Final Report

Digital Information System for the Oruro Department, Bolivia (ATN/SF-1812-BO)

August 1982

Principal Investigators: Luis A. Bartolucci and Terry L. Phillips
Time Period: August 1, 1980 - July 28, 1982
Submitted to: Programa ERTS/Bolivia
GEOBOL
Casilla 2729
La Paz Bolivia

LARS Contract Report 080182
Laboratory for Applications of Remote Sensing
Purdue University
West Lafayette, Indiana 47906
USA

FINAL REPORT

Period: August 1, 1980 - July 28, 1982

Project: Digital Information System for the Oruro Department, Bolivia (ATN/SF-1812-BO)

Principal Investigators: Luis A. Bartolucci and
Terry Phillips
Purdue University/LARS
1220 Potter Drive
West Lafayette, Indiana 47906
USA

Submitted to: Programa ERTS/Bolivia
GEOBOL
Casilla 2729
La Paz, Bolivia

August 1982

Laboratory for Applications of Remote Sensing
Purdue University
West Lafayette, Indiana 47906
USA

TABLE OF CONTENTS

ACKNOWLEDGMENTS

A simple review of this project will quickly show that an investigation of this magnitude could not be accomplished successfully by one individual or even a few persons; hence, this final report is the culmination of the cooperative effort and interest of many colleagues from LARS and other offices of Purdue University. It would be impossible to thank everyone who made a contribution to this investigation.

However, the principal investigators wish to acknowlege particularly the essential input provided by Ing. Carlos Valenzuela, Visiting Scientist from Bolivia, Dr. Carlos E. Brockmann, Mr. Gary M. Brammer, Mr. David L. Snyder, Ms. S. Kay Hunt, Mr. Joel Kessler from LARS, Ing. Percy Grundy and Ing. Leonardo Prudencio from ERTS/GEOBOL, and Ing. Pierre M. Adrien, soil scientist from the InterAmerican Development Bank without whose vital contributions this project would not have been possible.

Obviously all the quarterly progress and final reports required countless hours of typing, duplicating, collating, etc. which was accomplished diligently through the combined efforts of many of the clerical staff of LARS. This important effort was led and coordinated so well by Mrs. Darlys C. McDonald, whose dedication to this important project was exemplary.

Luis A. Bartolucci
Terry L. Phillips
Principal Investigators

EXECUTIVE SUMMARY

Essentially this project consisted of the conceptualization, and design of a digital Geographic Information System (GIS) for the entire territory of the Republic of Bolivia, and the development and implementation of this system for the Oruro Department.

A simplified schematic configuration of the Bolivian GIS is illustrated in Figure A, and the natural resources, environmental and socio-economic elements of the geocoded image plane data base are shown in Figure B.

The most significant achievements derived from this project include:

- The design of a digital Geographic Information System of national scope.
- A thorough investigation and subsequent development of an algorithm that defines the optimum resource map projection for Bolivia, i.e. the ALBERS conical equal-area projection.
- Development of hierarchical classification schemes (legends) for the various thematic elements of the Bolivian GIS (classification coding).
- Development and implementation of an addressing scheme for storing georeferenced data in a digital image format.
- Development of a method for storing large quantities of data for natural resources inventories that allows managers, planners and decision makers to obtain useful information in an interactive mode at national, regional (Departmental) and local levels.

- The design of this system provides the capability to store, manage, analyze and update effectively the country's natural resources, environmental and socio-economic data.
- The design of digitizing software for implementation on a microprocessor (APPLE II plus microprocessor).
- Creation of a digital Landsat MSS mosaic for the Oruro Department complete with a quantitative planimetric accuracy assessment.
- Development of computer programs for the conversion of topographic maps into Digital Terrain Models (DTM).
- Definition of procedures to determine quantitatively the mapping and tabulation errors resulting from digitization of resource maps.
- Determination of an appropriate "cell-aggregation" method for the Landsat MSS mosaic data.

The principal investigators of this project believe that the most important achievement derived from this cooperative effort between the Bolivian ERTS/GEOBOL Program and Purdue/LARS has been the effective transfer of the technology through the training of Bolivian technical personnel.

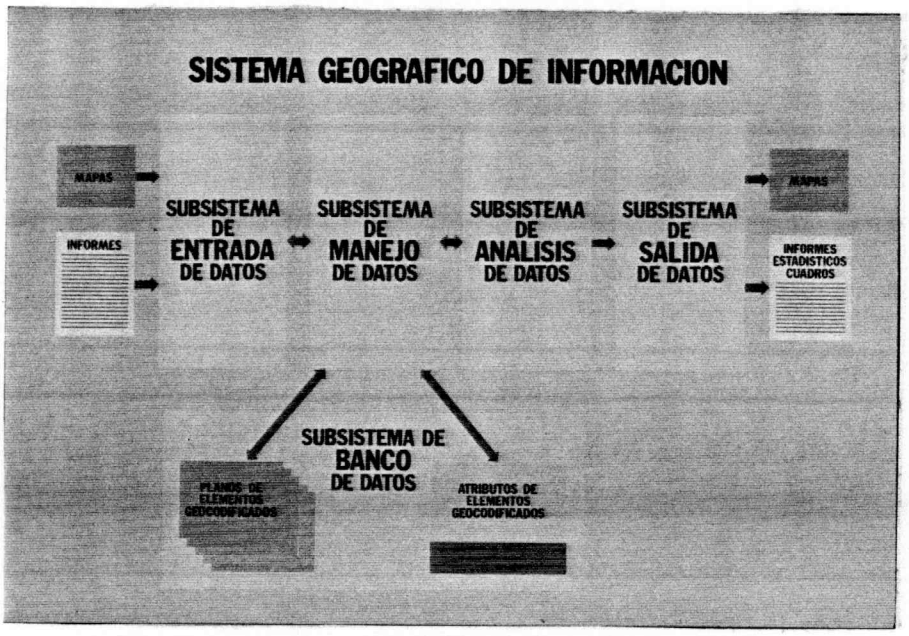


Figure A. Schematic configuration of the Bolivian GIS.

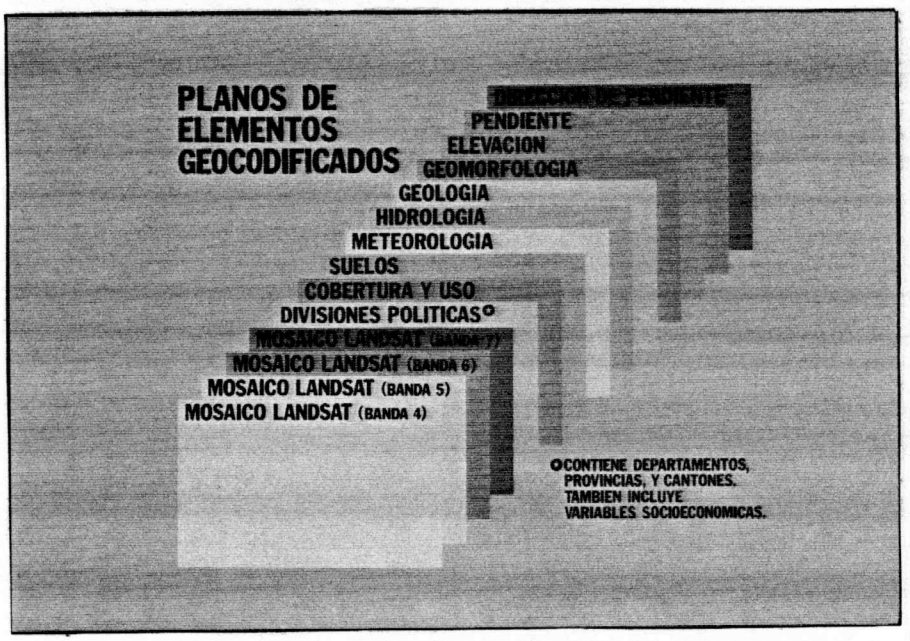


Figure B. Elements included in the geocoded image plane data base of the Oruro Department GIS.

Final Report

DIGITAL GEOGRAPHIC INFORMATION SYSTEM FOR THE ORURO DEPARTMENT, BOLIVIA (Project ATN/SF-1812-BO)

INTRODUCTION

This final report summarizes the results obtained during the period of performance of this project, i.e., August 1, 1980 through July 28, 1982. The detailed technical accomplishments related to this project have been documented in a series of Quarterly Progress Reports which have been already delivered to the Bolivian ERTS/GEOBOL Program. These reports include the following LARS publications:

LARS Contract Report 110180, November 1980, 117 pages.

LARS Contract Report 020181, February 1981, 125 pages.

LARS Contract Report 050281, May 1981, 38 pages.

LARS Contract Report 080181, August 1981, 76 pages (Part 1).

LARS Contract Report 080281, August 1981, 349 pages (Part 2).

LARS Contract Report 110181, November 1981, 62 pages.

LARS Contract Report 020282, February 1982, 201 pages.

In addition, a technical report prepared by the Jet Propulsion Laboratory (JPL) documenting the creation of the Landsat digital mosaic entitled "The Oruro, Bolivia Mosaic" authored by A.L. Zobrist, N.A. Bryant, and R.G. McLeod (204 pages) was also delivered to the Bolivian ERTS/GEOBOL Program.

This report also contains the documentation of 1) the latest software developed for the Bolivian GIS Input Subsystem, i.e. the topographic data input programs, 2) the Bolivian data base query capabilities, 3) the

conversion of all the software that was originally developed and implemented for a PDP 11 minicomputer into an Input software package for an APPLE II Plus microcomputer*, and 4) all the products that are to be delivered to the Bolivian ERTS/GEOBOL Program.

This project was funded by the Interamerican Development Bank (IDB) through an agreement for technical cooperation between the IDB and the Bolivian Geological Survey (GEOBOL's ERTS Program).

INPUT OF TOPOGRAPHIC DATA

The INPUT SUBSYSTEM developed for the Bolivian GIS was described in detail in the August 1981 progress reports (LARS Contract Reports 080181 and 080281). This subsystem was able to capture three types of data sets through the digitization process, i.e. centroids, arcs, and checkpoints. In addition to these three types of data, currently, the input subsystem has a capability to handle contoured topographic data. In other words, topographic maps can be digitized and converted into elevation, slope and aspect (azimuth) geocoded digital planes. The development and implementation of this part of the GIS Input Subsystem was carried out by Dr. Minoru Akiyama.

---

* This task was not contemplated in the original budget and project implementation plan. However, since the Bolivian ERTS/GEOBOL Program was able to purchase only an APPLE II Plus microcomputer instead of a PDP 11/34 minicomputer, the "extra task" of converting all the Input programs was carried out.

The digitization of terrain information (topographic maps) is somewhat different from the process of digitizing resource maps, such as land use or soils maps, in that the former requires interpolation of elevation values and estimation of slope and aspect, while the rasterization of other elements requires only filling-in (painting) and labeling the areas within the polygons that delineate the different classes of the resource map. In order to accomplish the interpolation of elevation and calculation of the slope and aspect for every pixel in the digital geocoded image plane, a mathematical model (algorithm) had to be developed. The algorithm developed and implemented as part of the Bolivian GIS Input Subsystem is described in the following section of this report.

The Terrain Information System Algorithm. A Terrain Information System or Digital Terrain Model (DTM) consists of elevation, slope and aspect data that is obtained from digitized contour lines. The input data for this system is digitized contour data and optionally a sequence of point elevation data along the frame line of the area being digitized.

In this section, the algorithm of the system is described. Although the entire system is composed of three steps, 1) automatic arc closure, 2) projection transformation, and 3) rasterization, the main step is the rasterization step.

In the rasterization routine, terrain information at each pixel center are calculated from contour line data which have been editted and transformed to Albers coordinates through the automatic arc closure and projection transformation steps. Figure 1 shows an example of input

contours, and Figure 2 illustrates the output grid corresponding to the input showed in Figure 1. Elevation, slope and aspect at each of those grid points are calculated one by one using the rasterization routine.

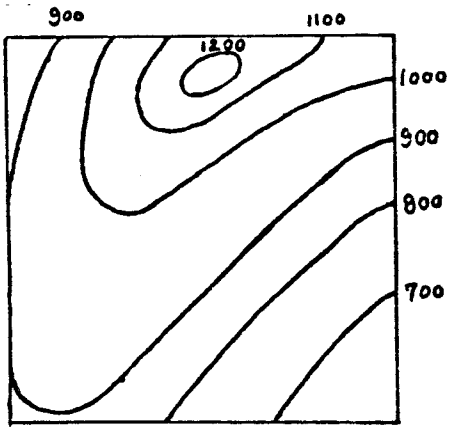


Figure 1. Input contours.

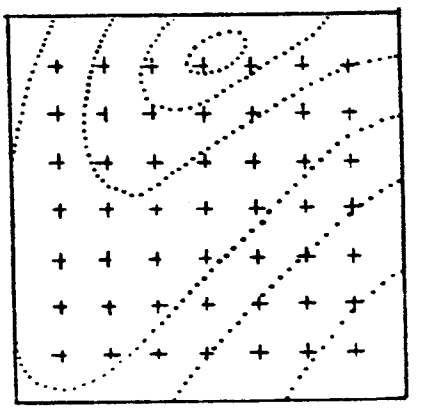


Figure 2. Output grids.

First, contour lines are transformed to four sets of intersections of a contour line and a straight line parallel to the x, y and diagonal axes, as shown in Figures 3 through 6.

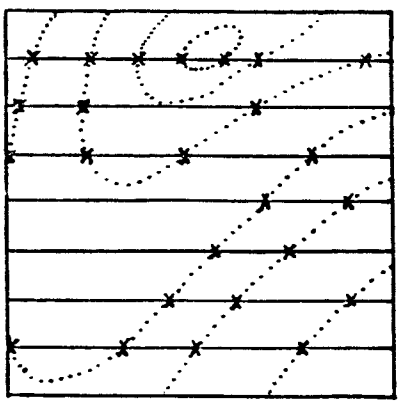


Figure 3. Intersections of contours and x-parallel lines.

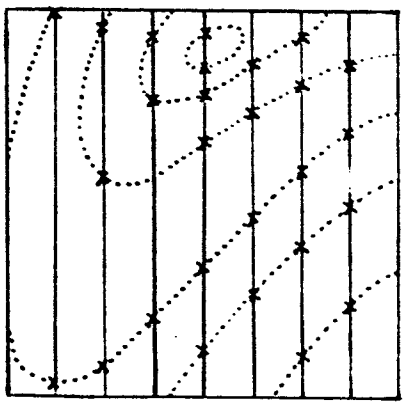


Figure 4. Intersections of contours and y-parallel lines.

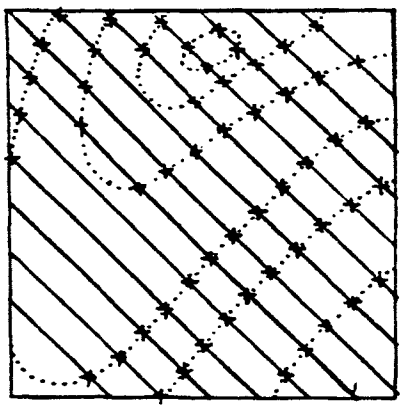


Figure 5. Intersections of contours and NW-SE diagonal lines.

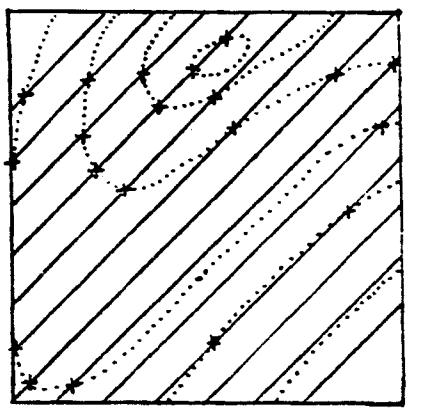


Figure 6. Intersections of contours and NE-SW diagonal lines.

In addition to these data points, this system allows boundary point data (as shown in Figure 7) to be obtained from the maps. If the boundary data exists, elevations at the intersections of the frame line of the digitized area and the parallel straight lines of four directions are interpolated using these boundary point data.

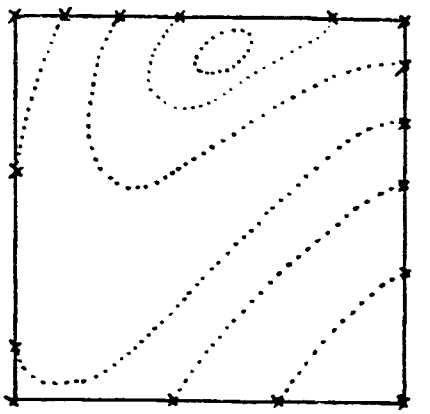


Figure 7. Boundary data.

Although the intersections are obtained in order of input contour lines, they should be rearranged for convenience in further calculations. Intersections of x-parallel lines (Figure 3) are sorted from

north to south and on the same line from west to east. Y-parallel intersections (Figure 4) are sorted from west to east and north to south. Intersections of diagonal lines as shown in Figure 5 are sorted from northeast to southwest and northwest to southeast. And in the case of Figure 6, they are sorted from northwest to southeast and northeast to southwest.

The order of x-parallel intersections matches the order of the terrain information calculation. However, in other files, order of intersections after sorting does not match the order of calculation. For example, in the case of y-parallel intersections, each line corresponds to the column of the output grid. Therefore, one should begin with the first line and proceed to the last line to process one line of the output grid. Then one should go back to the first line again and again in order to process the subsequent lines of the output grid. For this reason, y-parallel and two diagonal line intersection files are stored as direct access files, while x-parallel line intersections are stored as a sequential access file.

After rearranging the intersection files, the terrain information is calculated pixel by pixel. To calculate the terrain information of the center pixel of the sample area (as shown in Figure 8) the intersections along the four lines passing through the center point are determined.

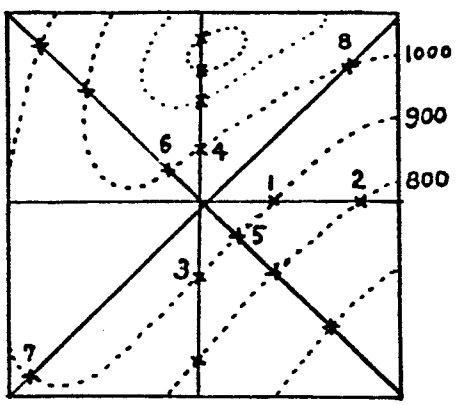


Figure 8. Terrain information calculation.

Then the search of points is performed in such a manner that the point which first exceeds the center point is chosen as the first point and then one goes backwards to let the point first encountered not to be the second point. If the z values of those two points are different, the search will be terminated along the line. If the z values are identical, the search continues forward and backward alternatively until a point with different z value is encountered. If there are not any more points in the forward or backward directions, the search is performed on one side only.

In the case of Figure 8, along the x-parallel line two points on the right (1 and 2) are chosen since there are no points on the left of the center point and the z values of them are not equal. For other lines, one point for each side of the center point is chosen as indicated by numbers 3 through 8 in Figure 8.

The elevation interpolation takes place along one of the four lines. To determine the line to be adopted, the conditions of the distribution of points are evaluated for each line. The evaluation is performed by the following formulas:

$$G(I)=G1(I)+G2(I)+G3(I)+5.0 \qquad (1)$$

$$G1(I)=1.0/(NX(I)-1.9) \qquad (2)$$

$$G2(I)=200.0/(AX1(I)+50.0) \qquad (3)$$

$$G3(I)=200.0/(AX2(I)+50.0) \qquad (4)$$

where NX(I) is the number of points, AX1(I) and AX2(I) are distances from the object point to the first and the second point, respectively, and I=1,2,3 and 4 indicates each of the four lines. In addition, if the first and the second points are located on the same side of the object point,

G(I) is defined as:

G(I)=G(I)-13.0 (if NX(I) = 2) (5)

G(I)=G(I)-4.0 (if NX(I) > 2) (6)

The optimum line for the elevation interpolation is determined as that of the maximum G(I) among the four lines.

The interpolation is performed by means of a polynomial approximation. If the first two points surrounding the object point have different z values, a simple linear interpolation is adopted, which is supposed to be more accurate than a higher degree polynomial interpolation. The relationship between the number of points and G(I) is as follows:

G1(I)=10.0 when NX(I)=2

G1(I)=0.9 when NX(I)=3

G1(I)=0.48 when NX(I)=4

G1(I)=0.32 when NX(I)=5

However, if the object point is not in between two points, the G(I) value decreases a great deal.

When the number of points is identical, G(I) attains a higher value as the distance from the object point to the first two points is smaller. Figure 9 illustrates this relationship. Usually, 40.0 of AX(I) corresponds to 1mm distance on a map.

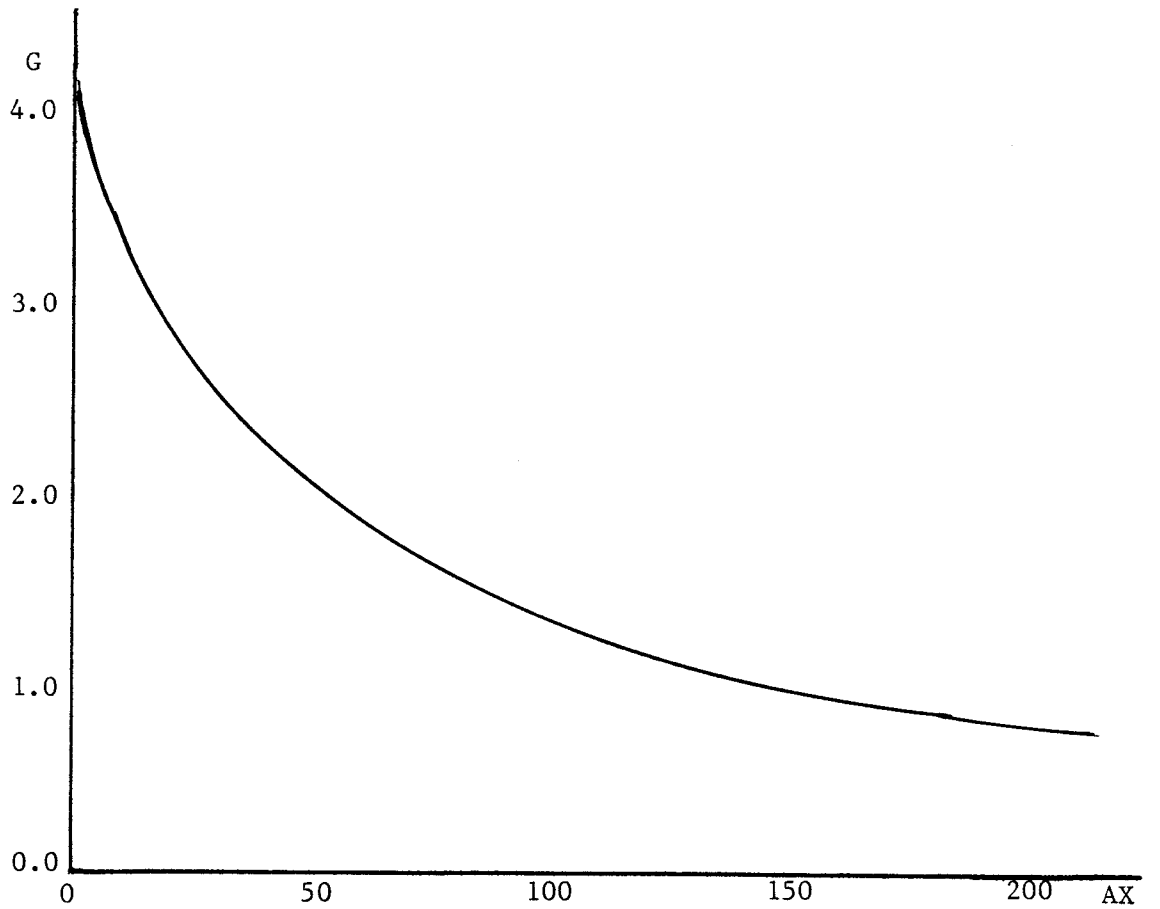


Figure 9. Relation between AX(I) and G2(I).

Some examples of the evaluation are given here so that one can get some idea as to how each situation will be evaluated.

Example 1: NX=2, data points are on both sides.

AX1= 50, AX2= 50 ... G=19.0

AX1= 50, AX2=200 ... G=17.8

AX1=200, AX2=200 ... G=16.6

Example 2: NX=3, data points are on both sides.

AX1= 50, AX2= 50 ... G=9.9

AX1= 50, AX2=200 ... G=8.7

AX1=200, AX2=200 ... G=7.5

Example 3: NX=2, data points are on one side only.

AX1= 50, AX2= 50 ... G=6.0

AX1= 50, AX2=200 ... G=4.8

AX1=200, AX2=200 ... G=3.6

Example 4: NX=3, data points are on one side only.

AX1= 50, AX2= 50 ... G=5.9

AX1= 50, AX2=200 ... G=4.7

AX1=200, AX2=200 ... G=3.5

After the line where the interpolation takes place was chosen, the interpolation is performed in the following manner:

Let x_i (i=1,...,NX) be the distance with sign from the object point to the data points, and z_i (i=1,...,NX) be the associated z values. If $z_1 \neq z_{NX}$ and if NX>2 $z_1 = z_2 = \ldots = z_{NX-1}$. Therefore, the (NX-1)-th order polynomial z=f(x) which passes (x_i, z_i) (i=1,...,NX) is written as

$$z-z_1 = a(x-x_1)(x-x_2)\ldots(x-x_{NX-1}) \tag{7}$$

This function passes (x_{NX},z_{NX}) too, therefore

$$z_{NX}-z_1 = a(x_{NX}-x_1)(x_{NX}-x_2)\ldots(x_{NX}-x_{NX-1}) \tag{8}$$

Then

$$a = (z_{NX}-z_1)/(x_{NX}-x_1)(x_{NX}-x_2)\ldots(x_{NX}-x_{NX-1}) \tag{9}$$

The z value at the object point is obtained by letting x be zero in formula (7).

$$z_0=z_1+a(-x_1)(-x_2)\ldots(-x_{NX-1}) \tag{10}$$

$$=z_1+(z_{NX}-z_1)x_1x_2\cdots x_{NX-1}/(x_1-x_{NX})\cdots(x_{NX-1}-x_{NX}) \qquad (11)$$

In the case of NX=2, formula (11) is rewritten as

$$\begin{aligned} z_0 &= z_1+(z_2-z_1)x_1/(x_1-x_2) \\ &= (x_1z_2-x_2z_1)/(x_1-x_2) \end{aligned} \qquad (12)$$

In the case of NX=3, formula (11) becomes

$$z_0=z_1+(z_3-z_1)x_1x_2/(x_1-x_3)(x_2-x_3) \qquad (13)$$

Going back to the example of Figure 8, NX=2 for all four lines. However, on the x-parallel line two points are located on one side only. Therefore, the x-parallel line is evaluated to be the least optimum. Other lines are compared by their distances from the center point to the data points. One can see in Figure 8 that the NW-SE diagonal line (with points 5 and 6) is optimum. And interpolation is then performed along this line.

Once the elevation interpolation has been completed, then one has to proceed to the estimation of the slope and aspect. Similar to the elevation interpolation, first one evaluates four lines to choose the optimum two lines to calculate the slope and aspect. Then evaluation is performed by the following formula:

$$G(I)=AX1(I)+AX2(I) \qquad (I=1,2,3,4) \qquad (14)$$

The two lines with G(I) are chosen.

In the case shown in Figure 8, obviously the NW-SE diagonal line with points 5 and 6 and the y-parallel line with points 3 and 4 are chosen.

After two lines are determined, slope elements along those lines are estimated. Let z_0 be the interpolated elevation at the object point (x=0), and (x_1,z_1), (x_2,z_2) be the first two data points entered for the elevation

interpolation along the line, then the slope element is obtained as the slope at the center point (x=0) of the second order polynomial which passes $(0,z_0)$, (x_1,z_1) and (x_2,z_2). As the polynomial passes $(0,z_0)$ it is written as

$$z-z_0=ax^2+bx \tag{15}$$

The slope at x=0 is obtained as

$$SL(I) = \left(\frac{dz}{dx}\right)_{x=0}=b \tag{16}$$

As the polynomial passes (x_1,z_1) and (x_2,z_2), then

$$z_1-z_0=ax_1^2+bx_1 \tag{17}$$

$$z_2-z_0=ax_2^2+bx_2 \tag{18}$$

By solving these equations we obtain

$$SL(I) = b = \frac{x_1^2(z_2-z_0)-x_2^2(z_1-z_0)}{x_1^2x_2-x_2^2x_1} \tag{19}$$

$$= \frac{(z_2-z_0)(x_1^2-x_1x_2)-(z_1-z_0)(x_2^2-x_1x_2)+(z_2-z_1)x_1x_2}{x_1x_2(x_1-x_2)}$$

$$= \frac{z_1-z_0}{x_1} + \frac{z_2-z_0}{x_2} - \frac{z_1-z_2}{x_1-x_2} \tag{20}$$

If x_1, x_2 or (x_1-x_2) is close to zero, SL(I) becomes very sensitive to the little change of x and can be affected a great deal by small errors. Therefore, if the absolute value of x_1, x_2 or (x_1-x_2) is smaller than 30.0, which usually corresponds to 0.75mm on a map, SL(I) is obtained by one of the following equations:

$$SL(I) = \frac{1}{2}\left(\frac{z_1-z_0}{x_1} + \frac{z_2-z_0}{x_2}\right) \tag{21}$$

$$SL(I) = (z_1-z_2)/(x_1-x_2) \tag{22}$$

$$SL(I) = (z_1 - z_0)/x_1 \qquad (23)$$

$$SL(I) = (z_2 - z_0)/x_2 \qquad (24)$$

Thus far, one has obtained two slope elements of different directions. Therefore, we can obtain a slope vector by combining the two elements. However, there are four lines in every 45 degrees, and the two lines chosen here are not always orthogonal to each other. Therefore, we have to transform the pair of slope elements to the orthogonal pair if necessary.

Figure 10 illustrates the axes considered here. Let us suppose that the slope vector is explained as (SLX,SLY), then each slope elements along the four lines are written as

$$SL(1)=SLX \qquad (25)$$

$$SL(2)=-SLY \qquad (26)$$

$$SL(3)=(SLX-SLY)/2^{\frac{1}{2}} \qquad (27)$$

$$SL(4)=-(SLX+SLY)/2^{\frac{1}{2}} \qquad (28)$$

Then we can easily obtain SLX and SLY from any two of the above equations.

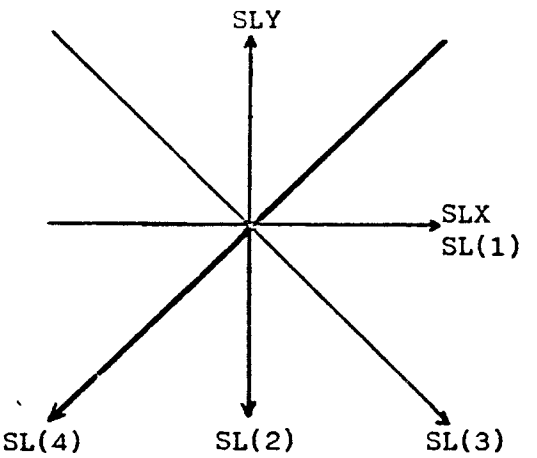


Figure 10. Axes for the slope calculation.

The slope vector (SLX,SLY) can define completely the local slope. However, from the user's point of view, the maximum slope and its azimuth are more useful than the slope vector. Therefore, the slope vector is transformed to the maximum slope and its azimuth. The transformation is performed as follows:

$$\text{Slope} = (SLX^2+SLY^2)^{\frac{1}{2}} \tag{29}$$

$$\text{Aspect} = \tan^{-1}(SLX/SLY) \tag{30}$$

Job Control Routine Flowchart. The terrain information system is initialized from CMS by issuing the "TERRAIN" command. The overall process consists of three steps, i.e. (1) the automatic arc closure step, (2) the projection transformation step and (3) the rasterization step. Each step can be entered independently or sequentially by specifying the desired step number from a terminal keyboard in accordance with the direction displayed soon after the "TERRAIN" command is issued.

EXEC FILE : TERRAIN

TERRAIN

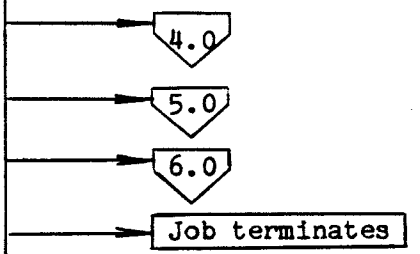
1.0 TERRAIN is entered from CMS by issuing the "TERRAIN" command.

2.0 LARSYS-disk attached.

3.0 Direction displayed.


```
*************************************************************************
*************************************************************************
***                                                                   ***
***    TERRAIN INFORMATION SYSTEM                                     ***
***                                                                   ***
***  THIS IS A MULTI-STEP PROCESS..                                   ***
***                                                                   ***
***     STEP1 :  ARC CLOSURE ROUTINE                                  ***
***     STEP2 :  PROJECTION TRANSFORMATION ROUTINE                    ***
***     STEP3 :  TERRAIN INFORMATION ROUTINE                          ***
***                                                                   ***
***  PLEASE ENTER WHERE TO START (1,2 OR 3)                           ***
***  THE WORD 'EXIT' TO EXIT                                          ***
***                                                                   ***
*************************************************************************
*************************************************************************
```

3.1 Job entry point is decided according to the job step number entered from keyboard.



4.0 Automatic arc closure routine starts.

4.1 Notes displayed.

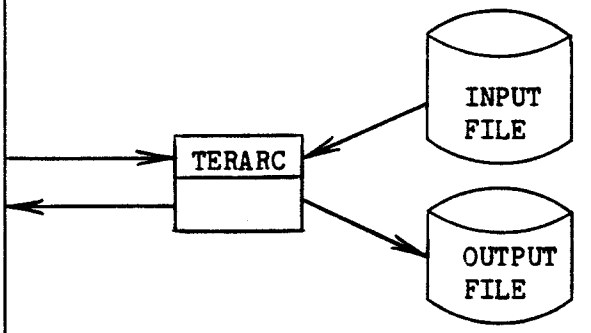
```
**************************************************
**                                              **
**     CLOSURE ROUTINE                          **
**                                              **
**  PLEASE ENTER INPUT FILENAME AND FILETYPE    **
```

```
**  OF DIGITIZED CONTOURS  (FN  FT)          **
**                                           **
***********************************************
```

4.2 Input file name and file type from keyboard

4.3 Check the existence of the input file.

4.4 Loads and executes TERARC.



4.5 Input file is replaced by output file.

4.6 Direction displayed.

```
***********************************************
**                                           **
**  WOULD YOU LIKE TO EXIT IN ORDER TO RUN   **
**  THE EDITMAP ROUTINE ??   (YES,NO)        **
**                                           **
***********************************************
```

"YES" → Job terminates

"NO" → 5.0

5.0 Projection transformation routine starts.

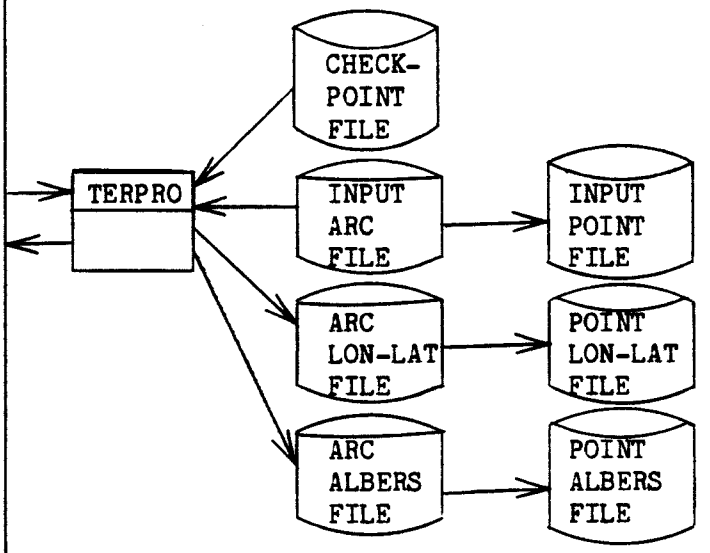
5.1 Direction displayed.

```
*************************************************************
**                                                         **
**     PROJECTION TRANSFORMATION ROUTINE                   **
**                                                         **
**  PLEASE ENTER INPUT FILENAME AND FILETYPE OF            **
**  CLOSED CONTOURS...   (FN  FT)                          **
**                                                         **
**  NOTICE :  YOU NEED TO HAVE A CHECKPOINT FILE           **
**            (FILETYPE = CHECK)                           **
**                                                         **
*************************************************************
```

5.2 Input file name and file type from keyboard.

5.3 Check the existence of the contour file and checkpoint file.

5.4 Loads and executes TERPRO.



5.5 Direction displayed.

```
**********************************************
**                                          **
**  WOULD YOU LIKE TO EXIT ??  (YES,NO)     **
**                                          **
**********************************************
```

"YES" → Job terminates

"NO" → 6.0

6.0 Rasterization routine starts.

6.1 Direction displayed.

```
****************************************************
**                                                **
**     TERRAIN INFORMATION ROUTINE                **
**                                                **
**  PLEASE ENTER INPUT FILENAME AND FILETYPE      **
**  OF CONTOUR ADDRESS DATA  (FN  FT)             **
**                                                **
****************************************************
```

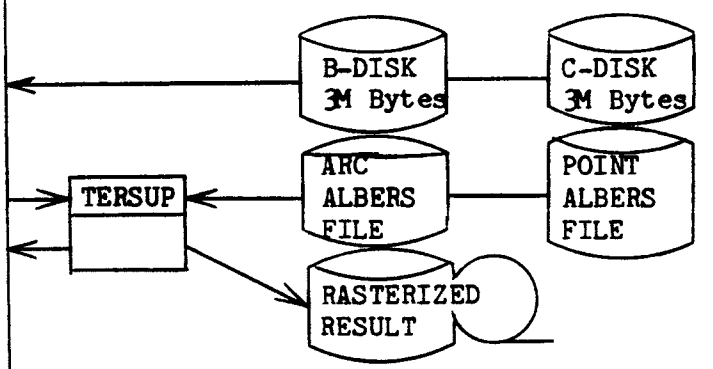
6.2 Input file name and file type from keyboard.

6.3 Check the existence of the contour file

6.4 Two temporary disks attached.

6.5 Loads and executes TERSUP

7.0 Job terminates.



Automatic Arc Closure Routine Flowchart. The input data for this routine is digitized contour data, and is replaced by the closed contour data through the execution.

The basic idea of the automatic arc closing is to let the x,y coordinates of both ends of an arc be the same if the distance between them is shorter than a given threshold.

The program was modified from the arc closure routine in "MAPPRO" (program name = ARCLOS).

One of the major modifications of the MAPPRO routine is that this program (MAPPRO) processes only linear features (contour lines), on the other hand "ARCLOS" processes arcs in addition to linear features. Another modification is that this program uses altitude in deciding whether the arcs should be closed or not, thus avoiding the connecting of arcs of different elevations. Another modification is that ARCLOS" may expect the connection of more than two arc ends, however, this program does not expect such situation.

Program limitations

Maximum number of arcs = 1,000

Maximum number of points in an arc = 4,000

PROGRAM FILE NAME : TERARC

TERARC

1.0 TERARC is entered from the TERRAIN exec routine.

2.0 Store beginning and ending points of each arc.

2.1 Read arc header

2.2 Read arc data and store end points.

2.3 Store associated data of end points.

2.4 Repeat the procedure from 2.0 through 2.3 for all arcs.

3.0 Closure distance threshold determination.

```
**************************************************
**************************************************
***                                            ***
***  PLEASE ENTER A CLOSURE DISTANCE IN INCHES.***
***                                            ***
***  (.25 IS SUGGESTED... HIT RETURN TO .25)   ***
***                                            ***
**************************************************
**************************************************
```

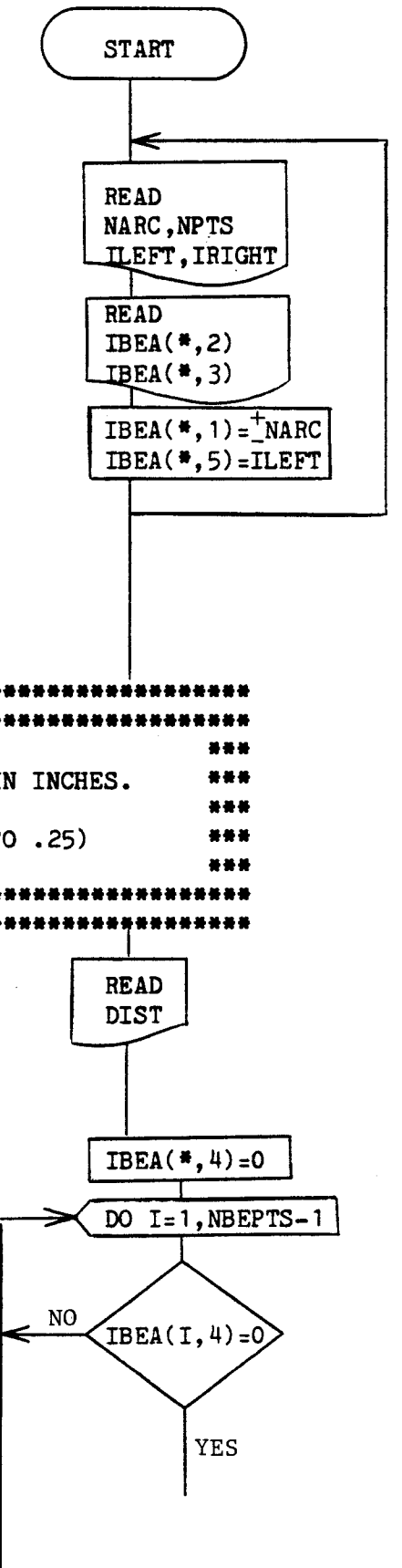
3.1 Input distance in inches

4.0 Compare end points each other to find match pairs.

4.1 Matching indicator initialized.

4.2 Repeat the following procedure through 5.3 for all end points.

4.3 Get one end point to be compared. If matching indicator is not zero skip the point.



4.4 Search matching point among all other points.

4.5 Skip the point if matching indicator is not zero or altitude is not same as point I.

4.6 Calculate the distance between point I and K.

4.7 If the distance is shorter than the threshold they are considered as a matched pair.

4.8 If the point I is isolated write the information about the point on a closure report.

5.0 Get the middle point of the pair.

5.1 Store the matched pair information.

5.2 Let matching indicator be one.

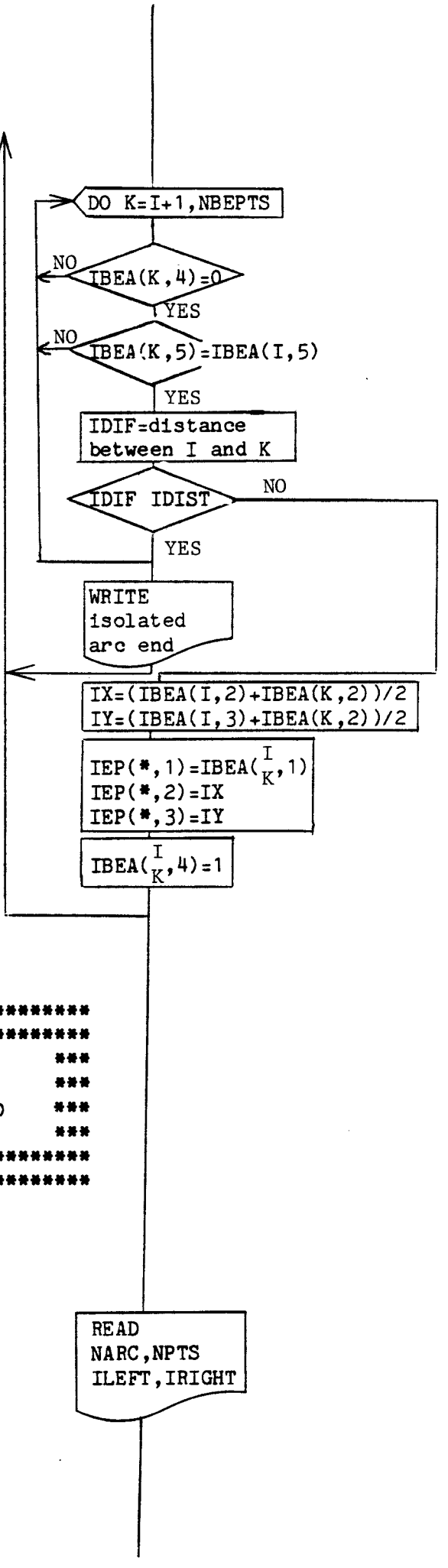
5.3 Proceed to the next point.

