KANSAS GEOLOGICAL SURVEY
OPEN-FILE REPORT 88-45c

GIMMAP Software and Technical Information

by

Charles G. Ross

Disclaimer

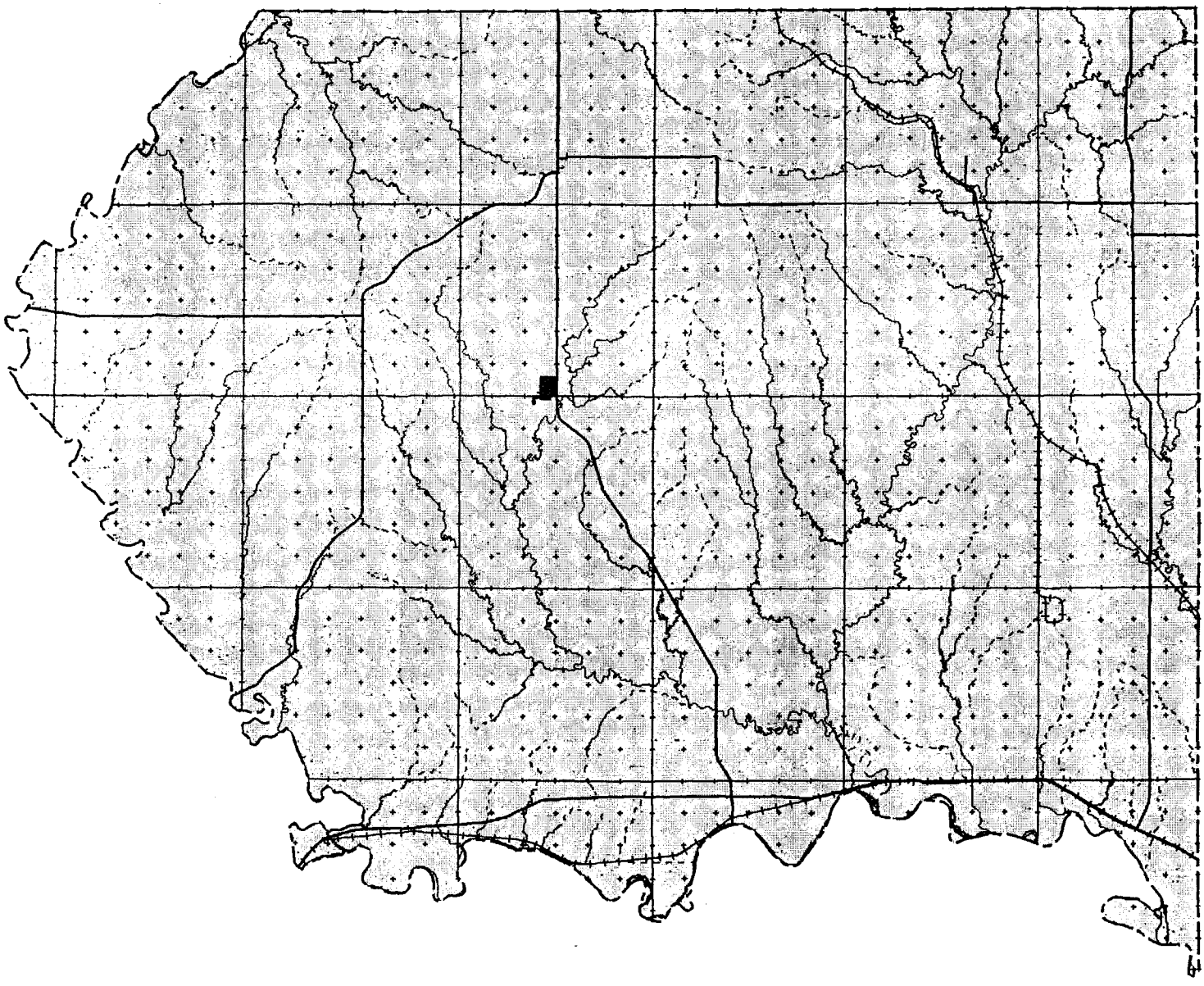
The Kansas Geological Survey does not guarantee this document to be free from errors or inaccuracies and disclaims any responsibility or liability for interpretations based on data used in the production of this document or decisions based thereon. This report is intended to make results of research available at the earliest possible data, but is not intended to constitute final or formal publications.

KANSAS GEOLOGICAL SURVEY
1930 Constant Avenue
University of Kansas
Lawrence, KS 66047

Missing Page #18 from Technical Notes

GIMMAP Software & Technical Information

by Charles G Ross



KANSAS GEOLOGICAL SURVEY

TO:

Joe's mom, Gene, Van, and Bear

"the best part about writing is the finishing"

(quoted by Steven Spielberg)

"All good writing is swimming under water
and holding your breath."

F. Scott Fitzgerald

Foreword

The compilation of this information about the **G**eodata **I**nteractive **M**anagement **M**ap **A**nalysis and **P**roduction or **GIMMAP** ("jim-map") system has been a long process, beginning with the first programs written in 1977. As with the conquest of the great mountains, there have been many different attempts made over the years. This set of separate volumes has finally resulted from the union of the efforts of those people involved directly in the creation and use of GIMMAP with the support of the administration of the Kansas Geological Survey.

First, I must acknowledge the people who have contributed to the completion of this work.

I thank my wife, Gina, and my children, Catherine and Patrick, who are my reason for living. And I must credit Gina, whose love and encouragement helped in many ways to bring me to the light at the end of this tunnel.

Trang Cao initiated GIMMAP in Orleans, France while he was working for the BRGM there. He brought ideas and the foundation for the system when he was Visiting Research Scientist in the 1977-78 academic year. His ideas and good will in working with me made the potential for an automated cartography system a reality at the Survey. His work is a shining example of what can be.

David Collins helped to provide momentum, direction and many, small packets of inspiration, and helped greatly to organize and edit the final effort.

Finally, this work would not have been completed without the commitment to the definition and prioritization of the project by Dr. Lee Gerhard, Director of the Kansas Geological Survey.

Creation of these four volumes of GIMMAP documentation has been a long and arduous process, marred by difficulties which might have been avoided. Important lessons have been learned about the process of creating software documentation and I would feel remiss if some of these were not passed on to. The foreword may not be the best place to air these thoughts, but I feel they are of a fundamental importance.

Of the lessons learned, these might be the most important:

1. The first step is to clearly define in writing the task which is to be done. This includes identifying the target audience, the purpose and function, and the form and scope of the documentation.

2. For the job to be well-defined and well done, a group of people must participate in overseeing and performing the task at all stages. This group should consist of administrators, people who use or benefit from the software, the software author(s), and experts in software, education, technical writing and marketing. Strong and generous support for this group and the documentation authors by the administration in organization, project definition and the provision of resources is required for success.

3. The software author is probably not the best nor even a good choice to be the author of the documentation. The skills required to write software documentation are not the same as those required to write computer software. Generally and historically, an experienced writer with knowledge of the software is the best choice. The software author is the best source of technical information on the software, prepared as draft material for the software documentation during implementation of the software. Comprehensive documentation of a complex software system such as GIMMAP can not be properly prepared in scattered pieces of time by software authors who have too many responsibilities in creating, expanding and maintaining the software.

4. The group of experts (above) should estimate the cost of the project if it is to be done right. Money, time, personnel and other resources should be discussed and consensus reached. To do this, clear definition of the priorities of all affected personnel must be made. Overlapping expectations coupled with improperly low estimations of the size of the task have led to disappointments in the past.

5. The reason(s) for writing the software in the first place should be well understood and accepted by the group as the foundation of the discussions implied above.

II. B A S I C S E Q U E N C E

II.0 INTRODUCTION

This section describes the programs of the Basic Sequence described above. The programs in this sequence are used to take map data from an original source map, generate a map database for the map, edit to correct graphical errors, create and edit zones for colored or shaded areas, create and edit symbology, and then extract data from the digital database to reproduce an acceptable facsimile of the original.

The programs included in the Basic Sequence are:

1. MAPDIG - Map digitizing
2. SYNEDT - Syntax editing
3. PARGEN - Point and arc generation
4. NODGEN - Node generation
5. GRFCHK - Graphical checking
6. GRFEDT - Graphic editing
7. PLTGEN - Plot generation
8. ZONGEN - Zone generation
9. ZONEDT - Zone editing
10. CYMBAL - Symbology creation and editing
11. MAPGEN - Map generation

Each of these programs will be treated in some detail with a section devoted to as many as six topics, though some topics may be excluded or modified for some programs.

The six topics which are generally included in discussions of the programs are:

A. Introduction - introductory material

B. General - objectives, theory, terminology, flow

C. Operation - dialogue, sequence, interaction, options

The fourth and fifth topics for program discussion cover information for the programmer or technical person. Both may include several sub-topics:

D. Programmer's Overview -

1. introduction to routines
2. global storage and variables
3. files accessed
4. memory and other requirements

E. Routines in Detail -

1. Section for each routine
2. Functions, purposes
3. Major local variables
4. Related routines

The sixth topic may vary widely in its form, but will generally be an attempt to illustrate the use of the program by including example data sets along with the associated specific dialogue and other pertinent information for an actual run of the program or at least a bona fide simulation:

F. Example Run - a practical example of program use

The programs described in this section are ordered as listed above, by order of their general usage within the basic sequence rather than in alphabetical order. A brief description of all programs by alphabetical order is found in the Introductory section (Introduction to Software).

II. B A S I C S E Q U E N C E

II.1 MAPDIG - THE MAP DIGITIZING PROGRAM

Introduction

The MAPDIG program operates to support the interactive capture of graphic data from maps and the conversion of that data to a digital form for processing by the GIMMAP system. The **digitizing** of map data, the conversion of the graphic form into the computer-usable form, is the beginning step in the map-making process which is supported by GIMMAP.

The MAPDIG program runs in conjunction with a digitizing table and an associated operator terminal. The digitizing table includes a moveable cursor with a minimum number (four) of distinguishable signals to support various functions of the software. The buttons may be mapped to functions of the process via assignments existing in a parameter file. Once programmed, one button may signal a node and another may signal a flag for a feature code to be entered.

The MAPDIG program attempts to adhere to and support the rules of the GIMMAP syntax (see the SYNEDT program) which define the form of nodes, flags, headers, and interior or isolated points in the output file. The syntax also defines the order and relationships between these objects in the **Digitized Data Format** (DDF) file which is built by MAPDIG.

As part of the setup procedure and the DDF file, the MAPDIG program creates and writes the map header and the control points of the file. The map header is built to contain information for the GIMMAP system such as the projection type, scale, operator's initials and date. The control points are digitized in a special GIMMAP procedure which checks the map being digitized for gross distortion errors in excess of certain limits. This operation is applied to the primary type of map being digitized for GIMMAP - the USGS 7.5' quadrangle map only. Some tests are also performed on the consistency of the operator.

The routines which make up the MAPDIG program are described in general below.

The MAPDIG Routines

BELCMD

The BELCMD routine is called to set the mode of the terminal bell to on or off. When the bell mode is on, normal digitizing does not cause the bell to ring, but special requests and errors

do cause the bell to ring so the operator can observe a difference. When the bell mode is off, essentially the reverse is true. Normal digitizing causes the bell to ring and special events cause the bell to remain silent, again to attract the operator's attention.

CHRINT

The CHRINT routine is called to decode a string of decimal digits to produce an integral number such as a feature code. The digits are entered by the digitizing operator for entry into the DDF file and must be converted to number form.

DIGITZ

The DIGITZ routine represents the command state of the MAPDIG program in which the program is awaiting the selection of the digitizing operator by the cursor buttons or the terminal keyboard. When the operator selects an action, the DIGITZ routine deciphers the action adn branches to the appropriate routine to perform the desired action. If the operator selects the feature code or header flag button, then a feature code must be entered at the keyboard. Then, the routine branches to the appropriate routine to support the digitization of the next feature.

DIGLIN

The DIGLIN routine controls the digitization of all line-type arcs according to the rules of the GIMMAP syntax for digitizing. For line-type arcs, the pattern is to digitize the first node of the arc, the interior points, and the last node. The nodes must be differentiated from the interior points in the DDF file by existing as consecutive, duplicate points. This may be achieved by using the node function button of the cursor or by actually the nodes twice (unless prohibited by the software).

Discontinuities are points in digitizing where the beginning of the next is not at the same location as the end of the previous node, but where both arcs are of the same code. These must be signalled by discontinuity flags in the DDF file. The flags are entered by application of the discontinuity button, and are placed between arcs in the file which are not continuous.

The operator may place a header flag (a double discontinuity flag) to signal a change to a new feature code, which is entered at the keyboard. This follows the end node of the last arc for the previous feature code. The DIGLIN routine prints warning messages when syntax or other errors are found, no correction is attempted and the program continues to process digitized data. The terminal bell is not sounded during normal digitizing, but is reserved for announcing errors and special conditions.

DIGPNT

The DIGPNT routine controls the digitization of isolated point arcs, each of which is digitized by pressing the Point or Node button one time only. Each digitized point represents one isolated point arc at the designated location. The Header button is used to end the digitizing of these isolated points, perhaps to reset the feature code for different isolated points. The DIGPNT routine prompts for all z-values if they are to be entered with the points.

GETCTL

The function of the GETCTL routine is to control digitization of the map **control points** under the rules of GIMMAP. The control points lie at the corners of the map area, and are digitized in the counter-clockwise order, beginning and ending at the southwest corner (which is digitized twice for accuracy control).

The (averaged) location for the southwest corner is used to determine offsets used to translate the data so that the southwest corner is at (1.0,1.0) and the southwest and southeast corners are used to determine the sine and cosine of a rotation angle which makes the line through these two corners horizontal (the y-value of both will be 1.0). The values are set up for the transformation of the digitized data during the session.

When the control points are not located at the corners of the map area (according to the projection parameters), then some other set of rectangular points with known geographic coordinates must be digitized as the **non-standard control points.** For these points, the operator must enter the geographic coordinates describing the rectangle.

GETTBL

The GETTBL directly receives all coordinate information from the digitizing table. The GETTBL routine responds to any error conditions by reporting the error to the terminal. Then the table file is closed and reopened, and the input is retried.

HLPCMD

The HLPCMD is called to display a complete listing of the commands available in the MAPDIG program.

KEYBRD

The KEYBRD routine accepts command input and new feature codes at the keyboard, calls the PARSER and CHRINT routines to decode the input, and returns the results.

MAIN

The MAIN routine is the central routine for the MAPDIG program. The MAIN routine initializes global variables and sets up the environment. The files are opened (keyboard input, terminal output, and the parameter file - cursor buttons) and the cursor button assignments are read from the parameter file. MAIN requests the name of the output DDF file, opens it and puts the map header (gotten from the operator) in the file.

The MAIN routine then calls the GETCTL routine to get the control points and to set the transformation parameters for the digitized data. Then, MAIN calls the DIGITZ routine to oversee the digitization of the feature data. When digitizing is complete, MAIN closes the files and reports the number of nodes and points digitized.

MSG

The MSG routine is called to write warning messages and status messages to the terminal, with each message selected by a unique number. The MSG routine also rings the terminal bell by calls to the RNGBEL routine if ringing is not turned on for normal point digitizing.

PARSER

The PARSER routine has the task of analyzing the command or feature code input from the keyboard to determine what the next recognizable token (unit of command syntax) in the command is. The type of the token is determined to be numeric or non-numeric. This information is then returned.

RNGBEL

The RNGBEL routine sends a character to ring the terminal bell. If Bell Mode is ON (bell rings for all digitized points), then the bell is not rung for error messages or unusual conditions. If the Bell Mode is OFF, the opposite is true. Thus, regardless of the setting, errors are signalled to the operator by the change (to presences or to absence) of the bell.

SETCBP

The SETCBP routine sets up the cursor button programming as specified in the parameter file. The name of a file containing the assignments of cursor buttons (by number or letter) to functions (nodes, flags, headers,...) in digitizing. If the file is found and opened, the new assignments are read and performed to reset the

list of default button assignments. If the (parameter) file is not found and successfully read, the defaults remain intact. In either case, the resulting assignments are listed on the terminal screen.

TRANS

The TRANS routine uses the values set in the GETCTL for the original southwest corner (XSW, YSW), the desired location for the southwest corner (NEWXSW, NEWYSW), and the sine and cosine of the angle of rotation required to orient the map so that the line from the southwest corner through the southeast corner is horizontal (if necessary) to translate all digitized data.

WRTDDF

The WRTDDF routine has a collection of entry points for the different functions it provides. The WRITPT entry point writes out the coordinates of transformed data to the output DDF file. The WRITFC entry point writes each new feature code to the output DDF file, possibly with a count of z-values when appropriate. No call is made to TRANS for feature codes. The WRITFL entry point writes header and discontinuity flags to the output DDF file with no call for transformation. Finally, the DELPNT entry point causes the output DDF file to backspace, thus deleting a single, previously written record. All affected entry points update the point counts by incrementing or decrementing the counters.

II. B A S I C S E Q U E N C E

II.2 THE SYNEDT PROGRAM

A. Introduction

The **SYNEDT** program, for syntax editing, provides a complete inspection of the content of a **raw** (straight from the digitizer without modification) digitized data format (**DDF**) file. The entire structure of the DDF file must adhere completely to the rules of the digitizing **syntax**, which define the form and meaning of the DDF file.

The SYNEDT program inspects the DDF file from its beginning to the end, from the first record containing the map header to the last double flag indicating the end of the file. When errors are found in the file, the SYNEDT program attempts to correct the errors so that the file will adhere to the syntax rules.

Although some errors make little sense and do not occur in the real world of digitizing in the GIMMAP system, they must be corrected to make the file follow the syntax so that subsequent programs can use the **clean** or **syntactically clean** (containing no syntax errors) DDF files without error. For this reason, all errors are corrected to fit the syntax even though the resulting data may not make sense in the normal course of digitizing.

B. General Information

The objectives of the SYNEDT program are to inspect the DDF file submitted by the operator, recognize and correct any part of the file which does not conform to the rules of the digitizing syntax, and to report on statistics of the data in the file and the activities of the program. The resulting output file of the SYNEDT program is to be a syntactically-clean DDF file which can be used as input to the PARGEN program without causing any errors based on syntax rules.

The digitizing syntax is defined in terms of the objects and relationships between objects which are being digitized. In this case, the map features recognized by GIMMAP are those which the syntax uses to define rules of form and sequence. For example, arcs are sequences of single (interior) points describing the path of the arc, with an endpoint location on each end described as consecutive identical points.

The syntax may be understood as a form and format describing the content of the DDF file. As such, the DDF file consists of three distinct parts:

1. **The Map Header**

2. **The Control Points**

3. **Map Feature Information**

The map header is contained in the first record in the DDF file. It is a collection of values entered by the operator and stored as character (rather than numerical) values in whatever order the operator chooses. Generally, it includes the base name of the map, the operator's initials, the projection type, the map scale, digitizing session number, the date and other information pertinent to the file. The map header is displayed by editing programs, and is available to operator inspection, but has no other function for GIMMAP programs.

The control points are represented as five pairs of points represented in projected (x,y) coordinates. The five points are found in the five records immediately following the map header (the first record in the file) in the DDF file. These points represent the digitized locations of the four corners of the map area digitized in order from the southwest to the southeast, the northeast, the northwest and the southwest a second time.

The bulk of the information in the DDF file is usually the map feature information. This information consists of two types of entities. One is the arc type feature, the other is the point type feature. Both of these features are preceded by a **feature code header** which identifies the type of feature by inclusion of a feature code. Both are followed by either a **discontinuity flag** or a **header flag**, to mark the end of the feature and to indicate the beginning of some new entity.

The arc type feature is the string of single points defining the path of a linear arc feature. The string of single points is begun by duplicated points (consecutive identical points) called a **start node** and ended by a second pair of consecutive duplicated points called an **end node.** When the end node of one arc is the same as the start node of the next to be digitized, then only one pair of duplicated points is entered to serve as the start node of one and the end node of the other. This node is referred to as a **continue node.**

Where arcs are not connected or continuous, so that the end node of one arc is not the same as the start node of the next arc to be digitized then one of two things is done. If the second

arc has the same feature code as the first, then a discontinuity flag, a point with negative coordinates such as (-1.0,-1.0), is entered after the end node of the first arc and before the start node of the second arc. As an example:

```
    .
    .                Previous feature data
    .
X1 , Y1
X1 , Y1              Start node for arc 1
    .
    .
    .                Interior points for arc 1
    .
    .
X2 , Y2              End node for arc 1
X2 , Y2
-1.,-1.              Discontinuity Flag
X3 , Y3
X3 , Y3              Start node for arc 2
    .
    .
    .                Interior points for arc 2
    .
    .
X4 , Y4
X4 , Y4              End node for arc 2
```

When the next arc to be digitized does not have the same feature code as the previous one, the new feature code must be signalled to the program. This is done by inserting a **header flag**, two consecutive points with negative coordinates - equal to two consecutive discontinuity flags, following the end node of the last arc digitized. The header flag signals that a **feature code header**, a point with the feature code as the x coordinate and zero as the y coordinate, is to follow so that all following features will be assigned the new feature code until a new code is found. The feature code header is also used to signal the end of the DDF file.

The feature code header is an **arc header** and precedes arc type features if the code in the x value is greater than 1999 and less than 10000. If the code is between 1000 and 1999, the feature code header is a **point header** and is followed by point type features.

Point features represent single locations known as **isolated point** features in GIMMAP. In the syntax of the DDF file, these features are represented each as a single point, with no special flags between them since each feature is completely described by the single location. Following a point header, all points are

treated as isolated points until a feature code header is found to reset the feature code and perhaps the feature type.

The following example illustrates all the entities of syntax as described above. A sample map is shown in Figure II.2.?? to illustrate the map data from which the description and sample file content are taken.

The digitization of the sample map may be performed in many different sequences, but each approach should result in a DDF file which is syntactically correct and structurally equivalent to the others. First, let us assume that the map header has been properly entered and the five control points have been digitized. The procedure for digitizing this sample map is as follows.

Note that nodes are described as being digitized twice. This action produces a duplicate pair of coordinates in the DDF file as needed for recognition by the SYNEDT program. In fact, this may be accomplished by pushing the digitizing button twice or by pushing a special node button once, with the same effect.

The arc header is entered for the arcs connecting nodes 1, 2, 3, 4, and 5, assuming these arcs all have the same feature code. The operator then digitizes node 1 twice (start node) and then the interior points along arc 1 to node 2 (once each). The operator may continue digitizing to node 3 without entering a discontinuity flag, provided that node 2 is digitized twice.

Node 2 is a continue node, acting as an end node for arc 1 and a start node for arc 2. The operator digitizes node 2 twice and then, without entering a flag, digitizes the interior points along to node 3. This procedure is repeated at node 3 (also a continue node) in digitizing arc 3 to node 4.

Notice that if node 4 did not exist, nodes 2 and 3 would be connected to each other by two different arcs. This example of **bridging** was corrected by the placement of an **artificial node** (4) to avoid the ambiguity arising when the arcs must be identified by the node numbers at their endpoints.

The operator digitizes node 4 twice and then digitizes the interior points to node 2, which is an end node. Notice that the terms start, end, and continue are relative to the sequence of digitizing; they are not defined by the graphic structures.

The operator digitizes node 2 twice as the end node. Since the next arc is not continuous from node 2, (but does have the same feature code), the operator enters a discontinuity flag and moves to the next arc. Start node 3 is digitized twice and the interior points (if any) to node 5 are digitized. Then node 5 is digitized twice as the end node.

The next feature to be digitized, according to the sequence selected by the operator, is an Isolated Point feature. This requires a change in feature code and the assumed discontinuity. This information is signalled by entering a header flag followed by a point header containing the new feature code.

Since isolated points are indicated in the point header, the operator digitizes point 6 only one time. The next feature, also an isolated point (7), obviously follows a discontinuity. This situation is assumed to exist for isolated points and so no flag is entered. It is also possible that the feature code for point 7 could be different than point 6. This would require insertion of another header flag and point header. However, in this case, the codes are identical and point 7 is digitized once following point 6.

The operator enters a header flag and a new (line-type) arc header before proceeding to node 8. This arc, starting and ending at node 8 is an **island** - an area surrounded completely by another area. Since no node is defined by arc intersection, the operator has selected an artificial node to serve as both the start and end node of the arc. Node 8 is digitized twice, the interior points are digitized, and node 8 is digitized twice for a second time. Since this is the end of the data, a header flag is entered to complete the file.

The DDF file resulting from the sample map when digitized as described above is shown below:

```
Mapname,Op,Date,Projection...      Partial Map Header
              XSW,YSW          Southwest control point
              XSE,YSE          Southeast control point
              XNE,YNE          Northeast control point
              XNW,YNW          Northwest control point
              XSW,YSW          Southwest control point
      featurecode, 0.      Arc Header (Line type feature code)
              X1 , Y1          Node 1
              X1 , Y1          (Start node for arc 1)
                 .
                 .             Interior points for arc 1
                 .
              X2 , Y2          Node 2
              X2 , Y2          (continue node)
                 .
                 .             Interior points for arc 2
                 .
              X3 , Y3          Node 3
              X3 , Y3          (continue node)
                 .
                 .             Interior points for arc 3
                 .
```

```
          X4 , Y4          Node 4
          X4 , Y4          (continue node)
             .
             .             Interior points for arc 4
             .
          X2 , Y2          Node 2
          X2 , Y2          (end node for arc 4)
          -1.,-1.          Discontinuity Flag
          X3 , Y3          Node 3
          X3 , Y3          (start node for arc 6)
             .
             .             Interior points for arc 6
             .
          X5 , Y5          Node 5
          X5 , Y5          (end node for arc 6)
          -1.,-1.
          -1.,-1.          Header Flag
 featurecode, 0.           Point Header (point feature code)
          X6 , Y6          Isolated Point 6
          X7 , Y7          Isolated Point 7
          -1.,-1.
          -1.,-1.          Header Flag
 featurecode, 0.           Line Header (line feature code)
          X8 , Y8          Artificial Node 8
          X8 , Y8          (start node for arc 10)
             .
             .             Interior points for Island arc 10
             .
          X8 , Y8          Node 8
          X8 , Y8          (end node for arc 10)
          -1.,-1.
          -1.,-1.          End of DDF file
```

The digitizing syntax is codified by identifying a set of **states** required to model the syntax editing process; a set of **tokens** to describe all possible entities to be found in the raw, input DDF file; and a set of **actions** to be taken to correct syntax errors or to proceed with the file.

The states, tokens and actions for the syntax are all listed below in Figure II.2.?. The syntax for GIMMAP digitizing is coded in two different forms as illustrated in the **state diagram** which shows all transitions between the states according to the tokens encountered, and in the **transition matrix** which shows in tabular form all state-to-state transitions according to tokens and the actions taken by the program. The state diagram is illustrated in Figure II.2.? and the transition matrix is illustrated in Figure II.2.?.

In processing the input data, SYNEDT reads the points into a **Point Buffer**, which holds enough points to properly analyze for

and correct syntax errors. Points enter the Buffer on the upper end, are analyzed in the middle beginning at the **active position** of the Buffer, and output data is written to the clean DDF file from the lower (first) position of the Buffer. The tokens are recognized at the active position and the actions are applied to the content of the Point Buffer. States and connections between the states (transitions) represent the rules of the syntax. The state names generally reflect the type of token which was last found:

States For The SYNEDT Program

1. PH - Header for point type arc, with code < 2000
2. AH - Header for line type arc, with code > 2000
3. SN - Start node, double point
4. EN - End node, followed by a flag
5. IP - Interior point, single point for line type arc
6. SP - Single point, single point for point type arc
7. DF - Discontinuity Flag, negative coordinates
8. HF - Header Flag, double DF

Tokens are found by analyzing the contents of the XY Point Buffer at the active locations. Each token is unique and tokens are designed so that any configuration in the Buffer will result in the recognition of one of the listed tokens.

Tokens For The SYNEDT Program

1. DF - Discontinuity Flag
2. HF - Header Flag
3. PH - Point Header
4. AH - Arc Header
5. SC - Start or Continue Node
6. EF - End Node With Flag
7. SP - Single Point

Actions listed here are each represented by a single digit in the Transition Matrix. The entries in the Transition Matrix group multiple actions for a transition into a single number of two or more digits. The digits in multiple action transitions are listed **in the reverse order** of their intended use. That is, the action indicated by the least significant digit is done first and the action represented in the most significant digit is last. Negative numbers indicate a transition in violation of the rules of the GIMMAP digitizing syntax.

Actions For The SYNEDT Program

1. OI - Output from top of Point Buffer, shift Buffer up through all positions and then input to last position

2. CP - Copy previous point, copies from position above the current position to the position above that

3. DC - Delete the current point, deletes the point in the active position of the Buffer by shifting into it

4. RP - Report the first point or start node of a new arc and report error counts with the location

5. RH - Report line or point type header, set processing mode to line or point and set feature code to apply

6. IFH - Insert flag for Header, inserts a Header Flag in the Point Buffer immediately above the active position

7. DP - Delete the previous point, deletes the point in the position above the active position in the Buffer

8. IH - Insert feature code header, inserts a header above the active position in the Buffer - the type of header depends on the data below the active position

C. Operation

Operation of the SYNEDT program requires a short dialogue to select files and set a few values governing the recognition of nodes, translation of data, and default z-values. The dialogue begins (in FILOPN) with the opening of the DDF files.

C(omputer): Enter Output File name?

O(perator): Operator enters the clean output filename

C: Enter Input File name?

O: Operator enters the input DDF filename

C: SYNEDT displays the map header from the Input DDF file and offers three options:

1 = continue (accept the map header)
0 = change the (map) header
-1 = stop (the SYNEDT program)

O: The operator enters the selected option

[If option = 0]

C: SYNEDT requests the new header

O: The operator enters new header

[End of option]

C: SYNEDT reports the node recognition threshold with options:

1 = accept current threshold
0 = reject and reset threshold

O: The operator selects an option

[If option = 0]

C: SYNEDT requests new threshold

O: The operator enters the new value

The threshold cycle is repeated until accepted

[End of option]

C: SYNEDT reports the value of the X offset used for translating output data when selected with options:

1 = accept the X offset
0 = reject and reset the X offset

O: The operator selects an option

[If option selected is 0]

C: SYNEDT requests new X offset

O: The operator enters new X offset

The X offset cycle is repeated until accepted

[End of option]

C: SYNEDT reports the Y offset with options:

1 = accept the Y offset
0 = reject and reset the Y offset

O: The operator selects an option

[If the option = 0]

C: SYNEDT requests the new Y offset

O: The operator enters new Y offset

The Y offset cycle is repeated until accepted

[End of option]

C: SYNEDT requests a default value for Z-values for Isolated Point arcs

O: The operator enters the default Z-value

This ends the dialogue between SYNEDT and the operator. The program then begins the cycle of processing the DDF input to correct syntax errors. This cycle consists of the three steps of (1) evaluating the token in the active positions of the Point Buffer, (2) executing actions indicated by the transition matrix to correct and proceed, and (3) changing the state according to the transition matrix, the previous state and the current token.

When the end of the file is found, SYNEDT produces a report of the counts of data types found, counts of syntax errors and corrections, and counts of each type of action taken by SYNEDT in correcting the data.

D. Programmer's Overview

In this section, various aspects of the SYNEDT program are examined from the more technical viewpoint of a programmer. The information in this section is introductory and not necessarily complete. The source code itself contains complimentary material necessary to a complete understanding of the software.

1. Introduction to Routines

All routines which comprise the SYNEDT program are written in the FORTRAN77 programming language. The routine names and the basic function of each are listed in alphabetical order:

ACT - Perform indicated actions with Point Buffer

FILOPN - Open all files

INIT - To initialize the state of the Point Buffer

INP - To input points to the Buffer

MAIN - Controlling routine with set-up and dialogue

OUTP - To output data to the clean DDF file

REPORT - Reports counts of data, errors and actions

SHIFT - Moves data through the Point Buffer

TERM - Empty Buffer at termination

TOKEN - Recognize token in the Buffer active position

TRANS - Translates output data

2. Global Storage and Variables

The global variables of the SYNEDT program are listed below according to the COMMON areas in which they are declared. Each COMMON area is listed with the name bounded by slashes as in the program. Each variable is described with pertinent information.

COMMON/COUNT/

ICNT(9) - Counts objects in the Point Buffer:

(1) discontinuity flags	(6) end nodes
(2) header flags	(7) single points
(3) isolated point headers	(8) isolated point arcs
(4) line arc headers	(9) all arcs
(5) start or continue nodes	

NACTEX(8) - Counts use of the eight SYNEDT actions

IECNT(3) - Counts of errors

(1) All errors	(2) Deletions
(3) Insertions	

MODE - Type of arcs being processed, 1 = isolated point arcs, 2 = line type arcs

IGNOR - Number of positions in the Point Buffer to pass over on start-up or after deletion

IFILL - Number of positions in the Point Buffer to fill with new data on start-up or regular input

IOCNT(2) - Count of input (1) and output (2) data points in the Point Buffer

COMMON/FC/

IIFC - Unit number (file code) for Terminal Input file

IOFC - Unit number for Terminal Output file

IDFC - Unit number for the raw, input DDF file

ICFC - Unit number for the clean, output DDF file

COMMON/LOCNT/

NAERR - Total number or arc errors found

NINS - Number of insertions for error corrections

NDEL - Number of deletions for error corrections

COMMON/MC/

NUMFC - Number of feature codes to be applied (for the multiple feature code option)

MFC(4) - Storage for (up to four) feature codes for the multiple feature code option

COMMON/TM/

MAT(9,7,2) - The Transition Matrix, with row numbers for the current state, columns as tokens, actions (1) and next states (2)

COMMON/TXYTR/

THRESH - The distance criteria used to determine when two consecutive points lie close enough together to be considered as a single node

XOFF - Translation in the x coordinate to be applied to points at output

YOFF - Translation in the y coordinate to be applied to points at output

COMMON/Z/

NUMZIN - Number of Z values, found on input tokens

NUMZOUT - Number of Z values, on output points

DEFAULT - Default Z value applied to unused Z values

3. Files Accessed By SYNEDT

The SYNEDT program accesses only four files. These are the input and output files for the interactive terminal; the input file of "raw", unedited digitized data; and the output file of (syntactically) clean DDF data.

Unit Number	File Name	Characteristics
IIFC (11)	Terminal In	Input from terminal, padded, sequential
IOFC (10)	Terminal Out	Output to terminal, sequential with FORTRAN carriage control
IDFC (12)	Raw DDF Input	Sequential, padded, input, maximum record length = 124B
ICFC (13)	Clean DDF Output	Sequential, output

4. Memory and Other Requirements

The SYNEDT program has no memory or other requirements which are noteworthy. The data points are processed in an essentially sequential manner passing through a buffer which holds only five points at a time. The program itself is not overly large, and it requires no large data structures. The largest single array has only 126 elements and requires only 252 bytes of storage.

E. SYNEDT Routines in Detail

The routines for SYNEDT are listed above. In this section, they are developed with some general information including the argument list, a description of the functions of the routine, a listing of other routines called, a list of routines which call it, an overview of the routine flow, and a list of major local variables with their type and an explanation of the use of the variable and other pertinent information.

ACT (XY, IBSIZ, IPOS, NACT, ICODE, NTOKE)

The ACT routine performs the actions indicated by the action code (NACT) to correct syntax errors or to report events and put clean data in the output file. Routines called by ACT are INP,

OUTP, SHIFT, TOKEN, and TRANS. The ACT routine is called only by the MAIN routine.

The ACT routine receives the action number, the Buffer and pointers, current token and feature code to insert (if a feature code header is missing) from the MAIN routine. The action number is divided into primitive action numbers, each a single digit, to perform the series of primitive actions indicated.

Each primitive action number is compared to the eight action numbers recognized by ACT. When a match is made, the indicated primitive action is executed, usually affecting the contents of the Point Buffer (XY).

Negative action numbers contained in the Transition Matrix and sent to ACT by MAIN indicate an illegal transition or syntax error in the data. The error count is increased and the action number is made positive for normal processing.

The insertion and deletion primitive actions occur only when the transition indicated by the Transition Matrix is a error in the syntax. Error counts for these operations are incremented each time the operations are performed.

The major variables used by the ACT routine are:

Variable Name	Type/Size	Use
XY(IBSIZ,9)	REAL*4	Point Buffer of IBSIZ points each with X, Y, and 7 optional z-values
IBSIZ	INTEGER*4	Size of XY Buffer, set at 5 in MAIN
IPOS	INTEGER*4	"Active Position" in the XY Buffer
NACT	INTEGER*4	Action number from the Transition Matrix, combines primitive actions
ICODE	INTEGER*4	Feature code to insert when missing
NTOKE	INTEGER*4	Token number in active position

FILOPN (no arguments)

The FILOPN routine opens all files associated with the SYNEDT program (see above). The output file name is requested and obtained from the operator, and the file is opened. Then, the input file is similarly identified and opened. The FILOPN routine calls no other routines and is called only by MAIN.

The FILOPN routine first opens the standard input and output files for the terminal. FILOPN then asks the operator to select the input DDF file. When the file is selected, FILOPN opens the file. A similar procedure then occurs for the output DDF file.

The major variables used by the FILOPN routine are:

Variable Name	Type/Size	Use
IO	CHARACTER*60	Character variable used for the filenames to open both the input and the output DDF files

INIT (XY, IBSIZ, IPOS, NSTAT)

The INIT routine initializes the Point Buffer (XY), loading it up to the active position with unedited data from the input DDF file. The INIT routine also sets initial counts for input and output data, and sets the initial state and other flags. It calls INP, OUTP, and SHIFT, and is called only by MAIN.

The INIT routine first checks the Buffer pointers and size and sets the current state value as HF for header flag. The beginning position for initial data is set and the Point Buffer is filled with that amount of data (the control points). The control points are shifted so the last one is in the position above the active position. Thus the first header should reside in the active position.

The major variables used by the INIT routine are:

Variable Name	Type/Size	Use
XY, IBSIZ, IPOS		See ACT above
NSTAT	INTEGER*4	Number of the current state, set by ACT using the Transition Matrix

INP (XY, IBSIZ, NPOS)

The INP routine inputs new data into the indicated position (NPOS) of the Point Buffer (XY). The INP routine reads the X and Y coordinates and attempts to read Z-values when they expected. The INP routine loads a special (-999,-999) point to signal when the end of the file has been encountered. The INP routine does not call other routines and is called by ACT, INIT, and TOKEN.

The INP routine reads the next point values into position NPOS in the Buffer. First, the z-values in that position are set

to the default value (DFAULT). The next record from the input file is read with the X and Y coordinates and optional z-values saved in the Buffer. For isolated points, the number of z-values is set. If it is a header, the value may be reset. The count of input values is incremented.

If an end-of-file condition is encountered, INP branches to a section which sets the input values in the Buffer to very large negative values (-999.999,-999.999) as a signal to MAIN if the file really is at the end. INP then cycles to retry the input. If end-of-file occurs twice then the signal is returned.

The major variables used by the INP routine are:

Variable Name	Type/Size	Use
XY, IBSIZ, NPOS		See ACT above
NTRY	INTEGER*4	Marks first pass for failure retry

MAIN

The MAIN routine controls the flow of the SYNEDT program from calling the FILOPN routine to open the files, through the cycle of transitions from state to state via tokens and actions, to calling the REPORT routine to display the final counts of the errors, corrections and other actions taken. MAIN calls the ACT, FILOPN, INIT, OUTP, REPORT, SHIFT, TERM, and TOKEN routines.

MAIN initializes the transition matrix, default values for the global variables, sets values to control the initialization of the Point Buffer, and calls INIT to enter initial values in the Buffer. The MAIN routine provides the dialogue between the program and the operator, described above.

The **Transition Matrix** is represented by a three dimensional array (MAT) with nine rows representing current states and seven columns representing the input tokens found in the Point Buffer. Transitions from state to state are defined in the State Diagram, and are paralleled in the Transition Matrix. The two values stored in the Transition Matrix for each state/token element are (1) the action number (combines multiple primitive actions), and (2) the next state as defined by the syntax in the State Diagram.

MAIN provides the basic cycle for analysis and correction of input DDF data to produce clean output DDF data. MAIN calls the TOKEN routine to identify the token in the active position of the Point Buffer. With the token and current state, MAIN calls the ACT routine to perform the actions and sets the next state using values from the Transition Matrix.

When the INP routine, acting on the instruction of the ACT routine, encounters the end-of-file for the input DDF file, MAIN calls the TERM routine to empty the remainder of the Point Buffer to the output DDF file. Before termination, MAIN calls REPORT to list counts of data, errors, and actions taken.

The major variables used by the MAIN routine are:

Variable Name	Type/Size	Use
XY, IBSIZ, IPOS NSTAT, NTOKE	See ACT above	
MHEAD	CHARACTER*64	Contains the Map Header
ICODE	INTEGER*4	A default feature code to replace missing feature codes
LSTCOD	INTEGER*4	Stores the previous feature code to avoid redundant headers
IRESP	INTEGER*4	Stores operator integer responses

OUTP (XY, IBSIZ, NPOS)

The OUTP routine writes a point from the first position of the Point Buffer to the clean output DDF file, on the instruction of the calling routine. More accurately, the position from which the point is taken is variable (NPOS), but is usually the top of the Buffer. OUTP writes out z-values when required, and updates the counts of data output to the clean DDF file. OUTP calls the TOKEN and TRANS routines and is called by the ACT, INIT, MAIN, and TERM routines.

The OUTP routine resets multiple feature code values to zero and checks for conditions (filling initially) where no output is actually performed. The count (ICNT) for the current token is incremented, and the point to be written is translated by TRANS. The point is written with z-values (if indicated) and the output count and other counts are also incremented.

The major variables used by the OUTP routine are:

Variable Name	Type/Size	Use
XY, IBSIZ	See ACT above	
NPOS	INTEGER*4	The position from which data is to be written to the clean DDF file

REPORT (MHEAD)

The REPORT routine is executed after editing is complete on the DDF data. REPORT lists the various counts kept during syntax editing, and reports counts of input and output records; header, threshold and offset values; counts of token types found; counts of syntax errors, deletions, and insertions; and counts of each type of action taken by ACT. REPORT is called by MAIN.

The major variables used by the REPORT routine are:

Variable Name	Type/Size	Use
MHEAD		See MAIN above

SHIFT (XY, IBSIZ, N1, N2)

The SHIFT routine is used to shift the contents of the Point Buffer to make room for a new point in the bottom of the Buffer. Points are shifted up one location beginning at N1 and ending at N2. The shift is made by copying a point from one position into the position immediately above it. The shift may also be used to delete a point by shifting into its position without copying it or writing it to the output DDF file. SHIFT does not call other routines and is called by the ACT, INIT, MAIN and TOKEN routines.

The major variables used by the SHIFT routine are:

Variable Name	Type/Size	Use
XY, IBSIZ		See ACT above
N1	INTEGER*4	The upper position in the Buffer where shifting begins
N2	INTEGER*4	Lower position in the Buffer where shifting ends (last one to shift)

TERM (XY, IBSIZ, IPOS)

The TERM routine is called at the end of the input file to clear the remaining points from the Buffer and write them to the clean, output DDF file. Output begins at the top of the Buffer and continues to the position above the active position. TERM calls the OUTP routine and is called by the MAIN routine.

