110,1

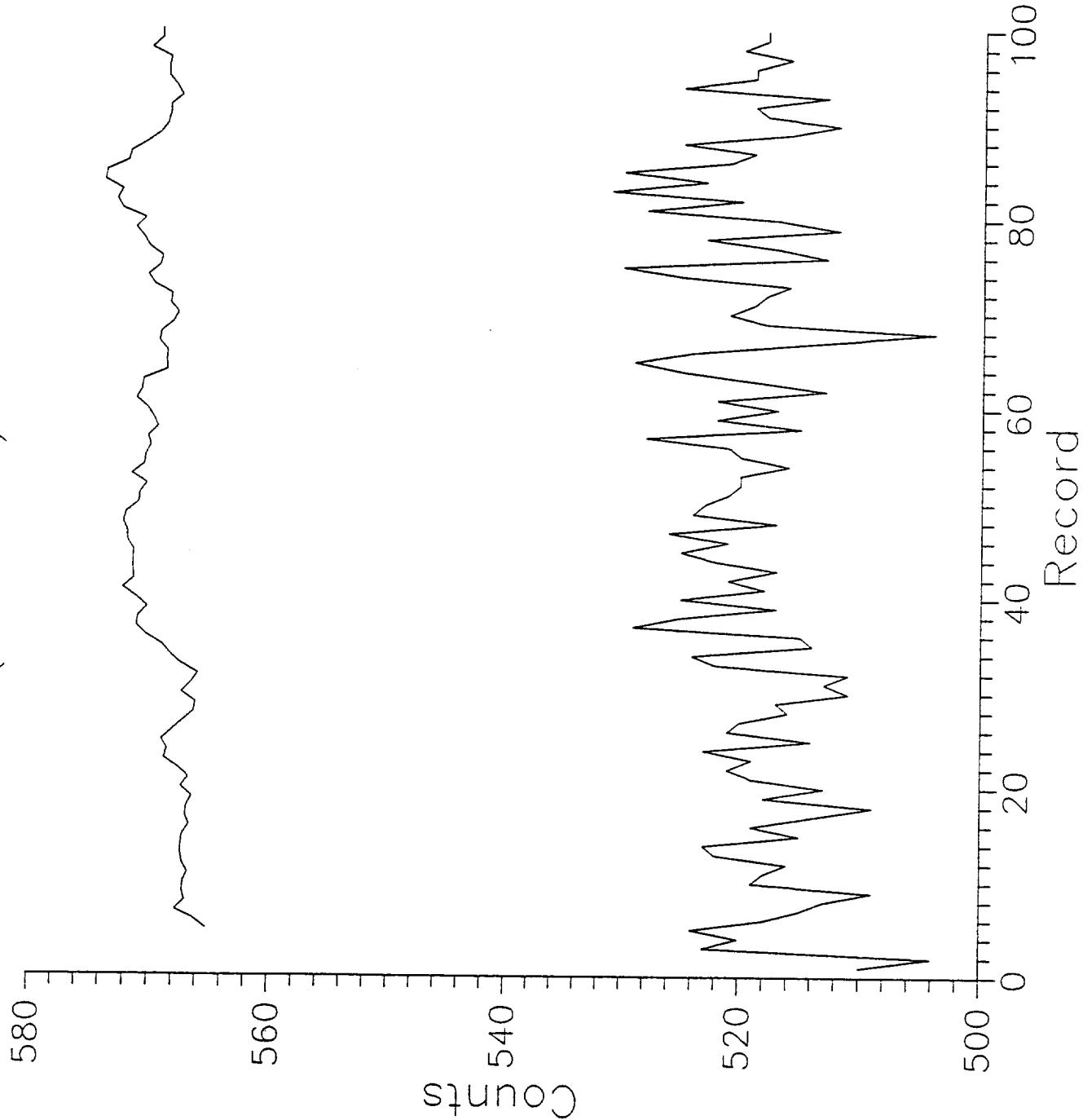
Record	Scanline	HHMMSSSS	BB1T	BB2T	BB1C	BB2C	Slope	Intercept
2100	13001	19265600	-22.22	-2.99	200	364	1.454648E-01	2.890211E+01
2101	13002	19265600	-22.22	-2.99	194	365	1.395101E-01	3.093010E+01
2102	13003	19265600	-22.22	-2.99	203	359	1.529245E-01	2.695137E+01
2103	13004	19265600	-22.22	-2.99	201	360	1.500392E-01	2.783719E+01
2104	13005	19265600	-22.22	-2.99	204	362	1.509888E-01	2.719335E+01
2105	13006	19265700	-22.22	-2.99	198	366	1.420013E-01	2.987880E+01
2106	13007	19265700	-22.22	-2.99	195	371	1.355467E-01	3.156344E+01
2107	13008	19265700	-22.22	-3.00	193	364	1.394302E-01	3.108504E+01
2108	13009	19265700	-22.22	-3.00	184	362	1.339470E-01	3.334882E+01
2109	13010	19265700	-22.22	-2.99	206	361	1.539111E-01	2.628937E+01
2110	13011	19265700	-22.22	-2.99	197	363	1.437122E-01	2.968375E+01