5.4 All end points have been compared.

```
**************************************************
**************************************************
***                                            ***
***  ALL ARCS HAVE BEEN CLOSED...              ***
***  A CLOSURE REPORT IS BEING PRINTED         ***
***                                            ***
**************************************************
**************************************************
```

6.0 Correction of the end points of an arc. Repeat the following procedure through 6.8 for all arcs.

6.1 Read arc header.



6.2 Read arc Data.

6.3 Check whether this arc is to be corrected.

6.4 Correction of the beginning point.

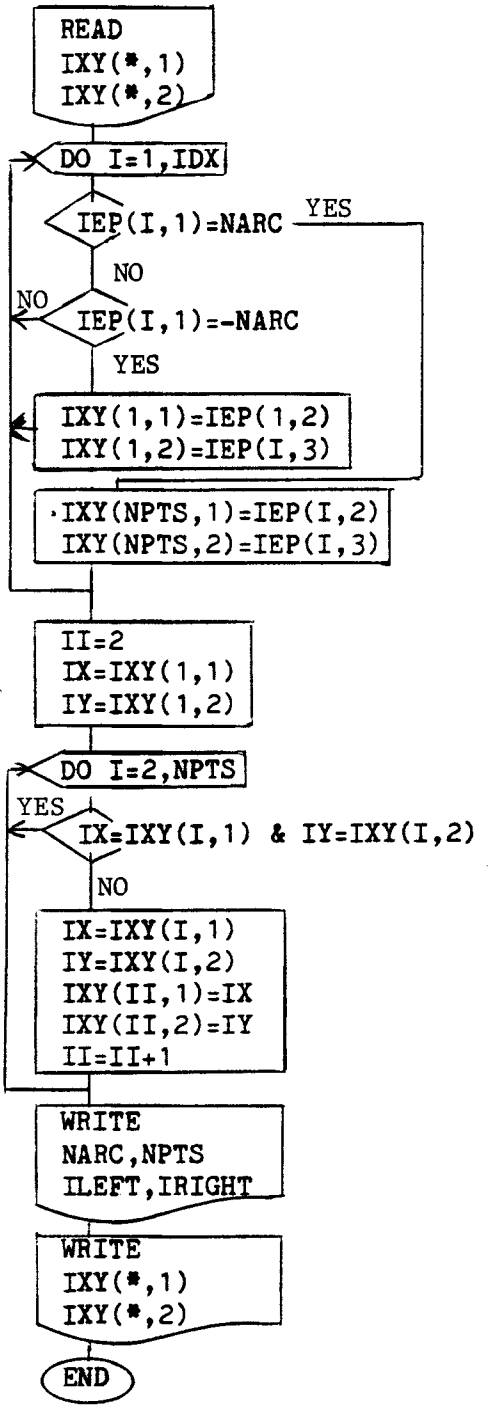
6.5 Correction of the ending point.

6.6 Remove any point that is the same as the previous one.

6.7 Write arc header.

6.8 Write arc data.

7.0 Job terminates.



Projection Transformation Routine Flowchart. The input data for this routine are checkpoint data, closed contour data and boundary point elevation data.

The coefficients of the transformation from x,y coordinates of the digitizer coordinate system to longitude and latitude are obtained by means of the least square method using checkpoint data. Then the contour data and the boundary point data are transformed from x,y to Albers addresses through longitude and latitude.

The output data consists of transformed contour and boundary point data in a longitude-latitude coordinate system and in Albers addresses as well.

The transformation from x,y to longitude-latitude is accomplished by the second order polynomial approximation, and the transformation from longitude-latitude to Albers addresses is accomplished by the exact transformation function.

Program limitations

Maximum number of checkpoint = 50

Maximum number of points in an arc = 4,000

PROGRAM FILE NAME : TERPRO

TERPRO

1.0 TERPRO is entered from TERRAIN exec routine.

2.0 Bolivian projection standards are set.
P1 & P2 are the standard parallels. MN is the standard meridian. RE & RF are the equatorial and the polar radious respectively.

3.0 Gridding factor determination.

3.1 Get the scale of the digitized map recorded on the top of the checkpoint file.

3.2 Calculate the gridding factor to be recommended.

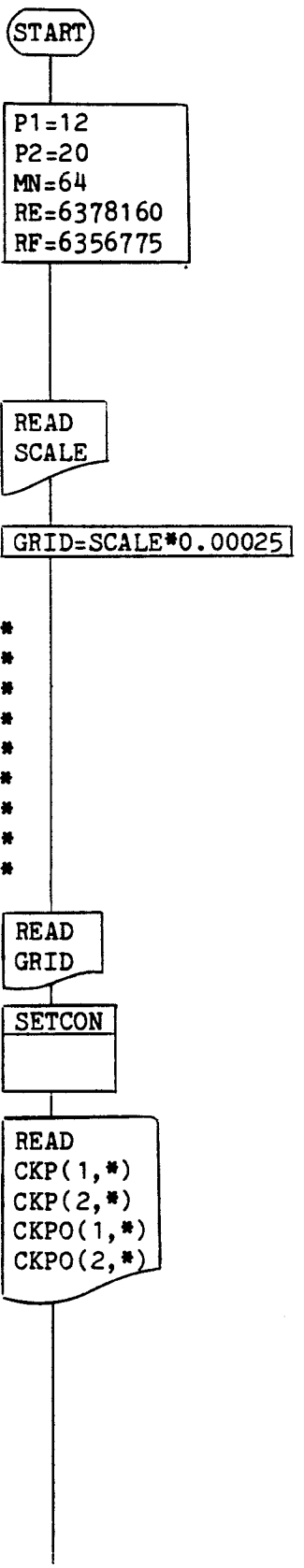
```
**********************************************
*                                            *
*  WITH A MAP SCALE OF ########.###          *
*  THE PROPOSED GRIDDING FACTOR IS ####.#    *
*                                            *
*  PLEASE ENTER A NEW GRIDDING FACTOR, OR    *
*  HIT THE RETURN KEY TO KEEP THE SAME...    *
*                                            *
**********************************************
```

3.3 Input the gridding factor.

4.0 SETCON is called to set the Bolivian projection constants.

5.0 Read and store the checkpoint data.

6.0 Transformation (x,y to lon-lat) coefficients calculation.



6.1 Set the observation equations of the least square method.

6.2 Set the normal equation of the least square method.

6.3 Solve the normal equation.

6.4 Transform checkpoint from x-y to longitude-latitude.

6.5 Call ALBERS to transform checkpoint from longitude-latitude to Albers addresses.

6.6 Transformation performance report is printed.

6.7 Maximum residual is calculated.

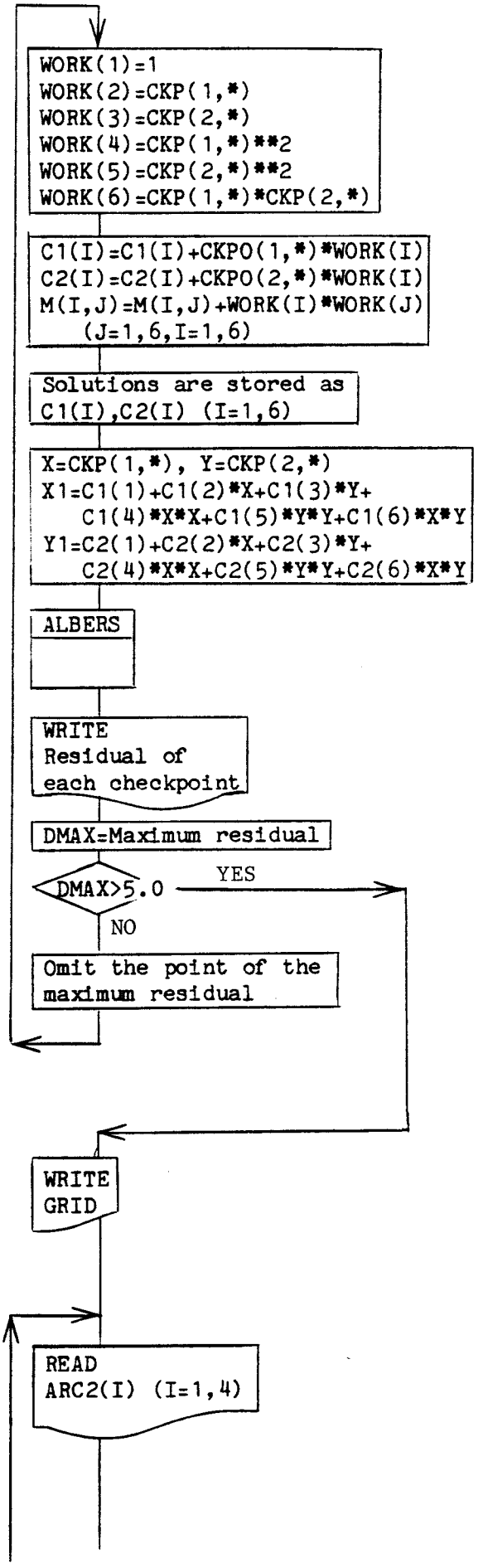
6.8 If the maximum risidual is within the limit (5 pixels), Proceed to transform the data. If not, omit the point of the maximum residual and go back to 6.0 to repeat the same procedure.

7.0 Transformation of the contour data.

7.1 Write the gridding factor on the top of the Albers address contour file.

7.2 Repeat the following procedure through 7.8 for all arcs.

7.3 Read arc header.



7.4 Read arc data.

7.5 Transform each arc point from x-y to longitude-latitude.

7.6 Call ALBERS to transform from longitude-latitude to albers addresses.

7.7 Output arc header and arc data of Albers addresses.

7.8 Output arc header and arc data of longitude-latitude.

8.0 Transformation of the boundary point data.

8.1 Repeat the following procedure through 8.8 for all points.

8.2 Read point header.

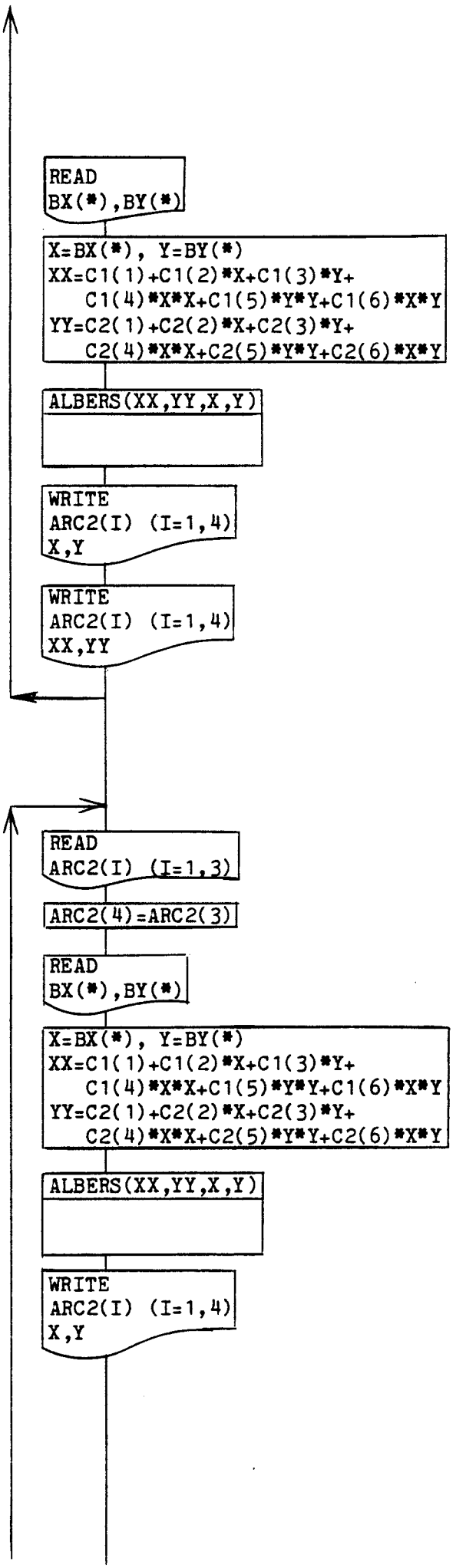
8.3 Let ARC2(4)=ARC2(3).

8.4 Read point data.

8.5 Transform each boundary point from x-y to longitude-latitude.

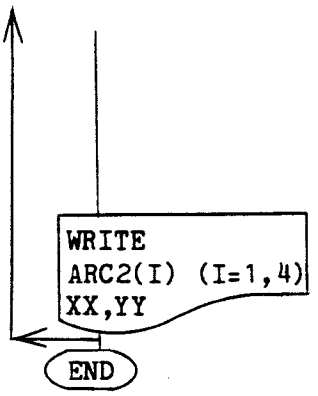
8.6 Call ALBERS to transform from longitude-latitude to Albers addresses.

8.7 Output point header and point data of Albers addresses.



8.8 Output point header and point data of longitude-latitude.

9.0 Job terminates.



SUBROUTINES called from TERPRO

SETCON

Purpose : Set the Bolivian projection constants.

Input data : COMMON/STANP/ P1,P2,MN,RE,RF,GRID

P1,P2 : Standard parallels.

MN : Standard meridian.

RE : Equatorial radius

RF : Polar radius

GRID : Gridding factor.

Output data : COMMON/SETC/ RD,C1,C2,C3,C9,A1,A2,A3,N

RD : Transformation factor from degree to radian.

C1-C9 : Local sphere radius parameters.

A1-A3 : Developmental radius parameters.

N : Special constant.

ALBERS(LN,LT,XX,YY)

Purpose : Transformation from longitude-latitude to Albers addresses.

Input data : LN,LT & COMMON/SETC/

LN : Input longitude.

LT : Input latitude.

Output data : XX,YY

XX : Output x address in Albers projection.

YY : Output y address in Albers projection.

Rasterization Routine Flowchart. The input data for this routine are contour and boundary point data in Albers addresses. Using these data, the elevation, slope and aspect at each pixel center are estimated and recorded on a tape.

The overall procedure can be divided into three steps. In the first step, contour and boundary point data are transformed into four sets of intersections of contours and the lines passing through a pixel center and parallel to x-axis, y-axis and two diagonal axes. Then, in the second step, each file of intersections is sorted out by their ascending order. Finally, the terrain information at a pixel center are estimated by interpolating those intersections.

Program limitations

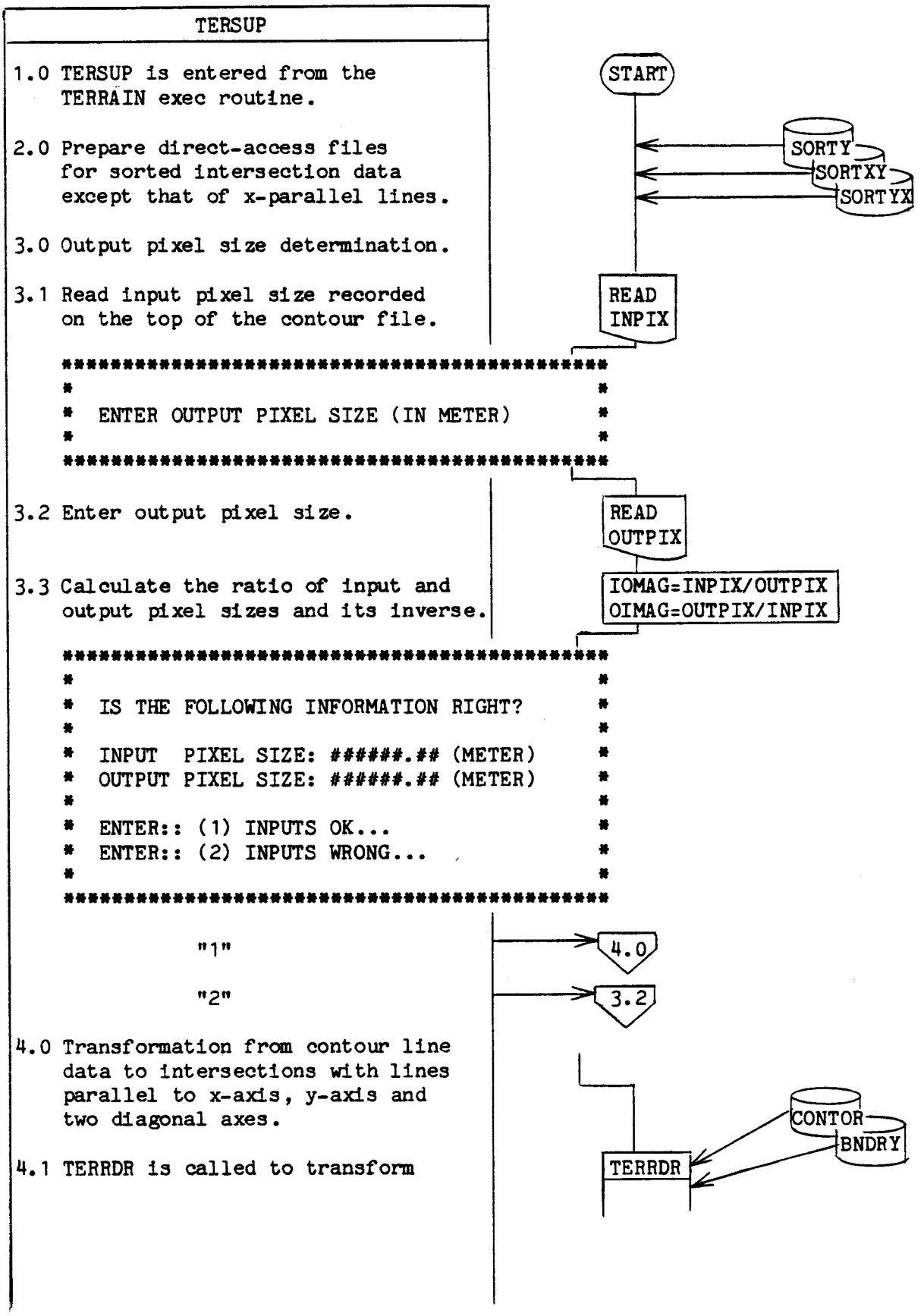
Maximum number of pixels in one line = 500

Storage capacity for sorted intersections of y-parallel lines = 0.48M

Storage capacity for sorted intersections of diagonal lines = 0.72M * 2

Storage capacity for sorted intersections of x-parallel lines
and the rasterized results = 0.96M

PROGRAM FILE NAME : TERSUP



contour and boundary point data to intersection data.

4.2 Check the storage capacity for the sorted intersection data.

5.0 Sorting of the intersection data.

5.1 TERSOR is called to sort out the intersection data of x-parallel axis.

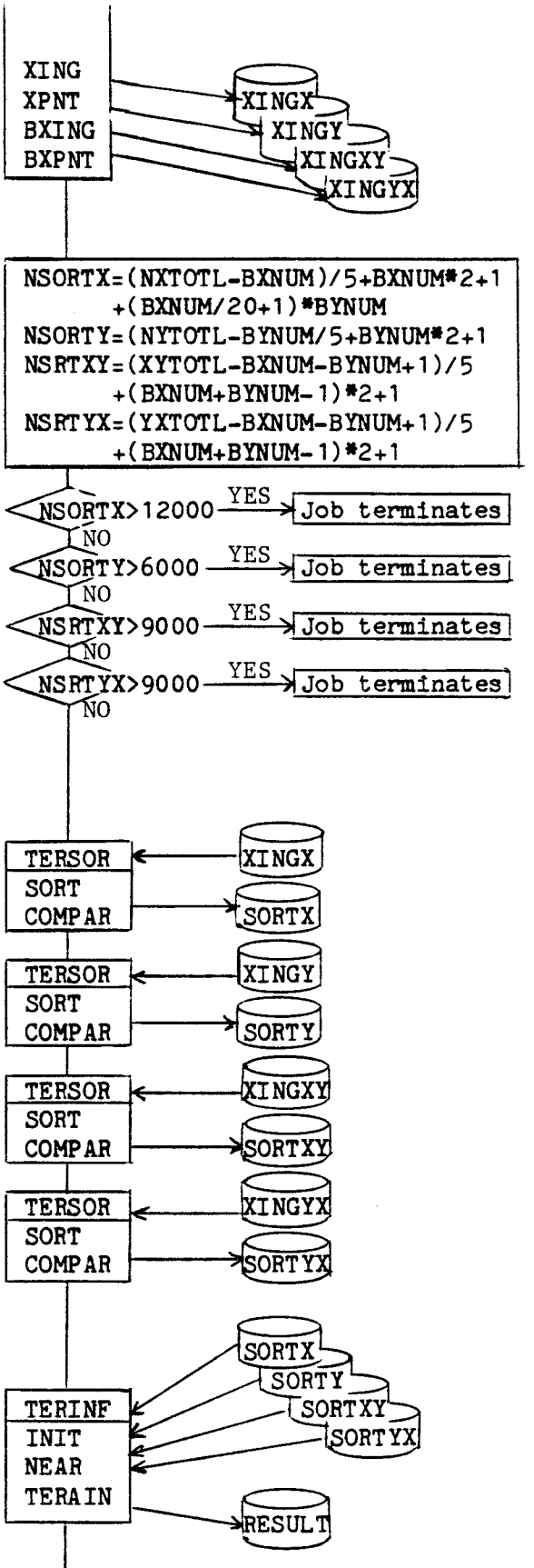
5.2 TERSOR is called to sort out the intersection data of y-parallel axis.

5.3 TERSOR is called to sort out the intersection data of diagonal axis.

5.4 TERSOR is called to sort out the intersection data of the other diagonal axis.

6.0 Terrain information calculation.

6.1 TERINF is called to calculate terrain information.



7.0 Output onto tape.

```
**********************************************
*                                            *
*  WOULD YOU LIKE TO KEEP THE RESULT         *
*  ON TAPE ?  (1) : YES / (2) : NO           *
*                                            *
**********************************************
```

"1"

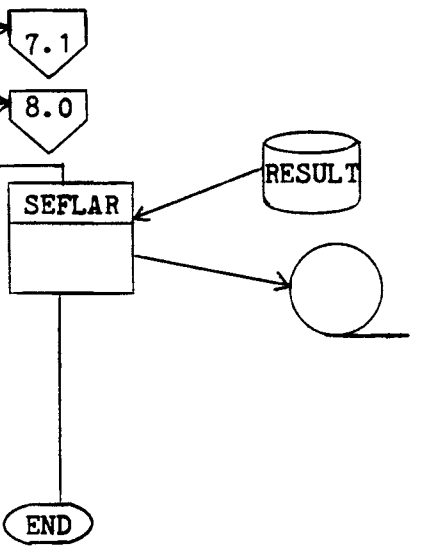
"2"

7.1 SEFLAR is called to send the result onto tape.

8.0

*** SUCCESSFUL RUN ***

Job terminates.



SUBROUTINES called from TERSUP

TERRDR

Purpose : Transform contour and boundary point data into intersection data.

Input data : Disk files

file type = CONTOR : Contour line data in Albers addresses.

file type = BNDRY : Boundary point data in Albers addresses.

Output data : Disk files

file type = XING* : Intersection data of contours and a set of parallel

* = X,Y,XY or YX lines of the direction corresponding to *

Using subroutines

XING : Call XPNT four times for a line between two successive points in an arc in order to obtain the intersections.

XPNT : Find any of the parallel lines which intersect the line between two given points and obtain the intersections.

BXING : Read all the boundary point data and call BXPNT four times for a line between two successive points in order to obtain intersections.

BXPNT : Find any of the parallel lines which intersect the line between two given points, then obtain the intersections and interpolate their elevations.

TERSOR

Purpose : Sort out intersections by their ascending order.

Input data : Disk files

file type = XING* : Intersection data before sorting.

* = X,Y,XY or YX

Output data : Disk files

file type = SORT* : Sorted intersection data.

* = X,Y,XY or YX

Using subroutines

SORT : Sorting program by means of a "tree" method.

COMPAR : Combine two or more sorted files into a large sorted file by comparing the current top data of each input file.

TERINF

Purpose : Calculate terrain information at the center of each pixel in the area to be rasterized.

Input data : Disk files

file type = SORT* : Sorted intersection data.

* = X,Y,XY or YX

Output data : Disk file

file type = RESULT : Rasterized result.

Using subroutines

INIT : Read the intersection data of a given line from a direct-access file and fill them into a corresponding data array.

NEAR : Find the points necessary to calculate the terrain information.

TERAIN : Calculate the terrain information of a given point.

SEFLAR

Purpose : Dump the rasterized result recorded on a disk file onto magnetic tape in MIST format.

Input data : Disk file

file type = RESULT : Rasterized result.

Output data : Tape file and disk file

Tape file : Rasterized result in MIST format.

Disk file : file type = CC : CC file for CDISPLAY function.

Using subroutines

TAPOP : Perform various kinds of tape operations.

SUBROUTINE NAME : TERRDR

TERRDR

1.0 TERRDR is called from TERSUP.

2.0 Determination of the area to be rasterized.

2.1 Read boundary point data point by point.

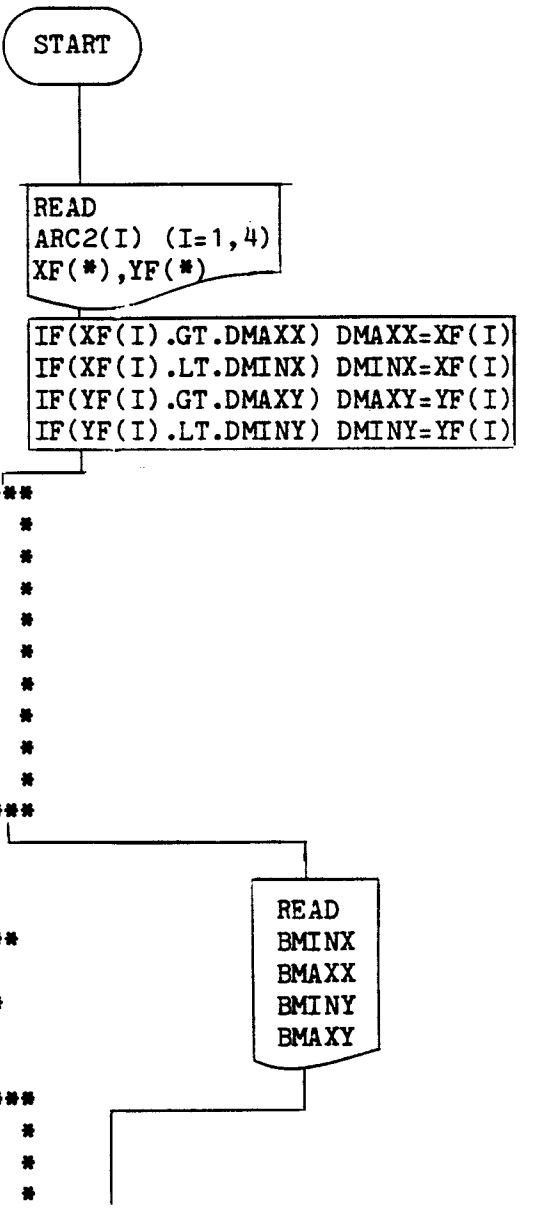
2.2 Find the maximum and the minimum x and y.

```
**********************************************
*                                            *
*  X,Y DATA RANGE OF INPUT CONTOURS          *
*                                            *
*  X:  FROM ########  TO ########            *
*  Y:  FROM ########  TO ########            *
*                                            *
*  PLEASE ENTER THE FOLLOWING INFORMATION    *
*  ABOUT THE AREA TO BE RASTERIZED           *
*                                            *
**********************************************
```

2.3 Input the area to be rasterized.

```
***  ENTER THE INITIAL PIXEL (X) NUMBER  ***
***  ENTER THE FINAL PIXEL (X) NUMBER  ***
***  ENTER THE INITIAL LINE (Y) NUMBER  ***
***  ENTER THE FINAL LINE (Y) NUMBER  ***

**********************************************
*                                            *
*  X,Y DATA RANGE OF INPUT CONTOUR           *
*                                            *
*  X:  FROM ########  TO ########            *
*  Y:  FROM ########  TO ########            *
*                                            *
*  AREA TO BE RASTERIZED                     *
*                                            *
*  X:  FROM ########  TO ########            *
*  Y:  FROM ########  TO ########            *
*                                            *
*  IS THIS INFORMATION RIGHT?                *
```



```
*                                                *
*  ENTER:: (1)  INPUTS OK...                     *
*  ENTER:: (2)  INPUTS WRONG...                  *
*                                                *
**************************************************
```

"1"

"2"

2.4 Calculate the number of columns and lines, and the maximum and the minimum values for the diagonal axes.

3.0 Transformation from contours to intersections.

3.1 Initialization.

3.2 Read arc header.

3.3 Read arc data.

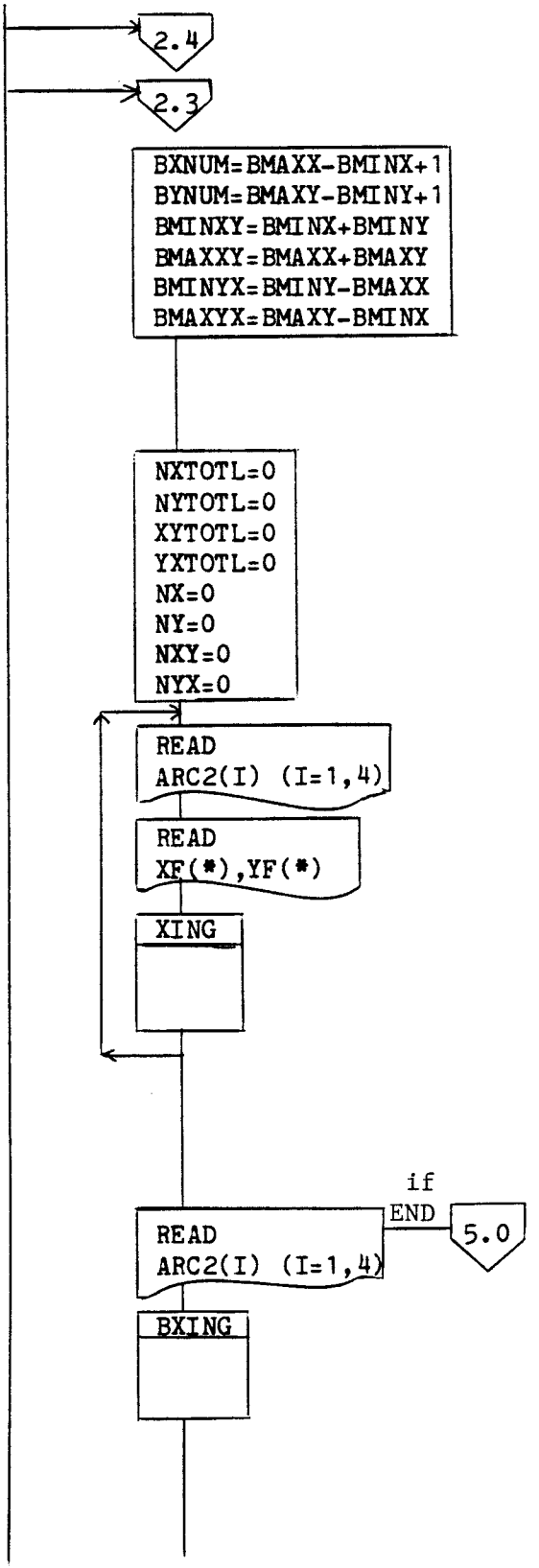
3.4 Call XING to transform contours into intersections.

3.5 Go back to 3.2 and repeat the same procedure until every contour has been processed.

4.0 Transformation from boundary point data to intersections.

4.1 Confirm that boundary point data exist.

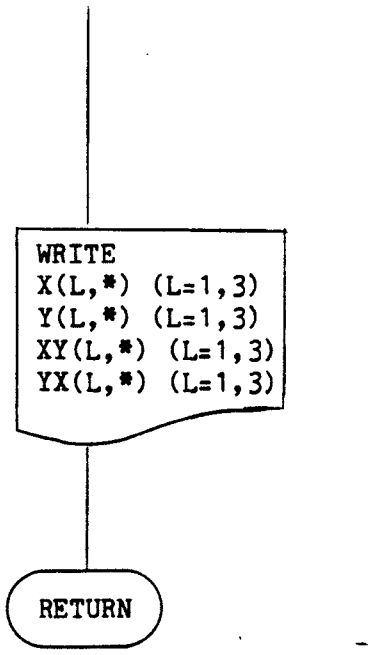
4.2 Call BXING to transform boundary point data to intersections.



5.0 Dump the intersection data
left in the data array onto
disk files.

```
**********************************
*                                *
*  CONVERSION COMPLETED          *
*                                *
**********************************
```

6.0 Return to TERSUP



SUBROUTINE NAME : XING(NF,ZO,XF,YF,X,Y,XY,YX,FDIM,XDIM)

Input data : NF,ZO,XF,YF

NF : Number of points in the contour.

ZO : Altitude of the contour.

XF(*),YF(*) : X,Y coordinates of the points.

Output data : X,Y,XY,YX

X(1,*) : Line number (y-coordinate) of intersection of x-parallel lines.

X(2,*) : x-coordinate of intersection of x-parallel lines.

X(3,*) : Altitude of intersection of x-parallel lines.

Y,XY,YX : Similarly, the array for intersections of y-parallel and two diagonal lines respectively.

XING

1.0 XING is called from TERRDR.

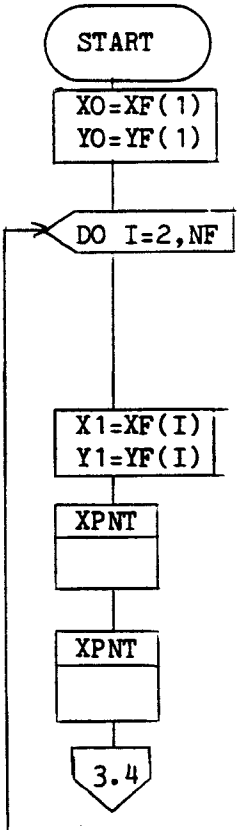
2.0 Initialization : Set the initial point of the line segment.

3.0 Repeat the following procedure through 3.6 to find any of the parallel lines which intersect a contour line segment from (X0,Y0) to (X1,Y1).

3.1 Set the other end of the line segment.

3.2 Call XPNT to obtain intersections of the line segment and x-parallel lines.

3.3 Call XPNT to obtain intersections of the line segment and y-parallel lines.

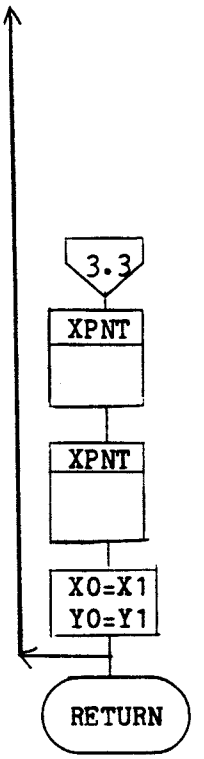


3.4 Call XPNT to obtain intersections of the line segment and the diagonal lines.

3.5 Call XPNT to obtain intersections of the line segment and the other diagonal lines.

3.6 Replace (X0,Y0) by (X1,Y1) and go back to 3.0 to proceed to the next line segment.

4.0 Return to TERRDR.



SUBROUTINE NAME : XPNT(IOUT,XO,YO,IYO,X1,Y1,IY1,MINY,MAXY,X,NX,
NXTOTL,XDIM,OIMAG,SHIFT,ZO)

Input data : XO,YO,X1,Y1,IYO,IY1,MINY,MAXY,ZO

XO,YO,X1,Y1 : Both ends of the line segment being considered.

IYO,IY1 : IYO=INT(YO+SHIFT)*IOMAG , IY1=INT(Y1+SHIFT)*IOMAG

In order to keep the original data precision, the unit of those data XO,YO,X1,Y1 corresponds to the input pixel size (INPIX). However, parallel lines are supposed to pass through the center of the output pixels, therefore, IYO and IY1 which correspond to the output pixel size (OUTPIX) are introduced in addition to YO and Y1.

MINY,MAXY : Data range of the line numbers to be considered.

ZO : Altitude of the contour being processed.

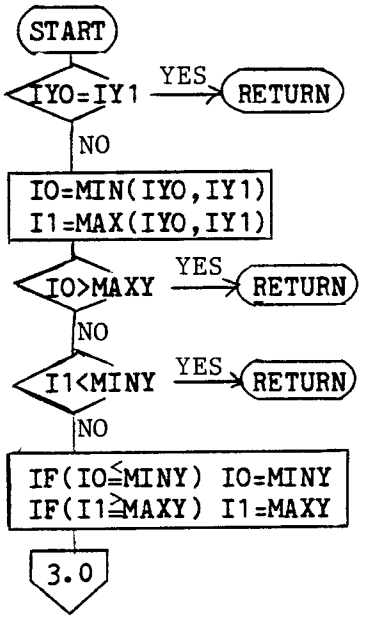
Output data : X

X(1,*) : Line number of an intersection in output pixel unit.

X(2,*) : Coordinates of an intersection along the line ten times the input pixel unit.

X(3,*) : Altitude of an intersection in meters.

XPNT
1.0 XPNT is called from XING. 2.0 Find if there exist any of the parallel lines which intersect the segment of the contour line.



3.0 Set the coefficients to obtain intersections.

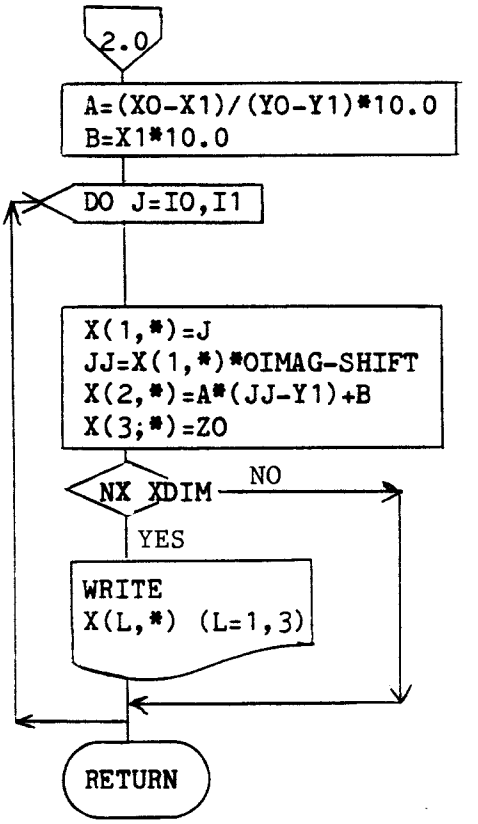
4.0 Repeat the following procedure through 4.3 for all lines intersecting the segment.

4.1 Calculate the coordinates of the intersection

4.2 If the data array is full, then dump the contents onto disk file.

4.3 Go back to 4.0 and proceed to the next line.

5.0 Return to XING.



SUBROUTINE NAME : BXING

BXING

1.0 BXING is called from TERRDR.

2.0 Read all boundary point data.

2.2 Keep the associated data.

3.0 Transformation from boundary point data to intersections.

3.1 Repeat the following procedure through 3.5 for all line segments between two successive points.

3.2 Call BXPNT to obtain intersections of the line segment and x-parallel lines.

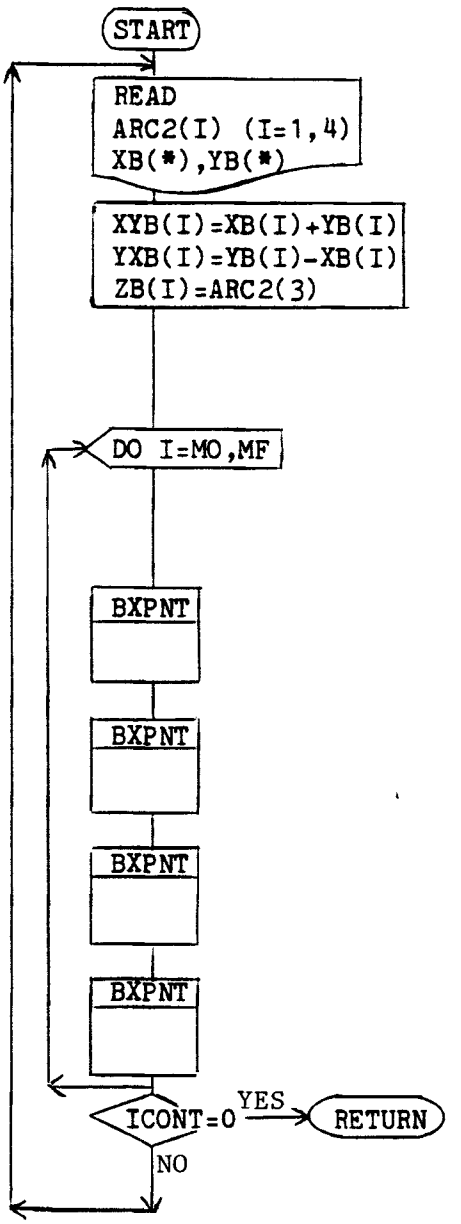
3.3 Call BXPNT to obtain intersections of the line segment and y-parallel lines.

