3 1176 00140 2768

NASA Contractor Report 159203

NASA-CR-159203
1980 0011580

DEVELOPMENT AND USAGE OF A FALSE COLOR DISPLAY TECHNIQUE FOR PRESENTING SEASAT-A SCATTEROMETER DATA

Clayton B. Jackson

RESEARCH TRIANGLE INSTITUTE
Research Triangle Park, North Carolina 27709

NASA Contract NAS1-15338 (Task 1)
January 1980

LIBRARY COPY

JUL 14 1980

LANGLEY RESEARCH CENTER
LIBRARY, NASA
HAMPTON, VIRGINIA

NASA
National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

ACKNOWLEDGEMENT

This report was prepared by the Research Triangle Institute, Research Triangle Park, North Carolina, for NASA under Contract No. NAS1-15338. The work was administered by personnel in the Flight Electronics Division at Langley Research Center, with Lyle C. Schroeder serving as the Langley Technical Representative, and William L. Gratham providing preliminary task assignments.

This study was conducted and authored by Clayton B. Jackson of the Electronic Systems Department. Dr. Charles L. Britt, Jr., Special Project Coordinator, served as Project Leader, with James B. Tommerdahl, Division Director, acting as Laboratory Supervisor. The final manuscript was typed by Claudine E. Wilson.

N80-19862#

TABLE OF CONTENTS

Page

1.0 INTRODUCTION 1
1.1 Requirements 4
1.2 Outline 4

2.0 FALSE COLOR ALGORITHMS 6
2.1 Data Gridding and Blending 6
2.2 Map Projections 8
2.2.1 Mercator 8
2.2.2 Polar Stereographic 10
2.3 Display File Preparation 11

3.0 PROGRAM DESCRIPTION 13
3.1 SASS Display Generation 13
3.1.1 Preprocessing 13
3.1.2 Postprocessing 16
3.2 Display Annotation 17
3.3 Map Outline Generation 19
3.4 Merging 22

4.0 PROGRAM OPERATION 24
4.1 Execution of FALSCLR 24
4.1.1 The Control Deck 25
4.1.2 Data Cards 25
4.2 Generating Annotation 28
4.3 Execution of MAPGEN 30
4.4 Merging the Files 32

5.0 THE DIGITAL DISPLAY EQUIPMENT 36
5.1 Description of the System 36
5.2 System Operation 37

6.0 FUTURE DEVELOPMENTS 39
6.1 Data Reduction 39
6.2 Display Generation Changes 40

7.0 REFERENCES 42

1.0 INTRODUCTION

The objective of this report is to describe a computer generated false color display program which creates digital multicolor graphics. The primary goal of this program is to display geophysical surface parameters measured by the SEASAT-A satellite scatterometer (SASS). This False Color technique shall be used to assist the NASA personnel at Langley Research Center involved in the SASS experiment.

The SEASAT-A satellite has a special purpose microwave radar scatterometer which is used to measure the normalized radar cross section (NRCS) of the earth's surface. While operating in its various modes, the satellite can measure NRCS for fifteen angles of incidence both forward and aft of the flight path. The cross-section measurements are used to infer surface roughness and reflectivity. Over the ocean the NRCS can be used to calculate surface wind speeds and define ice/water boundaries [ref.1]. Overland it has been conjectured that the NRCS could demonstrate seasonal variations in soil moisture [ref.2].

False color displaying of data sets is simular in principle to contouring or gray-scaling. The data is incrementally scaled over the range of acceptable values and each increment and its data points are assigned a color. The advantage of the false color display is that it visually infers cool or weak data versus hot or intense data by using the rainbow of colors. For example, with wind speeds, levels of yellow and red could be used to imply high winds while green and blue would imply calmer air.

SASS data is sorted into geographic regions and the final false color images are projected onto various world maps with superimposed land/water boundaries. Figures 1.1 and 1.2 are two examples of the SASS data taken during orbit 331, and are shown here in a polar stereographic and mecator map format, respectively. It is this final format that will provide an excellent qualitative as well as quantitative look at the SASS sensor measurements.

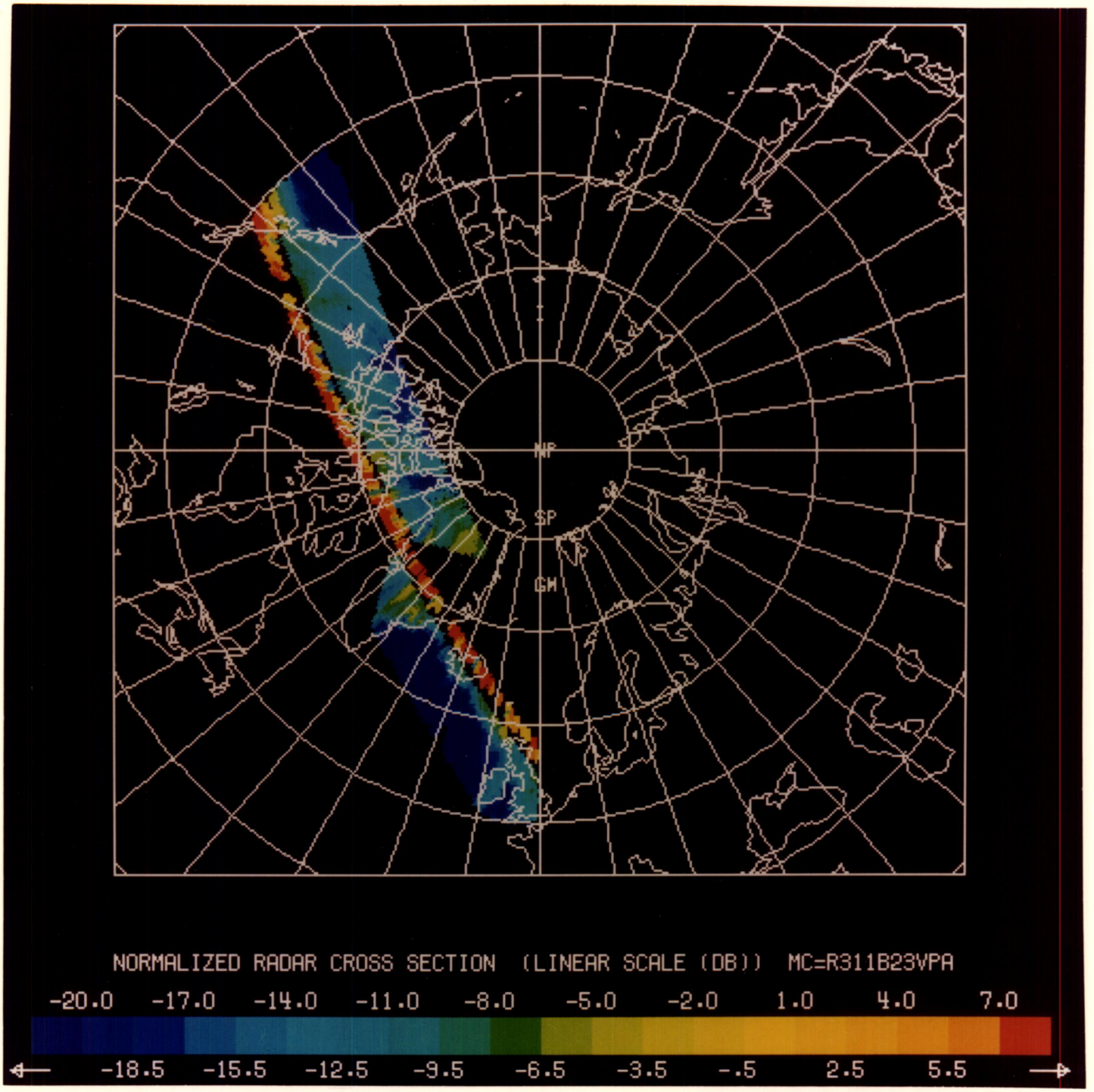

Figure 1.1 The final display format using a Polar Stereographic map projection. Data is from beams 2 and 3 of REV 331.

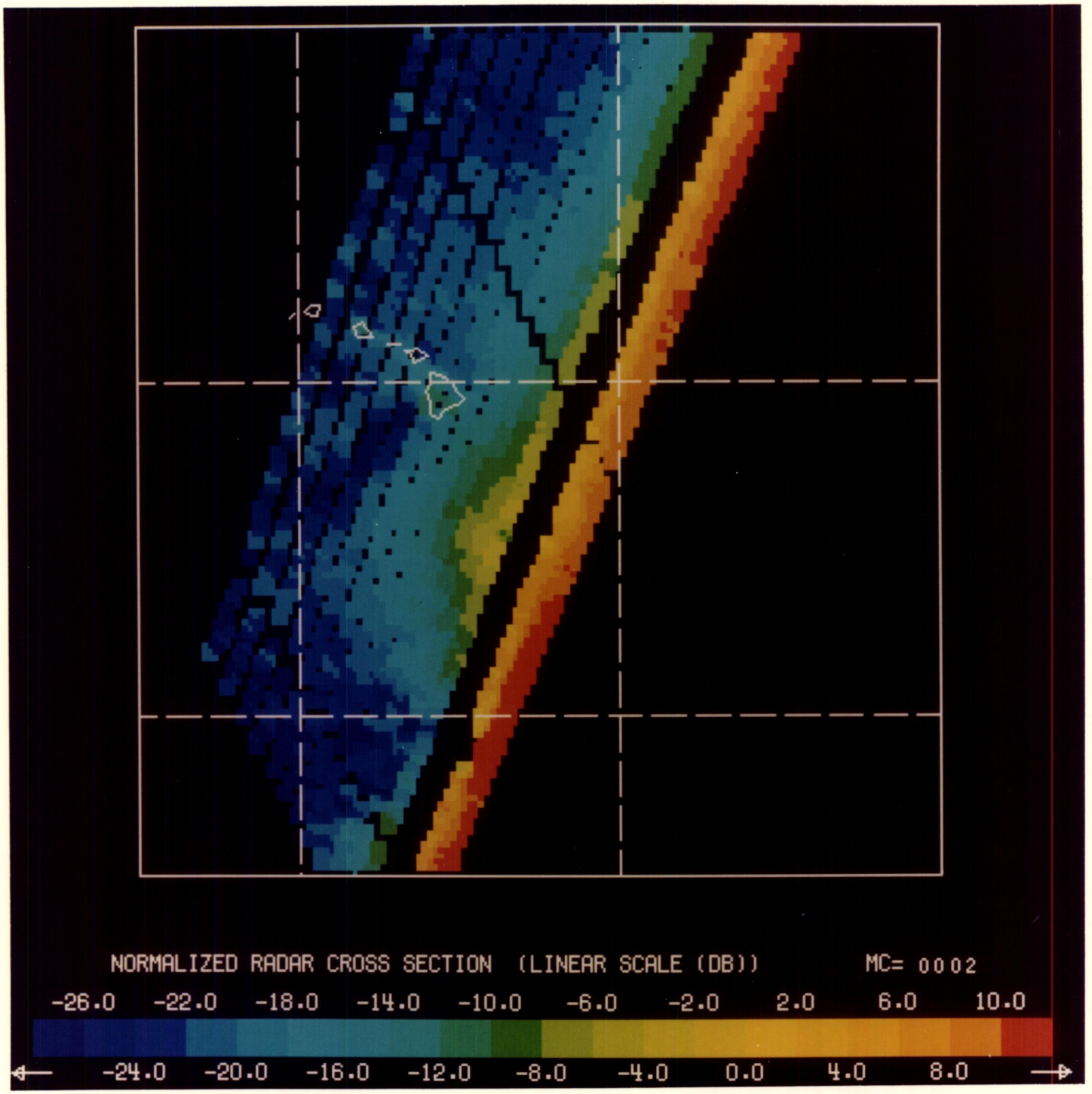

Figure 1.2 The final display format using a Mercator map projection. Data is taken about Hawaii during Hurricane Fico from beam 2 of REV 331.

1.1 Requirements

SEASAT-A data is received and processed by Jet Propulsion Laboratories which then issues the appropriate sensor data on mag tapes to various contractors. At the time of this report, a complete tape with the final format was not available. However, several tapes have been issued which have partial but adequate sensor data records for demonstration of the program

It was a requirement that the final program be adaptive in the sense that the user should be able to control data selection and the final false color format. The basic specifications imposed on the software can be listed as follows:

- The SASS sensor records shall be read and sorted by mode, beam number, polarization and cell number into earth-located data subsets.
- The data subset shall be gridded using selectable pixel sizes, and an optional pixel blending feature shall be used to interpolate for missing SASS data.
- The gridded data shall be displayed in either a mecator or polar stereographic map format.
- The display color scale shall have a selectable linear, logarithmic or exponential relationship to the data.
- The final display shall be superimposed over land/water boundaries.

1.2 Outline

The handling of the SASS data and the ultimate creation of a false color display is handled in a self-contained routine called FALSCLR. Called by the main routine are several subroutines which use display algorithms to create the false color data file. The purpose of Section 2.0 is to describe the various gridding, smoothing and map projection algorithms.

It became apparent during the course of the software development that four separate programs were necessary to fulfill the task requirements. Besides FALSCLR, two routines were developed to handle display

annotation and land/water boundaries, respectively. A fourth program was developed to merge these three display files into a complete false color picture. Section 3.0 is devoted to the description of each of these four routines.

Section 4.0 contains a description of how to coordinate the four routines into a meaningful job. The section contains a complete outline of program operation so that the routines can be run by the user who is reasonably familiar with the false color equipment. Since job control cards are an important part of program control; and since it is possible to run multiple programs with the proper cards, a discussion of candidate control cards is included. Also, described in detail are all possible data cards along with their input formats.

A discussion on the operation of the digital display system is contained in section 5.0. Included is a brief description of the display equipment and a simple guide to its operation as it might apply to the false color display techniques mentioned here. The user may find this section useful in his initiation to the system.

Section 6.0 is devoted to a discussion of future development in program software. It will be necessary to eventually adapt the routines to handle new data tapes as they arrive. It has also been suggested that a routine should be developed to do selective normalization of the SASS data. These and future modes of operation are the subject of this section.

2.0 FALSE COLOR ALGORITHMS

The purpose of this section is to detail the algorithms used by the false color program to generate packed integer map projection files from the raw SASS data. All the following algorithms are executed within the framework of the basic raw data to display routine called FALSCLR. The algorithms are very general and can easily be adapted to similar problems where data must be digitized.

2.1 Data Gridding and Blending

The SASS measurement of the normalized radar cross section (NRCS) is taken along swaths of area over the earth that correspond to about 950 kilometers along either side of the satellite ground flight path. The data is taken continuously as the satellite orbits around the earth, and the final data tapes contain segments from these orbits. Once the data has been sorted into geographic subsets a large file of unequally spaced data points remains. To digitize these SASS data, one first extrapolates equally spaced points within the desired geographic region. Then the proper SASS data value is assigned to each of these points by interpolation. The operation is referred to as data gridding and is performed in a routine called SSGRID.

The gridding routine is designed to project unequally spaced points onto a grid matrix which is a linear latitude-and-longitude matrix of equally spaced grid boxes called pixels. The pixel size in kilometers is a variable controlled by the user. The final grid matrix has dimensions n x m given by:

$$\begin{aligned} n &= (\phi_1 - \phi_2) * 111.112 \text{ km/pixel size} \\ m &= (\psi_1 - \psi_2) * 111.112 \text{ km/pixel size} \end{aligned} \qquad (2\text{-}1)$$

where ψ_1, ψ_2, ϕ_1, ϕ_2 define the latitude and longitude coordinates, respectively, for the region of interest. The angular size of each pixel is defined as:

$$\begin{aligned} \Delta\psi &= (\psi_1 - \psi_2)/m \\ \Delta\phi &= (\phi_1 - \phi_2)/n \end{aligned} \qquad (2\text{-}2)$$

Each pixel is assigned a data value by computing the weighted average of data points within the pixel. The weighting used is known as inverse squared distance or gravity weighting. Only those data values which fall within a particular pixel can contribute to the value of that pixel.

The SASS data file contains unequally spaced data points $U_k(a_k, \psi_k, \phi_k)$; where a_k, ψ_k, ϕ_k are the SASS parameter value, latitude position, and longitude position, respectively, for each data point k. The gridding algorithm is defined as follows:

For each data value U_k in the SASS data file, perform the following functions:

a) compute the matrix address (i,j) of the grid pixel containing this point by:

$$i = \phi_k/\Delta\phi + 1$$
$$j = \psi_k/\Delta\psi + 1 \qquad (2\text{-}3)$$

b) define the coordinates of the grid pixel (i,j) as x_{ij}, y_{ij} and compute the dimensionless range r_k as:

$$r_k^2 = \left(\frac{y_{ij} - \psi_k}{\Delta\psi}\right)^2 + \left(\frac{x_{ij} - \phi_k}{\Delta\phi}\right)^2 \qquad (2\text{-}4)$$

c) compute the running sums P(i,j) and Q(i,j) as follows:

$$P(i,j) = \sum_k a_k/r_k^2$$
$$Q(i,j) = \sum_k 1/r_k^2 \qquad (2\text{-}5)$$

After the entire data set has been exhausted, the pixel data values V(i,j) are given by:

$$V(i,j) = P(i,j)/Q(i,j) \qquad (2\text{-}6)$$

If no data falls within a particular pixel, that pixel is flagged for missing data.