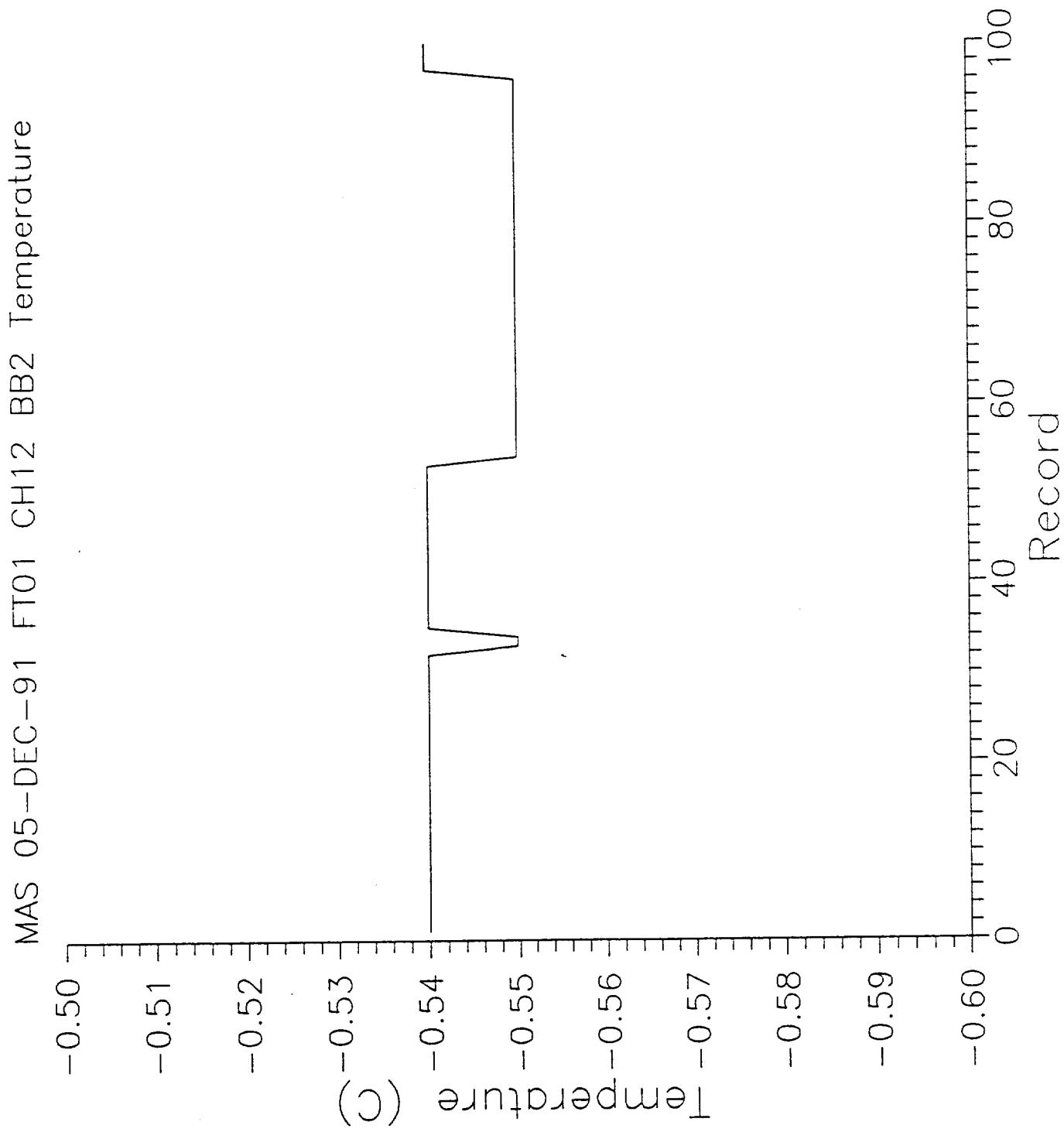
MAS 05-DEC-91 FT01 C 12 BB1 Counts
Original count data and 9 point running average
smoothed data (+50 counts)