3.4 Call BXPNT to obtain intersections of the line segment and the diagonal lines.

3.5 Call BXPNT to obtain intersections of the line segment and the other diagonal lines.

4.0 If all points have been processed, then return to TRRDR.

4.1 Go back to 2.0 and proceed to the rest of the data.



SUBROUTINE NAME : BXPNT(IOUT,XB,YB,ZB,BDIM,IP,NF,MINY,MAXY,
X,NX,NXTOTL,XDIM,IOMAG,SHIFT)

Input data : XB,YB,ZB,IP,MINY,MAXY

XB(*),YB(*),ZB(*) : Boundary point data array

IP : Current Point number to be considered in the array. The line segment being considered is from (XB(IP-1),YB(IP-1)) to (XB(IP),YB(IP))

MINY,MAXY : Data range of the line number to be considered.

Output data : X

X(1,*) : Line number of an intersection in output pixel unit.

X(2,*) : Coordinates of an intersection along the line ten times the input pixel unit.

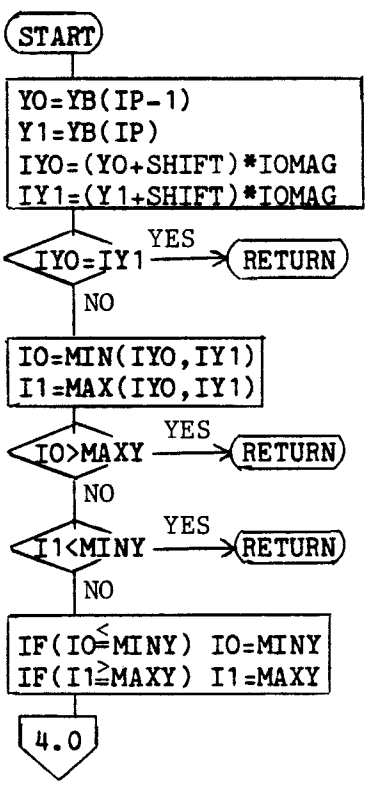
X(3,*) : Altitude of an intersection in meter.

BXPNT

1.0 BXPNT is called from BXING.

2.0 Set both ends of the line segment.

3.0 Find if there exist any of the parallel lines which intersect the line segment.



4.0 Set the coefficients to obtain intersections.

5.0 Repeat the following procedure through 8.1 for all lines intersecting the segment.

5.1 Calculate the planary coordinates of the intersection.

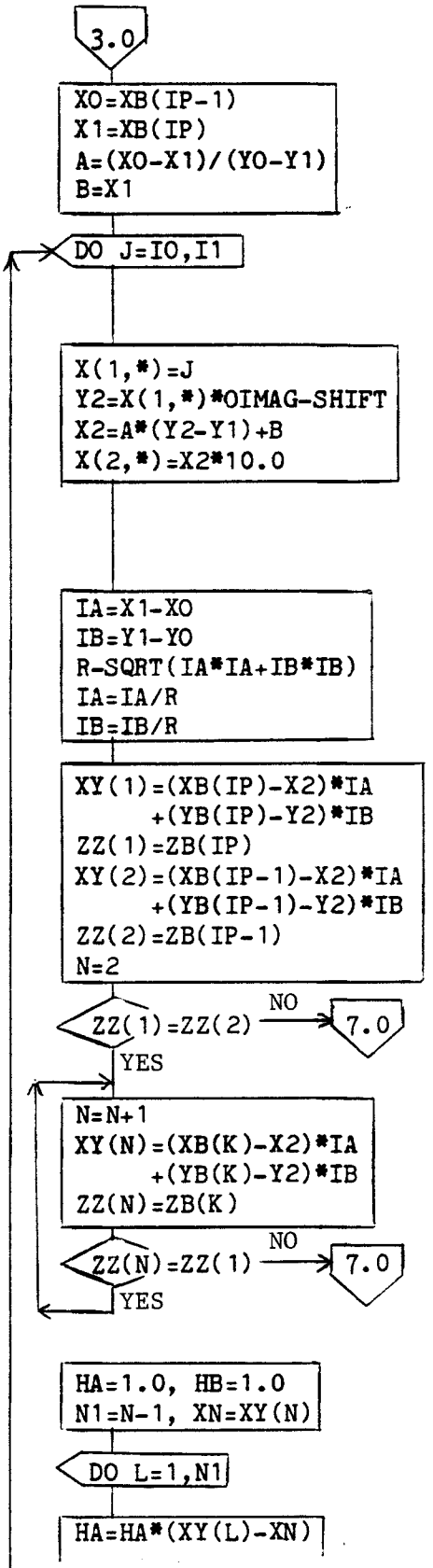
6.0 Interpolation of the altitude of the intersection.

6.1 Determination of the axis where the approximation is taken place.

6.2 Set the nearest two points.

6.3 If the altitudes of those two points are the same, get the next nearest points until a different altitude is found.

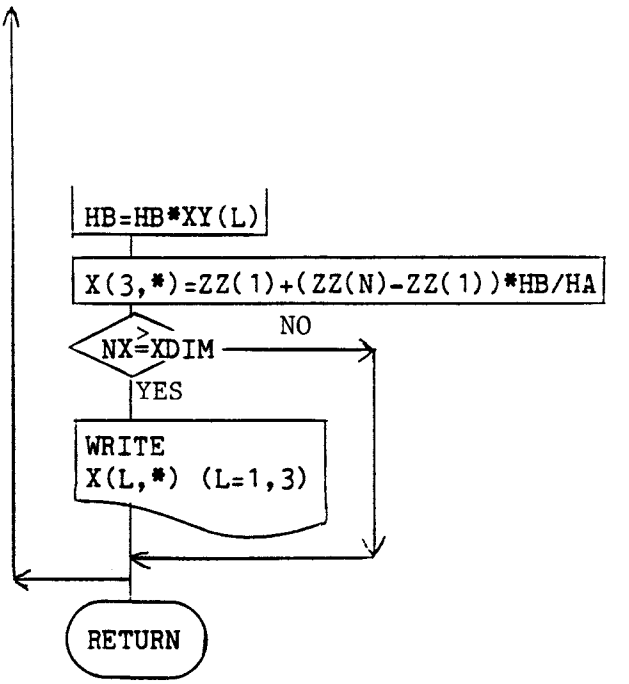
7.0 Altitude interpolation. Interpolation is performed by an N1-th order polynomial.



8.0 If the data array is full, then dump the contents onto a disk file.

8.1 Go back to 5.0 to proceed to the next line segment.

9.0 Return to BXING.



SUBROUTINE NAME : TERSOR

TERSOR

1.0 TERSOR is called from TERSUP.

****
* *
* BEGINNING TO SORT *
* *
****

2.0 Set the temporal files.

3.0 Because of the limitation of core memory locations, it may be necessary to divide the input data into some parts so that the amount of each part of data can be within the limitations.
Find out how many parts it should be divided into, and then follow the different procedures according to the number of parts.

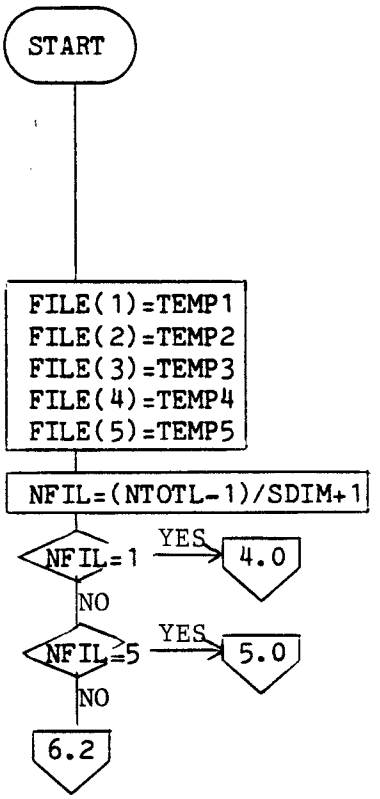
4.0 In case the total amount of data is small enough to sort out in one time.

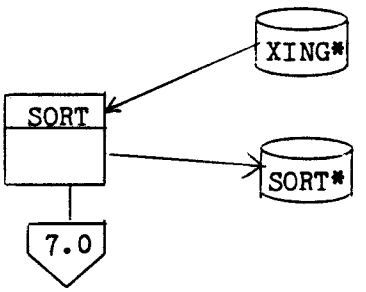
4.1 Call SORT for sorting. Sorted file is written on SORT* file.

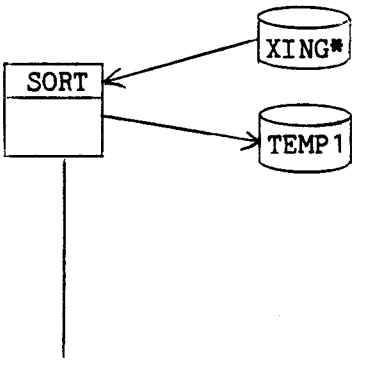
4.2 Go to 7.0 to return to TERSUP

5.0 In case the number of parts is five or more.

5.1 Call SORT for sorting. Sorted file is written on temporal file.







5.2 Repeat the following procedure through 5.5 until the number of remaining parts is less than three.

5.3 Call SORT three times for sorting. Sorted files are written on temporal files.

5.4 Call COMPAR to combine four sorted files into a large file.

5.5 If the number of remaining part is less than three proceed to 6.2, otherwise go back to 4.2.

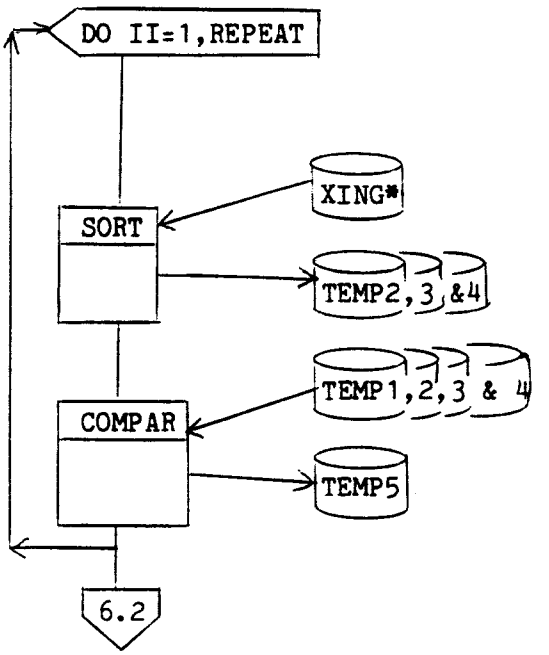
6.0 In case the number of parts is two, three or four.

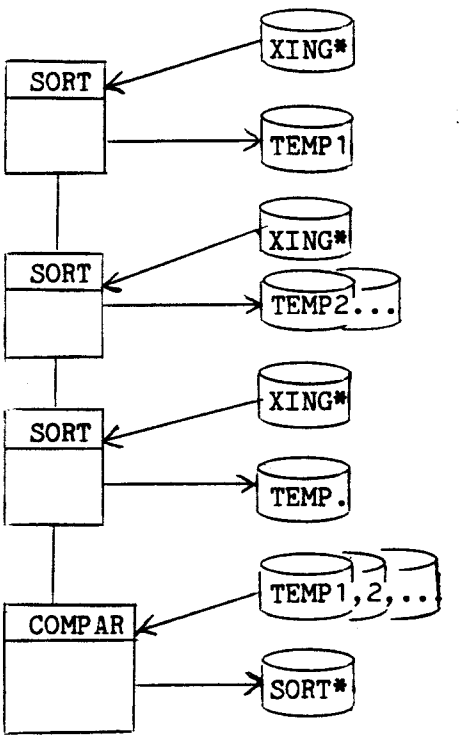
6.1 Call SORT for sorting. Sorted file is written on a temporal file

6.2 Call SORT repeatedly until one part is remaining.

6.3 Call SORT to sort out the rest of the data.

6.4 Call COMPAR to combine sorted files into a large file. Combined file is written on SORT* file.





7.0

```
**********************************
*                                *
*  SORTING COMPLETED             *
*                                *
**********************************

Return to TERSUP.
```

RETURN

SUBROUTINE NAME : SORT

SORT

1.0 SORT is called from TERSOR.

2.0 Read a set of data to be sorted out.

3.0 Set up tree indicators.

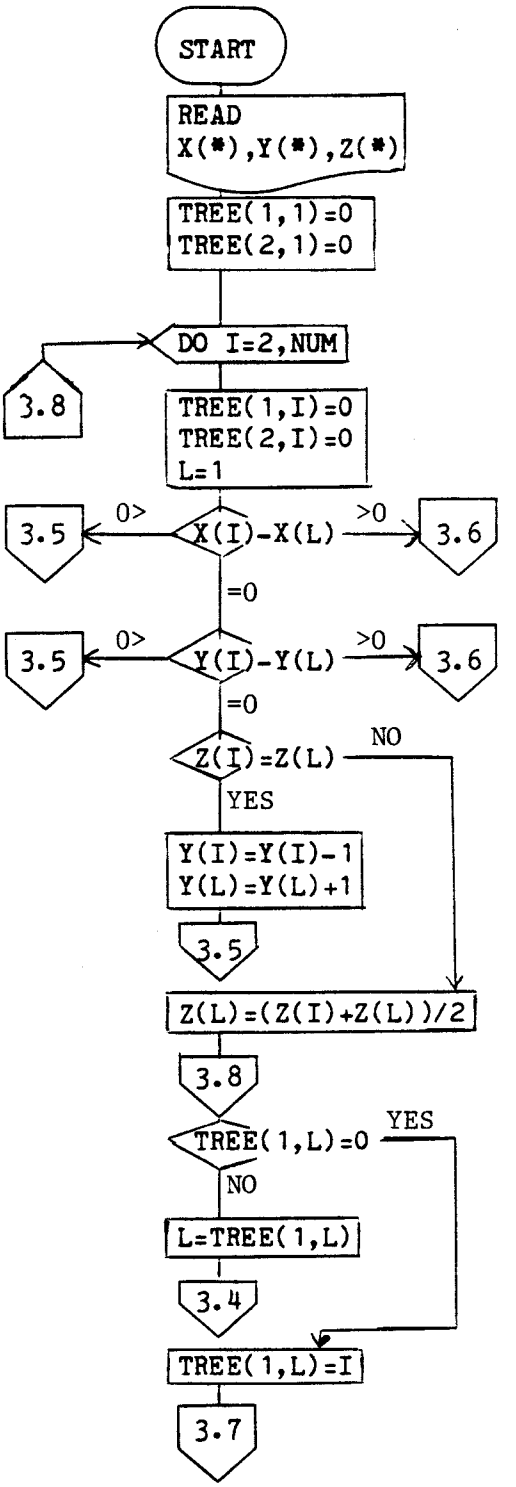
3.1 Initialization.

3.2 Repeat the following procedure through 3.8 for all points.

3.3 Initialization.

3.4 Compare point I with point L to determine the location of the point I in a "tree" structure.
Comparison is performed primarily by their X values and secondarily by their Y values. If both X and Y values are identical, combine two points by averaging their Z values. If the Z values are also the same, split those two points by adding one to the Y value of point L and subtracting one from the Y value of point I.

3.5 In the case that point I is smaller than point L, proceed to the next branch of the sorting tree. If the tree indicator TREE(1,L)=0, it is supposed to be the current terminal of the branch and is the location of the point I.



3.6 In the case that point I is larger than point L, proceed to the next branch of the sorting tree. If the tree indicator TREE(2,L)=0, it is supposed to be the current terminal of the branch and be the location of the point I.

3.7 Let the tree indicator TREE(3,I) be L

3.8 Go back to 3.2 and repeat the same procedure for the next point.

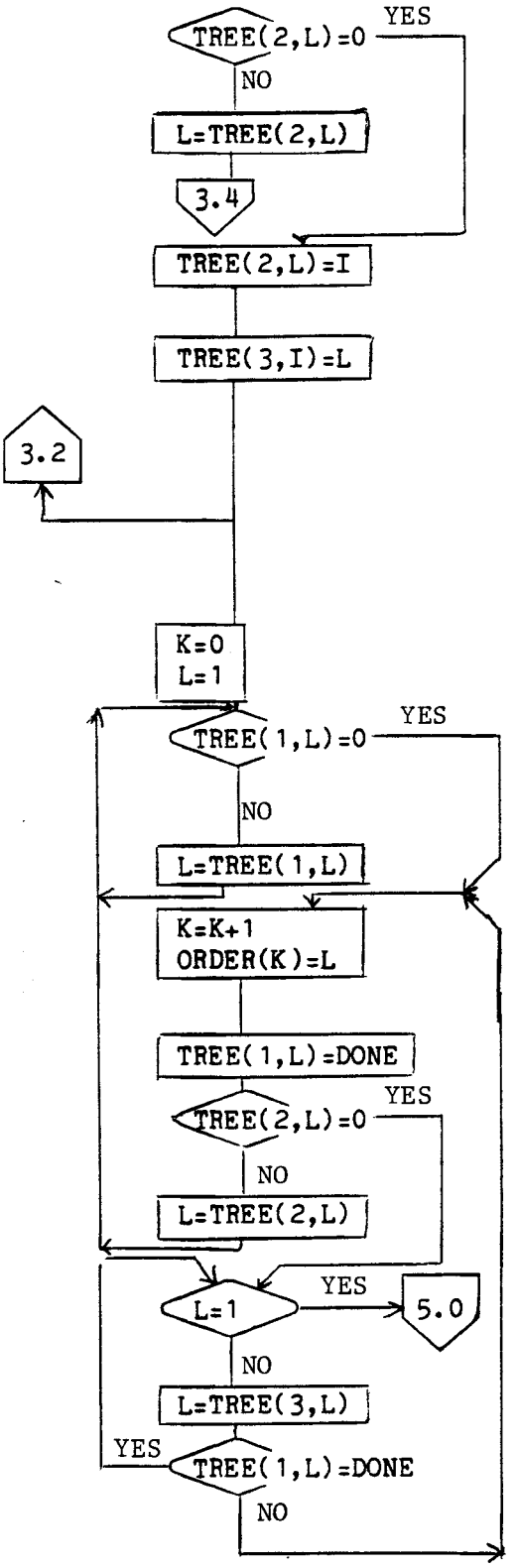
4.0 Trace the sorting tree and transform it to an order array.

4.1 Start with the first point.

4.2 Look for the smallest terminal in the tree.

4.3 If the point L is the current smallest terminal, let L be the K-th point in ORDER array.

4.4 Eliminate point L from the tree and proceed to the next branch.



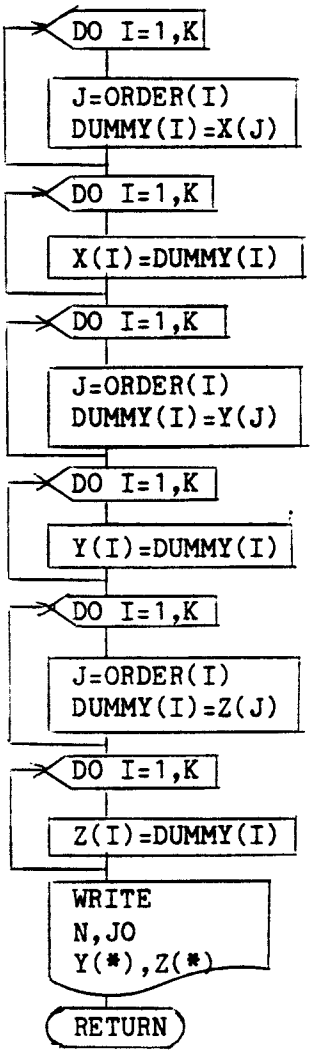
5.0 Sorting X(*) array.

5.1 Sorting Y(*) array.

5.2 Sorting Z(*) array.

6.0 Output the sorted results.

7.0 Return to TERSOR.



SUBROUTINE NAME : COMPAR

Since the number of the memory locations is limited, it is not always possible to sort out the data file by executing the SORT routine just one time. Therefore, it is necessary to divide the input file into some parts so that each part can be sorted by one time of execution of SORT routine. This subroutine COMPAR is to perform the final sorting of those individually sorted files.

As each individual file has been sorted, a comparison can be made among the top records of each files. The smallest record among them is supposed to be the first record for the final sorted file. Then transfer the record to the final file and take the next record of that file as the top record of the file. Then a comparison will be made among those records once again until all input files are empty.

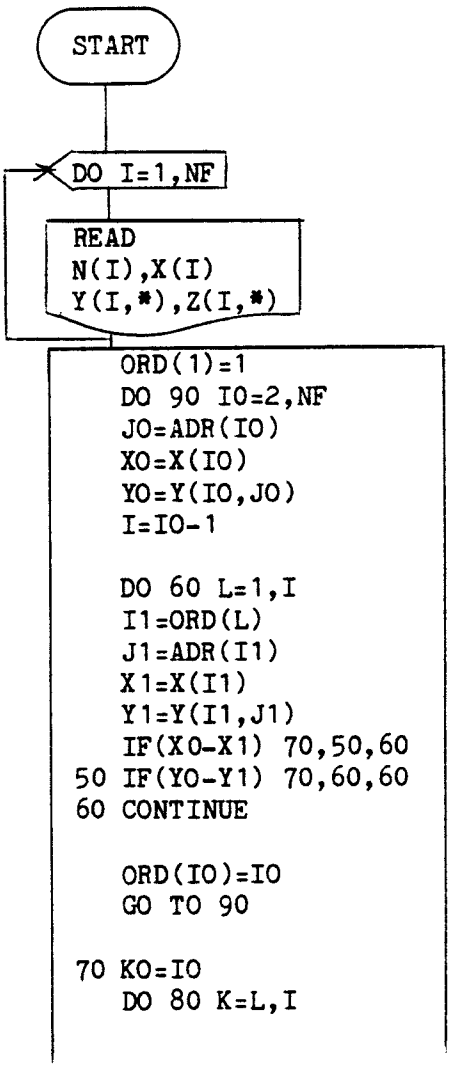
COMPAR

1.0 COMPAR is called from TERSOR.

2.0 Initialization.

2.1 Read the first data set from each input files.

2.2 Sort the input files by their top records.
ORD(I) : Order of the input file.
ADR(I) : Address of the current top record in the input data array.



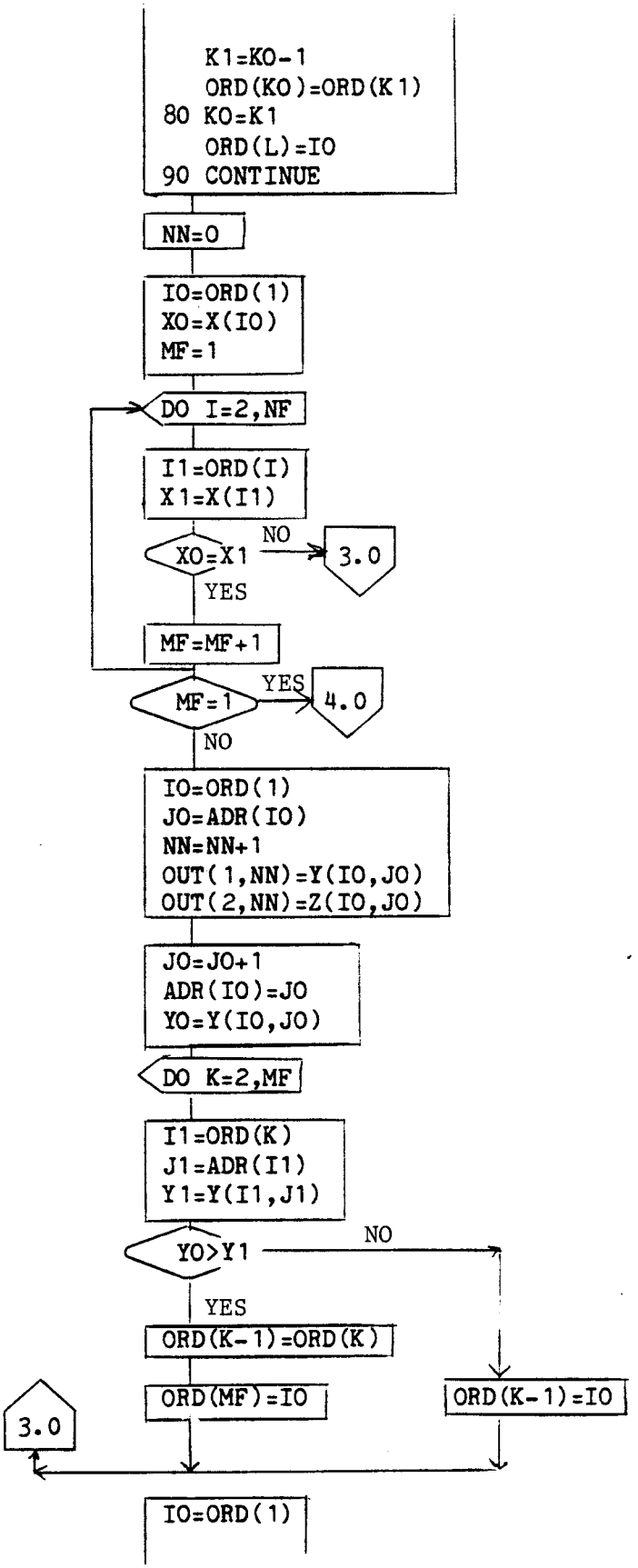
2.3 Set up some variables.
NN : Number of data in output array.
MF : Number of input files with the smallest X value.

3.0 Transfer the smallest record among the top records of the input files to an output array. If there is only one file which has the smallest X value, go to 4.0

3.1 Replace the top record by the next record.

3.2 Compare YO with other top records to determine the position of YO.

4.0 In the case that MF=1,just

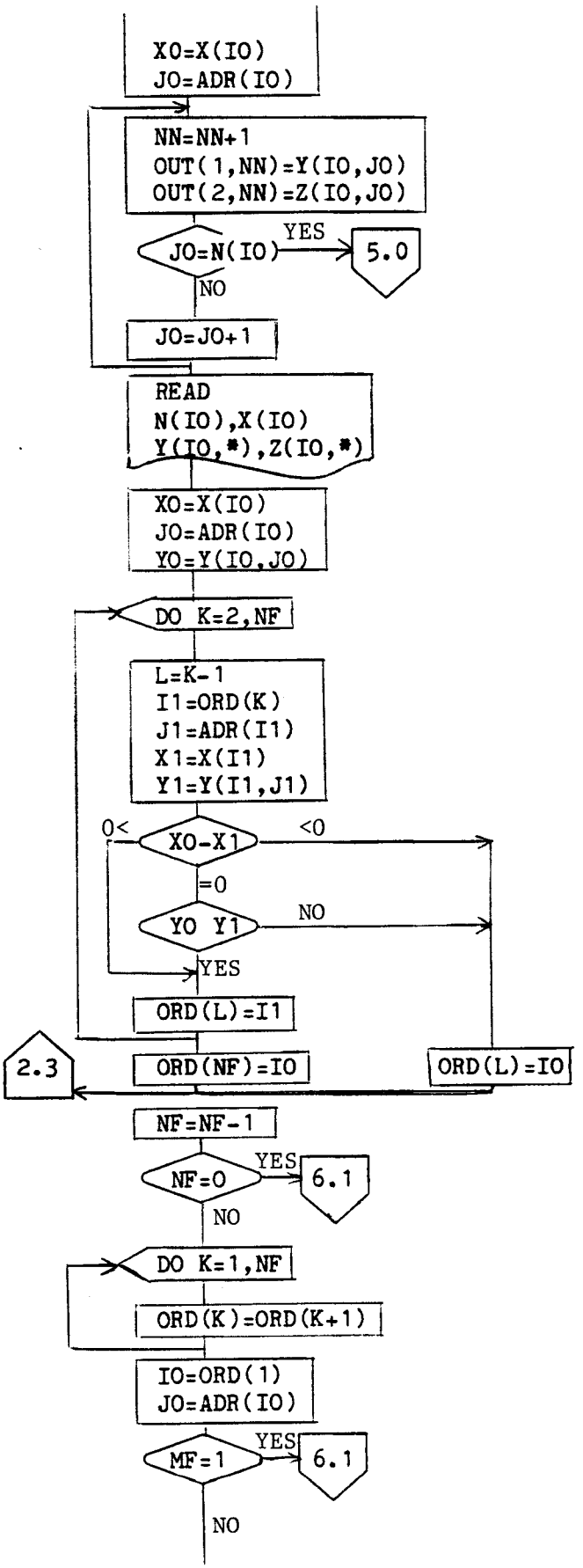


transfer all the data of that data set of the same X value to the output array.

5.0 Read the next data set.

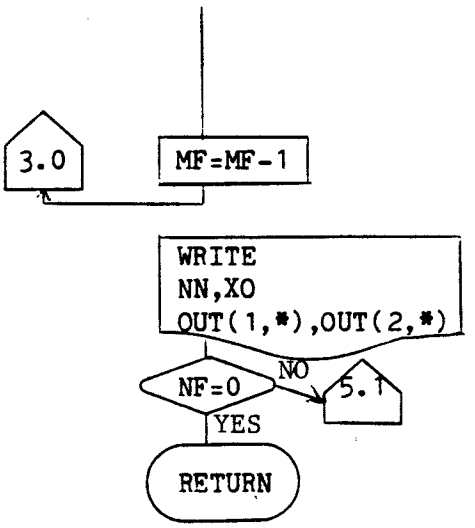
5.1 Compare the top record of the new data set with other top records of the input files to determine the position of the new data set.

6.0 When one file has been empty eliminate the file and rearrange the rest files.



6.1 All the data with X=XO has been sorted out, output them from output array to the disk file.

7.0 Return to TERSOR.



SUBROUTINE NAME : TERINF

TERINF

1.0 TERINF is called from TERSUP.

```
**********************************
*                                *
*  RASTERIZATION BEGINS          *
*                                *
**********************************
```

2.0 Initialization.

2.1 Call INIT to fill data in XDATA array.

3.0 Repeat the following procedure through 5.2 for all lines.

3.1 Read and fill data in YDATA array.

4.0 Repeat the following procedure through 4.3 for all pixels.

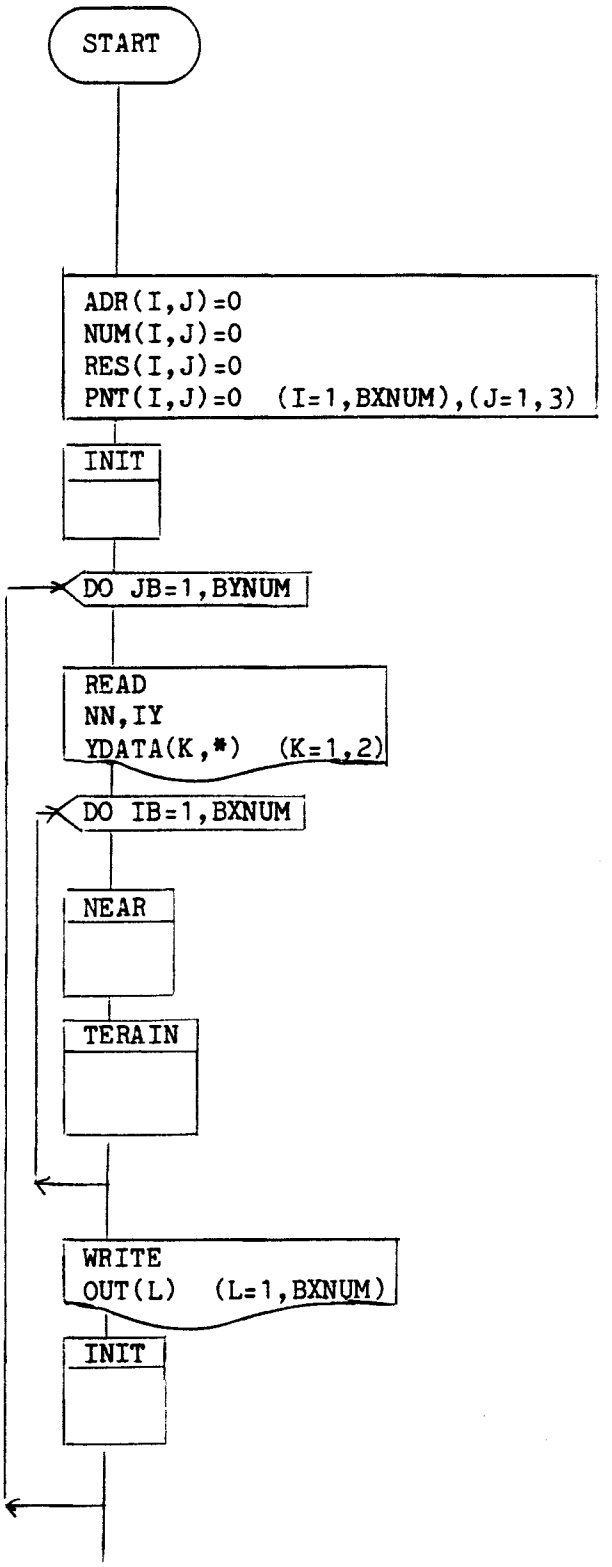
4.1 Call NEAR four times to find the points surrounding the pixel center.

4.2 Call TERAIN to calculate terrain information.

4.3 Go back to 4.0 and proceed to the next pixel.

5.0 Output one line result.

5.1 Call INIT to replace XDATA array for the next line.

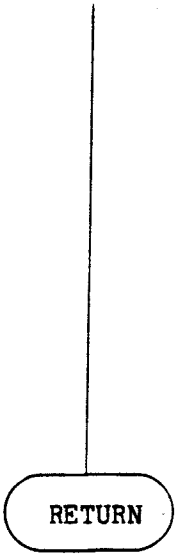


5.2 Go back to 3.0 and proceed
to the next line.

6.0

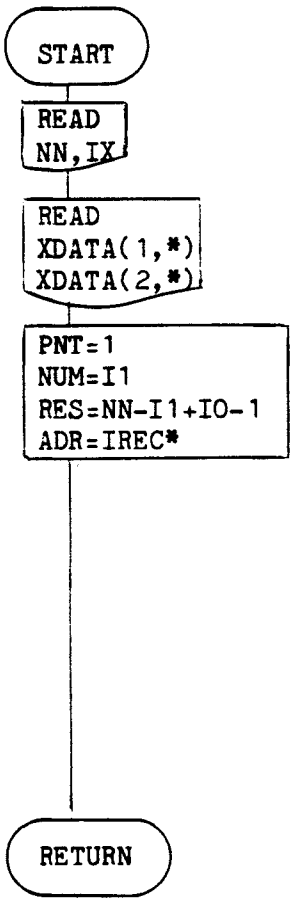
```
**********************************
*                                *
*  RASTERIZATION COMPLETED       *
*                                *
**********************************
```

Return to TERSUP.



SUBROUTINE NAME : INIT

INIT
1.0 INIT is called from TERINF. 2.0 Read point header. 2.1 Read point data. 3.0 Set up some variables. PNT : Current point number in the array. NUM : Number of points in the array. RES : Number of points of the line which exceed the memory limitation and could not be stored in the array. ADR : The address of the initial point of the remaining data (RES) in direct access file. (if RES=0 then ADR=0) 4.0 Return to TERINF.



SUBROUTINE NAME : NEAR

This subroutine NEAR is to find points necessary to calculate the terrain information. First the nearest two points, one for each side of the pixel center are chosen. And if their Z values are different, they are considered to be sufficient for the terrain information calculation. However, if their Z values are identical, one has to get the next nearest points until the point with different Z value is encountered. Searching of the points is performed on both sides of the pixel center alternatively.

NEAR

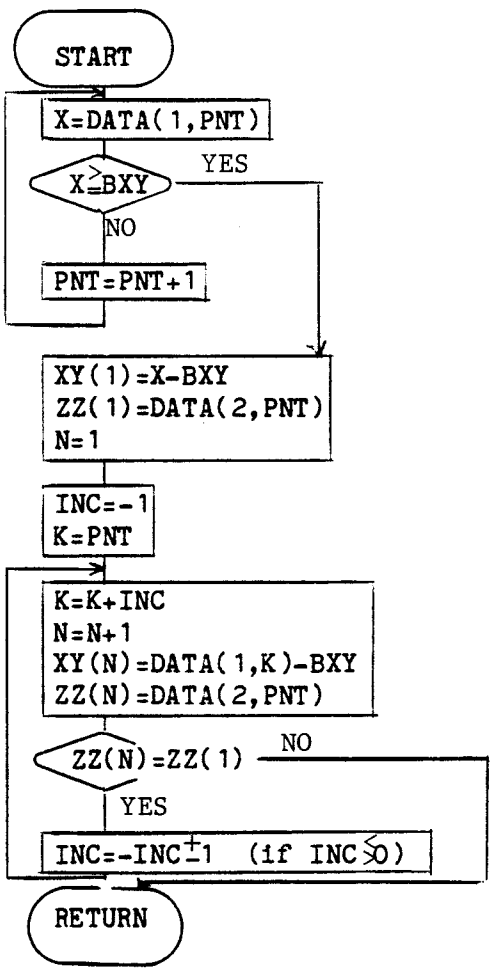
1.0 NEAR is called from TERINF.

2.0 Look for the point that first exceeds the pixel center.

2.1 Let the point that first exceeded the pixel center be the first data point.

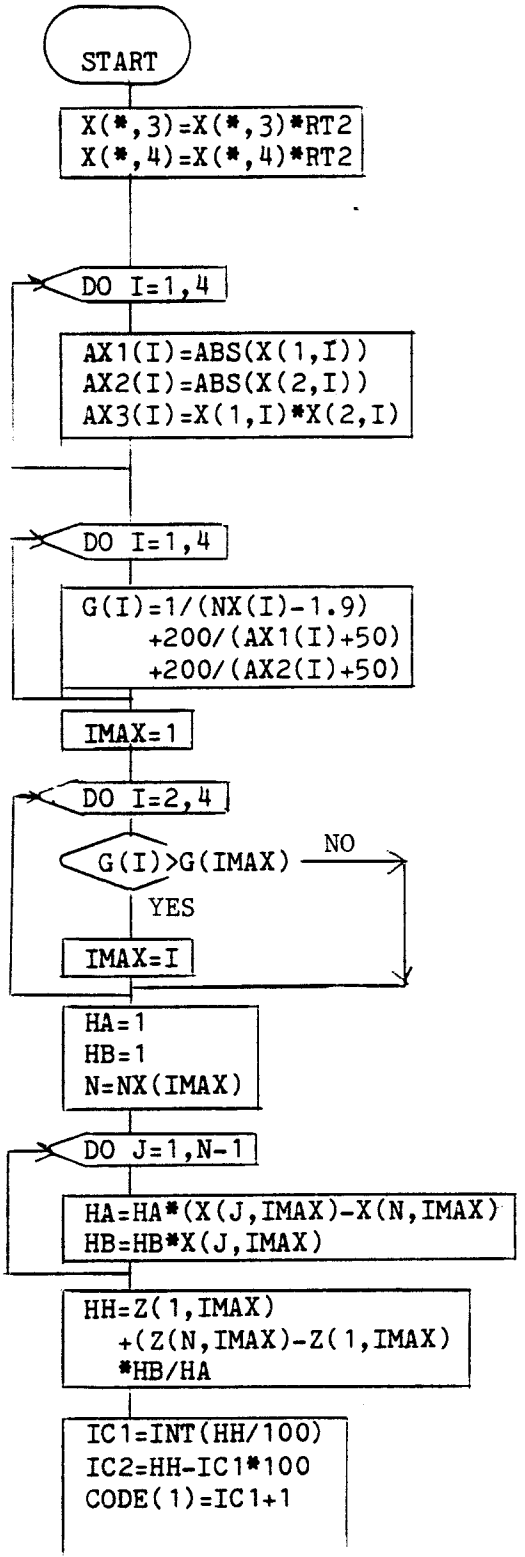
3.0 Look for other points.

4.0 Return to TERINF.



SUBROUTINE NAME : TERAIN

TERAIN
1.0 TERAIN is called from TERINF.
2.0 Diagonal line data are multiplied by RT2 to conform the unit length to horizontal and vertical lines.
3.0 Set up some variables.
4.0 Calculation of altitude.
4.1 Evaluate the quality of data distribution.
4.2 Select the optimum direction to calculate the altitude.
4.3 Altitude interpolation.
4.4 Coding of altitude.



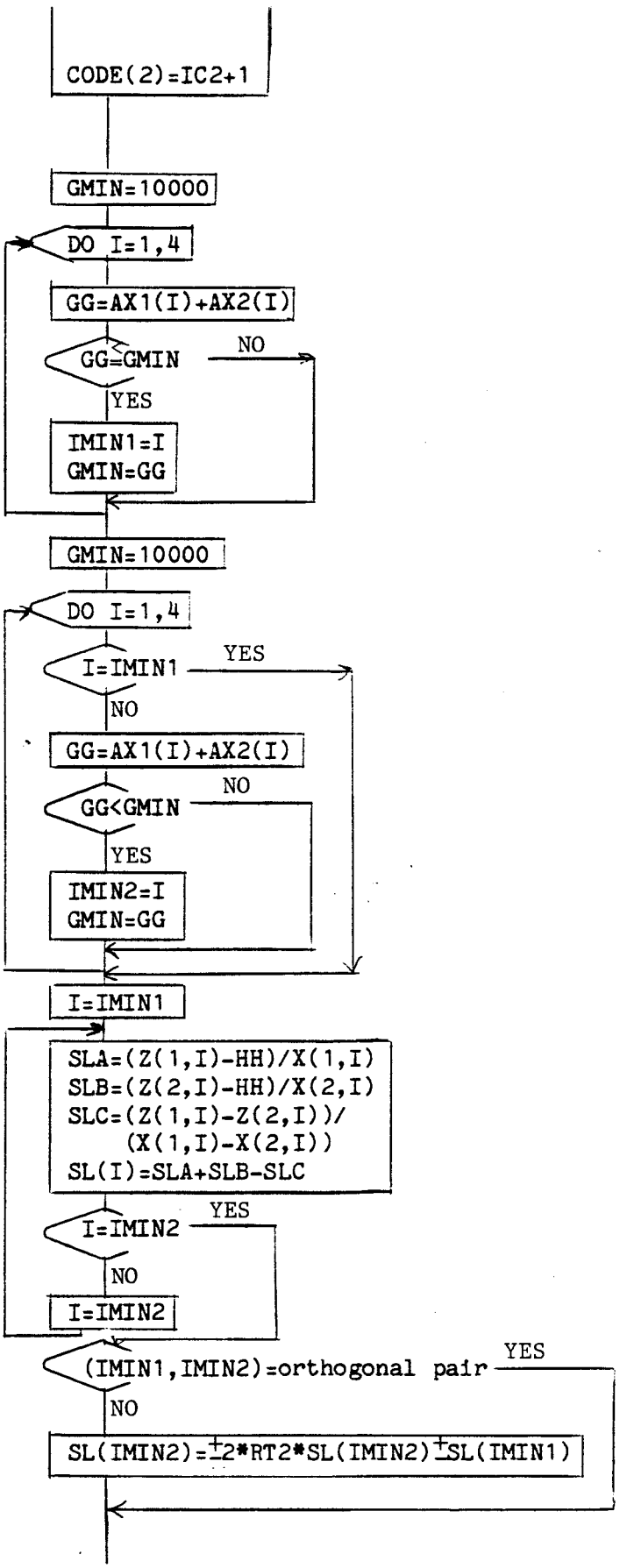
5.0 Calculation of slope and aspect.