The major variables used by the TERM routine are:

Variable Name	Type/Size	Use
XY,IBSIZ,IPOS	See ACT above	
NEND	INTEGER*4	The last position in the Buffer from which data is output

TOKEN (XY, IBSIZ, IPOS, NTOKE)

The TOKEN routine examines the Point Buffer beginning at the **active position** (IPOS) to identify the kind of token which exists next in the input stream. This token is one of the seven token types listed above, and is used to determine the transition from the current state to the next and the actions to be taken. The TOKEN routine calls the INP and SHIFT routines and is called by the ACT, MAIN, and OUTP routines.

The TOKEN routine performs a series of tests on the values in the Buffer to eliminate possible tokens until a test is passed or until all possibilities have been eliminated. When the token is identified, the token number (NTOKE) is set and returned.

The major variables used by the TOKEN routine are:

Variable Name	Type/Size	Use
XY,IBSIZ,IPOS,NTOKE	See ACT above	
XD	REAL*4	Stores X-difference between points to check for nodes in Buffer
YD	REAL*4	Stores Y-difference for node check

TRANS (X, Y)

The TRANS routine performs the (optional) translation of the points from the Point Buffer prior to writing them to the output DDF file. Point locations are passed to TRANS and are translated according to the offsets kept in global variables (XOFF, YOFF). The TRANS routine is called by the ACT and OUTP routines.

The major variables used by the TRANS routine are:

Name	Type/Size	Use
X	REAL*4	X coordinate for the point in the Buffer to be translated
Y	REAL*4	Y coordinate to be translated

F. Example Run With Sample Data

This section presents a sample run of the SYNEDT program to display the form of the input data, the process and dialogue for operation of the SYNEDT program, the syntax errors found and the corrections made, and the resulting content of the "cleaned" DDF output file.

For this example, a small input DDF file was created from a legitimate DDF file from the digitizing process, by reducing the size of the file and introducing a few selected syntax errors. The resulting input DDF file is listed below. The various parts of the DDF file content and the syntax errors have been noted for easy reference.

Input DDF File Content:

```
WY1959B_D1 WW 17-MAY-88 MP 24000. PR1975    (Map Header)
    .999    .999    (Control Points SW)
  18.732   1.000      (SE)
  18.710  23.733      (NE)
   1.003  23.743      (NW)
   1.001   1.001      (SW)
   7777.0       .0    (Line Type Header, feature code=7777)
    .999   1.661      (Double point, Start Node)
    .999   1.661
   2.359   1.672      (Interior Points)
   3.458   1.674
   4.719   1.670
   5.219   1.670                (*Error* Should be a Double Point)
  -1.000  -1.000      (Discontinuity Flag)
   5.222   3.678                (*Error* Should be a Double Point)
   5.232   6.085      (Interior Points)
   5.236   7.805
   5.240   9.317
   5.240   9.317
  -1.000  -1.000                  (*Error* Should be Header Flag)
   5555.0       .0    (Line Header, code = 5555)
  16.130  12.164      (Start Node)
  16.130  12.164
  17.900  10.116      (End Node, no interior points)
  17.900  10.116
  -1.000  -1.000      (Header Flag)
  -1.000  -1.000
  17.935  10.089                          (*Error* Missing Header)
  17.935  10.089      (Start Node)
  18.071   9.993
  18.307   9.847
  18.644   9.667                            (*Error* Triple Point)
  18.644   9.667
```

```
18.644   9.667
18.686   9.652
18.714   9.643
18.721   9.638
18.729   9.636      (End Node)
18.729   9.636
-1.000  -1.000
-1.000  -1.000
                    (End of file)
```

The following dialogue is not a verbatim report created by the SYNEDT program, but it does list all the interaction between the operator and the program in nearly verbatim fashion. This listing reflects the actions of both the operator and the SYNEDT program in processing the sample input file listed above. The responses of the operator are underlined. **Comments are in bold.**

Sample run of SYNEDT:

```
Enter the Name of the Output DDF file?   SAMPLE.OUT

Enter the Name of the Input DDF file?  SAMPLE.IN

The Map Header for the Input file is:

          WY1959B_D1 WW 17-MAY-88 MP 24000. PR1975

Your options are      1 = Continue
                      0 = Change Header
                     -1 = Stop

Enter option?  1

Node Recognition Threshold =  .0025

Enter  1 to accept,  0 to reject?  1  (If 0, reset Threshold)

X offset = 0.000

     Enter  1 to accept,  0 to reject?  1   (Reset if 0)

Y offset = 0.000

     Enter  1 to accept,  0 to reject?  1   (Reset if 0)

Enter default Z value?  0.   (No Z-values used here)

-   -   -   -   -   -   -   -   -   Syntax Editing Begins
```

```
LINE HEADER, feature code =  7777

Arc 1   X = .999   Y = 1.661    (Starting node location)

***  2  Insertions   0  Deletions   2  Errors

     (Insertions are data added, deletions are data removed -

          Errors are total number of syntax errors corrected)

Arc 2   X = 5.222   Y = 3.678

LINE HEADER, feature code = 5555

Arc 3   X = 16.130   Y = 12.164
*** Arc Header Missing,  Code 15555 Inserted

LINE HEADER, feature code = 15555

Arc 4  X = 17.935   Y = 10.089

*** TRIPLICATE POINT DELETION    (3 identical consecutive points)

***  0  Insertions   1  Deletion   1  Error

Arc  5  X = 18.644    Y = 9.667

-    -    -    -    -    -    -    -    - Syntax Editing Complete

SYNDET REPORT

HEADER =
          WY1959B_D1 WW 17-MAY-88 MP 24000. PR1975        ...C
THRESHOLD  =  0.0025

X OFFSET  =  0.000

Y OFFSET  =  0.000

INPUT COUNT  =  35    OUTPUT COUNT  =  38    (data points in/out)

NO. ARCS  =  5    (number of arcs found)

NO. NODES  =  9    (number of nodes found)

NO. INT. PTS.  =  10    (number of interior points found)

NO. SINGLE PT. ARCS  =  0    (number of isolated point arcs)

NO. TYPE 1 HEADERS  =  0  (number of isolated point headers)
```

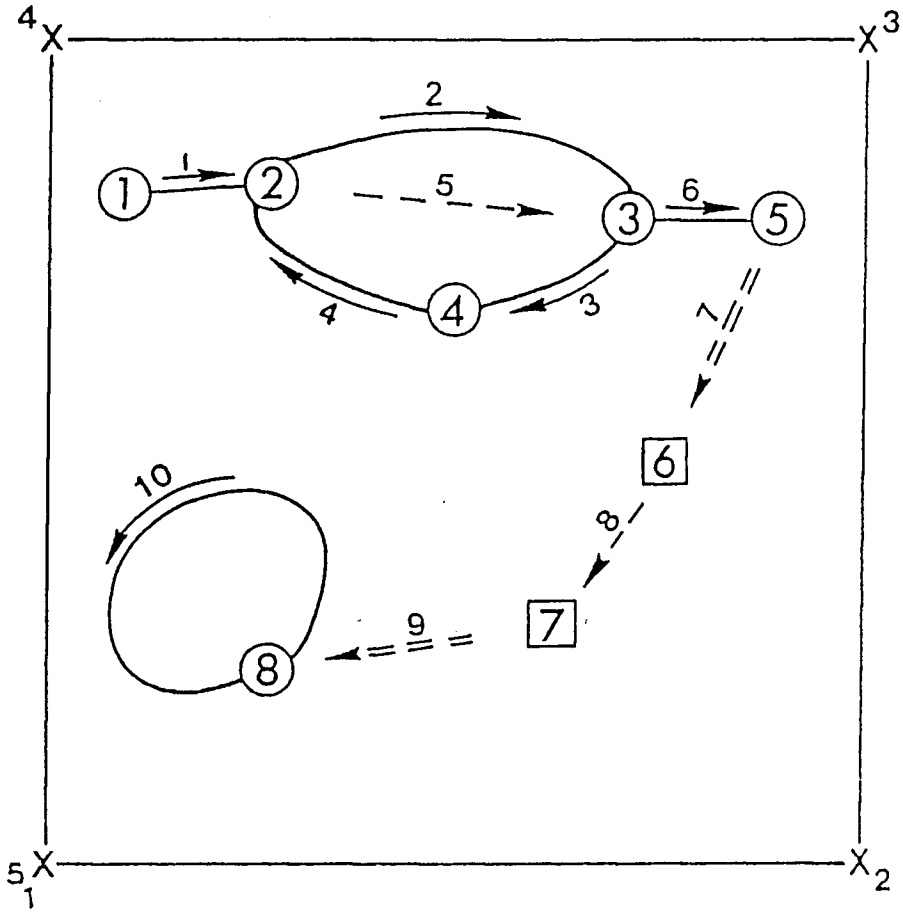
```
NO. TYPE 2 HEADERS  =  3    (number of linear arc headers)

**  TOTAL ERROR COUNTS  **

     ERRORS  =  5    (number of syntax errors)

     DELETIONS  =  1

     INSERTIONS  =  4

**  TOTAL ACTION COUNTS  **    (counts of SYNEDT actions)

     OI = 34

     CP = 3

     DC = 0

     RP = 5

     RH = 3

     IFH = 0

     DP = 0

     IH = 1

*EOR*                          (END OF REPORT,  STOP)
```

The output from the example run of the SYNEDT program (shown in dialogue form above), based on the use of the input DDF file listed above is shown here. The errors noted on the input DDF file above have been corrected, and the corrections are noted. Note that the characters "...C" have been appended to the header to indicate that data in the file has been "cleaned".

Sample Output DDF File:

```
WY1959B_D1 WW 17-MAY-88 MP 24000. PR1975                    ...C
    .999   .999
  18.732  1.000
  18.710 23.733
   1.003 23.743
   1.001  1.001
  7777.0      .0
    .999  1.661
    .999  1.661
```

```
 2.359  1.672
 3.458  1.674
 4.719  1.670
 5.219  1.670      (End Node Duplicated)
 5.219  1.670
-1.000 -1.000
 5.222  3.678
 5.222  3.678      (Start Node Duplicated)
 5.232  6.085
 5.236  7.805
 5.240  9.317
 5.240  9.317
-1.000 -1.000      (Discontinuity changed to Header Flag)
-1.000 -1.000
 5555.0      .0
16.130 12.164
16.130 12.164
17.900 10.116
17.900 10.116
-1.000 -1.000
-1.000 -1.000
15555.0      .0    (Header Inserted with code = 15555)
17.935 10.089
17.935 10.089
18.071  9.993
18.307  9.847
18.644  9.667
18.644  9.667      (Triplicate Point reduced to Continue Node)
18.686  9.652
18.714  9.643
18.721  9.638
18.729  9.636
18.729  9.636
-1.000 -1.000
-1.000 -1.000
```



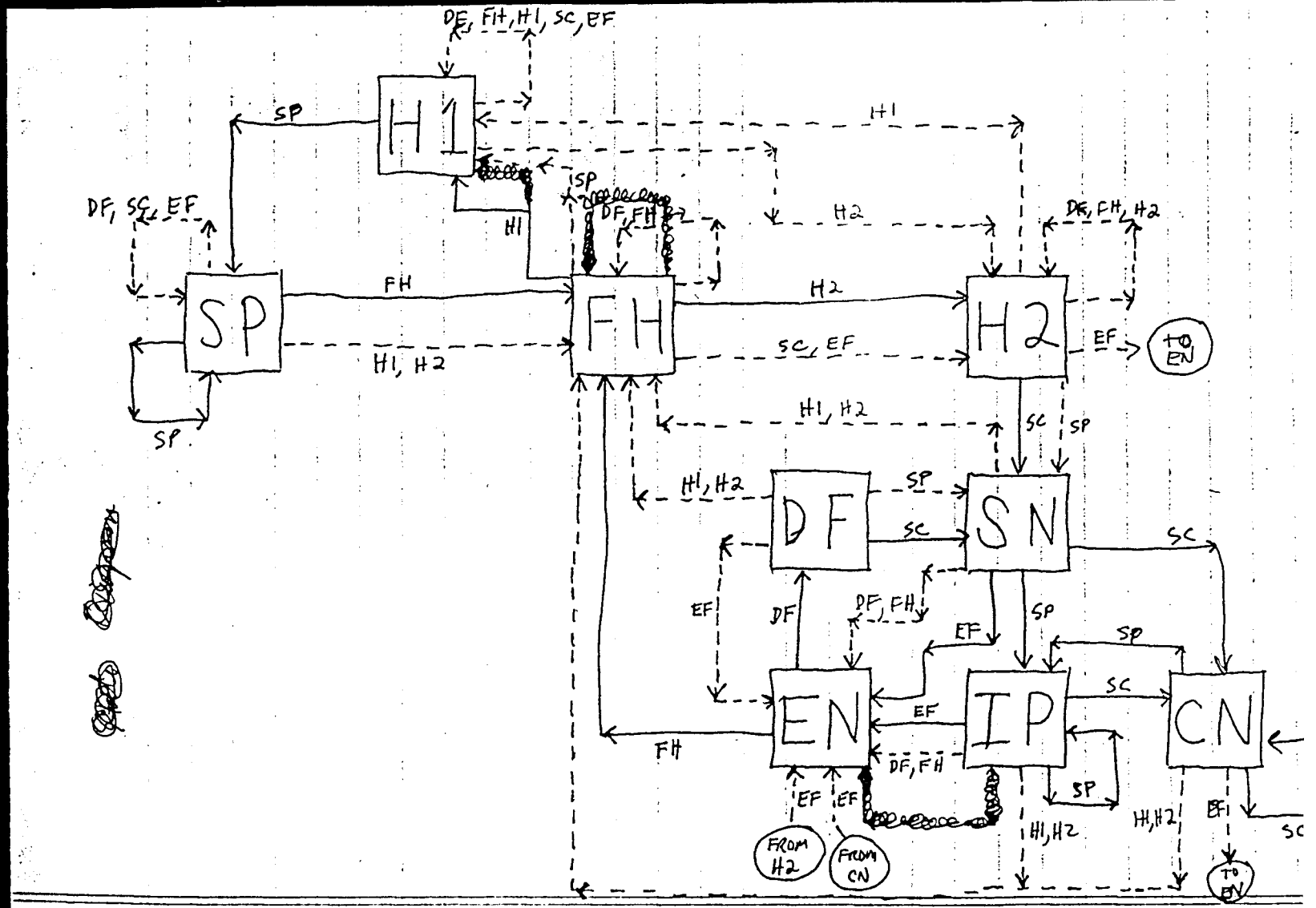
Symbology

(n)	Node
[n]	Isolated point
~	Arc
X	Control points
——→	Digitizing sequence
- - -→	Discontinuity
= = =→	Feature code change

STATES \ EVENTS	DF	FH	PH	AH	SC	EF	SP
PH	-3 / H1	-33 / H1	-57 / H1	-57 / H2	-3 / H1	-3 / H1	41 / SP
AH	-3 / H2	-33 / H2	-57 / H1	-57 / H2	411 / SN	-42211 / EN	-421 / SN
SN	-22 / EN	-22 / EN	-622 / FH	-622 / FH	411 / CN	11 / EN	1 / IP
CN	0 / —	0 / —	-6 / FH	-6 / FH	411 / CN	11 / EN	1 / IP
EN	1 / DF	11 / FH	0 / —	0 / —	0 / —	0 / —	0 / —
IP	-2 / EN	-2 / EN	-62 / FH	-62 / FH	411 / CN	11 / EN	1 / IP
SP	-3 / SP	11 / FH	-6 / FH	-6 / FH	-3 / SP	-3 / SP	41 / SP
DF	0 / —	0 / —	-2 / FH	-2 / FH	411 / SN	-42211 / EN	-421 / SN
HF	-3 / FH	-33 / FH	51 / H1	51 / H2	-58 / H2	-58 / H2	-58 / H1

Transition Matrix For SYNEDT

ACTION / NEXT STATE



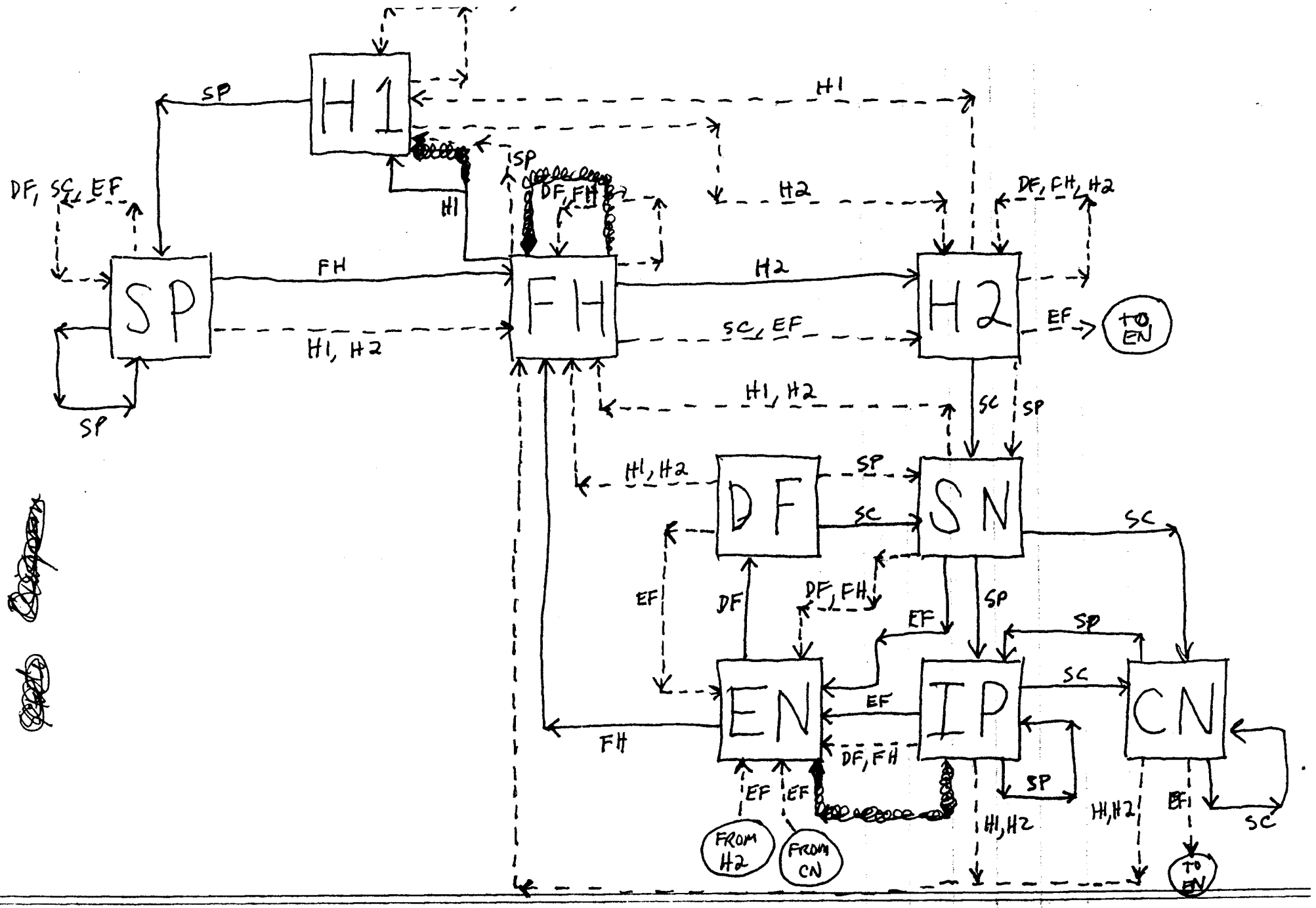
State Diagram For SYNEDT

Solid = Normal Transiti
Dashed = Error Transit

STATES	DF	FH	PH	AH	SC	EF	SP
PH	-3 / H1	-33 / H1	-57 / H1	-57 / H2	-3 / H1	-3 / H1	41 / SP
AH	-3 / H2	-33 / H2	-57 / H1	-57 / H2	411 / SN	-42211 / EN	-421 / SN
SN	-22 / EN	-22 / EN	-622 / FH	-622 / FH	411 / CN	11 / EN	1 / IP
CN	0 / —	0 / —	-6 / FH	-6 / FH	411 / CN	11 / EN	1 / IP
EN	1 / DF	11 / FH	0 / —	0 / —	0 / —	0 / —	0 / —
IP	-2 / EN	-2 / EN	-62 / FH	-62 / FH	411 / CN	11 / EN	1 / IP
SP	-3 / SP	11 / FH	-6 / FH	-6 / FH	-3 / SP	-3 / SP	41 / SP
DF	0 / —	0 / —	-2 / FH	-2 / FH	411 / SN	-42211 / EN	-421 / SN
HF	-3 / FH	-33 / FH	51 / H1	51 / H2	-58 / H2	-58 / H2	-58 / H1

Transition Matrix For SYNEDT

ACTION / NEXT STATE



State Diagram For SYNEDT

Solid = Normal Transition
Dashed = Error Transition

II. B A S I C S E Q U E N C E

II.3 THE PARGEN PROGRAM

A. Introduction

The **PARGEN**, (for point-arc generation), program converts syntactically-clean data from a DDF file which is output from the SYNEDT program and creates the direct-access Arc, Point and Qplot files for the cartographic database.

The PARGEN program interprets the sequence of points in the clean DDF file according to the rules of the digitizing syntax to construct arc records for each isolated point feature and each linear arc. The interior points of the linear arcs are placed in linked records in the Point file with a value in the associated arc record to point to the chain.

The input file is expected to be clean according to the syntax, and any syntax errors found in the file will cause PARGEN to halt operation. Similarly, all points in the DDF file are expected to lie in the designated projection area. Points which do not are brought to the nearest edge or corner of the area.

Upon completion of the creation of the Point and Arc files, PARGEN creates a display file called the **QPLOT** file in parallel to the Point file. The Qplot file consists of point coordinates in a ready-to-display form for the unique view of the complete map area at its largest scale.

The PARGEN program may be operated in either the **initialize** or the **reentrant** mode. The initialize mode will cause any data residing in the Arc and Point files to be deleted and only the new data in the input file to be entered there. The reentrant mode allows new data in the input DDF file to be added to data already existing in the database.

B. General Information

The main objective of the PARGEN program is to accept clean DDF data from the input file (which has been produced as output by the SYNEDT program); to interpret the data stream according to the rules of the digitizing syntax; to recognize arcs, endpoints and interior points; and to construct the Point, Arc and Qplot files to model the original map data.

A second major objective of the PARGEN program is to correct the map data for errors in the physical form of the source map. This means that errors due to shrinking, stretching or wrinkling of paper maps prior to digitizing must be compensated for to some degree. This can be accomplished by relying on the mathematical

model which describes where each point on the curved surface of the earth should be placed on the flat, paper map.

This 'mapping' from the earth's surface to a flat map is called the **projection** of the map. Many different kinds of map projections may be used to produce different kinds of desired qualities (e.g. true scale, true angles and shapes...). The value of this information is that the projection type and the values (parameters) used to specify the projection define where all points should be found.

Thus, given points on a map which are of known locations on the surface of the earth, the projection information can predict where these points should be on the map. The predictions can then be compared to the actual locations on the map to obtain an approximation of the distortion errors, assuming them to be linear accross the map.

The standard format of a DDF file in GIMMAP includes the five digitized **control points** at the beginning of the file. These points are the digitized locations of four points of known location (in geographic or longitude and latitude coordinates) on the earth, usually the four corners of the map (with the first point digitized twice as a test for consistency).

The comparison of the control points to those predicted by the map projection produces values for the translation (movement along a constant distance in X or Y), scaling (multiplication of coordinates by constants), and rotation (movement along circular arcs about the origin) of map data to properly correct map data for distortion errors in the paper map. These values are then applied to all data entered from the paper map in that digitizing session.

The process of projection converts map data in geographic coordinates to projected or cartesian coordinates based on the projection chosen and the parameters defining its use. There is no limit to the number of different projections, but there are basic classes of projections. Among these are the **conic** map projections which are based on the construction of one or more cones, intersecting the earth or tangent to its surface at user-selected locations. Often the selected cone crosses the surface at lines of latitude known as **standard parallels.**

The GIMMAP system projection library **PROMAN** supports two conic projections, the **Modified Polyconic** projection and the **Lambert Conformal Conic** projection. The Modified Polyconic projection is a special conic projection in that it assumes a different (tangent) cone for each latitude, thus using infinitely many cones for any area. The Lambert Conformat Conic assumes a single cone intersecting at two standard parallels. Of course, GIMMAP can also treat a map as though it has no projection at all, even though it does.

Beyond selection of the type of projection, several other

parameters must also be specified to fix the transformation to cartesian (x,y) coordinates. First among these is the model of the earth which whose surface is to be transformed. As in most things, there are many different "standards" which are used. The model adopted by GIMMAP (and presumably that used by the USGS in their 7.5' quad maps and others) is the Clarke 1866 model. Exact values for the axes in that model may be found in **Richardus** (P., and Adler, R. K., (1972),Map Projections, North-Holland Publ. Co., Amsterdam, 174 p.)

The **map scale** is expressed on most maps by a scale bar or as 1:**number** where **number** represents the **scale factor.** The map scale represents the ratio between a distance (between two objects) on the map and the corresponding distance on the surface of the earth (between the corresponding objects). For example, if one inch on the map represents a mile on the surface of the earth, then the map scale is 1/(1 x 5,280 x 12) = 1/63,360 written on a map as 1:63,360 and spoken as "one to 63,360". Here, the scale factor is 63,360.

When speaking of map scales, some refer to "large-scale" and "small-scale" maps. Here, the terms "large" and "small" refer to the fraction which is the ratio described above, not to the scale factor itself. Thus, a large-scale map (large fraction) refers to one with a small scale factor (e.g. scale = 1:1,000). Such a map requires large map size to map small areas. Hence, a large-scale map is a close-up view of a small area on earth. A small-scale map (e.g. scale = 1:1,000,000) is a distant view of a large area on the earth.

Given the projection type, the earth model and the map scale factor (often referred to as the map scale), only the **projection area** must be specified to completely fix the transformation by the map projection. The projection area is a rectangle described in geographic coordinates (longitude and latitude) which contains the entire area to be mapped. It is described by the parameters: **south, north, west, east** in degrees.

The rectangle defined by these values defines the location of the control points and is used to locate the lat/long **frame** selected at plot time. This rectangle should completely contain all map data in the are and be centered on the map area as well as possible in its initial definition. For the projection, it is a bounding rectangle, but it may serve many purposes such as a clip window for plot files.

When the standard control points defined as the corners of the projection can not be digitized (they may not all appear on the map), then any four rectangular points of known geographic coordinates may be used. In this case, the **non-standard control points** option is used. The selected points are digitized and the locations in longitude and latitude are appended to the digitized (x,y) locations.

When processing the clean DDF input files, PARGEN assumes and follows the rules of the syntax as defined above. Feature codes follow header flags, nodes precede and follow discontinuity flags, all the rules of the syntax are expected. There are two special features which are accepted by SYNEDT. Isolated points may have up to seven **z-values** associated with the location of each point. This is signalled by the y-value of the feature code header for the isolated points. If greater than zero, then all points until the next header must have z-values.

It is also possible to associate a **second feature code** with a set of linear arcs. This is done by setting the y-value in the feature code header for the arcs to a value greater than 1999. If this is done, then the arcs which follow this header up to the next feature code header will all be duplicated with one arc using the feature code in the x-value and a second arc using the feature code in the y-value.

The **node recognition threshold** is the distance at which two consecutive points in the DDF file are considered to be identical and thus represent an arc endpoint or a node. This value may be selected by the operator and is dependent on the resolution and the accuracy of the digitizing equipment. The value used in operations at the Kansas Geologicval Survey to construct the Kansas Cartographic Database is 0.0025 inches working on a table with very good resolution. Nodes in this system are made to be identical by the digitizing software.

C. Operation

The operation of PARGEN requires a short interaction between the computer and the operator to set-up initial values, specify the projection information and to identify files. The majority of time spent by the program is to input clean data from the DDF file and to create the Arc, Point and Node files. In creating these files, PARGEN also creates the first and second **information records** in the Point file containing the map header and values defining the projection, control points and extremes (see above, I.5 Digital Representation of GIMMAP Data).

C(omputer): DIGITIZED DATA FILENAME?

O(perator): Operator enters the DDF file name

C: ARC FILENAME?

O: Operator enters the Arc file name

C: POINT FILENAME?

O: Operator enters the Point file name

C: QUICK PLOT FILENAME?
O: Operator enters the Qplot file name

C: PARGEN displays the map header from the DDF file...
Input Header = <map header>, Enter 1 to continue?

O: If the header is correct, enter 1. Enter 0 to stop.

C: PARGEN reads the control points from the DDF file...
Node Recognition Threshold = <default>, Enter 1 to accept?

[Begin option]

O: Operator accepts threshold by entering 1

OR

O: operator rejects threshold, entering 0

C: Enter new threshold value?

O: operator enters new threshold value

C: cycle above to 'Node Recog...Enter 1 to accept?'

[End Option]

C: Projection possibilities are:

0 = No Projection
1 = Modified Polyconic
2 = Lambert Conformal Conic

Enter projection type?

O: Operator enters selection for projection type. If map is not a 7.5' quad then use -1 or -2

C: Enter the SCALE factor?

O: Operator enters the scale factor, as in 1:scale factor

C: [If 7.5' quad] Enter row, column numbers?

O: Operator enters unique row and column numbers for the quad map from the (artificial) state grid

C: Select Re-entry (0) or Initialization (1)?

O: Operator enters 0 to append new data to old

OR

Operator enters 1 to erase old and enter only new data

C: Your choices are:

Map projection = <your choice>

Scale Factor = <your choice>

Row = <your choice> Column = <your choice>

** Re-entry option selected **

OR

** Initialization option selected **

Enter 1 to accept or 2 to retry?

O: Enter 1 if selections as desired or 0 to re-select

I [Begin Initialization Option]
I
I C: ** Point File Initialization **
I PARGEN reports (through the RAFMAN library) statistics
I about the Point file and suggests a maximum size to
I initialize the file to (300)
I Enter new max. file size?
I
I **O: Operator sets the size of the Point file in records**
I
I C: File code 2 initialized to <max> records with length 68
I
I ** Arc File Initialization **
I PARGEN reports stats for the Arc file and suggests a
I maximum size for it (200)
I Enter new max. file size?
I
I **O: Operator sets the size of the Arc file in records**
I
I C: Please enter South, North latitudes in degrees?
I
I **O: Operator enters south and north boundaries of the**
I **projection area in decimal degrees latitude**
I
I C: Please enter West, East longitudes in degrees?
I
I **O: Operator enters west, east boundaries of area as**
I **positive or negative degrees longitude**
I
I [Option - if Lambert projection selected]
I
I C: Please enter south north standard parallels in degrees?
I
I **O: Operator enters standard parallels of projection**

I
I [End of Lambert Option]
I
I [End of Initialization Option]

R [Begin Re-entry Option]
R
R C: PARGEN compares projection, scale and row/column
R with values in the database, using old when different
R
R Enter 1 if control points are non-standard?
R
R **O: Operator enters 0 if corners of projection area used**
R **for control points or 1 if they were not**
R
R [Begin option for non-standard control points]
R
R C: For the non-standard control points...
R Please enter South, North latitudes in degrees?
R
R **O: Operator enters the bottom and top of the rectangle**
R **which locates the non-standard control points**
R
R C: Please enter the West, East longitudes in degrees?
R
R **O: Operator enters west and east as positive or negative**
R
R [End of non-standard control point option]
R
R [End of option for Re-entry]

C: Operator input is now complete. PARGEN now processes the input file according to the syntax to create the Point and Arc files. When input data is complete, the Qplot file is created as a display image of the Point file and a reported is printed detailing the results of PARGEN, some values stored in the information records of the Point file and the translation, scaling and angle of rotation values applied to the data.

D. Programmer's Overview

1. Introduction to Routines

All routines which comprise the PARGEN program are written in the FORTRAN77 programming language. The routine names and the basic function of each are listed in alphabetical order:

ADDBUF - Stores and manages one record of interior points

BUFCLS - Fills and empties buffer for last point record

CREATQ - Creates Qplot file parallel to Point file

FILOPN - Opens terminal I/O, DDF and database files

MAIN - Initial values, operator dialogue, PAG and report

NTHSTH - Sets standard parallels for 7.5' quads in Kansas

PAG - Point/Arc Generation from flean DDF data

PUTARC - Creates and writes (line) arc records with report

PUTISO - Creates and writes arc records for isolated points

SETT - Set parameters for translation, rotation, scaling

TRANS - Use parameters to fit new points to projection area

2. Global Storage and Variables

The global variables of the PARGEN program are listed below according to the COMMON areas in which they are declared. Each COMMON area is listed with the name bounded by slash lines as it is in the program. Each variable is described with pertinent information.

/BUFCOM/

BUFF(16) - buffer of 8 pairs of interior point coordinates

NBUF - fixes the maximum number of entries in BUFF (16)

NXTBUF - next available location for filling in BUFF

XSAV - x-coordinate of first point after BUFF is full

YSAV - y-coordinate of first point after BUFF is full

NXTREC - next "to-be-used" record in the Point file

/FC/

IAFC - unit number (file code) of the Arc file

IDFC - unit number of the DDF input file

IIFC - unit number of the terminal input file

IOFC - unit number of the terminal output file

IPFC - unit number of the Point file

IQFC - unit number of the Qplot file

/MAP/

SOUTH - southern edge of projection area (originally in degrees north latitude)

NORTH - northern edge of projection area

DELPHI - height of projection area (north-south)

CLON - center of longitude of map area

EAST - eastern edge of projection area (originally in degrees west longitude)

WEST - western edge of projection area

SPS - south standard parallel for Lambert projection (originally in degrees north latitude)

SPN - north standard parallel for Lambert projection

XB,YB,BASE - used by PROMAN (projection library) to define cones for the Lambert and Polyconic projections

SCALE - the map scale factor, map scale = 1:scale factor

IPRO4 - projection type (0=none, 1=Polyconic, 2=Lambert

/MAXCOM/

MAXROW(61) - Used to determine which zone (north or south) a Kansas quad is in to set the standard parallels for the Lambert projection. Each entry represents a column in the fixed grid of rows and columns for the state. Each value is the maximum row number for quads in the column to be in the south zone.

/MC/

NUMFC - number of multiple feature codes for a group

MFC(4) - storage for up to 4 multiple feature codes

/MM/

XMIN - minimum x-coordinate for data in the database

YMIN - minimum y-coordinate for map data

XMAX - maximum x-coordinate for map data

YMAX - maximum y-coordinate for map data

/TRAN/

CPTSN(10) - control points from the incoming DDF file (order: xsw,ysw,xse,yse,xne,yne,xnw,ynw,xsw2,ysw2)

XSCALE - scaling factor applied to x-coordinates

YSCALE - scaling factor for y-coordinates

SINT - sine of rotation angle

COST - cosine of rotation angle

CORN(9) - projected corner points of map area (order as above for 1-8, 9 is minimum y)

3. Files Accessed By PARGEN

The PARGEN program accesses six files. These are the input and output files for the interactive terminal; the input file of (syntactically) clean DDF data; and the Arc, Point, and Qplot files of the map database.

Unit Number	File Name	Characteristics
IIFC (11)	Terminal In	Input from terminal, padded, sequential
IOFC (10)	Terminal Out	Output to terminal, sequential with FORTRAN carriage control
IDFC (12)	Clean DDF Input	Sequential, padded, input, maximum record length = 124B
IAFC (1)	Arc file	Direct access, in/out, 42B
IPFC (2)	Point file	Direct access, in/out, 68B
IQFC (3)	Qplot file	Direct access, in/out, 36B

4. Memory and Other Requirements

The PARGEN program has no memory or other requirements which are noteworthy. Incoming DDF data enters as a stream of points which are directed into interior point records in the Point file for the most part. At any time, at most one arc and one record of points is kept in memory along with information on the map projection and rectification of incoming points. Statistics are kept on many aspects of the previous as well as current run, but no large memory requirements are involved. The PARGEN program is involved in substantial amounts of disk I/O in getting the input DDF information and setting up or extending the database files.

E. PARGEN Routines in Detail

The routines for PARGEN are listed above. In this section, they are developed with some general information including the argument list, a description of the functions of the routine, a listing of other routines called, a list of routines which call it, an overview of the routine flow, and a list of major local variables with their type and an explanation of the use of the variable and other pertinent information.

ADDBUF (XIN, YIN, NUMERR)

THE ADDBUF routine adds interior points of an arc to an eight point buffer (BUFF) until the buffer is full or until there are no more points for the arc. When the buffer is full, a record is gotten from the freelist of the Point file and the buffer content is written to the record which is linked to the previous interior point records for the arc via the NEXT pointers.

When the buffer is full, the next point sent to ADDBUF is stored in a special save location (XSAV,YSAV), and the buffer is not immediately stored in the Point file. This saves disk I/O in the case that the extra point is the end node (not determined until the next point in the file is found to be identical) since that point need not be 'erased' from an already written Point file record. When the subsequent point is not identical to the saved point, the record is then written to the file.

New points to be added to the buffer are sent as (XIN, YIN) and are put in the next available location (NXTBUF). The number of points allowed in the buffer is limited by NBUF, initially set to 16 (maximum number of entries) for 8 points. The ADDBUF routine also maintains the pointer which marks the start record in the Point file for creation of parallel records in the Qplot file. This value is the smallest record number gotten for writing a new Point file record.

The major variables used by the ADDBUF routine are:

Variable Name	Type/Size	Use
XIN,YIN	REAL*4	Point sent to be added to the point buffer
X,Y	REAL*4	Internal copy of new point
BUFF(16)	REAL*4	Buffer for 8 pairs of points (order: X1,Y1,X2,Y2...X8,Y8)
NBUF	INTEGER*4	Maximum entries in BUFF (16)
NXTBUF	INTEGER*4	Next available BUFF position
XSAV,YSAV	REAL*4	First point after BUFF full
IPFC	INTEGER*2	Unit number of Point file
IQSTRT	INTEGER*4	Start record for creation of parallel Qplot records

BUFCLS (IPFPR)

The BUFCLS routine closes the point buffer when the end node for an arc has been identified. Since the last point added to the buffer was the first of two duplicate points signifying the end node, it must be removed from the buffer before the last point record is written. If it was the first interior point, the arc will have none so its point record is returned and IPFPR set to zero. All unused positions in the buffer are set to (-1,-1) to indicate they are not used in the point record, and the record is written to the Point file with NEXT equal to zero.

The major variables used by the BUFCLS routine are:

Variable Name	Type/Size	Use
BUFF,NXTBUF,NBUF, XSAV,YSAV	See ADDBUF above	
IPFPR	INTEGER*4	Pointer to the First Point Record of interior points for an arc
IZERO	INTEGER*4	Zero value for NEXT pointer of the last Point file record for an arc

CREATQ (no arguments)

The CREATQ routine creates records of point coordinates

expressed in the coordinate system used directly for the graphic display screen by programs such as GRFEDT and ZONEDT. Each record in the Point file represents a string of points along the path of an arc. These locations are representative of projected coordinates as found in the source map in digitizing. The CREATQ routine converts these coordinates into the **screen coordinates** of the points as displayed in a maximal view of the map (so that the entire map area is visible, shown as large as possible on the display screen).

Given the starting record number (IQSTRT) in the Point file, all subsequent Point records are read, their points converted to screen coordinates, and the result written to the parallel record in the Qplot (**Quickplot**) file. Unused points (-1,-1) in a record are passed along as unused (-1,-1) in the Qplot records.

Conversion from projected coordinates to screen coordinates involves a translation from (1,1), the lower-left corner of the projected area rectangle to (290,50), the desired lower-left corner for the graphics display screen. Conversion also requires multiplying (translated) coordinates by a scaling factor. This factor is the ratio of the range of screen coordinates for the area of map display to the range of projected coordinates for the map in the larger of the X or Y dimensions. This conversion produces the desired display with no distortion of the image.

The major variables used by the CREATQ routine are:

Variable Name	Type/Size	Use
IQSTRT,BUFF	see above in ADDBUF	
IACCR	INTEGER*4	Number of Accounting record used by RAFMAN etc.
IOFX,IOFY	INTEGER*2	Screen coordinates of lower-left corner of display area
RANGE	REAL*4	Projected range of map area (maximum of X and Y ranges)
XMINW,YMINW	REAL*4	Southwest corner in projected coordinates based on RANGE centered on old center of area
SRANG	REAL*4	Ratio of screen to projection coordinate ranges
NEXT	INTEGER*4	Pointer to next Point record (= 0 for last of an arc)
IBUF(16)	INTEGER*2	Buffer for Qplot record values

FILOPN (no arguments)

The FILOPN routine opens all files accessed by the PARGEN program. The terminal input and output files are opened without prompting and then the DDF and database file names are requested, given as each is opened by FILOPN. The FILOPN routine is called

only by the MAIN routine and calls no other routines in PARGEN.

The major variables used by the FILOPN routine are:

Variable Name	Type/Size	Use
IO	CHARACTER*60	Stores file names
IAFC,IDFC,IIFC, IOFC,IPFC,IQFC	INTEGER*2	Unit numbers for Arc, DDF, Terminal Input/Output, Point, and Qplot files

MAIN

The MAIN routine is responsible for setting up the initial values of variables, including the two basic options and the parameters for the map projection, calling the PAG routine to perform the basic functions of PARGEN, and producing a report on the statistics of the previous and current run.

MAIN first calls the RMIOFC routine to initialize the GRFMAN graphics library and the FILOPN routine to have all files opened. The header from the input DDF file is read and listed to be checked by the operator as verification of the correct choice for the input file. The new control points (CPTSN) are read from the input DDF file, and the node recognition threshold (THRESH) is set by the operator.

The MAIN routine then obtains the projection type, scale and row and column numbers from the operator. The operator selects either the initialization or re-entry option. The initialization option assumes that no data is in the database or that all data in the database is to be deleted. Only the new data in the input DDF file is to be in the database. Re-entry merely appends the new data to that which already exists in the database.

The initialization option requires a complete specification of the projection parameters and selection of the sizes of the Arc and Point files in number of records. The re-entry option reports differences if the projection, scale or row and column values differ from those in the database. If so, those in the database are used.

If non-standard control points are used then the geographic coordinates defining the rectangle are obtained. The translation rotation and scaling values are set up in a call to the SETT routine, and all is ready for the processing of the input data. The PAG (Point-Arc Generation) routine is called to convert the input DDF data into the Arc and Point files.

When DDF processing is complete, the information records in the Point file are updated and a report is generated for the

previous run (if not initializing) and for the current run. The report includes headers, numbers of arcs and numbers of point records, projection information and map data extremes (minimum and maximum X and Y coordinates). Finally, the CREATQ routine is called to create Qplot records parallel to the Point file.