MAS 05-DEC-91 FT^r : CH12 BB2 Counts
Original count data and 9 point running average
smoothed data (+50 counts)





**MODIS Science Data Support Team (SDST)
Coding Recommendations for the MODIS Science Team
(DRAFT)**

J. J. Pan
Research and Data Systems Corporation
(301) 982-3738

In this draft, several recommendations are listed for structured coding in FORTRAN since many of the team members will use FORTRAN to develop their programs. A good structured design in coding will make the code easy to read, maintain, and test.

A special effort must be made to avoid, in advance, any potential problems which could arise in individual programs when performing system integration. The following paragraphs provide some examples of "poor" and "good" coding.

1. Data Evaluation and File Manipulation

Basically there are three components in a program (Fig. 1),

- (1) Input -- Read data,
- (2) Algorithm -- Compute solution, and
- (3) Output -- Write solution.

In the input procedure, a flag is recommended to indicate whether the input data is valid or not. The flag might indicate

- . data type -- e.g., integer, real, etc.
- . data size -- e.g., data volume or array size, and
- . data range -- e.g., the minimum and maximum values of valid data.

Example:

```
Read(5,* , Error=99) (Data(K) , K=1,Length)
      .....
99 Write(6,20)
20 Format(" Error in Input Data ...")
      .....
```

If the input data is invalid, an appropriate action, such as to print an error message and stop execution, will be taken.

The output data from the algorithm computation will be formatted appropriately, particularly if a subsequent processing or display is necessary. This step is heavily dependent on the data flow in the whole system.

Complete the OPEN and CLOSE files statements in a program internally. Also, specify the input and output file names, rather than use the default file name (e.g., FOR008.DAT). For example, an OPEN statement using a VAX system might be

```
No = 8
Input = 'Image.dat'
Open(Unit=No, File=Input, Status='Old', Access='Direct',
$      Recl=256)
```

2. Structured Coding

The following three basic coding structures will be sufficient for most scientific programs (Fig .2)

- (1) Sequential,
- (2) Selection, and
- (3) Iteration.

There is a single entry and a single exit in each structure. This will make the code easier to test. Decompose a lengthy subroutine into several smaller subroutines for easy maintenance.

3. IF-THEN Selection Structure

Here are several recommendations in IF-THEN structure.

- (1) Avoid IF-THEN-IF structure, use IF-THEN-ELSE-IF structure.

Poor	Good
<pre>IF (M .GT. N) THEN IF (I .GT. J) THEN M = I ELSE M = J ENDIF ELSE M = N ENDIF</pre>	<pre>IF (M .LT. N) THEN M = N ELSE IF (I. GT. J) THEN M = J ELSE M = I ENDIF</pre>

(2) Avoid null THEN statement.