5.1 Look for the direction of the minimum distance from the pixel center to the first two points.

5.2 Look for the direction of the second minimum distance.

5.3 Estimation of slope along the two directions.

5.4 Transform the pair of slope elements to the orthogonal pair.



5.5 Transform the orthogonal pair of slope elements to slope and aspect.

```
SLP=SL(IMIN1)**2
   +SL(IMIN2)**2
SLP=ATAN(SQRT(SLP))*DGRD
AS=ATAN2(SL(IMIN2),
         SL(IMIN1))
AS=AS*DGRD+ASO
```

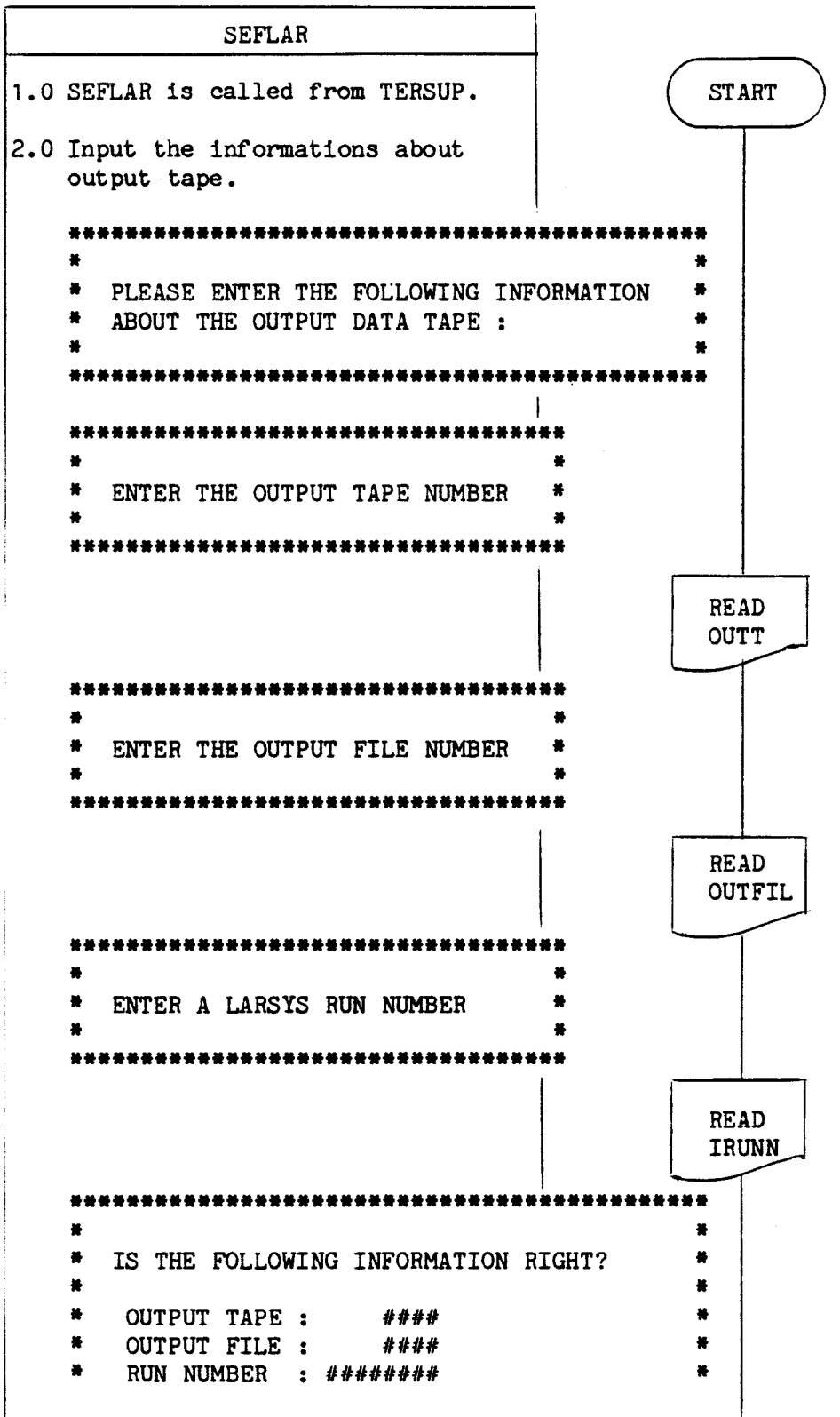
5.6 Coding of aspect and slope.

```
CODE(4)=AS*0.5+1
CODE(3)=function of SLP
```

6.0 Return to TERINF.

RETURN

SUBROUTINE NAME : SEFLAR



```
*                                                 *
*  ENTER ::  (1)  INPUTS OK...                    *
*  ENTER ::  (2)  INPUTS WRONG...                 *
*                                                 *
***************************************************
```

"1" → 3.0

"2" → 2.0

3.0 Preparation for tape output.

```
OUTSMP=BXNUM+CAL
ADJUST=MOD(OUTSMP,4)
ADD=4-ADJUST
OUTSMP=OUTSMP+ADD
CALBYT=OUTSMP-BXNUM
```

```
*********************************************************
*********************************************************
***                                                   ***
***  OUTPUT LINES WILL HAVE ###### SAMPLES            ***
***  PER LINE WITH ###### BYTES AT THE END            ***
***  BEING EQUAL TO ZERO......                        ***
***                                                   ***
*********************************************************
*********************************************************
```

3.1 MOUNT is called to ask to mount tape.

MOUNT

3.2 TOPRW is called to rewind tape.

TOPRW

3.3 TOPFF is called to position tape.

TOPFF

3.4 Set up ID.

```
ID(1)=OUTT
ID(2)=OUTFIL
ID(3)=IRUNN
ID(5)=4 : Channels
ID(21)=BMINY
ID(22)=BMINX
ID(6)=OUTSMP
ID(20)=BYNUM
```

3.5 TOPWR is called to write ID.

TOPWR

4.0

4.0 Write down on tape.

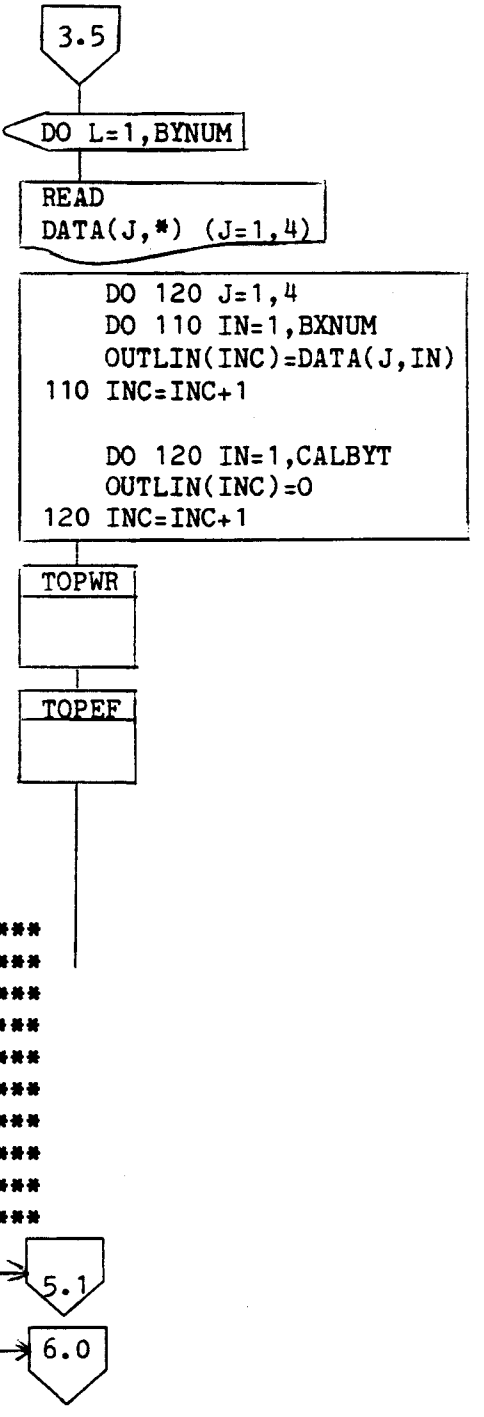
4.1 Read one line data from disk.

4.2 Transfer data to output array.

4.3 TOPWR is called to write on tape.

4.4 TOPEF is called twice to write EOT.

5.0 Create CC-file for CDISPLAY function.




```
**************************************************
**************************************************
***                                            ***
***  DO YOU WISH TO HAVE A CDISPLAY DECK       ***
***  CREATED FOR THIS IMAGE?                   ***
***     (1) YES                                ***
***     (2) NO                                 ***
***                                            ***
**************************************************
**************************************************
```

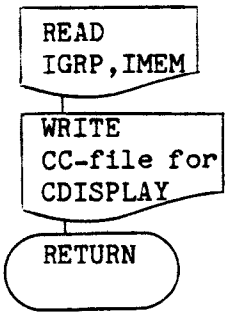
"1" → 5.1

"2" → 6.0

5.1 Information inputs

```
**********************************
**********************************
**                              **
**  ENTER A FILE NAME           **
**                              **
**********************************
**********************************
```


```
**********************************
**********************************
**                              **
**  ENTER PDP GROUP,MEMBER      **
**  (DEFAULT TO 300 300)        **
**                              **
**********************************
**********************************
```



5.2 Write CC-file.

6.0 Return to TERSUP.

Code Tables. The rasterized results are recorded on tapes in four channels in MIST format. Channel 1 and 2 contain the elevation data, where channel 1 contains the higher digits of the elevation in every one hundred meters, and channel 2 contains the lower two digits of elevation in every one meter. Channels 3 and 4 contain the slope and aspect information respectively.

The following are the codes for the four channels:

CHANNEL 1 : Higher digits of elevation.

CODE=(elevation/100)+1

CODE	elevation(meter) from	to
0	NO DATA	
1	0	99
2	100	199
3	200	299
4	300	399
5	400	499
6	500	599
7	600	699
8	700	799
9	800	899
10	900	999
.....		
80	7900	7999
254	ERROR DATA	
255	OUT OF THE COUNTRY.	

CHANNEL 2 : Lower digits of elevation.

CODE=MOD(elevation,100)+1

CODE	elevation(meter)
0	NO DATA
1	0
2	1
3	2
4	3
5	4
6	5
7	6
8	7
9	8
10	9
.....	
100	99
254	ERROR DATA
255	OUT OF THE COUNTRY.

CHANNEL 3 : Slope.

CODE	slope(degree) from	to	slope(%) from	to
0	NO DATA			
1	0.0	0.5	0	1
2	0.5	1.0	0	1
3	0.5	1.0	1	2
4	1.0	1.5	1	2
5	1.0	1.5	2	3
6	1.5	2.0	2	3
7	1.5	2.0	3	4
8	2.0	2.5	3	4
9	2.0	2.5	4	5
10	2.5	3.0	4	5
11	2.5	3.0	5	6
12	3.0	3.5	5	6
13	3.0	3.5	6	7
14	3.5	4.0	6	7
15	3.5	4.0	7	8
16	4.0	4.5	7	8
17	4.5	5.0	7	8
18	4.5	5.0	8	9
19	5.0	5.5	8	9
20	5.0	5.5	9	10
21	5.5	6.0	9	10
22	5.5	6.0	10	11
23	6.0	6.5	10	11
24	6.0	6.5	11	12
25	6.5	7.0	11	12
26	6.5	7.0	12	13
27	7.0	7.5	12	13
28	7.0	7.5	13	14
29	7.5	8.0	13	14
30	7.5	8.0	14	15
31	8.0	8.5	14	15
32	8.5	9.0	14	15
33	8.5	9.0	15	16
34	9.0	9.5	15	16
35	9.0	9.5	16	17
36	9.5	10.0	16	17
37	9.5	10.0	17	18
38	10.0	10.5	17	18
39	10.0	10.5	18	19
40	10.5	11.0	18	19
41	10.5	11.0	19	20
42	11.0	11.5	19	20
43	11.0	11.5	20	21
44	11.5	12.0	20	21
45	11.5	12.0	21	22
46	12.0	12.5	21	22
47	12.0	12.5	22	23
48	12.5	13.0	22	23
49	12.5	13.0	23	24
50	13.0	13.5	23	24
51	13.0	13.5	24	25
52	13.5	14.0	24	25
53	14.0	14.5	24	25
54	14.0	14.5	25	26
55	14.5	15.0	25	26
56	14.5	15.0	26	27
57	15.0	15.5	26	27
58	15.0	15.5	27	28
59	15.5	16.0	27	28
60	15.5	16.0	28	29
61	16.0	16.5	28	29
62	16.0	16.5	29	80
63	16.5	17.0	29	30
64	16.5	17.0	30	31
65	17.0	17.5	30	31
66	17.0	17.5	31	32
67	17.5	18.0	31	32
68	17.5	18.0	32	33
69	18.0	18.5	32	33
70	18.0	18.5	33	34

CHANNEL 3 : Slope. (Cont.)

CODE	slope(degree) from	to	slope(%) from	to
71	18.5	19.0	33	34
72	18.5	19.0	34	35
73	19.0	19.5	34	35
74	19.0	19.5	35	36
75	19.5	20.0	35	36
76	19.5	20.0	36	37
77	20.0	20.5	36	37
78	20.0	20.5	37	38
79	20.5	21.0	37	38
80	20.5	21.0	38	39
81	21.0	21.5	38	39
82	21.0	21.5	39	40
83	21.5	22.0	39	40
84	21.5	22.0	40	41
85	22.0	22.5	40	41
86	22.0	22.5	41	42
87	22.5	23.0	41	42
88	22.5	23.0	42	43
89	23.0	23.5	42	43
90	23.0	23.5	43	44
91	23.5	24.0	43	44
92	23.5	24.0	44	45
93	24.0	24.5	44	45
94	24.0	24.5	45	46
95	24.5	25.0	45	46
96	24.5	25.0	46	47
97	25.0	25.5	46	47
98	25.0	25.5	47	48
99	25.5	26.0	47	48
100	25.5	26.0	48	49
101	26.0	26.5	48	49
102	26.0	26.5	49	50
103	26.5	27.0	49	50
104	26.5	27.0	50	52
105	27.0	27.5	50	52

CODE	slope(degree) from	to	slope(%) from	to
106	27.0	27.5	52	54
107	27.5	28.0	52	54
108	28.0	28.5	52	54
109	28.0	28.5	54	56
110	28.5	29.0	54	56
111	29.0	29.5	54	56
112	29.0	29.5	56	58
113	29.5	30.0	56	58
114	30.0	31.0	56	58
115	30.0	31.0	58	60
116	30.0	31.0	60	62
117	31.0	32.0	60	62
118	31.0	32.0	62	64
119	32.0	33.0	62	64
120	32.0	33.0	64	66
121	33.0	34.0	64	66
122	33.0	34.0	66	68
123	34.0	35.0	66	68
124	34.0	35.0	68	70
125	34.0	35.0	70	72
126	35.0	36.0	70	72
127	35.0	36.0	72	74
128	36.0	37.0	72	74
129	36.0	37.0	74	76
130	37.0	38.0	74	76
131	37.0	38.0	76	78
132	37.0	38.0	78	80
133	38.0	39.0	78	80
134	38.0	39.0	80	82
135	39.0	40.0	80	82
136	39.0	40.0	82	84
137	40.0	41.0	82	84
138	40.0	41.0	84	86
139	40.0	41.0	86	88
140	41.0	42.0	86	88

CHANNEL 3 : Slope. (Cont.)

CODE	slope(degree)		slope(%)	
	from	to	from	to
141	41.0	42.0	88	90
142	41.0	42.0	90	92
143	42.0	43.0	90	92
144	42.0	43.0	92	94
145	43.0	44.0	92	94
146	43.0	44.0	94	96
147	43.0	44.0	96	98
148	44.0	45.0	96	98
149	44.0	45.0	98	100
150	45.0	46.0	100	105
151	46.0	47.0	100	105
152	46.0	47.0	105	110
153	47.0	48.0	105	110
154	47.0	48.0	110	115
155	48.0	49.0	110	115
156	48.0	49.0	115	120
157	49.0	50.0	115	120
158	50.0	51.0	115	120
159	50.0	51.0	120	125
160	51.0	52.0	120	125
161	51.0	52.0	125	130
162	52.0	53.0	125	130
163	52.0	53.0	130	135
164	53.0	54.0	130	135
165	53.0	54.0	135	140
166	54.0	55.0	135	140
167	54.0	55.0	140	145
168	55.0	56.0	140	145
169	55.0	56.0	145	150
170	56.0	57.0	145	150
171	56.0	57.0	150	160
172	57.0	58.0	150	160
173	57.0	58.0	160	170
174	58.0	59.0	160	170
175	59.0	60.0	160	170
176	59.0	60.0	170	180
177	60.0	62.0	170	180
178	60.0	62.0	180	190
179	62.0	64.0	180	190
180	62.0	64.0	190	200
181	62.0	64.0	200	210
182	64.0	66.0	200	210
183	64.0	66.0	210	220
184	64.0	66.0	220	230
185	66.0	68.0	220	230
186	66.0	68.0	230	240
187	66.0	68.0	240	250
188	68.0	70.0	240	250
189	68.0	70.0	250	260
190	68.0	70.0	260	270
191	68.0	70.0	270	280
192	70.0	75.0	270	280
193	70.0	75.0	280	290
194	70.0	75.0	290	300
195	70.0	75.0	300	350
196	70.0	75.0	350	400
197	75.0	80.0	350	400
198	75.0	80.0	400	450
199	75.0	80.0	450	500
200	75.0	80.0	500	...
201	80.0	85.0	500	...
202	85.0	90.0	500	...
254	ERROR DATA			
255	OUT OF THE COUNTRY.			

CHANNEL 4 : Aspect.

Azimuth of the steepest slope measured clockwise from the north in degrees.

CODE=(aspect(degree)/2)+1

CODE	aspect(degree) from	to
0	NO DATA	
1	0	2
2	2	4
3	4	6
4	6	8
5	8	10
6	10	12
7	12	14
8	14	16
9	16	18
10	18	20
.....		
180	358	360
254	ERROR DATA	
255	OUT OF THE COUNTRY.	

DATA BASE QUERY CAPABILITIES

One of the most important features of a digital Geographic Information System is its query capabilities, which can provide answers to a range of questions (from very simple to very complex questions). In order to answer very complex questions, mathematical models (algorithms) would have to be developed and implemented as part of the Modeling or Analysis Subsystem of the GIS.

To answer simple questions, one can use logic operations of the AND, OR type (Boolean Algebra). To demonstrate how a simple query capability would answer simple questions from the Bolivian data base, the following eight sample questions were posed to the Bolivian GIS. The procedures to obtain the answers are also described in the following paragraphs:

Questions.

1. What is the area of the Poopo Province? What percent of the Oruro Department is this area?

2. In which province is the canton Totora located?

3. What is the area of Lake Popo from the land use element and from the soils element?

4. How many cantons and provinces contain a part of Lake Popo? What is the area of the lake in each of these?

5. Make a table of all provinces which contain barren lands and show the

area of each kind of barren land in these provinces.

6. Create three tables showing the areas of soils, land use, and watersheds in the canton Chipaya.

7. For the land use - crop lands - what kinds of soils exist and what are their areas?

8. What is the area of crop lands (land use) in footslopes (soils) in Salinas des Garci Mendoza (canton)? In which watershed or watersheds are these located?

Answers.

The method of answering these questions and the derived answers are as follows:

1. Q. What is the area of the Poopo Province? What percent of the Oruro Department is this area?

   A. This question can be answered by getting a map of the political boundaries and also a dump of the political boundaries attribute record. From the dump one can find out which cantons are in the Poopo Province and then get the areas of each of these cantons from the histogram at the end of the map of the political boundaries. The total area of the Oruro department can also be obtained from the histogram at the end of the map.

   Area of Poopo province = 198,950 hectares

Area of Oruro department = = 5,187,900 hectares

. . Poopo is 3.83% of the Oruro department.

2. Q. In which province is the canton Totora located?

A. This answer can be found directly from a dump of the political boundaries attribute file.

Totora is located in the Sajama province.

3. Q. What is the area of Lake Popo from the land use element and from the soils element?

A. This question can be answered by making a map of the land use and a map of the soils with the boundaries of lines 1950 to 2150 and columns 550 to 850 (this isolates Lake Popo). The area to the lake will be the area on the map histograms that is assigned to lakes.

The area of Lake Popo from the land use is 228,525 hectares.

The area of Lake Popo from the soils element is 227,475 hectares.

4. Q. How many cantons and provinces contain a part of Lake Popo? What is the area of the lake in each of these?

A. This question can be answered by merging the land use and political boundaries images together to form a third image. The programs DRIVER FORTRAN and IMLOG FORTRAN accomplish this by ANDing or ORing two images together for a specified set of fill characters in each image. Depending on the AND or OR operation the fill character from the first

image (this first image is named L0500Q10 IMAGE2) or a zero is written out to the third image (L0500Q10 IMAGE3). This third image can then be renamed to correspond with a certain attribute file and then a map can be made.

Thus to answer this question one runs DRIVEF EXEC (this exec loads IMLOG FORTRAN and DRIVER FORTRAN and starts them). The first step was to rename the elements L0500Q10 politica to L0500Q10 IMAGE1 and L0500Q10 LANDCUS to L0500Q10 IMAGE2. Then specify an OR operation using no fill characters in the first image and fill character 17 for the second image (this fill character corresponds to lake in the land use image). The programs output L0500Q10 IMAGE3 which was renamed to L0500Q10 politica so the image would correspond to the political boundaries attribute file. Subsequently one makes a political boundaries map using the same lines and columns as in question 3 to isolate lake Poopo. The histogram of this map shows the cantons and their areas that are part of Lake Poopo.

From the map of the merged images:

Cantons which Contain Lake Popo	Area of canton in Lake Popo
1) Anacacato	28,475 hectares
2) Pampa Aullagas	15,000 hectares
3) El Choro	17,275 hectares
4) Condo K	3,050 hectares
5) Andamarca	19,925 hectares
6) Pazna	45,075 hectares
7) Quillacas	5,325 hectares
8) Orinoca	33,525 hectares
9) Huancane	500 hectares
10) Santiago De Huari	20,200 hectares
11) Poopo	900 hectares

12) Utavi	37,000 hectares
13) Guadalupe	2,275 hectares

Also from the dump of the political boundaries attribute record that is attached to question 1 one can find the provinces that these cantons belong to.

Provinces which contain Lake Popo	Area of province in Lake Popo
1) Cercado	17,275 hectares
2) Avaroa	59,825 hectares
3) Carangas	53,450 hectares
4) Poopo	45,975 hectares
5) Ladislao Cabrera	15,000 hectares
6) Saucari	37,000 hectares

5. Q. Make a table of all provinces which contain barren lands and show the area of each kind of barren land in these provinces.

A. To make this table one had to merge a land use image and a political boundaries image together for each province. For example, to get the land use for the province Cercado one runs the driver exec with land use as IMAGE 1 and political boundaries as IMAGE 2. Then one specified an OR function with the fill characters 60 - 68 and 244 specified (these are the fill characters assigned to the cantons in Cercado) and no fill characters specified for the land use.

After obtaining the map of the province one can find which fill characters were assigned to barren lands from a dump of the land use attribute file (see attached output). From this information one is able to get the areas of barren lands that were assigned to each province, as illustrated in Table 1.

Table 1. Barren Lands (Hectares).

PROVINCE	ROCK OUTCROPS AND RANGE AND/OR SHRUBLAND IN DRY ENVIRONMENT	SALT FLAT	SALINE LANDS	SANDY AREAS	ROCK OUTCROPS	TOTAL BARREN LANDS
CERCADO	1,600	0	3,500	225	0	5,325
ATAHUALLPA	12,150	103,900	64,275	1,325	25	181,675
LADISLAO CABRERA	10,900	59,500	77,900	0	5,650	153,950
SAJAMA	7,150	0	38,675	800	5,525	52,150
POOPO	0	0	0	0	0	0
AVAROA	0	0	5,600	0	0	5,600
SAUCARI	0	0	41,675	15,050	0	56,725
LITORAL	0	0	50,625	250	0	50,875
PANTALEON DALENCE	0	0	275	0	0	275
CARANGAS	5,700	0	67,600	21,475	0	94,775
TOTAL	37,500	163,400	350,125	39,125	11,200	601,350

6. Q. Create three tables showing the areas of soils, land use, and watersheds in the canton Chipaya.

A. These tables were obtained by running the DRIVER EXEC with soils as Image 1 and political boundaries as Image 2. Then one specifies an OR function with no fill characters specified for Image 1 and fill character 18 specified for Image 2 (18 is the fill character Chipaya is assigned). Then we renamed L0500Q10 Image 3 to L0500Q10 soils and made a map. The map's histogram contained all the information that was needed to make the soils table. Using a similar procedure one can produce the land use and watersheds tables.

Soils in the canton Chipaya

1)	Temporary flooded depressions	6,275 hectares
2)	Lakes	5,650 hectares
3)	Delta Plains	14,100 hectares
4)	Moderately dissect volcanic plain	9,925 hectares
5)	Sand fields - established dunes	300 hectares
6)	Alluvial fans	8,325 hectares
	Total	44,575 hectares

Land Use in the canton Chipaya

1)	Lakes	6,700 hectares
2)	Range/shrubland in temporary wet environment and range/shrubland in saline lands	200 hectares
3)	Rivers	950 hectares
4)	Range/shrubland in temporary wet environment and sandy areas	450 hectares
5)	Saltflat	5,175 hectares
6)	Saline lands	20,325 hectares
7)	Range and/or shrubland in saline lands	1,400 hectares
8)	Range/shrubland in saline land and sandy areas	75 hectares
9)	Permanent wet and/or floodedland	8,475 hectares
10)	Highways, roads, railroads, pipelines, etc.	825 hectares
	Total	44,575 hectares

Watersheds in the canton Chipaya

1) Lauca	29,500 hectares
2) Coipasa Saltflat	7,750 hectares
3) Barras	4,300 hectares
4) Sabaya	2,975 hectares
Total	44,575 hectares

7. Q. For the land use - crop lands - what kinds of soils exist and what are their areas?

A. This question can be answered by merging a land use image with a soils image. One runs the driver routine with soils as Image 1 and lands use as Image 2. Then one has to specify an OR function with no fill characters specified for Image 1 and fill characters 13 and 14 specified for Image 2. Subsequently one renames L0500Q10 IMAGE3 to L0500Q10 SOILS and then one makes a soils map. The histogram of this map has the information to answer this question. (Note: The total area from the histogram of the attached map is 163,900 hectares where the total croplands area from a land use map is 163,925 hectares. Thus the soils image has one point unassigned that is assigned in the land use image).

8. Q. What is the area of crop lands (land use) in footslopes (soils) in the Salinas de Garci Mendoza (canton)? In which watershed or watersheds are these located?

A. This can be answered by merging the land use image with the soils image and then merging the resulting image to make a land use map to answer the first part of the question. The second part of the question

can be answered by merging the image from the first part with the hydrology image and then making a hydrology map using the resulting image.

One has to run the driver exec with land use as Image 1 and soils as Image 2. Then one specifies an AND function with fill characters 13 and 14 (crop lands) specified for Image 1 and fill characters 42 and 46 (footslopes) specified for Image 2. Subsequently one renames L0500Q10 IMAGE3 to L0500Q10 IMAGE1 and set political boundaries to Image 2. Then one runs the driver exec again and specifies an OR function with no fill characters specified for Image 1 and fill character 34 (salinas de Garci Mendoza) specified for Image 2. Then rename Image 3 to Image 2 and set hydrology to Image 1. Then one runs the DRIVER EXEC again and specifies an OR function with no fill characters specified for Image 1 and fill characters 13 and 14 (the crop lands still remaining on Image 2) for Image 2. Then make a land use map using Image 2 to answer the first part of the question and a hydrology map using Image 3 to answer the second part of the question.

- The area of crop lands in footslopes in Salinas de Garci Mendoza is 24,1075 hectares.

- These are located in the following watersheds: Salar De Coipasa, Cancosa, Totora, Wiljahuira, Taman Khasa, Waycojahuira.

APPLE II Plus SOFTWARE CONVERSION

As indicated in page 17 and Figure 6 of the first quarterly progress report (LARS Contract Report 110180) the original hardware configuration specified to support the Bolivian GIS included a DEC PDP 11/34 minicomputer, because the Bolivian ERTS/GEOBOL Program was in the process of obtaining the funds to purchase this type of minicomputer. Therefore, all the computer programs (software) to support the Input Subsystem were developed for and implemented in a PDP 11/34 minicomputer, as documented in Figure 3 and pages 25 through 35 of the LARS Contract Report 080181 - Part I. However, due to unforeseen events in Bolivia, ERTS/GEOBOL was not able to acquire a PDP 11/34 minicomputer. Nevertheless, the Bolivian ERTS/GEOBOL Program purchased an APPLE II Plus microprocessor after most of the software development for the PDP 11/34 was completed.

In order to provide Bolivia with a digitizing capability (Input Subsystem) and after discussions between Purdue/LARS and ERTS/GEOBOL personnel, the decision was made to convert the already developed PDP 11/34 software into a set of computer programs for the APPLE II Plus microprocessor.

Three major problems were encountered in the conversion of the routines that interface a DEC PDP 11/34 minicomputer to a Talos table digitizer. The first and most difficult problem was replacing the direct input/output routines supplied by the DEC operating system with APPLE assembly language routines. The BASIC language supplied with the APPLE microcomputer is too slow to handle the data rate of the table digitizer,

so the receiving and buffering of data from the table digitizer required assembly language routines. The second problem encountered was that the smaller amount of memory on the APPLE microcomputer required the single, large multifunction routine on the 11/34 to be broken down into many smaller routines on the APPLE microcomputer. The smaller routines then had to be tied together in a logical manner to provide ease of use for the operator and continuity between the routines. The third problem was the transfer of digitized data from the APPLE microcomputer to a main-frame host computer for analysis. The DEC PDP 11/34 provided utility routines to format and transfer the digitized data to a larger host computer at a high data rate. The APPLE microcomputer did not have these utility routines, and the data rate to the host computer could not be as great as the PDP 11/34. Routines to transfer the digitized data from the APPLE microcomputer to the host computer were written and algorithms to compress the data were developed to increase the effective data rate to the host computer.

The digitizing routines written in APPLE II BASIC and ASSEMBLY computer languages are included in Appendix A of this report.

PRODUCTS TO BE DELIVERED TO ERTS/GEOBOL

The first delivery of products from this project was made in August 1981, when the Bolivian visiting computer scientists returned to Bolivia with a set of computer tapes containing the software developed to support the Input Subsystem and the corresponding Users and System's documentation (LARS Contract Reports 080181 and 080281).

These tapes contained the following computer programs for both the PDP 11 minicomputer and an IBM 4341 main-frame computer:

PROGRAM: DIGIT

PDP-11 FORTRAN SUBROUTINE LIBRARY

PROGRAM: LITER

PROGRAM: ARCLOSE

PROGRAM: EDITMAP

PROGRAM: FINCON

PROGRAM: MLTFIT

MODULE: BDCVAL

MODULE: CTLWRD

MODULE: IDNAME

MODULE: LOC

PROGRAM: MLTFIT PRINTOUTS

PROGRAM: FINBAK

PROGRAM: PROJECTI

PROGRAM: BOUNDRY

MODULE: MOUNT

MODULE: MOVBYT

MODULE: TAPOP

RASTERIZATION ALGORITYM

Landsat MSS Digital Mosaic. The Landsat MSS digital mosaic of the Oruro Department is being delivered to ERTS/GEOBOL in a set of 12 (1600 BPI) computer tapes containing 16 quadrangles of 2000 by 2000 pixels. Each of these pixels covers an area on the ground equal to 0.25 hectares (50 x 50 meters). The tape number, quadrangle number, and the spectral band information for the Oruro Landsat MSS digital mosaic is given in Table 2.

Table 2. List of Tapes Containing the Digital Landsat Mosaic of Oruro, Bolivia.

Tape No.	Quadrangle No.	Spectral Band (μm)
5616	1 - 6	0.5 - 0.6
3167	1 - 6	0.6 - 0.7
4093	1 - 6	0.7 - 0.8
3828	1 - 6	0.8 - 1.1
5617	7 - 12	0.5 - 0.6
5280	7 - 12	0.6 - 0.7
4094	7 - 12	0.7 - 0.8
3833	7 - 12	0.8 - 1.1
5618	13 - 16	0.5 - 0.6
3331	13 - 16	0.6 - 0.7
4095	13 - 16	0.7 - 0.8
4096	13 - 16	0.8 - 1.1

Figure 11 shows the color infrared composite image of the entire Landsat MSS digital mosaic covering the Oruro Department, and Figures 12 through 27 show the 16 individual mosaic quadrangles. In addition, two enlarged color infrared prints of the digital mosaic (approximate scale of 1:1,000,000) are also being delivered to ERTS/GEOBOL.

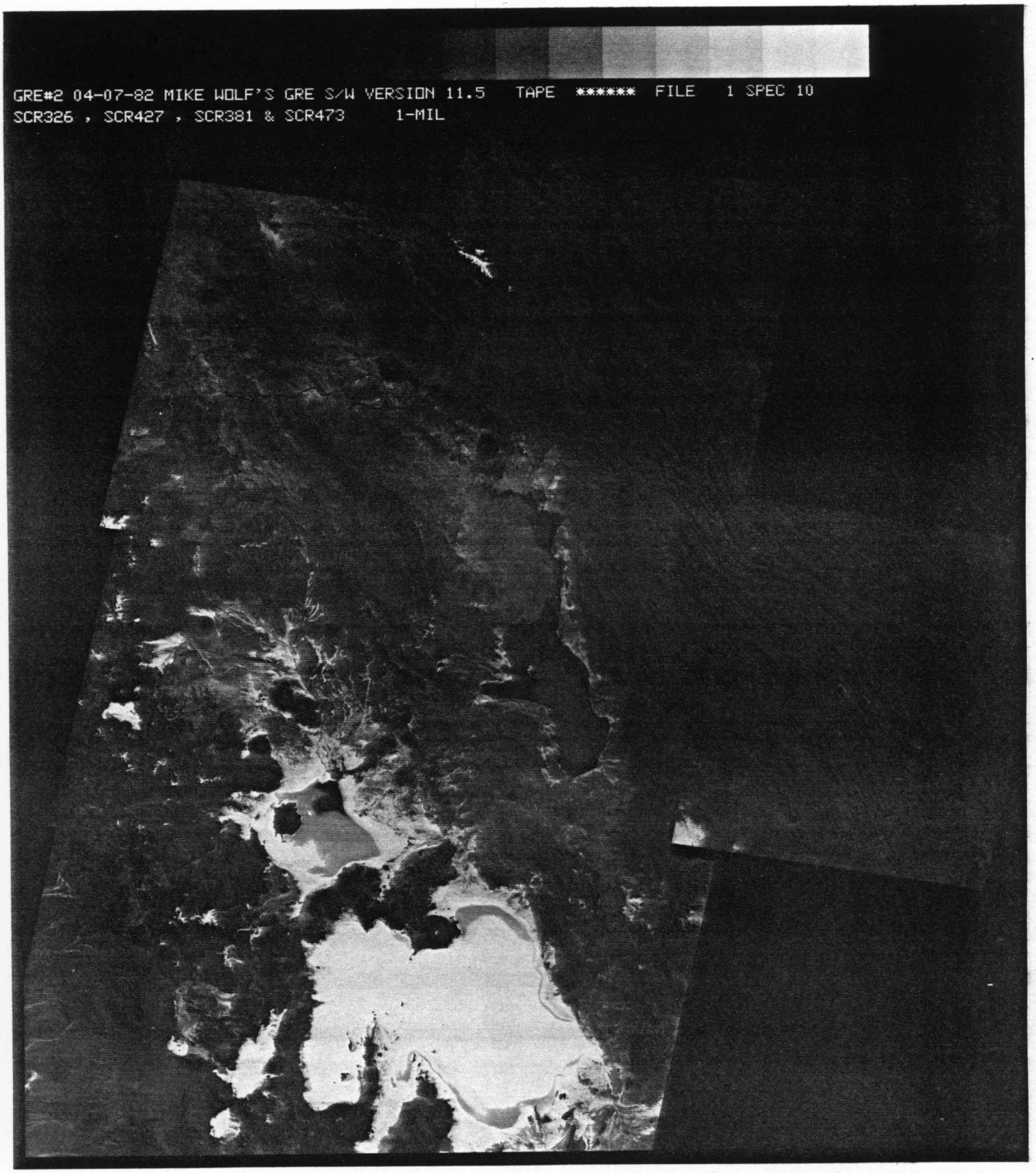


Figure 11. Color Infrared Composite image of the digital Landsat MSS mosaic of the Oruro Department.

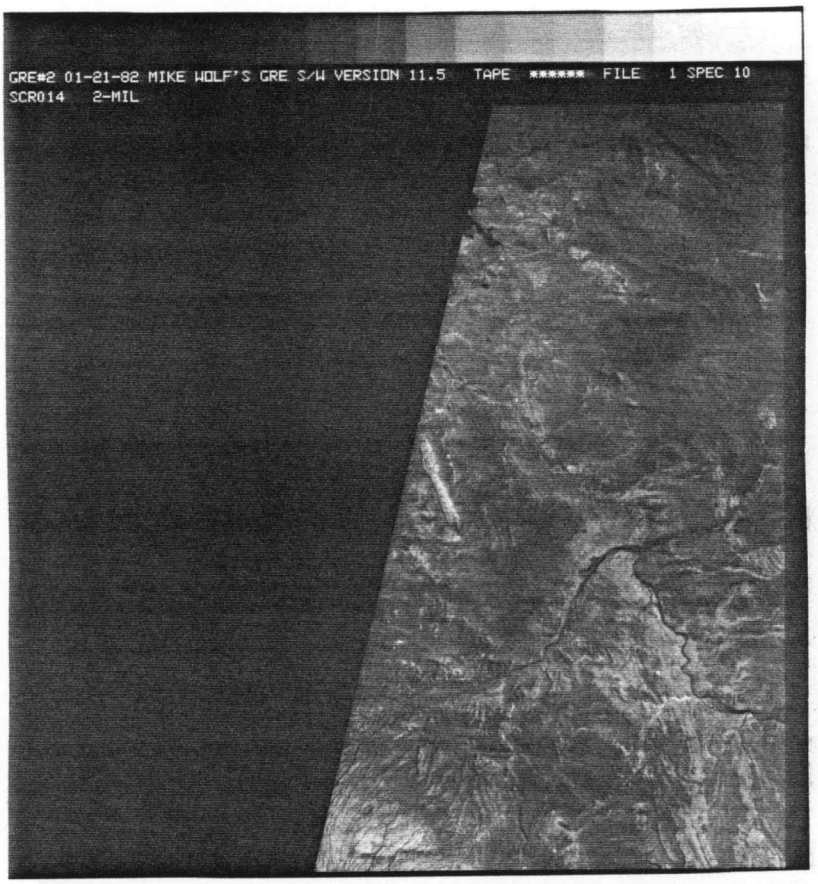


Figure 12. Digital Landsat mosaic quadrangle No. I.

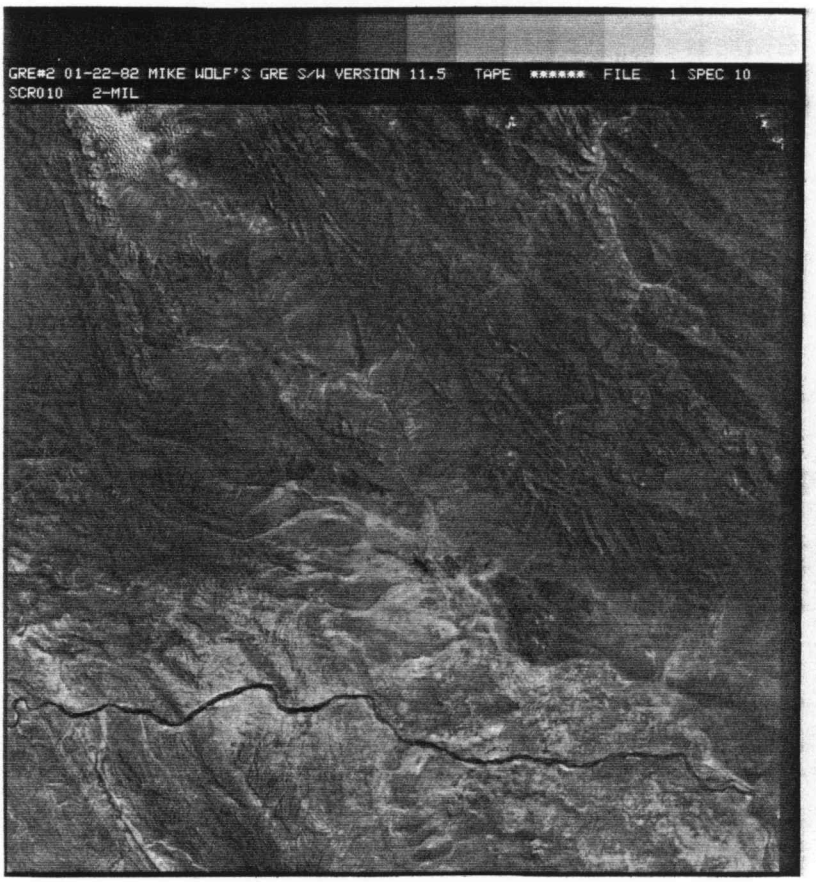


Figure 13. Digital Landsat mosaic quadrangle No. II.