Associated with the gridding process is an optional pixel blending feature. Upon request, a routine called SMTHGD is called to interpolate data values for pixels which were flagged for missing data. The pixel value is computed as the weighted average of the pixels surrounding that particular pixel. As in the gridding routine, gravity weighting is used to interpolate this value. The degree of blending is user controlled. The value of the grid pixel (i,j) is found to be:

$$V(i,j) = \sum_{\ell k} W_{ijk\ell} \cdot V(k,\ell) \qquad (2\text{-}7)$$

where (k,ℓ) is the 2-dimensional address of the pixels, (1) surrounding the pixel (i,j) and (2) having range $r_{k\ell}$ from the center of this pixel. $W_{ijk\ell}$ are the normalized weights associated with the pixels $V(k,\ell)$ having range $r_{k\ell}$.

2.2 Map Projections

The map projection algorithms are designed to perform the non-linear transformation of gridded SASS data onto two common map types. The mercator and polar stereographic maps were selected because they complement each other in areas of minimum land mass distortion, and SASS data analysis can be effectively done with these map types.

2.2.1 Mercator.- The mercator map is a cylindrical projection of the earth's surface and is ideally suited for the tropic latitudes [ref.3]. Full world projections are possible with this map type; however, considerable vertical map distortion will result. The properties of the mercator map are that the longitude lines are parallel and are of equal distance throughout the map. The latitude lines are also parallel, but their spacing is a function of latitude. The coordinates of the map are those used for the grid matrix, and the map projection is scaled to fit within the user's specified pixel dimensions.

The algorithm for projecting the grid matrix onto the mercator map can be throught of as a magnifying routine which stretches or shrinks the grid matrix to fit the map. The longitudinal or grid column magnifying factor Δy

is constant throughout the transformation and is given by:

$$\Delta y = [n_1/n] \tag{2-8}$$

where n_1 and n are the longitudinal pixel sizes of the map and grid, respectively. The latitudal or grid row magnifying factor is a logarithmic function of latitude and is calculated for each row in the grid matrix. First, a scaling factor and the map's projected equator distance are given by:

$$s = m_1/(a-b)$$
$$eq = m_1 \cdot a/(a-b) \tag{2-9}$$

where m_1 is the map's latitudal pixel size and a and b are given as

$$a = \log_e(\tan(45° + \psi_1/2))$$
$$b = \log_e(\tan(45° + \psi_2/2)) \tag{2-10}$$

Then, for the jth row of the grid matrix, we find:

$$c_j = \log_e(\tan(45° + (\psi_1 - j.\Delta y)/2))$$
$$d_j = s \cdot c_j \tag{2-11}$$

The jth latitudal magnifying factor is given as:

$$\Delta x_j = [eq - d_j] \tag{2-12}$$

The map values M(i,j) are finally given as:

$$\text{For } k=1, \ldots, \Delta x_j \text{ and } j = 1, \ldots, \Delta y \tag{2-13}$$

$$M(k,\ell) = V(i,j)$$

where V(i,j) are the ith, jth grid values.

2.2.2 Polar stereographic.- The polar stereographic map is a plane conformal-azimuthal projection that is ideally suited for the polar regions. Because it is a continuous map of the globe, it is more useful in hemisphere charts than the mercator. The polar stereographic map has latitude lines which are closed concentric circles about the pole and the longitude lines which radiate outward from the center. The map is scaled to fit within user inputted dimensions and has an outermost latitude ψ_3 such that $\psi_3 \leq \psi_2$.

Projecting the grid onto the map is completely non-linear and is considerably more complicated than the mercator. Basically, the algorithm calls for calculating the center latitude and longitude of every pixel in the map matrix thereby determining which grid pixel is associated with it. The calculation goes as follows: given an $n_2 \times n_2$ map matrix calculate the following scale factor:

$$s_1 = 1/2 \cdot n_2/\tan(45° - \psi_3/2) \tag{2-14}$$

where ψ_3 is the standard or outermost latitude. For each ℓth, kth map pixel compute

$$r^2 = (n_3 - \ell)^2 + (n_3 - k)^2 \tag{2-15}$$

where $n_3 = n_2/2$ and r is the radial distance to the pixel from the map's center. Next compute the pixel's latitude $\psi_{\ell k}$ and longitude $\phi_{\ell k}$ as follows:

$$\psi_{\ell k} = 2 \cdot \tan^{-1}(r/s_1) \tag{2-16}$$

$$\phi_{\ell k} = \tan^{-1}((n_3 - k)/(n_3 - \ell)) + 180°$$

The grid matrix element address (i,j) which corresponds to the ℓth, kth map element is found as:

$$i = (\phi_1 - \phi_{\ell k})/\Delta\phi$$

$$j = (\psi_1 - \psi_{\ell k})/\Delta\psi \tag{2-17}$$

Finally the map value $M(\ell,k)$ is given as

$$M(\ell,k) = V(i,j) \tag{2-18}$$

where $V(i,j)$ is the ith, jth grid value.

2.3 Display File Preparation

To display the map projection on the digital display equipment, a special raster file must be created on magnetic tape. This file will contain packed integer values which have a functional relationship to the map's SASS parameter values. A color scaling algorithm is designed to perform the transformation of the map values into scaled integer values which ultimately represent different colors.

It was decided that at most 20 unique colors should be used at any time in the false color display. These colors will have either a linear, logarithmic or exponential relationship to the SASS data. The color algorithm goes as follows:

Define an upper bound M_1 and lower bound M_2 on the map data values $M(i,j)$, and let N_c be the number of colors desired. Compute ΔM, the color incrementing value in map data units as follows:

Linear scale:

$$\Delta M = (M_1 - M_2)/N_c \tag{2-19}$$

Logarithmic scale:

$$\Delta M = N_c/\log_e (M_1 - M_2 + 1) \tag{2-20}$$

Exponential scale:

$$\Delta M = (M_1 - M_2)/\log_e (N_c + 1) \tag{2-21}$$

For each ith, jth map matrix element compute a color integer value N(i,j) as follows

$$\text{If } M(i,j) \geq M_1, \qquad \text{let } N(i,j) = N_c$$
$$\text{If } M(i,j) \leq M_2, \qquad \text{let } N(i,j) = 1 \tag{2-22}$$

If M(i,j) is within M_1 and M_2 then, depending on the functional relationship, N(i,j) is chosen as follows:

Linear scale

$$N(i,j) = (M(i,j) - M_2)/\Delta M + 2 \tag{2-23}$$

Logarithmic scale

$$N(i,j) = \Delta M \cdot \log_e(M(i,j) - M_2 + 1) + 2 \tag{2-24}$$

Exponential scale

$$N(i,j) = 1 + e^{(M(i,j)-M_2)/\Delta M} \tag{2-25}$$

For map values which were flagged for missing data, zero is assigned to N(i,j).

Once the integer assignment is complete, it is necessary to pack the array N(i,j) such that only eight bits are used to represent each integer on the magnetic tape file. This eight bit form is necessary since the tape drive is controlled by a Kennedy model 9218 format control unit which is IBM compatible. Since the CDC computer uses a standard 60 bit word, it is necessary to pack 7 1/2 eight bit words into each word in the output list. This is accomplished by obtaining the lower eight bits of each N(i,j); bit shifting each word a multiple of eight bits; and packing these final, reconstructed words into an output array. This final array is written to tape using free format to guarantee that all 60 bits are transferred.

3.0 PROGRAM DESCRIPTION

This section is devoted to the description of the entire false color display software. Four distinct program units were developed to ease program execution and reduce time and cost by eliminating repetitive runs. The first three units each generate a display file containing part of the total picture. The fourth program unit merges or superimposes these partial images into the complete digital display. Figure 3.1 is the simplified flow diagram describing the total job. A listing of each of these routines is found in the Appendix.

3.1 SASS Display Generation

Routine FALSCLR is a Fortran program designed to process and graphically display various SASS data sets. The program has two logical modes of operation which are defined here as preprocessing and postprocessing. These two modes can be executed jointly or separately at the users descretion. The program's final output is a digital display file containing integer-represented SASS values projected onto one of two map images. Refer to the job flow diagram in Figure 3.2.

3.1.1 Preprocessing.- In the preprocessing mode of operation, FALSCLR accesses the SASS data tapes and performs data sorting and selecting. These SASS tapes contain segments of the satellite's revolutions; and within each segment, there are unique sensor data records. Each record contains the mode of operation defined by the mode, polarization and beam number, the satellite's geographic location, and the fifteen measurements of NRCS taken from the particular antenna in use. A subroutine called DATAIN was designed to read these SASS tapes and pick out the appropriate data. Upon execution, this subroutine prompts the user to define the geographic coordinates of the desired data subset, to enter the desired mode of operation and to specify which of the fifteen NRCS measurements are needed. After the tape search is complete, the final data subset is written to a working file which is accessable to the postprocessing routines.

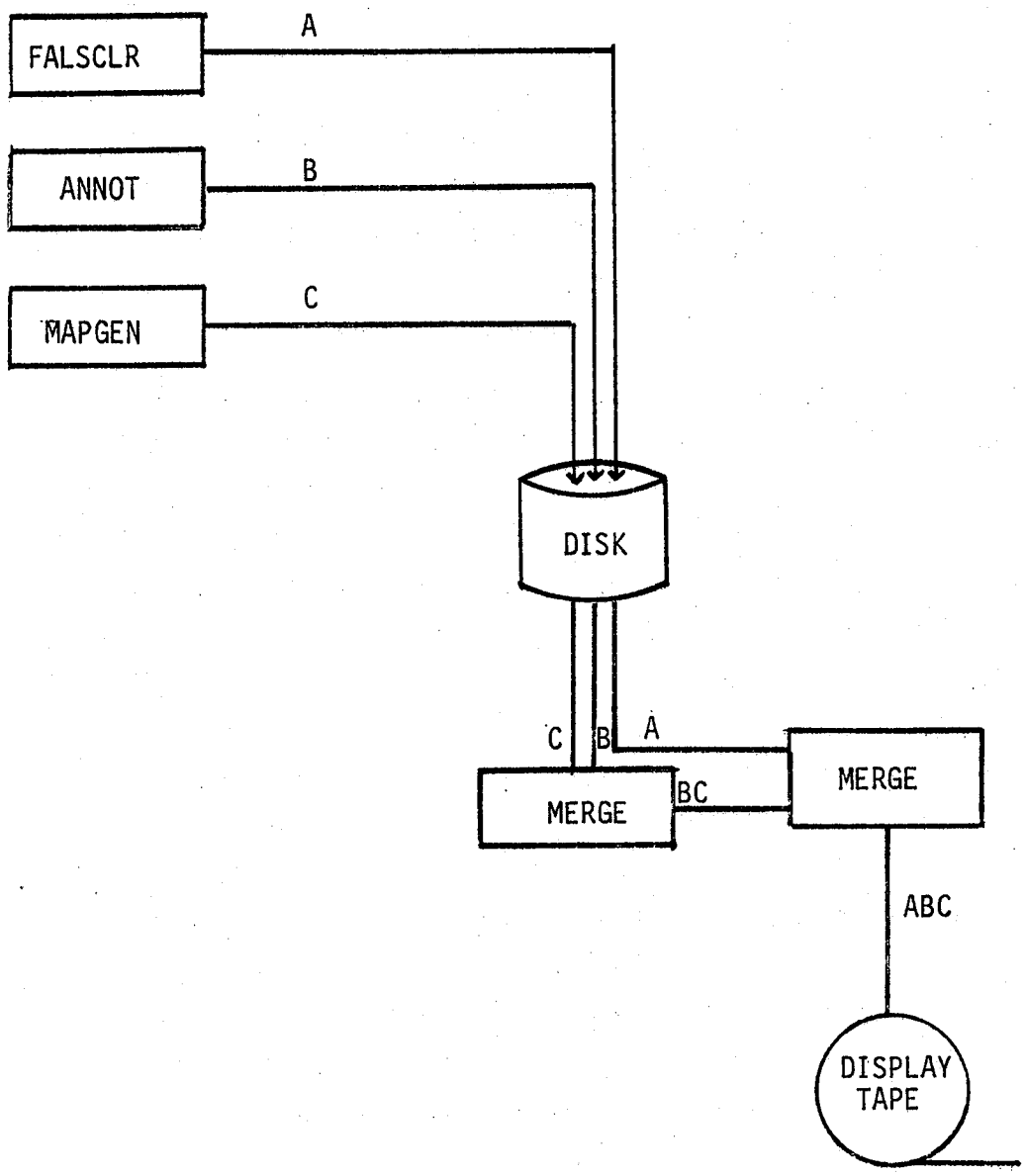

Figure 3.1 Simplified flow diagram of the complete job.

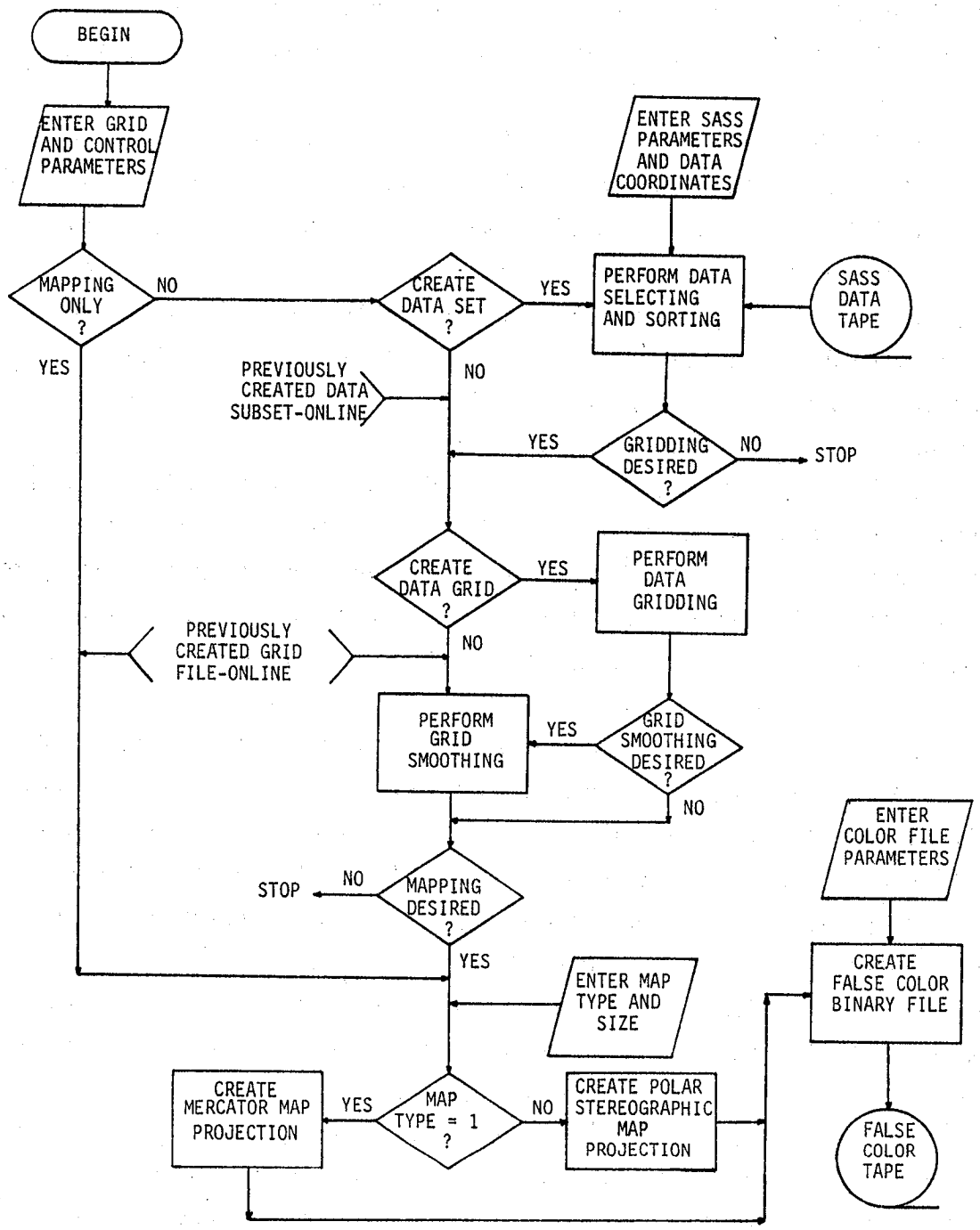

Figure 3.2 Flow diagram of program FALSCLR.

3.1.2 Postprocessing.- The postprocessing mode contains routines to access the preprocessed data subset and create the final display file. There are six subroutines invoked by FALSCLR to perform this operation. The algorithms used by these routines are discussed in detail in the previous section. The final output file is format compatible with the digital display equipment and is stored on disk for future access.

Two of the six subroutines, SSGRID and SMTHGD, are designed to access the data subset and create an equally-spaced grid matrix of SASS values. Subroutine SSGRID requires the user to define the geographic bounds of the grid and the size in kilometers of the grid's square pixels. The routine then develops a linear latitude-and-longitude grid matrix within the requested bounds. SASS values are then assigned to each grid pixel by weighted interpolation of all the SASS data geographically located within the pixel.