Poor

```
IF (N .LE. 0) THEN  
    ELSE N = N-1  
ENDIF
```

Good

```
IF (N .GT. 0) N = N-1
```

(3) Avoid arithmetic IF, use logical IF.

Poor

```
IF (N) 10, 20, 30  
10 N = N + 3  
GO TO 99  
20 N = N + 2  
GO TO 99  
30 N = N + 1  
99 CONTINUE
```

Good

```
IF (N .LT. 0) THEN  
    N = N + 3  
ELSE IF (N .EQ. 0) THEN  
    N = N + 2  
ELSE  
    N = N + 1  
ENDIF
```

(4) Avoid GOTO statement.

(5) Avoid the comparison of two real numbers in an IF statement.

Poor

```
IF (A .EQ. B) THEN  
    .....
```

Good

```
EPS=0.00001  
IF (ABS(A-B) .LE. EPS) THEN
```

4. Code Readability

Make the code easy to read and understand. For example, to create a unit matrix A,

Poor

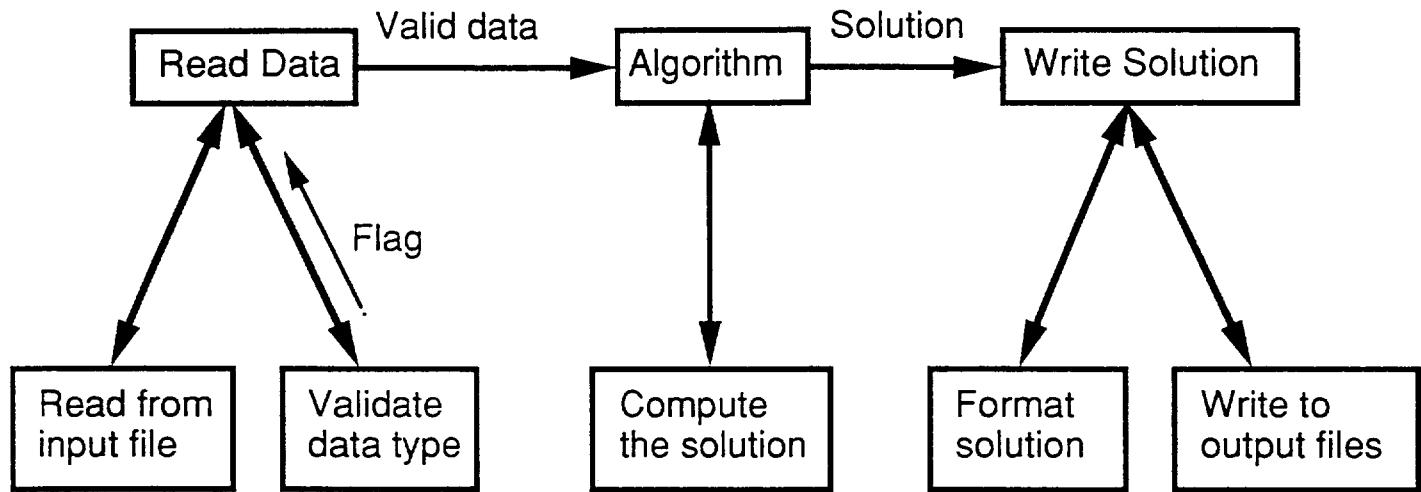
```
Do 25 I = 1, N  
Do 25 J = 1, N  
25 A(I,J) = (I/J) * (J/I)
```

Good

```
Do 25 I = 1, N  
Do 15 J = 1, N  
15 A(I,J) = 0.0  
25 A(I,I) = 1.0
```

5. Code Checking.

- (1) Initialize all variables and arrays at the appropriate locations.
- (2) Check the computational accuracy carefully when double precision is required for some variables.
- (3) Check the values of variables to avoid underflow and overflow problems.