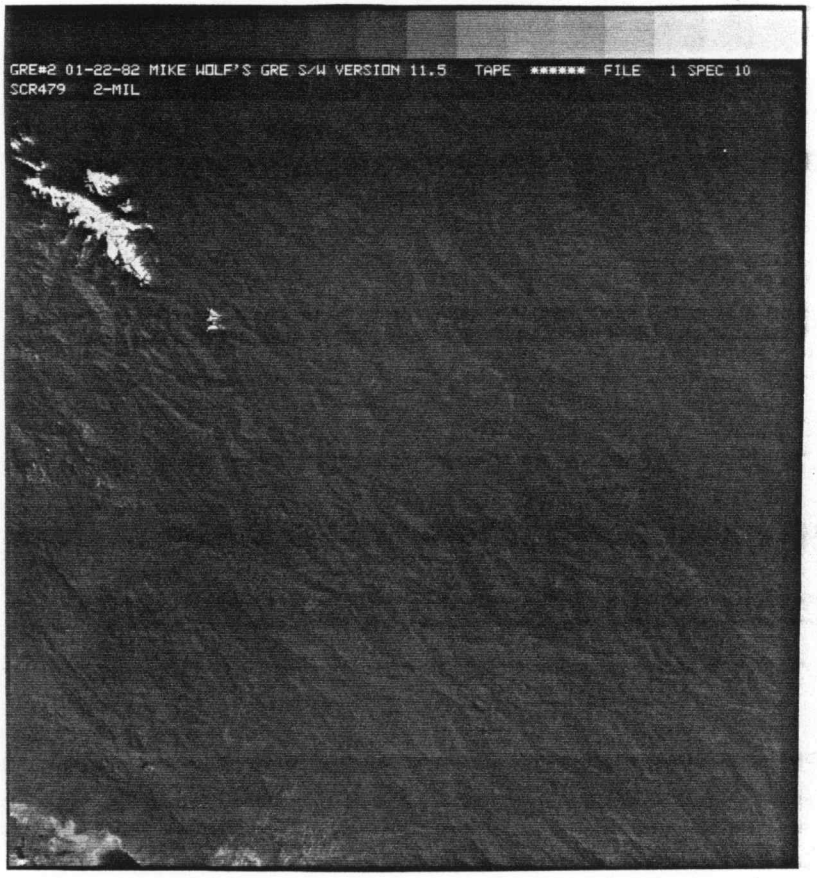


Figure 14. Digital Landsat mosaic quadrangle No. III.

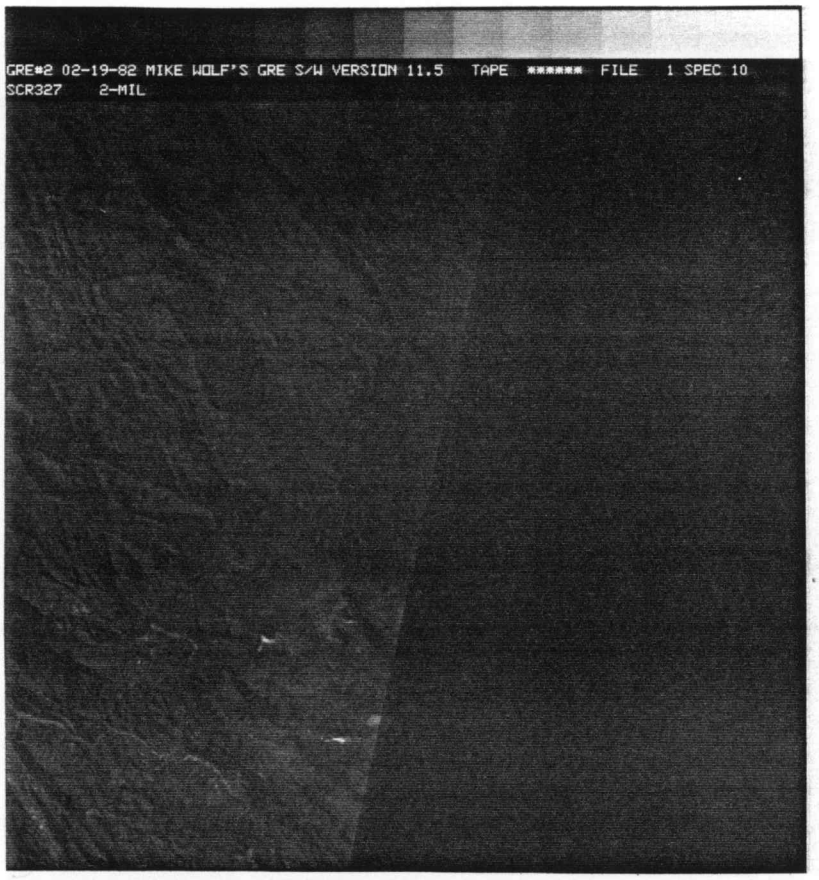


Figure 15. Digital Landsat mosaic quadrangle No. IV.

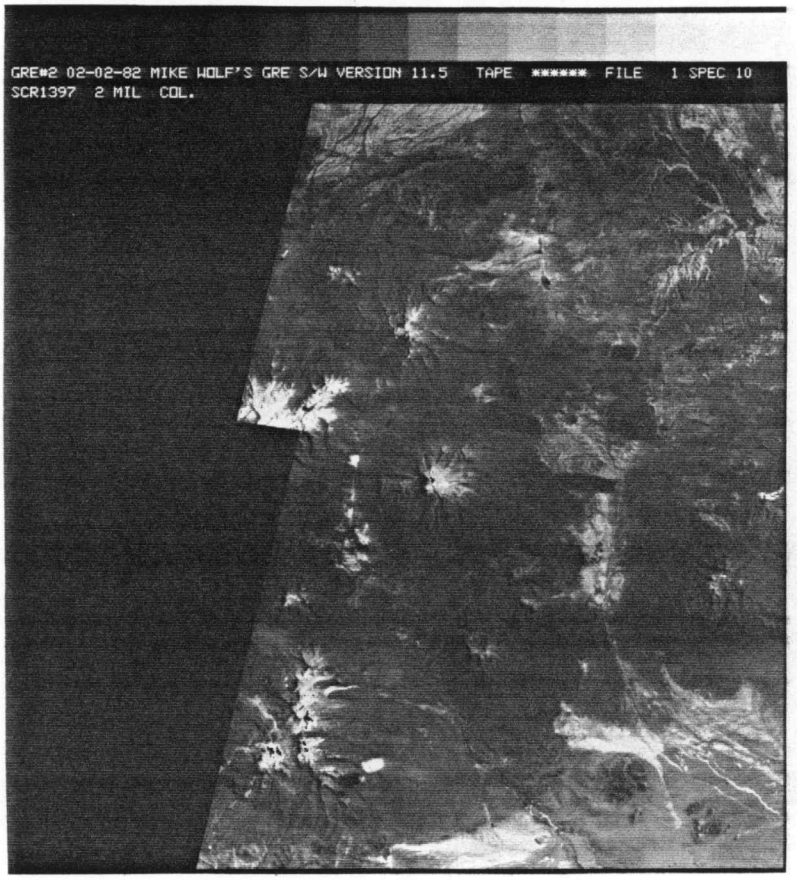


Figure 16. Digital Landsat mosaic quadrangle No. V.

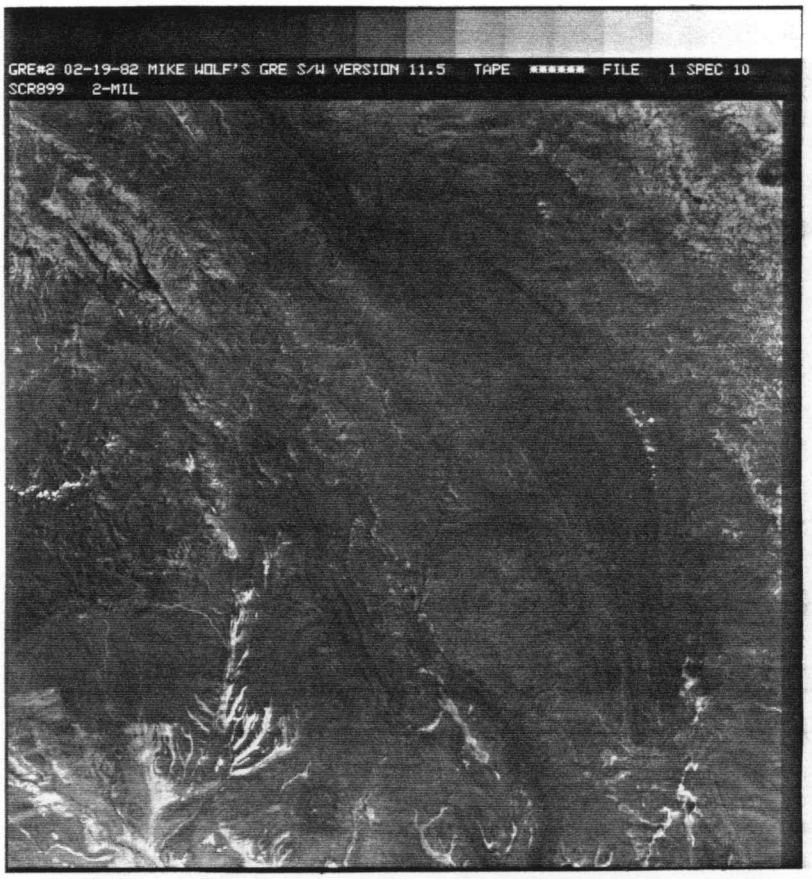


Figure 17. Digital Landsat mosaic quadrangle No. VI.

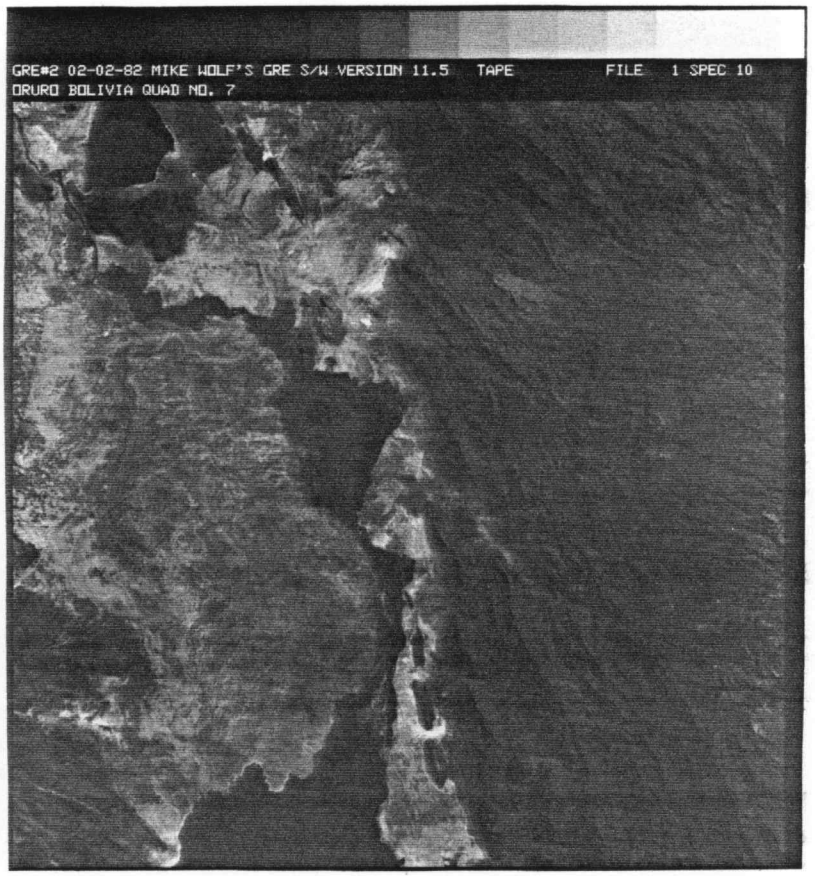


Figure 18. Digital Landsat mosaic quadrangle No. VII.

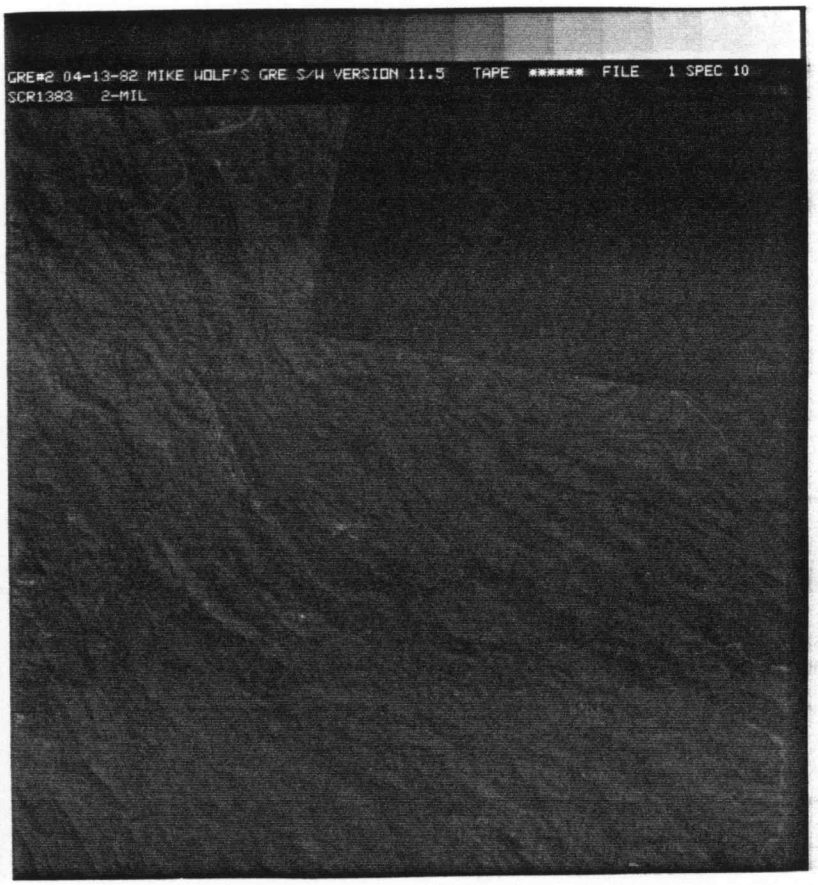


Figure 19. Digital Landsat mosaic quadrangle No. VIII.

Figure 20. Digital Landsat mosaic quadrangle No. IX.

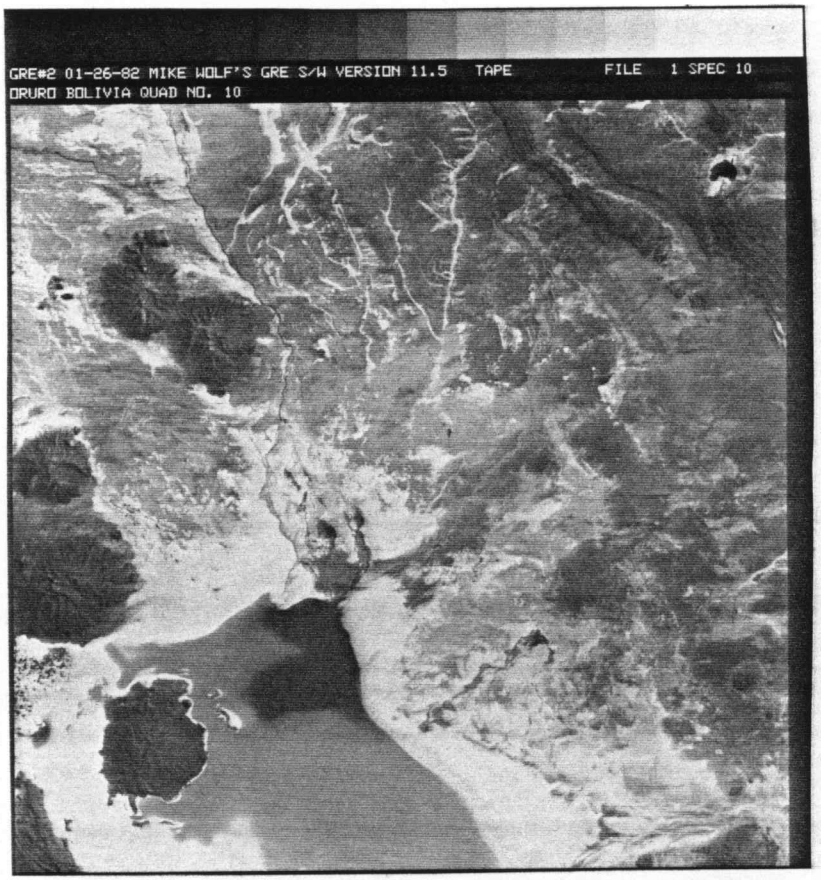


Figure 21. Digital Landsat mosaic quadrangle No. X.

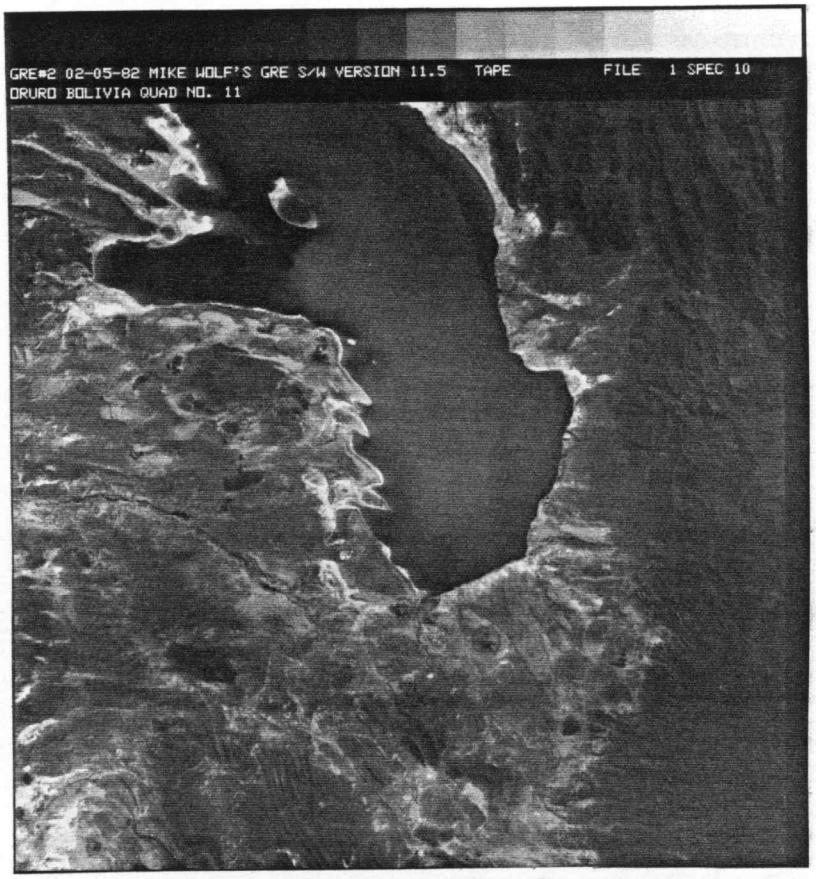


Figure 22. Digital Landsat mosaic quadrangle No. XI.

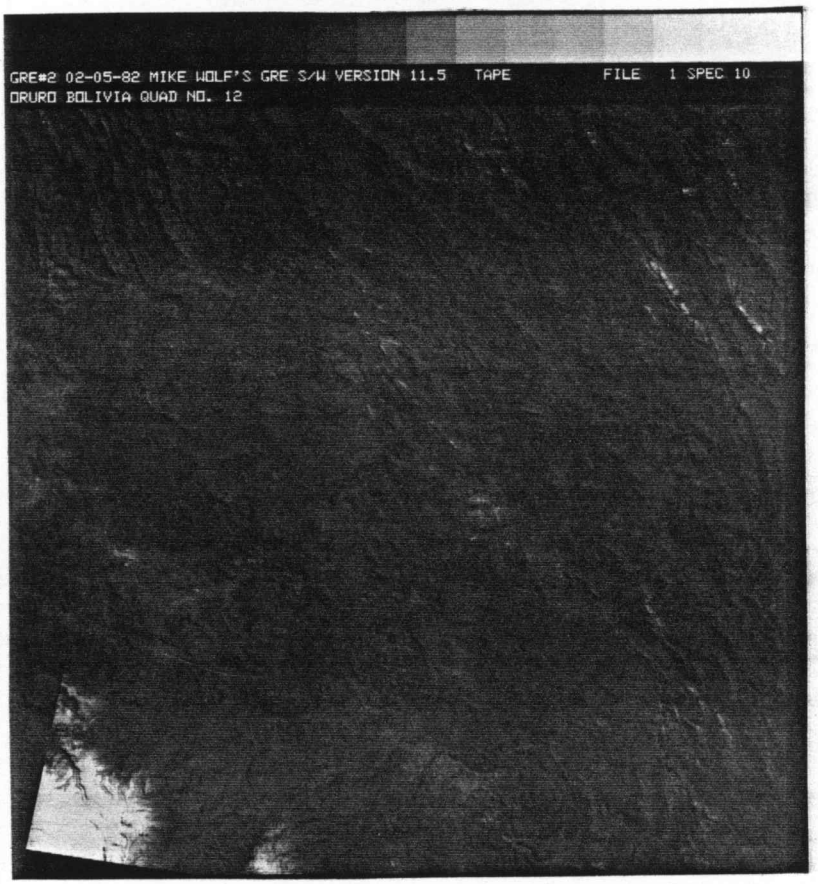


Figure 23. Digital Landsat mosaic quadrangle No. XII.

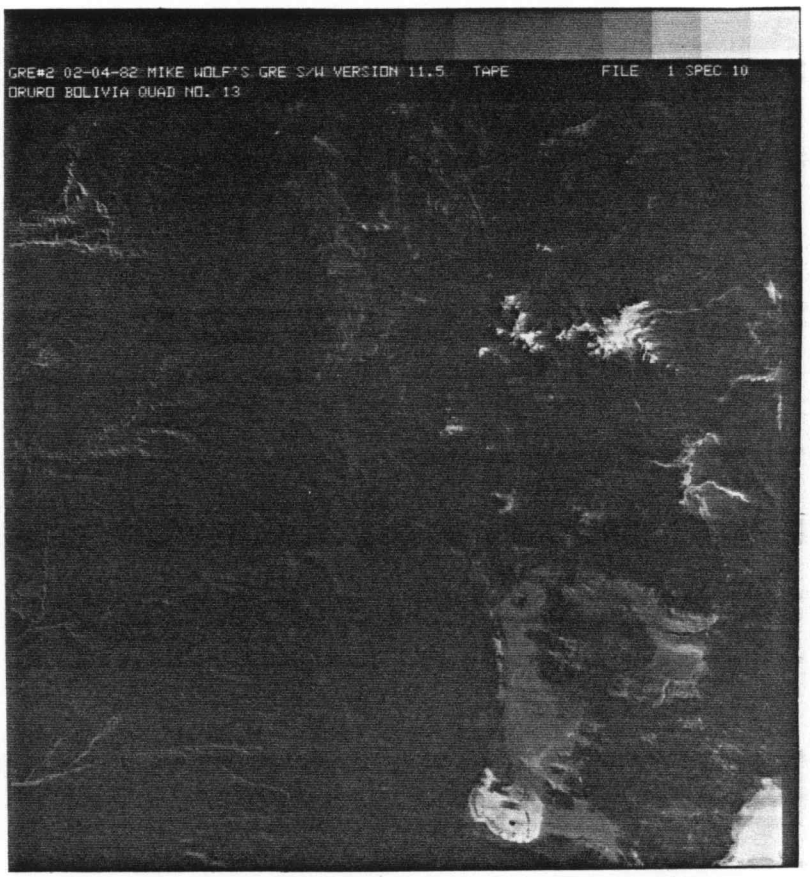


Figure 24. Digital Landsat mosaic quadrangle No. XIII.

Figure 25. Digital Landsat mosaic quadrangle No. XIV.

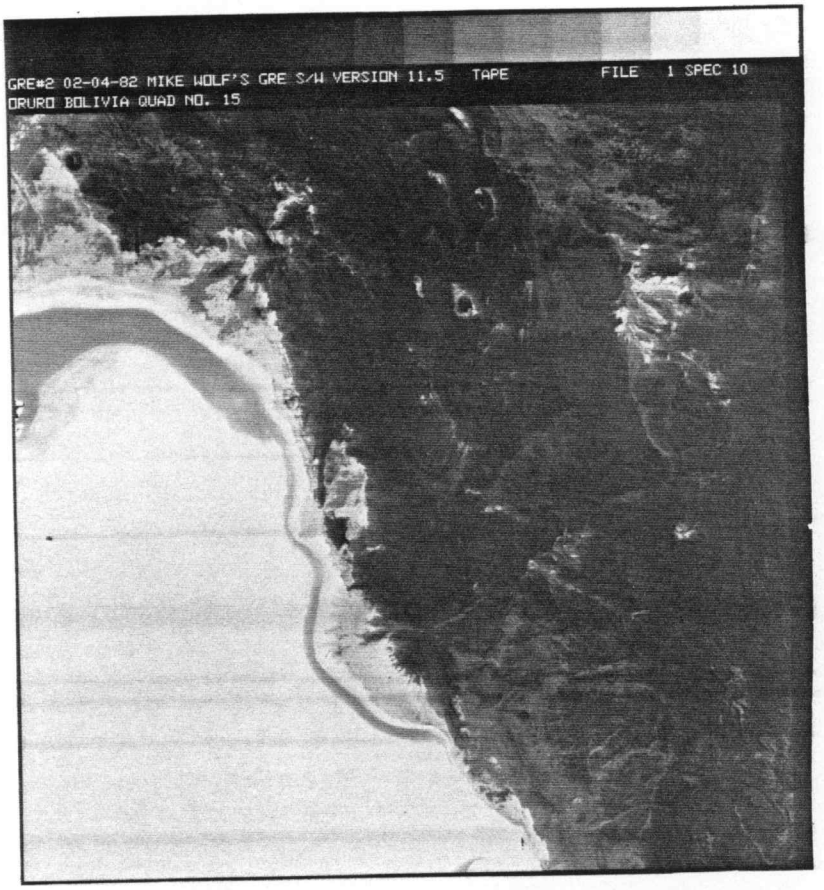


Figure 26. Digital Landsat mosaic quadrangle No. XV.

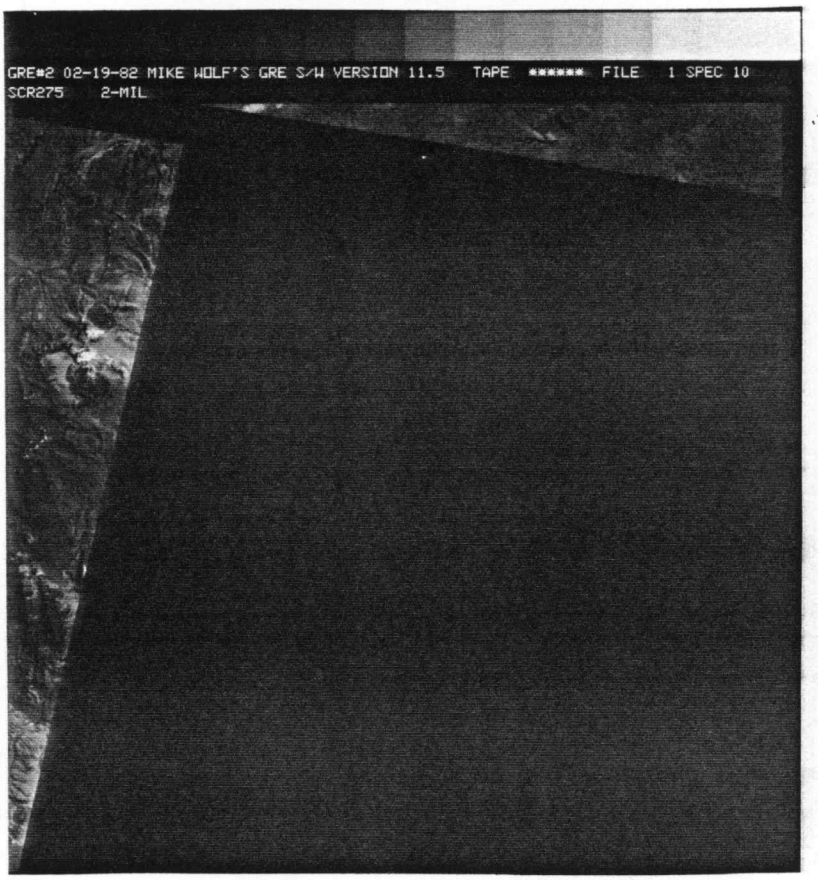


Figure 27. Digital Landsat mosaic quadrangle No. XVI.

Arcs and Centroid Digital Files. Also a computer tape (Tape No. 2092) containing the digitized arcs and centroids for the different elements of the Bolivian geocoded image data base is being delivered. This tape contains 6 files of longitude and latitude digitized data. Tables 3 - 8 describe the contents of each one of these 6 files.

Computer Generated Maps. A set of four computer generated (gray-scale) maps for each of the elements included in the geocoded image data base are being delivered to ERTS/GEOBOL. These maps have a horizontal scale of 1:472,441 and a vertical scale of 1:566,929, and the difference in scales is solely due to the physical limitations of the printer/plotter output device, which has an 8/10 characters aspect ratio (60 horizontal and 72 vertical dots per characters). A photographically reduced example of one of these maps is shown in Figure 28.

Photographic Color Maps. Two sets of original color 35 mm slides for each of the elements included in the geocoded image data base are also being delivered. Color prints obtained from these slides are shown in Figures 29 through 47.

Videotapes. In order to demonstrate all the procedures involved in converting natural resources, environmental, and socio-economic information from conventional maps into digital geocoded images, a videotape was produced and will be delivered to ERTS/GEOBOL. This color videotape has a duration of 38 minutes and 24 seconds, and it has been duplicated in three different types of tape formats: a) 3/4 inch professional, b) VHS, and c) Betamax videocassette formats.

Table 3. Tape and file number and a list of files and corresponding topographic base maps for the Geology Element for the Oruro Department GIS.

Tape (2092), File (1)

File Name	Topographic Base Map
Geon1C loncen[1]	
Geon1C lonlat[2]	SE 19-7 Corocoro
Geon2C loncen	
Geon2C lonlat	SE 19-8 Cochabamba
Geon3C loncen	
Geon3C lonlat	SE 19-10 Payachatas
Geon4C loncen	
Geon4C lonlat	SE 19-11 Corque
Geon5C loncen	
Geon5C lonlat	SE 19-12 Uncia
Geon6C loncen	
Geon6C lonlat	SE 19-15 S. de Garci Mendoza
Geon7C loncen	
Geon7C lonlat	SE 19-16 Rio Mulatos

[1] Loncen. File containing the geographic coordinate values (longitude and latitude), of the digitized centroids (point data) used for the rasterization (painting) process.

[2] Lonlat. File containing the geographic coordinate values (longitude and latitude), of the digitized lines enclosing thematic units (arcs data).

Table 4. Tape and file number, and a list of files and corresponding topographic base maps for the Watersheds Element for the Oruro Department GIS.

Tape (2092), File (2)

File Name	Topographic Base Map
CNS1C loncen[1]	
CNS1C lonlat[2]	SE 19-7 Corocoro
CNS2C loncen	
CNS2C lonlat	SE 19-8 Cochabamba
CNS3C loncen	
CNS3C lonlat	SE 19-10 Payachatas
CNS4C loncen	
CNS4C lonlat	SE 19-11 Corque
CNS5C loncen	
CNS5C lonlat	SE 19-12 Uncia
CNS6C loncen	
CNS6C lonlat	SE 19-15 S. de Garci Mendoza
CNS7C loncen	
CNS7C lonlat	SE 19-16 Rio Mulatos

[1] Loncen. File containing the geographic coordinate values (longitude and latitude), of the digitized centroids (point data) used for the rasterization (painting) process.

[2] Lonlat. File containing the geographic coordinate values (longitude and latitude), of the digitized lines enclosing thematic units (arcs data).

Table 5. Tape and file number, and a list of files* and corresponding topographic base maps for the Soils Element for the Oruro GIS.

Tape (2092), File (3)

File Name	Topographic Base Map	
Sue1C lonlat[2]	SE 19-7	Corocoro
Sue2C lonlat	SE 19-8	Cochabamba
Sue3C lonlat	SE 19-10	Payachatas
Sue4C loncen[1]		
Sue4C lonlat	SE 19-11	Corque
Sue5C lonlat	SE 19-12	Uncia
Sue6C lonlat	SE 19-15	S. de Garci Mendoza
Sue7C lonlat	SE 19-16	Rio Mulatos

---

[1] Loncen. File containing the geographic coordinate values (longitude and latitude), of the digitized centroids (point data) used for the rasterization (painting) process.

[2] Lonlat. File containing the geographic coordinate values (longitude and latitude), of the digitized lines enclosing thematic units (arcs data).

* The rasterization process used in "painting" the thematic units did not use centroids in elements without the "loncen" file, while it did use the centroids in those that have the "loncen" file.

Table 6. Tape and file number, and a list of files* and corresponding topographic base maps for the Land Cover/Land Use Element for the Oruro Department GIS.

Tape (2092), File (4)

File Name	Topographic Base Map	
USO1C lonlat[2]	SE 19-7	Corocoro
USO2C lonlat	SE 19-8	Cochabamba
USO3C lonlat	SE 19-10	Payachatas
USO4C loncen[1]		
USO4C lonlat	SE 19-11	Corque
USO5C lonlat	SE 19-12	Uncia
USO6C lonlat	SE 19-15	S. de Garci Mendozo
USO7C lonlat	SE 19-16	Rio Mulatos

---

[1] Loncen. File containing the geographic coordinate values (longitude and latitude), of the digitized centroids (point data) used for the rasterization (painting) process.

[2] Lonlat. File containing the geographic coordinate values (longitude and latitude), of the digitized lines enclosing thematic units (arcs data).

* The rasterization process used in "painting" the thematic units did not use centroids in elements without the "loncen" file, while it did use the centroids in those that have the "loncen" file.

Table 7. Tape and file number, and a list of files* and corresponding topographic base maps for the Political Boundaries Element for the Oruro Department GIS.

Tape (2092), File (5)

File Name	Base Map
Polito lonlat	Instituto Nacional de Estadisticas (INE)

Table 8. Tape and file number, and a list of files* and corresponding topographic base maps for the Geomorphology Element for the Oruro Department GIS.

Tape (2092), File (6)

File Name	Topographic Base Map
MORF1 loncen[1]	
MORF1 lonlat[2]	SE 19-7 Corocoro
MORF2 loncen	
MORF2 lonlat	SE 19-8 Cochabamba
MORF3 loncen	
MORF3 lonlat	SE 19-10 Payachatas
MORF4 loncen	
MORF4 lonlat	SE 19-11 Corque
MORF5 loncen	
MORF5 lonlat	SE 19-12 Uncia
MORF6 loncen	
MORF6 lonlat	SE 19-15 S. de Garci Mendoza
MORF7 loncen	
MORF7 lonlat	SE 19-16 Rio Mulatos

---

[1] Loncen. File containing the geographic coordinate values (longitude and latitude), of the digitized centroids (point data) used for the rasterization (painting) process.

[2] Lonlat. File containing the geographic coordinate values (longitude and latitude), of the digitized lines enclosing thematic units (arcs data).

* The rasterization process used in "painting" the thematic units did not use centroids in elements without the "loncen" file, while it did use the centroids in those that have the "loncen" file.

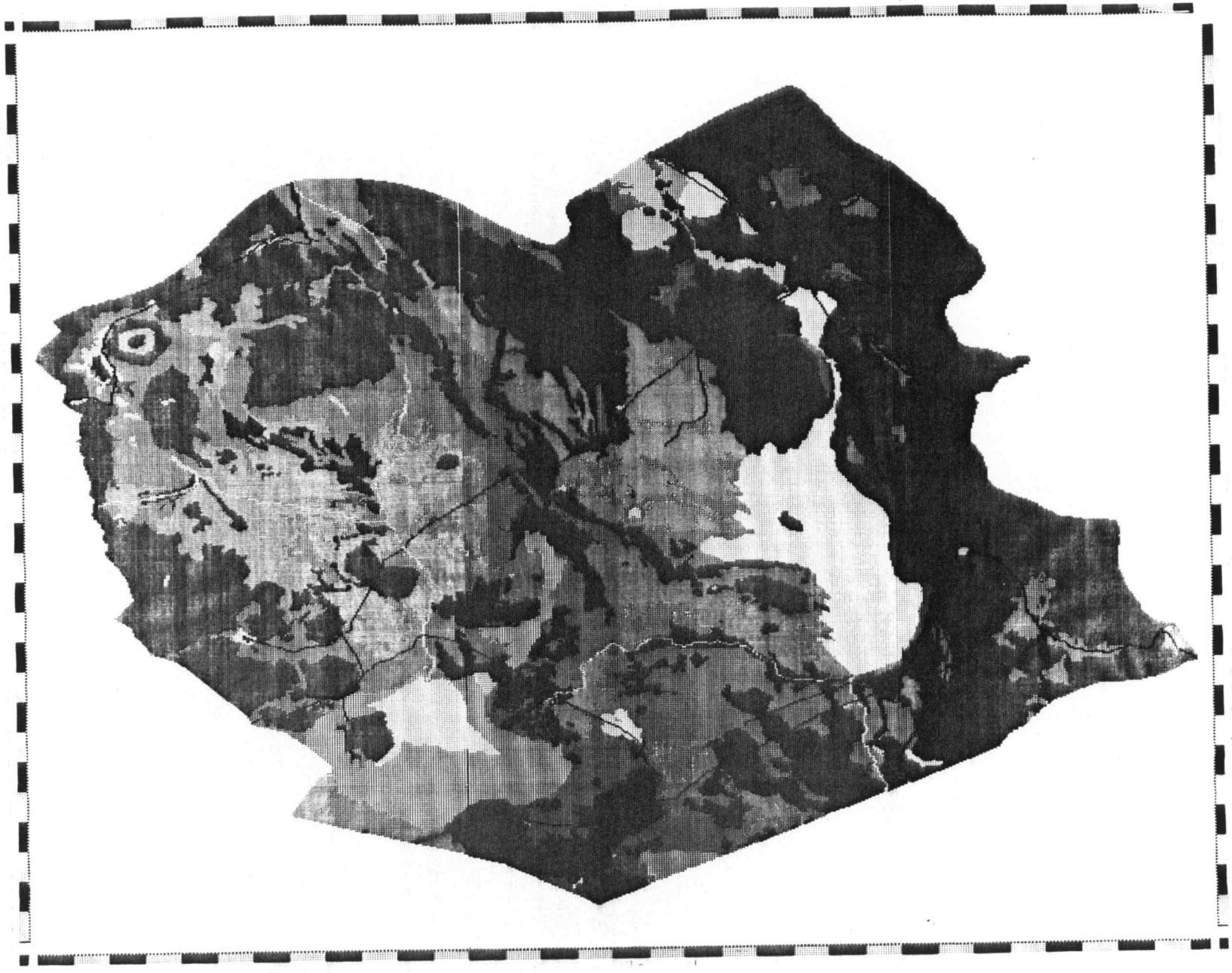


Figure 28. Photographically reduced example of a computer generated map.

Figure 29. Landsat MSS digital mosaic included in the geocoded image plane data base of the Oruro Department.

Figure 30.

Figure 31.

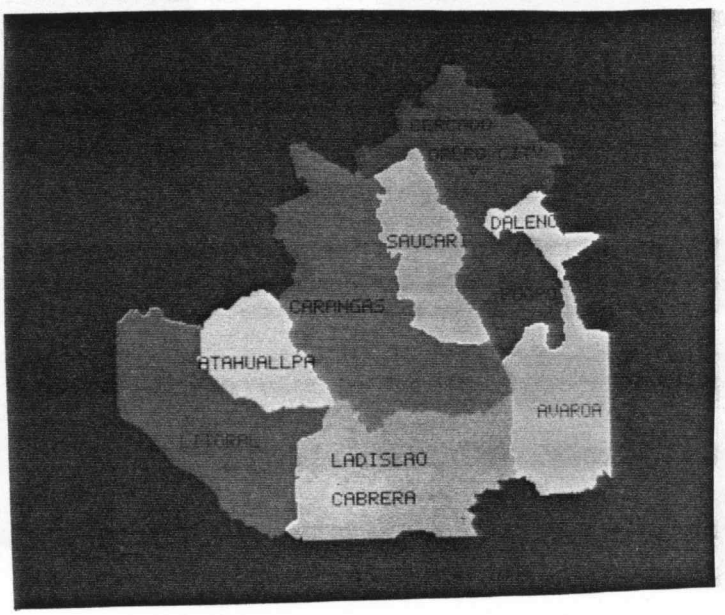


Figure 32.

Figures 30, 31 and 32. Color-coded digital maps corresponding to the Political Boundaries element of the Oruro Department.

Figure 33.

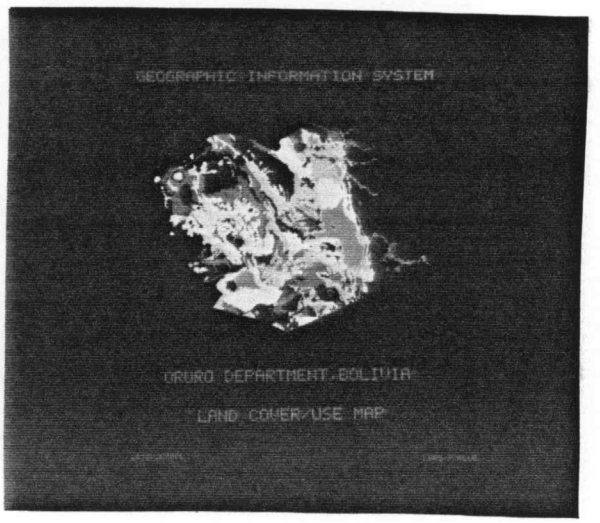


Figure 34.

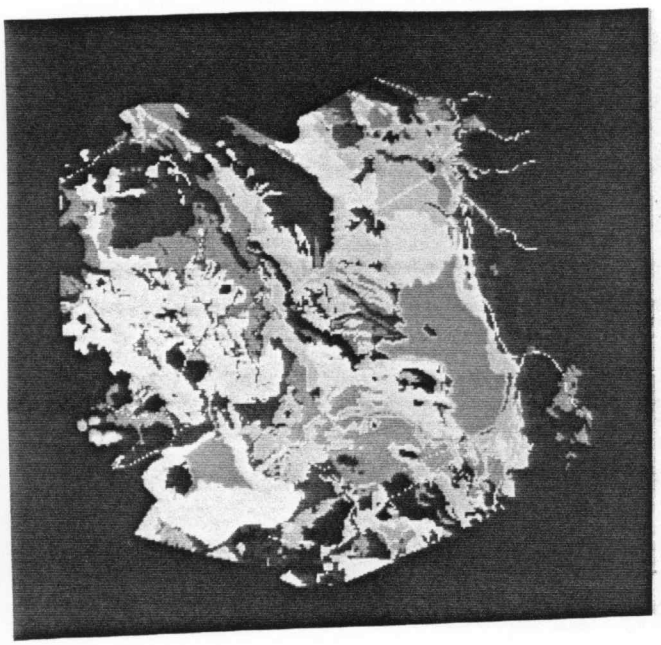


Figure 35.