At the option of the user, a grid smoothing routine SMTHGD is invoked. This routine attempts to define a SASS value to those grid pixels containing no SASS data points. The interpolation is done by obtaining the weighted average of the pixels surrounding that pixel. The smoothed grid matrix is written to a working file which is passed on to later stages of the program.

The next two subroutines in the postprocessing operation, MERCAT and POLAR, were designed to operate exclusively of one another. They both access the above grid matrix and project the data values onto a nonlinear map matrix. If invoked, subroutine MERCAT obtains the grid file and projects this file onto a log-linear matrix defined by the equations of a mercator map projection. The mercator map is a cylindrical projection which has linearly spaced longitude lines and logarithmicly spaced latitude lines. The size of the map is user specified and should be such that minimal area distortion is maintained.

Paralleling MERCAT is subroutine POLAR which projects the gridded data onto a map matrix defined by the equations of the polar stereographic map. This map type is a plane conformal projection which has concentric rings of latitude, about either pole, spaced tangentially from the map's center. The longitude lines are equally spaced and

radiate outward from the center. The map matrix dimensions are square and, though user specified, there should be some constant value for all projections of this type.

The final two subroutines in the postprocessing mode are routines COLORS and PACKEM. These two programs create the proper display file for projection on the digital display equipment. Subroutine COLORS accesses the map file described above and redefines the SASS values as integers. These integers have either a linear, logarithmic or exponential relationship to the SASS data. There are at most twenty integer values which ultimately represent different colors in the false color display.

Following COLORS is subroutine PACKEM which creates an output file which is a packed version of the file created by COLORS. To be compatible with the digital display tape drive, the above integer values should be written to tape using only the lowest eight bits. Routine PACKEM accomplishes this by packing several CDC 60 bit words with 7 1/2 eight bit integers. It then outputs the 60 bit words to magnetic tape. This tape file can be displayed on the digital CRT for test purposes; however, the picture is not complete until this display is matched with those from the other program units.

3.2 Display Annotation

Routine ANNOT is a Fortran program designed to create annotation for the final false color display. ANNOT generates a plot vector file which contains non-standard plot calls. This plot file is eventually converted to a special raster plot file which is compatible with the display equipment. This file is saved on disk to be merged later with the false color file generated in FALSCLR. Since many displays would likely use the same annotation, this display file could be used repeatedly at a substantial savings of time and cost. Refer to the flow diagram in Figure 3.3

The annotation generation by ANNOT is controlled by the user at the time of execution. A ten character map code is entered which generally defines the final displays using this annotation. Other various control values are input which define the color legend. This

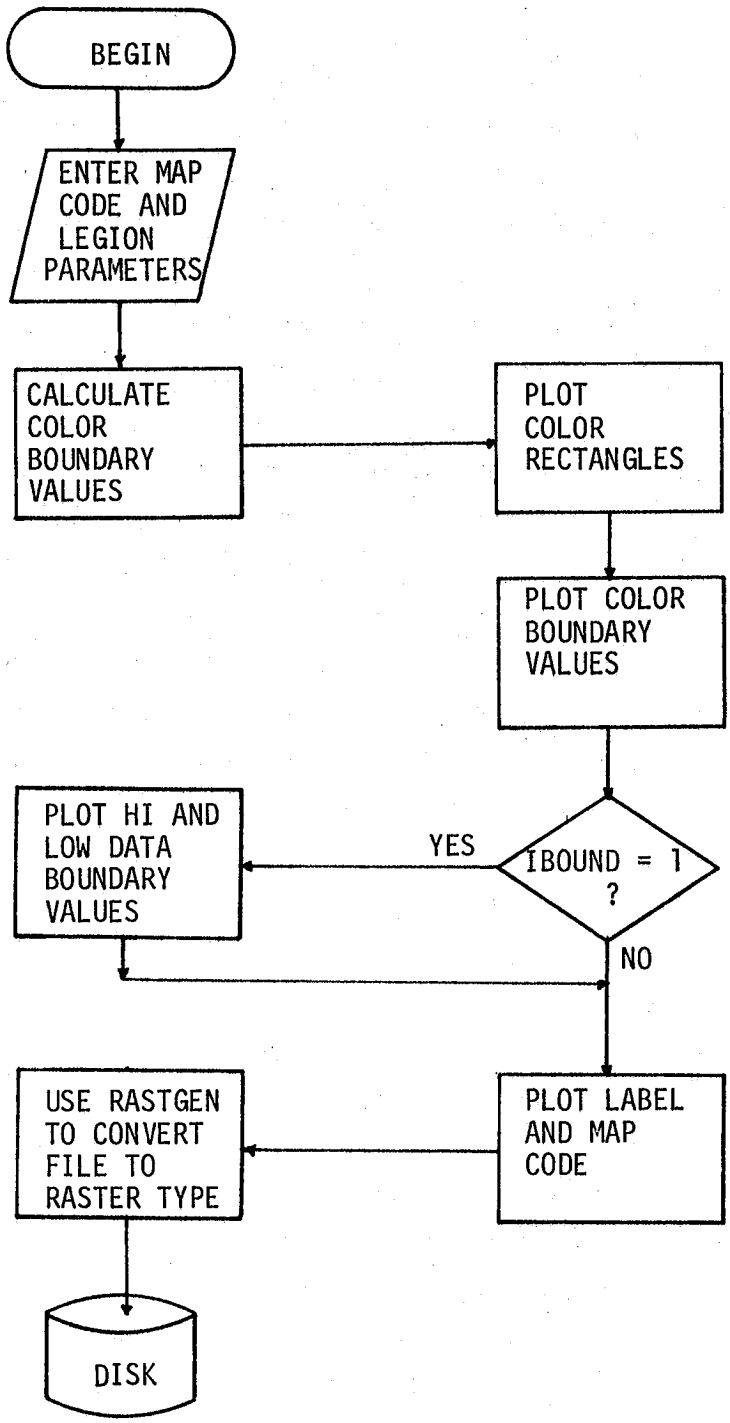

Figure 3.3 Flow diagram of routine ANNOT.

legend contains color rectangles defining the total band of colors used and the color boundary data values in decibels. There is also a map label which states that this is a map of normalized radar cross section and that the scale is either linear, logarithmic or exponential in decibels.

The annotation plot file is generated by invoking various plot subroutines from the Langley Research Center's (LRC) graphics package. These subroutines create a plot vector file that generally would be accessed by the CalComp plotter for plotting the file image onto paper. However, routine ANNOT uses several non-standard CalComp plot commands which make the file incompatible with the CalComp plotter. Instead of plotting the file, a special raster generation routine called RASTGEN is implemented which converts the plot vector file to a raster type file. This raster file is saved on disk for future merging with the other display files.

Program RASTGEN was developed at LRC by Robert Oakes and revised for general usage by Frances T. Meissner. It is used to convert any plot vector file containing the proper non-standard CalComp plot calls into a device-compatible image file for displaying on the digitizing equipment. RASTGEN is located on disk and is available to anyone who knows the proper user number. This disk file is a source image and must be compiled before execution. The raster file created by the program always contains 512 records each having 70 words. This translates to 512 x 512 optical pixels when seen on the CRT or photographed by the Dicomed imaging recorder.

3.3 Map Outline Generation

Routine MAPGEN is a Fortran program which generates a plot file containing map images. These map images are either polar stereographic or mercator and contain latitude and longitude lines and superimposed land/water boundaries. The plot file is converted to a special raster file as was done for program ANNOT. The raster file is saved on disk for future merging with the other display files. Refer to Figure 3.4 for the job flow diagram.

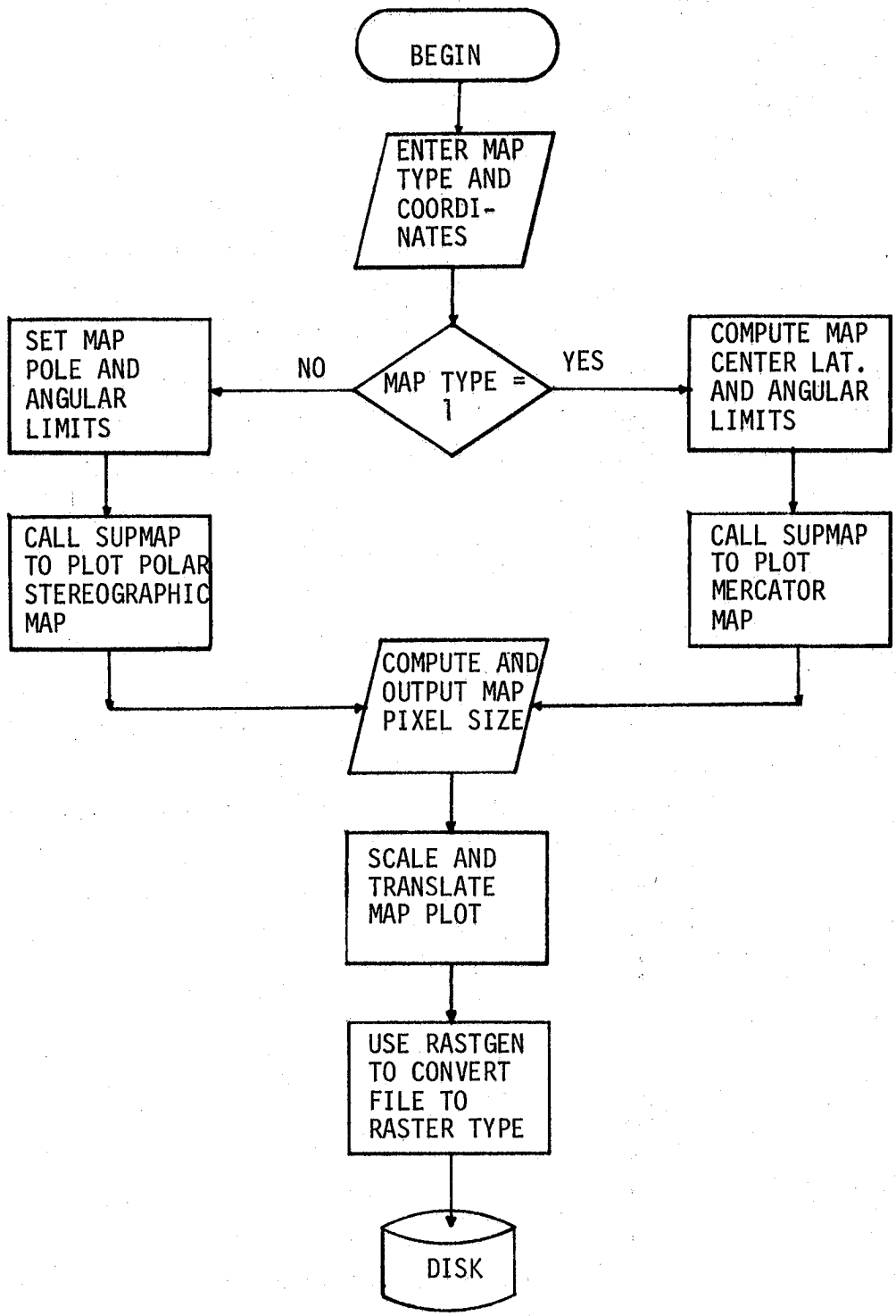

Figure 3.4 Flow diagram of routine MAPGEN.

MAPGEN creates a plot file by making use of a special graphics package available at LRC. This package is called SUPMAP and was developed by the National Center for Atmospheric Research (NCAR) in Boulder, Colorado. The routines in this package are designed to create a plot file which is compatible with the CalComp plotter. The SUPMAP plotting options available to user are : (1) a geographic region can be plotted using one of nine possible map projections; (2) if desired, land/water boundaries are plotted using dashed or solid lines, (3) a border can be drawn around the plot frame; (4) U.S. state borders can be plotted; and (5) the latitude and longitude lines can be plotted at selectable intervals. MAPGEN makes use of all but option 4. The latitude and longitude line intervals are chosen to be ten degrees but can easily be changed. The plotted land/water boundaries have a specified resolution of one degree which is the one limiting factor when generating detailed map projections.

SUPMAP has been revised and adapted to the LRC facilities by Patricia A. Winters. A new plot call translator was developed which allows the user to scale the plot image and offset the plot origin. This option was necessary so that the final plot image is identical in scale and location to the data display file created in FALSCLR. The plot translator also includes various non-standard CalComp plot calls similar to those mentioned in ANNOT.

The plot vector file created by MAPGEN is temporarily stored on disk. Execution of RASTGEN completes the necessary conversion of this file into a display image raster file, much the same as in ANNOT. This raster file also contains 512 records each having 70 words. In this form, it is compatible with the display equipment; and, if projected onto the CRT, uses 512 x 512 optical pixels. Once generated this map file could be used repeatedly for displays requiring the same map type and area. The file is saved in the user's disk area along with the files created by FALSCLR and ANNOT. Together they complete the total false color display.

3.4 Merging

Routine MERGE is a Fortran program designed to do a bit-by-bit merge of two binary files. The routine is intended to superimpose two display files and output this result to a new file. If run twice using the files created by FALSCLR, ANNOT, and MAPGEN, routine MERGE can create the complete false color display of the SASS data. Refer to the flow diagram in Figure 3.5.

To execute MERGE, two raster display files must be available to the program. At least one file must have 512 records, such as those created by ANNOT and MAPGEN. If the second file is smaller than this, an offset is computed which properly aligns the eight bit integer values of the two files. This offset is necessary for all files created by FALSCLR, which are at most 400 records by 54 words in size.

After the two files have been properly aligned, a record from each file is read using free format reads. The two records are then merged by performing a bit-by-bit OR on every common bit. This bit-by-bit OR can be described as the logical summing of two bits such that if one or both of the two bits have the value 1, then the resulting sum equals 1. This summing has the desirable feature of allowing the map outlines, whose integer values are 255, to mask out any SASS data jointly located in the same spot. This maintains the integrity of the map with only a minimal loss to the data set. After the merging is complete, the results are written to a new file, and the next two file records are read. The final output file is temporarily stored on disk until it can be copied to magnetic tape.

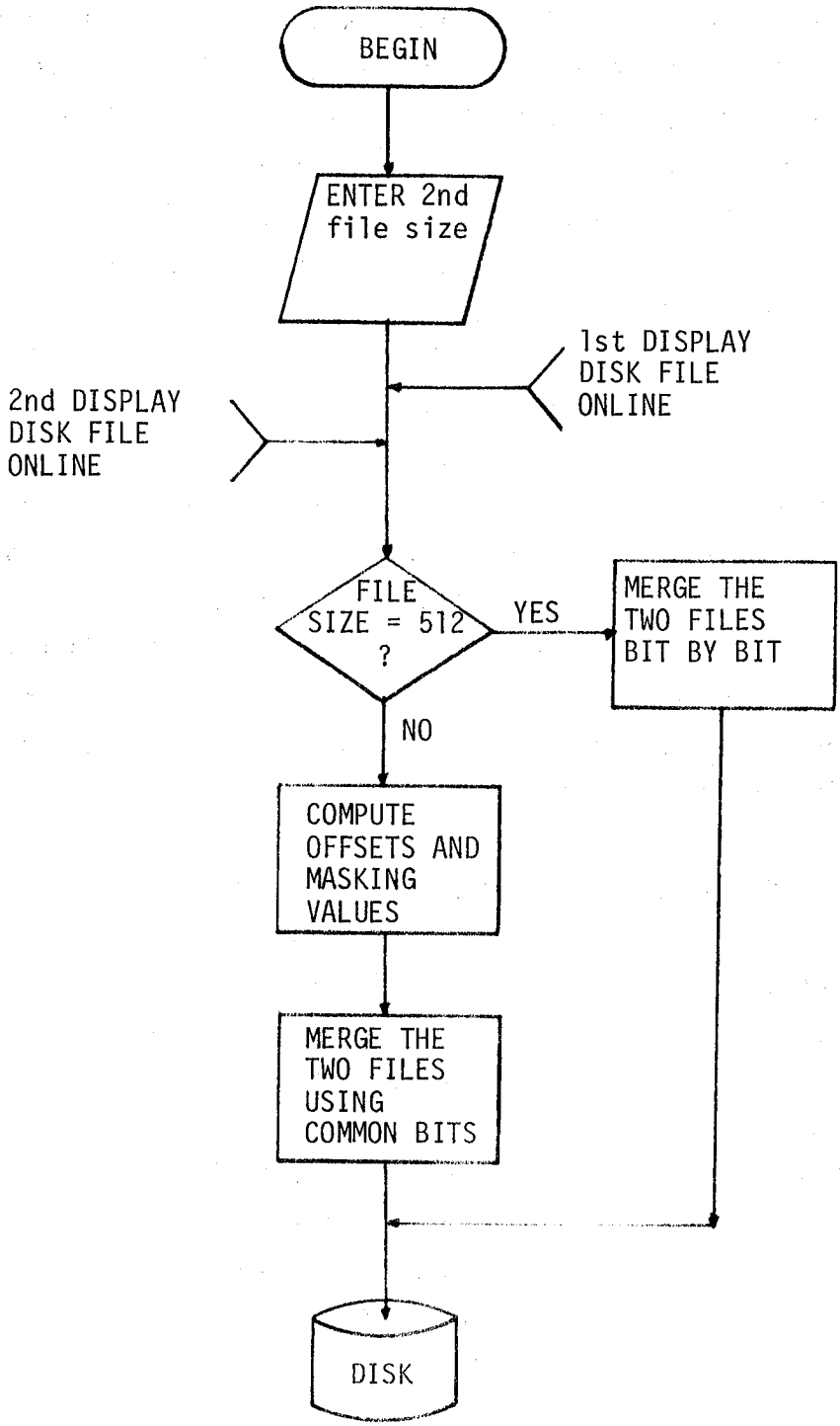

Figure 3.5 Flow diagram of routine MERGE.