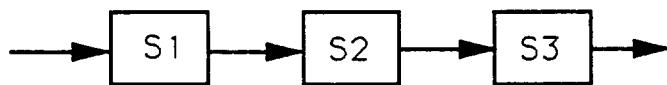


Flag is used to control invalid input data

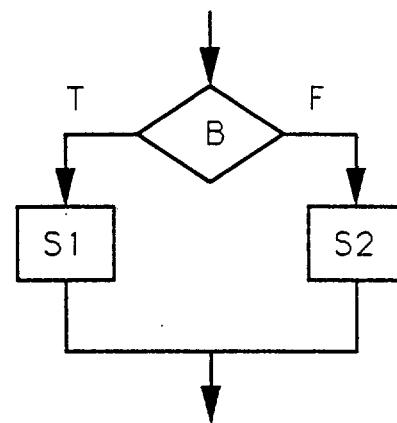
Figure 1. Data Flow Diagram In Structured Design

-- Single entry, Single exit --

1. Sequential: S1; S2; S3



2. Selection: IF B THEN S1
ELSE S2
END IF



3. Iteration: WHILE B DO S

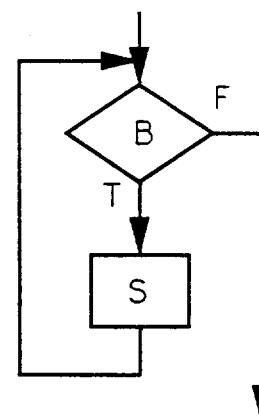


Figure 2. A sufficient set of structured coding

PRELIMINARY OUTLINE AND DRAFT

MODIS Science Data Support Team

Software and Data Management Plan

PRELIMINARY OUTLINE AND DRAFT

Table of Contents

1	Introduction	1
1.1	Purpose	1
1.2	Scope	1
2	MODIS Mission Overview	2
2.1	Mission Objectives	2
2.2	Scientific Objectives	2
3	MODIS Instrument Overview	2
3.1	Experimental Objectives	2
3.2	Instrumentation	2
3.3	Purpose	2
3.4	Operation/Data Modes	2
4	MODIS End-to-End Data Flow	2
4.1	Space Segment	2
4.2	Ground Segment	2
4.3	Science User Segment	2
5	MODIS Software and Data Policy Overview	3
6	MODIS Science Software	3
6.1	Level-1 Software	3
6.1.1	Level-1A Design	3
6.1.2	Level-1B Design	3
6.1.3	Code Development	3
6.1.4	Test Plan & Test Data Development	3
6.1.5	Test and Delivery to ECS	3
6.2	Level-2 Software	3
6.2.1	Coding Recommendations	3
6.2.2	Level-2 Processing Shell	3
6.2.3	Review of Team Member Algorithms	3
6.2.4	TM Code Development	3
6.2.5	Integration of Team Member Code	3
6.2.6	Test and Delivery	3
6.2.7	End-to-End Test	3
6.2.8	Post Launch Activities	3
6.3	Utilities	3
6.4	Configuration Management	3
7	MODIS Data Processing	3
7.1	Data Descriptions	3
7.2	Data Processing and Analysis	3

PRELIMINARY OUTLINE AND DRAFT

7.3	Data Calibration and Validation	3
7.4	Data Products	3
7.5	Data Formats, Rate and Volume	3
8	MODIS Data Storage and Archiving	3
8.1	Information Archive	3
8.1.1	Directories and Catalogs	3
8.1.2	Metadata	3
8.1.3	Browse Data	3
8.1.4	Software and Documentation	3
8.2	Data Archive	3
8.3	Data Security	3
9	MODIS Data Access and Distribution	3
10	MODIS Data Discard Policy	3

PRELIMINARY OUTLINE AND DRAFT

1 Introduction

1.1 Purpose

The purpose of this MODIS Software and Data Management Plan is to address the policy of the MODIS Science Team for management of the MODIS science data and the software which will be developed for the purpose of processing these data and generating data products on an operational basis. This software and data management policy will be coordinated with the EOS Project to assure consistency with the requirements of the EOSDIS. A complete life-cycle approach to MODIS software and data management is essential, considering the relative time scales of the EOS mission and technology advancement.