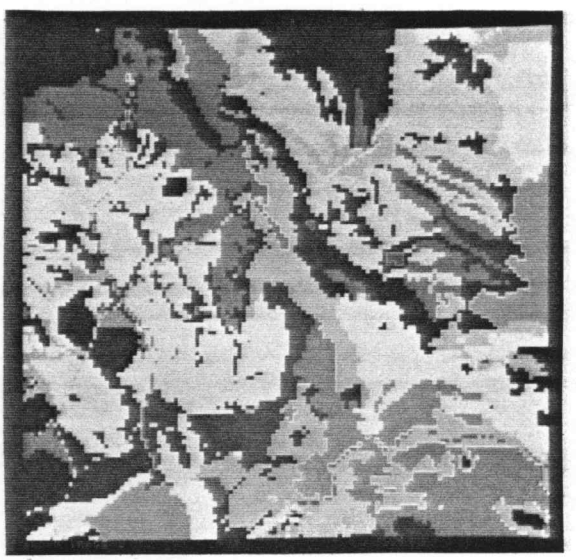


Figure 36.

Figures 33, 34, 35 and 36. Color-coded digital maps corresponding to the Land Cover/Land Use element of the Oruro Department.

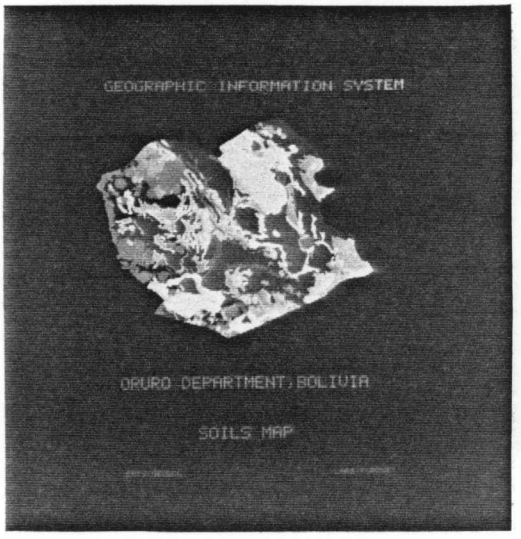


Figure 37.

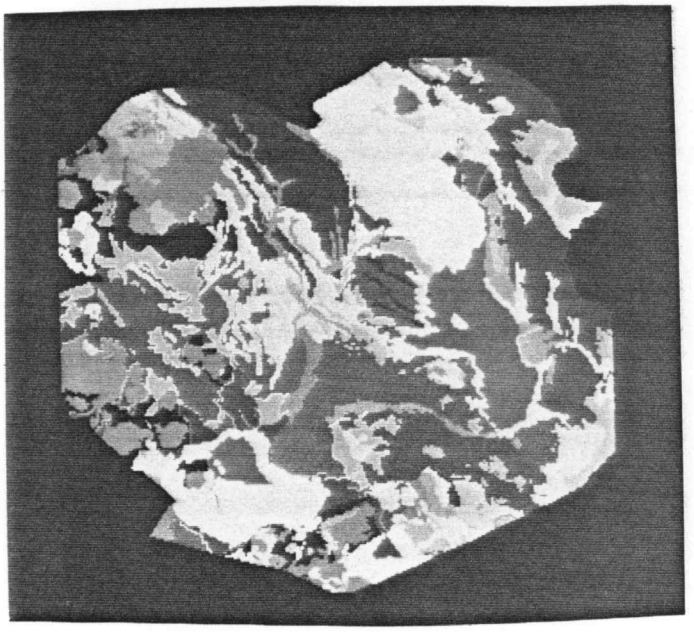


Figure 38.

Figures 37 and 38. Color-coded digital maps corresponding to the Soils element of the Oruro Department.

Figure 39.

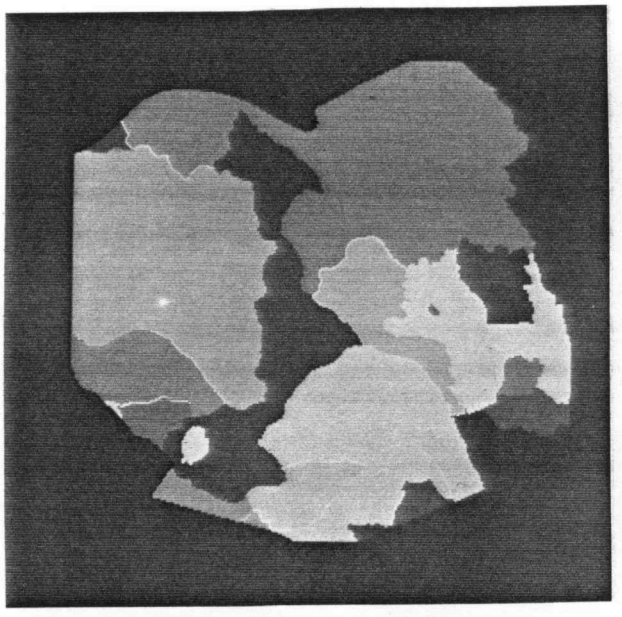


Figure 40.

Figures 39 and 40. Color-coded digital maps corresponding to the Watershed element of the Oruro Department.

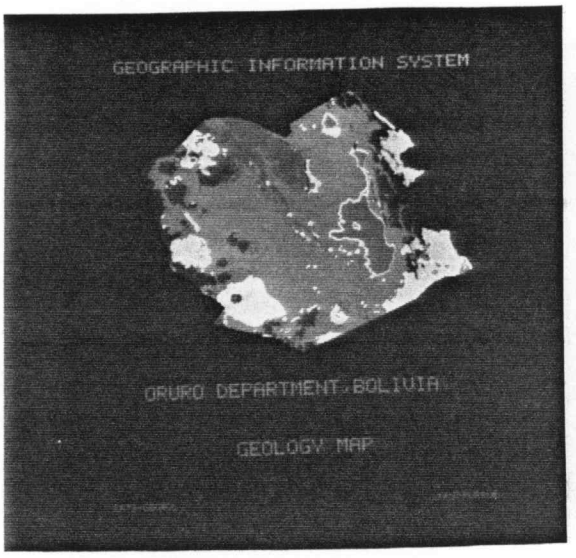


Figure 41.

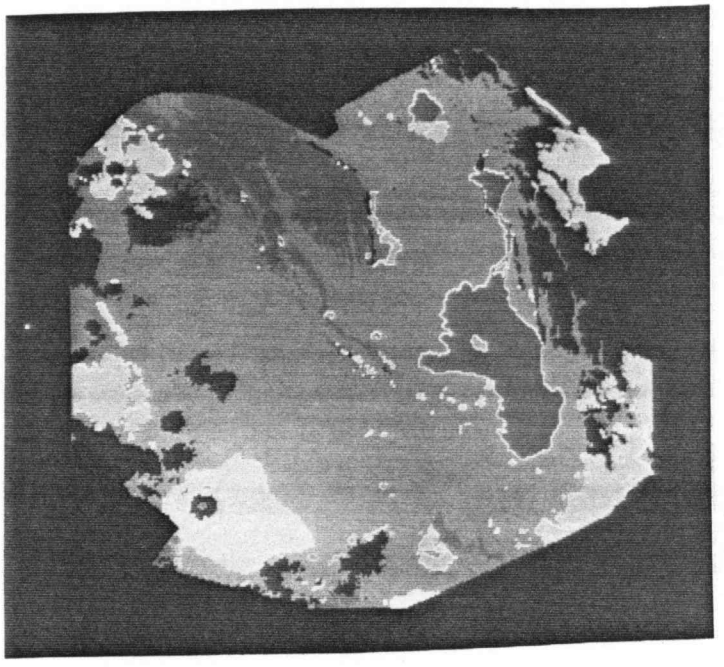


Figure 42.

Figures 41 and 42. Color-coded digital maps corresponding to the Geology element of the Oruro Department.

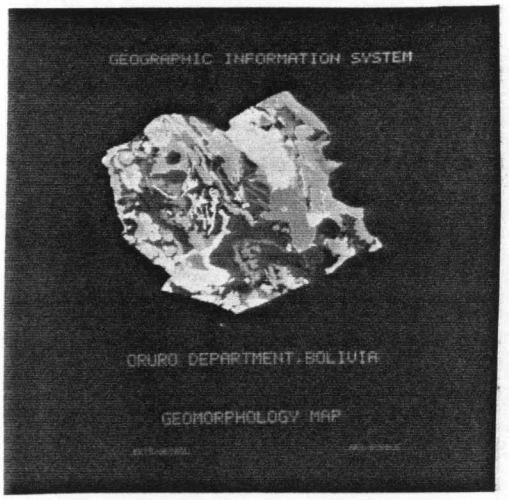


Figure 43.

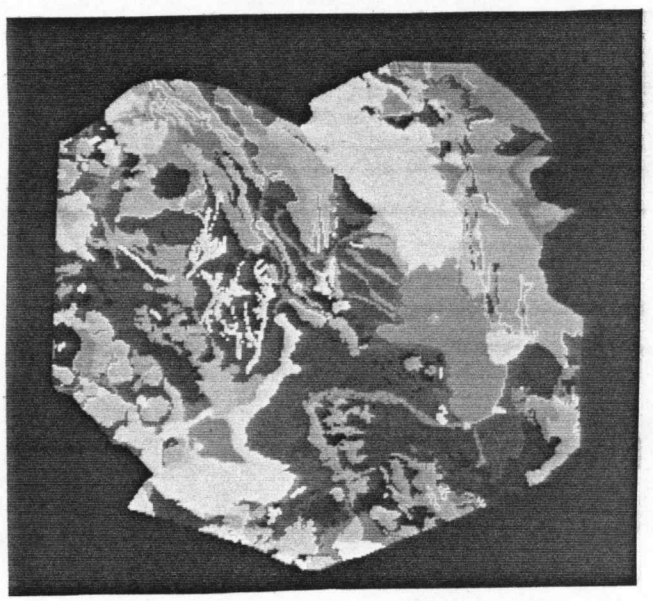


Figure 44.

Figures 43 and 44. Color-coded digital maps corresponding to the Geomorphology element of the Oruro Department.

Figure 45. Color-coded Digital Terrain Model (DTM), Elevation element for a small portion of the Uncia topographic map of the Oruro Department.

Figure 46. Color-coded Digital Terrain Model (DTM), Slope element for a small portion of the Uncia topographic map of the Oruro Department.

Figure 47. Color-coded Digital Terrain Model (DTM), Aspect (Azimuth) element for a small portion of the Uncia topographic map of the Oruro Department.

Image Plane Data Base Digital Files. All the raterized GIS elements of the Oruro image plane data base have been stored in a computer tape (Tape No. 5653) in 12 different files as indicated in Table 9.

Table 9. Rasterized Image Plane Data Base Elements.

Tape No. 5653

File No.	Element Code	Element Name
1	L0500Q10	Politica
2	L0500Q10	Lancuse
3	L0500Q10	Soils
4	L0500Q10	Hydrolog
5	L0500Q10	Geology
6	L0500Q10	Geomorph
7	L0500Q10	Landmss4
8	L0500Q10	Landmss5
9	L0500Q10	Landmss6
10	L0500Q10	Landmss7
11	L0500Q10	Boundry
12	L0500Q10	Elevation

The element code contains the information regarding the "Level" of the data base, which in this particular case is L0500 corresponding to the Departmental Level of detail composed of 500m x 500m cells (pixels), and the member of the quadrangle that for the Oruro Department has a designation Q10. Each one of these elements is a square composed of 900 x 900 pixels as indicated by the number of records listed in Table 10.

Table 10. Contents of Tape No. 5653

FILENAME	FILETYPE	FM	FORMAT		RECS	BLKS	DATE	TIME
L0500Q10	BOUNDRY	D1	F	900	900	793	4/15/82	14.29.00
L0500Q10	ELEVATIO	D1	F	900	900	793	6/30/82	11.24.00
L0500Q10	GEOLOGY	D1	F	900	900	793	5/20/82	15.40.00
L0500Q10	GEOMORPH	D1	F	900	900	793	6/30/82	10.26.00
L0500Q10	HYDROLOG	D1	F	900	900	793	4/22/82	16.57.00
L0500Q10	LANDCUSE	D1	F	900	900	793	4/12/82	20.30.00
L0500Q10	LANDMSS4	D1	F	900	900	793	4/08/82	15.40.00
L0500Q10	LANDMSS5	D1	F	900	900	793	4/07/82	23.26.00
L0500Q10	LANDMSS6	D1	F	900	900	793	4/15/82	14.26.00
L0500Q10	LANDMSS7	D1	F	900	900	793	4/08/82	14.34.00
L0500Q10	POLITICA	D1	F	900	900	793	3/01/82	14.57.00
L0500Q10	SOILS	D1	F	900	900	793	6/30/82	10.48.00

Attribute Data Base Files. All the hierarchical attribute information contained in the Oruro Department Attribute Data Base has been stored in eleven consecutive files of Tape No. 5654 that will be delivered to ERTS/GEOBOL. Table 11 contains a brief description of these files.

Table 11. Attribute Data Base Files.

Tape No. 5654

File 2	Bolivia Database File
File 3	Level File
File 4	Quadrant File
File 5	Element File
File 6	Politica File 1 Politica File 2 Politica File 3 Politica File 4
File 7	Landcuse File 1 Landcuse File 2 Landcuse File 3 Landcuse File 4
File 8	Soils File 1 Soils File 2 Soils File 3
File 9	Hydrolog File 1 Hydrolog File 2 Hydrolog File 3
File 10	Geology File 1 Geology File 2 Geology File 3
File 11	Geomorph File 1 Geomorph File 2 Geomorph File 3
File 12	Backup File Banner File Editrai File Database Logfile Albers File

Imput Subsystem Software. The computer programs that support the process of map digitization and that are being delivered to the ERTS/GEOBOL counterpart have been stored in File No. 2 of Tape No. 5654. A brief description of what is accomplished by each one of these programs is given in Table 12.

Table 12. DIGITIZING PROGRAMS

ALBERS - Computes Albers addresses from latitude/longitude.

ALBET - Supervises Albers address calculations.

ARCLOS - Arc closure processor.

EDITMAP - Take table digitized data and display in graphics form and edit arc lines.

MULREG - Performs multiple regression analysis for digitized checkpoints to calculate coefficients to transform digitized data into latitude and longitide.

MYFDIG - PDP program to supervise table digitizing.

PROJECTI - Project latitude/longitude coordinates into Albers addresses.

RASCEN - Places digitized arcs into an image file on tape with areas filled with characters or only boundaries drawn.

SETCON - Set-up Albers projection constants.

SETPAR - Set-up Albers addressing parameters.

TERARC - Automatic arc closure of terrain data.

TERPRO - Transform contour/boundary data into latitude/longitude as well as Albers addresses.

TERSUP - Rasterized terrain data.

TRANCO - Translates x-y addresses into latitude-longitude addresses.

Data Base Software. All the computer programs that support the Bolivian GIS data base have been stored in File No. 1 of Tape No. 5654, and a brief description of each one of these programs is given in Table 13. A listing of the total contents of Tape No. 5654 is included in Table 14.

Table 13. BOLIVIAN DATA BASE PROGRAMS

AUX - Auxilliary programs to support the mapping functions for the data base display.

BACK - Subroutine to move backwards through the data base.

BADCMD - Write out error message that a bad command was encountered.

BANMSS - Write out banner message for data base management system.

BANNER - Read and create banner message.

BKUP - Save data base information on magnetic tape.

BLDDAT - Read in data base attribute information.

CHANGE - Change information held in current data base record.

CHGBND - Change boundary information.

CHGCHR - Change fill character values.

CHGDAT - Change date in standard record.

CHGFIL - Revise file number.

CHGHD1 - Change standard header 1 in attribute file.

CHGHD2 - Change standard header 2 in attribute file.

CHGHIS - Change history documents.

CHGNAM - Revise standard names.

CHGPIX - Revise area/pixel value.

CHGPRJ - Change projection name.

CHGREF - Revise reference code.

CHGSP1 - Revise standard parallel value 1.

CHGSP2 - Revise standard parallel value 2.

CHGSTM - Revise standard meridian value.

CHGSUB - Revise sub-dominate code.

CHGSYN - Write out syntax of change commands to aid user.

CHGTAP - Revise data tape number

CHGUPD - Change up pointer.

COMMON - Declaration of all common variables.

COMPIM - Create disk for merge function.

COMTAL - Send data base image file to PDP for COMTAL display.

CREATE - Create shade fill characters.

DATDMP - Dump data listing to line printer for current element.

DEFDE - Define data element.

DEL - Delete record information.

DISKTR - Transfer disk data to LARSYS tape format.

DOWN - Move down through data base.

DSKNAM - Set up disk names.

DSKTYP - Set up disk types.

EDIMAG - Edit image file information.

EDIPNT - Edit pointer information.

EDITTR - Edit edittrail file.

ELEBLD - Prompt user for information to construct an element record.

ELENM - Print element code names.

FLLFND - Find fill character values.

GADREC - Get one record of data from image.

GARLST - Get pointer list information.

GARPL - Get pointer location.

HELPM - Write out help messages to aid user.

IMLOG - Read image information.

INS - Input record information.

IOOP - Read and write element information.

LOCARD - Locate requested records.

MAPIT - Draw a printronix map from LARSYS tape or image disk file.

MGMT - Database management main program.

MOVV - Merge new LARSYS tape images out to disk.

NXT - Go to next record.

NXTF - Finds the next available record to insert into, checks garbage file, then to bottom of the file.

PARCHG - Parse out proper change commands.

PARCMD - Determine command request.

PARENT - Determine parent pointer and record.

PASSW - Set privilege password.

PLACEW - Convert alpha characters into integer values.

QDES - Query user for description record information.

QELEM - Query user for element record information.

QINFO - Query user for database information.

QLEVEL - Query user for level record information.

QQUAD - Query user for quadrant record information.

READDB - Read a database record.

RESTOR - Restore database from back up save tape.

SHIFTR - Shift contents data array to right until there is a non-blank character in the right-most byte.

SKIPB - Skip blanks on input line.

SKIPOP - Skip over special characters in input line.

SKIPZB - Skip to a blank on input line.

SKIPZN - Skip to a number from 0 to 9 on input line.

SKIPZP - Skip to a special character on input line.

TROUBL - Record user trouble information.

VICMER - Vicer merge processor.

WRTDES - Write description record information.

ZERO - Zero-out data file areas on disk.

Table 14. Contents of Tape No. 5654

FILENAME	FILETYPE	FM	FORMAT	RECS	BLKS	DATE	TIME
EDITMAP	EXEC	D2	F 80	17	2	10/19/82	15.37.00
MAPPRO	EXEC	D2	F 80	252	20	10/19/82	15.37.00
MGMT	EXEC	D1	V 74	40	2	10/19/82	15.37.00
RASCEN	EXEC	D2	V 65	141	5	10/19/82	15.37.00
TERRAIN	EXEC	D5	V 74	223	8	10/19/82	15.37.00
ALBERS	FILE	D5	F 80	4	2	7/02/82	9.48.00
BACKUP	FILE	D1	F 80	6	2	6/30/82	13.44.00
BANNER	FILE	D1	F 80	16	3	4/08/82	17.17.00
DATABASE	FILE	D1	F 328	2	2	3/05/82	14.09.00
EDITRAIL	FILE	D1	F 80	1524	121	10/21/82	10.26.00
ELEMENT	FILE	D1	F 1400	13	19	6/30/82	11.36.00
LEVEL	FILE	D1	F 96	2	2	2/06/82	13.52.00
QUADRANT	FILE	D1	F 194	2	2	4/05/82	18.55.00
GEOLOGY	FILE1	D1	F 220	4	2	2/04/82	10.37.00
GEOMORPH	FILE1	D1	F 220	9	3	6/29/82	15.35.00
HYDROLOG	FILE1	D1	F 220	4	2	9/24/81	15.33.00
LANDCUSE	FILE1	D1	F 220	22	6	1/14/82	11.44.00
POLITICA	FILE1	D1	F 220	7	3	1/27/82	13.22.00
SOILS	FILE1	D1	F 220	5	3	1/21/82	21.40.00
GEOLOGY	FILE2	D1	F 220	7	3	2/04/82	10.37.00
GEOMORPH	FILE2	D1	F 220	21	6	6/29/82	20.51.00
HYDROLOG	FILE2	D1	F 220	6	3	9/24/81	15.33.00
LANDCUSE	FILE2	D1	F 220	27	7	1/14/82	11.44.00
POLITICA	FILE2	D1	F 220	10	4	1/27/82	13.22.00
SOILS	FILE2	D1	F 220	17	5	1/21/82	21.40.00
GEOLOGY	FILE3	D1	F 220	15	5	2/04/82	10.37.00
GEOMORPH	FILE3	D1	F 220	106	24	6/29/82	21.08.00
HYDROLOG	FILE3	D1	F 220	32	8	4/22/82	14.32.00
LANDCUSE	FILE3	D1	F 220	81	19	1/14/82	11.45.00
POLITICA	FILE3	D1	F 220	100	23	1/27/82	13.22.00
SOILS	FILE3	D1	F 220	50	12	1/21/82	21.40.00
LANDCUSE	FILE4	D1	F 220	3	2	1/12/82	11.48.00
POLITICA	FILE4	D1	F 220	113	25	1/27/82	13.23.00
ALBERS	FORTRAN	D1	F 80	53	5	10/20/82	14.43.00
ALBET	FORTRAN	D1	F 80	107	9	10/20/82	14.43.00
ARCLOS	FORTRAN	D1	F 80	270	22	10/20/82	14.43.00
AUX	FORTRAN	D1	F 80	837	66	10/19/82	13.53.00
BACK	FORTRAN	D1	F 80	84	7	10/19/82	13.53.00
BADCMD	FORTRAN	D1	F 80	112	9	10/19/82	13.53.00
BANMSS	FORTRAN	D1	F 80	109	9	10/19/82	13.53.00
BANNER	FORTRAN	D1	F 80	92	8	10/19/82	13.53.00
BKUP	FORTRAN	D1	F 80	225	18	10/19/82	13.53.00
BLDDAT	FORTRAN	D1	F 80	140	11	10/19/82	13.53.00
CHANGE	FORTRAN	D1	F 80	245	20	10/19/82	13.53.00
CHGBND	FORTRAN	D1	F 80	140	11	10/19/82	13.54.00
CHGCHR	FORTRAN	D1	F 80	94	8	10/19/82	13.54.00
CHGDAT	FORTRAN	D1	F 80	78	7	10/19/82	13.54.00
CHGFIL	FORTRAN	D1	F 80	80	7	10/19/82	13.54.00
CHGHD1	FORTRAN	D1	F 80	100	8	10/19/82	13.54.00
CHGHD2	FORTRAN	D1	F 80	100	8	10/19/82	13.54.00
CHGHIS	FORTRAN	D1	F 80	104	9	10/19/82	13.54.00
CHGNAM	FORTRAN	D1	F 80	88	7	10/19/82	13.54.00
CHGPIX	FORTRAN	D1	F 80	77	7	10/19/82	13.54.00
CHGPRJ	FORTRAN	D1	F 80	89	7	10/19/82	13.54.00
CHGREF	FORTRAN	D1	F 80	76	6	10/19/82	13.54.00
CHGSP1	FORTRAN	D1	F 80	88	7	10/19/82	13.54.00
CHGSP2	FORTRAN	D1	F 80	88	7	10/19/82	13.54.00
CHGSTM	FORTRAN	D1	F 80	88	7	10/19/82	13.54.00
CHGSUB	FORTRAN	D1	F 80	84	7	10/19/82	13.54.00
CHGSYN	FORTRAN	D1	F 80	291	23	10/19/82	13.54.00
CHGTAP	FORTRAN	D1	F 80	80	7	10/19/82	13.54.00
CHGUPD	FORTRAN	D1	F 80	71	6	10/19/82	13.54.00
COMMON	FORTRAN	D1	F 80	62	5	10/19/82	13.54.00
COMPIM	FORTRAN	D1	F 80	87	7	10/19/82	13.54.00
COMTAL	FORTRAN	D1	F 80	374	30	10/19/82	13.54.00
CREATE	FORTRAN	D1	F 80	73	6	10/19/82	13.54.00
DATDMP	FORTRAN	D1	F 80	471	37	10/19/82	13.54.00
DEFDE	FORTRAN	D1	F 80	109	9	10/19/82	13.54.00
DEL	FORTRAN	D1	F 80	93	8	10/19/82	13.54.00
DISKTR	FORTRAN	D1	F 80	335	27	10/19/82	13.54.00
DOWN	FORTRAN	D1	F 80	77	7	10/19/82	13.54.00
DSKNAM	FORTRAN	D1	F 80	83	7	10/19/82	13.54.00
DSKTYP	FORTRAN	D1	F 80	75	6	10/19/82	13.54.00
EDIMAG	FORTRAN	D1	F 80	336	27	10/19/82	13.54.00
EDIPNT	FORTRAN	D1	F 80	99	8	10/19/82	13.54.00
EDITTR	FORTRAN	D1	F 80	105	9	10/19/82	13.54.00
ELEBLD	FORTRAN	D1	F 80	144	12	10/19/82	13.54.00
ELENM	FORTRAN	D1	F 80	119	10	10/19/82	13.54.00
FLLFND	FORTRAN	D1	F 80	72	6	10/19/82	13.54.00

FILENAME	FILETYPE	FM	FORMAT		RECS	BLKS	DATE	TIME
GADREC	FORTRAN	D1	F	80	14	2	10/19/82	13.54.00
GARLST	FORTRAN	D1	F	80	92	8	10/19/82	13.54.00
GARPL	FORTRAN	D1	F	80	78	7	10/19/82	13.54.00
HELPM	FORTRAN	D1	F	80	499	39	10/19/82	13.54.00
IMLOG	FORTRAN	D1	F	80	40	4	10/19/82	13.54.00
INS	FORTRAN	D1	F	80	187	15	10/19/82	13.54.00
IOOP	FORTRAN	D1	F	80	125	10	10/19/82	13.54.00
LOCARD	FORTRAN	D1	F	80	106	9	10/19/82	13.54.00
MAPIT	FORTRAN	D1	F	80	993	78	10/19/82	13.55.00
MGMT	FORTRAN	D1	F	80	549	43	10/19/82	13.55.00
MOVV	FORTRAN	D1	F	80	499	39	10/19/82	13.55.00
NXT	FORTRAN	D1	F	80	75	6	10/19/82	13.55.00
NXTF	FORTRAN	D1	F	80	123	10	10/19/82	13.55.00
PARCHG	FORTRAN	D1	F	80	303	24	10/19/82	13.55.00
PARCMD	FORTRAN	D1	F	80	95	8	10/19/82	13.55.00
PARENT	FORTRAN	D1	F	80	79	7	10/19/82	13.55.00
PASSW	FORTRAN	D1	F	80	70	6	10/19/82	13.55.00
PLACEW	FORTRAN	D1	F	80	133	11	10/19/82	13.55.00
PROJECTI	FORTRAN	D1	F	80	131	11	10/20/82	14.43.00
QDES	FORTRAN	D1	F	80	113	9	10/19/82	13.55.00
QELEM	FORTRAN	D1	F	80	114	9	10/19/82	13.55.00
QINFO	FORTRAN	D1	F	80	71	6	10/19/82	13.55.00
QLEVEL	FORTRAN	D1	F	80	111	9	10/19/82	13.55.00
QQUAD	FORTRAN	D1	F	80	132	11	10/19/82	13.55.00
READDB	FORTRAN	D1	F	80	77	7	10/19/82	13.55.00
RESTOR	FORTRAN	D1	F	80	148	12	10/19/82	13.55.00
SETCON	FORTRAN	D1	F	80	108	9	10/20/82	14.43.00
SETPAR	FORTRAN	D1	F	80	50	4	10/20/82	14.43.00
SHIFTR	FORTRAN	D1	F	80	79	7	10/19/82	13.55.00
SKIP8	FORTRAN	D1	F	80	71	6	10/19/82	13.55.00
SKIPOP	FORTRAN	D1	F	80	74	6	10/19/82	13.55.00
SKIP2B	FORTRAN	D1	F	80	79	7	10/19/82	13.55.00
SKIP2N	FORTRAN	D1	F	80	80	7	10/19/82	13.55.00
SKIP2P	FORTRAN	D1	F	80	74	6	10/19/82	13.55.00
TERARC	FORTRAN	D1	F	80	201	16	10/20/82	15.24.00
TERPRO	FORTRAN	D1	F	80	318	25	10/20/82	15.25.00
TERSUP	FORTRAN	D1	F	80	2406	188	10/20/82	15.25.00
TRANCO	FORTRAN	D1	F	80	129	11	10/20/82	14.43.00
TROUBL	FORTRAN	D1	F	80	112	9	10/19/82	13.55.00
VICMER	FORTRAN	D1	F	80	314	25	10/19/82	13.55.00
WRTDES	FORTRAN	D1	F	80	152	12	10/19/82	13.55.00
ZERO	FORTRAN	D1	F	80	79	7	10/19/82	13.55.00
MYFDIG	FTN	D1	F	80	679	55	10/22/82	14.48.00
DATABASE	LOGFILE	D1	F	80	159	14	10/21/82	10.26.00

CONCLUSIONS AND RECOMMENDATIONS

Conclusions

Bolivia as many other developing countries in the world has a tremendous need for basic, up-to-date, and readily available information about its resources. To provide planners and decision makers with the required information, a new data management technology (digital Geographic Information System) was designed for Bolivia.

The system is an invaluable tool for a broad range of applications throughout the different phases of a development project, such as its planning and preparation, monitoring, and evaluation.

In addition, this system would be useful not only for more effective and efficient storage, management, analysis and retrieval of information, but it can become the basis for an integrated interinstitutional communications network.

Recommendations

In order to successfully implement an operational digital Geographic Information System in Bolivia, the following recommendations are provided:

- Continue with the development of the system.
- Since one of the factors that allowed the success of this project was the invaluable contribution of the Bolivian natural resources expert that was in residence at Purdue/LARS for the entire duration of the project, it is recommended that also a computer

science expert should be assigned to work closely with the Purdue/LARS team.

- Continue to obtain the resource information (maps) at higher levels of detail (larger scales) for input into the data base, using both remote sensing technology and conventional methods.
- Continue to create digital Landsat MSS mosaics that eventually would cover the entire Bolivian territory.
- Develop hierarchical classification schemes (legends) for the various elements (disciplines) that would be included in the data base.
- Once the Input Subsystem is completely implemented in Bolivia, it is recommended that the process of digitization be continued, particularly the digitization of the 1:50,000 topographic maps.
- To implement the Bolivian GIS on an operational basis, it is recommended that the required specialized hardware be purchased.
- Finally, in order to harness the full potential of the system, it is of paramount importance to continue with the process of transferring the technology within Bolivia to the different institutions that deal with planning and decision-making at the national, departmental, and local levels of government.