4.0 PROGRAM OPERATION

This section is devoted to discussion of the operation of the four display generation routines. It is intended to assist those users who are familiar with the NOS control system and the Fortran extended version 5 language as used by the CDC computer at Langley Research Center. The section contains a brief description of the general job cards and deck setups. A discussion of each input card and its effect on job flow is also included. Reference is made to the source listings which can be found in the Appendix.

4.1 Execution of FALSCLR

As mentioned earlier, FALSCLR's primary goal is to generate a false color display of only the SASS data. It expects input from the data tapes and generates a local file containing the display image. Intermediate files are created during execution which contain (1) the data subset, (2) the raw grid matrix, (3) the smoothed grid matrix and, (4) the map projection matrix. Table 4-1 gives the local internal names and description of each of these files.

TABLE 4-1. FILES COMMON TO ROUTINE FALSCLR

FILE NAME	DESCRIPTION
TAPE1	Unsmoothed grid matrix
TAPE2	Smoothed grid matrix
TAPE3	Unchanged Mercator Map matrix
TAPE4	Unchanged Polar Stereographic map matrix
TAPE8	Final false color display
TAPE9	SASS data subset
TAPE55	SASS magnetic tape

The program can be run in parts by having the proper files available at time of execution. If, for example, the smoothed grid file were created

at an earlier time, the program could be executed by starting with the mapping routines and extracting the needed data from this grid file. This option and those like it are controlled by the user's input.

4.1.1 The Control deck - Routine FALSCLR must be executed as a remote batch job because it requires about 127K octal of core memory. The source deck is typically on data cards which are read by the card reader and the object code is stored on disk using the file name FALSCLR. The control deck to perform this operation is given in Figure 4.1.

To run the job, the user needs only access the file FALSCLR and begin execution. The general job card deck for a complete start to finish program is shown in Figure 4.2. In the figure, the 'SASS tape' name is the external NOS label of the magnetic tape. The 'user disk file name' is any file name the user wishes to give the final false color display. Finally, the EOR and EOF refer to the multipunched end of record (789) and end of file (6789) cards, respectively.

To run the program in part, the job card deck must be modified to make available the proper data files (see Table 4-1). If, for example, the user wishes to only create the data subset and save it on disk, then file TAPE8 is changed to TAPE9 in the deck above. Later, a display could be generated from TAPE9 by changing the LABEL card to a GET(TAPE9-'data subset name') and keeping the SAVE(TAPE8=...) in its place. Several other options such as this are available by selecting the proper file names. Refer to the flow diagram in Figure 3.2.

4.1.2 Data Cards - The number of data cards needed for any one job varies depending on what operations are done. There are several combinations of cards all of which are mentioned here. All input to the program uses list directed format where commas and end of cards are the value separators. It is necessary to refer to the comments at the end of the program FALSCLR in Appendix B to assist in the description of the data card deck.

The first data card for FALSCLR is a program control card which sets the starting and stopping points of program execution. This card is always the first card in the data deck. It is referred to in the input comments as CARD 1. This card determines the nine possible modes

```
FALSCLR,T50,CM77000.
USER...
CHARGE...
FTN(B=FALSCLR,OPT=2,A,R)
SAVE(FALSCLR)
EXIT.
-EOR-
       source deck
-EOF-
```

Figure 4.1 Creating the task image of FALSCLR.

```
FALSCLR,T500,CM130000.
USER...
CHARGE...
GET(FALSCLR)
LABEL(TAPE55,NT,PO=R,F=I,LB=KL,R,VSN='SASS tape')
FALSCLR.
RETURN(TAPE55)
REWIND(TAPE8)
SAVE(TAPE8='user disk file name')
EXIT.
-EOR-
       data cards
-EOF-
```

Figure 4.2 Control deck for execution of FALSCLR

of program operation. These modes are combinations of distinct program sections and are given as follows:

Mode 1 - Data Sorting
Mode 2 - Data Sorting + Gridding
Mode 3 - Data Sorting + Gridding + Smoothing
Mode 4 - Data Sorting + Gridding + Smoothing + Mapping
Mode 5 - Gridding
Mode 6 - Gridding + Smoothing
Mode 7 - Gridding + Smoothing + Mapping
Mode 8 - Smoothing + Mapping
Mode 9 - Mapping

Here, mapping refers to the creation of the map projection and converting it to the false color file. Depending on the mode, there are seven other possible input cards each of which is described in the program listing. Table 4-2 shows which input cards are necessary for each mode of operation.

TABLE 4-2. NEEDED INPUT FOR EACH MODE OF FALSCLR

MODE NUMBER	INPUT CARD NUMBERS
1	1,2,3,4,5,6
2	1,2,3,4,5,6
3	1,2,3,4,5,6
4	1,2,3,4,5,6,7,8
5	1,2
6	1,2
7	1,2,7,8
8	1,2,7,8
9	1,7,8

There are several points worth noting here about certain data cards. First, if the grid coordinates are needed (Card 2), then the longitudes must be in degrees east. The SASS tapes use a 0 to 360 east

scale which causes discontinuity at the Greenwich meridian. The user, however, can input a negative value for the minimum east longitude whenever the 0 degree line is crossed. The software checks for this condition which simplifies the input somewhat.

Another point to note is that if data sorting is required, then multiple modes of data selection are possible. If multiple modes are used, then more data cards are needed besides those shown in Table 4-2. This may not be clear in the source listing. Card 3 sets the number of modes and is followed by that number of data cards defining each mode. The observed NRCS measurements will be the same for each mode.

Finally, if the mapping is performed, the user should recognize that the upper and lower data bounds and the number of colors input determine the data incrementing value. This value sets the color boundary values which are ultimately written on the display with one decimal place accuracy. It is recommended that the user check the software beforehand to see if the computed incrementing and color boundary values are appropriate (use EQS. 2-19 to 2-25).

4.2 Generating Annotation

The execution of ANNOT is straightforward and requires only a minimum of user interaction. The software is designed so that a complete label and map legend are developed from just two input cards. The job must be submitted as a remote batch job because it accesses the routine RASTGEN which requires up to 177k octal of core memory. The final display is usually stored on the user's disk as was done in FALSCLR.

The source deck for ANNOT is typically punched on data cards. It is initially compiled and the object module is stored on disk. The control cards to perform this operation is shown in Figure 4.3.

To run the program, the user must access the file ANNOT, the graphics package LRCGOSF, and a special library file FTNMLIB before beginning execution. Following execution of ANNOT, the user must again access the library FTNMLIB, plus the user created routine RASTGEN and a user library called SASLIB. These files are used to creat the display format image file. The job card deck to generate the display is given

```
ANNOT,T50,CM77000.
USER...
CHARGE...
FTN(B=ANNOT,A,R)
SAVE(ANNOT)
EXIT.
-EOR-
        source deck
-EOF-
```

Figure 4.3 Creating the task image of ANNOT.

```
ANNOT,T300,CM177000.
USER...
CHARGE...
GET(ANNOT)
ATTACH(LRCGOSF/UN=LIBRARY)
ATTACH(FTNMLIB/UN=LIBRARY)
ANNOT.
GET(RASTGEN/UN=__________)
GET(SASLIB/UN=___________)
FTN(I=RASTGEN,A,L=0)
ATTACH(FTNMLIB/UN=LIBRARY)
FILE(TAPE15,RT=S,BT=C)
LDSET(FILES=TAPE15,LIB=SASLIB/FTNMLIB)
LGO.
REWIND(TAPE15)
SAVE(TAPE15='user disk file name')
EXIT.
-EOR-
        data cards
-EOF-
```

Figure 4.4 Control deck for execution of ANNOT.

in Figure 4.4. The 'user disk file name' is again any file name the user wishes to call the annotation file. The routines RASTGEN and SASLIB are located on another users disk area. Their user numbers have been omitted by request, but may be obtained by contacting Frances Meissner at the Langley computing facility.

Upon execution of ANNOT, the user must enter two data cards. The first input is a ten character map code used to catalog the final display. The second card sets up the color scale annotation and the map legend. Refer to the program listing in the Appendix for a better description of these two cards.

The display generated by ANNOT is self-contained, meaning that it could by itself be projected onto the digitizing equipment. The display file is somewhat large for an indirect access disk file; however, because a particular file could be used repeatedly for displays having the same annotation, only a few selections of annotation need be stored at any one time. If storage does become a problem, it is recommended that the files be stored on magnetic tape and later copies to disk when needed.

4.3 Execution of MAPGEN

The MAPGEN software has been standardized so that only one input card is necessary to generate a unicolored projection of either the Mercator or Polar Sterographic map with superimposed land/water boundaries. The map projection is scaled before output so that its largest dimension is set to a preset pixel size. Because the program accesses several large routines, it must be run as a remote batch job. Like FALSCLR and ANNOT, the display file that is generated is saved on the user's disk.

The source deck for MAPGEN is usually punched on data cards. For convenience, it is read, then compiled, and the object code is saved on disk. The job deck to do this is given in Figure 4.5.

```
MAPGEN,T50,CM177000.
USER...
CHARGE...
FTN(B=MAPGEN,A,R)
SAVE (MAPGEN)
EXIT.
-EOR-
        source deck
-EOF-
```

Figure 4.5 Creating the task image of MAPGEN.

When executing MAPGEN the user must make available the NCAR map plotting package and the raster conversion routine RASTGEN. the complete job deck to create either of the two map projections is given in figure 4.6. Here again, the 'user disk file name' is any name the user wishes to give the map display file. The omitted user numbers can be obtained by contacting Patricia A. Winters at the Langley computing center. It should be noted that TAPE15 (the display file) is copied in part to a file called TAPE66 which is then saved on disk. In this special usage of RASTGEN, a null file gets appended to TAPE15. This null file is undesirable and is eliminated by the copy statement.

When MAPGEN is executed, one input card is entered which controls all the necessary map functions. This card is explained in the comments of the MAPGEN listing. Briefly, this card defines the map type and geographic region of interest. From this information, a rectangular map image is created whose largest side is scaled to 400 pixels. The rectangular dimension of the map are output to the user so that they may be used as input to FALSCLR to generate the properly sized data map projection. It should again be mentioned that the resolution of the map image is at present only one degree which must be taken into consideration for any map generated by this routine.

4.4 Merging the Files

Routine MERGE is a general purpose display file merging routine. It expects at least one of the two files it merges to have 512 records. The source file is coded on data cards and initially it is compiled and the object code is stored on the user's disk. The job deck to do this is shown in Figure 4.7.

Because routine MERGE needs less than 77K octal core memory to run, the user has the option of running the program from the real time terminals. The option is recommended because it will simplify the manipulation of the many files which the user may wish to merge. Also, since all working files remain local to the user until he logs off, errors can quickly and easily be corrected at one session. The general real time control deck is given in Figure 4.8. In the deck, TAPE15 is

```
MAPGEN,T500,CM177000.
USER...
CHARGE...
GET(CLIB,UN=________)
GET(GLIB,UN=________)
ATTACH(LRCGOSF/UN=LIBRARY)
ATTACH(FTNMLIB/UN=LIBRARY)
ATTACH(TAPE3=MAPDATA/UN=_______)
GET(MAPGEN)
LDSET(LIB=CLIB/CLIB/FTNMLIB/LRCGOSF,PRESETA=NGINF)
MAPGEN.
GET(RASTGEN/UN=_________)
GET(SASLIB/UN=__________)
FTN(I=RASTGEN,A,R,L=0)
ATTACH(FTNMLIB/UN=LIBRARY)
FILE(TAPE15,RT=S,BT=C)
LDSET(FILES=TAPE15,LIB=SASLIB/FTNMLIB)
LGO.
REWIND(TAPE15)
COPY(I=TAPE15,O=TAPE66,TC=EOF,N=1)
REWIND(TAPE66)
SAVE(TAPE66='user disk file name')
EXIT.
-EOR-
       data cards
-EOF-
```

Figure 4.6 Control deck for execution of MAPGEN.

```
MERGE,T50,CM77000.
USER...
CHARGE...
FTN(B=MERGE,A,R,L=0)
SAVE(MERGE)
EXIT.
-EOR-
       source deck
-EOF-
```

Figure 4.7 Creating the task image of MERGE.

```
-log in-
BATCH,77000
GET(MERGE)
GET(TAPE15='#1 user disk file name')
GET(TAPE8='#2 user disk file name')
MAP(OFF)
MERGE.
        enter data
SAVE(TAPE44 ='new user disk file name')
REWIND(MERGE)
-continue as above or log off-
```

Figure 4.8 Terminal Control Commands for Execution of MERGE.

the display file which must have 512 records. TAPE44 is the merged version of TAPE8 and TAPE15, and can be saved on disk or merged further with another display file. Refer to the flow diagram in Figure 3.1.

The only input required by MERGE upon execution is the dimension of the second display file. If this file has 512 records like the first file, a simple one-to-one bit merge is performed on the two files. If smaller, offsets are used to properly center the two files before merging. In either case, the final file has 512 records. Refer to the listing for MERGE in the Appendix.

5.0 THE DIGITAL DISPLAY EQUIPMENT

This section is devoted to a brief discussion of the display equipment and the steps necessary to operate it. The user who wishes to become adept with the system might find this section a useful beginning. Later, only a few hands-on sessions are necessary to become proficient. The equipment is located at the LRC Computing facility and is available to all contractors. Kept with the equipment is a brief manual containing a summary of all the steps and commands used to operate the system.

5.1 Description of the System

The heart of the digital system is the Intel 80/20 microprocessor. It has been programmed to control the various display and tape drive operations. The microprocessor has been periodically reprogrammed whenever new functions are needed. At present, it has no other purpose than to control display generation.

A CRT is used to communicate with the 80/20. Various three character commands can be entered which control tape drive operations, color graphic presentation and photographic equipment operations. The CRT is also used to display system parameters such as the color scales, the image scales and test pattern values.

The display data is presented to the system by magnetic tape using either 800 or 1600 BPI. The tape drive is controlled by a Kennedy Model 9218 format control unit. Upon command, the Kennedy control unit directs the drive to read/write files, rewind or skip several files, and consecutively read/rewind/read ... a file when that file is being photographed.

The display image is visually displayed on a 19" Aydin color television. This television has an optical resolution of approximately 512 by 460 pixels where each pixel corresponds to an eight bit color valve in the tape file. The projected image can be shifted around or out of the viewing screen so that all parts of an image larger than 512 x 460 pixels can be seen.

The display image is photographed by a complex instrument called a Dicomed Image Recorder. At the option of the user, a photograph can be taken of the image on either Polaroid Instant color film or 4 x 5 inch color negatives. The recorder performs this task by exposing the film to a

miniature CRT which displays the color image one pixel at a time. The CRT repeatedly projects the image through a blue, green or red filter to achieve the final color photograph. The recorder has three pixel resolutions given as low - 512 x 512, medium - 1024 x 1024, and high - 2048 x 2048. For SASS display purposes, only low resolution is needed.

5.2 System Operation

The digital equipment is usually powered up and ready for operation the morning of every working day. If an asterisk appears on the CRT, then the system is loaded and ready for use. The first step for the user is to mount and load his nine track tape and set the drive to the proper BPI. Once the tape drive is online the system is ready to receive data and display the image.

There are several checks the user should make before beginning the tape operations. The user should enter the three character commands which initializes (1) the color scale the standard test pattern and (2) the image scale to the standard size and magnification. The standard color scale is a band of colors ranging from blue to green to red to white in 63 discrete steps. This scale can be checked by displaying the system generated test pattern. The image size is initialized to 512 x 512 pixels with a one-to-one magnification.

Once the system is initialized, the user should position his tape at the desired file by entering the proper tape drive commands. The file size in records and eight bit pixels should be known beforehand or can be determined by issuing a file check command. Once this file size is known, the user must reset the standard image size to a value less than or equal to this size. This is needed since the tape drive has no file control to prevent it from overshooting a particular file. The data can now be read and displayed on the color television. It is noted that the SASS display file uses only the lower six bits for color representation.

The colors used in the SASS display are at the lower end of the standard color scale or mainly the blues. This scale will be changed at the CRT by first commanding the presentation of the color scale. The user then must key in the desired integer to color values. These color values are

combinations of percentages of blue, green and red which result in any of 63^3 possible color hues. Some trial and error is needed to obtain an acceptable color scale.

If desired the user can photograph his display by using the imaging recorder. The user must first mount either the Polaroid or negative film holders plus film onto the top of the recorder. Next, the recorder must be set to the proper resolution and functional color scale. Finally, the user must position his tape at the beginning of desired file and then issue the proper three character film making commands. The tape will slowly advance thru the file three times, where each passage extracts only that data containing blue, green or red, respectively. After recording the image, the film is removed and developed. The tape file can now be rewound to a new file or unloaded if the job is complete.

6.0 FUTURE DEVELOPMENTS

This section is devoted to the discussion of future developments in the false color display generation techniques. This report has detailed the software as it stands to date; however, it is fully expected that the program shall be updated as future needs arise. Several possible modifications have already been suggested from the work done, and they will be mentioned here. Also several features will be discussed which were unnecessary or too complex to be included in the present software.