1.2 Scope

This MODIS Software and Data Management Plan will define the nature of the software to be developed for processing MODIS science data, and it will outline the procedures to be followed in software development, porting, integration, testing, validation, maintenance, modification, and documentation. A configuration management system plan will be included for the protection and assurance of the MODIS operational software and data products.

The plan will also describe the manner in which the MODIS instrument data will be acquired, calibrated, validated, earth located, processed, archived and distributed to the science users, within the EOSDIS data management structure. The end-to-end data flow from its origin at the MODIS instrument to the science product archival and distribution will be described in the EOSDIS context.

The collection and management of input data, ancillary data and data from external sources will be described as they relate to the processing and production of MODIS products. The manner in which reference data for ground control points, digital elevation and terrain models, atmospheric models, coastline definitions, etc. will be collected and integrated into the system will be specified.

The Plan will describe the output products to be produced, including standard, quicklook, browse, and special products. The descriptions will include the volume, product level, and format of the output data. The use of IWG approved standard formats will be emphasized.

PRELIMINARY OUTLINE AND DRAFT

The software development and validation schedule will be consistent with the EOS Science Software Development Schedule. It will be based upon three initial software versions prior to launch:

- B Test migration from the SCF to the EOSDIS, exercise interfaces, and test execution in operational environment.
- V1 Correct any problems in the B Version, complete operator interface, generate all messages.
- V2 Software ready for launch. Final integration, test of operations procedures, training of operations staff.

Each delivery will include software, test data, user's guide, operations guide, and software version description.

Consideration will be given to the processing and management of quick look data for field experiments, targets of opportunity and special investigations. A plan will be provided for the generation and delivery of metadata and browse products.

The responsibilities of the MODIS Science Data Support Team will be identified in terms of data product requirements, operational scenarios, algorithm integration and testing, design and implementation of the operational processing system, data product validation, integration of computer resources, and development of documentation during the definition, prelaunch and postlaunch phases.

2 MODIS Mission Overview

- 2.1 Mission Objectives
- 2.2 Scientific Objectives

3 MODIS Instrument Overview

- 3.1 Experimental Objectives
- 3.2 Instrumentation
- 3.3 Purpose
- 3.4 Operation/Data Modes

4 MODIS End-to-End Data Flow

- 4.1 Space Segment
- 4.2 Ground Segment
- 4.3 Science User Segment

PRELIMINARY OUTLINE AND DRAFT

5 MODIS Software and Data Policy Overview

6 MODIS Science Software

- 6.1 Level-1 Software
 - 6.1.1 Level-1A Design
 - 6.1.2 Level-1B Design
 - 6.1.3 Code Development
 - 6.1.4 Test Plan & Test Data Development
 - 6.1.5 Test and Delivery to ECS
- 6.2 Level-2 Software
 - 6.2.1 Coding Recommendations
 - 6.2.2 Level-2 Processing Shell
 - 6.2.3 Review of Team Member Algorithms
 - 6.2.4 TM Code Development
 - 6.2.5 Integration of Team Member Code
 - 6.2.6 Test and Delivery
 - 6.2.7 End-to-End Test
 - 6.2.8 Post Launch Activities
- 6.3 Utilities
- 6.4 Configuration Management

7 MODIS Data Processing

- 7.1 Data Descriptions
- 7.2 Data Processing and Analysis
- 7.3 Data Calibration and Validation
- 7.4 Data Products
- 7.5 Data Formats, Rate and Volume

8 MODIS Data Storage and Archiving

- 8.1 Information Archive
 - 8.1.1 Directories and Catalogs
 - 8.1.2 Metadata
 - 8.1.3 Browse Data
 - 8.1.4 Software and Documentation
- 8.2 Data Archive
- 8.3 Data Security

9 MODIS Data Access and Distribution

10 MODIS Data Discard Policy