APPENDIX A

```
10  REM  DIGITIZATION PROGRAMS 9/21/82
20  HIMEM: 38400:X =  FRE (0)
30 D$ =  CHR$ (4)
40  HOME
50  PRINT "DIGITIZING SYSTEM PURDUE/LARS  9/14/82": PRINT
60  HTAB (10): INVERSE : PRINT "DIGITIZING": NORMAL
70  PRINT "       1-DIGITIZE CHECK POINTS OR": PRINT "          CHECKPOINTS"
80  PRINT "       2-DIGITIZE ARCS"
90  PRINT "       3-DIGITIZE LINEAR FEATURES"
95  PRINT
100  HTAB (10): INVERSE : PRINT "DATA TRANSFER": NORMAL
110  PRINT "        4-SEND CENTROIDS TO HOST"
120  PRINT "        5-SEND CHECK POINTS TO HOST"
130  PRINT "        6-SEND ARCS OR LINEAR FEATURES": PRINT "          TO HOST"
135  PRINT
140  HTAB (10): INVERSE : PRINT "UTILITIES": NORMAL
150  PRINT "        7-PLOT ARCS ON APPLE"
160  PRINT "        8-RUN APPLE AS A TERMINAL"
170  PRINT "        9-RETURN TO APPLE BASIC"
180  PRINT "SELECT:";
190  GET SL$: PRINT SL$
195  HOME : PRINT "LOADING SELECTED PROGRAM"
200  ON  VAL (SL$) GOTO 220,230,240,250,260,270,280,290,370
210  GOTO 40
220  PRINT D$;"RUN POINTS"
230  PRINT D$;"RUN ARCS"
240  PRINT D$;"RUN LINFEAT"
250  PRINT D$;"RUN SENDCEN"
260  PRINT D$;"RUN SENDCKP"
270  PRINT D$;"RUN SENDARCS"
280  PRINT D$;"RUN PLOTARCS"
290  HOME
300  INVERSE : PRINT "TERMINAL MODE  TYPE CTRL Z TO EXIT": NORMAL
310  POKE 34,1
320  PRINT D$;"BRUN TERM.BIN"
330  POKE 34,0
340  GOTO 40
350  REM
360  REM
370  HOME
380  END
```

```
10  REM   ARC ROUTINE 9/21/82
20  HIMEM: 8191
30  HOME
40 MV = 0.02
50 LC = 1.41421
60 SC = 1.11111111
70 SS% = 255
80 LS = 256
90 MS = 512
100 D$ =  CHR$ (4)
110  PRINT D$;"BLOAD DIGPT.BIN,D1"
120  PRINT D$;"BLOAD MPTS.BIN,D1"
130  PRINT D$;"BLOAD ONERR.BIN,D1"
140  INPUT "ENTER ARC FILENAME: ";FI$
150  HOME
160  IF FI$ = "" THEN 140
170  PRINT "MARK TOP LEFT CORNER WITH CURSOR"
180  GOSUB 1880
190 PA = 24588
200  GOSUB 1790
210 XO = X
220 YO = Y
230  HOME : PRINT "MARK TOP RIGHT CORNER WITH CURSOR"
240  GOSUB 1880
250  IF  ABS (X - XO) < 1000 THEN 230
260  GOSUB 1790
270  HOME
280  HOME : PRINT "MARK BOTTOM RIGHT CORNER WITH CURSOR"
290  GOSUB 1880
300  IF  ABS (Y - YO) < 1000 THEN 280
310  GOSUB 1790
320 XI = X - XO
330 YI = YO - Y
340 X2 = 251.1 / XI
350 Y2 = 191 / YI
360  IF Y2 < X2 THEN X2 = Y2
370 XI = XI * X2 * SC
380 YI = YI * X2
390  HCOLOR= 3
400 ND% = 1
410  HOME : PRINT "CLEAR ALL ARCS IN PICTURE?(Y OR N) ";
420  GET SL$: PRINT SL$
430  IF SL$ = "N" THEN 480
440  IF SL$ <  > "Y" THEN 410
450 ND% = 0
460  HGR2
470  HPLOT 0,0 TO XI,0 TO XI,YI TO 0,YI TO 0,0
480  HGR
490 ND% = 0
500 AN% = 1
510  GOSUB 2110
520  REM
530  HOME : TEXT : HGR
540  PRINT "ENTER ARC ";AN%;" AREA LEFT:";
```

```
550  INPUT "  ";AL%
560  IF AL% >  = 0 GOTO 590
570  GOSUB 2110
580  GOTO 530
590  PRINT "ENTER ARC ";AN%;" AREA RIGHT:";
600  INPUT " ";AR%
610  IF AR% >  = 0 GOTO 640
620  GOSUB 2110
630  GOTO 530
640 V% = AN% / LS
650  POKE 24577,V%
660 V% = (AN% / LS - V%) * LS
670  POKE 24576,V%
680 V% = AL% / LS
690  POKE 24581,V%
700 V% = (AL% / LS - V%) * LS
710  POKE 24580,V%
720 V% = AR% / LS
730  POKE 24583,V%
740 V% = (AR% / LS - V%) * LS
750  POKE 24582,V%
760  REM
770  PRINT "DIGITIZE ARC ";AN%; CHR$ (7)
780  GOSUB 2590
790  CALL 37649
800 NP =  PEEK (24578) + LS *  PEEK (24579)
810  IF NP < 2 THEN 790
820 XL =  PEEK (24584) * MS
830 XB =  PEEK (24585) * MS
840 YL =  PEEK (24586) * MS
850 YB =  PEEK (24587) * MS
860 X1 = 251.1 / (XB - XL)
870 Y1 = 159 / (YB - YL)
880  IF Y1 < X1 THEN X1 = Y1
890 WX =  INT (1 / X1 * 25.11) / 100
900 WY =  INT (1 / X1 * 16) / 100
910  PRINT WX;" X ";WY;" INCH WINDOW"
920  PRINT NP;" POINTS IN ARC"
930 V% = 4 / X1 + .5
940 SL% = V% / LS
950  POKE 37645,SL%
960 SL% = (V% / LS - SL%) * LS
970  POKE 37644,SL%
980  CALL 38075
990 XS = X1
1000 XF = XL
1010 YF = YB
1020  GOSUB 2720
1030  HPLOT X,Y
1040  CALL 38132
1050  GOSUB 2720
1060  HPLOT  TO X,Y
1070  IF  PEEK (37648) <  > 0 THEN 1040
1080  REM
1090  PRINT "SAVE THIS ARC? ";: GET SL$: PRINT SL$
```

```
 1100  IF SL$ =  CHR$ (78) THEN 530
 1110  IF SL$ <  >  CHR$ (89) THEN 1090
 1120 ND% = 1
 1130  POKE 49234,00
 1140  POKE 49237,00
 1150  POKE 230,64
 1160 V% = 4 / X2 + .5
 1170 SL% = V% / LS
 1180  POKE 37645,SL%
 1190 SL% = (V% / LS - SL%) * LS
 1200  POKE 37644,SL%
 1210  CALL 38075
 1220 XS = X2
 1230 XF = XO
 1240 YF = YO
 1250  GOSUB 2720
 1260  IF X < 0 THEN X = 0
 1270  IF X > 279 THEN X = 279
 1280  IF Y < 0 THEN Y = 0
 1290  IF Y > 191 THEN Y = 191
 1300  HPLOT X,Y
 1310  CALL 38132
 1320  GOSUB 2720
 1330  IF X < 0 THEN X = 0
 1340  IF X > 279 THEN X = 279
 1350  IF Y < 0 THEN Y = 0
 1360  IF Y > 191 THEN Y = 191
 1370  HPLOT  TO X,Y
 1380  IF  PEEK (37648) <  > 0 THEN 1310
 1390  REM
 1400 NB% = NP * 4 + 128
 1410  ONERR  GOTO 1490
 1420  PRINT D$;"UNLOCK ";FI$;"/ARC"; STR$ (AN%);",D2"
 1430  POKE 216,00
 1440  POKE 49236,00: POKE 49233,00
 1450  HOME : TEXT
 1460  INVERSE : PRINT FI$;"/ARC";AN%;" ALREADY EXISTS": NORMAL
 1470  GOSUB 2120
 1480  GOTO 1410
 1490  POKE 216,0
 1500  CALL 38390
 1510  PRINT : PRINT : PRINT
 1520  ONERR  GOTO 1950
 1530  PRINT D$;"BSAVE ";FI$;"/ARC"; STR$ (AN%);",A$6000,L"; STR$ (NB%);",D2"
 1540  POKE 216,0
 1550 AN% = AN% + 1
 1560  REM
 1570  POKE 49236,00
 1580  PRINT "SELECT: 1-DISPLAY ARC"
 1590  PRINT "         2-DISPLAY PICTURE"
 1600  PRINT "         3-DIGITIZE NEXT ARC"
 1610  PRINT "         9-GO TO MENU";
 1620  POKE 49235,00
 1630  GET SL$
 1640  ON  VAL (SL$) GOTO 1670,1710,520,1630,1630,1630,1630,1630,1750
```

```
1650  GOTO 1630
1660  REM
1670  POKE 49236,00
1680  POKE 49235,00
1690  GOTO 1630
1700  REM
1710  POKE 49234,00
1720  POKE 49237,00
1730  GOTO 1630
1740  REM
1750  GOSUB 2110
1760  GOTO 1570
1770  REM
1780  REM
1790 GA = 37312
1800  FOR K = 0 TO 3
1810 V% =  PEEK (GA + K)
1820  POKE PA,V%
1830 PA = PA + 1
1840  NEXT K
1850  RETURN
1860  REM
1870  REM
1880  CALL 37318
1890 X =  PEEK (37312) + LS *  PEEK (37313)
1900 Y =  PEEK (37314) + LS *  PEEK (37315)
1910  PRINT  CHR$ (7)
1920  RETURN
1930  REM
1940  REM
1950  POKE 49236,00: POKE 49235,00
1960  POKE 216,0
1970 ER =  PEEK (222)
1980  IF ER = 9 THEN 2010
1990  PRINT "DISK ERROR ";ER;" - CORRECT PROBLEM THEN"
2000  GOTO 2060
2010  PRINT "DISKETTE IS FULL"
2020  ONERR  GOTO 2040
2030  PRINT D$;"DELETE ";FI$;"/ARC"; STR$ (AN%)
2040  POKE 216,0
2050  PRINT "INSERT EMPTY DISKETTE THEN"
2060  PRINT "HIT ANY KEY TO CONTINUE"
2070  GET SL$
2080  GOTO 1410
2090  REM
2100  REM
2110  POKE 49236,00: HOME : TEXT
2120  PRINT "          1-DIGITIZE ARCS"
2130  PRINT "          2-CHANGE FILENAME"
2140  PRINT "          3-CHANGE ARC NUMBER"
2150  PRINT "          4-CHANGE MIN VECTOR LENGTH"
2160  PRINT "          5-LIST FILES"
2170  IF ND% < 1 THEN 2200
2180  PRINT "          6-DISPLAY ARC"
2190  PRINT "          7-DISPLAY PICTURE"
```

```
2200  PRINT "         9-STOP"
2210  PRINT "SELECT: ";
2220  GET SL$: PRINT SL$
2230  ON  VAL (SL$) GOTO 2260,2290,2350,2390,2440,2500,2540,2210,60000
2240  GOTO 2210
2250  REM
2260  POKE 49232,00
2270  RETURN
2280  REM
2290  PRINT "CURRENT FILENAME IS ";FI$
2300  INPUT "ENTER NEW FILENAME: ";FI$
2310  IF FI$ <  > "" THEN 2110
2320  HOME
2330  GOTO 2290
2340  REM
2350  PRINT "   NEXT ARC NUMBER IS ";AN%
2360  INPUT "ENTER NEW ARC NUMBER: ";AN%
2370  GOTO 2110
2380  REM
2390  PRINT "CURRENT MIN VECTOR = ";MV;" INCHES"
2400  INPUT "ENTER NEW MIN VECTOR(INCHES): ";MV
2410  GOSUB 2590
2420  GOTO 2110
2430  REM
2440  HOME
2450  PRINT D$;"CATALOG,D2"
2460  INVERSE : PRINT "HIT ANY KEY TO CONTINUE": NORMAL
2470  GET SL$
2480  GOTO 2110
2490  REM
2500  POKE 49234,00: POKE 49236,00: POKE 49232,00
2510  FOR IW = 1 TO 2000: NEXT IW
2520  GOTO 2110
2530  REM
2540  POKE 49234,00: POKE 49237,00: POKE 49232,00
2550  FOR IW = 1 TO 2000: NEXT IW
2560  GOTO 2110
2570  REM
2580  REM
2590 V% = 1000 * MV * LC
2600 SL% = V% / LS
2610  POKE 37645,SL%
2620 SL% = (V% / LS - SL%) * LS
2630  POKE 37644,SL%
2640 V% = 3000 * MV * LC
2650 SL% = V% / LS
2660  POKE 37647,SL%
2670 SL% = (V% / LS - SL%) * LS
2680  POKE 37646,SL%
2690  RETURN
2700  REM
2710  REM
2720 X =  PEEK (37636) + LS *  PEEK (37637)
2730 Y =  PEEK (37638) + LS *  PEEK (37639)
2740 X = (X - XF) * XS * SC
```

```
2750 Y = (YF - Y) * XS
2760  RETURN
60000  HOME
60010  PRINT "WAIT JUST A SECOND"
60020  PRINT D$;"RUN DIGITIZE,D1"
60030  END
```

```
10  REM   CHECK POINT AND CENTROID ROUTINE 9/21/82
20  REM  INITIALIZE MAP SIZE AND POSITION
30  REM  ********************************************************************************
40  LOMEM: 16384
50  HIMEM: 37311
60  DIM LT(100),LO(100),X%(500),Y%(500),XM%(500),YM%(500),CN%(500)
70 D$ =  CHR$ (4)
80 SC = 1.11111111
90  PRINT D$;"BLOAD DIGPT.BIN,D1"
100  PRINT D$;"BLOAD TERM.BIN,D1"
110  PRINT D$;"BLOAD SEND.BIN,D1"
120  TEXT : HOME
130  PRINT "MARK TOP LEFT CORNER WITH CURSOR";
140  GOSUB 4020
150 XOG = X
160 YOG = Y
170  PRINT "MARK TOP RIGHT CORNER WITH CURSOR";
180  GOSUB 4020
190 XAXIS = X
200 YAXIS = Y
210 A = YAXIS - YOG
220 B = XAXIS - XOG
230 C =  SQR (A * A + B * B)
240 CO = B / C
250 SI = A / C
280  REM
290  REM
300  PRINT "MARK BOTTOM RIGHT CORNER WITH CURSOR";
310  GOSUB 4020
340 XL = CO * (X - XOG) + SI * (Y - YOG)
350 YL = SI * (X - XOG) - CO * (Y - YOG)
360  IF XL < 100 THEN 120
370  IF YL < 100 THEN 120
380 XS = 251.1 / XL
390 YS = 159 / YL
400  IF YS < XS THEN XS = YS
410 XI = XL * XS * SC
420 YI = YL * XS
430 X3 = XI - 3
440 Y3 = YI - 3
450  HGR
460  HCOLOR= 3
470  HPLOT 0,0 TO XI,0 TO XI,YI TO 0,YI TO 0,0
480  REM
490  REM  SET UP TYPE OF DATA
500  REM
510  POKE 232,64
520  POKE 233,147
530  POKE 36,0
540  POKE 37,22
550  PRINT
560  ONERR  GOTO 2030
570  PRINT "ENTER   1-CHECK POINTS"
580  PRINT "        2-CENTROIDS"
```

```
590  INPUT "SELECT: ";TY
600  PRINT
610 NP = 500
620  ON TY GOTO 640,670
630  GOTO 570
640  INPUT "MAP SCALE 1:";MS
650  INPUT "ENTER 1-ROW,COL OR 2-LAT,LONG: ";KEY
660 NP = 100
670  FOR I = 1 TO NP
680  ONERR  GOTO 2030
690 LP = I
700  GOSUB 7030
710  POKE 216,0
720  PRINT  CHR$ (7);
730  REM
740  REM  GET DATA POINT AND PLOT IT
750  REM
760  PRINT : PRINT "MARK POINT WITH CURSOR";
770  GOSUB 4020
800 X = X - XOG
810 Y = Y - YOG
820 X = CO * X + SI * Y
830  IF X < 0 THEN 980
840  IF X > XL THEN 980
850 X%(I) = X + .5
860 Y = SI * X - CO * Y
870  IF Y < 0 THEN 980
880  IF Y > YL THEN 980
890 Y%(I) = Y + .5
900 XM%(I) = X%(I) * XS * SC
910 YM%(I) = Y%(I) * XS
920  HPLOT XM%(I),YM%(I)
930 M = I
940  GOSUB 8030
950  NEXT I
960  REM
970  REM
980  INVERSE : PRINT "POINT OUTSIDE OF DEFINED AREA": NORMAL
990  GOTO 760
2000  REM
2010  REM  HANDLE ERROR
2020  REM
2030  POKE 216,0
2040  PRINT "ENTER:  1-TO CONTINUE"
2050  PRINT "        2-EDIT"
2060  PRINT "        3-QUIT"
2070  INPUT "SELECT: ";SEL
2080  ON SEL GOTO 2100,3030,5030
2090  GOTO 2040
2100  ONERR  GOTO 2030
2110 LP = I
2120  RESUME
3000  REM
3010  REM  EDIT DATA
3020  REM
```

```
3030 IC = I - 1: IF IC < 1 THEN 3340
3040 M = IC: GOSUB 8030
3050  PRINT "USE <- OR -> TO SELECT PT,<CR>TO EDIT"
3060  GOSUB 9030
3070  IF SX = 149 THEN 3110: REM  RIGHT
3080  IF SX = 136 THEN 3150: REM  LEFT
3090  IF SX = 141 THEN 3190: REM  EDIT
3100  GOTO 3040
3110  HCOLOR= 3: HPLOT XM%(IC),YM%(IC)
3120 IC = IC + 1
3130  IF IC > I - 1 THEN IC = 1
3140  GOTO 3040
3150  HCOLOR= 3: HPLOT XM%(IC),YM%(IC)
3160 IC = IC - 1
3170  IF IC < 1 THEN IC = I - 1
3180  GOTO 3040
3190 M = IC: GOSUB 8030: PRINT "0-EXIT,1-CHANGE,2-DELETE,3-EDIT"
3200  HCOLOR= 3: HPLOT XM%(IC),YM%(IC)
3210  GOSUB 9030
3220 SEL = SX - 176: REM  176 IS 0
3230  IF SEL < 0 OR SEL > 3 THEN 3190: REM  WRONG KEY
3240  ON SEL GOTO 3260,3290,3040
3250  GOTO 2040: REM   ON ZERO
3260 LP = IC:M = IC: GOSUB 8030
3270  GOSUB 7030
3280  GOTO 3040
3290 N = IC: HCOLOR= 4
3300  HPLOT XM%(N),YM%(N)
3310  HCOLOR= 0: HPLOT XM%(IC),YM%(IC)
3320 I = I - 1:IC = IC - 1: IF IC < 1 THEN IC = I - 1
3330  IF I > 1 THEN 3360
3340  PRINT "NO POINTS LEFT"; CHR$ (7); CHR$ (7)
3350  FOR DL = 1 TO 200: NEXT DL: HCOLOR= 3: GOTO 2040
3360  IF N = I THEN 3040
3370  FOR J = N + 1 TO I
3380 X%(J - 1) = X%(J)
3390 Y%(J - 1) = Y%(J)
3400 XM%(J - 1) = XM%(J)
3410 YM%(J - 1) = YM%(J)
3420 LT(J - 1) = LT(J)
3430 LO(J - 1) = LO(J)
3440 CN%(J - 1) = CN%(J)
3450  NEXT J
3460  GOTO 3040
3470  REM
4000  REM  CONVERT DIGITIZER DATA
4010  REM
4020  CALL 37318
4030 X =  PEEK (37312) + 256 *  PEEK (37313)
4040 Y =  PEEK (37314) + 256 *  PEEK (37315)
4050  PRINT  CHR$ (7)
4060  RETURN
5000  REM
5010  REM  SEND DATA TO HOST OR SAVE ON DISK
5020  REM
```

```
5030  IF I > 1 THEN 5060
5040  PRINT "NO DATA POINTS"
5050  GOTO 60000
5060  TEXT
5070  HOME
5080  PRINT "ENTER:  1-SEND TO IBM"
5090  PRINT "        2-SAVE ON DISK"
5100  PRINT "        3-DIGITIZE POINTS"
5110  PRINT "        4-QUIT"
5120  INPUT "SELECT: ";SEL
5130  ON SEL GOTO 5150,5750,450,60000
5140  GOTO 5080
5150  HOME
5160  INVERSE : PRINT "LOGON AND EXECUTE DIGRECV": NORMAL
5170  POKE 34,1
5180  CALL 37888
5190  POKE 34,0
5200  CALL 37826
5210  HOME
5220  PRINT "SENDING DATA"
5230  ON TY GOTO 5240,5320
5240 ST$ =  CHR$ (16)
5250  GOSUB 6030
5260 ST$ = "2 "
5270  GOSUB 6030
5272 ST$ =  CHR$ (13)
5274  GOSUB 6030
5280 ST$ =  STR$ (MS)
5290  GOSUB 6030
5300 ST$ =  CHR$ (13)
5310  GOSUB 6030
5320  FOR II = 1 TO I - 1
5330 ST$ =  CHR$ (16)
5340  GOSUB 6030
5350 ST$ =  CHR$ (50 + TY)
5360  GOSUB 6030
5370 ST$ =  CHR$ (13)
5380  GOSUB 6030
5390 ST$ =  STR$ (X%(II))
5400  GOSUB 6030
5410 ST$ =  CHR$ (44)
5420  GOSUB 6030
5430 ST$ =  STR$ (Y%(II))
5440  GOSUB 6030
5450 ST$ =  CHR$ (44)
5460  GOSUB 6030
5470  ON TY GOTO 5480,5550
5480 ST$ =  STR$ (LO(II))
5490  GOSUB 6030
5500 ST$ =  CHR$ (44)
5510  GOSUB 6030
5520 ST$ =  STR$ (LT(II))
5530  GOSUB 6030
5540  GOTO 5610
5550 ST$ =  STR$ (CN%(II))
```

```
5560  GOSUB 6030
5570 ST$ =  CHR$ (44)
5580  GOSUB 6030
5590 ST$ =  STR$ (II)
5600  GOSUB 6030
5610 ST$ =  CHR$ (13)
5620  GOSUB 6030
5630  NEXT II
5640 ST$ =  CHR$ (16)
5650  GOSUB 6030
5660 ST$ = "9 "
5670  GOSUB 6030
5680 ST$ =  CHR$ (13)
5690  GOSUB 6030
5700  HOME : INVERSE : PRINT "AFTER LOGGING OFF TYPE CNTRL Z TO EXIT": NORMAL
5710  POKE 34,1
5720  CALL 37888
5730  POKE 34,0
5740  GOTO 5070
5750  INPUT "ENTER FILENAME: ";FILE$
5760  PRINT D$;"OPEN ";FILE$;",D2"
5770  PRINT D$;"WRITE ";FILE$
5780  ON TY GOTO 5790,5800
5790  PRINT MS
5800  FOR II = 1 TO I - 1
5810  PRINT X%(II);" ";Y%(II);" ";
5820  ON TY GOTO 5830,5850
5830  PRINT LO(II);" ";LT(II)
5840  GOTO 5860
5850  PRINT CN%(II)
5860  NEXT II
5870  PRINT D$;"CLOSE ";FILE$
5880  GOTO 5070
6000  REM
6010  REM  BREAK STRING INTO CHARS AND SEND THEM TO HOST
6020  REM
6030 L =  LEN (ST$)
6040  FOR N = 1 TO L
6050 CH$ =  MID$ (ST$,N,1)
6060 CH% =  ASC (CH$)
6070  POKE 37824,CH%
6080  CALL 37837
6090  NEXT N
6100  RETURN
7000  REM
7010  REM  GET DATA PARAMETERS
7020  REM
7030  PRINT : ON TY GOTO 7040,7130
7040  ON KEY GOTO 7050,7080
7050  INPUT "ROW NUMBER: ";LT(LP)
7060  INPUT "COL NUMBER: ";LO(LP)
7070  RETURN
7080  INPUT "ENTER LATITUDE (D,M,S): ";DEG,MIN,SEC
7090 LT(LP) = DEG + MIN / 60 + SEC / 3600
7100  INPUT "ENTER LONGITUDE (D,M,S): ";DEG,MIN,SEC
```

```
7110 LO(LP) = DEG + MIN / 60 + SEC / 3600
7120  RETURN
7130  INPUT "ENTER CODE FOR CENTROID: ";CN%(LP)
7140  RETURN
8000  REM
8010  REM  PRINT DATA PARAMETERS
8020  REM
8030  ON TY GOTO 8040,8090
8040  ON KEY GOTO 8070,8050
8050  PRINT "LAT = ";LT(M);"  LONG = ";LO(M)
8060  GOTO 8100
8070  PRINT "ROW = ";LT(M);" COLUMN = ";LO(M)
8080  GOTO 8100
8090  PRINT "CENTROID(";M;") CODE = ";CN%(M)
8100  PRINT "X = ";X%(M);" Y = ";Y%(M)
8130  RETURN
9000  REM
9010  REM  POINT BLINK & KBD SCAN
9020  REM
9030  HCOLOR= 1
9040  HPLOT XM%(IC),YM%(IC)
9050  FOR DL = 1 TO 100: NEXT DL
9060  HCOLOR= 6
9070  HPLOT XM%(IC),YM%(IC)
9080  FOR DL = 1 TO 100: NEXT DL
9090 SX =  PEEK ( - 16384): POKE  - 16368,0
9100  IF SX < 127 THEN 9030: REM  NO KEY
9110  HCOLOR= 3: HPLOT XM%(IC),YM%(IC)
9120  RETURN
60000  HOME : PRINT "WAIT JUST A SECOND"
60010  PRINT D$;"RUN DIGITIZE,D1"
60020  END
```

```
10  REM   LINEAR FEATURE ROUTINE 9/21/82
20  HIMEM: 8191
30  HOME
40 MV = 0.02
50 LC = 1.41421
60 SC = 1.11111111
70 SS% = 255
80 LS = 256
90 MS = 512
100 D$ =  CHR$ (4)
110  PRINT D$;"BLOAD DIGPT.BIN,D1"
120  PRINT D$;"BLOAD MPTS.BIN,D1"
130  PRINT D$;"BLOAD ONERR.BIN,D1"
140  INPUT "ENTER LINEAR FEATURE FILENAME: ";FI$
150  HOME
160  IF FI$ = "" THEN 140
170  PRINT "MARK TOP LEFT CORNER WITH CURSOR"
180  GOSUB 1800
190 PA = 24588
200  GOSUB 1710
210 XO = X
220 YO = Y
230  HOME : PRINT "MARK TOP RIGHT CORNER WITH CURSOR"
240  GOSUB 1800
250  IF  ABS (X - XO) < 1000 THEN 230
260  GOSUB 1710
270  HOME
280  HOME : PRINT "MARK BOTTOM RIGHT CORNER WITH CURSOR"
290  GOSUB 1800
300  IF  ABS (Y - YO) < 1000 THEN 280
310  GOSUB 1710
320 XI = X - XO
330 YI = YO - Y
340 X2 = 251.1 / XI
350 Y2 = 191 / YI
360  IF Y2 < X2 THEN X2 = Y2
370 XI = XI * X2 * SC
380 YI = YI * X2
390  HCOLOR= 3
400 ND% = 1
410  HOME : PRINT "CLEAR ALL ARCS IN PICTURE?(Y OR N) ";
420  GET SL$: PRINT SL$
430  IF SL$ = "N" THEN 480
440  IF SL$ <  > "Y" THEN 410
450 ND% = 0
460  HGR2
470  HPLOT 0,0 TO XI,0 TO XI,YI TO 0,YI TO 0,0
480  HGR
490 ND% = 0
500 AN% = 1
510  GOSUB 2030
520  REM
530  HOME : TEXT : HGR
540  INPUT "ENTER LINEAR FEATURE CODE: ";LF%
```

```
550  IF LF% >  = 0 GOTO 580
560  GOSUB 2030
570  GOTO 530
580 V% = AN% / LS
590  POKE 24577,V%
600 V% = (AN% / LS - V%) * LS
610  POKE 24576,V%
620 V% = LF% / LS
630  POKE 24581,V%
640  POKE 24583,V%
650 V% = (LF% / LS - V%) * LS
660  POKE 24580,V%
670  POKE 24582,V%
680  REM
690  PRINT "DIGITIZE LINEAR FEATURE ";AN%; CHR$ (7)
700  GOSUB 2510
710  CALL 37649
720 NP =  PEEK (24578) + LS *  PEEK (24579)
730  IF NP < 2 THEN 710
740 XL =  PEEK (24584) * MS
750 XB =  PEEK (24585) * MS
760 YL =  PEEK (24586) * MS
770 YB =  PEEK (24587) * MS
780 X1 = 251.1 / (XB - XL)
790 Y1 = 159 / (YB - YL)
800  IF Y1 < X1 THEN X1 = Y1
810 WX =  INT (1 / X1 * 25.11) / 100
820 WY =  INT (1 / X1 * 16) / 100
830  PRINT WX;" X ";WY;" INCH WINDOW"
840  PRINT NP;" POINTS IN LINEAR FEATURE"
850 V% = 4 / X1 + .5
860 SL% = V% / LS
870  POKE 37645,SL%
880 SL% = (V% / LS - SL%) * LS
890  POKE 37644,SL%
900  CALL 38075
910 XS = X1
920 XF = XL
930 YF = YB
940  GOSUB 2640
950  HPLOT X,Y
960  CALL 38132
970  GOSUB 2640
980  HPLOT  TO X,Y
990  IF  PEEK (37648) <  > 0 THEN 960
1000  REM
1010  PRINT "SAVE THIS LINEAR FEATURE? ": GET SL$: PRINT SL$
1020  IF SL$ =  CHR$ (78) THEN 530
1030  IF SL$ <  >  CHR$ (89) THEN 1010
1040 ND% = 1
1050  POKE 49234,00
1060  POKE 49237,00
1070  POKE 230,64
1080 V% = 4 / X2 + .5
1090 SL% = V% / LS
```

```
1100  POKE 37645,SL%
1110 SL% = (V% / LS - SL%) * LS
1120  POKE 37644,SL%
1130  CALL 38075
1140 XS = X2
1150 XF = XO
1160 YF = YO
1170  GOSUB 2640
1180  IF X < 0 THEN X = 0
1190  IF X > 279 THEN X = 279
1200  IF Y < 0 THEN Y = 0
1210  IF Y > 191 THEN Y = 191
1220  HPLOT X,Y
1230  CALL 38132
1240  GOSUB 2640
1250  IF X < 0 THEN X = 0
1260  IF X > 279 THEN X = 279
1270  IF Y < 0 THEN Y = 0
1280  IF Y > 191 THEN Y = 191
1290  HPLOT  TO X,Y
1300  IF  PEEK (37648) <  > 0 THEN 1230
1310  REM
1320 NB% = NP * 4 + 128
1330  ONERR  GOTO 1410
1340  PRINT D$;"UNLOCK ";FI$;"/LIN"; STR$ (AN%);",D2"
1350  POKE 216,00
1360  POKE 49236,00: POKE 49233,00
1370  HOME : TEXT
1380  INVERSE : PRINT FI$;"/LIN";AN%;" ALREADY EXISTS": NORMAL
1390  GOSUB 2040
1400  GOTO 1330
1410  POKE 216,0
1420  CALL 38390
1430  PRINT : PRINT : PRINT
1440  ONERR  GOTO 1870
1450  PRINT D$;"BSAVE ";FI$;"/LIN"; STR$ (AN%);",A$6000,L"; STR$ (NB%);",D2"
1460  POKE 216,0
1470 AN% = AN% + 1
1480  REM
1490  POKE 49236,00
1500  PRINT "SELECT: 1-DISPLAY LINEAR FEATURE"
1510  PRINT "         2-DISPLAY PICTURE"
1520  PRINT "         3-DIGITIZE NEXT LINEAR FEATURE"
1530  PRINT "         9-GO TO MENU";
1540  POKE 49235,00
1550  GET SL$
1560  ON  VAL (SL$) GOTO 1590,1630,520,1550,1550,1550,1550,1550,1670
1570  GOTO 1550
1580  REM
1590  POKE 49236,00
1600  POKE 49235,00
1610  GOTO 1550
1620  REM
1630  POKE 49234,00
1640  POKE 49237,00
```

```
1650  GOTO 1550
1660  REM
1670  GOSUB 2030
1680  GOTO 1490
1690  REM
1700  REM
1710 GA = 37312
1720  FOR K = 0 TO 3
1730 V% =  PEEK (GA + K)
1740  POKE PA,V%
1750 PA = PA + 1
1760  NEXT K
1770  RETURN
1780  REM
1790  REM
1800  CALL 37318
1810 X =  PEEK (37312) + LS *  PEEK (37313)
1820 Y =  PEEK (37314) + LS *  PEEK (37315)
1830  PRINT  CHR$ (7)
1840  RETURN
1850  REM
1860  REM
1870  POKE 49236,00: POKE 49235,00
1880  POKE 216,0
1890 ER =  PEEK (222)
1900  IF ER = 9 THEN 1930
1910  PRINT "DISK ERROR ";ER;" - CORRECT PROBLEM THEN"
1920  GOTO 1980
1930  PRINT "DISKETTE IS FULL"
1940  ONERR  GOTO 1960
1950  PRINT D$;"DELETE ";FI$;"/ARC"; STR$ (AN%)
1960  POKE 216,0
1970  PRINT "INSERT EMPTY DISKETTE THEN"
1980  PRINT "HIT ANY KEY TO CONTINUE"
1990  GET SL$
2000  GOTO 1330
2010  REM
2020  REM
2030  POKE 49236,00: HOME : TEXT
2040  PRINT "         1-DIGITIZE LINEAR FEATURES"
2050  PRINT "         2-CHANGE FILENAME"
2060  PRINT "         3-CHANGE LINEAR FEATURE NUMBER"
2070  PRINT "         4-CHANGE MIN VECTOR LENGTH"
2080  PRINT "         5-LIST FILES"
2090  IF ND% < 1 THEN 2120
2100  PRINT "         6-DISPLAY LINEAR FEATURE"
2110  PRINT "         7-DISPLAY PICTURE"
2120  PRINT "         9-STOP"
2130  PRINT "SELECT: ";
2140  GET SL$: PRINT SL$
2150  ON  VAL (SL$) GOTO 2180,2210,2270,2310,2360,2420,2460,2130,60000
2160  GOTO 2130
2170  REM
2180  POKE 49232,00
2190  RETURN
```

```
2200  REM
2210  PRINT "CURRENT FILENAME IS ";FI$
2220  INPUT "ENTER NEW FILENAME: ";FI$
2230  IF FI$ <  > "" THEN 2030
2240  HOME
2250  GOTO 2210
2260  REM
2270  PRINT "   NEXT LINEAR FEATURE NUMBER IS ";AN%
2280  INPUT "ENTER NEW LINEAR FEATURE NUMBER  ";AN%
2290  GOTO 2030
2300  REM
2310  PRINT "CURRENT MIN VECTOR = ";MV;" INCHES"
2320  INPUT "ENTER NEW MIN VECTOR(INCHES): ";MV
2330  GOSUB 2510
2340  GOTO 2030
2350  REM
2360  HOME
2370  PRINT D$;"CATALOG,D2"
2380  INVERSE : PRINT "HIT ANY KEY TO CONTINUE": NORMAL
2390  GET SL$
2400  GOTO 2030
2410  REM
2420  POKE 49234,00: POKE 49236,00: POKE 49232,00
2430  FOR IW = 1 TO 2000: NEXT IW
2440  GOTO 2030
2450  REM
2460  POKE 49234,00: POKE 49237,00: POKE 49232,00
2470  FOR IW = 1 TO 2000: NEXT IW
2480  GOTO 2030
2490  REM
2500  REM
2510 V% = 1000 * MV * LC
2520 SL% = V% / LS
2530  POKE 37645,SL%
2540 SL% = (V% / LS - SL%) * LS
2550  POKE 37644,SL%
2560 V% = 3000 * MV * LC
2570 SL% = V% / LS
2580  POKE 37647,SL%
2590 SL% = (V% / LS - SL%) * LS
2600  POKE 37646,SL%
2610  RETURN
2620  REM
2630  REM
2640 X =  PEEK (37636) + LS *  PEEK (37637)
2650 Y =  PEEK (37638) + LS *  PEEK (37639)
2660 X = (X - XF) * XS * SC
2670 Y = (YF - Y) * XS
2680  RETURN
60000  HOME
60010  PRINT "WAIT JUST A SECOND"
60020  PRINT D$;"RUN DIGITIZE,D1"
60030  END
```

```
10  REM   PLOT ARC FILES ROUTINE 9/21/82
20  HIMEM: 16383
30 D$ =  CHR$ (4)
40 LS = 256
50 SC = 1.11111111
60  PRINT D$;"BLOAD MPTS.BIN,D1"
70  PRINT D$;"BLOAD ONERR.BIN,D1"
80  HGR2 : HCOLOR= 3:ND% = 0
90  POKE 49236,0: POKE 49233,0
100  HOME
110  INPUT "ENTER ARC FILENAME TO PLOT: ";FI$
120  INPUT "ENTER 1ST ARC TO PLOT: ";N%
130  INPUT "ENTER NUMBER OF ARCS TO PLOT (0=ALL): ";NP%
140 ND% = 0
150  REM
160  POKE 49237,0: POKE 49232,0
170  ONERR  GOTO 520
180  PRINT D$;"BLOAD ";FI$;"/ARC"; STR$ (N%);",D2"
190  POKE 216,0
200  IF ND% > 0 THEN 400
210 XO =  PEEK (24588) + LS *  PEEK (24589)
220 YO =  PEEK (24590) + LS *  PEEK (24591)
230 X =  PEEK (24596) + LS *  PEEK (24597)
240 Y =  PEEK (24598) + LS *  PEEK (24599)
250 XI = X - XO
260 YI = YO - Y
270 X2 = 251.1 / XI
280 Y2 = 191 / YI
290  IF Y2 < X2 THEN X2 = Y2
300 XC = X2 * SC
310 XI = XI * XC
320 YI = YI * X2
330  HPLOT 0,0 TO XI,0 TO XI,YI TO 0,YI TO 0,0
340 V% = 4 / X2 + .5
350 SL% = V% / LS
360  POKE 37645,SL%
370 SL% = (V% / LS - SL%) * LS
380  POKE 37644,SL%
390  REM
400 ND% = ND% + 1
410  CALL 38075
420  GOSUB 950
430  HPLOT X,Y
440  CALL 38132
450  GOSUB 950
460  HPLOT  TO X,Y
470  IF  PEEK (37648) <  > 0 THEN 440
480 N% = N% + 1
490  IF ND% = NP% THEN 90
500  GOTO 160
510  REM
520  REM
530  POKE 49236,0: POKE 49233,0
540  POKE 216,0
```

```
550  CALL 38390
560  HOME
570 EC =  PEEK (222)
580  IF EC = 6 OR EC = 9 THEN 690
590  IF EC = 8 THEN 650
600  PRINT "DISK ERROR - ";EC
610  PRINT "CORRECT PROBLEM-THEN HIT ANY KEY";
620  GET C$
630  PRINT
640  GOTO 160
650  INVERSE : PRINT FI$;"/ARC";N%;" IS UNREADABLE": NORMAL
660  PRINT "        1-RETRY READING"
670  GOTO 720
680  REM
690  REM
700  INVERSE : PRINT FI$;"/ARC";N%;" NOT FOUND": NORMAL
710  PRINT "        1-CONTINUE WITH NEXT DISK"
720  PRINT "        2-CONTINUE WITH ";FI$;"/ARC";N% + 1
730  PRINT "        3-SELECT NEXT FILE"
740  PRINT "        4-LIST ALL FILES"
750  PRINT "        5-DISPLAY PICTURE"
760  PRINT "        9-QUIT"
770  PRINT "SELECT:";
780  GET SL$
790  PRINT SL$
800  ON  VAL (SL$) GOTO 160,480,90,880,830,690,690,690,60000
810  GOTO 690
820  REM
830  POKE 49237,0: POKE 49232,0
840  FOR I = 1 TO 1000: NEXT I
850  POKE 49236,0: POKE 49233,0
860  GOTO 560
870  REM
880  HOME
890  PRINT D$;"CATALOG,D2"
900  INVERSE : PRINT "HIT ANY KEY TO CONTINUE": NORMAL
910  GET SL$
920  GOTO 560
930  REM
940  REM
950 X =  PEEK (37636) + LS *  PEEK (37637)
960 Y =  PEEK (37638) + LS *  PEEK (37639)
970 X = (X - XO) * XC
980 Y = (YO - Y) * X2
990  IF X < 0 THEN X = 0
1000  IF X > 279 THEN X = 279
1010  IF Y < 0 THEN Y = 0
1020  IF Y > 191 THEN Y = 191
1030  RETURN
60000  HOME : PRINT "WAIT JUST A SECOND"
60010  PRINT D$;"RUN DIGITIZE,D1"
60020  END
```

```
10  HIMEM: 37823
20 D$ =  CHR$ (4)
30  PRINT D$;"BLOAD TERM.BIN"
40  PRINT D$;"BLOAD SEND.BIN"
50  INPUT "ENTER FILENAME TO SEND: ";FI$
60  HOME : INVERSE : PRINT "EXECUTE APPLER TO RECEIVE ";FI$: NORMAL
70  POKE 34,1
80  CALL 37888
90  POKE 34,0
100  HOME
110  PRINT D$;"OPEN ";FI$
120  PRINT D$;"READ ";FI$
130  CALL 37826
140  ONERR  GOTO 360
150 NC% = 1
160  FOR TM = 1 TO 100: NEXT TM
170  GET CH$: PRINT
180  PRINT CH$;
190 CH% =  ASC (CH$)
200  IF CH% > 31 THEN 230
210  IF CH% > 6 AND CH% < 14 THEN 230
220  GOTO 170
230  POKE 37824,CH%
240  CALL 37837
250  IF CH% = 13 THEN 150
260 NC% = NC% + 1
270  IF NC% < 131 THEN 170
280  POKE 37824,13
290  CALL 37837
300  FOR NS = 1 TO 5
310  POKE 37824,32
320  CALL 37837
330  NEXT NS
340 NC% = 6
350  GOTO 170
360  POKE 37824,16
370  FOR TM = 1 TO 100: NEXT TM
380  CALL 37837
390  FOR TM = 1 TO 100: NEXT TM
400  POKE 37824,57
410  CALL 37837
420  POKE 37824,13
430  CALL 37837
440  PRINT D$;"CLOSE ";FI$
450  HOME : INVERSE : PRINT "AFTER LOGGING OFF TYPE CTRL Z": NORMAL
460  POKE 34,1
470  CALL 37888
480  POKE 34,0
490  HOME
500  END
```

```
10  REM   SEND ARC AND LIN FILE ROUTINE 9/21/82
20  HIMEM: 37800
30 D$ =  CHR$ (4)
40 LS = 256
50  PRINT D$;"BLOAD TERM.BIN,D1"
60  PRINT D$;"BLOAD SNDARC.BIN,D1"
70  PRINT D$;"BLOAD ONERR.BIN,D1"
80  HOME
90  INVERSE : PRINT "LOGON AND EXECUTE DIGRECV": NORMAL
100  POKE 34,1
110  CALL 37888
120  POKE 34,0
130  HOME
140  INVERSE : PRINT "TRANSFER DIGITIZED DATA TO HOST": NORMAL
150  PRINT
160  PRINT "        LIN-LINEAR FEATURES"
170  PRINT "        ARC-ARCS"
180  INPUT "ENTER: ";TY$
190  IF TY$ <  > "LIN" AND TY$ <  > "ARC" GOTO 130
200  PRINT
210  INPUT "ENTER FILENAME TO SEND: ";FI$
220  IF FI$ <  > "" THEN 250
230  POKE 37,5
240  GOTO 200
250  INPUT "ENTER 1ST ARC TO SEND: ";N%
260  ONERR  GOTO 480
270  PRINT D$;"BLOAD ";FI$;"/";TY$; STR$ (N%);",D2"
280  POKE 216,0
290 AN% =  PEEK (24576) + LS *  PEEK (24577)
300 NP% =  PEEK (24578) + LS *  PEEK (24579)
310 AL% =  PEEK (24580) + LS *  PEEK (24581)
320 AR% =  PEEK (24582) + LS *  PEEK (24583)
330  HOME
340  INVERSE : PRINT "FILE",TY$,"# PTS": NORMAL
350  PRINT FI$;"/";TY$;N%,AN%,NP%
360  IF TY$ = "LIN" GOTO 400
370  PRINT ,"AREA RIGHT = ";AR%
380  PRINT ,"AREA LEFT  = ";AL%
390  GOTO 410
400  PRINT  TAB( 13);"LINEAR FEATURE = ";AR%
410  PRINT "     POINTS SENT";
420  POKE 36,00
430  CALL 37376
440 N% = N% + 1
450  PRINT
460  GOTO 260
470  REM
480  REM
490  POKE 216,0
500  CALL 38390
510  HOME
520 EC =  PEEK (222)
530  IF EC = 6 OR EC = 9 THEN 640
540  IF EC = 8 THEN 600
```

```
550  PRINT "DISK ERROR - ";EC
560  PRINT "CORRECT PROBLEM-THEN HIT ANY KEY";
570  GET C$
580  PRINT
590  GOTO 260
600  INVERSE : PRINT FI$;"/";TY$;N%;" IS UNREADABLE": NORMAL
610  PRINT "        1-RETRY READING"
620  GOTO 670
630  REM
640  HOME
650  INVERSE : PRINT FI$;"/";TY$;N%;" NOT FOUND": NORMAL
660  PRINT "        1-CONTINUE WITH NEXT DISK"
670  PRINT "        2-CONTINUE WITH ";FI$;"/";TY$;N% + 1
680  PRINT "        3-SELECT NEXT FILE"
690  PRINT "        4-LIST ALL FILES"
700  PRINT "        9-QUIT"
710  PRINT "SELECT:";
720  GET SL$
730  PRINT SL$
740  ON  VAL (SL$) GOTO 260,440,130,770,530,530,530,530,830
750  GOTO 530
760  REM
770  HOME
780  PRINT D$;"CATALOG,D2"
790  INVERSE : PRINT "HIT ANY KEY TO CONTINUE": NORMAL
800  GET SL$
810  GOTO 530
820  REM
830  CALL 37712
840  HOME
850  INVERSE : PRINT "TYPE CTRL Z AFTER LOGGING OFF": NORMAL
860  POKE 34,1
870  CALL 37888
880  POKE 34,0
890  HOME : PRINT "WAIT JUST A SECOND"
900  PRINT D$;"RUN DIGITIZE,D1"
910  END
```

```
10  REM   SEND CHECK POINT FILE ROUTINE 9/21/82
20  HIMEM: 37823
30 D$ =  CHR$ (4)
40  PRINT D$;"BLOAD TERM.BIN,D1"
50  PRINT D$;"BLOAD SEND.BIN,D1"
60  PRINT D$;"BLOAD ONERR.BIN,D1"
70  HOME
80  INVERSE : PRINT "LOGON AND EXECUTE DIGRECV": NORMAL
90  POKE 34,1
100  CALL 37888
110  POKE 34,0
120  HOME
130  GOTO 680
140  REM
150  REM
160  HOME
170  ONERR  GOTO 640
180  PRINT D$;"UNLOCK ";FI$;",D2"
190  PRINT D$;"OPEN ";FI$;",D2"
200  PRINT D$;"READ ";FI$
210  POKE 216,0
220  HOME : PRINT "SENDING ";: INVERSE : PRINT FI$;: NORMAL : PRINT " TO HOST"
230  CALL 37826
240  POKE 37824,16
250  CALL 37837
260  POKE 37824, ASC ("2")
270  CALL 37837
280  POKE 37824,13
290  CALL 37837
300  GET CH$
310 CH% =  ASC (CH$)
320  POKE 37824,CH%
330  CALL 37838
340  IF CH% <  > 13 THEN 300
350 NR% = 1
360  REM
370  REM
380  CALL 37826
390  POKE 37824,16
400  CALL 37837
410  POKE 37824, ASC ("3")
420  CALL 37837
430  POKE 37824,13
440  CALL 37837
450  ONERR  GOTO 590
460  GET CH$
470  POKE 216,0
480 CH% =  ASC (CH$)
490  IF CH% = 32 THEN CH% = 44
500  POKE 37824,CH%
510  CALL 37837
520  IF CH% <  > 13 THEN 450
530  POKE 36,0
540  PRINT " CHECK PT ";NR%;" SENT";
```

```
550 NR% = NR% + 1
560  GOTO 380
570  REM
580  REM
590  PRINT D$;"CLOSE ";FI$
600  HOME
610  GOTO 660
620  REM
630  REM
640  HOME
650  INVERSE : PRINT FI$;" DOES NOT EXIST": NORMAL
660  POKE 216,0
670  CALL 38390
680  PRINT "TYPE CATALOG, QUIT, OR CHECK PT FILENAME"
690  INPUT "ENTER: ";FI$
700  IF FI$ <  > "" THEN 730
710  HOME
720  GOTO 680
730  IF FI$ <  > "CATALOG" THEN 800
740  HOME
750  PRINT D$;"CATALOG,D2"
760  INVERSE : PRINT "HIT ANY KEY TO CONTINUE": NORMAL
770  GET C$
780  HOME
790  GOTO 680
800  IF FI$ = "QUIT" THEN 820
810  GOTO 160
820  POKE 37824,16
830  CALL 37837
840  POKE 37824, ASC ("9")
850  CALL 37837
860  POKE 37824,13
870  CALL 37837
880  HOME
890  INVERSE : PRINT "AFTER LOGGING OFF TYPE CTRL Z": NORMAL
900  POKE 34,1
910  CALL 37888
920  POKE 34,0
930  HOME : PRINT "WAIT JUST A SECOND"
940  PRINT D$;"RUN DIGITIZE,D1"
950  END
```

```
10  REM   SEND CENTROID FILE ROUTINE 9/21/82
20  HIMEM: 37823
30 D$ =  CHR$ (4)
40  PRINT D$;"BLOAD TERM.BIN,D1"
50  PRINT D$;"BLOAD SEND.BIN,D1"
60  PRINT D$;"BLOAD ONERR.BIN,D1"
70  HOME
80  INVERSE : PRINT "LOGON AND EXECUTE DIGRECV": NORMAL
90  POKE 34,1
100  CALL 37888
110  POKE 34,0
120  HOME
130  GOTO 690
140  REM
150  REM
160  HOME
170  ONERR  GOTO 650
180  PRINT D$;"UNLOCK ",FI$;",D2"
190  PRINT D$;"OPEN ";FI$;",D2"
200  PRINT D$;"READ ";FI$
210  POKE 216,0
220  HOME : PRINT "SENDING ";: INVERSE : PRINT FI$;: NORMAL : PRINT " TO HOST"
230 NR% = 1
240  REM
250  REM
260  CALL 37826
270  POKE 37824,16
280  CALL 37837
290  POKE 37824, ASC ("4")
300  CALL 37837
310  POKE 37824,13
320  CALL 37837
330  ONERR  GOTO 600
340  GET CH$
350  POKE 216,0
360 CH% =  ASC (CH$)
370  IF CH% = 32 THEN CH% = 44
380  IF CH% <  > 13 THEN 510
390  REM
400  POKE 37824,44
410  CALL 37837
420 ST$ =  STR$ (NR%)
430 L =  LEN (ST$)
440  FOR N = 1 TO L
450 CN$ =  MID$ (ST$,N,1)
460 CN% =  ASC (CN$)
470  POKE 37824,CN%
480  CALL 37837
490  NEXT N
500  REM
510  POKE 37824,CH%
520  CALL 37837
530  IF CH% <  > 13 THEN 330
540  POKE 36,0
```

```
550  PRINT " CENTROID ";NR%;" SENT";
560 NR% = NR% + 1
570  GOTO 260
580  REM
590  REM
600  PRINT D$;"CLOSE ";FI$
610  HOME
620  GOTO 670
630  REM
640  REM
650  HOME
660  INVERSE : PRINT FI$;" DOES NOT EXIST": NORMAL
670  POKE 216,0
680  CALL 38390
690  PRINT "TYPE CATALOG, QUIT, OR CENTROID FILENAME"
700  INPUT "ENTER: ";FI$
710  IF FI$ <  > "" THEN 740
720  HOME
730  GOTO 690
740  IF FI$ <  > "CATALOG" THEN 810
750  HOME
760  PRINT D$;"CATALOG,D2"
770  INVERSE : PRINT "HIT ANY KEY TO CONTINUE": NORMAL
780  GET C$
790  HOME
800  GOTO 690
810  IF FI$ = "QUIT" THEN 830
820  GOTO 160
830  POKE 37824,16
840  CALL 37837
850  POKE 37824, ASC ("9")
860  CALL 37837
870  POKE 37824,13
880  CALL 37837
890  HOME
900  INVERSE : PRINT "AFTER LOGGING OFF TYPE CTRL Z": NORMAL
910  POKE 34,1
920  CALL 37888
930  POKE 34,0
940  HOME : PRINT "WAIT JUST A SECOND"
950  PRINT D$;"RUN DIGITIZE,D1"
960  END
```

```
                0010  ; SEND B TYPE ARC FILES TO
                0020  ; SERIAL I/O PORT 9/14/82
                0030        ORG 9200
                0040  ARCN    EQU 6000      ;ARC NUMBER
                0050  IPTS    EQU 6002      ;# OF PTS IN BUF
                0060  IPTSP1  EQU 6003
                0070  ALEFT   EQU 6004      ;LEFT AREA CODE
                0080  ARITE   EQU 6006      ;RIGHT AREA CODE
                0090  TLX     EQU 600C      ;TOP LEFT X
                0100  TLY     EQU 600E      ;TOP LEFT Y
                0110  TRX     EQU 6010      ;TOP RIGHT X
                0120  TRY     EQU 6012      ;TOP RIGHT Y
                0130  BUF     EQU 6080      ;X,Y POINTS
                0140  ADR     EQU 000A      ;PAGE 0 PTR
                0150  CNTRL   EQU C0A4      ;CNTRL, SLOT 2
                0160  IO      EQU C0A5      ;IO PORT,SLOT 2
                0170  WAIT    EQU FCA8      ;DELAY ROUTINE
                0180  COUT1   EQU FDF0      ;PRINT CHAR
                0190  PRNTAX  EQU F941      ;PRINT A,X
                0200  ASMTRM  EQU 9400      ;TERMINAL ROUTIN
                0210  ;
9200  A903      0220  SNDARC  LDA #$03      ;INIT SERIAL I/O