The major variables used by the MAIN routine are:

Variable Name	Type/Size	Use
CPTSN(10)	REAL*4	Control points for input DDF (order: XSW,YSW,XSE,YSE,...)
THRESH	REAL*4	Node recognition threshold
NPRO,IPRO IPRO4	INTEGER*2 INTEGER*4	Projection (0=none,1=MP,2=LC)
SKAL,SKALO SCALE	REAL*4 REAL*8 (DOUBLE PRECISION)	Map scale (DDF file, database) (in Projection library PROMAN)
NROW,NCOL NRO,NCO	INTEGER*2	Kansas quad row and column
NHEAD,OHEAD	CHARACTER*64	Map header (DDF file,database)
INIT	INTEGER*4	Initialization(1)/Re-entry(0)
SOUTH,NORTH, WEST,EAST	REAL*8 (DOUBLE PRECISION)	Edges of projection rectangle in degrees latitude/longitude
SPS,SPN	REAL*8 (DOUBLE PRECISION)	Standard parallells south and north in degrees latitude
XMIN,YMIN, XMAX,YMAX XMINO,YMINO, XMAXO,YMAXO	REAL*4	Map data extremes (DDF file) (and database)
SOUTHNS,NORTHNS, WESTNS,EASTNS	REAL*8	Rectangle for non-standard control points in degrees latitude and longitude
KBND	INTEGER*2	Feature code defining the map edge, used for moving points from the outside to the edge

NTHSTH (NROW, NCOL)

The NTHSTH routine is called to set the values of the north and south standard parallels for 7.5′ quad maps in the state of Kansas, given the row and column number of the map in the Kansas

grid of quad maps. In Kansas, there is a north zone and a south zone defined by an irregular line which follows county boundaries to divide the state approximately in half. Quad maps which lie on or below this boundary are in the south zone; those above it are in the north zone. All maps in the south zone use one set of standard parallels; those in the north use another.

The major variables used by the NTHSTH routine are:

Variable Name	Type/Size	Use
NROW,NCOL	INTEGER*2	Row and column numbers of map in grid of Kansas quads
SS,SN	REAL*8 (DOUBLE PRECISION)	South and north standard parallels set in degrees

PAG (THRESH, KBND)

The PAG (for point and arc generation) routine performs the interpretation of the input DDF file strictly according to the rules of the digitizing syntax to create the records of the Arc and Point files. The process of interpreting the syntax follows approximately the sequence below:

1. Read a point expected to have the feature code (this record may also be empty to signal the end of the file) Check for multiple feature codes and number of z-values Report the feature code from the header

2. If the feature code is for isolated points, read the next point, expecting an isolated point (or a flag to signal a new feature code header). For each isolated point, report the arc, create and write the arc record, and cycle to read the next point.

3. If the feature code signals linear arcs, then read the next point expecting duplicate coordinates for the start node. Report the arc, read any interior points to the next continue or end node. Empty the points to the Point file, report the arc and write the arc record to the Arc file.

4. If a flag was a discontinuity flag then read the first of the start node duplicates, then cycle back to number 3 to get the second half.

5. If the flag was a (double) header flag signalling a change in the feature code, go back to number 1.

6. If end-of-file, produce report and stop.

The report produced at the end of the file is presented in MAIN above, but the PAG routine may produce a small report in the case that a syntax error is encountered. This report includes a message stating the event, and the contents of the point buffer (BUFF) along with the next available location pointer for the point buffer. PAG also reports a location counter value which indicates the approximate location in the above sequence where the syntax error was found, and displays the next ten points in the input file to help locate the error there. Processing of the data is halted and PARGEN stops at this point.

The major variables used by the PAG routine are:

Variable Name	Type/Size	Use
THRESH	REAL*4	Node recognition threshold
KBND	INTEGER*2	Feature code of map boundary
LOC	INTEGER*2	General location of syntax error in PAG
KDARC	INTEGER*2	Feature code of input arc
NUMZ	INTEGER*4	Number of z-values
MAPEJ	INTEGER*4	Indicates if arc is on the edge or not (1=yes, 0=no)
XF,YF	REAL*4	First node for an arc
IAFC,IDFC,IPFC	INTEGER*4	Unit numbers (file codes) for the Arc, DDF, and Point files
NXTREC	INTEGER*4	Next pointer in Point record
XOLD,YOLD	REAL*4	Previous point location (used for duplicate point check)
XTR,YTR	REAL*4	Point location after transform
XL,YL	REAL*4	Last node for the arc

PUTARC (KOD, XF, YF, XL, YL, IPFPR, AL, XMN, YMN, XMX, YMX)

The PUTARC routine creates and writes the arc record for a new arc and reports the new arc to the output file for the PARGEN run. Using the RMGETR routine from the RAFMAN library, PUTARC removes an unused record from the freelist of the Arc file and writes the new arc record there. Included in the new arc record are the feature code, endpoint nodes, Point file pointer, the arc

length and the extremes of the arc (minimum/maximum X and Y). In the case where multiple feature codes are selected, the PUTARC routine creates extra copies of the arc record with the other feature codes.

The major variables used by the PUTARC routine are:

Variable Name	Type/Size	Use
KOD	INTEGER*2	Feature code of the new arc
XF,YF,XL,YL	REAL*4	First and last nodes
IPFPR	INTEGER*4	Point file pointer (record number of first Point record)
AL	REAL*4	Arc length in digitizer inches
XMIN,YMIN, XMAX,YMAX	REAL*4	Minimum and maximum X and Y coordinates for the arc
NUMA	INTEGER*4	Record number for new arc

PUTISO (KD, XF, YF, NUMZ, Z)

The PUTISO routine creates and writes a record in the Arc file for the new isolated point arc. Included in the record are the feature code, the location of the point and a maximum of seven z-values associated with the point. Unused z-values are set to -1.

The major variables used by the PUTISO routine are:

Variable Name	Type/Size	Use
KD	INTEGER*2	Feature code of the isolated point arc
XF,YF	REAL*4	Location of isolated point
NUMZ	INTEGER*4	Number of associated z-values
Z(7)	REAL*4	Associated z-values

SETT (no arguments)

The SETT routine sets the transformation paramaters to correct the incoming DDF data to fit the mathematical projection. Included in the transformation parameters are the translation in X and Y (thus forcing the southwest corner of the map to be at

(1.0,1.0), rotation about the southwest corner of the projection and scaling in X and Y coordinates. All the values used for the transformation are averages created by comparing the incoming control points to the known projection corners for the map area.

The translation parameters are calculated by averaging the first and fifth control points, where the fifth point is created by a second digitization of the southwest corner of the map. The averaged location is then compared to (1.0,1.0), the desired location of the southwest corner of the map. The difference is the translation vector for the incoming points.

The rotation parameters are calculated by using the slopes of lines from the first control point to the other three points to find the associated angles. These angles are compared to the corresponding angles in the projected corners of the area. The average of the difference angles determines the angle of rotation for all the incoming points.

The scaling parameters are found by conparing the lengths of the top/bottom and the right/left sides of the region defined by the control points with the corresponding lengths for the region defined by the projected points. The average ratios then provide the scale factors in X and in Y.

When all these values are found, the SETT routine reports them to the operator.

The major variables used by the SETT routine are:

Variable Name	Type/Size	Use
CPTSN(10)	REAL*4	Incoming control points
CPTS(8)	REAL*4	Control points with averaged first and fifth points
CORN(9)	REAL*8 (DOUBLE PRECISION)	Projected control points plus minimum Y for map area CORN(9)
XTRANS,YTRANS	REAL*4	X and Y translation values
XSCALE,YSCALE	REAL*4	X and Y scaling values
A1,A2,A3,A	REAL*8 (DOUBLE PRECISION)	Intermediate and final angles
SINT,COST	REAL*4	Rotation sine and cosine

TRANS (X, Y, NUMERR)

The TRANS routine performs the transformation which was set up in the SETT routine. All incoming points from the DDF file are sent to TRANS to have their locations corrected as specified above. In addition, the TRANS routine calls another routine in the projection library (called NORMAL) to cause the points to be deprojected (converted to longitude and latitude), so they can be compared to the projection area defined for the map. Points lying outside the map area are forced back to the nearest edge or corner. All points are then projected back to the cartesian or digitized coordinates. Points are only deprojected when the map has a projection. If no projection, the points are compared to a set of extremes for the map.

The major variables used by the TRANS routine are:

Variable Name	Type/Size	Use
X,Y	REAL*4	Point to be transformed
NUMERR	INTEGER*4	Used in the NORMAL routine for counting excessive errors in map projection
XT,YT	REAL*4	Distance from point to the southwest corner of the new control points
XR,YR	REAL*8 (DOUBLE PRECISION)	Point being reprojected
XMIN,YMIN, XMAX,YMAX	REAL*4	Minimum/maximum X and Y values for map without projection

F. Example Run With Sample Data

As before, the sample run presented in this section is designed to display the process and the dialogue of the PARGEN program and to show the form of the output listing in the form of reports and information presented by PARGEN. The form of the input data is precisely that of the output data from the SYNEDT program. In fact, the very output file from the SYNEDT example (SAMPLE.OUT) is used as the input file to the PARGEN program.

The form of the output files here is not easily shown, but is reflected in the information printed by the PARGEN program. Each arc record is reported in part, with values for the Point file pointers for each arc shown. This does not provide a complete listing of Point record contents, but it does give a general indication of the number of points for each arc.

Input DDF File Content (SAMPLE.OUT)

```
WY1959B_D1 WW 17-MAY-88 MP 24000. PR1975                    ...C
    .999   .999
  18.732  1.000
  18.710 23.733
   1.003 23.743
   1.001  1.001
   7777.0      .0
    .999  1.661
    .999  1.661
   2.359  1.672
   3.458  1.674
   4.719  1.670
   5.219  1.670
   5.219  1.670
  -1.000 -1.000
   5.222  3.678
   5.222  3.678
   5.232  6.085
   5.236  7.805
   5.240  9.317
   5.240  9.317
  -1.000 -1.000
  -1.000 -1.000
   5555.0      .0
  16.130 12.164
  16.130 12.164
  17.900 10.116
  17.900 10.116
  -1.000 -1.000
  -1.000 -1.000
  15555.0      .0
  17.935 10.089
  17.935 10.089
  18.071  9.993
  18.307  9.847
  18.644  9.667
  18.644  9.667
  18.686  9.652
  18.714  9.643
  18.721  9.638
  18.729  9.636
  18.729  9.636
  -1.000 -1.000
  -1.000 -1.000
```

Sample Run of the PARGEN Program

The following dialogue is not a verbatim report as created by the PARGEN program, but it does list nearly all interaction between the operator and the program as it would occur. This

listing reflects the application of the PARGEN program to the data shown above, and may omit other important dialogue which the PARGEN program may produce or require under other circumstances. As such, this example is designed to illustrate only the most basic operation of PARGEN. In the dialogue, the responses of the operator are underlined. **Comments are made in bold.**

DIGITIZED FILENAME? SAMPLE.OUT

ARC FILENAME? SAMPLEB_AW **(Arc file for database, the the operator may choose any)**

POINT FILENAME? SAMPLEB_PW **(Point file)**

QPLOT FILENAME? SAMPLEB_QW **(Qplot file)**

Input Header =

WY1959B_D1 WW 17-MAY-88 MP 24000. PR1975 ...C

(This map header shows the Kansas county + quad row and column + map layer + digitizing session number = base name, operator's initials, digitizing date, projection type, scale, revision date, and is marked ...C to show it has been cleaned by SYNEDT. This is shown to verify the correct input data is selected.)

Enter 1 to Continue? 1 **(verifying the DDF file)**

Node Recognition Threshold = 0.0025

Enter 1 to accept? 1 **(Consecutive points within this distance are considered a node. This value must be as small as possible as long as nodes are recognized. Enter 0 to change it.)**

Projection Possibilities are:

0 = No Projection
1 = Modified Polyconic
2 = Lambert Conformal Conic

Enter Projection Type? 1 **(Projection type is given by the MP in the Header in this case. If the projection area is not a 7.5′ quad, the projection type must be given as a negative number.)**

Enter Scale Factor? 24000. **(This is a 7.5′ quad map)**

Enter Row, Column numbers? 19,59 **(Only for Kansas 7.5′ quads)**

```
Select Re-entry (0) or Initialization (1)?   1
```

(This is the first data to be entered. Any existing data will be destroyed.)

```
Your choices are:

     Map projection =     MP

     Scale Factor =    24000.0

     Row =      19, Column =    59

     ** Initialization Option Selected **

Enter 1 to accept or 2 to Retry?   1
```

(The choices are correct and are accepted. Otherwise enter 2.)

Next, the PARGEN program calls the RAFMAN library to initialize the Point and Arc files. The suggested and current file sizes in records and record lengths are reported, and the operator selects the new size of the file. A positive value initializes the file to the chosen size. A negative size simply extends the current size of the file.

```
     * Point File Initialization *
Suggested MAX file size = 300  Record length = 68
Current MAX file size = 300  Record Length = 0
MAX < 0 for Extension  MAX > 0 for Initialization

     Enter new MAX file size?    25
```

(Only a few records are needed for this data set)

```
**  LENGTHS ARE NOT EQUAL  **
```

(Lengths will be unequal for the first initialization)

```
File Code 2 Initialized to 25 Records
With Record Length = 68
```

(And the same for the Arc file)

```
     * Arc File Initialization *
Suggested MAX file size = 300  Record length = 42
Current MAX file size = 300  Record Length = 0
MAX < 0 for Extension  MAX > 0 for Initialization

     Enter new MAX file size?    25

**  LENGTHS ARE NOT EQUAL  **
```

```
File Code 1 Initialized to 25 Records
With Record Length = 42
```

Next, PARGEN displays a block of double precision numbers which represent values associated with the map projection. Included in these numbers (see the PROMAN section) will be the locations of the projected corners of the projection area and the minimum Y coordinate at the bottom center of the area.

```
.682860140668021    .685041802459292     ...
          .
          .
          .

* *  Translation X, Y =   -.0005   -.0005

* *  Scale Factor X, Y =   .99984   1.00117

* *  Rotation Angle SIN, COS =   -.00002   1.00000
```

(These are the parameters set in the SETT routine to be used by the TRANS routine to correct for distortion errors as found in comparing the new control points to the projection corners. In this case, there is very little correction.)

Now, PARGEN begins the processing of the input DDF data in the PAG routine. Each arc has a line in the report showing the basic values, and each new feature code is clearly displayed.

```
-------------- Feature Code = 7777 ---------- (MAPEJ=0)
                                  (this feature is not a map edge)
Arc   Code       XF      YF       XL       YL        IPFPR
 2    7777     1.000   1.602    5.218    1.671         4
 3    7777     5.221   3.681    5.239    9.327         5
-------------- Feature Code = 5555 ---------- (MAPEJ=0)
Arc   Code       XF      YF       XL       YL        IPFPR
 4    5555    16.128  12.177  17.897   10.126         0
                                       (no interior points)
-------------- Feature Code = 15555 ---------- (MAPEJ=0)
Arc   Code       XF      YF       XL       YL        IPFPR
 5   15555    17.932   10.09   18.641    9.672         6
 6   15555    18.641   9.677   18.726    9.646         7
```

The end-of-file is found and the report is generated:

```
*  *  *  At the End of the Current Run  *  *  *
```
(A similar report for the previous run would be given if Re-entry selected)

```
   The Map Header = WY1959B_D1 WW 17-MAY-88 MP 24000. PR1975 ...C
   Number of Used Arcs = 4
   Maximum Arcs = 23
```

```
Number of Used Point Records = 3  (an error, this should be 4)
Maximum Point Records = 21

South = 39.125          (Projection area in degrees)
North = 39.250
West = -94.875
East = -94.750

XMIN =  1.00000         (Map extremes from projection or found)
YMIN =   .99695
XMAX = 18.73212
YMAX = 23.76430

CREATQ begins at 4      (Creation of the Qplot file starts at
                              the parallel Point record 4)
Ends at 7

STOP          (PARGEN stops after completing the job)
```

II. B A S I C S E Q U E N C E

II.4 THE NODGEN PROGRAM

A. Introduction

The **NODGEN** (for node file generation) program creates the records in the Node file from by collecting and analyzing the endpoints of the arcs. Each unique location at which an arc ends or two or more arcs intersect is recognized as a node and (at least) one record is created in the Node file. A single Node record may contain information for as many as four arcs which intersect at the node. When more than four arcs intersect at one location, multiple Node records are required. The additional records (beyond the first) vary slightly in form from the first record.

In each node, the location of the node is the average of the locations of all arc endpoints of the arcs which intersect at the node. When these final averaged locations are set in the Node file, the locations of the arc endpoints (in the Arc file) are updated to the final averages.

The process of recognizing nodes and creating appropriate records in the Node file follows four stages of operation. These are (1) fill the internal arrays with the endpoints from the Arc file while finding final averaged locations, (2) shuffle some nodes to make room for those which require more than one record (i.e. have degree > 4), (3) update the arc endpoint locations in the Arc file and set (ANCHOR) pointers in multiple-record nodes to connect extra records to the first, and (4) gather the node-arc information for each node and write it to a Node file record or to multiple records.

B. General Information

The **node** is defined as the endpoint of a linear arc which ends in the map interior, the endpoint of a linear arc which terminates at the map edge, the intersection point of two or more arcs or the location of an isolated point arc. In practice, it is sometimes more convenient and efficient to restrict the node definitions arising from arc intersections to those where the arcs are of like type. For example, intersections of small roads with interstate highways would produce nodes, but intersections of rivers with railroads might not (unless railroad bridges are of interest). All intersections can be treated as nodes, but they don't have to be when nothing is to be gained.

One special kind of node can be defined by the same kind of physical criteria as those above, but can not be located as such. Because the location must be chosen by the operator, this kind of node is called the **artificial node**. The need for the artificial node comes from two situations. First is the **simple island** zone, which is a polygon that shares no boundary with any other zone, and which is completely surrounded by a single other zone.

The arc which defines the island must be digitized, and the digitizing must begin and end at some point on the arc. But there is no naturally defined starting point. Thus, the operator marks a point to be used as the start and end of the arc. This point is an artificial node.

The second example is in a case called **bridging**. Imagine a wiggly, blue line on a map representing a river. The river comes to a point where there is a small lake resulting from a dam. At the other (downstream) side of the lake a single wiggly, blue line continues as the river moves away.

In digitizing this, the two wiggly lines are digitized as two lines. The lake is sigitized as two lines, one on each side of the lake. Both go from the end of the first wiggly line to the beginning of the second wiggly line.

In this case, we have the two obvious nodes where the wiggly lines of the river join the lines defining the two sides of the lake. One is at the upstream side of the lake and one is at the downstream side. Thus the two arcs defining the sides of the lake both connect the two nodes. They both go between nodes Nu and Nd. Identification of the arcs by their nodes would not be unique.

The solution to this bridging problem is to artificially split one of the arcs into two by inserting an artificial node at any point along the arc.

The node is a useful tool in automated cartography as a tool for identification or selection and as required information for describing the network or the topology of the map features. The node is a point at which some identifying mark or number can be displayed for identification or selection of map features. Arcs may be selected by specifying the endpoint nodes by number. This capability is useful for editing functions such as arc deletion, insertion or deletion of interior points, modification of feature code or interior point locations and many other arc-oriented functions.

As a key element of the network or topology of the map, the node connects locations (of nodes) with the set of intersecting arcs and the set of associated or (directly) connected nodes, each of which shows a corresponding connection back via an arc

with an opposite sign. (If an arc was digitized from node N1 to node N2, the arc number in the N1 record is positive, but in the N2 record it is negative). This information provides the ability to "travel" from any node to any connected node and to evaluate different optional paths. It also provides the ability to find the closed boundaries defining the zones or polygons by means of a simple **left-turn** or **face-finding** algorithm (see ZONGEN).

The number of arcs which intersect at a given node is called the **node degree.** While each Node record is limited to four arcs intersecting, there is (no longer) any restriction on the number of arcs which may intersect at asingle node, known as the **maximum node degree.** Although six may seem a reasonable limit in nature, maps often contain artificial information in addition to natural features and the maximum node degree may become quite large. The unlimited node degree is made possible by chaining together as many records (each wih four arc intersections) as needed.

The isolated point arc is assumed to be a degenerate linear arc with no interior points and both endpoint nodes the same. As such, it is represented as an arc record with a feature code, a single location, and (7) optional z-values. Each isolated point results in a node record, usually of degree one, but which may also occur at the intersection of linear arcs.

The node-arc pair (in the node record) for isolated points shows the arc connected to the same node (itself), and is not duplicated to show the arc as both positive and negative. In linear arcs, an island will usually have a single arc border broken by a single artificial node. This node will show a single node-arc pair with the arc shown only as positive and a node degree of one, the same as for the isolated point arc.

The NODGEN program uses the Arc file as its source of input to obtain the complete list of arc endpoints, including those of the isolated points. This set of all endpoints is then analyzed to find all the unique locations representing the nodes while the locations are maintained as running averages of all points found to be within a tolerance distance or **node threshold** (distance) of the working location.

The node threshold for recognizing unique node locations should be kept as small as possible so as to avoid the undesired migration of arc endpoints which results when distinct nodes are located closer together than the node threshold. However, the node threshold must also be large enough so that all arc endpoints which should be recognized as the same nodes are found to be close enough for NODGEN to do so. Selection of a node threshold depends on the resolution of the digitizing equipment as well as on the skill of the operator to digitize nodes in nearly the same place on each of several visits during a session.

The complete content of the Node file records is shown above (I.5 Digital Representation of GIMMAP Data). Basically, each Node record contains the location of the node, the degree of the node, the collection of node-arc pairs for the node, and a Next pointer to link multiple records together.

The location of the node is the averaged location of all arc endpoints which were located within the node threshold of the node. The degree of the node is the total number of arcs which intersected at the node, including all isolated points which were found to be located at the same place. The degree also specifies the number of node-arc pairs which exist in the node's records.

The node-arc pairs are the connections which occur at the node. The arc in each node-arc pair is an arc which is incident at (intersects another arc at) this node. The arc number may be positive if the arc begins here, or negative if it ends here (and starts at the other node). The node in each node-arc pair is the node at the other end of the arc in the pair, the other node. In the node record for the other node, there is a node-arc pair with this node and the same arc (with the opposite sign). Unused node-arc pairs in the last record for a node are set to zero.

The Next pointer in the node record connects the records for nodes with more than one record (i.e. degree greater than 4). The Next pointer has the record number of the next record in the singly-linked list for the node. There is no limit to the number of records in the list, that number is degree/4. All records in a muliple record list, beyond the first, will not need to store the degree. Instead, they have the record number of the first record for the node, stored as a negative. The Next pointer for the last record in the list (the first for nodes with only one record) is set to zero.

The NODGEN program requires only a small amount of dialogue with the operator to open the database. After naming the files, the operator must specify the node threshold

C. Operation

Operation of the NODGEN program requires a short dialogue to specify and open the database files, to initialize the Node file (always done, there is no re-entrance option), and to specify the node recognition threshold. For NODGEN, this threshold is the distance within which two nodes are considered to be identical. All nodes within this distance of each other are drawn together to their averaged location, and represented as a single node with multiple arcs incident.

The value used for the recognition threshold depends upon the quality of the digitizing equipment and the skill of the

digitizing operators to re-deigitize a node close to the first digitization. In general, this value should be less than or equal to 0.02 inch using a digitizing table with an accuracy (repeatability) of about 0.003 inch. Good operators may use a value equal to or less than 0.01 inch.

In some cases, it may be desirable to use a larger value to assure joining of poorly digitized nodes. At other times, the presence of distinct nodes at very close distances may require the use of a very small threshold value. When the database has been created by the reprojection of already cleaned data, the node recognition threshold is usually kept small.

Following the short dialogue, the NODGEN program proceeds through a series of four phases of operation to construct the Node file. The four phases perform the following tasks:

Phase I

Collect the arc endpoint locations from the linear arcs in the Arc file, checking each against the growing list to find those within the threshold. Average the node locations and keep track of how many and which arcs intersect at each. Also, keep track of the endpoint nodes for each arc.

Phase II

Re-arrange the nodes in the list to make room for those with degree (number of arcs incident) greater than four and move information so all arcs for a node are contiguous.

Phase III

Reset the averaged locations of the nodes in the information for the arcs and then analyze the arcs to calculate length and extremes (min/max x and y) for display and windowing. Add this information to the Arc file.

Phase IV

For each node, search the list of nodes for the arcs to find a complete list of **node-arc pairs** (incident arcs and nodes at the other end of the arcs). Create one or more Node file records containing the information for each node and write them to the Node file. Report statistics.

The dialogue and report of the NODGEN program is as follows:

C(omputer): Enter the Arc filename?

O(perator): Operator enters the Arc filename.

C: Enter the Node filename?

O: Operator enters the Node filename.

C: Enter the Point filename?

O: Operator enters the Point filename.
(the Point file is needed to update statistics)

C: "...Initialization of the Node file..."
(plus the usual RAFMAN file initialization dialogue reporting file statistics and requesting the new size of the Node file)

O: Operator sets the size of the Node file, generally near that of the Arc file, but varies with the type of data and the number of arc intersections. If too small, NODGEN will automatically increase the Node file size to that needed.

C: Enter the suggested Node Threshold Value?

O: Operator enters the node recognition threshold as described above. Usually between 0.05 - 0.2

C: Phase I... **(initial phase beginning)**

Phase II...

Phase III...

Phase IV...

(Report:)

For each record in the Node file the following values are reported:

NEXT - **(points to additional records for the node)**

DEGREE/ANCHOR -
(number of incident arcs for first records
-1 * record number of first record for others)

(the anchor ties all subsequent records to the first for nodes with greater than one record... i.e. with degree greater than 4)

XN,YN - **(averaged location of all endpoints which were originally within the threshold distance for this node)**

NA - **(up to 4 node-arc pairs per record)**

Report Counts:

(report includes counts for the following:)

Total Arcs
Last Arc
Total Nodes
Last Node

D. Programmer's Overview

1. Introduction to Routines

All the routines comprising the NODGEN program are written in the FORTRAN77 programming language, using the Data General version of that language for creation and testing. The names and basic functions of the routines are listed below:

ARCLMM - Examines arc to find length and extremes in X and Y

FILOPN - Opens terminal I/O, gets names/opens database files

MAIN - Initial setup, four phases of algorithm, report

MATCH - Compare new points to nodes, add or create new

2. Global Storage and Variables

The global variables used by the NODGEN program are listed below in groupings according to the COMMON storage areas in which they are stored. Each COMMON area is listed with the name of the area bounded by slashes as in the program. Each variable in each area is listed with a brief description of its use.

/ARCCOM/

KDARC - Feature code of an arc

XF,YF - The location of the first endpoint of the arc (in digitizer coordinates)

XL,YL - The location of the last endpoint of the arc (in digitizer coordinates)

IPFPR - Pointer to the first Point record for the arc

ALEN - Length of the arc in digitizer units

XMN,YMN - Minimum X, Y coordinates for the arc

XMX,YMX - Maximum X, Y coordinates for the arc

/FC/

IAFC - Unit number (file code) for the Arc file

IIFC - Unit number for terminal input

INFC - Unit number for the Node file

IOFC - Unit number for terminal output

IPFC - Unit number for the Point file

/UPDAT/

XN(10000) - X-coordinate for the new node locations

YN(10000) - Y-coordinate for the new node locations

IDEG(10000) - Degree (number of arcs incident) for the new nodes

N1N2(10000,2) - Start and end node numbers for the arcs

3. Files Accessed By PARGEN

The NODGEN program accesses five files. These are the terminal input and output files and three map database files. The map database files are the Arc file, the Node file and the Point file.

Unit Number	File Name	File Characteristics
IIFC (11)	Terminal Input	Input from terminal, padded, and sequential
IOFC (10)	Terminal Output	Output to terminal, sequential with FORTRAN carriage control
IAFC (1)	Arc File	Direct access, input/output, recordlength = 42B(ytes)

IPFC (2)	Point File	Direct access, input only, recordlength = 68B
INFC (3)	Node File	Direct access, output only, recordlength = 28B

4. Memory and Other Requirements

The NODGEN program has some modest requirements for internal memory, the size of which depends on how the limits have been set on the maximum number of arcs and nodes. These values may be set by the programmer to increase/decrease the allowed number of arcs and nodes while creating a corresponding increase/decrease in the amount of memory required for the program. Initially, both of these values are set at 10,000. The upper limit for these values is determined only by available memory in the computer.

E. NODGEN Routines in Detail

A listing of the routine names and basic functions is given above. In this section, the routines are developed in much more detail including general information and the argument list, a description of the funtions of the routine, a listing of other routines called, a list of which routines call each routine, an overview of the routine flow, and a list of major local variables with their type and an explanation of the use of the variable and other pertinent information.

ARCLMM (IPFPR, XF, YF, XL, YL, ALEN, XMN, YMN, XMX, YMX)

The ARCLMM routine calculates the length and extremes in X and Y for the given arc. The arc resides in the database and the complete string of interior points in the Point file is examined in this process. The endpoints given to ARCLMM are the newly averaged locations of the nodes which correspond to the endpoints of the arc as calculated by NODGEN. The length is in digitizer units and the extremes (minimum/maximum X and Y) are in digitizer coordinates (unless the database has been created from a source other than the digitizer).

The chain of Point file records (pointed to by IPFPR and linked together by the NEXT pointers) is followed from beginning to end. All points in each record (ending with possible negative values for unused coordinates in the final record) are used to calculate the length and extremes. The first Point record is pointed to by IPFPR if there are interior points. Otherwise, the value of IPFPR is zero. Calculations include the endpoint nodes.

The major variables used by the ARCLMM routine are:

Variable	Type/Size	Use
IPFPR	INTEGER*4	Pointer to first Point record
XF,YF	REAL*4	Location of first arc endpoint
XL,YL	REAL*4	Location of last arc endpoint
ALEN	REAL*4	Arc length (digitizer units)
XMN,YMN	REAL*4	Minimum X,Y for arc
XMX,YMX	REAL*4	Maximum X,Y for arc
IAFC	INTEGER*2	Unit number for Arc file
IPFC	INTEGER*2	Unit number for Point file
NEXT	INTEGER*4	Pointer to next Point record
BUFF	REAL*4	Coordinates for a Point record

FILOPN (no arguments)

The purpose of the FILOPEN routine is to ask the operator for the names of three database files and to open these files and the terminal input and output files. The database files are all opened as direct access files (each with its own record length as shown below). The Arc and Node files are opened to allow input or output while the Point file is restricted to input. The Arc file is used for both input and output and the Node file is only used for output by the NODGEN program. The terminal input/output files are opened with default options except that the output file is set for FORTRAN carriage control.

The major variables used in the FILOPN routine are:

Variable	Type/Size	Use
IO	CHARACTER*60	Gets file name and passes to OPEN
IIFC	INTEGER*2	Terminal input unit number
IOFC	INTEGER*2	Terminal output unit number
IAFC	INTEGER*2	Arc file unit number
INFC	INTEGER*2	Node file unit number

IPFC	INTEGER*2	Point file unit number

MAIN

The MAIN routine performs the entire task of generating the Node file from the Arc file in a process which is divided into four distinct phases and an initialization part. In the process, each unique node location must be recognized as the (averaged) location of the arc endpoints which lie within a set distance called the node recognition threshold. For each node, the pairs of incident arcs and connected nodes must be collected and stored in a chain of linked Node records. And, to maintain consistency, the arc endpoint locations must be reset in the Arc file records to coincide with the averaged locations in the Node file.

The initialization part calls the FILOPN routine to open the terminal and database files. Next, the RMIOFC routine from the RAFMAN library, is called to identify file unit numbers for later use by other RAFMAN routines (esp. RMGETR and RMINIT). The Arc file content is checked against a minimum size, and the Node file is initialized by a call to RMINIT, with the initial maximum file size specified by the operator. Finally, the node recognition threshold is specified by the operator and Phase I begins.

In Phase I, the complete set of arc endpoints taken from the linear arcs in the Arc file is collected and analyzed to set up a table of unique nodes with averaged locations (stored in XN, YN). When arc endpoints are recognized and added to a node, the node degree is increased by one. If an endpoint is not matched to any existing node, a new node is created.

In Phase I, the arc records are read in sequential order from the Arc file. For isolated points, the last endpoint is set to equal the first one (XL=XF and YL=YF). Then, the endpoints are compared to the existing node locations (in XN,YN) by calls to the MATCH routine. In that routine, matched nodes are added in the list, locations averaged, and the appropriate node numbers are assigned to the arc endpoints.

Whenever an arc endpoint does not fall within the specified node recognition threshold, a new node is added to the list in Phase I. The initial location of the node is the location of the arc endpoint and the degree is set to one. The arc's endpoint node (in N1N2) is set to that of the new node, whatever was next on the list (NEXTN). Finally, the two endpoint nodes for the arc are compared to see if they are the same. If so, the degree of the node is decremented by one, because a single arc has been counted as two arcs by mistake.

Phase II involves a bit of bookkeeping to prepare for proper handling of nodes whose degree exceeds four and which therefor

require more than a single record in the Node file. The object here is to move entries (in XN,YN and IDEG) so that the records for all nodes are contiguous.

In this process, nodes which follow nodes of degree greater than four are moved to make room for the additional information for the node. The moved nodes are put at the end of the list of current nodes (i.e. at position NEXTN), leaving a slot for the extra entries for the node. As many slots are emptied via the move as are needed for the node with higher degree. In order to prevent moving the information for some nodes twice, an analysis is made to determine precisely where to begin moving nodes to.

In order to keep track of moved nodes, the degree value (IDEG) at the old location is set to -1 * the new location. With this information, references to the moved node may be resolved to find the location to which it was moved. The index value at which the node information is stored corresponds to the record numbers in the Node file where the information is to be stored.

In Phase III, the endpoints of the arcs are reset so that all arcs will be drawn exactly to the averaged node location. The new and (at least temporarily) fixed endpoint locations allow the calculation of the length and extremes values for the arcs. And the node numbers assigned to the arc endpoints (N1N2) can be reset to reflect any moves made in Phase II.

Each arc is read from the Arc file in sequence. The node numbers assigned to the arc endpoints are checked to see if the nodes have been moved in Phase II, and are reset if so. The locations of the arc endpoints are reset to the averaged node locations. The ARCLMM routine is called to calculate the arc length and min/max X and Y coordinates by examining the interior points from the Point file.

When this has been finished, the updated arc information is written to the Arc file. In the case of isolated point arcs, the only operation required is to update the location of the isolated point as that of the averaged node location, and then write the record to the Arc file. This process is performed for all arcs in the Arc file of the database.

In Phase IV, the collections of node-arc pairs (in NA) which constitute the primary topology of the map are gathered from the endpoint node information (N1N2) for the arcs. One or more node records is compiled (containing location, degree and node-arc pairs) and is written to the Node file. Unused node-arc pairs in the last record for a node are set to zero. For nodes with more than one record, all subsequent records will have -1 * ANCHOR in the IDEG position, where ANCHOR is the record number of the first record for the node.

For each node, the endpoint node list is checked to find all occurences of the node. The first endpoint is checked, then the last endpoint is checked. If either is, the node then a node-arc pair is added to the record. Whenever the record gets full, it is written to the Node file. When the record is not the first, the IDEG value is set as above. When the last record is to be written, unless it is full, all the unused pairs are set to zero so they will not be used by mistake. This is redundant because of the degree value but is done for clarity.

The NEXT pointer in the records for a node is set to the record number (in the Node file) of the next record for the node except for the last record for a node. The NEXT pointer for the last node in the chain is set to zero (0). So, nodes of degree four or less require a single record with NEXT = 0.

At the end of Phase IV, a small report is presented. This report includes the following values:

Total Arcs	(number of arcs in the Arc file)
Last Arc	(highest arc record number used)
Total Nodes	(number of nodes in the Node file)
Last Node	(highest node record number used)

The major variables used in the MAIN routine are:

Variable	Type/Size	Use
XN(10000) YN(10000)	REAL*4	Coordinates for node locations
IDEG(10000)	INTEGER*2	Degree of nodes
N1N2(10000,2)	INTEGER*2	Endpoint nodes associate with arcs
NA(4,2)	INTEGER*2	Node-arc pairs for one node record
THRSH	REAL*4	Node recognition threshold
NEXTN	INTEGER*4	Next available node record number
NARC	INTEGER*2	Arc record number
NODE	INTEGER*2	Node record number
NEXT	INTEGER*2	Next record in Node file chain

MATCH (NARC, X, Y, N12, NODE, THRSH)

The MATCCH routine compares a given point (X,Y) to a node location for a selected position (NODE) in the list of existing node locations (XN,YN). Based on the node recognition threshold (THRSH), the MATCH routine determines if the point is within the distance defined as tha for points intended as identical nodes. If the point is not within this distance, MATCH does nothing.

If the point is determined to be the same as the indicated node, the (running) averages for the location of the node, (XN and YN), are updated to include the new point location. Then, the degree of the node is increased by one and the node is added as the endpoint node for the arc (in N1N2).

The major variables for the MATCH routine are:

Variable	Type/Size	Use
NARC	INTEGER*2	Arc number (for setting N1N2)
X,Y	REAL*4	Endpoint location
XN(10000), YN(10000)	REAL*4	Node locations
N1N2(10000,2)	INTEGER*2	Arc endpoint node numbers
IDEG(10000)	INTEGER*2	Node degrees
NODE	INTEGER*2	Node entry to be compared
THRSH	REAL*4	Node recognition threshold
ID	INTEGER*2	Single node degree

F. Example Run With Sample Data

This sample run of the NODGEN program is based on the status of the database files in the sample runs of the SYNEDT and PARGEN programs as discussed and illustrated above. The Arc and Point files used as input for this run of NODGEN are as they were shown to be created for those programs.

Sample Run of the NODGEN Program

Enter Arc File name? **SAMPLEB_AW** **(operator names the Arc file)**

Enter Node File name? **SAMPLEB_NW** **(operator names the Node file)**

Enter Point File name? **SAMPLEB_PW** **(operator names the Point file)**

...Initialization of the NODE file...

Suggested MAX file size = 10000 Recordlength = 28
Current MAX File size = 10000 Recordlength = 0

MAX < 0 for Extension, MAX > 0 for Initialization

(use a positive to initialize the file, the initial size is dependent on the size of the Arc file see above)

Enter new MAX file size? **25** **(initial size for Node file)**

*** LENGTHS ARE NOT EQUAL *** **(normal for file initialization)**
FILE CODE 3 INITIALIZED TO 25 RECORDS
WITH RECORD LENGTH = 28

The Suggested Node Threshold is 0.02

Enter the Node Threshold value? **0.01** **(in this example, the node locations are known to be good)**

... Phase I ... **Phase I is beginning...**

.... Phase II .... **And II...**

..... Phase III ..... **And III...**

...... Phase IV ...... **And finally, IV**

The report of the node records follows. Included are the node (record) numbers, NEXT record in the chain pointer, Degree/Anchor value, node location (x,y), and the node-arc pairs:

Node	Next	D/A	X	Y	N1	A1	N2	A2	N3	A3	N4	A4
2	0	1	1.00,	1.66	3	2	0	0	0	0	0	0
3	0	1	5.22,	1.67	2	-2	0	0	0	0	0	0
4	0	1	5.22,	3.68	5	3	0	0	0	0	0	0
5	0	1	5.24,	9.33	4	-3	0	0	0	0	0	0
6	0	1	16.13,	12.18	7	4	0	0	0	0	0	0
7	0	1	17.90,	10.13	6	-4	0	0	0	0	0	0
8	0	1	17.93,	10.10	9	5	0	0	0	0	0	0
9	0	2	18.64,	9.68	8	-5	10	6	0	0	0	0
10	0	1	18.73,	9.65	9	-6	0	0	0	0	0	0

And the small report:

```
TOTAL ARCS=      5 LAST ARC=      7
TOTAL NODES=     9 LAST NODE=    10
```

And NODGEN halts:

```
STOP
```

II. B A S I C S E Q U E N C E

II.5 THE GRFCHK PROGRAM

A. Introduction

The GRFCHK (graphical checking) program is designed to perform a detailed series of checks on the graphical qualities in the cartographic database. In many cases, if errors are found, they are reported and corrections are made to the database (if it is possible). In other cases, the errors are only reported. Some of the errors found by GRFCHK may arise as the result of operator oversight. Others probably arise from errors in the software (any software) that accesses the database. In fact, the GRFCHK program is primarily looking to detect inconsistencies in the database which are most likely caused by software.

Operations performed by the GRFCHK program are limited to the basic files in the cartographic database. These files are the Arc, Point, Node and Qplot files which have been created by programs of the basic sequence at the point GRFCHK is run. The checks performed by GRFCHK are to locate and possibly correct any errors created by the construction programs. In addition, GRFCHK may be used after the GRFEDT program has been used for graphical editing to find errors which may have been created by GRFEDT, or may be used at any other time to locate suspected errors.

Because GRFCHK makes changes directly to the database, it is recommended that a back-up copy be made of the database prior to the execution of GRFCHK. Files that contain certain errors may be changed by GRFCHK in ways that do not conform with the GIMMAP database structures and which may not be as easy to reconstruct as those with the original errors.

B. General Information

The GRFCHK program performs checks and corrections on errors that occur in the database as a result of a natural phenomenon of map data, errors caused by the digitizer operator, and other errors which result from a combination of digitizing and program actions. The majority of the work of the GRFCHK program lies in the comprehensive checking of the graphical information in the map database for errors of logic and consistency. Among the errors which are detected and corrected by GRFCHK are those of **bridging**, **duplicate arcs**, and **collapsed arcs**.

The term "bridging" refers to a specific graphical situation which occurs naturally on maps. Bridging is the connection of two different nodes by more than one arc. That is, node A and node B

are connected by two or more arcs so that identification of arcs by unique endpoints is no longer possible. In GIMMAP, it should always be possible to say that node A is connected to node B by arc C, where there is only one arc C. This should be true for all nodes and arcs in the database (but is not an absolute requirement...it is left to the operator's discretion).

"Duplicate arcs" occur (ostensibly) when the digitizer operator digitizes an arc twice. This is recognized by the GRFCHK program when a pair of arcs exists between the same two nodes (bridging) and the arcs have the same feature code. In this situation, GRFCHK attempts to determine if the two arcs are the same by measuring the area under each arc. If the areas are within a tolerance value, the arcs are considered identical.

"Collapsed arcs" are linear arcs (feature code > 1999) which connect one node to itself and which have no interior points. In other words, they are physically the same as isolated point arcs, but with a feature code indicating they should be linear. All arcs which connect one node to itself are checked for the absence of interior points which indicates a collapsed arc. It should be noted that it is legitimate for an arc with interior points to connect a node to itself in the case of islands. The existence of collapsed arcs is usually the result of extra buttons being pushed in digitizing or from certain error correction actions by the SYNEDT program.

These errors of bridging and duplicate or collapsed arcs are all corrected by the GRFCHK program. Where bridging is found, one of the arcs is split into two arcs, creating a new node. A new arc is created for half of the original arc, and the interior points are divided in half. The original nodes are updated to show connection to the new node by the original arc or by the new one (one positive, one negative).

The duplicate or collapsed arcs are corrected in similar ways. One of the duplicate arcs is deleted from the Arc file and the two nodes are updated to show only the remaining arc in the node-arc pairs. The collapsed arc is also deleted from the Arc file and the node-arc pair containing it for the node is removed. If that was the only node-arc pair, the node is deleted.