6.1 Data Reduction

As mentioned in section 3.0, the SASS data tapes used by this program were simplified versions of the future tapes to be sent from JPL. These tapes contain only the basic mode of operation parameters and the 15 NRCS measurements. The future SASS tapes will contain a high density of values such as (1) multiple cell coordinates, (2) wind speed calculations, (3) antenna look angles, and any of a multitude of system identification parameters [ref.1]. The software will clearly need to be updated to handle this mass of information.

The primary goal in the reduction of these new tapes will be to create a data subset which has only that information necessary to complete the desired false color display. This reduction operation will be handled in the preprocessing mode of routine FALSCLR and might extract new SASS parameters such as antenna incidence angles, wind speeds, and the cells' three geographic coordinates. The data subset thus created will then be available for any of the various data conditioning functions and ultimately presented to the postprocessing routines in the form of a measured SASS value and its geographic coordinates.

It should be noted that for false color displays exhibiting multiple satellite revolutions, some sort of data averaging will have occurred in the gridding routines in those areas where the revolutions cross each other. In order to control this averaging so that the results are meaningful, some sort of normalization must be performed on the data. To date, however, no formal method of normalizing SASS NCRS data for off-angle measurements has

been established. Since apparently no one particular equation will normalize the data properly over all measured surfaces, an alternate method has been suggested which would apply only to the area under observation. This method normalizes the data by first finding the means of the measurements from each cell over the area of interest and finally subtracting these means from the appropriate NCRS values. This method is being tested and will be considered as a future preprocessing operation.

6.2 Display Generation Changes

There are several changes that will likely be made in the display generation software. Two of the changes arise from the impact of the new SASS data tape. The others are changes that have been suggested from the results already obtained. This new round of improvements is intended to complete the requirement of standardization of the false color display techniques for general use.

The new SASS tapes offer new types and forms of data not applicable to the present software. The most significant of these is the three geographic coordinates given for each cell. These three coordinates correspond to the two cell corners and the cell's center. The cell's single NRCS measurement will be assigned to each of these coordinates. The gridding routine will need to be adapted to handle these three values such that the two cell corner values will contribute less significantly to the final grid matrix in relationship to the center value. The grid weighting coefficients for this operation will need to be determined before this change can be implemented.

Several other data types, such as wind speeds and NCRS standard deviations, might also be presented in a false color display. A few simple changes will be needed in the display annotation and scale magnitudes so that the program is flexible enough to allow any of these data types to be displayed with a minimum of user input. The changes will be implemented as the need arises.

There are two display changes suggested by the results obtained so far. First, the map code generated during the annotation is not flexible enough to allow repeated use of the annotation files with different data sets.

Therefore, a change will be made such that the map code will be generated during the final file merging. This code would be a four digit number which would uniquely define each display.

The second display change would be to print the latitude and longitude values on the map image, something the present software does not do. This improvement will probably have to be accomplished through the use of some of the subroutines available in the SUPMAP Graphics package. The final coordinate format will be determined when it has been ascertained which graphics routines can be used.

Finally, it has been suggested that other map projections be incorporated into the software. The most desirable projections, besides those already in use, would be the Lambert Conformal projection. This map type is ideal for the midlatitude regions [ref.3]. The SUPMAP package contains this projection as one of its nine map types. To make use of this option, a grid-to-Lambert projection routine will be required in the postprocessing mode of FALSCLR. The software for this routine is in the beginning stages and will be tested and implemented upon completion.

7.0 REFERENCES

1. Schroeder, L. C. (editor): "SeaSat-A Scatterometer (SASS) Validation and Experiment Plan," NASA Technical Memorandum 78751, May, 1978.

2. Miller, L. S.: "Investigation of the Applications of GEOS-3 Radar Altimeter Data in Remote Sensing of Land and Sea Features," NASA CR-141428, Applied Science Associates, Inc., Apex, N.C., August, 1977.