9202  8DA4C0    0230        STA CNTRL
9205  A915      0240        LDA #$15
9207  8DA4C0    0250        STA CNTRL
920A  A900      0260        LDA #$00      ;SET ADDR
920C  8D9A92    0270        STA NPTS      ;INIT # PTS SENT
920F  8D9B92    0280        STA NPTS+1
9212  850A      0290        STA ADR       ;OF ARC
9214  A960      0300        LDA #$60      ;HEADER
9216  850B      0310        STA ADR+1
                0320  ;
9218  A210      0330  DLE0    LDX #$10      ;GET DLE
921A  20D692    0340        JSR SCHAR     ;SEND CHAR
921D  20E192    0350        JSR CKECHO    ;GET ECHO
9220  B0F6      0360        BCS DLE0      ;IF NAK-RESEND
9222  A230      0370        LDX #$30      ;GET 0
9224  20D692    0380        JSR SCHAR     ;SEND CHAR
9227  20E192    0390        JSR CKECHO    ;GET ECHO
922A  B0EC      0400        BCS DLE0      ;IF NAK-RESEND
922C  201493    0410        JSR SNDCR     ;SEND CR
                0420  ;
922F  A013      0430  SHEAD   LDY #$13      ;SET BYTE PTR
9231  20B092    0440  NHBYT   JSR TLNIB     ;SEND LEFT NIB
9234  20E192    0450        JSR CKECHO    ;GET ECHO
9237  B0F6      0460        BCS SHEAD     ;IF NAK-RESEND
9239  209C92    0470        JSR TRNIB     ;SEND RIGHT NIB
923C  20E192    0480        JSR CKECHO    ;GET ECHO
923F  B0EE      0490        BCS SHEAD     ;IF NAK-RESEND
9241  88        0500        DEY           ;DEC BYTE PTR
9242  10ED      0510        BPL NHBYT     ;DO NEXT BYTE
9244  201493    0520        JSR SNDCR     ;SEND <CR>
                0530  ;
```

```
9247  A900     0540          LDA #$00      ;INIT # PTS SENT
9249  8D9A92   0550          STA NPTS
924C  8D9B92   0560          STA NPTS+1
924F  A980     0570          LDA #$80      ;SET ADDR
9251  850A     0580          STA ADR       ;OF X,Y
9253  A960     0590          LDA #$60      ;POINTS
9255  850B     0600          STA ADR+1
9257  A000     0610  SPTS    LDY #$00      ;SET BYTE PTR
9259  20B092   0620  NPBYT   JSR TLNIB     ;SEND LEFT NIB
925C  20E192   0630          JSR CKECHO    ;GET ECHO
925F  B0F6     0640          BCS SPTS      ;IF NAK-RESEND
9261  209C92   0650          JSR TRNIB     ;SEND RIGHT NIB
9264  20E192   0660          JSR CKECHO    ;GET ECHO
9267  B0EE     0670          BCS SPTS      ;IF NAK-RESEND
9269  C8       0680          INY           ;INC BYTE PTR
926A  C029     0690          CPY #$29      ;CHECK # BYTES
926C  30EB     0700          BMI NPBYT     ;DO NEXT BYTE
926E  201493   0710          JSR SNDCR     ;SEND <CR>
9271  A50A     0720          LDA ADR       ;INCR BASE
9273  18       0730          CLC           ;ADDRESS
9274  6914     0740          ADC #$14      ;PTR TO
9276  850A     0750          STA ADR       ;X,Y PTS
9278  9002     0760          BCC NINC      ;BY # BYTES
927A  E60B     0770          INC ADR+1
927C  AD0260   0780  NINC    LDA IPTS      ;COUNT # OF
927F  38       0790          SEC           ;X,Y POINTS
9280  E905     0800          SBC #$05      ;SENT
9282  8D0260   0810          STA IPTS
9285  AD0360   0820          LDA IPTS+1
9288  E900     0830          SBC #$00
928A  8D0360   0840          STA IPTSP1
928D  3007     0850          BMI END       ;IF ALL PTS
928F  D0C6     0860          BNE SPTS      ;DO NEXT PTS
9291  AD0260   0870          LDA IPTS
9294  D0C1     0880          BNE SPTS      ;DO NEXT PTS
9296  201493   0890  END     JSR SNDCR     ;SEND <CR>
9299  60       0900          RTS
929A  00       0910  NPTS    DFD 00        ;# PTS SENT
929B  00       0920          DFD 00
               0930  ;
               0940  ;  SUBROUTINES
               0950  ;
929C  ADA4C0   0960  TRNIB   LDA CNTRL     ;GET STATUS
929F  2902     0970          AND #$02
92A1  F0F9     0980          BEQ TRNIB     ;WAIT FOR RDY
92A3  B10A     0990          LDA (ADR),Y
92A5  290F     1000          AND #$0F      ;GET RIGHT NIBBL
92A7  AA       1010          TAX
92A8  BDC692   1020          LDA TAB,X     ;GET ASCII CHAR
92AB  AA       1030          TAX           ;SAVE FOR ECHO C
92AC  8DA5C0   1040          STA IO        ;SEND CHAR
92AF  60       1050          RTS
               1060  ;
```

```
92B0  ADA4C0  1070  TLNIB  LDA CNTRL    ;GET STATUS
92B3  2902    1080         AND #$02
92B5  F0F9    1090         BEQ TLNIB    ;WAIT FOR RDY
92B7  B10A    1100         LDA (ADR),Y
92B9  4A      1110         LSR          ;GET LEFT
92BA  4A      1120         LSR          ;NIBBLE
92BB  4A      1130         LSR
92BC  4A      1140         LSR
92BD  AA      1150         TAX
92BE  BDC692  1160         LDA TAB,X    ;GET ASCII CHAR
92C1  AA      1170         TAX          ;SAVE FOR ECHO C
92C2  8DA5C0  1180         STA IO       ;SEND CHAR
92C5  60      1190         RTS
              1200  ;
92C6  30      1210  TAB    DFD 30       ;ASCII 0
92C7  31      1220         DFD 31       ;1
92C8  32      1230         DFD 32       ;2
92C9  33      1240         DFD 33       ;3
92CA  34      1250         DFD 34       ;4
92CB  35      1260         DFD 35       ;5
92CC  36      1270         DFD 36       ;6
92CD  37      1280         DFD 37       ;7
92CE  38      1290         DFD 38       ;8
92CF  39      1300         DFD 39       ;9
92D0  41      1310         DFD 41       ;A
92D1  42      1320         DFD 42       ;B
92D2  43      1330         DFD 43       ;C
92D3  44      1340         DFD 44       ;D
92D4  45      1350         DFD 45       ;E
92D5  46      1360         DFD 46       ;F
              1370  ;
92D6  ADA4C0  1380  SCHAR  LDA CNTRL    ;GET STATUS
92D9  2902    1390         AND #$02     ;CHECK STATUS
92DB  F0F9    1400         BEQ SCHAR    ;WAIT FOR RDY
92DD  8EA5C0  1410         STX IO       ;SEND CHAR
92E0  60      1420         RTS
              1430  ;
92E1  ADA4C0  1440  CKECHO LDA CNTRL    ;GET STATUS
92E4  4A      1450         LSR
92E5  90FA    1460         BCC CKECHO   ;WAIT FOR RDY
92E7  ADA5C0  1470         LDA IO       ;GET ECHO
92EA  297F    1480         AND #$7F     ;REMOVE PARITY
92EC  8D1393  1490         STA ECHO     ;COMPARE ECHO
92EF  EC1393  1500         CPX ECHO     ;TO CHAR SENT
92F2  F01D    1510         BEQ ACK      ;SAME-OK
92F4  ADA4C0  1520  WNAK   LDA CNTRL    ;NOT SAME
92F7  2902    1530         AND #$02     ;GET STATUS
92F9  F0F9    1540         BEQ WNAK     ;WAIT FOR RDY
92FB  A915    1550         LDA #$15     ;GET CTRL-U
92FD  8DA5C0  1560         STA IO       ;SEND ASCII NAK
9300  ADA4C0  1570  WKAN   LDA CNTRL    ;GET STATUS
9303  4A      1580         LSR
9304  90FA    1590         BCC WKAN     ;WAIT FOR RDY
9306  ADA5C0  1600         LDA IO       ;GET ECHO
9309  297F    1610         AND #$7F     ;REMOVE PARITY
```

```
930B  C915     1620          CMP #$15      ;CHECK FOR NAK
930D  D0E5     1630          BNE WNAK      ;NO-RENAK
930F  38       1640          SEC           ;NAK-SET FLAG
9310  60       1650          RTS
9311  18       1660  ACK     CLC           ;OK-CLEAR FLAG
9312  60       1670          RTS
9313  00       1680  ECHO    DFD 00
               1690  ;
9314  ADA4C0   1700  SNDCR   LDA CNTRL     ;GET STATUS
9317  2902     1710          AND #$02
9319  F0F9     1720          BEQ SNDCR     ;WAIT FOR RDY
931B  A90D     1730          LDA #$0D
931D  8DA5C0   1740          STA IO        ;SEND <CR>
9320  ADA5C0   1750  WXON    LDA IO        ;GET CHAR
9323  AEA5C0   1760          LDX IO
9326  ACA5C0   1770          LDY IO
9329  297F     1780          AND #$7F      ;REMOVE PARITY
932B  C911     1790          CMP #$11      ;CHECK FOR XON
932D  D0F1     1800          BNE WXON
932F  F8       1810          SED           ;SET DEC MODE
9330  AD9A92   1820          LDA NPTS
9333  18       1830          CLC
9334  6905     1840          ADC #$05      ;INC # PTS SENT
9336  8D9A92   1850          STA NPTS
9339  AA       1860          TAX           ;PUT LSB IN X
933A  AD9B92   1870          LDA NPTS+1    ;GET MSB
933D  6900     1880          ADC #$00      ;DO DBL PREC INC
933F  8D9B92   1890          STA NPTS+1
9342  D8       1900          CLD           ;CLR DEC MODE
9343  2041F9   1910          JSR PRNTAX    ;PRINT # PTS SEN
9346  A900     1920          LDA #$00
9348  8524     1930          STA $24       ;SET CURSOR TOLE
934A  A980     1940          LDA #$80
934C  20A8FC   1950          JSR WAIT      ;GOTO APPLE WAIT
934F  60       1960          RTS
               1970  ;
9350  A903     1980  TERM    LDA #$03      ;INIT IO
9352  8DA4C0   1990          STA CNTRL
9355  A915     2000          LDA #$15
9357  8DA4C0   2010          STA CNTRL
935A  A210     2020          LDX #$10      ;GET DLE
935C  20D692   2030          JSR SCHAR     ;SEND CHAR
935F  20E192   2040          JSR CKECHO    ;GET ECHO
9362  B0EC     2050          BCS TERM      ;IF NAK-RESEND
9364  A239     2060          LDX #$39      ;GET 9
9366  20D692   2070          JSR SCHAR     ;SEND CHAR
9369  20E192   2080          JSR CKECHO    ;GET ECHO
936C  B0E2     2090          BCS TERM      ;IF NAK-RESEND
936E  201493   2100          JSR SNDCR     ;SEND <CR>
9371  60       2110  BOTTOM  RTS           ;GOTO BASIC
```

```
LABEL TABLE
ARCN   6000
IPTS   6002
IPTSP1 6003
ALEFT  6004
ARITE  6006
TLX    600C
TLY    600E
TRX    6010
TRY    6012
BUF    6080
ADR    000A
CNTRL  C0A4
IO     C0A5
WAIT   FCA8
COUT1  FDF0
PRNTAX F941
ASMTRM 9400
SNDARC 9200
DLE0   9218
SHEAD  922F
NHBYT  9231
SPTS   9257
NPBYT  9259
NINC   927C
END    9296
NPTS   929A
TRNIB  929C
TLNIB  92B0
TAB    92C6
SCHAR  92D6
CKECHO 92E1
WNAK   92F4
WKAN   9300
ACK    9311
ECHO   9313
SNDCR  9314
WXON   9320
TERM   9350
BOTTOM 9371
```

```
                0010  ; SEND A CHAR TO A SERIAL I/O PORT
                0020  ; ROUTINE 9/14/82
                0030  CNTRL  EQU C0A4      ;SLOT 2 CTRL
                0040  IO     EQU C0A5      ;SLOT 2 I/O
                0050  INPUT  EQU 93C0      ;SEND CHAR
                0060  OUTPUT EQU 93C1      ;RETURN CHAR
                0070         ORG 93C2
                0080  ;
                0090  ;  INITIALIZATION ROUTINE
                0100  ;
93C2  A903      0110  INIT   LDA #$03      ;MASTER CLEAR
93C4  8DA4C0    0120         STA CNTRL
93C7  A915      0130         LDA #$15      ;8 DATA,1STOP
93C9  8DA4C0    0140         STA CNTRL
93CC  60        0150         RTS
                0160  ;
                0170  ;  SEND ROUTINE
                0180  ;
93CD  ADA4C0    0190  SEND   LDA CNTRL     ;GET STATUS
93D0  2902      0200         AND #$02
93D2  F0F9      0210         BEQ SEND      ;WAIT FOR RDY
93D4  ADC093    0220         LDA INPUT     ;GET CHAR
93D7  8DA5C0    0230         STA IO        ;SEND CHAR
93DA  C90D      0240         CMP #$0D      ;<CR>?
93DC  F001      0250         BEQ WCR       ;CR-GET RESP
93DE  60        0260         RTS           ;NO CR-RETURN
93DF  ADA4C0    0270  WCR    LDA CNTRL     ;GET STATUS
93E2  4A        0280         LSR           ;STAT TO CARY
93E3  90FA      0290         BCC WCR       ;WAIT
93E5  ADA5C0    0300         LDA IO        ;GET CHAR
93E8  297F      0310         AND #$7F      ;REMOVE MSB
93EA  C911      0320         CMP #$11      ;CHECK FOR XON
93EC  D0F1      0330         BNE WCR       ;IF NOT-IGNORE
93EE  8DC193    0340         STA OUTPUT    ;SAVE CHAR
93F1  60        0350         RTS           ;RETURN


LABEL TABLE
CNTRL  C0A4
IO     C0A5
INPUT  93C0
OUTPUT 93C1
INIT   93C2
SEND   93CD
WCR    93DF
```

```
                0010  ; GET A SINGLE POINT FROM DIGITIZER
                0020  ; ROUTINE 9/14/82
                0030        ORG 91C0
                0040  CTRL  EQU C0B4      ;CTRL FOR SL#3
                0050  IO    EQU C0B5      ;IO PORT,SLOT#3
                0060  ;
91C0  00        0070  XVAL  DFD 00        ;X VALUE
91C1  00        0080  XVALP1 DFD 00
91C2  00        0090  YVAL  DFD 00        ;Y VALUE
91C3  00        0100  YVALP1 DFD 00
91C4  00        0110  BUT1  DFD 00        ;BUTTON 1 CODE
91C5  00        0120  BUT2  DFD 00        ;BUTTON 2 CODE
                0130  ;
91C6  A903      0140  DIGPT LDA #$03      ;SET SERIAL I/O
91C8  8DB4C0    0150        STA CTRL
91CB  A915      0160        LDA #$15
91CD  8DB4C0    0170        STA CTRL
                0180  ;
91D0  207D92    0190  SYNC  JSR GBYT      ;GET BUTTON CODE
91D3  8DC491    0200        STA BUT1
91D6  207D92    0210        JSR GBYT      ;GET BUTTON CODE
91D9  8DC591    0220        STA BUT2
                0230  ;
91DC  207D92    0240        JSR GBYT      ;GET X 1'S DIGIT
91DF  8DC091    0250        STA XVAL
                0260  ;
91E2  207D92    0270        JSR GBYT      ;GET X 10'S DIGI
91E5  A000      0280        LDY #$00
91E7  8CC191    0290        STY XVALP1    ;INIT MSBYTE
91EA  209292    0300        JSR X10       ;MULT X 10
91ED  20C592    0310        JSR XADD      ;ADD 10'S TO X
                0320  ;
91F0  207D92    0330        JSR GBYT      ;GET X 100'S DIG
91F3  A000      0340        LDY #$00
91F5  209292    0350        JSR X10       ;MULT X 100
91F8  209292    0360        JSR X10
91FB  20C592    0370        JSR XADD      ;ADD 100'S TO X
                0380  ;
91FE  207D92    0390        JSR GBYT      ;GET X 1K DIGIT
9201  A000      0400        LDY #$00
9203  209292    0410        JSR X10       ;MULT X 1000
9206  209292    0420        JSR X10
9209  209292    0430        JSR X10
920C  20C592    0440        JSR XADD      ;ADD 1K'S TO X
                0450  ;
920F  207D92    0460        JSR GBYT      ;GET X 10K DIGIT
9212  A000      0470        LDY #$00
9214  209292    0480        JSR X10       ;MULTI X 10K
9217  209292    0490        JSR X10
921A  209292    0500        JSR X10
921D  209292    0510        JSR X10
9220  20C592    0520        JSR XADD      ;ADD 10K'S TO X
```

```
                0530  ;
9223  207D92    0540        JSR GBYT      ;GET Y 1'S DIGIT
9226  8DC291    0550        STA YVAL
                0560  ;
9229  207D92    0570        JSR GBYT      ;GET Y 10'S DIGI
922C  A000      0580        LDY #$00
922E  8CC391    0590        STY YVALP1    ;INIT MSBYTE
9231  209292    0600        JSR X10       ;MULTI X 10
9234  20D492    0610        JSR YADD      ;ADD 10'S TO Y
                0620  ;
9237  207D92    0630        JSR GBYT      ;GET Y 100'S DIG
923A  A000      0640        LDY #$00
923C  209292    0650        JSR X10       ;MULTI X 100
923F  209292    0660        JSR X10
9242  20D492    0670        JSR YADD      ;ADD 100'S TO Y
                0680  ;
9245  207D92    0690        JSR GBYT      ;GET Y 1K DIGIT
9248  A000      0700        LDY #$00
924A  209292    0710        JSR X10       ;MULTI 1000
924D  209292    0720        JSR X10
9250  209292    0730        JSR X10
9253  20D492    0740        JSR YADD      ;ADD 1K'S TO Y
                0750  ;
9256  207D92    0760        JSR GBYT      ;GET Y 10K DIGIT
9259  A000      0770        LDY #$00
925B  209292    0780        JSR X10       ;MULTI X 10K
925E  209292    0790        JSR X10
9261  209292    0800        JSR X10
9264  209292    0810        JSR X10
9267  20D492    0820        JSR YADD      ;ADD 10K'S TO Y
926A  ADB4C0    0830  CRW   LDA CTRL     ;GET IO STATUS
926D  4A        0840        LSR           ;PUT IN CARRY
926E  90FA      0850        BCC CRW       ;WAIT FOR CR
9270  ADB5C0    0860        LDA IO        ;GET CHAR
9273  297F      0870        AND #$7F      ;REMOVE PARITY
9275  C90D      0880        CMP #$0D      ;CHECK FOR CR
9277  F003      0890        BEQ CROK      ;CR-DATA OK
9279  4CD091    0900        JMP SYNC      ;NOT CR-RESYNC
                0910  ;
927C  60        0920  CROK  RTS
                0930  ;
                0940  ;
                0950  ;  SUBROUTINES
                0960  ;
927D  ADB4C0    0970  GBYT  LDA CTRL      ;GET IO STATUS
9280  4A        0980        LSR           ;PUT IN CARRY
9281  90FA      0990        BCC GBYT      ;CHECK FOR CHAR
9283  ADB5C0    1000        LDA IO        ;GET CHAR
9286  290F      1010        AND #$0F      ;REMOVE ASCII
9288  C90D      1020        CMP #$0D      ;CHECK FOR CR
928A  D005      1030        BNE RGBYT
928C  68        1040        PLA           ;RESET STACK
928D  68        1050        PLA           ;PTR
928E  4CD091    1060        JMP SYNC      ;CR-RESYNC
9291  60        1070  RGBYT RTS           ;RETURN
```

```
                1080  ;
                1090  ;
9292  8CC392    1100  X10     STY X2P1      ;GENERAL DBL PRE
9295  0A        1110          ASL           ;X 10 ROUTINE
9296  8DC292    1120          STA X2        ;A=LSBYTE
9299  2EC392    1130          ROL X2P1      ;Y=MSBYTE
929C  ADC392    1140          LDA X2P1
929F  8DC492    1150          STA X20
92A2  ADC292    1160          LDA X2
92A5  0A        1170          ASL
92A6  2EC492    1180          ROL X20
92A9  0A        1190          ASL
92AA  2EC492    1200          ROL X20
92AD  18        1210          CLC
92AE  6DC292    1220          ADC X2
92B1  8DC292    1230          STA X2
92B4  ADC492    1240          LDA X20
92B7  6DC392    1250          ADC X2P1
92BA  8DC392    1260          STA X2P1
92BD  A8        1270          TAY
92BE  ADC292    1280          LDA X2
92C1  60        1290          RTS
                1300  ;
92C2  00        1310  X2      DFD 00
92C3  00        1320  X2P1    DFD 00
92C4  00        1330  X20     DFD 00
                1340  ;
                1350  ;
92C5  18        1360  XADD    CLC           ;DBL PREC ADD
92C6  6DC091    1370          ADC XVAL      ;A=LSBYTE
92C9  8DC091    1380          STA XVAL      ;Y=MSBYTE
92CC  98        1390          TYA
92CD  6DC191    1400          ADC XVALP1
92D0  8DC191    1410          STA XVALP1
92D3  60        1420          RTS
                1430  ;
                1440  ;
92D4  18        1450  YADD    CLC
92D5  6DC291    1460          ADC YVAL
92D8  8DC291    1470          STA YVAL
92DB  98        1480          TYA
92DC  6DC391    1490          ADC YVALP1
92DF  8DC391    1500          STA YVALP1
92E2  60        1510  END     RTS
```

```
LABEL TABLE
CTRL    C0B4
IO      C0B5
XVAL    91C0
XVALP1  91C1
YVAL    91C2
YVALP1  91C3
BUT1    91C4
BUT2    91C5
DIGPT   91C6
SYNC    91D0
CRW     926A
CROK    927C
GBYT    927D
RGBYT   9291
X10     9292
X2      92C2
X2P1    92C3
X20     92C4
XADD    92C5
YADD    92D4
END     92E2
```

```
                0010  ; GET MULTIPLE POINTS FROM DIGITIZER
                0020  ; ROUTINE 9/14/82
                0030        ORG 9300
                0040  DIGPT  EQU 91C6      ;X,Y ROUTINE
                0050  BUT1   EQU 91C4      ;BUTTON 1 CODE
                0060  XVAL   EQU 91C0      ;X LSBYTE
                0070  XVALP1 EQU 91C1      ;X MSBYTE
                0080  YVAL   EQU 91C2      ;Y LSBYTE
                0090  YVALP1 EQU 91C3      ;Y MSBYTE
                0100  BELL   EQU FBDD      ;.1 SEC TONE
                0110  CLICK  EQU C030      ;CLICK SPEAKER
                0120  ADR    EQU 000A      ;PAGE 0 PTR
                0130  ;
                0140  ;  HEADER DEFNS FOR ARC FILE
                0150  ;
                0160  ARCN   EQU 6000      ;ARC NUMBER
                0170  IPTS   EQU 6002      ;# OF PTS IN BUF
                0180  ALEFT  EQU 6004      ;LEFT AREA CODE
                0190  ARITE  EQU 6006      ;RIGHT AREA CODE
                0200  XMN    EQU 6008      ;X MIN OF ARC
                0210  XMX    EQU 6009      ;X MAX OF ARC
                0220  YMN    EQU 600A      ;Y MIN OF ARC
                0230  YMX    EQU 600B      ;Y MAX OF ARC
                0240  TLX    EQU 600C      ;TOP LEFT X
                0250  TLY    EQU 600E      ;TOP LEFT Y
                0260  TRX    EQU 6010      ;TOP RIGHT X
                0270  TRY    EQU 6012      ;TOP RIGHT Y
                0280  BRX    EQU 6014      ;BOTTOM RIGHT X
                0290  BRY    EQU 6016      ;BOTTOM RIGHT Y
                0300  BUF    EQU 6080      ;X,Y PT BUFFER
9300  8060      0310  BUFA   EQD BUF
                0320  ;
9302  00        0330  DPTS   DFD 00        ;PT DOWN CTR
9303  00        0340  DPTSP1 DFD 00
9304  00        0350  OLDX   DFD 00        ;OLD X LSBYTE
9305  00        0360  OLDXP1 DFD 00        ;OLD X MSBYTE
9306  00        0370  OLDY   DFD 00        ;OLD Y LSBYTE
9307  00        0380  OLDYP1 DFD 00        ;OLD Y MSBYTE
9308  00        0390  XDIF   DFD 00        ;DELTA X
9309  00        0400  XDIFP1 DFD 00
930A  00        0410  YDIF   DFD 00        ;DELTA Y
930B  00        0420  YDIFP1 DFD 00
930C  00        0430  SCAL   DFD 00        ;MIN MILS/PIXEL
930D  00        0440  SCALP1 DFD 00
930E  00        0450  BCAL   DFD 00        ;MAX MILS/PIXEL
930F  00        0460  BCALP1 DFD 00
9310  00        0470  EFLAG  DFD 00        ;END OF PTS FLAG
                0480  ;
```

```
9311  AD0093  0490  MPTS   LDA BUFA      ;SET BUF ADDR
9314  850A    0500         STA ADR
9316  AD0193  0510         LDA BUFA+1
9319  850B    0520         STA ADR+1
931B  A9CF    0530         LDA #$CF      ;SET MAX # PTS
931D  8D0293  0540         STA DPTS      ;TO 1999
9320  A907    0550         LDA #$07
9322  8D0393  0560         STA DPTS+1
9325  A900    0570         LDA #$00      ;INIT PT CT
9327  8D0260  0580         STA IPTS
932A  8D0360  0590         STA IPTS+1
932D  A9FF    0600         LDA #$FF      ;INIT OLD PT
932F  8D0593  0610         STA OLDXP1
9332  8D0793  0620         STA OLDYP1
9335  20C691  0630         JSR DIGPT     ;GET NEW PT
9338  205A94  0640         JSR SXYB      ;SAVE PT
933B  209194  0650         JSR IADR      ;INC ADR
              0660  ;
933E  20C691  0670  NEWPT  JSR DIGPT     ;GET NEW PT
9341  ADC491  0680         LDA BUT1      ;GET BUTTON CODE
9344  C902    0690         CMP #$02      ;CHECK CODE
9346  D003    0700         BNE NRED      ;IF RED-
9348  4CDC93  0710         JMP XMYM      ;STOP
              0720  ;
934B  ADC091  0730  NRED   LDA XVAL      ;DBL PREC X
934E  38      0740         SEC           ;DIFF CALC
934F  ED0493  0750         SBC OLDX      ;CALC LSBYTE DIF
9352  8D0893  0760         STA XDIF
9355  ADC191  0770         LDA XVALP1
9358  ED0593  0780         SBC OLDXP1    ;CALC MSBYTE DIF
935B  1011    0790         BPL CKXD
935D  8D0993  0800         STA XDIFP1    ;NEG-TAKE DBL
9360  A900    0810         LDA #$00      ;PREC ABS
9362  38      0820         SEC
9363  ED0893  0830         SBC XDIF
9366  8D0893  0840         STA XDIF
9369  A900    0850         LDA #$00
936B  ED0993  0860         SBC XDIFP1
936E  8D0993  0870  CKXD   STA XDIFP1    ;SAVE X MSB
              0880  ;
9371  ADC291  0890         LDA YVAL      ;DBL PREC Y
9374  38      0900         SEC           ;DIFF CALC
9375  ED0693  0910         SBC OLDY      ;CALC LSBYTE DIF
9378  8D0A93  0920         STA YDIF
937B  ADC391  0930         LDA YVALP1
937E  ED0793  0940         SBC OLDYP1    ;CALC MSBYTE DIF
9381  1011    0950         BPL CKYD
9383  8D0B93  0960         STA YDIFP1    ;NEG-TAKE DBL
9386  A900    0970         LDA #$00      ;PREC ABS
9388  38      0980         SEC
9389  ED0A93  0990         SBC YDIF
938C  8D0A93  1000         STA YDIF
938F  A900    1010         LDA #$00
9391  ED0B93  1020         SBC YDIFP1
9394  8D0B93  1030  CKYD   STA YDIFP1    ;SAVE Y MSB
              1040  ;
```

```
9397  AD0A93  1050  CKL1   LDA YDIF      ;CALC DBL PREC
939A  18      1060         CLC           ;VECTOR LENGTH
939B  6D0893  1070         ADC XDIF      ;L1=DX+DY
939E  8D0893  1080         STA XDIF
93A1  AD0B93  1090         LDA YDIFP1
93A4  6D0993  1100         ADC XDIFP1
93A7  8D0993  1110         STA XDIFP1
              1120  ;
93AA  AD0893  1130         LDA XDIF      ;CHECK FOR MAX
93AD  38      1140         SEC           ;CALC DBL PREC
93AE  ED0E93  1150         SBC BCAL      ;DIFF
93B1  AD0993  1160         LDA XDIFP1    ;DIFF=L1-BCAL
93B4  ED0F93  1170         SBC BCALP1
93B7  B012    1180         BCS BVEC      ;IF>BCAL,BIG VEC
              1190  ;
93B9  AD0893  1200         LDA XDIF      ;CHECK FOR MIN
93BC  38      1210         SEC           ;CALC DBL PREC
93BD  ED0C93  1220         SBC SCAL      ;DIFF
93C0  AD0993  1230         LDA XDIFP1    ;DIFF=L1-SCAL
93C3  ED0D93  1240         SBC SCALP1
93C6  B006    1250         BCS SVEC      ;SCAL<L1<BCAL
93C8  4C3E93  1260         JMP NEWPT     ;L1<SCAL,IGNORE
93CB  20DDFB  1270  BVEC   JSR BELL      ;BIG VECTOR-BELL
93CE  205A94  1280  SVEC   JSR SXYB      ;SAVE PT
93D1  209194  1290         JSR IADR      ;INC ADR
93D4  209D94  1300         JSR DECP      ;COUNT MAX PTS
93D7  B003    1310         BCS XMYM      ;QUIT IF C SET
93D9  4C3E93  1320         JMP NEWPT     ;ELSE-GET NXT PT
              1330  ;
              1340  ;  FIND X MIN,MAX AND Y MIN,MAX
              1350  ;
93DC  AD0260  1360  XMYM   LDA IPTS      ;SET # OF PTS
93DF  8D0293  1370         STA DPTS
93E2  AD0360  1380         LDA IPTS+1
93E5  8D0393  1390         STA DPTSP1
93E8  C900    1400         CMP #$00      ;CHECK FOR 0 PTS
93EA  D007    1410         BNE NONZ
93EC  AD0260  1420         LDA IPTS      ;CHECK LSBYTE
93EF  C900    1430         CMP #$00
93F1  F066    1440         BEQ RET       ;BOTH BYTES 0
93F3  AD0093  1450  NONZ   LDA BUFA      ;SET BUFFER ADDR
93F6  850A    1460         STA ADR
93F8  AD0193  1470         LDA BUFA+1
93FB  850B    1480         STA ADR+1
              1490  ;
93FD  A000    1500         LDY #$00      ;SET PTR TO MSBY
93FF  B10A    1510         LDA (ADR),Y
9401  4A      1520         LSR           ;MAKE LE 127
9402  8D0860  1530         STA XMN       ;INIT X MIN
9405  6901    1540         ADC #$01
9407  8D0960  1550         STA XMX       ;INIT X MAX
940A  A002    1560         LDY #$02      ;SET PTR TO MSBY
940C  B10A    1570         LDA (ADR),Y
```

```
940E  4A       1580          LSR             ;MAKE LE 127
940F  8D0A60   1590          STA YMN         ;INIT Y MIN
9412  6901     1600          ADC #$01
9414  8D0B60   1610          STA YMX         ;INIT Y MAX
9417  209D94   1620          JSR DECP        ;COUNT 1 PT
941A  B03D     1630          BCS RET         ;CHECK FOR THRU
941C  209194   1640          JSR IADR        ;INC ADR BY 4
               1650  ;
941F  A000     1660  FMNMX   LDY #$00        ;SET X PTR
9421  B10A     1670          LDA (ADR),Y
9423  4A       1680          LSR             ;MAKE LE 127
9424  CD0860   1690          CMP XMN         ;COMPARE TO X MI
9427  1005     1700          BPL CXMX
9429  8D0860   1710          STA XMN         ;NEW X MIN
942C  300A     1720          BMI CYMN        ;SKIP X MAX CHK
942E  CD0960   1730  CXMX    CMP XMX         ;COMPARE TO X MA
9431  3005     1740          BMI CYMN
9433  6901     1750          ADC #$01
9435  8D0960   1760          STA XMX         ;NEW X MAX
9438  A002     1770  CYMN    LDY #$02        ;SET Y PTR
943A  B10A     1780          LDA (ADR),Y
943C  4A       1790          LSR             ;MAKE LE 127
943D  CD0A60   1800          CMP YMN         ;COMPARE TO Y MI
9440  1005     1810          BPL CYMX
9442  8D0A60   1820          STA YMN         ;NEW Y MIN
9445  300A     1830          BMI NMM         ;SKIP Y MAX CHK
9447  CD0B60   1840  CYMX    CMP YMX         ;COMPARE TO Y MA
944A  3005     1850          BMI NMM
944C  6901     1860          ADC #$01
944E  8D0B60   1870          STA YMX         ;NEW Y MAX
9451  209194   1880  NMM     JSR IADR        ;INC ADR BY 4
9454  209D94   1890          JSR DECP        ;COUNT PTS
9457  90C6     1900          BCC FMNMX       ;DO NEXT PT
9459  60       1910  RET     RTS             ;RETURN
               1920  ;
               1930  ;  SUBROUTINES
               1940  ;
945A  A000     1950  SXYB    LDY #$00        ;SET BUF PTR
945C  ADC191   1960          LDA XVALP1      ;GET X MSBYTE
945F  8D0593   1970          STA OLDXP1      ;OLDX=NEWX
9462  910A     1980          STA (ADR),Y
9464  C8       1990          INY             ;INC PTR
9465  ADC091   2000          LDA XVAL        ;GET X LSBYTE
9468  8D0493   2010          STA OLDX        ;OLDX=NEWX
946B  910A     2020          STA (ADR),Y
946D  C8       2030          INY             ;INC PTR
946E  ADC391   2040          LDA YVALP1      ;GET Y MSBYTE
9471  8D0793   2050          STA OLDYP1      ;OLDY=NEWY
9474  910A     2060          STA (ADR),Y
9476  C8       2070          INY             ;INC PTR
9477  ADC291   2080          LDA YVAL470           RTS
               2480  ;
               2490  ;
```

```
94BB  AD0260   2500  INXXY  LDA IPTS      ;SET # OF PTS
94BE  8D0293   2510         STA DPTS
94C1  AD0360   2520         LDA IPTS+1
94C4  8D0393   2530         STA DPTSP1
94C7  AD0093   2540         LDA BUFA      ;SET BUFFER ADDR
94CA  850A     2550         STA ADR
94CC  AD0193   2560         LDA BUFA+1
94CF  850B     2570         STA ADR+1
94D1  A000     2580         LDY #$00      ;INIT 1ST VALUE
94D3  B10A     2590         LDA (ADR),Y
94D5  8D0593   2600         STA OLDXP1
94D8  C8       2610         INY
94D9  B10A     2620         LDA (ADR),Y
94DB  8D0493   2630         STA OLDX
94DE  C8       2640         INY
94DF  B10A     2650         LDA (ADR),Y
94E1  8D0793   2660         STA OLDYP1
94E4  C8       2670         INY
94E5  B10A     2680         LDA (ADR),Y
94E7  8D0693   2690         STA OLDY
94EA  209194   2700         JSR IADR      ;INC ADR BY 4
94ED  209D94   2710         JSR DECP      ;COUNT 1ST PT
94F0  8C1093   2720         STY EFLAG     ;INIT END FLAG
94F3  60       2730         RTS           ;RETURN
               2740  ;
94F4  A001     2750  NXXY   LDY #$01      ;SET INDX PTR
94F6  B10A     2760         LDA (ADR),Y
94F8  38       2770         SEC           ;DBL PR X CALC
94F9  ED0493   2780         SBC OLDX      ;CALC LSBYTE DIF
94FC  8D0893   2790         STA XDIF
94FF  A000     2800         LDY #$00
9501  B10A     2810         LDA (ADR),Y
9503  ED0593   2820         SBC OLDXP1    ;CALC MSBYTE DIF
9506  1011     2830         BPL XDCK
9508  8D0993   2840         STA XDIFP1    ;NEG-TAKE DBL
950B  A900     2850         LDA #$00      ;PREC ABS
950D  38       2860         SEC
950E  ED0893   2870         SBC XDIF
9511  8D0893   2880         STA XDIF
9514  A900     2890         LDA #$00
9516  ED0993   2900         SBC XDIFP1
9519  8D0993   2910  XDCK   STA XDIFP1
951C  A003     2920         LDY #$03
951E  B10A     2930         LDA (ADR),Y
9520  38       2940         SEC           ;DBL PR Y CALC
9521  ED0693   2950         SBC OLDY      ;CALC LSBYTE DIF
9524  8D0A93   2960         STA YDIF
9527  A002     2970         LDY #$02
9529  B10A     2980         LDA (ADR),Y
952B  ED0793   2990         SBC OLDYP1    ;CALC MSBYTE DIF
952E  1011     3000         BPL YDCK
9530  8D0B93   3010         STA YDIFP1    ;NEG-TAKE DBL
```

```
9533  A900     3020          LDA #$00      ;PREC ABS
9535  38       3030          SEC
9536  ED0A93   3040          SBC YDIF
9539  8D0A93   3050          STA YDIF
953C  A900     3060          LDA #$00
953E  ED0B93   3070          SBC YDIFP1    ;IF MSBYTE
9541  8D0B93   3080  YDCK    STA YDIFP1
                 3090  ;
9544  AD0A93   3100          LDA YDIF      ;CALC DBL PREC
9547  18       3110          CLC           ;VECTOR LENGTH
9548  6D0893   3120          ADC XDIF      ;L1=DX+DY
954B  8D0893   3130          STA XDIF
954E  AD0B93   3140          LDA YDIFP1
9551  6D0993   3150          ADC XDIFP1
9554  8D0993   3160          STA XDIFP1
                 3170  ;
9557  AD0893   3180          LDA XDIF      ;CALC DBL PREC
955A  38       3190          SEC           ;DIFF
955B  ED0C93   3200          SBC SCAL      ;DIFF=L1-SCAL
955E  AD0993   3210          LDA XDIFP1
9561  ED0D93   3220          SBC SCALP1
9564  B010     3230          BCS BDIF      ;IF>0-PLOT PT
9566  209D94   3240          JSR DECP      ;COUNT PTS
9569  B006     3250          BCS END       ;QUIT
956B  209194   3260          JSR IADR      ;INC ADR BY 4
956E  4CF494   3270          JMP NXXY      ;LOOK AT NEXT PT
9571  A901     3280  END     LDA #$01
9573  8D0293   3290          STA DPTS      ;SET DPTS TO 1
9576  A000     3300  BDIF    LDY #$00      ;INIT INDEX PTR
9578  B10A     3310          LDA (ADR),Y
957A  8D0593   3320          STA OLDXP1    ;SAVE X MSBYTE
957D  C8       3330          INY
957E  B10A     3340          LDA (ADR),Y
9580  8D0493   3350          STA OLDX      ;SAVE X LSBYTE
9583  C8       3360          INY
9584  B10A     3370          LDA (ADR),Y
9586  8D0793   3380          STA OLDYP1    ;SAVE Y MSBYTE
9589  C8       3390          INY
958A  B10A     3400          LDA (ADR),Y
958C  8D0693   3410          STA OLDY      ;SAVE Y LSBYTE
958F  209194   3420          JSR IADR      ;INC ADR BY 4
9592  209D94   3430          JSR DECP      ;COUNT PTS
9595  9005     3440          BCC RNXXY     ;RETURN
9597  A900     3450          LDA #$00
9599  8D1093   3460          STA EFLAG     ;SET END FLAG
959C  60       3470  RNXXY   RTS
```

```
LABEL  TABLE
DIGPT   91C6
BUT1    91C4
XVAL    91C0
XVALP1  91C1
YVAL    91C2
YVALP1  91C3
BELL    FBDD
CLICK   C030
ADR     000A
ARCN    6000
IPTS    6002
ALEFT   6004
ARITE   6006
XMN     6008
XMX     6009
YMN     600A
YMX     600B
TLX     600C
TLY     600E
TRX     6010
TRY     6012
BRX     6014
BRY     6016
BUF     6080
BUFA    9300
DPTS    9302
DPTSP1  9303
OLDX    9304
OLDXP1  9305
OLDY    9306
OLDYP1  9307
XDIF    9308
XDIFP1  9309
YDIF    930A
YDIFP1  930B
SCAL    930C
SCALP1  930D
BCAL    930E
BCALP1  930F
EFLAG   9310
MPTS    9311
NEWPT   933E
NRED    934B
CKXD    936E
CKYD    9394
CKL1    9397
BVEC    93CB
SVEC    93CE
XMYM    93DC
NONZ    93F3
FMNMX   941F
```

```
CXMX    942E
CYMN    9438
CYMX    9447
NMM     9451
RET     9459
SXYB    945A
CLIC    948D
IADR    9491
RADR    949C
DECP    949D
CKZ     94AB
CDECP   94B9
INXXY   94BB
NXXY    94F4
XDCK    9519
YDCK    9541
END     9571
BDIF    9576
RNXXY   959C
```

```
                0010  ;  ROUTINE TO CORRECT ONERR PROBLEMS
                0020  ;  9/14/82
                0030        ORG 95F6
95F6  68        0040  ONERR  PLA          ;GET ADDR FROM
95F7  A8        0050        TAY          ;STACK
95F8  68        0060        PLA
95F9  A6DF      0070        LDX $DF      ;GET NEW STACK P
95FB  9A        0080        TXS          ;RESET STACK PTR
95FC  48        0090        PHA          ;PUT ADDR ON
95FD  98        0100        TYA          ;STACK
95FE  48        0110        PHA
95FF  60        0120  END   RTS


LABEL TABLE
ONERR  95F6
END    95FF
```