A number of potential errors in the database are checked and reported by GRFCHK though no correction is attempted. Virtually every value and relationship between values in the database is examined for errors in certain aspects. Only the basic database files (Arc, Point, Node, Qplot) are tested since the others are usually not in existence at this stage. Among the tests performed are the following:

1. The values in the Accounting record of the freelist and the freelist chain in the Arc, Node and Point files. The chain must be complete and the records pointed to by the Predecessor and Successor pointers for each record must also be freelist records with legitimate pointers. The chain must be complete back to the LAST value in the Accounting record for the file.

2. In the Point file, the map projection must be legitimate, the scale must be acceptable, row and column numbers must be legal if present (0 otherwise), the minimum/maximum X and Y values must pass certain tests (non-negative..) and the flag in both information records must be set (-1).

3. The interior point chains for all linear arcs must be intact. The pointers to the first point records must point to the beginning of a chain of point records for the arc in which the NEXT pointers all point to legitimate Point file records in chains ending in records with zero pointers. No records in the Point file should be inaccessible - they must belong to an arc chain or the freelist.

4. The number of z-values indicated for an isolated point arc (in NUMZ) must be between 0 and 7. All unused z-values in the records should be set to a default value (0.) at this point.

5. The statistics for linear arcs (length and extremes in X and Y) are calculated and compared to those in the Arc file. If a large discrepancy is found, an error is reported.

6. The node-arc pairs in the Node file are cross-compared to check that each pair has a reciprocal node-arc pair in the node record of the other node. That is, if node A connects to node B by arc C, then node A should have the node-arc pair of (B,C) in a record. Conversely, node B should have the node-arc pair of (A,C) in a record. Of course, one of the C values would be positive (origination node) and the other would be negative (destination node).

7. The arc endpoint locations stored in the Arc file should match the appropriate node locations in the Node file. The two locations are compared and if they are not within a set tolerance, an error message is displayed.

8. The Node file is examined to check for several possible errors. The node degree listed in the first record for a node should equal the total number of node-arc pairs in the records for the node. The chained lists of records for a node should be properly connected by the NEXT pointers. The ANCHOR values for secondary records should point back to the first record (negatively). Unused node-arc pairs in the last record for a node should be set to zero. Finally, any **unnecessary nodes**

must be listed. Unnecessary nodes are those which have two arcs incident at them, and thus have a degree of two. They exist on lines between other nodes and divide the lines into two parts, usually unnecessarily. An unnecessary node may exist at a point on a line where another line crosses, where the node was overlooked on the other line.

C. Operation

The operation of the GRFCHK program begins with the opening of the files. First, the terminal input and output files are opened and then the map database files are opened. For GRFCHK, all four of the basic files are opened - the Arc, Point, Qplot, and Node files. The output file may be the terminal output file or a separate disk file. The pathnames for all the files are entered by the operator.

After the files are open, the RAFMAN library is initialized by a call to RMIOFC to set the file unit numbers for the various files used by GRFMAN. The Freelist Accounting records are read from the database files to get the record numbers of the last used records in each. The maximum number of arcs and nodes is compared to see if the database can be processed by GRFCHK. If it is too large, an error message is printed and GRFCHK stops.

Next, the sequence of routines to perform the checks and corrections begins with the CHKARC routine which examines and corrects the records in the Arc file and related records in the Point and Qplot files. This is followed by the CHKNOD routine which examines the Node file records and some of the topological relationships and the CHKGRAF routine which checks for internal Node file consistencies and consistency between Arc and Node records.

The LEGIT routine checks for illegitimate arcs and the BRIDGE routine looks for cases of bridging to be corrected. The INFO routine is called to check values in the information records of the Point file and FREELIST is called to seek errors in the freelist structures of the database files. Finally, the MIRROR routine is called to check parallelism between the Qplot and Point files and the CHAIN routine records in the Point file.

Each of the called routines produces its own report of errors and corrections made to the database so no comprehensive report is made at the end of the GRFCHK program.

Operation of the GRFCHK program follows the sequence given above. This program is initiated by a dialogue in which the operator identifies the database files to the program:

C(omputer): Enter Arc filename?

O(perator): The operator enters the Arc filename.

C: Enter Node Filename?

O: The operator enters the Node filename.

C: Enter Point Filename?

O: The operator enters the Point filename.

C: Enter Qplot filename?

O: The operator enters the Qplot filename.

C: Choose Output Device
0 - Terminal, 1 - Temp File

O: The operator selects either the terminal or a temporary disk file for output from the GRFCHK program.

[If Temp File selected then]

C: Enter Temp filename

O: Operator enters temporary filename for GRFCHK output

[End of option]

The main program then notes the execution of each of the routines with a message (after which the routines may print their error and correction messages):

```
C:  Entering CHKARC...
    Entering CHKNOD...
    Entering CHKGRAF...
    Entering LEGIT...
    Entering BRIDGE...
    Entering INFO...
    Entering FREELIST...
    Entering MIRROR...
    Entering CHAIN...
```

and then the GRFCHK program terminates.

D. Programmer's Overview

1. Introduction to Routines

All the routines comprising the NODGEN program are written in the FORTRAN77 programming language, using the Data General version of that language for creation and testing. The names and basic functions of the routines are listed below:

ARCLMM - calculate and compare arc length and extremes

AREA - calculate the area under an arc

BRIDGE - check for bridging condition and duplicate arcs

CHAIN - check Point records for freelist or point chains

CHKARC - check isolated point values and linear arc statistics

CHKGRAF - check node-arc reciprocity, node location consistency and unnecessary nodes

CHKNOD - check NEXT chains, degrees, anchors and unused pairs

CORRECT - corrects the bridging problem

DELETE - deletes duplicate arcs, returns point records and updates connected nodes

DUPCHK - compares two arcs for duplication

ECOUNT - counts errors and halts program if limit exceeded

FILOPN - opens the database and optional disk output files

FREELST - check and report errors in freelist of database files

INFO - checks the information record in the Point file

INTPTS - used to divide interior points when splitting arc to correct bridging problem

LEGIT - deletes illegitimate arcs and updates node records

MAIN - initialize files, find last records and call other routines to perform GRFCHK functions

MIRROR - check parallelism between Point and Qplot records

SHIFT - delete node-arc pair by shifting, reset node degree

2. Global Storage and Variables

The global variables used by the GRFCHK program are listed below in groupings according to the COMMON storage areas in which they are stored. Each COMMON area is listed with the name of the area bounded by slashes as in the program. Each variable in each area is listed with a brief description of its use.

blank

ERRCNT - total count of errors found in the database

ERRMAX - number of errors allowed before termination

/AR/

IARC(10000,4) - information about arcs

(1) feature code (3) "from" node (N1)

(2) point file pointer (IPFPR) (4) "to" node (N2)

AXY(10000,4) - arc endpoint locations

(1,2) XF,YF (3,4) XL,YL

NODE(10000,10) - node record information

(n,1) NEXT pointer to related records for node n

(n,2) node degree for node n

(n,3-4) first node-arc pair for node n

.
.
.

(n,9-10) last node-arc pair for node n

XYN(10000,2) - node (X,Y) locations

/FC/

IAFC - Unit number (file code) for the Arc file

IIFC - Unit number for terminal input

INFC - Unit number for the Node file

IOFC - Unit number for terminal output

IPFC - Unit number for the Point file

IQFC - Unit number for the Qplot file

ITFC - Unit number for Temporary output file

IFC - Unit number for output file whether it is the terminal or the temporary disk file

/LM/

IFIRSTA - First Freelist record in the Arc file

LASTA - Last used record in the Arc file

IFIRSTN - First Freelist record in the Node file

LASTN - Last used record in the Node file

IFIRSTP - First Freelist record in the Point file

LASTP - Last used record in the Point file

3. Files Accessed By GRFCHK

The GRFCHK program accesses up to seven files. These are the terminal input and output files, four map database files, and the optional temporary disk file for program output. The map database files are the Arc, Node, Point and Qplot files.

Unit Number	File Name	File Characteristics
IIFC (11)	Terminal Input	Input from terminal, padded, and sequential
IOFC (10)	Terminal Output	Output to terminal, sequential
IAFC (1)	Arc File	Direct access, xxxrecordlength = input/output, 42B(ytes)
IPFC (2)	Point File	Direct access, input/output, recordlength = 68B
INFC (3)	Node File	Direct access, input/output, recordlength = 28B

IQFC(4)	Qplot File	Direct access, input/output, recordlength = 36B
ITFC(13)	Temporary File	Sequential, program output

4. Memory and Other Requirements

The GRFCHK program has some modest requirements for internal memory, the size of which depends on how the limits have been set on the maximum number of arcs and nodes. These values may be set by the programmer to increase/decrease the allowed number of arcs and nodes while creating a corresponding increase/decrease in the amount of memory required for the program. Initially, both of these values are set at 10,000. The upper limit for these values is determined only by available memory in the computer.

E. GRFCHK Routines in Detail

A listing of the routine names and basic functions is given above. In this section, the routines are developed in much more detail including general information and the argument list, a description of the functions of the routine, a listing of other routines called, a list of which routines call each routine, an overview of the routine flow, and a list of major local variables with their type and an explanation of the use of the variable and other pertinent information.

ARCLMM (IPFPR, XF, YF, XL, YL, ALEN, XMN, YMN, XMX, YMX)

The ARCLMM routine calculates the length and extremes (the minimum and maximum X and Y values) in X and Y for the given arc. The arc resides in the database and the complete string of interior points, in the Point file, pointed to by IPFPR, is examined in this process. With the interior points, the arc endpoints (XF,YF) and (XL,YL) are used to determine the arc length and extremes. The length is in digitizer units and the extremes are in digitizer coordinates (unless the database has been created from a source other than the digitizer).

The chain of Point file records (pointed to by IPFPR and linked together by the NEXT pointers) is followed from beginning to end. All points in each record (ending with possible negative values for unused coordinates in the final record) are used to calculate the length and extremes. The first Point record is pointed to by IPFPR if there are interior points. Otherwise, the value of IPFPR is zero. Calculations include the endpoint nodes.

The major variables used by the ARCLMM routine are:

Variable	Type/Size	Use
IPFPR	INTEGER*4	Pointer to first Point record
XF,YF	REAL*4	Location of first arc endpoint
XL,YL	REAL*4	Location of last arc endpoint
ALEN	REAL*4	Arc length (digitizer units)
XMN,YMN	REAL*4	Minimum X,Y for arc
XMX,YMX	REAL*4	Maximum X,Y for arc
IPFC	INTEGER*2	Unit number for Point file
NEXT	INTEGER*4	Pointer to next Point record
BUFF	REAL*4	Coordinates for a Point record

AREA (A, XF, YF, XL, YL, IPFPR)

The AREA routine finds the area under the arc whose endpoint coordinates are given and whose interior points are found in the Point file beginning with the record pointed to by IPFPR. The area (A) is calculated in square digitizer units.

This area calculation is performed at the request of the DUPCHK routine which is attempting to determine if two candidate arcs are actually duplicate arcs. The areas under the two suspect arcs are calculated, and if they are within a tolerance percentage the area test indicates duplication.

The major variables used by the AREA routine are:

Variable	Type/Size	Use
A	REAL*4	Area under the arc
XF,YF	REAL*4	First node for the arc
XL,YL	REAL*4	Last node for the arc
IPFPR	INTEGER*4	Pointer to the first Point record
N	INTEGER*4	Next record in arc's Point chain

BUFF(16)	REAL*4	Eight points in a Point record
X,Y	REAL*4	Previous point for calculation of trapezoidal areas under arc

BRIDGE (no arguments)

The BRIDGE routine examines the database searching for the presence of bridging errors and the existence of duplicate arcs. Both of these errors are found by comparing the "other" nodes which are connected to a given node. All possible pairs of other nodes are so compared in this way. When the two nodes are identical, either a bridging error exists or the arcs are duplicate arcs.

For each identical pair of connected, other nodes, the BRIDGE routine calls the DUPCHK routine to determine if the two arcs are actually duplicates according to certain criteria (see DUPCHK). If so, BRIDGE calls the DELETE routine to delete the second arc found in the list of the central node.

In the case where duplicate arcs are not found, a bridging error exists. This case is dealt with by calling the CORRECT routine to correct the bridging error. Correction includes the splitting of one of the arcs at approximately the middle interior point. This creates a new node which connects the original nodes and removes the bridging condition.

The major variables used by the BRIDGE routine are:

Variable	Type/Size	Use
LASTN	INTEGER*4	Last possible used Node record +1
NODE(10000,10)	INTEGER*2	Contains Node record information except location for up to 10000 nodes
NEXT	INTEGER*4	Next other node for comparison
IDEG	INTEGER*4	Degree of node being examined
NA(20,2)	INTEGER*4	Node-arc pairs for central node
IT	INTEGER*4	Node number for connected node
IA1,IA2	INTEGER*4	Record numbers of arcs being compared
K1,K2	INTEGER*4	Offsets in NA list
N1	INTEGER*4	Record number of node being examined

CHAIN (no arguments)

The CHAIN routine examines the Point file to search for any records which belong neither to the Freelist nor to any chain of interior points for an arc. This process sets values in an array to -1, indicating "inaccessible" record, for all possible records in the file. The Freelist is then followed from beginning to end and records found there have their values set to (1) "accessible". Finally, all interior point chains for all arcs in the Arc file are searched according to their SUCCESSOR pointers. All records found by this process are marked positive for "accessible".

The list of markers is then searched for any that remain with a negative value. These records are then reported as inaccessible point records (if NEXT is non-negative) or as inaccessible Freelist records (otherwise).

The major variables used by the CHAIN routine are:

Variable	Type/Size	Use
LASTP	INTEGER*4	Last possible used Point record +1
LASTA	INTEGER*4	Last possible used Arc record +1
IPT(20000)	INTEGER*4	Point record accessibility table
KD	INTEGER*2	Arc feature code
NEXT	INTEGER*4	Pointer to next Point chain record
IPFPR	INTEGER*4	Pointer to first Point chain record

CHKARC (no arguments)

The CHKARC routine reads the arc records from the database and checks the z-values of the isolated point arcs and the statistics of the linear arcs. The arc values are stored in the IARC and AXY arrays in the /AR/ common area for use by other routines as well.

The feature code and the location of isolated point arcs are stored in the arrays, but z-value information is not stored there. For isolated point arcs, the number of z-values (NUMZ) is checked to be from zero to seven. If there are less than seven z-values, the unused ones are set to zero.

For the linear arcs, the arc feature code, point file pointer and endpoint locations are read and stored in the arrays. The statistical values are read for the testing phase only and are not

stored in the arrays. The ARCLMM routine is called to re-calculate the arc length and the arc X and Y extremes based on the current content of the Point file records for the arc. These new values are compared to the existing ones and must be within a threshold value or an error is reported, corrected and written to the file.

The major variables used by the CHKARC routine are:

Variable	Type/Size	Use
KDARC	INTEGER*2	Arc feature code
XF,YF	REAL*4	Location of first arc endpoint or location of isolated point arc
XL,YL	REAL*4	Location of last arc endpoint
NUMZ	INTEGER*4	Number of used z-values (0-7)
Z(7)	REAL*4	Z-values for isolated point arc
IPFPR	INTEGER*4	First record in Point file chain for linear arcs
IARC(10000,4)	INTEGER*4	Stores arc information including the feature code and Point file pointer
AXY(10000,4)	REAL*4	Stores are endpoint locations
ALEN,BLEN	REAL*4	Arc length, file and re-calculated
XMN,YMN, XMX,YMX	REAL*4	Arc extremes from file
BXMN,BYMN BXMX,BYMX	REAL*4	Arc extremes calculated from endpoint locations and Point file records
EPS	REAL*4	Threshold for statistics comparisons

CHKGRAF (no arguments)

The CHKGRAF routine examines three locational and graphical or topological aspects of information in the database. For each arc in the database, these three tests are performed in the three phases of the CHKGRAF routine.

In the first phase, the location of the endpoints of the arc being examined are compared to the location of the node(s) at the endpoints of the arc. In the case of isolated point arcs, the single location of the isolated point is compared to the location

of the node. For linear arcs, the locations of both endpoints of the arc are compared to the locations of the nodes at the end of the arc as indicated by the Node file. Where differences greater than the threshold value (EPS) are found, an error message is reported and the location is corrected in the database.

In the second phase, the node connections implied by the arc are checked for the appropriate reciprocity implied by the data structures for GIMMAP. The arc's endpoints must be represented by nodes in the Node file, with each node containing a reference to the other in its node-arc pairs. Furthermore, the arc in the node-arc pair of the "from" node (where the arc begins) must be positive and the arc in the "to" node (where the arc ends) must be negative. For isolated point arcs, the two nodes are the same and the arc is listed in the node-arc pairs only once. Errors in the second phase are reported but not corrected.

The third phase of CHKGRAF is an examination of the nodes for possible **unnecessary nodes**, which are nodes of degree 2 where the feature codes of the two arcs are the same. In other words, these are nodes which seem to unnecessarily split a single arc into two arcs. There are cases in which such splitting may serve a useful purpose, but unnecessary nodes are those which serve no purpose and which should be removed to unite the two arcs into one. (Such nodes may be useful to divide long arcs into manageable pieces).

The third phase searches the records of the Node file in order to locate nodes of degree two. When one is found, the connected arcs are examined to determine if both are linear type arcs and if both have the same feature code. When such a suspected unnecessary node is found, a report is made. No attempt is made to correct the problem since the node may be necessary as suggested above.

The major variables used by the CHKGRAF routine are:

Variable	Type/Size	Use
EPS	REAL*4	Threshold for equality testing errors
LASTA	INTEGER*4	Last possible used Arc record +1
LASTN	INTEGER*4	Last possible used Node record +1
NODE(10000,10)	INTEGER*2	see BRIDGE above
IARC(10000,4) AXY(10000,4)	INTEGER*4 REAL*4	see CHKARC above
XYN(10000,2)	REAL*4	Node X,Y locations
N1,N1T	INTEGER*4	Record number of first node for arc

N2,N2T	INTEGER*4	Record number of last node for arc

CHKNOD (no arguments)

The CHKNOD routine is responsible for filling the NODE and XYN arrays with information about the records in the Node file. The NODE array is filled with the NEXT pointers (for multiple records), the node degree (ANCHOR value for multiple records), and the node-arc pairs for the nodes. The IARC array is also filled with the node numbers of the endpoint nodes of all of the arcs.

After filling the arrays with this information, the CHKNOD routine performs a number of checks on the node information. Among these are checks on the chains for nodes with multiple records (those whose degrees exceed four) by following the NEXT pointers, checks that the node degrees are consistent with the contents of the node records, checks that ANCHOR values of all records beyond the first point back to the first, and checks that all unused node-arc pairs (indicated by the node degree) are set to zero.

The major variables used in the CHKNOD routine are:

Variable	Type/Size	Use
LASTN, NODE, XYN, IARC	see above	
IDEG	INTEGER*2	Node degree
NEXT	INTEGER*4	Next Node record for multiple records
IA	INTEGER*4	Arc record number from node-arc pair
IN	INTEGER*4	Node record number from node-arc pair

CORRECT (N1, NA2, K1, K2)

The CORRECT routine is called when the BRIDGE routine detects a bridging problem (two arcs which share the same pair of nodes). The CORRECT routine is called to correct the bridging problem by splitting one of the arcs into two arcs, creating a new node. In this process, the arc indicated in the node-arc pair (in NA2) by the first pointer in the argument list (K1) is the arc to be split unless it has no interior points. In that case, the arc indicated in the node-arc pair pointed to by K2 is split.

Once the arc to be split has been determined, the INTPTS routine is called to perform the division of the interior points

in the Point file and to parallel this new construction in the records of the Qplot file. The coordinates of the last point in the middle record of the interior point chain are used as the point at which the split takes place, and are the coordinates returned by INTPTS as the location of the new node.

Next, the ARCLMM routine is called for each of the two arcs created by the split of the original arc. This creates the values for the arc length and X/Y extremes statistics for these arcs. The values for the new node location, arc endpoints, new node number, and Point file pointer are updated in the arrays and in the Arc and Point files of the database. The changed connections between nodes in the network and the introduction of a new arc and a new node are reflected by changes in the node-arc pairs in the Node file and the creation of records in the Arc and Node files.

The major variables used in the CORRECT routine are:

Variable	Type/Size	Use
IARC, AXY, NODE, XYN	see above	
N1	INTEGER*4	Node on one end of arc to split
N2	INTEGER*4	Node on other end of arc to split
NA2(20,2)	INTEGER*2	All node-arc pairs for node N1 (assumed to be 20 or less)
K1	INTEGER*4	Position of one node-arc pair in NA2 with N2 and potential arc to split
K2	INTEGER*4	Position of other node-arc pair with N2 and potential arc to split
NARC	INTEGER*4	Record number of arc to be split
NA(4,2)	INTEGER*2	Node-arc pairs for one record of node N1 or node N2
NEW(4,2)	INTEGER*2	Node-arc pairs for new node record
IPFPR,IPFPRN	REAL*4	First record in the Point file for original and new arcs
NEWA,NEWN	INTEGER*4	Record number for new arc and node
XNEW,YNEW	REAL*4	Location of new node
XF,YF XL,YL	REAL*4	First, last endpoints of original arc

DELETE (IA1, IA2, N1, N2, NA, K2, IDEG)

The DELETE routine is called by the BRIDGE routine whenever duplicate arcs are found to delete one of the two arcs. Of the two arc numbers passed (IA1,IA2), the second is the arc to be deleted. If the arcs are isolated point arcs then a number of checks are performed before the second arc is deleted. If the arcs are linear arcs then the interior points are deleted. In either case, the node-arc pairs of the two nodes (N1,N2) are updated.

For isolated point arcs, the z-values must be checked to see that they agree in number and value. If they do, then the second arc is deleted. If they do not agree, then one of two alternatives occurs. If the two sets of z-values can be packed into a single isolated point arc record, then the z-values are combined into the first arc and the second is deleted. If they can not be combined (there are more than seven), then the two records are kept.

In either case, the node-arc pairs containing the deleted arc are removed from the node records of the two nodes. This is performed by use of the SHIFT routine to shift all subsequent node-arc pairs for the two nodes up into the position containing the arc which is deleted. Finally, the degrees of the nodes are decreased by one.

After the isolated point checks have been performed and the interior points of linear arcs have been deleted, the node-arc pairs of the affected nodes are updated. Finally, the arc record of the deleted is returned to the Freelist of the Arc file.

The major variables used by the DELETE routine are:

Variable	Type/Size	Use
IARC,NA	see above	
IA1,IA2	INTEGER*4	The duplicate arcs
N1,N2	INTEGER*4	The two nodes connected by the arcs
NA(4,2)	INTEGER*4	One record of node-arc pairs for N1
K2	INTEGER*4	Location of duplicate arc in NA
IDEG	INTEGER*4	Degree of node N1
KD1,KD2	INTEGER*2	Feature codes of the duplicate arcs
XF1,YF1 XF2,YF2	REAL*4	Locations of isolated point arcs

Z1(7),Z2(7)	REAL*4	Z-values of isolated point arcs
NUMZ1,NUMZ2	INTEGER*4	Number of z-values of isolated points
XF,YF XL,YL	REAL*4	Location of first/last arc endpoints

DUPCHK (IA1, IA2, IRESP)

The DUPCHK routine is responsible for determining if two arcs are actually duplicates. For isolated point arcs, the only check needed is to test if the feature codes are the same since the location of the nodes must be the same. For linear arcs, the arc lengths and areas under the arcs are calculated (via the AREA) routine. The DUPCHK routine is called from BRIDGE when two nodes are connected by two arcs. The DUPCHK routine signals its finding to BRIDGE by setting IRESP to 1 if the arcs are duplicate and to zero if they are not.

The major variables used by the DUPCHK routine are:

Variable	Type/Size	Use
IARC	see above	
IRESP	INTEGER*4	Indicates duplicate arcs (=1) or not (=0) on return to BRIDGE
KD1,KD2	INTEGER*2	Feature codes of the two arcs
XF1,YF1,XL1, YL1,XF2,YF2, XL2,YL2	REAL*4	First and last arc endpoints of the two potentially duplicate arcs
IPFPR1,IPFPR2	INTEGER*4	First Point records for the two arcs
ALEN1,ALEN2	REAL*4	Lengths of the two arcs
AREA1,AREA2	REAL*4	Areas under the two arcs

ECOUNT (no arguments)

The ECOUNT routine is called to maintain a running count of the total number of errors encountered by the GRFCHK program. Each routine in the GRFCHK program that detects errors calls the ECOUNT routine in every case of error detected. The ECOUNT routine counts the total number of errors (ERRCNT) as the program executes, and compares the count to a maximum (ERRMAX) number of errors allowed.

The purpose of limiting the total number of errors is included to prevent unnecessary processing of databases which have in some manner been damaged or have not been properly or completely processed. Such databases may contain nothing but errors in the view of GRFCHK, and no purpose is served in reporting them. If the maximum number of errors is exceeded, an error message is printed and the program is halted.

The major variables used in the ECOUNT routine are:

Variable	Type/Size	Use
ERRCNT	INTEGER*4	Count of the total errors found
ERRMAX	INTEGER*4	Maximum number of errors

FILOPN (no arguments)

The purpose of the FILOPN routine is to ask the operator for the names of the database files to be opened, and to open these files along with the terminal input and output files. Optionally, a report file (on disk) may be named and opened if the terminal output is not used for this purpose.

The database files are opened as direct access files (each with its own record length as shown below). The database files are opened for both input and output since all must be read for the examinations of the GRFCHK program and all may be written to perform corrective functions. The terminal input/output files are opened with default options.

The output from the GRFCHK program may be directed to the terminal output (terminal screen) file or to an optional temporary disk file which is named by the operator.

The major variables used in the FILOPN routine are:

Variable	Type/Size	Use
IO	CHARACTER*60	File name selected by Operator
IIFC	INTEGER*2	Terminal input unit number
IOFC	INTEGER*2	Terminal output unit number
IAFC	INTEGER*2	Arc file unit number
INFC	INTEGER*2	Node file unit number

IPFC	INTEGER*2	Point file unit number
IQFC	INTEGER*2	Qplot file unit number
ITFC	INTEGER*2	Temporary file unit number
IFC	INTEGER*2	Report file unit number

FREELST (no arguments)

The FREELST routine is called from the MAIN routine to check the Freelist chains of the Arc, Node and Point files. The Freelist chains begin with the Accounting record, the first record in each file. The Accounting record contains a pointer (IFIRST) to the top of the chain of Freelist records and another pointer (LASTx) to the last record in the chain.

The FREELST routine begins with the first record and follows the chain through the last record of the Freelist chain, examining each Freelist record for three things. First, the first value in a Freelist record must be a negative integer (INTEGER*2, usually a -1). The two remaining values (IPRED and ISUCC) are pointers to the predecessor and successor records in the Freelist chain. These values must be consistent with the values in those records.

The FREELST routine does not correct any problems it finds in the Freelist chain of any of the three files. It does produce an error message describing the problem and calls the ECOUNT routine to increment the count of total errors.

The major variables used in the FREELST routine are:

Variable	Type/Size	Use
IFLAG	INTEGER*2	Flag marking records as Freelist
LASTA, LASTN, LASTP	INTEGER*4	Last potential Arc,Node,Point record
L(3)	INTEGER*4	Same as LASTA, LASTN, LASTP
IFIRST	INTEGER*4	First Freelist record in chain (not necessarily first in the file)
IFT(3)	INTEGER*4	Contains FIRST values for Arc, Node and Point files
IPRED,ISUCC	INTEGER*4	Predecessor, successor record numbers for a Freelist record

INFO (no arguments)

The INFO routine checks some of the map information stored in the first records of the Point file. The first record in the file (as in all GIMMAP database file) is the Accounting record, which contains the Freelist chain pointers and record size. Following the Accounting record in the Point file are two records containing information about the map. The first Information record contains the Map Header, a 64-character text record which usually includes the map base name, operator initials, date and projection values. This record is not checked by INFO.

The second Information record contains information about the map projection, extremes and identifying row and column numbers for standard 7.5' quadrangle maps. The map projection type indicator names the projection type (0=none, 1=polyconic...) and must have a value from -2 to 2. Maps with projection must have a positive map scale value. Quadrangle map row and column numbers must be in the ranges 1-26 and 1-61. Minimum X and Y values must be less than the maxima and greater than 0. And the flag value at the beginning of the Information record must be negative. Errors found in these checks will be reported but only the flag value error is corrected.

The major variables used by the INFO routine are:

Variable	Type/Size	Use
IFLAG	INTEGER*2	Negative value at beginning of record marking it as Information or Freelist
IACCR	INTEGER*4	Number of Accounting record
INFO2	INTEGER*4	Number of second Information record
NPRO	INTEGER*2	Projection type indicator
SCALE	REAL*4	Map scale factor
NROW,NCOL	INTEGER*2	Quadrangle map row and column numbers
XMN,YMN,XMX, YMX	REAL*4	Map/Projection min/max X and Y values

INTPTS (XF, YF, XL, YL, XNEW, YNEW, IPFPR1, IPFPR2)

The INTPTS routine is called by the CORRECT routine whenever a bridging problem has been encountered and one arc is to be split to create a new node. The INTPTS routine is called to divide the interior points of the arc to be split into two parts. The process depends on the number of records of interior points for the arc.

This number is determined by INTPTS reading through the chain of interior point records in the Point file, counting the number of used records for the arc.

There are three general cases when the interior points are to be divided. One is when the arc has no interior points. In that case, a node is created by averaging the locations of the endpoint nodes of the arc being split. The second case is an arc with more than one record of interior points. In this case, the middle point record (number of records/2) is selected and the last point in this record is removed and used as the new node. The remaining records are used for the new arc.

In the third case, the arc has only one record of interior points. If this record has more than two points, the middle point is used as the new node and the remaining points are removed and placed in a new Point record for the new arc. If the record has only one or two points, the second or only point is used and the new arc will have no interior points.

At the conclusion of the update of the records in the Point file, the INTPTS routine updates the information in the parallel Qplot file to correspond to the changes in the Point file.

The major variables used by the INTPTS are:

Variables	Type/Size	Use
IREC	INTEGER*4	Number of record to read/write
NEXT	INTEGER*4	Pointer to next record in chain
NUM	INTEGER*4	Number of point records
XF,YF,XL,YL	REAL*4	Arc first/last endpoint locations
XNEW,YNEW	REAL*4	Location of new node
IPFPR1,IPFPR2	INTEGER*4	Pointer to the first point record of the original and new arc
BUFF(16)	REAL*4	Contains points from 1 Point record
IBUF(16)	INTEGER*2	Contains points from 1 Qplot record
ONE	REAL*4	-1. value for unused Point entries
NONE	INTEGER*2	-1 value for unused Qplot entries

LEGIT (no arguments)

The LEGIT routine is called from the MAIN routine to check the arcs for a special case called the **illegitimate arc.** This case is when a linear arc (feature code > 1999) has no interior points (the value of IPFPR is zero) and the arc endpoints are the same. When this is found, the illegitimate arc is deleted by a call to SHIFT to remove the arc from the Node records and a call to RMPUTR to return the Arc record to the Freelist.

The tests performed on the arc include a check to see that the feature code is greater than 2000. Then, the interior points are checked. Arcs with no interior points are candidates for deletion, but arcs with only a single interior point may also be deleted if the other tests are positive. Two tests are performed to determine if the two endpoint nodes are at the same location. The first is to compare the X and Y coordinates of the two points. These must be within a threshold distance for the arc to be deleted. A second test calculates the distance between the two points, which again must be less than a threshold value.

The major variables used in the LEGIT routine are:

Variable	Type/Size	Use
LASTA	INTEGER*4	Last potential Arc record
ARC	INTEGER*4	Number of Arc record to examine
IARC(10000,4)	INTEGER*4	Arc information (KDARC,IPFPR,N1,N2)
AXY(10000,4)	REAL*4	Arc endpoint locations
EPS	REAL*4	Coordinate equality threshold
DIST	REAL*4	Distance between arc endpoints
THRSH	REAL*4	Distance threshold

MAIN

The MAIN routine has few tasks to perform other than to call other routines to perform all the tasks of the GRFCHK program. First, the FILOPN routine is called to open the database files. The RMIOFC routine is called to initialize the RAFMAN routines, and the accounting records of the Arc, Node, and Point files are read to get the record number of the last potential used record in each. The numbers of arcs and nodes are checked against the maximum (MAX) number that GRFCHK can process. Finally, the other routines of the GRFCHK program are called to perform the GRFCHK functions.

The major variables used in the MAIN routine are:

Variable	Type/Size	Use
MAX	INTEGER*4	Maximum number of arcs and nodes
ERRCNT	INTEGER*2	Count of errors discovered
ERRMAX	INTEGER*2	Maximum number of errors allowed
LASTA,LASTN, LASTP	INTEGER*4	Last potential Arc, Node or Point record in the file

MIRROR (no arguments)

The MIRROR routine is called by the MAIN routine to compare the Qplot file to the Point file. Each chain of Point file records for an arc should have a parallel record in the Qplot file which contains the parallel points, converted for screen display. The Next pointers in the two files should be identical, and the content of the records should be parallel in terms of used and unused values.

The major variables used in the MIRROR routine are:

Variable	Type/Size	Use
NP	INTEGER*4	Next record in a Point chain
NQ	INTEGER*4	Next record in a Qplot chain
BUFF	REAL*4	Buffer of 8 points in Point record
IBUF	INTEGER*2	Buffer of 8 points in Qplot record

SHIFT (N, IA)

The SHIFT routine is called by the DELETE routine to remove the node-arc pair containing an arc which is being deleted from one of the endpoint nodes. The remaining node-arc pairs are shifted up to fill in the record and the node degree is decremented by one. The changes are then written to the corresponding records in the Node file.

The major variables used in the SHIFT routine are:

Variable	Type/Size	Use
N	INTEGER*4	First record for node containing the node-arc pair to be deleted
IA	INTEGER*4	Arc number of pair to be deleted
NODE(10000,10)	INTEGER*2	Node information (NEXT,IDEG,NA(4,2))
IDEG	INTEGER*4	Node degree
NT,NT2	INTEGER*4	Indices used for shifting entries

II. B A S I C S E Q U E N C E

II.6 GRFEDT - THE GRAPHICAL EDITING PROGRAM

Introduction

Errors enter the cartographic database from many different sources throughout the mapping process. Many different kinds of errors may be introduced by the operator when a map is digitized, when the digitized data is used to create the database, or even in examining the map in the planning stages. Other errors may be caused by the software which edits the digital data and creates the database. Still other errors may exist in the source data itself, or may be caused by software or operator after database creation. Finally, map data represents features on the earth which themselves may change in time, requiring changes in the cartographic database.

For these reasons, a program exists which allows the operator to make changes to the content of the cartographic database. In general, changes made by the **interactive graphical editing** program **GRFEDT** are to the graphical aspects of the cartographic data. For example, locations of isolated points, nodes and interior points of arcs can be changed to new locations selected graphically by the operator. Arcs, points and nodes may be created and deleted, and connections of arcs at nodes may be modified by creating new nodes or deleting unnecessary nodes. And the feature codes which define the types of features can be changed individually or in groups.

The GRFEDT program is an **interactive** program, allowing the operator to view the map data, locate errors to be corrected, and make the changes, all as the operator views the map data. GRFEDT offers the operator a list of commands called a **menu**, from which the operator selects either **modifying** or **utility** commands. The modifying commands actually make the changes to the contents of the database as directed by the operator. The utility commands are used to find and identify errors by relocating or changing the size of the view, filtering features for display, or by otherwise enhancing the operator's view of the data.

The GRFEDT program updates the contents of the database after the operator has selected and executed a modifying command and has (usually) verified. Only then does the program change something in the database, but this is done immediately. The changes made during an editing session are immediately reflected in subsequent displays of the map data during that session. It is recommended that this program is set up so that the results of a session may be abandoned if so desired.

The GRFEDT program requires a **graphics display terminal** to display the map data and to graphically select items to be edited. The graphics terminal must support the display of line or **vector**

information in which map features are represented by one or more vectors defined by two points in a given sequence. This ability may be provided by either a **vector graphics terminal** or by most of the **color, raster graphics terminals** which are capable of providing color displays with simulated vector display.

General Information

The GRFEDT program allows the operator to select an area of the map database to view for the purpose of seeking and correcting errors. The area to be viewed is chosen graphically by selection of two opposite corners of a rectangle, called a **window** on the map data. The corners of this window are selected by pointing to their locations on the screen by use of the **crosshair cursor**, a graphics device built into the terminal.

The cursor is displayed as a single vertical line and a single horizontal line which intersect in a single point on the screen and which may be moved across the extremes of the screen by turning the **thumbwheel** controls. The thumbwheels are two small wheels which can be moved by fingertip to cause independent movements in the two lines of the cursor. The lines of the cursor are displayed in such a way that their presence and movement does not change the display of map data, essentially on a different display plane.

The terminal screen is divided into three distinct areas for the GRFEDT program. Most of the screen is used for the display of the map in an area representing a square in the coordinate system of the map, though this is not actually a square on the terminal. The lower portion of the screen contains information about the map database and the current display, including the map header, the map extremes displayed in the current view, and the contents of the current **mask** (see below). The third area of the terminal screen contains the conversation area, where the program and the operator converse in the selection and operation of commands.

The map display area on the terminal screen is of fixed size, but the area of the map to be displayed may be of any size chosen by the operator. (Thus, the scale of the displayed data may vary to virtually any value with the area selected). There is one fixed view of a map database which is recognized and used by the GIMMAP system. This is the view which transforms the range of the data in the map database, stored in what is referred to as **virtual** or **world** coordinates, to fit exactly the range of the map display area in its **screen** coordinates.

This maximal view of the complete database area displays all the map data at once in the greatest screen area possible. Other views of the complete map area may be selected, but their smaller scale would reduce the clarity of the detail of the map. It is exactly this maximal view of the complete map area for which the Qplot file has been created. Using the Qplot file, this view can

be displayed very quickly. This view is useful for initial views and for determining completion of the editing process, and is thus used frequently.

The selection of features for display may be restricted in order to focus attention on certain subsets of the map data or to reduce the amount of information on the screen to increase clarity. This is controlled by use of the feature code mask, which provides up to five code ranges of features allowed for display. The mask values are set by the operator selecting a low and a high feature code, defining a group of features which will then be displayed. Features which are qualified (i.e. whose feature codes lie in one or more of the ranges) will be displayed if they lie in the view being displayed. All other features are ignored.

In addition to the five basic code ranges of the mask, there are two additional ranges which are set by the GRFEDT program to force the display of basic, important features which assist in providing a frame of reference for the operator to recognize the setting of the map display. These are initially set to show the county and state lines and quadrangle boundaries, but may be reset by the user with proper access.

The values of the mask are displayed in the lower part of the terminal screen area, along with the map header and the minimum and maximum X and Y values of the current view. Each mask range is labelled by an index from 1 to 7, with the low and high limits for each range separated by a hyphen. Unused ranges are set to a high and a low of zero.

The GRFEDT program operates in either of two modes which provide a safer, more friendly and helpful environment for the learning operator and a more efficient, quicker environment for the experienced operator. In the **verify mode**, the program provides increased information, checking and verification of map errors and program actions, prior to modification of the database. Turning the verify mode off provides the fastest operation and requires the minimum effort in finding and correcting errors.

The GRFEDT program is generally operated on a copy of the map database files so that changes can be made to the current map as it is configured in the database. The use of a copy of the map database allows the operator to recover from errors made in the editing session or from catastrophic events such as system or program error. In such a case, the copied version is deleted and the old version is kept and only the current editing session is lost. In contrast, editing a single copy of the database might result loss of all the work done on the database. When an editing session is successful, only the new version is saved.

The GRFEDT program is operated by the use of a **menu** of commands which the operator may select to perform the desired editing functions. The menu consists of twenty-seven separate commands, each of which may be selected by an abbreviation of even

the first letter of the command name, with accompanying arguments to specify the objects of the command. The menu of the GRFEDT program contains the collection of modifying and utility commands of GRFEDT. A brief description of the commands is given below.

The GRFEDT Menu

Command	Use
ALL [NUMBER]	Display all unmasked arcs in complete-map view
BELL [ON/OFF]	Turn bell on or off for ringing to prompt user
BRIDGE node1 [node2]	Create an arc between two nodes
CODE node1 [node2]	Review and change an arc's feature code
DELETE node1 [node2]	Delete interior points of an arc
DISPLAY [MASK]	Display all arcs in groups by feature codes
END	Stop execution of the GRFEDT program
FRESH [NUMBER]	Erase and draw unmasked arcs in current view
HELP [command]	List commands with argument information
INSERT node1 node2	Add interior points to an arc
JOIN node1 node2	Merge two nodes into one, joining all connected arcs at a single endpoint
KILL node1 [node2]	Delete an arc and it's interior points
LOCATE [node1 [node2]]	Report location of the cursor, a node, or interior points in virtual coordinates
MASK [FRESH/RESET] [RESET/FRESH]	Set the feature code mask
MOVE NODE1 [NODE2]	Move a node or interior points of an arc
NUMBER [node/DEGREE/ZONES]	Post node numbers, node degrees or zone numbers for displayed features
PATH node	List node-arc pairs for a node, with feature codes, masking and in-view information
POINT	Create new isolated point arcs
RECLASSIFY oldcode newcode	Change feature codes for an arc group
RELEASE low-highcode	Delete arcs in a range of feature codes

REMOVE node/ALL	Delete unnecessary node(s), merging arc(s)
RESTORE [NUMBER]	Restore a previously SAVEd map view
SAVE	Save the current view definition for RESTORE
SPLIT node1 node2	Split an arc into two, creating a new node
VERIFY [ON/OFF]	Turn verify mode on or off, or toggle
WINDOW [NUMBER]	Select new map area from the current view
ZOOM [NUMBER]	Select new view with zooming window
ZV (invisible to Menu)	Turn on the Zone Viewing Option

The function of the GRFEDT program is to use the utility and modifying commands to correct errors found in the map features as represented in the database. There are many different kinds of errors which are to be found and corrected by the operator using this program.

GRFEDT Routines in Detail

ALL

The ALL commands provides a fixed quick view (via the preset Qplot file) of all the map data for the database. The preset view is that minimum area which encompasses the bounding rectangle defined by the map extremes which in turn are fixed by the projection parameters. All displays all map features whose feature codes qualified (i.e. not masked by the MASK). Arcs are displayed as lines from node through interior points to node, and points are displayed as plus signs. Certain features (map edge, county line, etc.) are set to appear as dashed or dotted.

ARCLMM

Calculates the length and extremes (X and Y min. and max.) for the full length of a line-type arc including all interior points. These statistics are in database units (usually digital units), are updated whenever changes are made to the nodes or interior points of arcs.

BNA

Sets the graphics terminal for alphanumeric output to a new (clear) line in the conversation area (a task which may cause

erasure of the screen and redrawing of the display when the conversation area is full), and rings the terminal bell if Bell mode is on.