3. Faucier, W. J. : Principles of Meteorological Analysis, University of Chicago Press, 1955.

```
      PROGRAM FALSCLR(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE1,
     *                TAPE2,TAPE3,TAPE4,TAPE8,TAPE9,TAPE55)
C
C
C *** THIS PROGRAM WAS CREATED ON APRIL 12, 1979 BY CLAYTON B. JACKSON WHILE ***
C *** UNDER THE EMPLOYMENT OF THE RESEARCH TRIANGLE INSTITUTE OF N.C., IT    ***
C *** WAS DESIGNED TO SATISFY IN PART THE CONTRACT REQUIREMENTS UNDER NASA   ***
C *** CONTRACT NAS1-15338. THE SOFTWARE HAS BEEN REVISED AND TESTED AS OF    ***
C *** JUNE 18, 1979.                                                         ***
C
C
C     THIS PROGRAM IS DESIGNED TO GENERATE FALSE COLOR DISPLAYS OF SEASAT-A
C     SCATTEROMETER DATA WITHIN SELECTABLE GEOGRAPHIC REGIONS. IT IS INTENDED
C     TO RUN ON THE CONTROL DATA'S CYBER 170 SERIES, CYBER 70 SERIES, SERIES
C     6000 AND SERIES 7000 COMPUTERS USING THE FTN5 LANGUAGE. THE PROGRAM
C     UNIT PERFORMS THE FOLLOWING FUNCTIONS:
C
C     1). THE RAW SASS DATA IS SORTED BY GEOGRAPIC LOCATION AND VARIOUS SASS
C         PARAMETERS, AND THE RESULTING DATA IS WRITTEN TO TAPE9. REFER
C         TO ROUTINE DATAIN.
C
C     2). TAPE9 ABOVE IS READ AND THE APPROPRIATE DATA IS SELECTED FOR GRIDDING.
C         THE FINAL GRID MATRIX IS WRITTEN TO TAPE1. REFER TO ROUTINE SSGRID.
C
C     3). THE GRIDDED DATA (TAPE1) IS SMOOTHED BY INTERPOLATION FOR MISSING OR
C         SPARSE DATA. THIS SMOOTHED GRIDDED DATA IS WRITTEN TO TAPE2. REFER TO
C         ROUTINE SMTHGD.
C
C     4). THE GRIDDED DATA (TAPE2) IS PROJECTED ONTO A MERCATOR AND/OR POLAR
C         STEREOGRAPHIC MAP. THE FINAL MAP DATA VALUES ARE INTEGERS
C         REPRESENTING FALSE COLOR SCALING OF THE SASS DATA. THIS MAP DATA IS
C         WRITTEN TO TAPE8. REFER TO ROUTINES MERCAT, POLAR, COLORS AND PACKEM.
C
C     -----------------------
C
C     THE ABOVE 4 FUNTIONS NEED NOT BE RUN CONSECUTIVELY. ANY OF THE FIRST 3
C     STEPS COULD HAVE BEEN RUN PREVIOUS AND THE RESULTS STORED ON DISK OR
C     MAG TAPE. VARIOUS INPUT CARDS ALLOW YOU TO ENTER AND LEAVE THE MAIN
C     PROGRAM AT SEVERAL LOCATIONS.
C
C     -----------------------
C
C
C     ***** PLEASE REFER TO END OF MAIN PROGRAM FOR COMPLETE DESCRIPTION OF ALL
C           THE NECESSARY DATA CARDS *****
C
      DIMENSION GRID(400,20),SUM(400,20),INDEX(401),P(400),SS(400),
     *          IFILL(400)
      COMMON/XMAP/ULT,DLT,ULG,DLG
      MAXSZ=400
      PRINT 3999 , 1
C
C **  PROGRAM CONTROL INPUT
C
      READ(5,*) IGRID,INTERP,IDATA
      IF(IGRID) 135,45,45
   45 PRINT 3999 , 2
C
C **  GRID COORDINATES ENTERED
C
      READ(5,*) ULT,DLT,ULG,DLG,GRDSZ
      IF(DLT.LT.-85.) DLT=-85.0
      IF(ULT.GT.85.) ULT=85.
C
C **  SET GRID MATRIX DIMENSIONS
C
      M=(ULT-DLT)*111.112/GRDSZ+1.
      IF(M.GT.MAXSZ) GO TO 399
      IF(DLG.LT.ULG) GO TO 21
      DLG=DLG-360.
   21 CONTINUE
      N=(ULG-DLG)*111.112/GRDSZ+1.
      IF(N.GT.MAXSZ) GO TO 399
      IF(IDATA) 155,175,155
C
C **  CALL DATA SELECTION ROUTINE
C
  175 CALL DATAIN(IDATA,55)
      IF(IDATA.EQ.0) GO TO 699
  155 REWIND 9
      IF(INTERP.LT.0) GO TO 599
      IF(IGRID.GT.4) GO TO 255
      WRITE(1,302) N,M,ULT,DLT,ULG,DLG
C
C **  PERFORM GRIDDING
C
      CALL SSGRID(N,M,IDATA,GRID,SUM,20)
```

```
      REWIND 1
      IF(INTERP.LT.2) GO TO 55
  255 CONTINUE
      WRITE(2,302) N,M,ULT,DLT,ULG,DLG
      READ(1,302)
C
C **  SMOOTH GRID MATRIX
C
      CALL SMTHGD(INTERP,GRID,N,M,20,SUM)
      REWIND 2
      IF(IGRID.EQ.0) GO TO 499
      GO TO 35
   55 READ(1,302)
C
C **  WRITE FILE TAPE2
C
      WRITE(2,302) N,M,ULT,DLT,ULG,DLG
      DO 36 I=1,M
      READ(1,401) (P(J),J=1,N)
   36 WRITE(2,401) (P(J),J=1,N)
      ENDFILE 2
      ENDFILE 2
      REWIND 2
      IF(IGRID.EQ.0) GO TO 499
      WRITE(6,4003)
      GO TO 35
  135 READ(2,302) N,M,ULT,DLT,ULG,DLG
      REWIND 2
      PRINT 3999 , 7
C
C **  MAP TYPE AND DIMENSIONS ENTERED
C
   35 READ(5,*) N1,N2,N4,ZLAT,MAPTYP
      IF(EOF(5)) 40,25,40
   25 IREC=N2+1
      IF(N1.GT.MAXSZ.OR.N2.GT.MAXSZ) GO TO 199
      IF(MAPTYP.GT.1) GO TO 31
C
C **  PERFORM MECATOR MAP PROJECTION
C
      CALL MERCAT(N,M,SS,N1,N2,INDEX,IFILL,P,IREC)
      ENDFILE 8
      WRITE(6,4001) N1,N2,ULT,DLT,ULG,DLG
      GO TO 40
   31 IF(N4.EQ.0.OR.N4.GT.20) N4=20
      B1=N2
   32 B2=B1/N4
      IF(AINT(B2).EQ.B2) GO TO 33
      N4=N4-1
      GO TO 32
C
C **  PERFORM POLAR STEREOGRAPHIC PROJECTION
C
   33 CALL POLAR(N,M,ZLAT,GRID,N1,INDEX,P,IFILL,N4,IREC)
      ENDFILE 8
      WRITE(6,4002) N1,N2,90.,ZLAT,360.,0.
   40 ENDFILE 8
      REWIND 8
      REWIND 2
      REWIND 9
      STOP
  399 WRITE(6,3000) ULT,DLT,ULG,DLG,GRDSZ
      STOP
  199 WRITE(6,3001) N1,N2,MAXSZ
      STOP
  499 WRITE(6,3002) N,M,ULT,DLT,ULG,DLG,GRDSZ
      REWIND 9
      IF(INTERP.LT.2) GO TO 501
      WRITE(6,3003) INTERP
      STOP
  501 WRITE(6,3004)
      STOP
  599 WRITE(6,5000)
      STOP
  699 WRITE(6,5001)
      STOP
  302 FORMAT(2I3,4F10.4)
  401 FORMAT(10F11.4)
 3000 FORMAT(1H0,50HTHE AREA OF INTEREST IS TOO LARGE OR THE GRID SIZE,
     *15H IS TOO SMALL--,/1X,35HCHANGE THE FOLLOWING TO GET NO MORE,
     *22H THAN A 400 X 400 GRID,//1X,5HNLAT=,F7.2,5X,5HSLAT=,F7.2,5X,
     *7HELONG1=,F8.2,5X,7HELONG2=,F8.2,5X,10HGRID SIZE=,F6.1,4H KM.)
 3001 FORMAT(1H0,10HEITHER N1=,I4,2X,6HOR N2=,I4,2X,12H ARE GREATER,
     *5H THAN,I4,2X,39HREENTER N1,N2-THE FALSE COLOR GRID SIZE)
 3002 FORMAT(26H0THE COMPUTED SASS GRID IS,I4,10H PIXELS BY,I4,
     *51H PIXELS AND IS LOCATED WITHIN THE FOLLOWING REGION:,//1X,10X,
     *5HNLAT=,F7.2,5X,5HSLAT=,F7.2,5X,7HELONG1=,F8.2,5X,7HELONG2=,F8.2,
     *//1X,10X,36HAND THE GRID BOXES ARE APPROXIMATELY,F7.2,1X,
```

```
     *9HKM SQUARE)
 3003 FORMAT(51H0THE GRID WAS SMOOTHED BY INTERPOLATING FOR MISSING,
     *30H DATA USING AN INTERP VALUE OF,I2,2X,17H(REFER TO SMTHGD))
 3004 FORMAT(40H0NO SMOOTHING WAS PERFORMED ON GRID DATA)
 3999 FORMAT(16H0ENTER CARD NO. ,I1)
 4001 FORMAT(51H0A MERCATOR MAP PROJECTION OF SASS DATA WAS CREATED,
     *18H WITH GRID SIZE OF,I4,10H PIXELS BY,I4,7H PIXELS,/1X,6HAND IS,
     *33H LOCATED IN THE FOLLOWING REGION:,//1X,10X,5HNLAT=,F7.2,5X,
     *5HSLAT=,F7.2,5X,7HELONG1=,F8.2,5X,7HELONG2=,F8.2)
 4002 FORMAT(54H0A POLAR STEREOGRAPHIC MAP PROJECTION OF SASS DATA WAS,
     *26H CREATED WITH GRID SIZE OF,I4,10H PIXELS BY,I4,7H PIXELS,/1X,
     *39HAND IS LOCATED IN THE FOLLOWING REGION:,//1X,5HNLAT=,F5.1,5X,
     *5HSLAT=,F7.2,5X,7HELONG1=,F6.1,5X,7HELONG2=,F4.1)
 4003 FORMAT(1H1)
 5000 FORMAT(47H0ONLY ROUTINE DATAIN WAS PERFORMED ON SASS DATA)
 5001 FORMAT(39H0NO DATA WAS FOUND IN AREA OF INTEREST-,
     *17H CHECK DATA CARDS)
C
C
C     THE FOLLOWING IS A COMPLETE SUMMARY OF EACH INPUT CARD---
C
C     CARE MUST BE TAKEN WHEN WRITTING THE INPUT CARDS SINCE THE PROGRAM
C     DOES NOT CHECK FOR IMPROPER INPUTS.
C
C     **  ALL INPUT IS LIST DIRECTED WITH COMMAS USED AS FIELD SEPERATORS  **
C
C     THE CARDS ARE AS FOLLOWS: -----
C
C
C
C     CARD  1. ENTER PARAMETERS IGRID, INTERP AND IDATA WHERE:
C
C                        ( <0 PROCEDE DIRECTLY TO MAPPING
C                  IGRID= ( =0 PERFORM ALL BUT MAPPING
C                        ( >0 <5 PERFORM EVERTHING
C                        ( >4 PERFORM SMOOTHING AND MAPPING ONLY
C
C                        ( <0 PERFORM ONLY DATA SORTING AND SELECTING
C                  INTERP=( =1 PERFORM EVERYTHING EXCEPT SMOOTHING
C                        ( >1 PERFORM EVERYTHING- INTERP SETS AMOUNT OF
C                             SMOOTHING. REFER TO ROUTINE SMTHGD.
C
C                        ( <0 DATA SUBSET DONE EARLIER- UNKNOWN AMOUNT OF DATA
C                  IDATA= ( =0 PERFORM DATA SORTING AND CONTINUE
C                        ( >0 DATA SUBSET ALREADY DONE- IDATA=NO. DATA POINTS
C
C
C     CARD  2. ENTER GRID COORDINATES - NORTH AND SOUTH LATITUDES AND MOST EAST
C              AND LEAST EAST LONGITUDES- ALSO ENTER GRID BOX SIZE IN KM.
C              *** THIS CARD OMITTED IF IGRID IS NEGATIVE ***
C
C
C      ******* OMIT CARDS WITH ASTERISKS IF IDATA IS NONZERO OR IGRID < 0 *****
C
C *** CARD  3. ENTER NMODES- THE NUMBER OF DATA SORTING CONTROL TRIOS.
C              ** IF NMODES EQUAL ZERO(NO SORTING) OMIT CARDS 4 TO 3+NMODES ***
C
C
C *** CARD  4.       ENTER A SASS MODE, BEAM NUMBER AND POLARIZATION ON EACH OF
C *** CARD  5.       THE NMODES INPUT CARDS. ENTERING ZERO FOR ANY PARAMETER
C ***    .   .       CAUSES NO SORTING FOR THAT PARAMETER FOR THAT PARTICULAR
C ***    .   .       INPUT CARD.
C *** CARD 3+NMODES
C
C
C         *** ASSUMING NMODES=1 (FOR EXAMPLE) THE INPUT CARDS CONTINUE AS: ***
C
C *** CARD  5. ENTER CELL NUMBERS TO BE USED IN SORTING-** A BLANK (ALL 0'S)
C              CARD IMPLIES ALL CELLS ARE USED.
C
C
C *** CARD  6. ENTER COORDINATES OF DATA SUBSET- NORTH AND SOUTH LATITUDES AND
C              MOST EAST AND LEAST EAST LONGITUDES.** NEGATIVE LONGITUDES MAY
C              BE USED EVEN THOUGH DATA TAPES DO NOT.**
C
C
C     ****** OMIT ALL FURTHER INPUT CARDS IF IGRID= 0 OR INTERP < 0 ******
C
C     CARD  7. ENTER MAP SIZE IN PIXELS- WIDTH X HEIGHT- ENTER MAP WORKING
C              HEIGHT <=20 PIXELS - ENTER ZLAT, THE SOUTHERN MOST LATITUDE FOR
C              POLAR STEREOGRAPHIC MAPS - AND ENTER MAPTYPE WHERE:
C
C                   MAPTYPE=( <=1 MERCATOR MAP PROJECTION
C                           ( >1  POLAR STEREOGRAPHIC MAP PROJECTION
C
C
C     CARD  8. ENTER FALSE COLOR PARAMETERS ISCALE, ICOLOR, IBOUND, SIGHI,
C              AND SIGLO WHERE:
```

```
C
C                    (1 LINEAR COLOR TO DATA RELATIONSHIP
C          ISCALE=   (2 LOGARITHMIC COLOR TO DATA RELATIONSHIP
C                    (3 EXPONENTIAL COLOR TO DATA RELATIONSHIP
C
C          ICOLOR=   NUMBER OF FALSE COLORS DISPLAYED  <= 20
C
C          IBOUND=   (1 DATA ABOVE AND BELOW LIMITS IS NOT COLORED
C                    (2 DATA ABOVE AND BELOW LIMITS BOTH HAVE A COLOR
C
C          SIGHI =   MAXIMUM SASS DATA VALUE TO BE SCALED
C
C          SIGLO =   MINIMUM SASS DATA VALUE TO BE SCALED
C
      END
```

```
      SUBROUTINE DATAIN(IDATA,IUNIT)
C
C
C     THIS ROUTINE ESTABLISHES A SASS DATA SUBSET AND OUTPUTS IT TO FILE TAPE9.
C     THE PROGRAM USES THE PRESENT TAPE FORMAT AND SCANS THE SASS DATA BY USER
C     SELECTED MULTIPLES OF MODES, BEAM NUMBERS AND POLARIZATIONS FOR ANY OR ALL
C     OF THE ANTENNA'S 15 CELLS. THE GEOGRAPHIC REGION OF THE SUBSET IS SET
C     WITHIN THIS ROUTINE AND DOES NOT NECESSARILY CORRESPOND TO THE GRIDDED
C     AREA. REFER TO MAIN PROGRAM FOR DESCRIPTION OF INPUT CARDS.
C
C
      DIMENSION MODES(15),IBEAMS(15),IPOLS(15),ICELLS(15)
      REAL NLAT
      IDATA=0
      ASSIGN 1 TO IGO
      ASSIGN 9 TO JGO
      ASSIGN 12 TO MGO
      PRINT 3999 , 3
      READ(5,*) NMODES
      IF(NMODES) 20,20,35
C
C **  DATA SELECTORS MODE,BEAM,POLARIZATION AND CELL NOS. INPUTTED
C
   35 DO 130 I=1,NMODES
      PRINT 3999 , 4
  130 READ(5,*) MODES(I),IBEAMS(I),IPOLS(I)
   20 PRINT 3999 , 5
      READ(5,*) (ICELLS(J),J=1,15)
      IF(ICELLS(1).EQ.0) ASSIGN 13 TO MGO
      PRINT 3999 , 6
C
C **  DATA SUBSET COORDINATES INPUTTED
C
      READ(5,*) NLAT,SLAT,ELGHI,ELGLO
      XDLG=ELGLO
C
C **  CHECK FOR NEGATIVE LONGITUDES
C
      IF(ELGLO.LT.0.) XDLG=360.+ELGLO
      IF(ELGLO.LT.0.) ASSIGN 2 TO IGO
      IF(NLAT.GT.100.) ASSIGN 10 TO JGO
      IF(SLAT.GE.NLAT) ASSIGN 10 TO JGO
      IF(ELGLO.LT.ELGHI) GO TO 56
      IF(ELGLO.EQ.0.) GO TO 156
      XDLG=ELGLO
      ELGLO=ELGLO-360.
      ASSIGN 2 TO IGO
      GO TO 56
  156 ASSIGN 10 TO JGO
   56 CONTINUE
C
C **  READ SASS TAPE
C
  150 READ(IUNIT,200) MODE,IBEAM,IPOL,AZVSAT
      IF(EOF(IUNIT)) 30,132,30
  132 IMEAS=0
      IF(NMODES.EQ.0) GO TO 170
C
C **  PERFORM DATA SELECTION
C
      DO 140 I=1,NMODES
      IF(MODES(I)) 4,4,3
    3 IF(MODE.NE.MODES(I)) GO TO 140
    4 IF(IBEAMS(I)) 6,6,5
    5 IF(IBEAM.NE.IBEAMS(I)) GO TO 140
    6 IF(IPOLS(I)) 8,8,7
    7 IF(IPOL.NE.IPOLS(I)) GO TO 140
    8 IMEAS=1
  140 CONTINUE
      IF(IMEAS) 170,33,170
  170 DO 120 I=1,15
      READ(IUNIT,201) ICELL,SIGZ,XLAT,XLONG
      IF(EOF(IUNIT)) 30,133,30
  133 GO TO MGO,(12,13)
   12 IMEAS=0
      DO 180 J=1,15
      IF(ICELLS(J).EQ.0) GO TO 15
      IF(ICELL.NE.ICELLS(J)) GO TO 180
      IMEAS=1
  180 CONTINUE
   15 IF(IMEAS) 13,120,13
   13 GO TO IGO,(1,2)
    2 IF(XLONG.GE.XDLG) XLONG=XLONG-360.
    1 GO TO JGO,(9,10)
    9 IF(XLAT.GT.NLAT.OR.XLAT.LT.SLAT) GO TO 120
      IF(XLONG.GT.ELGHI.OR.XLONG.LT.ELGLO) GO TO 120
C
C **  WRITE FILE TAPE9 CONTAINING DATA SUBSET
```

```
C
  10  WRITE(9,301) SIGZ,XLAT,XLONG
      IDATA=IDATA+1
 120  CONTINUE
      GO TO 150
  33  DO 160 I=1,15
      READ(IUNIT,201)
      IF(EOF(IUNIT)) 30,160,30
 160  CONTINUE
      GO TO 150
  30  ENDFILE 9
      ENDFILE 9
      REWIND 9
C
C **  OUTPUT NO. DATA POINTS
C
      WRITE(6,400) IDATA
      RETURN
 101  FORMAT(I4)
 102  FORMAT(3I3)
 103  FORMAT(15I3)
 104  FORMAT(4F10.0)
 200  FORMAT(3I3,F10.2)
 201  FORMAT(I3,4F8.2)
 301  FORMAT(3F10.4)
 400  FORMAT(31H0** NO. OF GRIDDED DATA POINTS=,I6,3H **)
3999  FORMAT(16HCENTER CARD NO. ,I1)
      END
```

```
      SUBROUTINE SSGRID(N,M,ND,GRID,SUM,N2)
C
C
C     THIS ROUTINE GRIDS THE SASS DATA INTO A GRID MATRIX WHERE THE GRID BOXES