BRIDGE

The operator selects (in the command) two existing nodes (which may be the same for duplicating arcs which begin and end at the same node - islands) which are to be bridged by a new arc to be created. The new arc will originally be a straight-line connecting the two nodes with no interior points. The operator must select a feature code for the new arc. The new arc is created, and the degrees and node-arc pairs for the two nodes in the Node file are updated. If the two nodes were previously connected, the new arc may be as described above oar may duplicate the path of the existing arc, but must have a different feature code. If Verify mode is on, the two nodes are verified prior to bridging.

CHKNDS

Uses the SHWPT routine to verify the selection of a pair of nodes for an arc prior to performing a modification. Both points must be verified in order to signal verification to the calling routine and thus both must exist in the current view.

CODE

Code is used to change the feature code of an arc. The arc is selected by the node numbers (or single node to isolated points) and the selection of the arc is verified by the node or nodes selected. If the arc is line-type arc, the line is redrawn as a bold dashed line using the SHWARC routine three times in rapid succession to allow the operator to locate and verify. If the feature code is one reserved for special drawing (map edge, county etc.), the user must pass the UNLOCK (see UNLOCK routine) test to verify authority to make the change. The operator selects a new code which may not attempt to change an isolated point arc to a line-type or vice-versa. Finally, the point or a part of the arc must lie inside the current view or the change can not be made. When all has been accepted, the feature code of the arc is modified in the database.

The CODE routine is used by the RECLASS routine to modify the feature codes of classes of features in a single command. When this is done, CODE is called for each arc to be changed with the arc number (-1 * the arc number as the node1 argument) and the new feature code (as the node2 argument). No dialog is performed, and the code is changed if the basic requirements are met.

CODMSK

Compares a feature code to the code mask to determine if the feature code is masked or unmasked. Also, the indicator returned to the calling program shows if the feature code is one for the special class of arcs (map edge, county, etc.) which are displayed as special lines and always displayed as framework re reference lines for the map.

DELETE

Delete is used to remove one or more interior points from a line-type arc. Thus, the arc selected must have at least one interior point. The arc is selected by naming the two end point node numbers, which may be the same in the case of islands. For the arcs whose codes put them in the special class along with map edge arcs, the UNLOCK procedure is required prior to access.

Points to be deleted, usually due to being clearly in the wrong location, (such as spikes), or points in excess, are selected by use of the graphics terminal cursor (a pointing device). Along with the selection of the point selection, the operator types cursor control character to initiate selection. If verify mode is on, the interior point of the selected arc which is nearest the cursor location is verified by the SHWPT routine. Once selected, (and verified), the point is removed and the remaining points are shifted appropriately. The arc, point and display (Qplot) data are updated completely.

If the cursor control character was 'L', exit occurs after deleting the point. If it was 'E', exit occurred before (immediately). Otherwise the routine cycles to allow selection of more points for deletion.

DISPLA

The DISPLA routine provides a controlled display of all features in the database in groups by single feature codes. This is, all section corners (f.c. = 1111) are displayed. Following this, all township corners (f.c. = 1112) are displayed. All displays are the full (Qplot) display of the entire map area. After each code is displayed, the program pauses for inspection or for the operator to initiate a hardcopy of the screen. At this time the operator may escape the command by entering the escape character in the angle brackets of the prompt: <escape character> which in this case is a zero.

The codes displayed by DISPLA may be one of two options. The default is to display all features of all codes in the database. If the MASK option is chosen, only those codes which are specified in the (seven) ranges of the mask will be displayed. When the last display has been completed, DISPLA will restore the screen to the

view prior to execution of DISPLA.

FILOPN

Opens the four basic files of the map database as generic files (ARC NODE POINT QPLOT). Thus, the files to be used by GRFEDT must be renamed to these names prior to execution and returned to the original names when finished.

FINDPT

Search for the interior point of a specified arc which is closest to a point selected by the operator pointing with the cursor. The normal (Euclidean) distance is calculated for all points in the interior point chain, and the location (record number and offset) of the closest to the selected location is maintained. This routine is called by other routines which perform editing operations requiring selection of an interior point,such as DELETE, MOVE, and SPLIT.

FRESH

Erases the screen and creates a fresh view (including all prior modifications) OF all features within the current view whose feature codes are not masked. The display includes a frame box (for clipped windows) and textual information describing the Bell Mode, Verify Mode, Projection Type, Mask settings, map scale, map header (from the last enter DDF file), and the extremes of the current view. The FRESH routine is called by all routines which produce new or changed view (e.g. ALL, WINDOW, ZOOM,...). The view to be displayed is adjusted to a square (in user and screen coordinates) to prevent distortion of shapes and sizes. Special line types (dashed, dotted,...) are used for special codes (map edge...) and features are drawn form the display (Qplot) file for the special whole map view, or are clipped for other views.

HELP

To display one or more command names with associated parameters to assist the operator in using the GRFEDT program. The HELP command, given alone, will produce a complete listing of all GRFEDT menu commands with arguments and rules for interpreting the listing. If the name of a command is given, HELP will list only that command, and when a single letter or two are provided, all commands which begin with that letter (or letters) will be displayed.

If Verify Mode is OFF, only the command lines are displayed with associated arguments. When Verify Mode is ON, each command line is preceded by a line of text which defines the general use

of the command. When the complete menu is given, a set of rules accompanies the list. Included are (1) commands may abbreviated by one or more letters, (2) Arguments in UPPER CASE are literal,(3) Arguments in Lower case require a number or none, (4) Commas may be used where blanks are shown as separators, (5) Optional arguments are shown in[square brackets], and (6) Options are separated by a slash, such as [option1 / option2] ... meaning that either option 1 or option 2 or neither may be used.

INSERT

Insert allows the operator to add interior points to an existing line-type arc. While viewing the arc (or a portion), the operator points to places where new points are to be added, to change or define the path of the arc. Each new point is inserted between the closest two points to the selected location (chosen via the cursor) if the two points are consecutive. If the closest points are not consecutive, the newly added point is inserted next to the point which is closest to the selected location.

The arc endpoint nodes are compared to each selected point location as wells as the interior points, since a new point can be inserted next to either of the two nodes. If the selected arc is one of the special arcs (such as a map edge arc) then the UNLOCK procedure is required to allow the operator to proceed. When the point to be inserted is selected and verified (if MODEM is ON), it is added to the interior points of the arc, creating new records if necessary (and updating the Qplot and Arc file). In the selection of the point, the character typed (the cursor control character) will control the action taken. If the character is 'L' (last) the routine is ended after insertion. If the character is 'E' (exit) no insertion occurs. All other characters cause insertion and cycling to select additional points.

JOIN

Draws together two separate nodes (with connected arcs) to form a single node at which all the arcs from the two nodes are incident. JOIN is used to close gaps between arcs which may not have been digitized close enough to their actual intersection. It may also be used to connect arcs on which nodes were not originally digitized but were later created through SPLIT routine. Both nodes must have at least one unmasked arc in the current view, and if Verify Mode is ON, both nodes must appear in the view.

When the join occurs, the node-arc pairs of the second node are added to those of the first. The second node is removed while the first remains. If the two nodes are connected, the operator must explicitly force the join. The location for the merged node may be one of the original node locations or an average of the two, depending on the edge status of the two nodes and the desire of the operator. If both nodes or edge nodes (lie on the map edge or

other arcs) the operator selects one original location to be used. If only one is on the edge, the operator may select either original location or the average.

KILL

The KILL routine is used to delete an arc which lies between two specified nodes or is an isolated point-type arc specified by one node. The arc to be deleted must be unmasked, and is drawn by SHWPT or SHWARC for operator verification before the deletion can be performed. Once verified, the arc record is removed and associated node records are modified to remove the node-arc pairs for the arc, and may be deleted in some cases. Interior points of line-type arcs are also deleted.

The KILL routine is also called by the RELEASE routine as it deletes all arcs in a range of feature codes. In this call, the first node number is the negative of the arc to be released. For this case, no verification is performed, and only the actual release of the arc record and any associated point records is done. The node record updating for all nodes is accomplished by the RELEASE routine after calling KILL for all the arcs to be deleted.

LOCNOD

Called as a result of the operator selecting the LOCATE command with a single node number, the LOCNOD routine provides the location (in user X,Y coordinates) of the specified node. If Verify Mode is ON, the node must lie in the current view and is verified via SHWPT. If Verify Mode is OFF, the node may lie outside the current view.

LOCPNT

The LOCPNT routine is called for the LOCATE command when the operator has selected two nodes or none. If two nodes are selected, an arc is specified, and the routine will return the user coordinates for the interior point closest to the location pointed to by the cursor. If no nodes are specified, the routine will provide the user coordinates of the cursor location wherever the operator places it. If Verify MOde is ON then a selected arc must be in the current view and be verified by the operator.

The cursor control character determines the action of the LOCPNT routine. If it is 'E' (for exit) the routine does nothing further. If it is 'L' (for last) the location of the next selected point is reported, and LOCPNT returns to the calling routine. Any other character continues the cycle of selecting a point and reporting the location.

MAIN

The MAIN routine initializes global variables for file management, the Bell and Verify Modes, and the feature code Mask with codes for the special (edge, etc.) arcs. MAIN opens a number of files and calls FILOPN to open the database files and calls GMCINI, GMBINI, and RMIOFC to initialize graphics management and random access file management routines. The database is accessed to gain accounting information (number or records, etc.) about the files, to gather arc feature codes and extremes, and to set the end point node numbers for all the arcs.

The extreme values for the map are used to set the initial view - the maximal view that shows the entire map area at the greatest possible magnification (largest possible scale), and to set this view in the graphics software. The terminal input and output files are opened for a third time (first for secondary dialog, second for map data display) to support command selection and communication.

Finally, the screen is erased and the basic background information is drawn (edge arcs, etc.) as a reference to future operations. Then the SELCOM (select command routine is called to initiate the editing session. From then on, SELCOM controls the flow of the program until the END command returns control to MAIN which ends the session.

MASKS

The MASKS routine is called to reset the values in the feature code MASK which is used to screen out unwanted and unnecessary features from the map display. Only those arcs which have feature codes within at least one of the seven code ranges in the MASK will be drawn in the display. If the operator selects the RESET option in the command, then the Mask is set to a different position in which all arcs (codes 0-19999) are allowed, including the basic set of special (map edge, etc.) arcs. Otherwise, the operator must select a row in the Mask for which a new low and high code will be given to define a range of codes for inclusion in the display.

The operator may select as many rows and set as many ranges as desired. When all have been set, the row is selected as zero (0) to end further definitions. When the ranges have been set, the option to renew the display by calling the FRESH routine is offered, unless the Fresh option was selected in the original command line. If selected, the screen is erased and drawn according to the new Mask.

MOVNOD

Called when the operator selects the MOVE command with a single node, the MOVNOD routine allows the operator to change the

location of a node selected by number. The node must lie in the current view and must have at least one incident arc whose code is not masked. If the node lies on the map edge (or on any of the special arcs), the UNLOCK procedure must be performed before any modification can occur.

Once the selected node is checked for these criteria, (and verified when Verify Mode is ON), the operator may select a new location for the node (also verified if Verify Mode is ON) which is then relocated to the new location. When complete, the new location is updated in the node record as well as in the arcs connected to the node. Changing the node location changes the length and extremes of the arcs, which must be re-calculated.

MOVPNT

The MOVPNT routine is called when the operator selects the MOVE command with two nodes, indicating the desire to change the location of one or more interior points of the arc specified by the two nodes. The arc must have at least one interior point, be unmasked and, if it is a special (edge, etc.) arc then the UNLOCK procedure must be completed.

The operator selects an interior point to be moved by pointing with the cursor. The FINDPT routine is used to determine the closest interior point to the position of the cursor, and this point is verified via SHWPT if Verify Mode is ON. The operator then selects the new location for the interior point by pointing with the cursor. The change is then displayed as a flashing line from the preceding point to the new location and back to the succeeding point, to demonstrate the new path of the arc. If accepted, the change is made.

As with other commands, the character typed when selecting a location with the cursor (the cursor control character) determines the actions of the MOVPNT routine. If this character is 'E' no moves occur. If it is 'L', then a single point is moved and control returns to the menu. All other characters cause cycling from point selection to new location specification to modifying the database.

NODARC

The NODARC routine is called by all routines which perform operations on line-type arcs to find the arc number of an unmasked arc connecting two selected nodes. When more than one arc exits between the two nodes, the last one found in the records of the first node is the one selected. The arc number is returned as a positive value if the arc begins (as digitized) at the first node and as a negative when the arc begins (as digitized) at the second node. If no unmasked arc is found, the arc number is returned as zero, which is not necessarily an error. Two edge indicators are also set to show if the nodes are connected to one or more special

(edge, etc.) arcs.

A mask value is returned to show if the selected arc is one of the special (edge, etc.) arcs or not. If no unmasked arcs are found between the two nodes then the mask value is set to the total number of arcs connecting the two nodes (including those that are masked), or, in the case of an error in the database, the mask value is set to define the file error.

A special message value is sent to NODARC routine to indicate whether error messages are to be printed by NODARC or are reserved for the calling routines. Some errors messages are printed regardless of the message value. In the event of file errors, a negative value is returned to the calling routine through the message value to indicate the type of error which has occurred.

NULINE

The NULINE routine moves the cursor in the Conversation Area of the display to the next line to prepare for input or output. The routine generally moves the cursor a single line, but may be directed to move it through multiple lines to adjust to multi-line output. If there is not enough room in the Conversation Area for the desired number of new lines, the display is erased and renewed via FRESH. This resets the current line to the top of the Conversation Area.

NUMBER

The basic purpose of the NUMBER routine is to post node numbers beside (so the node is at the lower - left corner of the number) the nodes in the current view. When the NUMBER command is selected with no parameters, all nodes in the current view are numbered. This numbering provides the reference (record) numbers that allow the operator to select a node or to identify an arc (by two node numbers) for editing operations.

There are three additional options for the NUMBER command and routine. If the operator wishes to mark a single node by its number, that number is given as a parameter to the command. The DEGREE option causes all nodes in the current view (with at least one unmasked arc) to be numbered with the degree of the node - the number of arcs connected at the node. This option assists the operator in locating missing nodes and unjoined arcs to be corrected.

The fourth option is ZONE, which may only be used when the databases has an associated zone file and which has been enabled by turning on the ZON (Zone Viewing ON) parameter with the ZV command, prior to attempting this option. If Zone Viewing is ON, the (center-of-mass) zone marks for all zones in the current view are drawn as plus symbols and the zone numbers are posted at these

marks. The value of this option is that regions of the map area may be edited by selecting the zone (region) so that a new view will be based on the extremes of the zone. Thus, editing may be done on a county - by - county basis by selecting county zones by their posted zone numbers.

PATHS

The PATH command requires a single node number as it's parameters. This number is then sent to the PATHS routine which collects the complete list of nodes connected to the selected node with the associated arc numbers. Each arc number is set positive if the arc begins at the selected node. The feature code of each arc is collected along with indicators(Y=Yes, N=No) to show if the arc is in the current view and if the arc is masked. The values for each node - arc pair are displayed in the conversation area. This information is valuable to the operator for determining the structure of the map features and for identifying the correct arcs to be edited.

POINT

The POINT routine is called to create one or more isolated point arcs at locations specified by the operator pointing with the cursor. The operator selects any location in the current view with the cursor. The cursor control character (key pushed to select the location) determines the action. If it is 'E' no further action occurs. If it is 'L', the location selected will be the last isolated point created. Any other key will cause the routine to cycle for another point after this one is created.

If Verify Mode is ON, the selected location is verified via the SHWPT routine. Next, the operator must supply a feature code for the new arc. At this point, the escape character of zero is shown in angle brackets (<0>) to indicate that the command can be exited by typing a zero. The feature code must be that of an isolated point - between 1000 and 1999 or between 11000 and 11999. The new arc and node records are created and added to the database.

QPLOT

The QPLOT routine is used to display a single arc for the special (Qplot) complete - map display at the maximum scale. For this display, the point coordinates for all interior points are already calculated and stored in the Qplot files, since they are fixed for the special view. Thus, the records for the arc need only be read and written to the graphics display terminal. No clipping is required because all points are guaranteed by the view to be shown.

This routine uses the buffered graphics facilities of the graphics management library (GRFMAN) to produce the fastest possible display of the complete map in the special view. Lines have been masked by the calling routine, and special line types are requested through a single argument.

QUPDAT

Whenever interior points are moved, added or deleted, both the point file and the qplot files must be updated. The qplot file contains the parallel screen coordinates (based on the special, full = map maximum scale view) as converted from the user coordinates in the point file. The QUPDAT routine performs this conversion to update the qplot file for an arc or (if given a negative starting record number) to update a single (modified) record.

RECLASS

The RECLASSIFY command requires an old feature code and a new feature code. The RECLASS routine uses these values and calls the CODE routine to change the feature code of all arcs which have the old value to the new value. The old and new codes must be of the same type (isolates point on line - type arcs).

Since this command will cause extensive changes in the database, the UNLOCK procedure is required prior to the operation. If Verify Mode is OFF, all qualified arcs (whether masked or unmasked, in the current view or not) will have old codes changed to the new code. If Verify Mode is ON, each arc must be verified by the operator. If an arc is rejected, the operator will be given a chance to exit the RECLASSIFY command. If the routing changes all the arcs with the old code to the new, the number of these arcs is reported.

RELEASE

The RELEASE command parallels the RECLASSIFY command but performs arc deletion rather than changes in the feature code. This command requires a low and high feature code (separated by a hyphen) to define a range of arcs which is to be deleted. The use of this command is to remove unwanted groups of arcs without spending a great deal of time to locate, identify and verify each one. The RELEASE routine uses the KILL routine to perform the deletion of each arc and it's points, then performs the update of the node file after all arcs are deleted. Only arcs that have feature codes in the specified range which lie (at least in part) within the current view and candidates for deletion.

Due to the power and potential damage of this routine, the UNLOCK procedure is always performed. Next, the operator is

required to specifically force Verify Mode to OFF if verification is not desired. When Verify Mode is OFF, each arc is checked to see if at least part of the arc lies in the current view. If so, KILL is called to delete the arc. When Verify Mode is ON, the KILL routine will use the SHWARC routine which checks to see if the arc lies totally outside the current view. If it does, the operator is given the option to exit the RELEASE command. If at least part of the arc lies in the current view, the operator verifies the arc after it is flashed on the screen, and then the arc is deleted. If the verification is denied, the operator may also escape RELEASE.

When all arcs in the range have been released, the node - arc pairs in the associated node records are removed and the node degrees adjusted. The RELEASE routine ends by reporting the total number of arcs deleted in the Conversation Area.

REMALL

The REMALL routine is called when the ALL option is selected with the REMOVE command. The object is to remove nodes of degree two (two arcs intersect at the node) when the codes of the two arcs are the same, which may occur by incorrect digitizing (points too close) or as the result of arc deletions or other editing operations. The REMOVE routine deletes a single such node, joining the two arcs to a single arc. The REMALL routine removes all such nodes only in the current view (if Verify Mode is OFF or there is no moving view), saving much time for the operator who usually does not need to verify this operation.

The REMALL routine always requires the UNLOCK procedure because of the potential for damage to the database. If Verify Mode is ON, the operator is offered an option for each node to be displayed in a moving, close - up view for verification - an option that may be required to clearly examine each case. If selected, the operator may specify the size of the close - up view. If the moving is selected, all nodes in the database that qualify can be removed. If no moving view is selected (or Verify MOde is OFF) only the nodes in the current view may be removed. When Verify MOde is ON, each node is verified via the SHWPT routine. If any node is not verified, the operator is given the option to exit the REMALL command. At the conclusion of the command, the total number of nodes removed is reported in the Conversation Area.

REMOVE

When nodes exist at the joint, of exactly two arcs of equal feature code, the nodes may or may not serve a legitimate function. They may have been created to break a long arc into manageable pieces or may result from mistakes made in digitizing. Nodes of degree 2 whose arcs are both of the same code may be removed via the REMOVE command. In this process, the two arcs are joined

together by merging the interior points (reversing one arc if necessary) into a single point chain and adding the node location as a new interior point. The REMOVE command is essentially the inverse of the SPLIT command.

If Verify Mode is ON, the node selected by number, is verified via the SHWPT routine. The feature code of the two arcs to be merged must be the same and must be unmasked. Furthermore, the node must not be connected to itself (by an island arc).

The REMOVE routine is called by the REMALL routine to remove all qualified nodes in a view or in the map (Verify Mode = OFF). In this case, the node number of - 999 is passed to REMOVE, which gains appropriate information through values in the COMMON storage areas.

REVARC

When the REMOVE routine must reverse an arc in order to merge two arcs, the REVARC routine is called. REVARC swaps the locations of the two endpoints of the arc, reverse the string of interior points both within Point records and in the order of the Point records, and produces the appropriate parallel order in the display (Qplot) file.

SELCOM

The SELCOM routine governs the selection and execution of commands for the GRFEDT program. SELCOM accepts input from the operator through the keyboard, where each character typed is added to a command line that is terminated by a newline (or carriage return) or after receiving eighty characters. Within each command line, the operator may have entered one or more commands with associated arguments, separated by semi - colons.

Each command in the line must contain one or more letters specifying the command type, followed by arguments identifying the objects of the command. The arguments are either alphanumeric or integer values, with alphanumeric values possibly abbreviated by 1 or 2 characters. Command names may be abbreviated by as little as the first character, and are distinguished by the nature of their arguments.

When the command line has been analyzed to identify the command type and to evaluate the arguments, the indicated routine is given the arguments, and the desired function is performed. Upon completion of a command, control passes back to SELCOM. When the END command is executed, SELCOM passes control back to the MAIN routine to terminate the editing session.

SHWARC

When the selection of an arc for editing is to be verified by the operator because Verify Mode is ON, the SHWARC routine is used. SHWARC displays the selected arc as a flashing, dashed line by means of the VPLOT routine for clipped windows or the QPLOT routine for the unclipped, special view of the whole map. Flashing is accomplished by replotting the arc multiple times in temporary (WRITE-THRU in TEKTRONIX) mode. The operator is asked to accept or reject the arc. If the arc is selected does not lie, at least in part, within the current view,is rejected by SHWARC after posting an error message when Verify Mode is ON.

SHWNEW

When an interior point is added or modified and Verify MOde is ON, the SHWNEW routine is called to show the proposed change, similar to the SHWARC routine. SHWNEW draws a flashing (multiple drawing in temporary mode), dashed line from the preceding point to the point being edited, to the succeeding point, the operator is asked to select an option from accepting, rejecting or retrying. If the center point does not lie in the current view, rejection is automatic following an error message when Verify Mode is ON.

SHWPT

When editing operations require verification of a node, an interior point or a screen location, the SHWPT routine is used to create a flashing dashed hourglass symbol and a cross - cursor (a horizontal and vertical line intersecting at point), both centered at the point, to identify the point to the operator. The flashing symbols are generated by re - drawing the symbol multiple times in temporary mode. The operator may select or reject the point. If the selected point is not in the current view, the SHWPT routine rejects the point with an error message when Verify Mode is ON.

SHWVW

The selection of a rectangular area for windowing to a new view is verified (when Verify Mode is ON) by the SHWVW routine. A flashing dashed rectangle displays the new view area, or at least that part which is within the current view, by redrawing the rectangle multiple times with temporary mode. The operation may accept or reject the new view.

SPLIT

The SPLIT routine provides a mechanism for the operator to create nodes within arcs where they may have been overlooked in the digitizing stage. The arc to be split into two arcs at a

selected interior point is specified by its two endpoint nodes. The selected arc must have at least a part in the current view, must be unmasked, and must be verified by the operator when Verify Mode is ON. Also, if Verify MOde is ON, the selected point must be verified.

When the arc is split, the interior points are divided at the selected point with the second part disassociated from the original arc and associated with a new arc. A new node is created at the point selected for the split, which has been removed from the interior points of both arcs. The feature code of the new arc is set to that of the original arc, and statistics are re - generated for both arcs. If the original arc had no interior points, the arc is split and the new node is located at the point selected by the cursor.

TOKEN

Given the command line from SELCOM, the TOKEN routine examines the character at the position specified and returns an indication of the type of token (command element) found there. Among the options are letters, digit,semicolon,newline,hyphen and comma among others.

UNLOCK

Certain options in editing are restricted to control their use, due to power of the operations and the potential to significant damage to the database resulting from improper use. In addition, certain special features (map edge, etc.) may be protected from modification by unauthorized operators. Both these functions are provided by use of UNLOCK procedure, as implemented in the UNLOCK routine.

This routine is essentially a dynamic password procedure which requires the operator to enter a sequence of five or more words, each of a minimum or greater number of characters. Given such a sequence (which is not echoed when typed), UNLOCK randomly selects a character in each of these words, printing the sequence number of each word along with the sequence number of the selected character for each. This results in six numbers presented to the operator, who must correctly enter the three selected characters in order.

If the operator successfully performs this procedure then the denied operation is allowed. If the operator does not get the three characters, the routine responds with 'OK?' which is a signal to restart the procedure by entering the password phrase of five or more words. Improper entry of the phrase causes rejection of the request.

VIEWRS

The VIEWRS routine is called by SELCOM to initiate the functions implied by the RESTORE or the SAVE commands. The SAVE command causes VIEWRS to save the extremes of the current view for possible later restoration by the RESTORE command. When VIEWRS is called for the RESTORE function, the current view is set to the saved view and FRESH is called to erase the screen and change to the save view. If RESTORE is attempted without a view having been saved, an error message in printed and no action is taken. Optionally, the FRESH call for the RESTORE function may include a request to number the nodes.

VPLOT

The VPLOT routine is used to draw an arc when the current view requires clipping (i.e. is not the special view). If at least part of the arc lies in the current view, the VPLOT routine will plot this part by converting the arc endpoints and interior points from the point file into screen coordinates, clipping at the current view boundary where necessary. Arcs may be drawn as solid lines or in temporary mode as dashed lines.

WINDOW

One of the two commands to change the map view is to use WINDOW. The WINDOW routine supports from basic options for selecting the new view, all of which provide some method for selecting four corners of a rectangle which defines the new view.

The basic option for selecting a new view is to use the cursor to select opposite corners (NW and SE or SW and NE) of the defining rectangle. These corners must lie within the current view; hence, the selected area will be smaller than the current view. The selected rectangle is verified when Verify Mode is ON, and the rectangle is adjusted by increasing its smallest dimension to equal the other dimension. Forcing the dimensions to be equal eliminates distortion in shape and scale which otherwise be present. A second option is to select the corners of the rectangle by entering their X,Y locations at the keyboard. Using this option, the new view can exceed the size and the boundaries of the current view, and need not even overlays the current view at all.

Another option for selection of the new view is to use the cursor control characters to specify a rectangle of the same size as the current view , but located in a direction specified by the cursor control characters. The direction specified by the cursor control character is one of the eight points of the compass: N = North, S = South, W = West, E = East, NW = Northwest, NE = Northeast, SW = Southwest, and SE = Southeast. The new view selected is define by the rectangle of the same size moved in the indicated direction with no overlap with the current view.

When overlap is desired to provide continuity and assist in recognition of features, the OVERLAP option may be selected. In this case, the new view is defined as above, except that it is moved only half as far. That is, the new view will overlap the current view (half for N,S,W, and E and a quarter for NW, SW, SE, and NE). Thus, the southwest corner of a view created by NE will be at the center of the current view if OVERLAP is on and at the Northeast corner of the current view otherwise.

The final option for selection of the new view is to select a zone and use the extremes define for that zone. Such an option allows editing by counties or other sub - areas to better organize and monitor editing operations. For this option to be available, the database must include a Zone file with valid zones constructed by the ZONGEN program and operator must have executed the ZV command to initialize zone viewing. When the operator selects the zone view option by typing Z as the cursor control character, the operator is asked to give the number of the zone to use. (When Zone Viewing is ON, zone numbers may be posted by the NUMBER command). As with the other options, the view rectangle is altered to prevent distortion and may be verified if Verify Mode is ON.

ZOOM

The ZOOM command allows the operator to select a new view by specifying the location of the center of the view and the size of its two dimensions. These two values define a rectangle similar to that of the WINDOW command, described above, and the consequence is the same. There are four methods to specify the new center of the view. The size is a single value in user coordinates (usually inches). Upon execution, the call to FRESH may include the option to number the nodes.

Selection of the new center is accomplished principally by selection of a point within the current view by means of the cursor. This method forces the new center to lie within the current view and for the new view to overly at least one quarter of its area with the current view. A second option allows selection of the new center as the location of an existing node. This option is triggered by using N as the cursor control character and specifying the node number when prompted. If the node exist, its location will be the new center. When Verify Mode is ON, the node must lie within the current view. But when Verify Mode is OFF, the node may lie anywhere.

A third option, signalled by S as the cursor control character, allows selection of the same center as the current view. This option allows the operator to zoom in or out on the same area as is displayed. The final option is to enter X as the cursor control character to signal that the new location is to be entered at the keyboard.

II. B A S I C S E Q U E N C E

II.7 PLTGEN - THE PLOT GENERATION PROGRAM

Introduction

The PLTGEN program is used to create edit plots from a map database on the vector plotter. These plots differ from those made for actual map products in several ways. The options available for line types and point symbols, feature colors, area and scale of map, and content selected are fixed to the edit plots. Only a basic subset of features are plotted for a single area at a fixed scale, with fixed symbols and minimal annotation.

The purpose of the edit plot is to provide the operator with a graphic representation of a database which may be compared to original source documents. To this end, the base name and projection information is used for the map title, special arcs have fixed line types or point symbols, nodes may be numbered with record numbers or node degrees. Fixed pen positions imply fixed set of colors assigned to groups of features.

The routines used by PLTGEN are:

CHKLIM - Adds offsets if selected

FILOPN - Opens database and plot files

GMPPEN - Change pen (1 of 4)

LNGSHRT - Plots dashed or long-short-short dashed line

MAIN - Sets up, initializes, calls FILOPN, offers title, offset and node degree options, creates fiducials, fiducials, initializes graphic routines, plots all arcs and calls Number.

MARK - Uses the TCORN1 and TCORN2 routines to plot the fiducials at the corners of the map

MHD2 - Secondary line dashing routine

NUMBER - Plots node numbers or degrees

PLTARC - Plots a line - type arc

TCORN1 - Plots outer triangular fiducials before plot

TCORN2 - Plots matching inner triangle after plot

TITLE - Sets up map title and calls GMPTTL to plot it

To run the PLTGEN program, the operator must supply the names of the database and plot files, the map tile information, and must select the node numbering option to use node record numbers or node degrees and the X and Y offsets. Descriptions of these routines are given below.

CHKLIM

All points that are to be plotted are checked to be positive so they can be plotted. The CHKLIM routine tests all points and adds the X and Y offsets to translate the map as defined by the operator.

FILOPN

All database files (ARC, POINT, NODE) used by the PLTGEN program are opened in the FILOPN routine. The operator is queried for the names of these files, which are opened as direct access files of varying lengths. The FILOPN routine also request the name of the single, plot file and opens it. In addition, the two Hershey alphabet files are opened to support generation of the map title and node numbering.

GMPPEN

The GMPPEN routine, a duplicate of a routine in the graphics management library - GRFMAN, changes the pen selection on the vector plotter to one of four possible positions. The purpose of GMPPEN is to provide control over the color applied to different map features.

LNGSHRT

The LNGSHRT routine provides the drawing of dashed and long-short-short dashed lines for special feature codes in the database. LNGSHRT produces plot output for a single arc following the path of the interior points of the arc.

MAIN

The MAIN routine calls the FILOPN routine to open all files and then sets up the titling option when desired. Then, the node numbering option is selected and the translation offsets are entered. The coordinates of the map area corners are calculated for maps with or without projection. These corners are used to locate the triangular registration marks, lying completely outside the map area. The plot signs are checked against the plotter size and the plot file is initialized by a call to GMPINI. All the arcs

are plotted with appropriate color and symbolization (color as selected by the plotter operator). Finally, TITLE is called to post the title, NUMBER adds the node numbers, MARK finishes the registration marks, and GMPFIN is called to close off the plot file.

MARK

The MARK routine is given the corner locations and told if the call is for the first or second pass. In the first pass, TCORN1 is called to draw a triangle at each corner point. These triangles are drawn so that one vertex in coincident with the corners, and the two sides at that vertex are horizontal and vertical. The opposite side is the perpendicular to a line from the center of the map to the corner. (That is, the SW triangle outer side runs NW to SE). When MARK is called at the end of the plot, TCORN2 is called to draw triangles at the corners by connecting the midpoints of the sides of the first triangles.

MHD2

The MHD2 (HD = Hashed/Dashed) routine, borrowed from the MAPGEN program is a secondary routine for drawing dashed and long-short-short dashed lines for special features (e.g. county lines). The MHD2 routine continues dashing from one point to the next, according to the specifications given for the desired dashing.

NUMBER

The NUMBER routine produces plots of numbers at all nodes by determining the number and passing the information to the GMPNUM routine in the graphics library. The number selected is either the node record number or the degree of the node. The arcs which intersect at the node are examined and the appropriate color is selected.

PLTARC

Using the GMPMOV and GMPDRW routines in the graphics library, the PLTARC routine converts the endpoints and the interior points of a linear arc into plot output. All points are dashed and translated via the CHKLIM routine.

TCORN1

The TCORN1 routine produces a triangle (described above in MARK) at the specified corner, using offsets provided to find the corners.

TCORN2

The TCORN2 routine draws a second triangle at the map corners as specified (see MARK) to correct the mid point of the sides of the triangles drawn by the TCORN1 routine.

TITLE

The TITLE routine sets up the titling of the maps being plotted, using selections made by the operator in the MAIN routine. Default values are set for the start location, character height and orientation, and GMPTTL creates the title.

II. B A S I C S E Q U E N C E

II.8 ZONGEN - THE ZONE GENERATION PROGRAM

Introduction

The primary function of the ZONGEN program is to construct the boundaries of the areal features (polygons) of the map from the topological information in the Node file (connections between nodes via arcs). The zone information is the topology of the areas of the map and is constructed from the Node file. The results of ZONGEN are the Zone file containing one record for each zone, and the Border file, containing one record for each arc that was used for zone boundary generation.

The process performed in ZONGEN is provided in detail below, but the basic process involves four operations or phases. The task is to find paths from a starting node around the boundary of a single zone and back to the start node for all areas, including islands and the outer border of the map. In GIMMAP, the zone boundaries are generated in the counter-clockwise direction. Each selected arc, qualified by operator-selected code ranges,acts as part of the border for two adjacent zones. In one, it is used in a positive sense - as digitized to be a part of the counter-clockwise boundary. For the adjacent zone, it must be used in the negative sense - . The reverse direction from digitization.

In the first phase, the selected arcs are collected each with an associated Border file record, and analysis of the endpoint and interior points generated the contributions made to the zone area and center-of-mass (initial zone mark or ID location). Also, the angles of incidence of the arc at the endpoint nodes are calculated as the angles formed by the first and last interior points (or other nodes if there are no interior points). These angles will provide the only information necessary to perform a "left-turn" at each node to guarantee generation of a single, counter-clockwise boundary. The statistical information generated in phase I is stored in a temporary, direct-access file to use in later phases.

In Phase II, the node-arc pairs to all selected arcs are collected from the Node file grouped according to the common node and ordered within groups so that the angles of incidence at the common node are in increasing order. This will allow a simple mechanism for performing left-turns by exiting a node via the node-arc pair above that of the entry. That is, exit via the arc whose angle of incidence is the next smallest (exit via the largest for the arc with the smallest angle).

In Phase III, the collection of node-arc pairs, known as **links**, is analyzed and the boundaries of the zones are weaved by following the connected links from a start node, around the zone

and back to the start node. The zone statistics are collected from the temporary file along with the mapping from Border record numbers to Arc records. The Border file record information is collected and stored, to be written in the rest phase.

Also in Phase III, ZONGEN checks for a number of special cases which require appropriate actions. First, two kinds of errors may exist in the database to prevent the proven generation of zone boundaries. The first is the **missing node** error. Another is the **wrong turn** error. The missing node occurs where boundary arcs intersect on the map, but were not digitized (or edited) properly to reflect the intersection in a node. The wrong turn problem is detected when boundary generation attempts to visit a node which is not the start node for a second time in the zone boundary. (there are some unusual cases in which this is permitted - but these may be altered to fit the normal rules).

The missing node and wrong turn errors are detected in situations in which the exact cause might not be determinable. When found, the error is reported along with information on the zone being generated and the node-arc pairs found. Then the ZONGEN program hats, since complete processing of zone boundaries is not possible until the error has been corrected.

The third phase also recognizes the existence of **simple island** zones, as zones of one arc - These are surrounded by another zone. The simple island zone must still describe the zone in a counter-clockwise manner. The island OF pointer for simple islands is set to the zone number of the island itself to distinguish it from non-island zones.

Phase III also recognizes the **complex island** zones. These are very special zones which arise from a specific physical situation as a natural result of the ZONGEN algorithm. The complex island is a group of two or more zones which share common borders, but which are as a group completely surrounded by another zone. That is, the <u>outer boundary</u> of the group of zones (none of which is an island) represents an island in the surrounding zone.

The complex islands always generate their boundaries in a <u>clockwise</u> direction - the opposite of other zones. Thus, the area generated for them is negative. This fact allows easy recognition. These zones arcs masked by setting the Island OF pointers to the negative of the zone number for the complex island. Complex islands are kept since they represent a proper part of the boundary for the surrounding zone. The largest single complex island is the outer boundary of the map.

When the complex island has only one link, then it is actually a reversed duplicate of a simple island zone. In this case, there is no need to keep such complex islands, when the corresponding simple island exists. Thus, the single arc complex island is discarded.

The fourth phase creates and writes the record of the Border file, setting the color code for the left and right zones. Once the file is complete, a report of run - time statistics is printed.

The algorithm for zone operation in detail follows a brief description of the ZONGEN routines.

ANGLE

The ANGLE routine is used to calculate the angles of incidence of the first or last interior points (or opposite node) of the arcs intersecting at a node. The incidence angles may then be used to determine the correct exit arc to perform a "left-turn" for a path entering at the entry arc. Angles are calculated in positive radians less than two pi.

ASTATS

The ASTATS routine is given the endpoints and Point file pointer for an arc, and proceeds to examine the path of the arc to determine Statistics for ZONGEN. Included in the statistics are the arcs contribution to the zones center-of-mass and the zones area. Also calculated are the angles of incidence for the arcs first node and first interior point and the same for the last node and interior point.

FILOPN

The FILOPN routine opens the terminal input and output file, the database files, and the temporary file used to store border statistics. The database files are opened as generic names (ARC, NODE, POINT, QPLOT, ZONE, BORDER), requiring the operator to rename the database files before and after zone generation.

GETLINK

The GETLINK routine follows the pointer given by one link (node-arc pair) to find the next link for a zone boundary. This process requires a search of the group of links for a given node to first locate the link containing the incoming arc from the previous node. Then, finding the proper exit link, according to the left-turn algorithm, is simply taking the next link up (next least index) in the group for the node or the last link in the group (when the incoming link was the first). Since links are zeroed out as they are used, it may be that no link is found. If so, the error is reported and the program halts.

MAIN

The major work of the MAIN routine has been described above. Operation details which were not included in the previous discussion are listed here.

The operator is allowed to select the number of code ranges which may be used to select arcs for zone generation. Up to five ranges (low and high feature codes) may be specified by the operator.

The zone mark size (which determines the size of the plus symbol for editing and plotting) is set by the operator. This size may be changed alter to increase visibility of the marks in large zones or to shrink marks to completely fit inside of small zones.

The level of (verbosity) output printed to the terminal to document the ZONGEN run may be set to one of three levels (low, medium, or high). The default zone color may be set to equal the zone numbers (and thus be unique for each), or may be set to a single value for all zones. The latter option may be preferable when all zones are set to the value of the largest group of zones. These, then would not require further coloring.

The operator is then asked to approximate the percentage of arcs in the database which are qualified by codes and the percentage of simple island zones. These values are used, along with the number of arcs and nodes to estimate the number of Zone and Border records which will be required for the zone generation process. These estimates are then used to assist the operator in setting initial sizes of the Zone and Border files for initialization.

REVARC

The interior points and endpoints (nodes) of an arc are reversed by REVARC,so that the first node becomes the last(and vice-versa) and the first interior point becomes the last (and vice-versa). The reversal occurs both in the Point and the Qplot files. The statistics for the arc do not change. The REVARC routine is called to reverse arcs for simple islands when the arc was digitized in a clockwise manner.

II. B A S I C S E Q U E N C E

II.9 ZONEDT - THE ZONE EDITING PROGRAM

Introduction

The main purpose of the ZONEDT program is to provide support for editing functions on the zone information in the cartographic database. Within this broad range are three primary functions which are supported by the ZONEDT program. The first function is to examine the zone marks of all zones to make certain that each lies within the boundary of the zone to which it belongs and to make certain the mark will be large enough to serve its function in the identification and selection of zones during edit and plot operations.

The second major operation is that of editing the colors of each zone to correspond with the primary values assigned to each zone. Assignment of these colors allows the proper production of color-separated materials for the publication of maps, as well as the retrieval and display of zones by colors for the purpose of display on graphics display terminals. Such displays support many interactive mapping operations.

The third major operation of the ZONEDT program is to link together the island zones with the zones which surround them. This process is necessary to recognize and produce the complete boundary of each zone for proper display and plotting for color maps. The linkage of islands is the process of pointing to a surrounding zone and then identifying each island (simple and complex) which lie in the boundary of that zone. The subsequent use of the surrounding zone will then access the boundaries of all island zones within the outer boundary to completely specify the correct area of color for the surrounding zone.

In addition to the three primary functions of the ZONEDT program, there are two secondary editing functions performed. The first is the deletion of simple island zones which are not needed for any reason. The second function is the creation of plot output according to the colors selected by the operator. The operator selects the output file and the ranges of zone colors which are to be converted to plot form.

The following are descriptions of the ZONEDT routines. Many of the routines listed below are equivalent to routines of the same name used by the GRFEDT program. These routines will not be described in any detail here, but may be referenced in the chapter on the GRFEDT program.

The ZONEDT Routines

ADDISL

The ADDISL routine is called when the operator has identified an island to lie within a surrounding zone, and the Island Of pointer for the island must be reset to the surrounding zone. In addition, the appropriate changes are made for the Island Within pointer of the surrounding zone and the Next Island pointers of any islands which are already linked to the surrounding island. The identification of the surrounding island adds information which is added to the Border records of the island zone, namely, the zone information for the zone on the other side of the arc.

ALL

The ALL routine refreshes the screen and redraws the map data for the special maximum-scale view of the entire area, just as the same routine for the GRFEDT program.

BNA

The BNA routine is called to ring the terminal bell, move to a new line in the Conversation Area, and to set the terminal mode to send and receive alphanumeric mode, comparable to the BNA routine in the GRFEDT program.

BNAISL

This is a special version of the BNA routine which serves the same functions, except that when the Conversation Area is full, a special call to FRESH is made to produce the desired view for the operation of the ISLAND routine.

CODMSK

The CODMSK routine performs the function of checking feature codes against the values in the code mask to determine if a zone is to be plotted, displayed or edited. The function is essentially the same as that for the CODMSK routine in the GRFEDT program, except that the codes being checked are zone colors, not feature codes of arcs.