C     HAVE A USER INPUTTED SIZE OF XXX KM. ALL SASS DATA FROM THE SELECTED DATA
C     SET WHICH FALL INTO EACH INDIVIDUAL GRID BOX IS AVERAGED BY INVERSE
C     SQUARE DISTANCE WEIGHTING AND IS STORED IN THE FINAL GRID MATRIX.
C
C
      REAL GRID(N,N2),SUM(N,N2)
      COMMON/XMAP/ULT,DLT,ULG,DLG
      DATA BAD/-88.88/
      MD=ND
      IF(ND.LT.0) MD=50000
      AN=N
      AN2=N2
      MX=0
      ASSIGN 1 TO IGO
      IF(DLG.GE.0.) GO TO 120
      XDLG=DLG+360.
      ASSIGN 2 TO IGO
  120 CONTINUE
C
C **  SET GRID BOX DIMENSIONS
C
      DSEG1=(ULG-DLG)/N
      DSEG2=(ULT-DLT)/M
   51 MX=MX+N2
      REWIND 9
      JUP=N2
C
C **  INTIALIZE GRID MATRIX
C
      DO 30 I=1,N
      DO 30 J=1,N2
      SUM(I,J)=0.
   30 GRID(I,J)=0.
      DLAT=ULT -DSEG2*MX
      ULAT=ULT-DSEG2*(MX-N2)
C
C **  READ DATA FILE TAPE9
C
      DO 10 I=1,MD
      READ(9,100) SIG,XLAT,XLONG
      IF(EOF(9)) 15,25,15
   25 CONTINUE
      IF(SIG.LE.BAD.OR.XLAT.EQ.BAD.OR.XLONG.EQ.BAD) GO TO 10
      GO TO IGO,(1,2)
    2 IF(XLONG.GT.XDLG) XLONG=XLONG-360.
    1 CONTINUE
      IF(XLAT.GT.ULAT.OR.XLAT.LT.DLAT) GO TO 10
      IF(XLONG.GT.ULG.OR.XLONG.LT.DLG) GO TO 10
C
C **  COMPUTE GRID VALUES
C
      B1=(XLONG-DLG)/DSEG1
      B2=(ULAT-XLAT)/DSEG2
      I1=MIN1(AN,B1+1)
      I2=MIN1(AN2,B2+1)
      R2=(B1-I1+.5)**2+(B2-I2+.5)**2
      IF(R2.LT.1.E-8) R2=1.E-8
      GRID(I1,I2)=GRID(I1,I2)+SIG/R2
      SUM(I1,I2)=SUM(I1,I2)+1./R2
   10 CONTINUE
   15 ND=I-1
      DO 20 I=1,N
      DO 20 J=1,N2
      IF(SUM(I,J)) 22,22,26
   26 GRID(I,J)=GRID(I,J)/SUM(I,J)
      GO TO 20
C
C **  FLAG NULL GRID BOXES
C
   22 GRID(I,J)=1.E5
   20 CONTINUE
      IF(MX.GT.M) JUP=N2+M-MX
C
C **  WRITE GRID MATRIX TO FILE TAPE1
C
      DO 70 J=1,JUP
   70 WRITE(1,300) (GRID(I,J),I=1,N)
      IF(MX.GE.M) GO TO 61
      GO TO 51
   61 ENDFILE 1
      ENDFILE 1
  100 FORMAT(3F10.4)
  300 FORMAT(10F11.4)
      RETURN
      END
```

```
      SUBROUTINE SMTHGD(IG,G,N,M,N2,XSUM)
C
C     THIS ROUTINE INTERPOLATES SASS DATA VALUES FOR GRID BOXES WHICH HAVE NO
C     VALUE DUE TO MISSING OR SPARSE DATA. A WEIGHTED AVERAGING IS DONE FROM
C     THE 5 X 5 GRID BOX MATRIX SURROUNDING THE EMPTY GRID BOX. THE WEIGHTS ARE
C     DETERMINED BY THE INVERSE SQUARED DISTANCE FROM THE KNOWN GRID BOX TO THE
C     EMPTY GRID BOX. *** PARAMETER INTERP IS INPUTTED IN THE MAIN PROGRAM AND
C     ITS VALUE DETERMINES THE AMOUNT OF GRID SMOOTHING. BELOW IS A SAMPLE OF A
C     5 X 5 GRID MATRIX WHERE THE CENTER BOX IS TO BE SMOOTHED. THE NUMBERS IN
C     THE BOXES ARE POSSIBLE VALUES OF INTERP. THEREFORE, THE BOXES WITH NUMBERS
C     LESS THAN OR EQUAL TO THE INPUTTED VALUE OF INTERP ARE THE ONES USED IN
C     THE SMOOTHING PROCEDURE. ***
C
C                          ---------------------
C                          | 6 | 5 | 4 | 5 | 6 |
C                          |---|---|---|---|---|
C                          | 5 | 3 | 2 | 3 | 5 |
C                          |---|---|---|---|---|
C                          | 4 | 2 | ? | 2 | 4 |
C                          |---|---|---|---|---|
C                          | 5 | 3 | 2 | 3 | 5 |
C                          |---|---|---|---|---|
C                          | 6 | 5 | 4 | 5 | 6 |
C                          ---------------------
C
      REAL G(N,N2),XSUM(N,N2)
      DATA FLAG/1.E5/
      DATA W1,W2,W3,W4,W5/1.,0.5,0.25,0.2,0.125/
      INULL=1
      IF(IG.GT.3) INULL=2
      N3=INULL*2+1
      ISP=N-INULL
      JSP=M-INULL
      JST=INULL
      IST=INULL+1
C
C**   READ GRID FILE TAPE1
C
      DO 10 I=1,N3
   10 READ(1,100)  (G(K,I),K=1,N)
      DO 57 I=1,INULL
   57 WRITE(2,100)  (G(K,I),K=1,M)
      J=IST
   46 JST=JST+1
      DO 44 I=1,INULL
      XSUM(I,1)=G(I,J)
      IXZ=ISP+I
   44 XSUM(IXZ,1)=G(IXZ,J)
      DO 30 I=IST,ISP
C
C**   COMPUTE SMOOTHED GRID VALUES FOR MISSING DATA ONLY
C
      IF(G(I,J).NE.FLAG) GO TO 28
      DSUM=0.
      TOT=0.
      GO TO (30,31,32,33,34,35),IG
   35 SUM=G(I-2,J-2)+G(I-2,J+2)+G(I+2,J-2)+G(I+2,J+2)
      ITOTO=(SUM+5.E4)/FLAG
      DSUM=DSUM+(SUM-ITOTO*FLAG)*W5
      TOT=TOT+(4.-ITOTO)*W5
   34 SUM=G(I-1,J-2)+G(I+1,J-2)+G(I-2,J-1)+G(I+2,J-1)
     *+G(I-2,J+1)+G(I+2,J+1)+G(I-1,J+2)+G(I+1,J+2)
      ITOTO=(SUM+5.E4)/FLAG
      DSUM=DSUM+(SUM-ITOTO*FLAG)*W4
      TOT=TOT+(8.-ITOTO)*W4
   33 SUM=G(I,J-2)+G(I-2,J)+G(I+2,J)+G(I,J+2)
      ITOTO=(SUM+5.E4)/FLAG
      DSUM=DSUM+(SUM-ITOTO*FLAG)*W3
      TOT=TOT+(4.-ITOTO)*W3
   32 SUM=G(I-1,J-1)+G(I+1,J-1)+G(I-1,J+1)+G(I+1,J+1)
      ITOTO=(SUM+5.E4)/FLAG
      DSUM=DSUM+(SUM-ITOTO*FLAG)*W2
      TOT=TOT+(4.-ITOTO)*W2
   31 SUM=G(I,J-1)+G(I-1,J)+G(I+1,J)+G(I,J+1)
      ITOTO=(SUM+5.E4)/FLAG
      DSUM=DSUM+(SUM-ITOTO*FLAG)*W1
      TOT=TOT+(4.-ITOTO)*W1
      IF(TOT) 29,28,29
   28 XSUM(I,1)=G(I,J)
      GO TO 30
   29 XSUM(I,1)=DSUM/TOT
   30 CONTINUE
C
C**   WRITE SMOOTHED GRID MATRIX TO FILE TAPE2
C
      WRITE(2,100)  (XSUM(II,1),II=1,N)
```

```
45   DO 50 K=2,N3
     DO 50 I=1,N
50   G(I,K-1)=G(I,K)
     READ(1,100) (G(II,N3),II=1,N)
     IF(EOF(1)) 55,46,55
55   N4=N3-1
     DO 60 I=IST,N4
60   WRITE(2,100) (G(II,I),II=1,N)
     ENDFILE 2
     ENDFILE 2
     REWIND 2
     REWIND 1
     RETURN
100  FORMAT(10F11.4)
     END
```

```
      SUBROUTINE MERCAT(N,M,X,N2,N1,INDEX,IFILL,Y,IR)
C
C     THIS ROUTINE PERFORMS A PROJECTION OF SASS GRIDDED DATA ONTO A MERCATOR
C     MAP HAVING THE SAME GEOGRAPHIC COORDINATES. THE USER SUPPLIES THE SIZE OF
C     THE RECTANGULAR MERCATOR MAP MATRIX IN PIXELS UP TO 400 X 400 PIXELS.
C
      DIMENSION INDEX(IR),X(N),Y(N2),IFILL(N2)
      COMMON/XMAP/ULT,DLT,ULG,DLG
      DATA A1,A2,A3/.7853981635,8.7266462609E-3,1.745329252E-2/
      CALL SECOND(TX)
      WRITE(6,500) 1,TX
      READ(2,201)
      CALL OPENMS(3,INDEX,IR,0)
      AX=ALOG(TAN(A1+ULT*A2))
      BX=ALOG(TAN(A1+DLT*A2))
C
C **   PROJECT EQUATOR LOCATION
C
      EQUAT=N1*AX/(AX-BX)
      SCALE=EQUAT / AX
      XLAT=(ULT-DLT) / M
      XINC=1.*N2/N
      JSTR=1
      DO 100 K=1,M
      ISTR=1
C
C **   READ GRID FILE TAPE2
C
      READ(2,101) X
C
C **   PROJECT GRID VALUES TO MAP MATRIX
C
      DO 10 I=1,N
      ISTP=I*XINC+0.5
      IF(ISTP.LT.ISTR) GO TO 10
      DO 20 J=ISTR,ISTP
   20 Y(J)=X(I)
      ISTR=ISTP+1
   10 CONTINUE
      DLAT=ULT-K*XLAT
      CX=ALOG(TAN(A1+DLAT*A2))
      ISTP=EQUAT-SCALE*CX
      IF (ISTP.GT.N1) ISTP=N1
      IF(ISTP.LT.JSTR) GO TO 100
      DO 60 L=JSTR,ISTP
      CALL WRITMS(3,Y,N2,L,0,0)
   60 CONTINUE
      JSTR=ISTP+1
  100 CONTINUE
      IF(ISTP.EQ.N1) GO TO 33
      DO 34 I=ISTP,N1
   34 CALL WRITMS(3,Y,N2,I,0,0)
   33 CALL SECOND(TX)
      CALL SECOND(TX)
      WRITE(6,500) 2,TX
C
C **   INVOKE COLORS ROUTINE
C
      CALL COLORS(N1,N2,Y,IFILL,INDEX,3,IR)
      CALL CLOSMS(3)
      CALL SECOND(TX)
      WRITE(6,500) 3,TX
  101 FORMAT(10F11.4)
  201 FORMAT(2I4,4F10.4)
  500 FORMAT(9HOSTEP NO.,I3,18H IN ROUTINE MERCAT,10H COMPLETED,F8.2,
     *36H SECONDS AFTER START OF MAIN PROGRAM)
      RETURN
      END
```

```
      SUBROUTINE POLAR(N,M,ZLAT,DS,N1,INDEX,P,IFILL,N4,IR)
C
C
C     THIS ROUTINE PERFORMS A PROJECTION OF SASS GRIDDED DATA ONTO A POLAR
C     STEREOGRAPHIC MAP WHOS GEOGRAPHIC CENTER IS THE POLE AND WHOS SOUTHERN
C     MOST LATITUDE IS USER INPUTTED. THE MAP PROJECTION IS A SQUARE MATRIX
C     WHERE USER SUPPLIES THE SIZE IN PIXELS UP TO 400 X 400 PIXELS.
C
C
      DIMENSION INDEX(IR),P(N1),DS(N,N4),IFILL(N1)
      COMMON/XMAP/UPLT,DNLT,ULG,DLG
      DATA A1,A2,A3/.7853981635,8.726646260E-3,1.745329252E-2/
      A4=57.29577951
      FLAG=-88.88
C
C  **  SET MAP TO NORTH OR SOUTH POLE
C
      PLAT=ZLAT
      IF(UPLT.LE.0..AND.DNLT.LT.0.) PLAT=-PLAT
      ULT=AMAX1(ABS(UPLT),ABS(DNLT))
      DLT=AMIN1(ABS(UPLT),ABS(DNLT))
      AN3=N1/2.+0.5
      CALL SECOND(TX)
      WRITE(6,500) 1,TX
      READ(2,601)
      N3=N1/2
      DO 2 I=1,N1
    2 P(I)=FLAG
      CALL OPENMS(4,INDEX,IR,0)
C
C  **  SET GRID DIMENSIONS
C
      XLAT=(ULT-DLT)/M
      XLONG=(ULG-DLG)/N
      SCALE=AN3/TAN(A1-PLAT*A2)
C
C  **  INITIALIZE MAP MATRIX FILE
C
      DO 10 I=1,N1
      CALL WRITMS(4,P,N1,I,0,0)
   10 CONTINUE
      CALL SECOND(TX)
      WRITE(6,500) 2,TX
      HLAT=ULT-90.
      R1=SCALE*TAN(A1-DLT*A2)
      R2=R1
      R3=SCALE*TAN(A1-ULT*A2)
      RZ=R1
      R=R1**2
      IF(DLG.LT.0.) GO TO 11
      IF(ULG.GT.180..AND.DLG.LT.180.) GO TO 41
      R1=-R1*COS(ULG*A3)
   30 R2=R2*COS(DLG*A3)
C
C  **  COMPUTE ACTIVE MAP MATRIX REGION
C
   40 V1=AN3-R1-1.
      V2=AN3+R2+1.
      IV1=MAX1(1.,V1)
      IV2=MIN1(N1*1.,V2)
      CALL SECOND(TX)
      WRITE(6,500) 3,TX
      IF(IV1.LE.IV2) GO TO 20
      IEXC=IV1
      IV1=MAX0(1,IV2-2)
      IV2=MIN0(N1,IEXC+2)
      GO TO 20
   41 IF(ULG.GE.270..OR.DLG.LE.90.) GO TO 21
      R2=R3
      IF((ULG-180.).LT.(180.-DLG)) GO TO 30
      R2=R2*COS(ULG*A3)
      GO TO 40
   21 IF((360.-ULG).GT.DLG) GO TO 30
      R2=R2*COS(ULG*A3)
      GO TO 40
   11 IF(ULG.GE.180..OR.DLG.LE.-180.) GO TO 40
      IF(ULG.LT.90..AND.DLG.GT.-90.) R1=-R3
      IF(ABS(DLG).GT.ULG) GO TO 13
      R1=R1*COS(ULG*A3)
      GO TO 40
   13 R1=R1*ABS(COS(DLG*A3))
      GO TO 40
C
C  **  COMPUTE GRID BOX ARRAY ADDRESSES
C
   20 DO 100 I=IV1,IV2
      XH=0.
      AP=R-(AN3-I)**2
```

```
      IF(AR) 100,103,104
  104 XH=SQRT(AR)
  103 IH1=N3-XH-1
      IH2=N3+XH+1
      CALL READMS(4,P,N1,I)
      DO 200 J=IH1,IH2
      RX=SQRT((AN3-J)**2+(AN3-I)**2)
      IF(RX.GT.RZ) GO TO 200
      I1=(HLAT+2.*A4*ATAN(RX/SCALE))/XLAT+1
      IF(I1) 200,200,59
   59 AI=I
      IF(AI-AN3) 32,33,34
   32 ALONG=ATAN((AN3-J)/(AN3-I))*A4+180.
      GO TO 55
   33 ALONG=90.
      IF(J.LE.N3) ALONG=270.
      GO TO 55
   34 ALONG=ATAN((AN3-J)/(AN3-I))*A4
      IF(ALONG.LT.0.) ALONG=ALONG+360.
   55 I2=(ALONG-DLG)/XLONG+1
      IF(I2.GT.N) GO TO 200
      IF(I2) 200,200,42
C
C  **  PACK GRID BOX ARRAY ADDRESSES INTO EACH MAP MATRIX ELEMENT
C
   42 P(J)=I2*1.E3+I1
  200 CONTINUE
      CALL WRITMS(4,P,N1,I,1,0)
  100 CONTINUE
      CALL SECOND(TX)
      WRITE(6,500) 4,TX
      IZ=1
      IH1=N4
  150 CONTINUE
      IX=IZ+N4-1
      IF(IX.GT.M) IH1=N4+M-IX
C
C  **  READ GRID FILE TAPE2
C
      DO 15 J=1,IH1
   15 READ(2,400) (DS(K,J),K=1,N)
      DO 220 J=IV1,IV2
      CALL READMS(4,P,N1,J)
      DO 230 K=1,N1
      IF(P(K)) 230,230,24
   24 I2=P(K)*1.E-3
      I1=P(K)-I2*1.E3
      IF(I2) 230,230,43
   43 IF(I1.GT.IX.OR.I1.LT.IZ) GO TO 230
      I1=I1+N4-IX
C
C  **  PROJECT PROPER GRID VALUES TO MAP MATRIX
C
      P(K)=DS(I2,I1)
      IF(DS(I2,I1).EQ.1.E5) P(K)=FLAG
  230 CONTINUE
      CALL WRITMS(4,P,N1,J,1,0)
  220 CONTINUE
      IZ=IZ+N4
      IF(IX.GE.M) GO TO 151
      GO TO 150
  151 CONTINUE
      CALL SECOND(TX)
      WRITE(6,500) 5,TX
      DO 160 I=1,N1
      CALL READMS(4,P,N1,I)
      DO 170 J=1,N1
      IF(P(J).GT.100.) GO TO 145
      IF(P(J)-FLAG) 170,145,170
C
C  **  FLAG NULL MAP ELEMENTS
C
  145 P(J)=1.E5
  170 CONTINUE
      CALL WRITMS(4,P,N1,I,1,0)
  160 CONTINUE
      CALL SECOND(TX)
      WRITE(6,500) 6,TX
C
C  **  INVOKE COLORS ROUTINE
C
      CALL COLORS(N1,N1,P,IFILL,INDEX,4,IP)
      CALL CLOSMS(4)
      CALL SECOND(TX)
      WRITE(6,500) 7,TX
  400 FORMAT(10F11.4)
  500 FORMAT(9HSTEP NO.,I3,27H IN ROUTINE POLAR COMPLETED,F8.2,
     *36H SECONDS AFTER START OF MAIN PROGRAM)
  601 FORMAT(2I4,4F10.4)
      RETURN
      END
```

```
      SUBROUTINE COLORS(N1,N2,Y,IF,INDEX,IUNIT,IR)
C
C
C     THIS ROUTINE PERFORMS A USER CONTROLLED CONVERSION OF MAP PROJECTED SASS
C     DATA INTO INTEGER VALUES RANGING FROM 0 TO 21 IN UNIT STEPS. THESE INTEGER
C     VALUES REPRESENT THE FALSE COLOR SCALE AND HAVE A SELECTABLE LINEAR,
C     LOGARITHMIC OR EXPONENTIAL RELATIONSHIP TO THE SASS DATA. USER SUPPLIES
C     TYPE OF SCALE, UPPER AND LOWER DATA VALUES AND NUMBER OF COLORS NEEDED UP
C     TO A MAXIMUM OF 20 COLORS. REFER TO MAIN PROGRAM FOR DESCRIPTION OF INPUT
C     CARDS.
C
C
      INTEGER UPBOUND,DNBOUND
      DIMENSION Y(N2),IF(N2),INDEX(IR)
      PRINT 3999 , 8
C
C **  COLOR SCALING AND DATA BOUNDS INPUTTED
C
      READ(5,*) ISCALE,ICOLOR,IBOUND,SIGHI,SIGLO
      ASSIGN 23 TO UPBOUND
      ASSIGN 23 TO DNBOUND
      IF(IBOUND.NE.2) GO TO 33
      ASSIGN 21 TO UPBOUND
      ASSIGN 22 TO DNBOUND
C
C **  SET MAX NO. OF COLORS
C
      IF(ICOLOR.GT.18) ICOLOR=18
   33 IF(ICOLOR.GT.20) ICOLOR=20
      XCLR=ICOLOR
      ZCLR=XCLR+1.0
      IBNDM1=IBOUND-1
      ITOP=(ICOLOR+IBNDM1*2)
C
C **  COMPUTE DATA INCREMENTING VALUE
C
      IF(ISCALE-2) 1,2,3
    1 DSIG=(SIGHI-SIGLO)/XCLR
      ASSIGN 4 TO JAIL
      GO TO 10
    2 DSIG=XCLR/ALOG(SIGHI-SIGLO+1.)
      ASSIGN 5 TO JAIL
      GO TO 10
    3 DSIG=(SIGHI-SIGLO)/ALOG(XCLR+1.)
      ASSIGN 6 TO JAIL
   10 DO 200 I=1,N1
      CALL READMS(IUNIT,Y,N2,I)
      DO 250 J=1,N2
      IF(Y(J).GE.SIGHI) GO TO UPBOUND,(21,23)
      IF(Y(J).LE.SIGLO) GO TO DNBOUND,(22,23)
C
C **  COMPUTE INTEGER COLOR VALUE
C
      GO TO JAIL,(4,5,6)
C
C **      LINEAR CONVERSION
C
    4 IX=(INT((Y(J)-SIGLO)/DSIG)+IBOUND)
      GO TO 250
C
C **      LOGARITHMIC CONVERSION
C
    5 IX=INT(DSIG*ALOG(Y(J)-SIGLO+1.))+IBOUND
      GO TO 250
C
C **      EXPONENTIAL CONVERSION
C
    6 IX=(INT(EXP((Y(J)-SIGLO)/DSIG))+IBNDM1)
      GO TO 250
C
C **  FLAG HI DATA
C