COLOR

The editing of zone color information is performed by the COLOR routine. The COLOR command allows for a number of options

which the operator selects by the choice of the arguments in the command. The options are passed to the COLOR routine in the form of two numeric arguments from the SELCOM routine.

When the argument typed with the COLOR command is the word ALL, the first number passed to the COLOR routine is -1 to indicate that all zones in the current view are to be colored before control returns to the menu. In this case, each zone in the database is checked to determine if its zone mark lies in the current view. Those that do are available for the zone coloring process.

When the single argument given with the COLOR command is a (zone) number, this number is passed along with a zero to the COLOR routine, to indicate that only that single zone is to be colored. When two numerical arguments, separated by a hyphen, are given with the COLOR command, the two numbers are passed to the COLOR routine to indicate that a group of zones of one color is to have the color changed to a new color. And, when no arguments are given, two zeros are given to the COLOR routine to signal the selection of one or more zones by the cursor for coloring.

The last option for the COLOR command is to use the DICTIONARY option, sent to the COLOR routine as the number one. This option calls the WEBSTER routine to provide a **dictionary** which translates between the CMY (cyan-magenta-yellow) system of defining colors by percentages of the three primaries and a special coding technique that stores these percentages as a single integer value. The coded value can then be assigned as the color of zones with the selected color, so the production values of percentage colors can be easily accessed.

For the cursor selection option, the zones to be colored must be in the current view, which is not changed during execution of the command. For the single zone, old-to-new color, and all zones in the view options, the operator selects a size which defines the **minimum view** which will be had of each zone to be colored. This minimum view will be centered on the extremes of the zone, and the view is adjusted to a larger size if the zone to be colored can not fit into the minimum view.

For the single zone and old-to-new color options, the new color is set specifically by the operator in the command for the latter, and after verification for the single zone option. For the all zones in the view and the cursor selection options, there is usually a number of colors to be specified so nothing is assumed. However, in both cases, there is an option to select a **default color** to be applied to all selected zones. This saves the operator a lot of typing during execution of the command.

In the old-to-new color, cursor selection, and single zone options, the operator clearly controls the selection of the zones to be colored. However, in the all zones option, the operator may find a very large set of zones to be colored. To control this set, an option allows the selection of a **minimum zone number** at which

to begin the coloring process - no zones of a lesser number will be colored.

In the case of the cursor selection option, the operator is prompted (if Verify Mode is ON) to select the zone to be colored and the zone is selected by locating the cursor closer to the mark of the zone to color than to any other zone. Once the zone or zones to be colored are selected, the view is reset to show the zone at a maximum view with a special, bold line boundary to make the zone and its mark stand out. In the cases where the new color is not fixed, the operator is prompted for the new color. In the case where a default color is in effect and Verify Mode is ON, the operator must verify the changing of the color.

If the operator attempts to change the color of a zone to the same color as an island within it or as its surrounding zone (if it is an island), the program warns the operator and asks the operator for specific confirmation. This condition can not be known prior to making the change in part of the database, so the operator is asked to keep the change and continue or to reverse the change to the original color.

At the end of the COLOR routine, the original view is restored as it was prior to execution of the routine. If the color all the zones in the view option was selected, the number of the last zone selected and colored is reported. Thus, if the operator wishes to continue coloring in small groups by increasing zone number, the minimum zone number for the next session is known.

DELETE

The DELETE routine is used to delete the Zone and Border records of a simple island zone selected by the operator. The zone to be deleted is selected by the operator using the cursor in the current view. This zone must be qualified according to the color mask and reside in the current view at least in part. The selected island is drawn in bold for verification prior to being deleted.

DRWLBL

The DRWLBL routine displays the labels for a zone, provided the zone has labels created (a function not yet available).

FILOPN

The FILOPN routine opens the basic terminal input and output files (opened multiple times for graphics and UNLOCK functions) and the database files are opened as generic files (ARC, POINT, NODE, QPLOT, ZONE, BORDER, LABEL, CALLIGRAPHY). Thus, the database files must be renamed before and after execution of ZONEDT.

FNDZON

The FNDZON routine is called to get the number of a zone for the various editing operations. The cursor is displayed and the operator positions it near the zone mark of the zone which is to be selected. The zones in the current view are examined to locate the zone whose mark is located closest to the position selected by the operator.

FRESH

The FRESH routine in ZONEDT performs essentially the same function as the FRESH routine in the GRFEDT program - the erasing of the display screen and redrawing of all map data in the current view, perhaps reset by a windowing command. However, for ZONEDT, the FRESH command has the option of drawing the zone marks and options for adding the zone numbers at the marks or the zone colors at the marks. If available, zone labels (not currently possible) can also be displayed.

When called from the COLOR, ISLAND, or MARK routines, the FRESH routine performs special options to assist the editing operation in progress. For example, when called from the ISLAND routine, FRESH marks only the island zones. Other special options to assist editing operations include drawing marks only (without erasing), drawing and marking all islands, and drawing and marking only unassociated islands (those without Island Of pointers set).

HELP

The HELP routine is called to display the names and arguments of the commands used in the ZONEDT routine. As in GRFEDT, the HELP routine in ZONEDT lists the basic form of each command for a single command, a group of commands beginning with the same letter, or for the entire menu. When Verify Mode is OFF, only the basic form is given for each. If Verify Mode is ON, an additional line of text is printed with each command to further define the commands.

ISLAND

A most important and necessary function in zone editing is the association of island zones to their surrounding zones. **Island linkage** is supported by the ISLAND command. The basic operation is to explicitly link together a **surrounding zone** with all of its islands, to make possible the correct display of the entire zone boundary for the surrounding zone. Proper understanding of this relationship is necessary to prevent unwanted phenomenon such as disappearing island zones or "bleeding" of colors across the island zones from the surrounding zones.

The operation of the ISLAND routine is to first select the (surrounding) zone whose islands are to be linked, unless that zone has been selected in the command arguments. The FNDZON routine is called to allow the operator to select the surrounding zone via the cursor. If Verify Mode is ON, the selected zone is verified. The view is set to encompass the entire scope of the surrounding zone, and the screen is refreshed for this new view. For the purposes of the ISLAND routine, only the surrounding zone and all islands in the view are drawn. Associated islands are drawn in solid lines and unassociated islands are drawn in dotted lines. Associated are marked in solid, the others are marked with dotted marks. And the surrounding zone is drawn in bold.

Selection of islands for association with the surrounding zone are made by positioning the cursor near the zone mark of the island to be associated and pressing a cursor control character which will be used to indicate the action to take. The options are:

'A' or 'a' or '+'	=	add or associate the island
'R' or 'r' or '-'	=	remove or dissociate the island
'E' or 'e'	=	exit the ISLAND command
'F' or 'f'	=	freshen the view

To add an island, the selected island must not be associated with any zone. If it is, a message is printed for the operator. Otherwise, the color of the island is compared to that of the surrounding zone. If the same, the user is asked to further verify the addition or cancel the operation. Once accepted, the in the actual association of the island is accomplished by a call to the ADDISL routine. If all is well, the island is associated by setting its Island Of pointer, the zone numbers and colors for the Border records of the island, and appropriate changes in the Island Within pointer of the surrounding zone and the Next Island pointers of other associated islands. Once the addition is successful, the zone mark is drawn as a solid plus to show the island has been associated and is no longer available.

In the event that an island has been incorrectly associated with a surrounding zone, the operator will need to dissociate or remove the island from the surrounding zone. For this purpose, the operator may use the R, r, or - cursor control characters to begin the removal of the island. First, the island must be verified to be associated with the surrounding zone. If it is not, a message is printed (including the island's actual surrounding zone if one exists) and the operation stops. If the island is associated, the removal is performed by the SUBISL routine, and an X is drawn on the zone mark of the island to indicate that the island is no longer associated. Subsequent fresh views will show the island to be available for association with a dotted rather than solid mark.

The process of adding and subtracting associations of islands

to the surrounding zone may be repeated until all are accomplished. The display, free of all other zones, promotes the recognition of completion of the island association process. The operator may, at any time, obtain a fresh view to clarify island status by use of the F or f cursor control character to invoke the FRESH command. And, once all the islands which are within the surrounding zone have been associated with that zone, the operator may exit the ISLAND command by use of the E or e cursor control character.

LABEL

The LABEL command would create zone labels, objects like the symbols created by the CYMBAL program - but with an association to a zone, and retrievable with that zone. The LABEL command has not yet been implemented.

MAIN

The MAIN routine in the ZONEDT program is responsible for the initialization of the environment necessary for zone editing. The global variables are set up, Bell Mode is set OFF, and Verify Mode is set ON initially. The screening MASK is set to allow no zone colors except those above 32,000 as reference zones. The FILOPN routine is called to open the database files, the graphics and the random access file management systems are initialized, and the terminal input and output files are opened for graphics.

The total number of zones is checked to see if there are too many zones to handle. The zones are read from the database to set the colors, locations, Island Of pointers and the extremes. The current view is set up to show the whole map at the maximum scale, and the terminal input and output files are opened for processing communication for menu commands. The SELCOM routine is called to accept and perform commands until the END command is selected, at which time the graphics system is terminated and the program ends.

MARK

The purpose of the MARK routine is to check and correct (if necessary) the location and size of the zone marks for the zones selected by the command options of the MARK command. The first option is to mark all the zones in the current view for which the zone colors are not masked. This option is selected by entering the word ALL as the argument to the MARK command.

The second option is to mark only the single zone named in the command line, and the third option is to mark the zones in the current view which are unmasked and which are selected by pointing to the zone marks with the cursor. The third option is selected by including no argument in the MARK command.

Zone marks are set initially (by ZONGEN) for each zone which is recognized. The zone mark location set by ZONGEN is the center-of-mass calculated from the paths of the boundary arcs. The zone mark size is set as a default value selected by the operator (may be made smaller for small zones). Both may be changed in ZONEDT.

Prior to the actual mark editing, the current environment is saved so it may be reset at the end of the MARK command. If the mark all zones option is selected, the operator has the option of setting a minimum zone number (as with the COLOR command) to limit the zones which may be marked. Selection of this minimum will allow the operator to mark a number of zones in sequence in each of several sessions. This orderly process allows the operator to be certain that all zones have had their marks checked.

Also, for the all zones option, the operator is allowed to set a default, fixed size for the zone mark if it is to be changed. If the default new size is not selected, a new size must be entered explicitly for each mark which is to be changed in size. For the cursor select option, each zone is selected by cursor in a call to the FNDZON routine.

Once the zone is selected and is found to be in the view and unmasked, the view is reset to give the largest complete view of the selected zone. The zone to be marked is drawn and marked in bold lines to make its current status clear. The operator is then given options for the cursor and the cursor control character:

B = change BOTH the position and size

E = EXIT the MARK command

O = the current mark is OK or acceptable

P = change the POSITION to the cursor location

S = change the SIZE to default or entered value

If the Position option is selected, then the location of the cursor at the time the P was entered becomes the new location of the mark. Thus, the cursor should be positioned to the desired location before using the P or the B options. The S option is used to change the size of the mark. The current size is printed, and the operator is requested to enter a new size. The printed size is of great importance since the scale of the display is different for each zone viewed. New sizes should be calculated by judging the view and applying a relative factor to the old size.

When the B, P, or S options are used, a modification is made. Following the modification, the same zone is redrawn to show the effects of the modification(s). The same five options are offered the operator. Repeated modifications may be made until the desired result is obtained. At that time, the zone mark must be accepted explicitly by the operator in order for the changes to become part

of the database. Exiting without acceptance with the OK option leaves the zone mark unchanged. When all zones are done, or the operator exits with the Exit option, MARK reports the last zone marked and accepted by the operator for future sessions.

MARKIT

The MARKIT routine is called to draw a plus mark at the location of a zone mark, along with the zone number (or color) at the zone mark when requested. All marks and numbers are drawn with the buffered graphics routines of the graphics system. The mark location must be in the current view or it will not be drawn. If the mark is in the current view, then any of the points generated to draw the plus sign which lie outside the view limits are reset to like on the edge of the view.

MASKS

The function of the MASKS routine are, similarly to the MASKS routine in the GRFEDT program, to reset the lower and upper limits on the code ranges for masking which features are to be displayed and which are to be restricted from display. The range to be modified is selected by the Row number, and a low and high color are entered. For ZONEDT, the MASKS routine masks zone colors rather than the feature codes masked in GRFEDT.

MRKISL

The MRKISL routine is used by the ISLAND routine to mark all the islands linked or associated to a surrounding zone, to prepare a view of these zones which promotes easy linkage of islands via the ISLAND command. MRKISL follows the chain of the Next Island pointers for the surrounding zone, calling MARKIT to draw the marks at the zone mark locations.

NULINE

The NULINE routine serves the same function as it does in the GRFEDT program, namely, to move the dialogue in the Conversation Area to a clean, new line to accept program or operator output.

NUMBER

The NUMBER routine performs essentially the same function as it does in the GRFEDT program. In ZONEDT, the NUMBER routine posts zone numbers at the zone mark locations. Here, the options are to number a single zone which is named in the NUMBER command, to number all zones in the current view, and to number the zones as they are selected by the operator via the cursor. In all cases,

the zones to be numbered must lie in the current view and their colors must be unmasked.

PLOT

The primary function of the PLOT command and routine is to produce plot output for the selection of zone boundaries made by the operator. Selection of the zone boundaries is made by setting up to five ranges of zone colors to be plotted and naming the plot file for the output. The zone color ranges are accomplished by resetting the color Mask temporarily (it is reset after the plot is created). Following these selections, the operator selects a number of possible options defining the plot file creation.

The operator enters a **scale factor** which is applied to all the coordinates of the zone boundaries as a multiplication factor which is applied after temporary translation to move the southwest corner of the map area to the origin. After the scaling operation, the points are translated back to the original position (disregarding offsets - see below). In some cases, it is desirable to translate the map to fit other plots. This is accomplished by selecting the **x and y offsets** for translation. If no translation is desired, these values are entered as zero.

The plot dimensions are checked against the limits of the plotting device and checked to see if the minimum coordinates of the plot might be negative. Any violations of the rules here may prevent the production of the plot, or may require the operator to explicitly force the plot production. This option is reserved to maps where generation of the maxima values is based on the map area boundaries, not the data points themselves.

One option to the PLOT command is to mark the zones, selected by adding the word MARK to the command. The option is offered to the operator if it has not already been selected. The operator may elect marks, no marks, or may choose to exit the command. If marks are selected, the operator may set a **minimum mark size**, which will override the mark size in the Zone record. This option is used to make certain that all marks are visible in a plot. In addition, the operator may select an additional scaling factor to be applied only to the size of the zone marks. This factor has the effect of making marks more or less visible.

If zone labels were available, they could be plotted for the selected zones (labels are not yet implemented. The operator may also select the addition of **registration marks** at the corners of a rectangle defined by the operator. If chosen, the size of the registration marks may be set by the operator. The operator may also choose to add an optional **origin mark** set to indicate the plot origin location.

Once the requested information has been plotted, the plot file is closed (after X-ing the plus sign used for the origin mark, if

selected), the mask is reset to original values, the number of arcs plotted is reported, and control returns to the SELCOM routine.

PLTARC

The PLTARC routine is used to plot a single zone boundary arc in the forward direction. The scale factor and the translation (x and y offset) values are applied to the first node, the interior points, and the last node for the arc. All points are converted to plot form by a call to the graphics routines GMPMOV and GMPDRW.

PLTLBL

The PLTLBL routine would draw a selected label with or without a hook (or leader) to the zone mark by converting the coordinates in the associated Calligraphy records to plot form by calls to the GMPMOV and GMPDRW routines. However, zone labels have not yet been implemented in ZONEDT.

PLTREV

The PLTREV routine is used to create a plot of a single zone boundary arc, but to plot it in the reverse order to that in which it is stored in the database to improve the quality of the plot or scribe by plotting a continuous boundary when possible. Reversal is achieved by plotting the last node, followed by the interior points in reverse order, and then the first node last. Plot conversion is performed by calls to GMPMOV and GMPDRW.

QPLOT

The QPLOT routine performs the same function in the ZONEDT program as it does in the GRFEDT program. For a specified arc, the QPLOT routine accesses the Qplot file to get the screen coordinates for the interior points of the arc and displays these points for the whole map, maximum scale view.

SELCOM

The SELCOM routine controls the flow of control and the menu selection of commands by the operator just as it does in the GRFEDT program, but with a somewhat different set of commands (see HELP). Basically, SELCOM gets and analyzes enough characters to recognize the command name and analyzes the arguments to find numerical or alphanumerical values which are then passed to the ZONEDT routine (usually) of the same name as the command. When no command can be recognized, or the arguments are in error, SELCOM prints an error message and tries again. Only when the operator selects the END command does control return to the calling routine MAIN.

SHWPT

The SHWPT routine performs the same function as it does in the GRFEDT program. A single point location is to be verified by the operator, after SHWPT marks its location in a flashing (temporary) cursor with an hourglass symbol. The point must lie in the current view. The operator may confirm it to be correct, reject it and try again, or reject it and exit the calling command.

SHWVW

The SHWVW command is the same as that used in the GRFEDT program to obtain operator verification of a selected new view. After flashing the square representing the new view, the operator is may accept or reject the proposed new view.

SHWZON

The SHWZON routine is called by various editing routines to draw the boundary of a selected zone (i.e. the surrounding zone for ISLAND) in a special, bold line representation which makes the zone easy to recognize. The boundary arcs are drawn in the order they exist in the Border file since the output is only for display on the screen. If selected, the zone mark will also be drawn.

SUBISL

When the ISLAND routine is used to dissociate a previously associated island zone from a surrounding zone, the SUBISL routine is called to remove the connections in the Island pointers of the island and surrounding zone records and update the Border arc records for the island zone. What is done in the ADDISL routine is undone by the SUBISL routine except that it is unnecessary to check the resulting color relationships.

TOKEN

The TOKEN routine examines the contents of the command buffer to determine the type of entity found there. The tokens identified by the TOKEN routine are passed to the SELCOM routine which uses the information to determine which routines to call to perform the desired command. The TOKEN routine is essentially identical to the same routine in the GRFEDT program.

UNLOCK

Just as in the GRFEDT program, the UNLOCK routine is called by various editing routines to restrict the use of some powerful

editing functions to advanced operators. The password procedure used here is the same in technique and operation as that used in the GRFEDT program.

VIEWRS

The ZONEDT program uses the same ability to save and restore one view (via the SAVE and RESTORE commands) as does the GRFEDT program. The VIEWRS routine provides that capability by saving the view extremes for a saved view and calling the FRESH routine to restore a view, after restoring the saved extremes. The VIEWRS routine is essentially identical to that in the GRFEDT program.

VPLOT

The VPLOT routine is essentially identical to the same routine in the GRFEDT routine. Its function is to plot a single arc on the graphics display terminal when the current view is not the special, all map view and clipping may be required.

WEBSTER

The conversion of colors for zones between the percentages of primary color form and the special ZONEDT coding is performed by the WEBSTER (as in dictionary) routine. The percentages may be for a three (cyan-magenta-yellow or CMY) color or a four (CMY + black) color system. The operator selects 3 or 4 colors, and then selects the direction to convert, percentages to code or vice-versa. Then, either the (3 or 4) percentages are given and the coded value is printed or the coded value is given and percentages are given.

The conversion process may be repeated for different values by selecting the direction of the conversion each time and entering the required values. When the direction is selected as zero, the number of colors may be reset. To exit the routine, the number of colors is selected to be zero.

WINDOW

The purpose of the WINDOW command is to provide methods for selecting a different viewing area. The WINDOW routine provides the methods of selecting this view, according to the command arguments, and in essentially the identical manner as the WINDOW routine in the GRFEDT program. Details of the operation and function for this routine may be found above in the GRFEDT information.

The options for windowing in ZONEDT are (1) selecting the two opposite corners of the desired view with the cursor, (2) entering X as the cursor control character to enter the new view extremes at the keyboard, (3) entering a direction (N,S,E,W,NW,NE,SE,SW) for

the new view (with or without overlap) through the cursor control characters, and (4) typing Z for cursor control to select a zone by number to define the new extremes. Once selected and verified (if Verify Mode is ON), the FRESH routine is called to erase the screen and draw unmasked zones for the new view.

ZOOM

The other form of new view selection is provided by the ZOOM command and routine. This routine is described in detail elsewhere since it is essentially identical to the ZOOM routine used in the GRFEDT program. As there, the options for selecting the new view include selecting the new center of the view by (1) using the location of the cursor, (2) entering the coordinates after using X as the cursor control character, (3) using the location of the zone mark of a selected zone after using Z as the cursor control character, or (4) using the current view center after using S as the cursor control character. Having selected the new center, the operator then enters the size of the new view in user coordinates. Once selected and verified (if necessary), the new view is drawn by the FRESH routine.

II. B A S I C S E Q U E N C E

II.10 CYMBAL - SYMBOLOGY EDITING PROGRAM

Introduction

The CYMBAL program provides a means to interactively generate and manipulate labeling and annotation features to match features in a GIMMAP cartographic database. Operating at a vector graphics terminal, a user may create annotations using any of the symbols from twenty-three Hershey Alphabet fonts, some of the alpha-numeric characters and some of special map symbols. These special, point symbols include basic, geometric symbols (triangles, squares, and circles), city symbols, special application symbols, scale bars, frames and boxes, and north arrows.

The CYMBAL program operates by opening the map database chosen by the operator and then displaying the map features as they would be displayed by the GRFEDT program for interactive editing. Map features are masked by ranges of feature codes and the operator may select any desired view of sub-areas of the map area to zoom in on an area for accurate placement of symbology. Thus, the operator is allowed to place, view, and edit symbology exactly in the place where it will appear on map plots. City names can be placed next to the city symbol, section numbers may appear in the center of the section, and stream names can follow the course of the stream (this must be placed manually; it is not automated).

In the selection and placement of map symbols with the CYMBAL program, the operator may select and set many options which control the form and location of the symbology. Foremost among these is the selection of the Hershey alphabet set of symbols from which the units of a map symbol are drawn. Most Hershey alphabets contain both upper and lower case alphabets, the digits, and most of the other keyboard characters. Selection of a particular character is made simply by typing the character. Differences in the Hershey sets lies in the font (e.g. Old English, Italic, Roman Gothic...) and in the thickness of the characters, generated by the number of pen strokes used (simplex, duplex, triplex: one, two, three). A complete chart of all Hershey alphabets may be plotted by running the HERSHPLOT program, and plotting the output file.

There are six other factors that the operator may select which determine the final form of the symbol being created. First, the content of the symbol is typed by the operator at the keyboard, and may be changed any time later. In addition, the operator must set or select five other facets of the symbol. First, the reference point must be selected by cursor to act as the location to which the symbol is justified as well as the location used subsequently for selection of the symbol by the operator via the cursor (for editing).

The operator must set the character height, which also affects the length of the overall symbol since the ratio between the height and width of characters is fixed. The symbol is justified to the reference point as left (lower-left corner at the reference point), right or centered. Each symbol is assigned a feature code so that symbols may be masked along with map features and may be selected by codes for plotting. Finally, the symbol is given an angle of orientation through which it is rotated, usually to match a feature on the map. All these aspects of the symbol, except the reference point, may be set in a **template** so the operator need not specify the same values redundantly.

There are essentially four conceptual levels of code that comprise this program. Level 1 is simply the MAIN program which receives control at program start-up, performs various initializing functions, and lastly calls SELCOM to begin interaction with the user. SELCOM represents level 2 where user commands are accepted, and control is passed to the appropriate level 3 subroutines. There is a unique level 3 subroutine to coordinate and execute each distinct user command. Code at all levels utilizes the support functions provided by the level 4 routines. Below, each of these routines is described in general.

The CYMBAL Routines

MAIN

This is the routine receiving control at program start-up. MAIN opens the various files used by the program. It then reads the ARC and SYMBOL files to build internal tables containing feature codes and spatial extent of each arc/symbol. The initial viewing parameters are set to display the entire map area, and the initial feature code masks for ARCs is set to State and County boundaries. Feature code masks for SYMBOLs are initially set to null. MAIN then passes control to SELCOM, which returns only after the user has signalled to terminate the session.

SELCOM

The level two SELCOM routine (for Selection of Commands) interacts with the user at the program command-line level to accept user commands, and call the appropriate subroutine to execute each unique command. SELCOM returns back to MAIN only after the user has entered and confirmed an "END" command.

User-Command Routines

These subroutines correspond one-to-one with the user commands available. Each subroutine gathers any additional information needed from the user, and then executes that command. A list of these subroutines/commands follows:

ALL

Sets the viewing parameters to display the entire map extent as in the GRFEDT program. The Mask now consists of separate code ranges for both the map features and the symbology. Only those symbols which are unmasked will be displayed. The ALL display is the whole map area, maximum scale display.

BAR

The BAR command allows the user to construct a map scale bar, of specified length and number of tick marks, in either miles or kilometers. The scale bar may be placed anywhere in the map area, and when plotted, it will display the true map scale according to the map projection parameters. The scale bar must be labeled by the operator creating symbols for the distances and scale units. Addition of the explicit map scale (1:scalefactor) by the creation of a symbol is useful to the map viewer. However, the map scale bar alone will maintain its accuracy even when the plot is scaled, while the explicit form will not.

BOX

The BOX routine generates a single or double box at a location selected by the operator. The cursor is used to select 2 opposite corners of the rectangle, defining both the size and location of the box. For double boxes, the separation distance (distance to the inner box from the fixed outer box) is entered by the operator. The box may be used for emphasizing other symbols, for creating an area for legend information, as boxes for colors in the legend, for special areas of text, or as the map frame.

CODE

The CODE routine allows the operator to change the feature code of a specified symbol which is selected by pointing to the symbol via the cursor.

COPY

The COPY routine duplicates an existing SYMBOL at a specified location. Using the COPY routine, the operator may point to an

existing symbol such as the highway symbol, a geologic symbol for a formation or any other point symbol or name which is repeated in the map area. The selected symbol may then be duplicated simply by pointing to a location at which it is placed. The specification of the other symbol aspects (see above) is not required, and the creation of the duplicated symbol is thus much faster than creating a new symbol for the copy.

CREATE

The CREATE routine creates a new map symbol. The operator specifies content, orientation, size, font (Hershey Alphabet set), feature code, and location (some of this information may be preset via the TEMPLATE command, and is not requested). Once created, the new symbol is displayed in the current view of the map, along with other, existing symbols. The operator may accept the new symbol or reject it and try again.

DELETE

The DELETE command is called to delete a specified symbol which is no longer wanted in the database. The operator selects the symbol to be deleted by use of the cursor, and the symbol is verified by the operator after a flashing box around the symbol shows that the correct symbol is identified. Once the symbol is deleted, it can not be returned except by re-creating in with the CREATE command.

DISPLAY

The DISPLAY routine displays the inventory of map features and symbols in groups according to feature codes. As with the DISPLAY command in the GRFEDT program, the CYMBAL program displays all map features and symbols of a single feature code, one code at a time. The display begins with the lowest possible code and works to the highest. After each group is drawn on the graphics terminal, the program pauses to allow the operator to examine the group or to get a hard-copy of the screen.

FRESH

The FRESH routine redraws the current view with all unmasked map features and all unmasked (according to the separate symbol mask feature code ranges) map symbols. Unlike the FRESH routine in the GRFEDT program, no feature numbering may be shown.

GRID

The GRID routine allows the operator to set up a grid of lines of specified spacing, to be superimposed on the map display for all subsequent map views, until it is removed by later use of the GRID command. The grid which is created is useful for aligning and guiding the placement of new symbols according to the locations of existing symbols. Using GRID, the operator may place a group of like symbols at (approximately) an equal distance from map features or other symbols. For example, township or range numbers may all be placed the same distance from the state boundary.

HELP

The HELP routine displays a list of available commands (as in the list of commands being defined here), plus some information for the operator about interacting with the program. For example, the ability to re-select the previously-selected command simply by typing the newline character.

HERSHY

THe HERSHY routine displays all the characters comprising a specified Hershey font, along with their keyboard equivalent. The Hershey Alphabet is selected from one of twenty-three sets which are available to the CYMBAL program. For most alphabets, all the characters are selected by the operator typing the character on the keyboard. For the sets with special graphic symbols which do not appear on the keyboard, the HERSHY command provides the keyboard equivalent - the key to type when the special character is to be entered in a symbol.

JUSTIFY

The JUSTIFY routine allows the operator to change the left, center, or right justification of a specified symbol. The symbol to be changed is selected via the cursor, and is verified according with a flashing box. The original justification is with respect to the location selected by the operator as the reference point. The new justification is attained by shifting the symbol to cause the appropriate part of the symbol (bottom-left, bottom-center, or bottom-right) to line up with the reference point.

LINES

The LINES routine allows the user to draw free-form, connected line segments upon the map. This is useful for drawing "hooks" or "leaders" from an identifying label to a region too small to contain it, for creating special new (straight-line) symbols, and for numerous other creative function in symbology editing.

MASK

The MASK routine allows the operator to change the feature code masks which are used to determine which map features and which map symbols are to be displayed and which are not. The operator may easily reset the codes to allow all features and all symbols to be displayed. As in the GRFEDT program, a special set of map features (map edge,...) are automatically displayed as reference or background features by some of the mask ranges.

PLOT

The PLOT routine generates a plot file of the symbols in the map database. The operator selects the output, plot file and the sets of symbols which are to be converted to plot format and put in the plot file. The symbols are selected by ranges of feature codes entered by the operator. In addition, the usual plot options are available including scaling and plot translation.

QUERY

The QUERY routine displays the descriptive information about a specified symbol. The operator selects the symbol by the cursor, it is verified by the flashing box, and the information is provided in the conversation area. The information provided includes all the aspects of the symbol which were set by the operator when the symbol was defined, including the content, height, justification, feature code, reference point location, and angle of orientation.

ROTATE

The ROTATE routine allows the operator the ability to change the angle of orientation for a specified symbol. The operator selects the symbol via the cursor and verifies the symbol by the flashing rectangle method. The angle of rotation may be set by entering the angle in degrees or by selecting an angle relative to the reference point with the cursor.

RESTORE

The RESTORE routine is used to restore a previously SAVEd view as the new current view. Once the SAVE command has been used to mark a view for future restoration, the RESTORE command may be used at any time to return to that saved view. Only one such view is saved. When the SAVE routine is used more than once, only the most recently SAVEd view is that which is restored.

SAVE

The SAVE routine saves the current viewing parameters. The operator may then perform any zooming and panning and symbology editing in the current view or elsewhere, and later 'RESTORE' the saved view. This allows the operator to organize the symbology placement and editing by areas which are saved so that particular operations may be performed within the area, and then the operator can return to the view to work in other, controlled sub-areas.

SCALE

The SCALE routine is used to change the size of a selected symbol in both its character height and its length. The symbol is selected by the cursor and verified. The character height is then scaled by the operator and the pen strokes defining the symbol are adjusted accordingly. Justification (left, right, or center) to the reference point is maintained.

TEMPLAT

The TEMPLAT routine allows the operator to set up a "template" defining one or more of the parameters of all subsequent symbols (until the template is changed or turned off). Any combination of symbol size, font, justification, orientation, and feature code may be preset in this manner.

TRANSL

The TRANSL routine allows the operator to move the location of the reference point of a selected symbol to a new location in the current view. The symbol to be moved is selected by the cursor and is verified. The new position of the reference point is then selected by the operator via the cursor. All points defining the pen strokes for the symbol are translated according to this move and the symbol remains the same size and justification.

WINDOW

The WINDOW routine allows operator to change the current view by selecting a rectangular area to define the new view. As with the GRFEDT program, the WINDOW command in CYMBAL offers a variety of methods to select the rectangle. One is to enter coordinates for opposite corners, one is to move in a direction specified by points of the compass (N, S, W, E, NW, NE, SW, SE) as the control characters typed with the cursor, and one is to select opposite corners via the cursor.

ZOOM

The ZOOM command allows the operator to change the current view to one which is centered about a point selected by the operator. The operator uses the cursor to select the center of the new view and specifies the size of the new view in user coordinates (usually inches). The current view size, shown as x and y extremes in the lower-right corner of the display, may be used as reference for setting the new size.

Utility Functions

The Utility subroutines/functions provide a set of utility functions for use by all levels of code. Their effect is to centralize commonly used code, and simplify higher level code.

DRAWING ROUTINES:

DRWSYM Draws a given symbol. Reads through the associated Symbol and Calligraphy records to generate the necessary pen-strokes.

DRWARC Draws a given Arc. Reads the associated Arc and Point records.

MOVE Generates a "move" pen-stoke. Input is in world coordinates. Output is transformed to display coordinates and clipped to the current view.

DRAW Companion routine for MOVE above.

COMMAND-LINE INTERFACE ROUTINES:

GETLIN Reads a line of input from the keyboard, storing it in a buffer for access by companion functions.

GETINT Returns a specified number of integers given in the command-line buffer. Prompts the user for more if an insufficient number were initially given.

GETREL Returns a specified number of real-values from the command-line buffer.

GETYN Returns user's response to a "YES/NO" question. A default response is provided, which is assumed if the user merely responds by typing RETURN.

CALLIGRAPHY RECORD I/O SUPPORT FUNCTIONS:

The "_OLD_" support functions operate on existing symbol and Calligraphy records. They are used to retrieve and/or modify symbols.

OPEN_OLD_CALLIG Reads the first calligraphy record of a given symbol, establishing this calligraphy chain as being "current"; all subsequent operations will apply to this chain until "closed".

GET_OLD_CALLIG Returns the "next" x/y pair from the current calligraphy-record chain.

GET_OLD_CONTENTS Returns the ASCII contents of the current calligraphy-record chain.

PUT_OLD_CALLIG Replaces the current x/y pair of the current calligraphy-record chain.

CLOSE_OLD_CALLIG Updates the last record of the current calligraphy-record chain.

The "_NEW_" support functions generate new instances of calligraphy records:

OPEN_NEW_CALLIG Initializes a new calligraphy-record chain.

PUT_NEW_CALLIG Places an x/y pair into the calligraphy records. Handles writing of records as they become full, and allocation of new records.

CLOSE_NEW_CALLIG Writes the last calligraphy record of the chain, ensuring the cyclical nature of the "next" pointers.

MISCELLANEOUS SUPPORT FUNCTIONS:

CREAT2 Companion function to the CREATE command above. This command gathers up necessary information from the user and the current template, for the creation of a new symbol.

FNDSYM Returns the symbol closest to the location of a user's graphical pick action.

INTSYM Called by MAIN to open the symbol/Calligraphy files. Checks for the existence of these files, and initializes them if new files are to be created.

NULINE Called to advance the command-line a specified number of lines down the screen. When the left-hand margin of the screen is full, the entire view is re-displayed.

SHWMBR Displays the "minimum bounding rectangle" of a given symbol on the screen in write-through mode.

SHWWND Displays the new view proposed by a user's actions, in write-through mode.

SYMTBL Enters a new symbol's feature-code and extent into the internal symbol table.

LIBRARIES:

RAFMAN77.LB The Random-Access File Management library for the control and access of the map database files.

GRFMAN77.LB The Graphics Management library to provide all low-level graphics functions on the display terminal.

II. B A S I C S E Q U E N C E

II.11 MAPGEN - THE MAP GENERATION PROGRAM

Introduction

Of the many programs in the Basic Sequence of the GIMMAP system, there are those which stand out in their function and structure as superior to the others. The MAPGEN program, like the GRFEDT program which preceded it, is one of the most significant programs in the GIMMAP system. The whole system may be viewed in very basic terms as having input, database construction, editing and addition of data, and output for map production functions. The MAPGEN program is the cornerstone of the output function of the system, providing the greatest single source for map production capabilities (other sources include the PLTGEN, CALPLOT, CYMBAL, ZMARKS and X2C programs) as well as for the creation of DDF output for various purposes (other sources are the DIGGEN and ARCHIVE programs).

The MAPGEN program has the basic function of producing plot files for map production. Actually, the plot file form is a basic option, with the alternative of producing output in the form of GIMMAP DDF (digitized data format) files instead of plot file form. The use of plot files for map production is obvious, but the use of producing DDF files is not. There are two basic uses for the DDF form of output from MAPGEN. The first is to provide clean, edited data for an area of one or more map databases for the purpose of **archiving** the data or for **transferring** the data to a second site.

Another use for the DDF form of output data from MAPGEN is to create files which may be used to create a new, or append features to, a database covering the whole area of one or more smaller map databases. One value of creating this new, **intermediate** database is to provide the operator with a complete view of the final map area for the purpose of adding map symbology. With the complete map area, this process is more likely to succeed with fewer errors made in collisions between symbols and features, and will result in better consistency of style and content of symbols across the whole map area.

Another very important function of the MAPGEN program is to perform the task of taking data from numerous, (usually) contiguous map databases and joining or **quilting** the data into a single map product. This process is performed by using the routines in the PROMAN (projection management) library to convert the coordinates from the input databases from projected x-y into geographic (lat-long) coordinates in a process called **deprojection.** The geographic coordinates are then universal, in the world-wide system, and are thus common for all areas being quilted. A common, output set of

projection parameters, selected by the operator, is then applied to all data from all the input map databases which are being joined by the MAPGEN program. The geographic coordinates are converted by the **projection** process into the common output projection and the resulting features will plot together as well as the corresponding earth features, if digitization and editing was done properly (and if the original document maps or digital sources were accurate).

Another (optional) feature of the MAPGEN program may be used in conjunction with the quilting capability to produce very useful results for either the plot or DDF option. The restriction of data to a specified rectangle, called **clipping**, is an option which the operator may choose to solve many different problems in mapping. Clipping causes all feature information outside the selected rectangle to be ignored, while all data in the rectangle is used. The operator may select the clipping window in projected x-y or in **geographic** coordinates (latitude and longitude). The projected clipping rectangle may be at the output projection limits or at any specified location. The geographic clipping rectangle may be set at the map projection area limits or other specified location.

There are two basic uses for the clipping function in the GIMMAP system. One is to use MAPGEN to clip map data from a large area (in a single map database) to create a map or DDF file which represents a small, sub-area of the larger area. Thus, the large area may be broken into numerous, clean clipped areas for viewing or for further processing. One example of this is to produce up to twelve rectangular base maps covering the state may be created from a single state database containing the basic features. These maps may be used so new features may be hand-drawn with the proper reference for accuracy.

The second basic use for the clipping capability in the MAPGEN program is to have data from a number of adjoining databases to be **reprojected** to a common output projection, and then clip a subset of this data only, in a rectangular area. Thus, a number of small pieces are essentially joined together, but are restricted to the area of interest by the clipping function. An example of this use of clipping would be the quilting of all counties to form a new map restricted to a geographic rectangle. Only the features within the rectangle are kept, others are discarded. Features crossing the clipping boundary are cut off at the boundary. The MAPGEN boundary tags each point as IN or OUT, and has coded a scheme for finding the intersection of the clipping boundary with any segment in which one point is out and the other point is in the clipping area.

Still other uses of the clipping ability may be discovered. In conjunction with a specialized coding of the features in the database, to incorporate some spatial information (such as using the county code in the feature codes of the map features), clipping may act to select subsets of data by more general boundaries (such as the non-rectangular county boundaries). Addition of generalized clipping (using any arbitrary boundary) would greatly increase the usefulness of these options in MAPGEN.

The clipping capability is supported by the MAPGEN program in either the projected x-y coordinates or in geographic (latitude-longitude) coordinates. In either case, the operator selects the minimum and maximum coordinates in the chosen system. The clipping rectangle is stored, and the MAPGEN program must then test all data points against the clipping rectangle. In the reprojecting of all the points from the input databases, each point is deprojected from x-y coordinates to geographic coordinates. If the clipping option is for clipping in geographic coordinates, then it is performed on each point at this stage. If clipping is in projected coordinates, then it is performed after each point is projected from geographic coordinates into the common, output projection.

In order for the MAPGEN program to perform the quilting function, it must be able to accept the data from a number of input map databases. For this end, MAPGEN allows the operator to specify the names of multiple input databases by placing them in a file which MAPGEN reads for this purpose. This file is the Names file, and is opened and read by MAPGEN in initial setup. The names in the Names file are expected to be the standard form for Arc file names in the GIMMAP system - they are also used to derive the Point file names based on this standard form. When a single database is to be used as the input source for MAPGEN, the name of the Arc file for this database may be given directly to MAPGEN by preceding the Arc file name with an asterisk (*). If desired by the operator, the MAPGEN program may use the contents of the Names file (if they are in the standard form for USGS 7.5' quad maps) to determine the extent of the map area, and the parameters for the area will not need to be entered in this case.

Since the MAPGEN program performs a reprojection of data to a (possibly) new scale, it is possible that the points in the input databases may be more dense than is necessary for the output map projection. The result of this is that points in plots may be too dense and cause the plot to take much longer than necessary. The same problem could cause undesired marks on the plot because of the slowness of pen movement. For DDF files, points which are too close together may cause problems in the construction of the new database (excessive nodes) or in the transfer of data (excessive size of files and length of file operations).

Of course, it is also possible that reprojection may cause points to become less dense than in the input databases, thus possibly causing different problems. The change to a smaller scale, such as in this case, is not recommended in most cases. Map data, for the most, part should be used at a scale which is larger than that of the input data.

To prevent these problems, the operator may set the size of a **weeding** threshold which is to be applied to all points as they are prepared for the output file, but only after application of the output projection. Each point is compared to the location of its predecessor and its successor (except endpoints), and those which

lie within the weeding threshold are discarded. At the end of the program, a count is given to report the number of points written to each output file and the total number of points which have been discarded according to the weeding threshold. The MAPGEN program suggests a value (one for plots and one for DDFs) for the weeding threshold, and the operator may use the default or reset the value to a desired one, usually depending on the scale change. Weeding can reduce the number of points output by a large amount.

Another function of the MAPGEN program for the plot option is to convert point locations into point symbols and to change simple, straight line representations of line arcs into annotated lines representing the real world features, as assigned by the operator. Each point in the map database has an associated feature code which the operator may assign to a particular point symbol with a particular size and other required aspects. Among the symbols available, the operator may select from the plus, the letter X, the star (*), the square, the triangle, the circle, the fiducial (a square with X and +), the hexagon, the solid circle, the solid square, the solid triangle, and a collection of oil and gas well symbols.

All the point symbols are centered at the point location which is specified by the feature in the map database. The size of the point symbol is selected by the operator in projected coordinates (usually inches), or may be specified as a function of the feature code of the point. To find the symbol's size, a base value which is specified by the operator, is subtracted from the point's feature code. The result is then multiplied by a scale factor also supplied by the operator. (Clearly, this technique is best taken into account in the original coding of the points). The effect of this method is to allow sizes of like point features to vary with differences in the value of their feature codes. For example, if the epicenters of earthquakes are coded to include the magnitude of the earthquakes, then earthquake symbols may be generated which reflect these magnitudes.

Similarly to point features, the linear features of the map databases may be annotated to produce line representations which clearly distinguish the lines of the map by more than just the color of the line. In some cases, the type of line selected for a feature may actually bear some special resemblance to the actual feature, such as using a hashed line for railroad features. The creation of these special line types occurs after all the points have been reprojected, the weeding threshold is applied to remove excess points, and clipping is performed at the selected boundary. Only then, with the actual set of points which is to be plotted, (but on the fly), are line types created.

The types of lines that may be selected in MAPGEN include the solid line (default for unspecified codes), the dashed line (operator specifies dash and gap lengths), the hashed line (the operator specifies hash width and separation), the double line (separation of the two lines parallel to the original), the long-

short and the long-short-short dashed lines (dash and gap lengths), and the double dashed line (separation of the two lines, and dash and gap lengths). A bold line may be approximated by choosing the double line with a very small separation distance. In addition, all the line types may be affected by the choice of pen types and widths (if possible) selected at plot time.

The quilting function is governed principally by the selection of the output projection parameters by the operator. The resulting map or DDF file is more grossly affected by these choices than by any other single factor short of the collection of input databases. These choices completely define the transformation process used to convert input data from multiple databases of differing projection types, scales, and areas; into the common output projection based on a common model of earth, projection object (e.g. cone) and all other relevant factors.

To completely define the output projection, (a process more fully defined in the section on the GIMMAP.PROJCT projection library), the operator must specify the projection type (modified polyconic, Lambert conformal conic,...), the scale factor (as in the scale is 1:scalefactor), the extremes of the map projection area (south, north, west, east, in degrees), and other parameters specific to the projection (e.g. the standard parallels for the Lambert conformal conic projection). In MAPGEN, the area extremes may be selected to be those already selected as the clipping window extremes when the clipping option is exercised.

The output data from the MAPGEN program is either in DDF or in plot form. This data is sent to multiple (as many as fifteen) output files which are named by the operator. As the operator defines each of the output files, a list of one or more (up to fifteen) ranges (low and high code) of features codes may be attached to each file. The feature code ranges define the set of map features which are to be placed into the attached file. Different map features are placed in different files for numerous reasons.

Primarily, features are segregated to allow the file to be plotted separately, usually to use a different color of pen for each kind of feature. Features are also separated to allow them to be combined in different ways to show different features on different maps of a given area. Linear features must be separated if they are to be converted for color plots by use of the X2C program. Each line type which is to have a unique color or line width must reside in a separate file.

For DDF output, this separation allows completely different databases to be created to process different feature types, which may improve the speed of the operation. Also, separated features may be archived to provide for faster restoration or transfer of specific feature types. The MAPGEN program is usually used with the DDF option to produce output files which are used to create an intermediate database for symbology placement. For this use, there

is no need to create more than a single DDF file containing all the selected features for the database, but there may be a need to create two or more of these intermediate databases.

For the DDF output option, the operator may select a number of options for the form of the data in the DDF file. First, the file may actually be a legitimate DDF file, immediately usable in the GIMMAP system. Alternatively, the coordinates may be put into the output files at the point where they have been deprojected and are in geographic coordinates. In this form, which is ideal for transferring data, the longitudes may be shown as a positive or a negative, as chosen by the operator. Also, in the geographic form, the file is not generated in the DDF form, but retains only the flags which signal discontinuities and feature code headers, both of which are optional.

The DDF file output option also provides an option to include the z-values associated with the isolated point arcs in the input databases, if any are found there. This option will produce the same syntax in the DDF files as is expected by the SYNEDT program when the file is edited for syntax. The feature code header may include the number of z-values for the points, and exactly this many z-values is to be included in each point record included in the DDF file.

In addition to the functions and options listed above, the MAPGEN program also offers a number of other options which the operator may select to further enhance the appearance and function of the map when the plot option is chosen. Among these is the addition of an **origin mark** which shows the location of the origin of the plot and is marked before and after the plot to check for any loss of origin in the vector plotter. This option is usually not selected for the color, raster plotter.

Other options of the MAPGEN program include the possibility of drawing a single or double frame around the map area. The map frame is a rectangle in x-y which can be drawn at the extremes of the projected area or at another, operator-selected location. The map frame may be a double frame, with the outer frame located at the operator's selection and the inner frame located at a distance specified by the operator.

The operator must specify **translation** values, or **offsets** in x and y, which are applied to all points at the end of the process of reprojection. These offsets may be used to translate the map to fit it with another plot such as a contour plot, produced by a program which does not honor the locations which are given to it. If the translation is not desired, the offsets can both be set to zero.

The operator may select to add the GIMMAP label to the plot. If added, the GIMMAP name is added to the lower-left part of the plot, but outside the map area. No translation of the map data is made to accommodate for this label. If the GIMMAP label is chosen,

additional options are available. The map projection and scale will be plotted along with the GIMMAP name as Modified Polyconic or Lambert Conformal Conic Projection and Scale = 1:scalefactor. In addition, the date and time of the production of the MAPGEN plot files may be added to the label beyond the scale factor. Since the date and time may change between the production of different plot files, these values should be left off all but one plot file.

Finally, the operator may select to create a set of **fiducials** or **registration marks** symbols made to mark the location of the four corners of the map projection area for the purpose of accurately putting together or registering two or more overlays for a single map. Another, equally important function of such registration marks is to provide the locations for the standard control points on the plot for the possibility of adding features to the plot and digitizing them according to the standard GIMMAP technique. The registration marks provided by MAPGEN may be selected as an option to or in addition to the map frame. They may be chosen to be automatically located at the corners of the map projection area or may be explicitly entered by the operator after viewing the extremes of the map plot.

The registration points will satisfy three functions. First, they will locate the corners of the map projection area (or other selected locations) for the purpose of registration. Second, they will have an outer triangle drawn prior to the plot and an inner triangle drawn at the end of the plot to show any loss of origin by the plotter. Finally, they will not overlap the map area in any way beyond the single points which are the selected corners. This way they can not obscure any map features, unless the features lie outside the map projection area. If desired, the four registration points can be automatically labeled with the longitude and latitude (in decimal degrees) for each corner, with symbols generated to be completely above or below the map area.

The MAPGEN Routines

The MAPGEN program consists of twenty-eight routines to perform the various functions in concert. Many of these routines have names which begin with the letters MLT, signifying that the routine is performing a basic function multiple times, usually to multiple output files. Each of the MAPGEN routines is described below in general terms.

DEPROJ

The DEPROJ routine is called to deproject a point which is in single precision form into geographic coordinates. The DEPROJ routine in turn uses the ROOK2E routine of the PROMAN projection library software to perform the actual deprojection. The DEPROJ

routine also calls the GMVCOD routine of the graphics management library (GRFMAN) to test whether the point is inside or outside of the current view - in this context, the view corresponds with the (optional) clipping window. A code is returned to indicate the result of this test, and may be used by the calling routine to perform the clipping if that option is selected.

FNDPRO

When the operator provides the name of the (Arc file) Names file, the file may be (optionally) read to determine the extent of the output map area as the minimum bounding rectangle for the complete set of input databases. This option is only available when the standard Arc file names for USGS 7.5' quadrangle maps are used in the Names file, and no other names are present. In this case, each name includes the (GIMMAP) row and column number for the grid of these maps in the state of Kansas (as used at the KGS).

Using these row and column numbers, the MAPGEN program can determine the extent of each map database in the Names file, and can thus calculate the bounding rectangle for the collection of all the map databases in the Names file list. If any non-standard Arc file name is found in the list, the routine abandons the attempt to automate recognition of the area, and asks the operator to enter the boundaries for the projection.

IMAGE

The purpose of the IMAGE routine is to recognize when an arc which is to be reprojected into the output projection does not lie at all inside the clipping window, if the clipping option has been selected. When it can be recognized that an entire arc will lie completely outside the clipping window, then the arc may be ignored and no reprojection of its points is necessary. This saves the MAPGEN program many calculations and reduces the time necessary to perform its functions. When arcs are not outside the clipping window, they are completely or partially inside. When they are completely inside, clipping need not be performed. Only when the arc is partially outside does clipping have to be done.

The IMAGE routine uses the extent or extremes of the bounding rectangle for an arc (or anything bounded by a rectangle) to make its calculations. By deprojecting and possibly reprojecting in the output projection, IMAGE creates points in the output space (either geographic or projected x-y) which can be compared to the extremes there to determine the possibility of overlap of the two areas. The code is set by a call to the GMVBCD routine of GRFMAN.

LENGTH

The LENGTH routine is called to calculate the length of an arc in the output projection space. The arc endpoints are reprojected into the output projection and the Point file records are read and the interior points are also reprojected. The complete string of points for the arc is used to find an accurate arc length according to the reprojected points.

LNGSHRT

One of the special line types which may be assigned to a group of arcs by specifying a range of feature codes is the long-short and the long-short-short dashed lines which are produced by the LNGSHRT routine. The long-short line type is often used to portray the county boundary and the long-short-short line often represents state boundaries. (In other contexts, these lines may represent state and national boundaries). The construction of these lines is based on the reprojected points which have been filtered by the clipping window when the clipping option has selected. Each point generated for the dashes must also be compared to the clip window (this is done by other routines called to perform low-level dashing).

To make certain the dashes come out even, the dash and gap lengths selected by the operator are slightly modified (according to the arc length), so that an even number of them will be drawn to end at the second node evenly. Several special cases, based on the length of the shorter arcs in comparison to the length of the dash pattern, require a special pattern to be drawn in lieu of the usual dashing. The MHD2 routine is called to create the dashing pattern from one point to the next, while maintaining the pattern established by the previous segment.

MAIN

The MAIN routine of the MAPGEN program provides the central focus of all the activities of the program. It is at the top of the pyramid which is the production of map products from GIMMAP databases. The MAIN routine encompasses the beginnings of all the activities of the MAPGEN program. It opens, closes, and accesses all the map databases listed in the Names file to get all map data for processing (except some access by special line routines). It directs the input, clipping, reprojection, weeding, and reforming for symbol annotation of all the map data sent to the output files.

The MAIN routine in the MAPGEN program sets up each input projection from values in the input map database; guides the basic construction of DDF or plot output to multiple output files and calls appropriate routines to perform the output to multiple files; oversees the creation of the map header, control points, feature codes and flags for DDF files; oversees the production of special

plot features like the GIMMAP label, time and date, registration marks and labels, map frames, and posting of the output projection name and the map scale.

The MAIN routine first opens the terminal input and output files for communication with the operator. Then, the operator is offered the option to ignore all map databases which lie outside the area of the output projection, to be specified later. These maps may be ignored or may be kept. Maps which are to the west or south of the area may result in creation of negative coordinates, and maps north and east may extend beyond the plotter limits. Both conditions should be avoided.

The MAIN routine then calls the SETUP routine to set nearly all the operator choices for the execution of the MAPGEN program. In SETUP, the output type (plot or DDF), clipping option, the Names file, output projection parameters, output files and associated code ranges, special point symbol assignments, assignments for the line type arc, and other options are all set by the operator. The MAIN routine then enters a loop which is designed to perform all the desired functions of MAPGEN for each map database in the Names file, in each pass through the loop.

For each database, the Arc and Point files are opened, the parameters for the input projection are read and that projection is set up in the input projection software (the PROMAN library). Certain illegal projection combinations (non-projected input to projected output and projected input to unprojected output without the lat/long form for output) are tested and rejected. The total number of arcs in the database is checked to see that it does not exceed the limit of ten thousand. The database and projection information is reported.

For all arcs in each map database, the MAIN routine first checks for special line types (for the plot option) and sets the specifications for that type from the list created in SETUP. The feature code is set for the arc so all arcs of this type are to be processed at one time (to reduce headers for DDF files).

If the arc type is for isolated points, all isolated points of the type set above are first checked for being in the clipping window or out. Then, the points are output to the DDF files as a single point or to the plot as the desired point symbol by calling the MLTSYM routine with specifications loaded earlier for the type of arc to be plotted. Counts of points rejected by the clipping window and those written or plotted are incremented for each.

If the arc type is for linear arcs, the IMAGE routine is called to determine if the arc is entirely outside the clipping window. Such arcs are skipped, and a count of rejections is incremented. The points are clipped if the clipping option is on, and counts are incremented for all arc points which are accepted as well as those rejected. Reprojected points which are in the clipping window must also be weeded to remove excess points created

by the change of scale. A count is also made of all points which are rejected on the basis of weeding.

All line arcs which require a special annotation style (such as double, hashed, dashed,...) are processed by a special routine in the MAPGEN program. All other arcs are processed in the MAIN routine with the start node checked first to be in or out, followed by all the interior points and then the end node. In some special cases where the arcs do not require interior points (according to the weeding criteria) following reprojection, the end node is drawn to or written following the start node. The start node is drawn with pen up for plots, and the end node in the DDF file is followed by a discontinuity flag.

At the end of the loop which processes all the arcs for a given database, the Arc and Point files for the database are closed and the report of the numbers of arcs and rejects for the database are made.

When all input databases have been processed, the plot or DDF files are ended, with the second part of the origin mark added to the plot files. The counts of arcs put in each of the output files is reported to the operator and the total number of points rejected by the weeding threshold is also printed before the MAPGEN program is halted.

MDBL2

The MDBL2 routine is one of the group of routines that produce the double line annotation for linear arcs in MAPGEN. This routine is called to perform one special calculation which is needed in the production of double or double-dashed lines. The calculation is to find the turning points of the two double-line edges at the middle point of a three point arc segment which begins at (X1,Y1), goes to (X2,Y2) and then to (X3,Y3). In this case, (X2,Y2) is the middle point. The double line edges begin at a distance (half the width of the double line) from the first point, go to the point at which they intersect with the line that bisects the angles formed by the three point segment, and then go directly towards the third point. At all times, the double line edges are parallel to the segment which is nearest them. Vertical and horizontal segments are treated as special cases which require less processing.

MDBL3

The MDBL3 routine performs the steps which are necessary to carry the process used in MDBL2 form one end to the other of the arc. Clearly, the MDBL2 routine, once started, can move the double line down the length of the arc by generating the turning points at each middle point along the way. The third point in each three point segment then becomes the next middle point, and the next turning points are calculated in the same fashion. However, the

starting points for the double edges at the very first point of the arc (the start node or the first point in the clipping window) must be calculated by the MDBL3 routine to provide the starting point for the entire arc. Similarly, the MDBL3 routine is used to find the ending points at the very last point of the arc (or the last point inside the clipping window), by turning the points around and giving the last point to MDBL3 as the first point in the three point arc segment.

MDD2

The MDD2 routine produces the plot output for a double dashed line, but only for a three point segment of the line as described in the MDBL2 routine above. The process of producing the double, dashed line is simply to generate the double line edges for each of the three point segments, but to then draw each edge using the dashed line algorithm defined elsewhere (see the MHD2 routine). The MLTDD routine sets up the double line and calls the MDD2 routine to generate the dashed lines for the right and the left sides of the double line.

MHD2

The MHD2 routine is called to generate the plot output for the hashed, dashed, long-short dashed, and the long-short-short dashed lines, but only for a single segment from one point to the next. The process of creating dashed or hashed lines between two points is one of moving a distance (set by specifications made by the operator in the SETUP routine) and then lifting the pen or drawing a hash mark. Then, moving a distance (the same for hashing, maybe different for dashed lines) and drawing a hash or putting the pen down. The process for two points may be repeated for each pari of points defining the arc, provided the state of the process (e.g. the distance travelled since the last hash or pen change) must be passed from each pair of points to the next in order to maintain the specifications governing the line type. Certain short arcs which can not display the desired line type or specifications are treated in special ways to produce acceptable results.

MLTCHK

Each of the (up to fifteen) output files for DDF or plot data is associated with its own list of (up to fifteen) feature code ranges which determine the map features to be sent to the file. The MLTCHK routine is called when a new feature code is to be used to determine which arcs in the current map database are processed and output. For all output files which have the new feature code qualified in one or more code ranges, a flag is set in an array which shows that the arc(s) of that code are to be sent to the file in which the code is accepted. By this scheme, each type of map feature may be placed in as many output files as desired.

MLTCIR

The MLTCIR routine is called to create plot output to draw a circle, centered at a specified location, of a size specified by the radius, (all specifications determined by the operator in the SETUP routine), and written to all of the multiple output files for which the feature code has been qualified by the MLTCHK routine. The circle is drawn by fixing the number of steps required to make the circle appear smooth. This number is dependent on the size of the circle, and is used to calculate the sine and cosine of the angle which creates the circle in the number of indicated steps. The points are created in a counter-clockwise direction starting at the zero angle (positive X axis) and each point is the rotation of the previous point through the angle calculated above. Each new point is added to all the plot files.

MLTDBL

The creation of double lines for plot output in the MAPGEN program is performed by the MLTDBL routine in conjunction with two other routines described above. The MDBL2 routine finds the turning points for the double line edges at the middle point of a three point segment, and the MDBL3 routine finds the two starting points for the double line edges at the specified distance from the endpoint or starting point for the arc. If the arc has no interior points, the process is a matter of calling MDBL3 twice and then performing the clipping, if necessary, for the two edges. At each endpoint (node) of each arc, the double line may be drawn to be even with the node or may leave a gap which is half the separation of the double lines, as chosen by the operator in SETUP.

For arcs with interior points, the MLTDBL routine uses both the MDBL2 and the MDBL3 routines to calculate a path along one side (the right) of the double line from the start node to the end node. Then, the process is reset, and a path is generated along the other side (left) from the start node to the end node. While both paths could be generated together, saving half of the work and time in accessing the database, the changing back and forth between the two sides of the double line would cause deterioration of the line quality on vector plotters.

MLTDD

The creation of double dashed lines (lines which are drawn by two dashed lines separated by a specified distance) is under the control of the MLTDD routine. The construction of double and dashed lines is described above in the MLTDBL and MHD2 routines. As with double lines, the beginning of double dashed lines may be even with the node or may leave a gap of half the separation of the double line, as selected by the operator. Lines which are too short for normal annotation are treated specially. Lines with no

interior points are plotted in a shorthand fashion. Other arcs are plotted by finding the correct starting points in a call to the MDBL3 routine and processing all interior points such that each is the middle point in a three point segment which is passed to the MDD2 routine to find the turning points.

Each point then becomes the next first point, and the old third point becomes the new middle point of the next three point segment. Finally, the end node becomes the final point and a second call to the MDD2 routine is used to find the last points for the two edges of the double dashed line. As with other dashed lines, the (reprojected) length of the arc is used to reset the dash length slightly to guarantee that the dash pattern will fit evenly along the length of the arc.

MLTDRW

When points are to be sent to the multiple output files for plotting, the MLTDRW routine is called. It is assumed that points have been checked against the weeding criteria, and that concern is not a factor in the MLTDRW routine. The primary responsibility of the MLTDRW routine is to perform the clipping function when that option has been selected. To do this, each point is sent to the GMVCOD routine of the graphics management library (GRFMAN) to find out if the point is inside or outside the window. If the clipping option is in latitude/longitude, then the geographic coordinates of the point (available from the reprojection) are used instead of the reprojected x-y coordinates.

If the point is the first of the arc, then being in or out fully determines the use of the point. For other points, the clip code received for the point is then used with the clip code of the previous point and knowledge of the type of clipping to determine what is done with the point. Either the point is plotted as is, the segment between it and its predecessor is clipped and that point is used (possibly along with the current one), or the point is discarded. In any case, the point location and the clip code of the point are saved as the new <u>previous</u> point for processing of the next point in the arc.

MLTFIN

When a plot file is to be closed, certain commands must be sent to the file to act as a warning to the plotter and plotter operator that the plot is ended. The sending of these special plot ending commands to the multiple (plot) output files used by the MAPGEN program is the task of the MLTFIN program.

the plot size or maximum coordinates in the file). The MLTINI routine performs this initializing function for all the files which were opened for plotting. The routine then reports the maximum plot size which was written to each file.

MLTIPT

Isolated point arcs which are written to the DDF output files, having passed the clipping and feature code mask tests, are written by the MLTIPT routine. Depending on the output form (projected or geographic coordinates) and the optional inclusion of z-values for each point feature, one of four possible formats may be used by the MLTIPT routine to write each point to the multiple output files.

MLTLIN

The MLTLIN routine is called to output the points of line arcs to the multiple DDF output files. This process is complicated only by the clipping window. If clipping is off, the reprojected point is written in the appropriate form (geographic or projected coordinates) with no other concerns. If the point is the first for the arc, then the MLTHEAD routine is called to add the feature code header to the files. However, when clipping is on, the process is more involved.

First, the clipping code for the point is obtained by calling the GMVCOD routine. Then, the combination of the clip codes for the current and the previous points and the type of clipping will determine the appropriate actions. The new point may be added to the output files without change, the segment may be clipped and the clip point (and possibly the new point) added to the output files, or the new point may simply be rejected. At the end, the new point location and clip code are saved for processing the next point of the same arc.

MLTMOV

The MLTMOV routine is called to cause a move (with pen up) to the specified point for the plot output to the multiple plot files selected by the operator. This operation is identical to the draw operation (MLTDRW) except that the pen is lifted for the move. The MLTMOV routine maintains the clipping option when the clipping mode is on by calling the GMVCOD routine to return the clip code for the point. This value, along with the location of the point is then saved for use with the next point. Since this point is only moved to, no real clipping is performed, but the point is not inside the window then it is not written to the files.

MLTFLG

The syntax of the GIMMAP digitized data format (DDF) file requires a special marking to separate the end node of an arc and the start node of the next arc when the two arcs are not connected (i.e. when there is a discontinuity). Similarly, the syntax also requires a clear marking to signal the end of an arc and the coming of a feature code header to specify the new feature code to apply to the following digitized features. In the first case, the syntax expects a pair of negative coordinates called a discontinuity flag, and in the second a double pair of such coordinates called a header flag. The MLTFLG routine is called to insert a single pair of negative coordinate in the multiple DDF output files.

MLTHD

Arcs which are to be hashed or dashed are processed by the MLTHD routine. In techniques described above, the MLTHD routine produces the hashed or dashed output to the multiple (plot) output files. Some work is done in this routine, and much is done by a call to the MHD2 routine. Arcs which are too short for normal processing of hashed or dashed lines are treated specially to produce acceptable results. Both kinds of line types require the adjustment of the dash, hash spacing or gap lengths to guarantee even cycles for the length of the arc. For lines of normal hashing or dashing, the line begins at the start node, follows the arc by calls to the MHD2 routine, and finishes at the end node. The points are reprojected in the MLTHD routine, but are not clipped there.

MLTHEAD

When the DDF option is selected, each change of the feature code in the processing of arcs for a map database requires a change in the feature code controlling the DDF file (unless the new code is the same as the previous one - a possibility excluded by the method of selecting arcs in the MAIN routine). Changing a feature code in the DDF file requires that a feature code header flag (two consecutive pairs of negative coordinates) is first entered to signal the coming feature code header, according to the rules of the GIMMAP digitizing syntax. The feature code header is simply a pair of coordinates (like a point), which contains the feature code as the x-value. Both the feature code header flag and the feature code header are put in the multiple DDF output files by the MLTHEAD routine.

MLTINI

When the plot option is selected in MAPGEN, all plot files selected by the operator must be initialized by MAPGEN to signal the plotter software certain information about the plot (such as

MLTSYM

The MLTSYM routine is used to create the special point symbols in the multiple plot output files for MAPGEN. The routine is given the location of a point, the type of symbol to be drawn, and the size of the symbol in plotter units. For solid-filled symbols, the routine is also given a separation distance between the multiple symbols of varying size. The routine then branches to areas where common line work is drawn for different symbols and to other areas to draw unique parts of the symbols, based on the symbol type that is specified.

Included in the symbols which the MLTSYM routine can create are the plus symbol (+, the default for point symbols), the letter X, the asterisk (*, equal to the plus and the X), the square, the circle, the triangle, the fiducial (a square with a plus and an X), a hexagon, and an inverted triangle. All these symbols are defined by the size of the symbol specified by the operator (e.g. the size = the side of the square or the diameter of the circle).

The MLTSYM routine also draws a set of symbols from the Oil and Gas industry including the oil well (a half-size solid circle), the gas well (a circle with 8 rays), the oil and gas well (a solid circle with 8 rays), the dry hole (a circle with 4 rays at NW, NE, SE, and SW angles), and the miscellaneous symbol (a circle with 3 rays at NE, NW, and S angles).

In addition, the MLTSYM routine can generate three symbols which require the pen separation parameter to guide the generation of solid fill. These symbols are the solid circle, the solid triangle, and the solid square. Successful creation of the solid symbols for a vector plotter may require some experimentation to produce aesthetic results without damage to the plot medium.

REGLAB

The REGLAB routine creates the labels for the registration marks, if both options are selected by the operator. The content of the labels, namely the locations of the points in geographic coordinates, is created in the MAIN routine before calling the REGLAB routine. The parameters guiding the creation and placement of the registration labels (such as font, height, orientation, ...) are also set in the MAIN routine. The REGLAB routine calls the GMPTTL routine in the GRFMAN library to generate the pen strokes of the labels and pass them to a single output file (the first).

REPRO

The reprojection of points from the input projection (in which they are stored in their map databases) to the output projection specified by the operator is controlled by the REPRO routine. Each point is deprojected according to the input projection. When

output is in geographic (latitude and longitude) coordinates, no further action is taken. When geographic clipping is on, then the geographic coordinates are saved for that purpose. When output is in projected coordinates, the geographic coordinates are projected according to the output parameters. In this case, the option of adding translation offsets may be employed also.

SETOUT

Selection of the multiple output files for either the DDF or the plot option in the MAPGEN program is performed by the SETOUT routine. A maximum of fifteen output files may be selected in this process, and each file may be associated by a maximum of fifteen feature code ranges, defining the types of features which are to be put in the file. The operator first names the file to be used and if the DDF option is on, then the operator enters a map header for this file. Clearly, each DDF file may have its own unique map header, distinct from the other DDF files.

Then, for each file the operator selects, the operator selects the feature code ranges for the file. Each code range is entered as a low and a high code in free format (values separated by commas or by one or more blanks or tabs). Each chosen range is checked to make sure that the codes are legal and the high code is higher than the low code. Errors cause retrys. When all ranges are entered, the process is halted by the operator entering a negative value for the low code.

The SETOUT routine then displays the choices of the operator as a table with a title of the file name selected, followed by the list of feature code ranges listed as low-high codes. The operator is asked to accept or reject the list by typing 1 or 0. If the list is rejected, the operator begins anew with the name of the file to be used. When the selection is accepted, the SETUP routine proceeds to open the file as a DDF or a plot file according to the selection of the operator. When all files have been selected, the SETOUT routine reports the total number of files and returns.

SETUP

The SETUP routine performs the comprehensive list of setup functions required to gather information from the operator and to prepare to perform all the functions of the MAPGEN program, as outlined in the MAIN routine section above. The functions of the SETUP routine are described here in an operational sequence as they occur in the execution of the routine. The detailed operations of the SETUP routine are presented here in only a general sense.

First, the Names file (containing the Arc file names for the map databases to be processed) name is given by the operator and the file is opened. If a single Arc file is used, the direct form (name preceded by a *) may be entered and that file opened as an

Arc file. The operator then selects the output form which defines whether the output is DDF or plot form, with or without clipping, and which type (projected x-y or geographic) clipping if chosen.

The plotter size maxima may be reset, and the clipping window is defined by the operator and set by a call to GMVWND, if that option is chosen. The window is defined by the bounding rectangle in projected x-y coordinates or geographic coordinates. If the clipping option is selected, the boundaries of the clipping window are displayed in the original units (either projected x-y or geographic coordinates in degrees) and in radians for comparison to the later displays of projection initialization.

For the DDF output option, the form of the output is set by the operator. It can either be in projected x-y or in geographic coordinates. If the latter, the longitude may be unsigned or may be signed negative for the west longitudes. In both cases, the geographic coordinates are in degrees.

Next, the operator selects the output projection parameters, beginning with the type of projection (None, Modified Polyconic, Lambert Conformal Conic, or the standard Kansas projection - the Lambert projection with fixed parameters). The output projection area must be defined by the minimum and maximum longitude and latitude of the bounding rectangle. For the standard Kansas map projection, these values are fixed (south = 36.875, north = 40.125, west = -102.125, east = -94.5). For standard USGS 7.5' quadrangle map names (a la GIMMAP system at the KGS), the list of Arc file names may be (optionally) read to determine (from the row and column numbers in the names) and set the area.

The map scale is requested and provided by the operator. The map scale must be positive, and determines the size of output products. If the Lambert Conformal Conic projection is selected, some additional parameters must be provided. The two standard parallels must be given (in degrees latitude). These values uniquely define the cone used in the reprojection process (these are defined for the standard Kansas projection as 33. and 45.).

The output projection parameters are listed for the operator as the projection is initialized. If no output projection has been selected, the operator is asked to enter the extreme values which define a bounding rectangle for the output area. The operator may then set the translation offsets to move all data upon output. The default weeding threshold is displayed and the operator may reset it to a new value. When the DDF option is on, the operator is asked to select options for geographic output, if selected, such as feature code headers, discontinuity flags, and the addition of z-values for isolated points.

For plot output, the operator selects the options of frames, registration marks, labels for registration marks, the location of registration marks, size of the marks, a single or double map frame and an origin mark to check loss of origin by the plotter. The

GIMMAP label with the output projection type and scale are optional for the plot files, and addition of the current date and the time of plot creation are an additional option usually added to only one of the multiple output files. The GIMMAP label is printed for the operator who may cancel its use.

Numerous checks are performed for the above options to determine if the original plot or its extra options cause the plot to exceed the limits of the plotter in any direction. For some of the cases, adjustment can be made to accommodate the situation, others require operator modifications or termination of the program.

The final operation in the SETUP routine is for the operator to define the type and specifications for the point symbols and the line annotation for plot output. This process is one in which the operator enters a line of values to specify each group of features which are to be plotted as special point symbols or as special annotated lines. The line of values entered contains the low and high code for the range of feature codes to which the annotation is to apply, and a number from the list presented by SETUP which specifies the point symbol or the line type.

Each of the point symbols and line types listed above has a code associated with it as displayed by SETUP. The operator types the code range of features followed by one of the code numbers and all features within the code range that are selected for output to the plot files will be plotted as the type of point or line which is specified. To completely specify the point symbols and line types for production, additional information (e.g. point size or length of dash or gap or hash) must be given by the operator. Up to twenty specifications may be made in a single MAPGEN run.

Technical Note

C. Ross

August, 1985

GRFMAN Routines

The Graphics Management Library

INDEX

IV. GRFMAN Plotter Routines (Plotter) 11

Plot a box	GMPBOX
Plot circle around point	GMPCIR
Plot a line to a point	GMPDRW
Plot dashed line between two points	GMPDSH
Finish plotting commands	GMPFIN
Plot frame box	GMPFRM
Plot hexagon around a point	GMPHEX
Initialize plotting	GMPINI
Move to a point	GMPMOV
Post an integer number	GNPNUM
Select Plotter pen position	GMPPEN
Plot plus symbol at point	GMPPLS
Plot alphanumeric Title	GMPTTL
Plot an X at a point	GMPXIT

V. Unbuffered Control Routines (TEKTRONIX) 16

Set mode to alpha-numeric	GMCAMD
Ring terminal bell	GMCBEL
Set character size	GMCCSZ
Erase screen, set alpha mode	GMCERA
Finish use of graphics routines	GMCFIN
Set graph mode (or pen up)	GMCGMD
Hardcopy terminal screen	GMCHCP
Initialize graphics routines	GMCINI
Advance alpha cursor (new line)	GMCNUL
Set Z mode (write thru, etc.)	GMCZMD

VIII. Unbuffered VIRTUAL Routines (TEKTRONIX) 26

Draw a box	GMVBOX
Draw a circle around a point	GMVCIR
Clip a line segment	GMVCLP
Calculate clip code	GMVCOD
Input cursor location	GMVCSR
Convert virtual to screen	GMVCVT
Draw solid line segment	GMVDRW
Draw dashed line segment	GMVDSH
Draw window frame	GMVFRM
Draw hourglass figure at point	GMVHRG
Return window limits	GMVLIM
Draw dark line (move)	GMVMOV
Post integer at point	GMVNVM
Draw plus symbol at point	GMVPLS
Post real number at point	GMVRNM
Set Virtual Window limits	GMVWND
Draw "X" symbol at point	GMVXIT

NOTE: Except where otherwise indicated, variables beginning with the letters I, J, K, L, M, N represent INTEGER*2 variables. All others should represent REAL*4 variables.

The GRFMAN (Graphics Management) library is written in FORTRAN 77 only.

The file name for linking is:

:KGSVS2: GIMMAP: GRFMAN77.LB

The HERSHEY4 File (Usually IHFC) is:

:KGSVS2: GIMMAP: HERSHY4

The ALPTRS4 file (IHPFC) is:

:KGSVS2: GIMMAP: ALPTRS4

I. Buffered Control Routines (GMB---)

GMBFIN - Finish Buffered Graphics

CALL GMBFIN

Called prior to GMCFIN, closes buffered environment.

GMBGMD - Set Graphics Mode

CALL GMBGMD

Changes terminal mode to graphics. Additional calls create dark vector for move.

GMBINI - Initiate Buffered Graphics

CALL GMBINI

This routine must be called following GMCINI since multi-tasking is used to perform buffered graphics.

GMBPLT - Empty the Plot Buffer

CALL GMBPLT.

GMBZMD - Set Z Axis Mode

CAll GMBZMD (MODEZ), where

MODEZ = 8*Axis Type + Line Type, where

Axis Type =

0 - Normal (Narrow Beam)

1 - Defocus (Wide Beam)

2 - Write-Thru (Temporary)

Line Type =

0 - Normal (No Dash)

1 - Dotted

2 - Dot Dash

3 - Short Dash

4 - Long Dash

II. Buffered Screen Routines (GMBS--)

GMBSDR - Draw Line Segment to Point

CALL GMBSDR (IX, IY), where

IX = X screen coordinate

IY = Y screen coordinate.

GMBSHR - Draw Hourglass Figure

CALL GMBSHR (IX, IY, ISIZ), where

IX, IY = Screen coordinates of center location

ISIZ = Size in screen units.

GMBSNU - Post Integer at Point

Call GMBSNU (IX, IY, NUM), where

IX, IY = Screen Coordinates of Lower-Left corner.

NUM = Integer to be posted.

GMBSPL - Draw Plus (+) Symbol

CALL GMBSPL (IX, IY, ISIZ), where

IX,IY = Screen coordinates of center location

ISIZ = Size in screen units.

GMBSXI - Draw "X" Symbol (XIT)

CALL GMBSXI (IX, IY, ISIZ), where

IX, IY = Screen coordinates of center location

ISIZ = Size in screen units.

III. Buffered Virtual Routines (GMBV--)

GMBVDR - Draw Line Segment to Point

CALL GMBVDR (X, Y), where

X, Y = Virtual coordinates of point.

GMBVHR - Draw Hourglass Figure

CALL GMBVHR (X, Y, ISIZ), where

X, Y = Virtual coordinates of center location

ISIZ = Size in screen units.

GMBVNU - Write a Number at a Point

CALL GMBVNU (X, Y, NUM), where

X, Y = Virtual coordinates of lower-left corner

NUM = Integer to be written.

GMBVPL - Draw Plus (+) Symbol

CALL GMBVPL (X Y, ISIZ), where

X, Y = Virtual coordinates of center location

ISIZ = Size in screen units.

GMBVXI - Draw "X" Symbol (XIT)

CALL GMBVXI (X, Y, ISIZ), where

X, Y = Virtual coordinates of center location

ISIZ = Size in screen units.

IV. Plotter Routines (GMP---)

GMPBOX - Plot a Box (square)

CALL GMPBOX (X, Y, SIZE), where

X, Y = Coordinates of the center of the box

SIZE = Length, in inches, of the sides.

GMPCIR - Plot a Circle of Given Radius at Center Point

CALL GMPCIR (XC, YC, RAD), where

XC, YC = Plotter coordinates of circle center

RAD = Radius in inches.

GMPDRW - Plot a Line to a Point

CALL GMPDRW (X, Y), where

X, Y = Plotter coordinates of the destination.

GMPDSH - Plot a Dashed Line between Two Points

CALL GMPDSH (X1, Y1, X2, Y2, DASH, GAP), where

X1, Y1 = Coordinates of First Point

X2, Y2 = Coordinates of Second Point

DASH = Length of Dash

GAP = Length of gap.

GMPFIN - Finish Plot Commands

CALL GMPFIN (XSIZE, YSIZE), where

XSIZE = Maximum x-dimension of plot

YSIZE = Maximum y-dimension of plot.

NOTE: Should be the last GRFMAN routine to be executed.

GMPFRM - Plot Frame (rectangle) at Indicated Maxima

CALL GMPFRM (XMN, YMN, XMX, YMX), where

XMN = X Minimum of frame

YMN = Y Minimum of frame

XMX = X Maximum of frame

YMX = Y Maximum of frame.

GMPHEX - Plot Hexagon Centered at Point

CALL GMPHEX (S,Y,SIZE), where

X, Y = Center of hexagon

SIZE = Distance from side to opposite.

GMPINI - Initiate Plotting

CALL GMPINI (XSIZE, YXIZE, IOFC), where

XSIZE = Maximum x-dimension of plot

YSIZE = Maximum y-dimension of plot

IOFC = Plot output file code.

NOTE: Must be the first GRFMAN plot routine to execute.

GMPMOV - Move Pen to a Point

CALL GMPMOV (X, Y), where

X, Y = Coordinates of the destination.

GMPNEW - Begin a New Plot, Same File

CALL GMPNEW (XSIZE, YSIZE), where

XSIZE, YSIZE are maximum dimensions of new plot.

GMPNUM - Plot an Integer at a Point

CALL GMPNUM (NUM, XL, YL, XR, YR, ISTR, HGT, ORI,
ISET, JUST IHFC, IHPFC), where

NUM = Integer to be plotted

XL, YL = Location of lower left corner

XR, YR = Location of lower right corner
(optional)

ISTR = Array for local storage

HGT = Character height in inches

ORI = Angle of orientation in positive degrees

ISET = Hershey character set number (1 - 23)

JUST = -1, left-justified
= 0, centered
= 1, right-justified

IHFC = File code for Hershey Alphabet file

IHPFC = File code for Hershey point file.

NOTE: ISTR must be a 100-element CHARACTER*1 array. The user must open the two Hershey files, HERSHEY4 and ALPTRS4.

GMPPEN - Select Pen Position

CALL GMPPEN (IPEN), where

IPEN = Pen Selection (1-4).

GMPPLS - Plot Plus Symbol (+)

CALL GMPPLS (X, Y, SIZE), where

X, Y = Location of center of symbol

SIZE = Total size of symbol in X and Y.

GMPSYM - Plot Point Symbols

CALL GMPSYM (X, Y, ISYM, SIZE, IFC), where

X, Y = Location of center of symbol

ISYM = Symbol Type

1 = +	5 = circle
2 = x	6 = triangle
3 = *	7 = fiducial
4 = square	8 = hexagon

. . . . . .

11 = oil well	14 = dry hole
12 = gas well	15 = miscellaneous
13 = oil & gas well	21 = inverted triangle

SIZE = Symbol size in inches

IFC = Plot file unit number.

GMPXIT - Draw "X" Symbol

CALL GMPXIT (X, Y, SIZE), where

X, Y = Location of center of symbol

SIZE = Total size of symbol in X and Y.

GMPTTL - Plot Alphanumeric Title

CALL GMPTTL (XL, YL, XR, YR, TTL, MARK, ORI, ISET, JUST, IHFC, IHPFC), where

XL, YL = Lower-left corner of Title

XR, YR = Optional lower-right corner,(-1., -1.) if unused.

TTL = Content of Title, CHARACTER*1, 100-element array.

MARK = Character used to mark end of content, a CHARACTER*1 variable.

ORI = Orientation in positive degrees

ISET = Hershey alphabet set

JUST = Justification as for GMPNUM

IHFC = Unit number of HERSHEY4 file

IHPFC = Unit number of ALPTRS4 file.

NOTE: The user must open the two Hershey files.

V. Unbuffered Control Routines (GMC---)

GMCAMD - Set Alpha Mode

CALL GMCAMD.

GMCBEL - Ring Terminal Bell

CALL GMCBEL.

GMCCSZ - Set Character Size

CALL GMCCSZ (ISIZ), where

ISIZ = Size of characters

1 = Largest

2 = Next largest

3 = Next smallest

4 = Smallest.

GMCERA - Erase Terminal Screen

CALL GMCERA.

GMCFIN - Finish Use of GRFMAN Routines

CALL GMCFIN (IX, IY, ISIZ), where

IX, IY = Screen coordinates of final position of cursor

ISIZ = Size of character upon termination.

NOTE: Should be last GRFMAN routine to be executed.

GMCGMD - Set Graphics Mode

Call GMCGMD

GMCHCP - Initiate Hardcopy of Terminal Screen

CALL GMCHCP.

GMCINI - Initiate GRFMAN Routines

CALL GMCINI (ISIZ, IIFC, IOFC), where

ISIZ = Character size selection

IIFC = Input File code (terminal)

IOFC = Output File code (terminal).

NOTE: Must be the first GRFMAN routine to execute. Also sets the initial screen and virtual windows. These may be reset by GMSWND and GMVWND.

GMCNUL - Move Cursor to Next Line

CALL GMCNUL (LINES), where

LINES = Number of lines remaining in current page.

NOTE: Will reset to top of page when full.

GMCZMD - Set Z Axis

CALL GMCZMD (MODEZ), where

MODEZ = 8*Axis Type + Line Type, with

Kansas Geological Survey

Open File Report 1988-45c

Missing Page #18

VI. Unbuffered Screen Routines (GMS---)

GMSBOX - Draw a Box (square)

CALL GMSBOX (IX, IY, ISIZ), where

IX, IY = Screen coordinates of center location

ISIZ = Size in screen units.

GMSCIR - Draw Circle of Given Radius at Center Point

CALL GMSCIR (IXC, IYC, IRAD, IX1, IY1, IX2, IY2, ICLIP),

IXC, IYC = Screen coordinates of circle center

IRAD = Radius in screen units

IXI, IYI = Screen coordinates of window minimum

IX2, IY2 = Screen coordinates of window maximum

ICLIP = Indicates whether clipping of circles at window edge is desired

(1 = clipping desired, 0 = No clipping).

NOTE: If clipping is not desired (ICLIP = 0) and if circle is too large to fit entirely within window limits, ICLIP is reset to -1 by the routine.

GMSCLP - Clip a Line Segment at Window Limits

CALL GMSCLP (IX1, IY1, IX2, IY2, ICOD), where

IX1, IY1 = Screen coordinates of point one

IX2, IY2 = Screen coordinates of point two

ICOD = Clip code of point one.

NOTE: Point two is assumed to be inside the screen window, while point one is assumed outside. The window limits must be set by a call to GMSWND and can be obtained using GMSLIM.

GMSCOD - Calculate Clipping Code for a Point

CALL GMSCOD (IX, IY, ICOD), where

IX, IY = Screen coordinates of point

ICOD = Clipping code returned

0 - Inside window

+1 - If less than Y minimum

+2 - If greater than Y maximum

+4 - If less than X minimum

+8 - If greater than X maximum.

NOTE: The code assigned is relative to the screen window as defined by the last call to GMSWND.