   21 IX=ITOP
      IF(Y(J).EQ.1.E5) GO TO 23
      GO TO 250
C
C **  FLAG LO DATA
C
   22 IX=1
      GO TO 250
C
C **  FLAG NULL DATA
C
   23 IX=0
  250 IF(J)=IX
C
C **  INVOKE PACK ROUTINE
C
```

```
      CALL PACKEM(IF,N2,N3)
C
C **  WRITE FALSE COLOR VALUES TO FILE TAPE8
C
      BUFFER OUT (8,1) (IF(1),IF(N3))
      IF(UNIT(8)) 200,301,301
  200 CONTINUE
      RETURN
  301 WRITE(6,400)
      RETURN
  400 FORMAT(18H0BUFFER OUT  ERROR,//1X)
 3999 FORMAT(16H0ENTER CARD NO. ,I1)
      END
```

```
      SUBROUTINE PACKEM(IF,N2,N3)
C
C     THIS ROUTINE PACKS THE INTEGER ARRAY IF(N2) SUCH THAT EACH 60 BIT WORD
C     IS REDUCED TO 8 BITS AND THESE 8 BITS ARE PACKED BACK INTO ARRAY IF.
C     THE EFFECTIVE SIZE OF ARRAY IF IS REDUCED TO N3=N2/7.5
C
      INTEGER IF(N2),IB(8)
      IX=0
C
C**   SET NO. OF 60 BIT WORDS NEEDED
      X=N2/7.5+.999
      N3=X
      N4=N3/2+1
      DO 10 I=1,N4
      DO 30 J=1,8
   30 IB(J)=0
      IX=IX+1
      K=(I-1)*15
C
C**   PACK FIRST 7 1/2 (8 BIT) WORDS INTO 60 BIT WORD
      DO 65 L=1,7
      KL=K+L
      IF(KL.GT.N2) GO TO 22
      N=(8-L)*8-4
      IB(L)=IF(KL).AND.377B
   65 IB(L)=SHIFT(IB(L),N)
      IF((KL+1).GT.N2) GO TO 22
      IB(8)=IF(K+8).AND.360B
      IB(8)=SHIFT(IB(8),-4)
      IF(IX)=IB(8).OR.IB(7).OR.IB(6).OR.IB(5).OR.IB(4).OR.IB(3).OR.
     *IB(2).OR.IB(1)
      DO 40 J=1,8
   40 IB(J)=0
      IX=IX+1
      K=K+8
      IB(8)=IF(K).AND.17B
      IB(8)=SHIFT(IB(8),56)
C
C**   PACK  NEXT 7 1/2 (8 BIT) WORDS INTO 60 BIT WORD
      DO 75 L=1,7
      KL=K+L
      IF(KL.GT.N2) GO TO 22
      N=(7-L)*8
      IB(1)=IF(KL).AND.377B
   75 IB(L)=SHIFT(IB(L),N)
   22 IF(IX)=IB(8).OR.IB(7).OR.IB(6).OR.IB(5).OR.IB(4).OR.IB(3).OR.
     *IB(2).OR.IB(1)
   10 CONTINUE
      RETURN
      END
```

```
      PROGRAM ANNOT(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
C
C *** THIS SOFTWARE WAS CREATED ON MAY 16, 1979 BY CLAYTON B. JACKSON WHILE   ***
C *** UNDER THE EMPLOYMENT OF THE RESEARCH TRIANGLE INSTITUTE OF N.C.. IT    ***
C *** WAS DESIGNED TO SATISFY IN PART THE CONTRACT REQUIREMENTS UNDER NASA   ***
C *** CONTRACT NAS1-15338.                                                   ***
C
C
C     THIS PROGRAM CREATES AN ANNOTATION PLOT FILE CONTAINING DISPLAY
C     LABELING AND A MAP LEGION.  THE PLOT FILE IS A SPECIAL RASTER FILE
C     AND IS COMPATIBLE WITH THE DISPLAY FILES GENERATED BY FALSCLR AND
C     MAPGEN.
C
C     THE INPUT CARDS ARE AS FOLLOWS:
C
C     CARD  1. ENTER A 10 CHARACTER MAP CODE ( FORMAT(A10) )
C
C     CARD  2. ENTER ICOLOR,IBOUND,ITYPE,SIGHI,SIGLO,AND XSCALE WHERE:
C
C                    ICOLOR = NUMBER OF DISPLAY COLORS
C
C                    IBOUND = 0 - UPPER AND LOWER BOUND DATA NOT DISPLAYED
C                             1 - UPPER AND LOWER BOUND DATA DISPLAYED
C
C                    ITYPE =  1 - LINEAR COLOR SCALE
C                             2 - LOGARITHMIC COLOR SCALE
C                             3 - EXPONENTIAL COLOR SCALE
C
C                    SIGHI  = UPPER BOUND ON DATA VALUES
C
C                    SIGLO  = LOWER BOUND ON DATA VALUES
C
C                    XSCALE = PLOT DIMENSIONS IN INCHES
C
C                    ***  FORMAT IS LIST DIRECTED  ***
C
      DIMENSION XNUM(21),MC(1)
      CALL PSEUDO
C
C **  ENTER MAP CODE
C
      READ(5,100) MC
  100 FORMAT(A10)
C
C **  ENTER CONTROL VALUES
C
      READ(5,*) ICOLOR,IBOUND,ITYPE,SIGHI,SIGLO,XSCALE
      INX=ICOLOR+1
      IF(IBOUND.EQ.1) INX=INX-2
C
C **  COMPUTE COLOR BOUNDARY VALUES
C
      JNX=INX-1
      XNUM(INX)=SIGHI
      IF(ITYPE-2) 5,6,7
    5 DSIG=(SIGHI-SIGLO)/JNX
      GO TO 8
    6 DSIG=JNX/ALOG(SIGHI-SIGLO+1.)
      GO TO 8
    7 DSIG=(SIGHI-SIGLO)/ALOG(FLOAT(INX))
    8 CONTINUE
      DO 30 J=1,JNX
      GO TO (1,2,3),ITYPE
    1 I=INX-J
      XNUM(I)=SIGHI-J*DSIG
      GO TO 30
    2 XNUM(J)=SIGLO+EXP((J-1)/DSIG)-1.
      GO TO 30
    3 XNUM(J)=SIGLO+DSIG*ALOG(FLOAT(J))
   30 CONTINUE
      SCALE=XSCALE/512.
      CALL CALPLT(SCALE,0,5)
      XRECT= (256.-ICOLOR*12.)*SCALE
      YRECT=15.*SCALE
      RW=24.*SCALE
      RH=18.*SCALE
C
C **  PLOT COLOR RECTANGLES
C
      DO 10 I=1,ICOLOR
      CALL CALPLT(I,4HMASK,4)
      X=XRECT+(I-1)*24.*SCALE
      CALL RECT(X,YRECT,RH,RW,0.0,3)
   10 CONTINUE
      CSIZE=7.*SCALE
      CALL CALPLT(63.0,4)
```

```
      XNOT=XRECT+10.*SCALE
      YNOT=YRECT+4.*SCALE + RH
      XOFSET=3.*SCALE
      IUP=1
      IST=(ICOLOR+1)/2
      IF(IBOUND.EQ.0) GO TO 15
      IUP=0
      IST=(IST/2)*2
C
C **  PLOT COLOR BOUNDARY VALUES
C
   15 DO 30 I=1,IST
      IDX=(I-1)*2+1+IUP
      X=XNOT+(I-1)*48.*SCALE
      X1=X
      IF(ABS(XNUM(IDX)).LT.10.) X1=X1+XOFSET
      IF(XNUM(IDX).GE.0.) X1=X1+XOFSET
      CALL NUMBER(X1,YNOT,CSIZE,XNUM(IDX),0.,1)
   30 CONTINUE
      ISP=INX-IST
      YNOT1=YRECT-4.*SCALE-CSIZE
      XNOT1=XRECT-14.*SCALE
      IF(IBOUND.EQ.1) XNOT1=XNOT1+48.*SCALE
      DO 40 I=1,ISP
      IDX=I*2-IUP
      X=XNOT1+(I-1)*48.*SCALE
      X1=X
      IF(ABS(XNUM(IDX)).LT.10.) X1=X1+XOFSET
      IF(XNUM(IDX).GE.0.) X1=X1+XOFSET
      CALL NUMBER(X1,YNOT1,CSIZE,XNUM(IDX),0.,1)
   40 CONTINUE
      IF(IBOUND.EQ.0) GO TO 55
      XDIS=9.*SCALE
      Y=YNOT1+3.*SCALE
      CALL PARROW(XRECT+XDIS,Y,XRECT-XDIS,Y,1,0.0)
      IF(ISP.GE.IST) Y=YNOT+3.*SCALE
      X=XRECT+ICOLOR*24.*SCALE
      CALL PARROW(X-XDIS,Y,X+XDIS,Y,1,0.0)
   55 CONTINUE
      X=40.*SCALE
      IF(ITYPE.EQ.1) X=56.*SCALE
      Y=55.*SCALE
C
C **  PLOT DISPLAY LABELING
C
      CALL NOTATE(X,Y,CSIZE,30HNORMALIZED RADAR CROSS SECTION,0.,30)
      IF(ITYPE-2) 65,75,85
   65 CALL NOTATE(999.,999.,CSIZE,21H  (LINEAR SCALE (DB)),0.,21)
      GO TO 95
   75 CALL NOTATE(999.,999.,CSIZE,26H  (LOGARITHMIC SCALE (DB)),0.,26)
      GO TO 95
   85 CALL NOTATE(999.,999.,CSIZE,26H  (EXPONENTIAL SCALE (DB)),0.,26)
   95 CONTINUE
      CALL NOTATE(999.,999.,CSIZE,5H  MC=,0.,5)
      CALL NOTATE(999.,999.,CSIZE,MC,0.,10)
      CALL CALPLT(0.,0.,999)
      STOP
      END
```

```
      PROGRAM MAPGEN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE3,TAPE8)
C
C
C *** THIS SOFTWARE WAS CREATED ON JUNE 5, 1979 BY CLAYTON B. JACKSON WHILE   ***
C *** UNDER THE EMPLOYMENT OF THE RESEARCH TRIANGLE INSTITUTE OF N.C.. IT    ***
C *** WAS DESIGNED TO SATISFY IN PART THE CONTRACT REQUIREMENTS UNDER NASA   ***
C *** CONTRACT NAS1-15338.                                                   ***
C
C
C     THIS PROGRAM GENERATES EITHER A MERCATOR OR POLAR STEREOGRAPHIC MAP PLOT
C     WHICH CONTAINS LATITUDE AND LONGITUDE LINES, KEY MERIDEANS AND POLES, AND
C     SUPERIMPOSED LAND/WATER BOUNDARIES. THE PLOT OFFSETS AND SCALE FACTORS
C     ARE CALCULATED SUCH THAT THE PLOT FILE IS DIRECTLY COMPATIBLE WITH THE
C     DISPLAY FILE CREATED BY PROGRAM FALSCLR.
C
C
C
C     THE ONE INPUT CARD IS DESCRIBED AS FOLLOWS:
C
C     ENTER ZSCALE, MAPTYPE, ULT, DLT, ULG, AND DLG WHERE:
C
C               ZSCALE = UNSCALED PLOT SIZE (IMAGE SIZE IN INCHES)
C
C               MAPTYPE = 1 - MERCATOR MAP
C                         2 - POLAR STEREOGRAPHIC
C
C               ULT =  NORTHERN MOST LATITUDE (MAPTYPE=1)
C                   =  OUTERMOST LATITUDE (MAPTYPE=2)
C
C               DLT =  SOUTHERN MOST LATITUDE   (=0 IF MAPTYPE=2)
C
C               ULG =  EASTERN MOST LONGITUDE   (=0 IF MAPTYPE=2)
C
C               DLG =  EASTERN LEAST LONGITUDE  (=0 IF MAPTYPE=2)
C
C
C          ** INPUT CARD USES LIST DIRECTED FORMAT **
C
C
      DATA A1,A2,A3/.7853981635,8.7266462609E-3,57.29577951/
C
C **  ENTER CONTROL VALUES
C
      READ(5,*) ZSCALE,MAPTYP,ULT,DLT,ULG,DLG
      IF(MAPTYP.GT.1) GO TO 40
C
C **  FIND CENTER LATITUDE
C
      CLAT=2.*A3*ATAN(SQRT(TAN(A1+ULT*A2)*TAN(A1+DLT*A2)))-90.
C
C **  SET ANGULAR LIMITS
C
      P1=ABS(ULG-DLG)*0.5
      P2=P1
      P3=ABS(CLAT-DLT)
      P4=ABS(ULT-CLAT)
      PG=ULG-P1
C
C **  PLOT MERCATOR
C
      CALL SUPMAP(9,CLAT,PG,0.,P1,P2,P3,P4,-4,10,-0,0,IER)
      IF(IER.NE.0) GO TO 50
C
C **  COMPUTE MAP IMAGE PIXEL SIZE
C
      DU=4.*A2*P2
      DV=ALOG(TAN(A1+P4*A2))+ALOG(TAN(A1+P3*A2))
      IF(DU.GT.DV) GO TO 41
      M=400
      N=400.*DU/DV + 0.5
      GO TO 61
   41 N=400
      M=400.*DV/DU + 0.5
   61 WRITE(6,5000) ULT,DLT,ULG,DLG,N,M
      GO TO 60
C
C **  SET CENTER LATITUDE TO NORTH OR SOUTH POLE
C
   40 CLAT=89.99999
      IF(DLT.LT.0..AND.ULT.LE.0.) CLAT=-89.99999
      P1=ABS(CLAT-ULT)
C
C **  PLOT POLAR STEREOGRAPHIC
C
      CALL SUPMAP(1,CLAT,0.,0.,P1,P1,P1,P1,-4,10,-0,0,IER)
      IF(IER.NE.0) GO TO 50
      N=400
      M=400
```

```
      WRITE(6,5001)  N,M
   60 CALL FRAME
C
C **  COMPUTE OFFSETS AND SCALE FACTORS
C
      SCALE=ZSCALE/512.
      XSCALE =AMAX1(10.*N/4608.,10.*M/4608.)
      YSCALE=XSCALE
      XOFF=0.5*(512-N)*SCALE-0.5*XSCALE
      YOFF=100.*SCALE-0.5*YSCALE
C
C **  CALL PLOT TRANSLATOR
C
      CALL METRST(XOFF,YOFF,XSCALE,YSCALE,SCALE)
      STOP
   50 WRITE(6,100) IER
      CALL SYSTEM(52,12HSUPMAP ERROR)
      STOP
  100 FORMAT(5H0IER=,I5)
 5000 FORMAT(54H0A MERCATOR MAP PLOT WAS GENERATED IN THE REGION OF --,
     */6X,5HNLAT=,F6.2,5X,5HSLAT=,F6.2,5X,7HELONG1=,F7.2,5X,7HELONG2=,
     *F7.2,//6X,10HTHE MAP IS,I4,15H PIXELS WIDE BY,I4,7H PIXELS,
     *5H LONG)
 5001 FORMAT(48H0A POLAR STEREOGRAPHIC MAP WAS GENERATED THAT IS,I4,
     *15H PIXELS WIDE BY,I4,12H PIXELS LONG)
      END
```

```
      PROGRAM MERGE(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE8,
     *              TAPE15,TAPE44)
C
C
C *** THIS SOFTWARE WAS CREATED ON MAY 14, 1979 BY CLAYTON B. JACKSON WHILE  ***
C *** UNDER THE EMPLOYMENT OF THE RESEARCH TRIANGLE INSTITUTE OF N.C.. IT    ***
C *** WAS DESIGNED TO SATISFY IN PART THE CONTRACT REQUIREMENTS UNDER NASA   ***
C *** CONTRACT NAS1-15338.                                                   ***
C
C
C
C     THIS PROGRAM IS DESIGNED TO MERGE TWO RASTER DISPLAY FILES BY PERFORMING
C     A BIT BY BIT OR ON EVERY COMMON BIT. ONE FILE MUST HAVE 512 RECORDS, WHILE
C     THE OTHER FILE SIZE IS USER INPUTTED. THE RESULTING FILE IS A RASTER
C     DISPLAY FILE WHICH IS THE SUPERIMPOSITION OF THE TWO INPUT FILES.
C
C
C     THE ONE INPUT CARD IS AS FOLLOWS:
C
C         ENTER N AND M WHERE:
C
C              N = HORIZONTAL PIXEL SIZE OF FILE 2 (NO. OF 8 BIT WORDS)
C
C              M = VERTICAL PIXEL SIZE OF FILE 2 (NO. OF RECORDS)
C
C         **  LIST DIRECTED FORMAT **
C
C
C
      INTEGER BUFF1(100),BUFF2(100)
      NWDS=100
      PRINT 100
C
C **  ENTER 2ND FILE SIZE
C
      READ(5,*) N,M
      IF(N.EQ.512.AND.M.EQ.512) GO TO 79
C
C **  COMPUTE BIT OFFSETS
C
      IXOFST=(512-N)/2 - 4
      IYOFST=412-M
      IXBITS=IXOFST*8
      IZBITS=IXBITS-(IXBITS/60)*60
      IYBITS=60-IZBITS
      IXWRD=IXBITS/60+1
C
C **  SET MASKING VARIABLES
C
      XMASK=.NOT.MASK(IYBITS)
      YMASK=.NOT.MASK(IZBITS)
      ZMASK=MASK(IZBITS)
      IF(512-N-IXOFST) 99,50,50
   50 IF(IYOFST.LT.0) GO TO 199
      IF(IYOFST.EQ.0) GO TO 15
C
C **  PERFORM Y OFFSET
C
      DO 10 I=1,IYOFST
      BUFFER IN (15,1) (BUFF2(1),BUFF2(NWDS))
      IF(UNIT(15)) 20,299,299
   20 NWDS=LENGTH(15)
      BUFFER OUT (44,1) (BUFF2(1),BUFF2(NWDS))
      IF(UNIT(44)) 10,399,399
   10 CONTINUE
   15 IDUMMY=0
C
C **  PERFORM MERGING - FILE SIZES UNEQUAL
C
      DO 150 I=1,M
      BUFFER IN (15,1) (BUFF2(1),BUFF2(NWDS))
      IF(UNIT(15)) 60,299,299
   60 NWDS=LENGTH(15)
      BUFFER IN (8,1) (BUFF1(1),BUFF1(NWDS))
      IF(UNIT(8)) 70,299,299
   70 MWDS=LENGTH(8)-1
      IDUMMY=SHIFT(BUFF1(1),-IZBITS).A.YMASK
      BUFF2(IXWRD)=IDUMMY.OR.BUFF2(IXWRD)
      DO 140 J=1,MWDS
C
C **  PERFORM X OFFSET
C
      ITEMP1=BUFF1(J).A.XMASK
      ITEMP1=SHIFT(ITEMP1,IYBITS).A.ZMASK
      ITEMP2=SHIFT(BUFF1(J+1),-IZBITS).A.YMASK
      IDUMMY=ITEMP1.OR.ITEMP2
C
C **  PERFORM THE BIT BY BIT OR
```

```
C
      BUFF2(IXWRD+J)=BUFF2(IXWRD+J).OR.IDUMMY
  140 CONTINUE
      J=MWDS+1
      IF(IZBITS.EQ.0) GO TO 144
      ITEMP1=BUFF1(J).A.XMASK
      IDUMMY=SHIFT(ITEMP1,IYBITS).A.ZMASK
      BUFF2(IXWRD+J)=BUFF2(IXWRD+J).OR.IDUMMY
  144 BUFFER OUT (44,1) (BUFF2(1),BUFF2(NWDS))
      IF(UNIT(44)) 150,399,399
  150 CONTINUE
C
C **  COMPLETE OUTPUT FILE
C
   34 BUFFER IN (15,1) (BUFF2(1),BUFF2(NWDS))
      IF(UNIT(15)) 130,230,299
  130 BUFFER OUT (44,1) (BUFF2(1),BUFF2(NWDS))
      IF(UNIT(44)) 34,399,399
  230 ENDFILE 44
      ENDFILE 44
      REWIND 44
      REWIND 15
      REWIND 8
      STOP
C
C **  PERFORM MERGING - FILE SIZES EQUAL
C
   79 DO 250 I=1,512
      BUFFER IN (15,1) (BUFF2(1),BUFF2(NWDS))
      IF(UNIT(15)) 330,299,299
  330 NWDS=LENGTH(15)
      BUFFER IN (8,1) (BUFF1(1),BUFF1(NWDS))
      IF(UNIT(8)) 340,299,299
  340 DO 260 J=1,NWDS
  260 BUFF2(J)=BUFF2(J).OR.BUFF1(J)
      BUFFER OUT (44,1) (BUFF2(1),BUFF2(NWDS))
      IF(UNIT(44)) 250,399,399
  250 CONTINUE
      GO TO 230
   99 WRITE(6,200) IXOFST
      STOP
  199 WRITE(6,300) M
      STOP
  299 WRITE(6,400)
      STOP
  399 WRITE(6,500)
      STOP
  100 FORMAT(29H0ENTER N,M THE MAP PIXEL SIZE)
  200 FORMAT(1H0,35HX-OFFSET IS SUCH THAT DATA WILL NOT,
     *18H FIT IN PLOT FRAME,5X,7HIXOFST=,I4,7H PIXELS)
  300 FORMAT(31H0Y-OFFSET IS LESS THAN ZERO--M=,I4,12H IS TO LARGE)
  400 FORMAT(25H0** ERROR ON BUFFER IN **)
  500 FORMAT(26H0** ERROR ON BUFFER OUT **)
      END
```