GMSCSR - Graphic Cursor Input

CALL GMSCSR (ICHAR, IX, IY), where

ICHAR = Cursor control character (CHARACTER*1)

IX, IY = Screen coordinates of cursor location.

GMSCVT - Coordinate Conversion from Screen to Virtual

CALL GMSCVT (IX, IY, X, Y), where

IX, IY = Screen coordinates

X, Y = Virtual coordinate

NOTE: This conversion is based on a mapping from the screen window (as set by the last call to GMSWND) to the virtual window (as set by GMVWND).

GMSDRW - Draw Line Segment to a Point

CALL GMSDRW (IX, IY), where

IX, IY = Screen coordinates of point to which segment is drawn.

GMSDSH - Draw Dashed Segment to a Point

CALL GMSDSH (IX, IY, LEN), where

IX, IY = Screen coordinates as for GMSDRW

LEN = Line type as in GMBZMD.

GMSFRM - Draw a Frame at Screen Window

CALL GMSFRM (LEN), where

LEN Line type as in GMBZMD.

GMSHRG - Draw Hourglass Figure at a Point

CALL GMSHRG (IX, IY, ISIZ), where

IX, IY = Screen coordinates of center location

ISIZ = Size in screen units.

GMSLIM - Return Screen Window Limits

CALL GMSLIM (IX1, IY1, IX2, IY2), where

IX1, IY1 = Screen coordinates of window minimum

IX2, IY2 = Screen coordinates of window maximum.

GMSMOV - Move Cursor to a Point

CALL GMSMOV (IX, IY), where

IX, IY = Screen coordinates of Point.

NOTE: Same as GMSDRW, but with a dark (invisible) segment.

GMSNUM - Post Integer at a Point

CALL GMSNUM (IX, IY, NUM), where

IX, IY = Screen coordinates of start location

NUM = Number to be posted.

GMSPLS - Draw Plus (+) Symbol at a Point

CALL GMSPLS (IX, IY, ISIZ), where

IX, IY = Screen coordinates of center location

ISIZ = Size in screen units.

GMSWND - Set Screen Window Limits

CALL GMSWND (IX1, IY1, IX2, IY2), where

IX1, IY1 = Screen coordinates of window minimum

IX2, IY2 = Screen coordinates of window maximum.

GMSXIT - Draw X at Point

CALL GMSXIT (IX, IY, ISIZ), where

IX, IY = Screen coordinates of center location

ISIZ = Size in screen units.

VII. Unbuffered Titling Routine (GMT---)

GMTITL - Display/Plot an Alphanumeric Title

CALL GMTITL (XL, YL, XR, YR, ISTR, ITERM, CHGT, ORI, ICSET, IJ, IDEV, IHFC, IHPFC), where

XL, YL = location of lower-left corner of the title.

XR, YR = location of lower-right corner of the title (negative if not specified)

ISTR = 100-element, CHARACTER*1 array which holds the title content

ITERM = Character which signals the end of the title, a CHARACTER*1 variable

CHGT = character height in inches

ORI = orientation of label in positive degrees

ICSET = HERSHEY character set (1-23)

IJ = justification of the title:

= -1 left justified

= 0 centered

= 1 right justified

IDEV = output device code:

= -1 XYNETICS plotter file

= 1 TEKTRONIX terminal

IHFC = file code of HERSHEY 4 (opened by user)

IHPFC = file code of ALPTRS4 file (opened by user).

NOTE: The following initilization routines must be called before calling these routines:

GMBINI and GMCINI - for interactive graphics terminal usage with buffered 1/O

GMPINI - for plotter output.

VIII. Unbuffered Virtual Routines (GMV---)

GMVBCD - Calculate -Overlap Code for Box

CALL GMVBCD (X1, Y1, X2, Y2, ICOD), where

X1, Y1 = Lower-left corner of box

X2, Y2 = Upper-right corner of box

ICOD = Overlap code

0 = Box entirely inside window

1 = Partially in and out of window

-1 = entirely outside of window.

GMVBOX - Draw a Box (square)

CALL GMVBOX (X, Y, ISIZ), where

X, Y = Virtual coordinates of center location

ISIZ = Size in virtual units.

GMVCIR - Draw Circle of Given Radius at Center Point

CALL GMVCIR (XC, YC, RAD, X1, Y1, X2, Y2, ICLIP),
where

XC, YC = Virtual coordinates of circle center

RAD = Radius in virtual units

X1, Y1 = Virtual coordinates of window minimum

X2, Y2 = Virtual coordinates of window maximum

ICLIP = Indicates whether clipping of circles at window edge is desired
(1 = clipping desired, 0 = No clipping).

NOTE: If clipping is not desired (ICLIP = 0) and if circle is too large to fit entirely within window limits, ICLIP is reset to -1 by the routine.

GMVCLP - Clip a Line Segment at Window Limits

CALL GMVCLP (X1, Y1, X2, Y2, ICOD), where

X1, Y1 = Virtual coordinates of point one

X2, Y2 = Virtual coordinates of point two

ICOD = Clip code of point one.

NOTE: Point two is assumed to be inside the virtual window, while point one is assumed outside. The window limits must be set by a call to GMVWND and can be obtained using GMVLIM. Changes X1, Y1 to a point on the window edge. Does not change X2, Y2, ICOD.

GMVCOD - Calculate Clipping Code for a Point

CALL GMVCOD (X, Y, ICOD), where

X, Y = Virtual coordinates of point

ICOD = Clipping code returned

0 - Inside window

+1 - If less than Y minimum

+2 - If greater than Y maximum

+4 - If less than X minimum

+8 - If greater than X maximum.

NOTE: The code assigned is relative to the virtual window as defined by the last call to GMVWND.

GMVCSR - Graphic Cursor Input

CALL GMVCSR (ICHAR, X, Y), where

ICHAR = Cursor control character (CHARACTER*1)

X, Y = Virtual coordinates of cursor location.

GMVCVT - Coordinate Conversion from Virtual to Screen

CALL GMVCVT (X, Y, IX, IY), where

X, Y = Virtual coordinates

IX, IY = Screen coordinates.

NOTE: This conversion is based on a mapping from the virtual window (set by the last call to GMVWND) to the screen window (set by GMSWND).

GMVDRW Draw Line Segment to a Point

CALL GMVDRW (X, Y), where

X, Y = Virtual coordinates of point to which segment is drawn.

GMVDSH - Draw Dashed Segment to a Point

CALL GMVDSH (X, Y, LEN), where

X, Y = Virtual coordinates as for GMVDRW

LEN = Line type as in GMBZMD.

GMVFRM - Draw Frame at Virtual Window

CALL GMVFRM (LEN), where

LEN specifies a line type as in GMBZMD.

GMVHRG - Draw Hourglass Figure at a Point

CALL GMVHRG (X, Y, ISIZ), where

X, Y = Virtual coordinates of center location

ISIZ = Size in screen units.

GMVLIM - Return Virtual Window Limits

CALL GMVLIM (X1, Y1, X2, Y2), where

X1, Y1 = Virtual coordinates of window minimum

X2, Y2 = Virtual coordinates of window maximum.

GMVMOV - Move Cursor to a Point

CALL GMVMOV (X, Y), where

X, Y = Virtual coordinates of point.

NOTE: Same as GMVDRW, but with a dark (invisible) segment.

GMVNUM - Post Integer at a Point

CALL GMVNUM (X, Y, NUM), where

X, Y = Virtual coordinates of start location

NUM = Integer to be posted.

GMVPLS - Draw Plus (+) Symbol at a Point

CALL GMVPLS (X, Y, ISIZ), where

X, Y = Virtual coordinates of center location

ISIZ = Size in screen units.

GMVRNM - Post Real Number at a Point

Call GMVRNM (X, Y, RNUM, NLEFT, NRITE), where

X, Y = Location of Lower-left corner of first digit

RNUM = Real Number to be posted

NLEFT = Number of digits posted left of decimal

NRITE = Number of digits posted right of decimal.

GMVWND - Set Virtual Window Limits

CALL GMVWND (X1, Y1, X2, Y2), where

X1, Y1 = Virtual coordinates of window minimum

X2, Y2 = Virtual coordinates of window maximum.

GMVXIT - Draw X at Point

CALL GMVXIT (X, Y, ISIZ), where

X, Y = Virtual coordinates of center location

ISIZ = Size in screen units.

Libraries - PROMAN

P R O J C T

Projection and Deprojection of Cartographic Data

Charles G. Ross
Automated Cartography
27 April 1987

The PROJCT System

The PROJCT system is a collection of FORTRAN77 subroutines adapted from GIMMAP. It is designed to perform map projection of points from the geographic (longitude-latitude) reference system to Cartesian reference systems in (x,y). It is also designed to 'deproject' points from (x,y) to longitude-latitude and to 'reproject' previously projected (x,y) points from one (known) projection to another.

The geographic reference system is used universally for locating points on the surface of the earth. It is thus the ideal intermediate reference system for translating between any two projected coordinate systems. However, it can not be used directly for cartographic display due to the great amount of distortion which results. Thus, both systems have wide usage, and conversion between the two is essential. This is the function of the PROJCT system.

Projection, Deprojection, and Reprojection

Map projection is a mathematical transferrence of points on the 3-dimensional surface of the earth to the 2-dimensional form of a paper map. It is the conversion from the geographic reference system of longitude and latitude to the common Cartesian coordinate system of (x,y). In the PROJCT system, only the single 'output' projection (defined by SETPRO) need be specified to perform projection.

Only two of many different map projections are supported in the PROJCT system. These are the Lambert Conformal Conic and the Modified Polyconic projections, both based on projection from the surface of the earth onto imaginary cones. Both projections require the specification of parameters such as the map scale (the ratio of the projected unit to the true unit on the surface of the earth), and the map area (area of interest) or the central lines of longitude and latitude.

Since the surface of the earth is 3-dimensional, and paper maps and graphics screens are 2-dimensional, no projection can simultaneously preserve the accuracy of scale, area and shape which would be ideal. Each unique projection will preserve parts of these, but all maps have distortions.

The conversion of projected (x,y) points to geographic (longitude-latitude) coordinates is called deprojection, and is essentially the inverse of the projection process. The same unique set of parameters used in the original projection must be provided to properly convert the points back into longitude and latitude. For deprojection, only the 'input' projection (defined by SETDPRO) need be defined for PROJCT.

In practice, deprojection is not the inverse of projection. Instead, knowledge of the projection provides an excellent first approximation of the latitude of the point being deprojected. Subsequent bracketing approximations are based on the errors of former ones until the point is located within acceptable tolerances. In USGS 7.5' maps (Scale 1:24,000), the first approximation is often the only one needed.

Projected points may be converted to other scales and projections through the process of reprojection. In this process, both an input (deprojection) and an output projection (projection) must be specified. The input projection parameters, used for deprojection, are set by a call to SETDPRO. The output projection parameters, used for projection following the deprojection, are set by a call to the SETPRO routine. Each point is deprojected according to the specified (original) input projection, then projected in the new projection.

Map Projection in the PROJCT System

The two map projections used in PROJCT (and generally in the state of Kansas) are both conic. The Lambert Conformal Conic projection is used in mid-latitudes with areas of large east-west extent. It uses two standard parallels and is the principal projection for maps of the whole state of Kansas. The Modified Polyconic projection is ideal for small areas in Kansas, and is based on a collection of cones, each tangent to the earth's surface at the latitude of the point to be projected.

The Lambert Conformal Conic projection is the most commonly used projection for the state of Kansas as a whole, because at mid-latitudes the projection minimizes distortion for areas of great east-west extent. For this reason, the Automated Cartography section has adopted it with the usual scale of 1:500,000 as their standard for whole-state maps and the GIMMAP system has incorporated this standard as an option wherever projections are performed. The standard is described below. The Lambert Conformal Conic projection is a conic projection with two standard parallels, and has been used traditionally by the U.S. Geological Survey whenever they map the whole state.

The Modified Polyconic projection is really a collection of many projections based on the cone. Points are projected on cones tangent to the earth at the latitude of each point. Unlike other projections, the points are not projected onto the same surface, but rather onto many surfaces. These surfaces are then (mathematically) positioned through simple scaling of the cones by the map scale. The result is a map which, at the scale of the USGS 7.5' quadrangle sheets (1:24,000), has distortion so slight as to be almost negligible for small areas.

For both the Lambert Conformal Conic and the Modified Polyconic projections, the map scale and the area of the projection must be described to PROJCT by the south and north latitudes and the west and east longitudes of a rectangangle bounding the area of interest to be mapped. The Lambert projection also requires specification of two standard parallels. For the standard Lambert projection for the state of Kansas, these values are shown below. Additional values control translation and location of the origin in the projection system.

Using the PROJCT System

Coordinates in the geographic system will be expressed as decimal degrees, i.e. 38 degrees, 37 minutes and 30 seconds will be expressed as 38.625 degrees. Longitudes in Kansas may be expressed as positive or negative. All longitudes are interpreted as west of the prime meridian. Coordinates in projected space may actually represent inches, miles or centimeters.

In the PROJCT software, there are three functions available. One is to project, one is to deproject, and the third is to reproject. Projection may occur given points expressed in longitude and latitude, and specifications for the exact (output) projection to be used. These are described below. Deprojection requires points in (x,y) and specifications for an (input) projection equivalent to the projection which originally created. Finally, a projected point in (x,y) created by the PROJCT system may be converted to longitude-latitude and then back into a different projection (x,y).

The object file containing the PROJCT routines is:

:UTIL:GIMMAP.PROJCT.OB

which must be linked to in the usual manner. The five routines needed for set-up, projection, deprojection and reprojection are:

SETPRO - Call to initialize the output projection, (before calling PROJCT or REPROJCT). Usually called once.

SETDPRO - Call to initialize the input projection (before calling DEPROJCT or REPROJCT). Called once.

PROJCT - Call to project from longitude-latitude to the (x,y) system. Call once for each point.

DEPROJCT - Call to deproject a projected (x,y) point back to longitude-latitude. Call once for each point.

REPROJCT - Call to deproject to longitude-latitude and then project to (x,y). Call once for each point.

Setting Up Input and Output Projections

To initialize the input and output projections, the following parameters must be set:

```
      INTEGER*4 NONEG,NPRO
      REAL EAST,NORTH,SCALE,SN,SOUTH,SS,WEST,XORG,XNE,XNW,
     +                                XSE,XSW,YORG,YNE,YNW,YSE,YSW
```

where,

NPRO = 1 for Modified Polyconic projection
= 2 for Lambert Conformal Conic projection
= 3 for the Standard Projection for Kansas (see below)

SCALE = The Scale Factor (Scale is 1 : Scale Factor)

SOUTH = Degrees latitude of South Edge of Area of Interest
NORTH = Degrees latitude of North Edge of Area of Interest
WEST = Degrees longitude of West Edge of Area of Interest
EAST = Degrees longitude of East Edge of Area of Interest

SS = South Standard Parallel in Degrees
SN = North Standard Parallel in Degrees

Note: Required for Lambert projection, For minimum distortion, set 1/6th of the map height from the south and north edges.

XORG = X-coordinate of southwest corner of Area of Interest
YORG = Y-coordinate of southwest corner of Area of Interest

Note: In GIMMAP, the southwest corner is usually set to (1,1). Use of the NONEG option (below) may overide these values.

NONEG = 0 for normal use (no special translation)
= 1 to translate if negative Y values are indicated
(Consult with Automated Cartography before using)

For the use of PROJCT or REPROJCT, the output projection must be set up. To do so, set the variables listed above and then execute:

```
      CALL SETPRO(NPRO,SCALE,SOUTH,NORTH,WEST,EAST,SS,SN,XORG,YORG,
     +                  NONEG,XSW,YSW,XSE,YSE,XNE,YNE,XNW,YNW,YMIN)
```

For DEPROJCT and REPROJCT, set the input projection variables to appropriate values (equal to those used to originally project the points) and call SETDPRO:

```
      CALL SETDPRO(NPRO,SCALE,SOUTH,NORTH,WEST,EAST,SS,SN,XORG,YORG,
     +                  NONEG,XSW,YSW,XSE,YSE,XNE,YNE,XNW,YNW,YMIN)
```

The following values are returned by SETPRO and SETDPRO:

XSW,YSW = Projected Southwest Corner of Area of Interest
XSE,YSE = Projected Southeast Corner of Area of Interest
XNE,YNE = Projected Northeast Corner of Area of Interest
XNW,YNW = Projected Northwest Corner of Area of Interest
YMIN = Minimum projected Y value in the Area of Interest
Note: Minimum Y occurs at center longitude, SOUTH latitude.

Standard Projection for the State of Kansas

When NPRO = 3 in the call to SETPRO or SETDPRO, the standard projection for the state of Kansas is assumed. The following assignments are made by the GIMMAP software, and these parameters need not be set by the user.

```
NPRO     =  2, Lambert Conformal Conic
SCALE    =  500000. , Scale is 1 : 500,000
SOUTH    =  36.875
NORTH    =  40.125
WEST     = 102.125
EAST     =  94.5
SS       =  33.
SN       =  45.
XORG     =  1.0
YORG     =  1.0
NONEG    =  0 , No Translation to Prevent Negative Y
```

Projection of Points After SETPRO Is Called

Each longitude-latitude point is projected with a single call to the PROJCT routine, passing longitude-latitude in decimal degrees and receiving (x,y) coordinates to be used in inches:

```
REAL PLAT,PLON,X,Y

     :

CALL SETPRO

     :

CALL PROJCT(PLON,PLAT,X,Y)
```

where

```
PLON,PLAT  = Longitude, latitude of the point
X,Y        = the projected point coordinates
```

Deprojection of Points After SETDPRO is Called

Deprojection of points which have been projected according to known parameters (especially if projected via GIMMAP) may be accomplished via the use of the SETDPRO and the DEPROJCT routines. To deproject points in (x,y) first set the parameters for deprojection, and call the SETDPRO routine one time. Then, to deproject each point, call the DEPROJCT routine:

```
INTEGER*4 SUCCESS
REAL PLAT,PLON,X,Y

     :

CALL SETDPRO

     :

CALL DEPROJCT(X,Y,PLON,PLAT,SUCCESS)
```

where

X,Y	= the projected coordinates of the point
PLON,PLAT	= longitude and latitude of deprojected point
SUCCESS	= 1 if deprojection was successful
	= 0 if deprojection failed (should always check)

Deprojection may fail, so the value of SUCCESS should be checked. Failures (with parameter values) should be reported.

Reprojection After SETPRO and SETDPRO are Called

To convert points from one projection to another, both SETPRO and SETDPRO must be called to define the parameters for both the "input" and the "output" projections. Set-up of these two routines is described above. Once both projections have been set up, points in the input projection may be converted to the output projection by a single call to REPROJCT for each point. The REPROJCT routine essentially calls the DEPROJCT routine and then gives the results to the PROJCT routine:

```
INTEGER*4 SUCCESS
REAL PLAT,PLON,X,Y

     :

CALL SETPRO
CALL SETDPRO

     :

CALL REPROJCT(XP,YP,XR,YR)
```

where

XP,YP	= Coordinates of point in old (input) projection
XR,YR	= Coordinates of point in new (output) projection
	= -9999.,-9999. if deprojection failed (always test)

Because it uses DEPROJCT, this routine may also fail. Errors should be reported along with parameter values.

IV.2 RAFMAN - Random File Management

Introduction

The GIMMAP system requires that the basic data files be of such a nature that any record can be accessed directly, and hence, the system is designed to use random files. Without random files, a record could only be accessed by searching (on the average) a majority of the file for a record with a particular key. This sequential process is costly, both in time and resources, and is circumvented by using random files.

The GIMMAP system also requires the capability of adding or deleting any given record, but has no concern for the chronological order in which additions and deletions occur. These facts suggest that a file management program would be feasible and desirable. Hence, at the least, one word of each deleted (or unused) record should be used to distinguish good (used) records from free (unused or deleted) records. This flagging process is one major feature of RAFMAN, the Random File Manager.

The second major feature is the linking of all free records in a doubly-linked chain. This is accomplished by using two words of each free record as pointers to the preceding and succeeding records in this "freelist." This feature prevents the need for file compaction which could become necessary after many records within the file were deleted. It also eliminates the necessity of searching the file for freelist (available) records when a new record is to be added.

The third feature of RAFMAN is the accounting record. This record is the first record of the file and contains a general picture of the file for easy inspection. It includes five integer values:

FIRST - A pointer to the start of the freelist

LAST - A pointer to the end of the freelist

MAX - The maximum length of the file in records

USED - The number of good or used records in the file

LREC - The length of the record in words

Use of this bookkeeping record is generally reserved for RAFMAN, and the above values are rarely altered by other programs.

File Structure

Each GIMMAP random file consists of three parts. The first part (A) consists of a record which contains accounting information for RAFMAN as described below. The second part (B) is the set of records which are currently used by the GIMMAP system. These records are not necessarily contiguous and their quantity is contained in the accounting record. The third part (C) consists of all other records in the file. This part is called the "freelist" since all records within it are free to be used by the system. All records in the freelist are doubly-linked by chain pointers.

(A) Accounting Record

The first record of the file is reserved for the accounting information. This accounting record contains five integer values and should not contain any others. These values are:

1) FIRST - The record number of the first available record in the freelist chain.

2) LAST - The record number of the last available record in the freelist chain.

3) MAX - The maximum number of records in the file (as set by RMINIT), limited by file creation parameters.

4) USED - The number of records in the file which do not belong to the freelist, that is, are being used by the system.

5) LREC - The fixed length (in words) of all records in the file.

(B) Used Records

There are only two restrictions on records used in the file when structured to use RAFMAN.

1. The record length must be set to at least five words to contain the values in the accounting record.

2. The first word used in a record should be a positive integer value. This is true if it is necessary for the system to distinguish between freelist records and used records (see below).

(C) Freelist

The freelist is a doubly-linked chain of records which are not currently used by the system. The advantage to having such a chain is

that used records may be released anywhere in the file and reclaimed without the need for compaction. A freelist record consists of three integer values:

1) FLAG - The first word of a freelist record will contain a -1 and may be used to distinguish freelist records from used records.

2) PRED - Contains the record number of the predecessor in the freelist chain.

3) SUCC - Contains the record number of the successor in the freelist chain.

The first and last records in the freelist are special since the first has no precedessor and the last has no successor. Hence, the first record in the chain has itself as a predecessor and the last record has itself as a successor.

RAFMAN Subroutines

There are seven routines which support the random file management in GIMMAP. These routines and appropriate calling sequences are described below.

RMCHKR - Check Record Number

This routine provides protection against access to records which lie outside the boundaries defined in the accounting record. Only those records which lie between the fixed accounting record and the last

available (MAX) record may be legitimately accessed. In the case of an error, the RMERRO routine is called to report the appropriate information, and the status flag is set to inform the calling routine of the error. It should be noted that the calling routine is responsible for determining if the selected record is a part of the freelist.

CALL RMCHKR (IREC, MAX, IFC, ISTAT), where

IREC = Record number

MAX = Maximum allowed record number

IFC = File code of associated file

ISTAT = 1, if IREC is legitimate

-1, otherwise.

RMERRO - Error Report

This routine reports a variety of errors related to random file management to the output file previously indicated to the RMIOFC routine. A call to RMERRO is made whenever any illegal record access is attempted, or when some inconsistency is found in the file. RMERRO prints a message containing the file code of the affected file and a predefined error code indicating the type of error. If this error occurs during an operation on the graphic display terminal, a hardcopy command will be issued to the terminal.

CALL RMERRO (ICODE, IFC), where

ICODE = Error code to be reported

IFC = File code of associated file.

RMGETR - Get a Record

This routine is called to obtain a record from the freelist. Provided that a record exists (the file is not full), the first available record is unchained from the freelist. If there is no available record, an error message is printed by RMERRO. Both conditions are indicated to the calling program which is responsible for resetting the freelist flag in the record.

CALL RMGETR (IFC, NUM), where

IFC = File code of affected file

NUM = Record number of the first available record in the freelist

= -1 if record is not freelist flagged.

RMINIT - Initialize a Random File

This routine initializes the account record contents and the free-list structure for the indicated file. The operator is informed of the current file size and the suggested size from the calling program. The operator then selects a maximum size for the file, and the appropriate record values are reset.

This routine is also called to extend the size of an existing file without restructuring the freelist. This occurs when RMGETR attempts to obtain a record from a full file, and causes the file MAX to be increased by 25 percent of its current value.

CALL RMINIT (IFC, MAXP, LENGR), where

IFC = File code of the file to be initialized

MAXP = Suggested maximum number of records

LENGR = Length of the records in words.

NOTE: The operator will be allowed to select the maximum record number based on the contents of the file and the value suggested in the call.

RMIOFC - Set Output File Code

This routine is called to inform the RAFMAN subsystem of the output file code for terminal output in case of error. This routine must be the first RAFMAN routine to be called.

CALL RMIOFC (IFC), where

IFC = The output file code.

RMPAGR - Page a Record (Point File Only)

This routine decodes a page number and datum offset number into a record number and intra-record offset value. It is used by GIMMAP programs to determine the location of coordinates in the Point File, which is a packed, paged structure.

CALL RMPAGR (NPAG, NUM, IREC, IPTR, ISIZ) where

NPAG = Page number containing the record

NUM = Datum number of item to be paged

IREC = Record number containing item

IPTR = Offset within the record

ISIZ = Number of data items within a record.

RMPUTR - Put a Record in the Freelist

This routine returns a record to the freelist. The freelist is restructured so that the most recently returned record is set to be the FIRST freelist record available. If the freelist flag is already set, an error is reported, and the calling routine is informed.

CALL RMPUTR (IRC, NUM), where

IFC = File code of affected file

NUM = Record number of record to be returned to the freelist

= -1 if record is already freelist flagged.

A.1 RAFMAN Error Codes

Error Number	Meaning
1	Attempt to RMPUTR record at or before accounting record.
2	Attempt to RMPUTR record beyond MAX record number.
3	Attempt to RMPUTR record which is in the freelist.
4	Attempt to RMGETR record not in the freelist.
5	Attempt to RMGETR record beyond MAX record number.
6	Attempt to access freelist record as a used record.
7	End-of-File before MAX record.
8	Attempt to access record at or before accounting record.
9	Attempt to access record beyond MAX record number.
10	Illegal degree in Node record.
11	Inconsistency between Node and Arc files.
12	Inconsistency within Node file.

L E O

CONVERSION BETWEEN LEGAL AND GEOGRAPHIC REFERENCE SYSTEMS IN KANSAS

By:

Charles G. Ross

Research Assistant
Technical Information Services
Kansas Geological Survey
August 1988

The LEO System

The existence of geologic and geographic data in differing reference systems (**geographic** coordinates or longitude-latitude, cartesian/**projected** coordinates or (x,y), and **legal** coordinates or township, range, and section) makes it necessary to provide conversions between these systems to fully utilize the data and to perform functions which are available only within a given system or are more efficient within that system.

In particular, much data (both digital and non-digital) exists in which locations are specified in the legal reference system whose spatial nature inhibits (if not prohibits) its use in mapping, graphics, and spatial analysis. This problem has not been overcome in the past because it has not been possible until recently to convert from legal to geographic coordinates with acceptable accuracy. LEO provides this function, resulting in geographic coordinates which can then be projected (using the **GIMMAP.PROJCT** system) into Cartesian coordinates suitable for the above functions.

The **LEO** conversion system (**LE**gal <=> g**EO**graphic), consists of a FORTRAN77 subroutine called LEOCVT and an associated spatial database, GIMMAP.LEOBASE. It functions to perform conversions between the geographic and legal reference systems for the state of Kansas. Given a legal location including township, range, section and subdivisions of the section, LEOCVT will produce the most accurate longitude and latitude, (the geographic location), available for the center of the selected sub-area.

Conversely, given a geographic location in the form of a longitude and latitude, LEOCVT will produce the legal description including the township, range, section and four successive levels of subdivision of the section. The smallest area specified will contain the original point given in longitude and latitude.

These conversions have been made possible by construction of the GIMMAP.LEOBASE database, essentially the collection of all township and section corners, spatially-ordered and represented as geographic locations. This information was originally extracted from the digitized (and edited) data contained in the Kansas Cartographic Database (KCD).

The KCD has been compiled by use of the **GIMMAP** (Geodata Interactive Management Map Analysis and Production) system in the Automated Cartography section at the Kansas Geological Survey from 1980 to the present, Nearly all the data in the KCD was digitized from the series of 7.5′ quadrangle maps produced by the U.S. Geological Survey.

The Legal Reference System

The legal (township-range-section) system is based on a set of surveyed grid corners and lines with an arbitrary origin at the intersection of the **base line** (at 40 degrees north latitude), and the **principal meridian** (the "6th", at 97 degrees, 23 minutes west longitude). The legal system of reference comprises a grid of (theoretically) equal-sized townships (six by six miles) whose locations are described as some number of rows south of the base line ("township 14 south"), and columns east or west of the principal meridian, ("range 4 east" or "range 22 west").

In Kansas, there are 35 rows of townships and 25 columns (range designations) east and 43 columns west of the principal meridian. Each township is divided into 36 sections, and each section is approximately one mile square. Sections are referred to by unique section numbers, 1-36. Within the township, section numbers are assigned in a snakelike (so-called "baustrophedonic") pattern. Section 1 is in the northeast corner of the township. Sections 2-6 proceed to the west, with section 6 in the northwest corner. Section 7 lies immediately below section 6, and sections 8-12 lie eastward from section 7. Numbering ends with section 36 in the southeastern corner of the township:

Section 6	5	4	3	2	Section 1
7	8	9	10	11	12
Section 18	17	16	Section 15	14	13
19	20	Section 21	22	23	Section 24
30	29	28	27	26	25
Section 31	32	33	34	35	Section 36

Specification of locations within this system apply only within the state of Kansas. They are composed of a township specification, a section specification, and various, optional forms of further specification (called subdivision) within the section. Township specification is by township and range (east or west) numbers, such as township 5 south and range 12 west (written as T5S and R12W). The section number (1-36) fixes the unique section within the township.

Specification of a legal location within the section may be performed in several ways. LEO supports four options of point specification within the selected section:

The first option is to make no further specification and use the center of the section (average of four corners).

The second option allows selection of one of the township corners by use of the appropriate letters for subdivision into quarter-areas (as shown below) in the first subdivision of the section. The CORNER = 1 option must be used, and the location of the selected township or section corner is returned.

The third option is to subdivide the section successively into smaller and smaller half parts for a maximum of four levels of subdivision. The subdivided areas may be 1/2, 1/4, 1/8 or 1/16 of the section. Each subdivision is specified by a letter (see below), and the selected point is the averaged center of the smallest specified sub-area. The specification of subdivisions is made by setting the subdivision values at each successive level using the following letters:

Specifying Half-Area Subdivisions

N	=>	North half of area
S	=>	South half
E	=>	Eastern half
W	=>	Western half

The fourth option for subdivision of sections is to divide by quarter-areas. As with half-areas, a maximum of four levels of subdividing is allowed. Thus, sub-areas of one-quarter, one-sixteenth, one-sixty-fourth and one-two hundred and fifty-sixth (1/4, 1/16, 1/64, 1/256) the original area may be generated. The averaged center of the smallest subdivision is the point used to perform the conversion. Selection of quarter-areas is made by either of two methods of letter assignment:

Specifying Quarter-Area Subdivisions

NE	or	A	=>	Northeast
NW	or	B	=>	Northwest
SW	or	C	=>	Southwest
SE	or	D	=>	Southeast

Another method for subdivision, not directly supported by the LEO system, is that of specifying location by distances or "footages" on the ground, as measured from section or township corners or lines. This method may be approximated with LEO by a technique described later.

Using the LEOCVT Routine

Legal descriptions may include up to 4 levels of quarter- or half-areas for subdivision of a section into an area as small as 1/256th of the original section. The location produced is that of the center of the smallest subdivided area specified (unless the CORNER = 1 option is used to select the exact corner of a section). Conversion from a geographic location to a legal one results in specification of all four levels of subdivision, any part of which may be used to define the area containing the original point.

LINK

To use the LEOCVT routine, the user must LINK the calling program to the LEOCVT object file:

```
:UTIL:GIMMAP.LEOCVT.OB
```

OPEN

Prior to using the LEOCVT routine for conversions, the user must open the database as follows:

```
      OPEN(LEOFC,FILE=':UTIL:GIMMAP.LEOBASE',ACCESS='DIRECT',
     +       FORM='UNFORMATTED',RECL=400,IOINTENT='INPUT')
```

where the unit number for the GIMMAP.LEOBASE database, LEOFC, is also passed as the unit number to the LEOCVT routine (see below).

CONVERSION

Conversion between reference systems is then performed by:

```
 CHARACTER EW*1,SUBD*2(4)
 INTEGER*2 CORNER,LEOFC,LORG,RANGE,SECT,SUCCESS,TSHIP
 REAL PLAT,PLON

       :


 (set values for TSHIP, RANGE, EW, SECT, CORNER, SUBD,
     LEOFC and LORG=1 for legal -> geographic)
OR
 (set PLON, PLAT, LEOFC and LORG=2 for geographic -> legal)

       :

 CALL LEOCVT(TSHIP,RANGE,EW,SECT,CORNER,SUBD,
+                                PLON,PLAT,LEOFC,SUCCESS,LORG)

       :

 (check the value of SUCCESS for successful conversion)
```

where

TSHIP = Township number (integer, 1-35)

RANGE = Range number (integer, 1-25E, 1-43W)

EW = Range Indicator, 'E' = East, 'W' = West

SECT = Section number (integer, 1-36)

CORNER = 0 for center of smallest subdivision
= 1 for section corner specified in SUBD(1)

SUBD(4) = Specification for section subdivisions

Each subdivision is applied to the area specified at the previous level, beginning at (1) as the largest to (4) as the smallest. If a subdivision is blank, subdivision stops. The specification of subdivision may be made as quarter or half- area designations.

For quarter-area designations:

'NE' or 'A ' => Northeast
'NW' or 'B ' => Northwest
'SW' or 'C ' => Southwest
'SE' or 'D ' => Southeast

For half-area designations:

'N ' => North
'S ' => South
'E ' => East
'W ' => West

Note: The smallest subdivision used is the last non-blank subdivision. All others must be blank:

SUBD(unused) = ' '

Note: When the CORNER = 1 option is used, the value of SUBD(1) is used to specify the section or township corner whose coordinates are desired.

PLON = Longitude in decimal degrees, positive or negative as input, positive on output

PLAT = Latitude in decimal degrees

LEOFC = Unit number of (OPENed) GIMMAP.LEOBASE

SUCCESS = 1 => **Successful Conversion**
= -1 => Township number out of range
= -2 => Range number out of range
= -3 => Illegal EW indicator value
= -4 => Section number out of range
= -5 => Illegal Subdivision #1
= -6 => Illegal Subdivision #2
= -7 => Illegal Subdivision #3
= -8 => Illegal Subdivision #4
= -9 => Illegal LORG indicator
= -10 => Invalid longitude for LEO/Kansas
= -11 => Invalid latitude for LEO/Kansas
= -12 => Insufficient information in LEOBASE
= -13 => Error in LEOBASE data
= -14 => Thrashing error at township level
= -15 => Thrashing error at section level

LORG = 1 => conversion from Legal to Geographic (township/range/section to long/lat)
= 2 => conversion from Geographic to Legal

Note on LEO System Errors

The LEOBASE database of section and township corners has been constructed from the Kansas Cartographic Database (KCD), a state-wide cartographic database for the state of Kansas. The features in the KCD were generally digitized from the USGS 7.5' quadrangle series of topographic maps covering the state at a scale of 1:24000. Errors certainly exist in the representation of features on these maps, and errors have occurred in creating the KCD from the maps. Therefor, some section/township corner locations may be in error beyond normal tolerances. All users of the LEO system are requested to report any such errors so that the database may be corrected for future use.

LEO has been tested minimally, and errors may yet be found. The value of SUCCESS should be checked after every call to the LEOCVT routine. SUCCESS values from -11 to -1 indicate user errors (in the specification of parameters). SUCCESS values below -11 indicate LEO system errors or lack of information.

Please report any unresolved errors (SUCCESS = 0) you may encounter. Such errors may be the result of problems in the database which could be fixed for future use. Also, report all errors in the range -12 to -15. When reporting errors, note the error number and if possible the exact conversion values which failed. Complete specification of the parameters passed to the LEOCVT routine and any values returned will be helpful. There are no guarantees for the accuracy of locations in the database or for the LEO conversion system, but in general both seem to be fairly good. Correction of GIMMAP.LEOBASE and LEO system errors will improve the system for the future use for everyone.

Exceptions to the Legal System in Kansas

One area of Kansas does not follow the structure for the legal system which is used in the rest of the state. This is the column of townships in Range 8 east, in which the western half of the western column of sections (6, 7, 18, 19, 30 and 31) are exceptionally wide due to the poorly planned meeting of two separate surveys. The problem was resolved at the county level, resulting in further subdivision into **lots** which are arranged and numbered in ways defined by each county.

The LEO system at present does not incorporate these county solutions, but instead assumes these townships are the same as all others. Any points lying in the westernmost sections of this range should first be converted to an appropriate quarter- or half-area location description before conversion by LEOCVT is performed. Furthermore, other areas around the Missouri river and elsewhere along the state boundary may fail in conversion due to a lack of complete data.

Approximation of Footages by Subdivision

Legal descriptions which use footages or distances from section or township corners or lines to describe locations within specified sections are not (**yet**) directly convertible in the LEO system. However, these locations may be converted approximately by using either of the following procedures.

First, convert the footages to fractions of the width/heigth of the section, assuming a constant size of 1 mile on a side (more accurate measures may be gained by projection of section corners obtained by LEO). Convert the fractions into appropriate subdivision notations and use LEO to convert the points.

Alternatively, the CORNER = 1 option may be used to obtain the locations of the four corners of the section. The fractions may then be applied directly to these values to produce the locations of the desired points.

MATERIALS REQUIRED FOR THE MAPREQUEST PROGRAM

INDEX

PAGE

FEATURE CODES FOR THE MAPREQUEST PROGRAM

******* ISOLATED POINTS: 1000 - 1999**

* 1100 Control Points

- 1110 Land Grid
  - 1111 Section Corners
  - 1112 Township Corners
- 1120 Control Stations
  - 1121 Horizontal
  - 1122 Vertical

* 1200 Towns, City Centers

Minor Code*	Meaning
1	Unincorporated
2	Incorporated
3	County Seat
4	State Capitol

*The minor code is the least significant digit.

1210		Population <		500
1220	500 <	Population <		1000
1230	1000 <	Population <		3000
1240	3000 <	Population <		10000
1250	10000 <	Population <		25000
1260	25000 <	Population <		50000
1270	50000 <	Population <		100000
1280	100000 <	Population		

* 1300 Wells,Springs

Minor Code	Meaning
1	Producing
2	Abandoned Dry
3	Abandoned Producing
4	Under Development

- 1310 Water Well
- 1320 Spring
- 1330 Oil Well
- 1340 Gas Well
- 1350 Oil and Gas Well
- 1360 Dry Hole

* 1400 Pits, Quarries, Mines

Minor Code	Meaning
0	Producing
1	Abandoned
2	Under Development

1410 Open Pit
1420 Mine
1430 Quarry
1440 Prospect
1450 Shaft or Tunnel Entrance

* 1500 Buildings

1510 Dwelling or Employment
1520 Church
1530 Cemetery
1540 Tower

1541 Radio
1542 Television
1543 Microwave
1544 Transmission Line
1545 Water
1546 Observation

1550 Power Substation
1560 School
1570 Compressor Station

***** **LINE ARCS: 2000-3999**

* 2100 Borders, Boundaries

2110 U.S. Land Survey Lines

2111 Township (East-West)
2112 Range (North-South)
2113 Section

2120 Cemetery
2130 Fence or Field Line
2140 Reservation
2150 Park

2151 National
2152 State
2153 County
2154 City

2160 Geo-Reference Lines

2161 Latitude
2162 Longitude

* 2200 Highways, Roads

Minor Code	Meaning
1	Interstate
2	Federal (U.S.)
3	State
4	County, City
5	Private
6	Interstate and Federal
7	Interstate, Federal, and State
8	Interstate and State
9	Federal and State

2210 Primary Highway, Hard Surface
2220 Secondary Highway, Hard Surface
2230 Light-Duty Road
2240 Unimproved Road
2250 Road Under Construction or Proposed
2260 Highway Ramps
2270 Dual Highway, Dividing Strip < 25 Feet
2280 Dual Highway, Dividing Strip > 25 Feet
2290 Spurs

* 2300 Railroads, Airports

2310 Single Track Railroad
2320 Multiple Track Railroad
2330 Railroad Bridge
2340 Railroad Tunnel
2350 Airport Boundary
2360 Airport Runway
2370 Landing Strip

* 2400 Lineal Hydrology

2410 Perennial Stream
2420 Intermittent Stream
2430 Aqueduct
2440 Aqueduct Tunnel
2450 Dam
2451 Dam with Road

2460 Dike or Levee
2461 Dike or Levee with Road

* 2500 Transmission Lines

2510 Electrical Power

2511 Power Line
2512 Substation

2520 Pipeline

2521 Gas
2522 Oil
2523 Water

2530 Telephone Line

* 2600 Intra-Formation Geology

2610 Visible Fault

2611 Transverse
2612 Thrust
2613 Normal

2620 Approximate Fault

2621 Transverse
2522 Thrust
2523 Normal

2630 Intrusives
2640 Folds

* 2700-2900 Unassigned

* 3000-3999 Elevation Contours

3ABC where ABC = elevation/10
A = Thousands
B = Hundreds
C = Tens

***** **ZONE BOUNDARY ARCS: 4000-4999**

* 4100 Areal Hydrology

4110 Perennial Stream
4120 Permanent Lake
4130 Intermittent Stream
4140 Intermittent Lake
4145 Land Subject to Controlled Inundation

4150 Reservoir
4160 Aquifer
4170 Swamp

* 4200 Inter-Formation Geology

4210 Actual Contact
4220 Approximate Contact
4230 Actual Faults

4231 Transverse
4232 Thrust
4233 Normal

4240 Approximate Faults

4241 Transverse
4242 Thrust
4243 Normal

4250 Folds

* 4300 Areal Boundaries

4310 State Boundary
4320 County Boundary
4330 Quadrangle Map edge
4340 City Boundary

4341		Population <		500
4342	500 <	Population <		1000
4343	1000 <	Population <		3000
4344	3000 <	Population <		10000
4345	10000 <	Population <		25000
4346	25000 <	Population <		50000
4347	50000 <	Population <		100000
4348	100000 <	Population		

4350 Groundwater Management District Boundary

Minor Code	Meaning
1-5	District Number

4360 County Seat - City Boundary

4361		Population <		500
4362	500 <	Population <		1000
4363	1000 <	Population <		3000
4364	3000 <	Population <		10000
4365	10000 <	Population <		25000
4366	25000 <	Population <		50000

4367 50000 < Population < 100000
4368 100000 < Population

4370 Hydrologic Basins

4373 Cataloging
4374 Accounting
4375 Subregional
4376 Regional

* 4400 Areal Resource Boundaries

4410 Oil or Gas Field
4420 Shallow Oil or Gas Field
4430 Oil or Gas Storage Area

***** **REDIGITIZED OR ADDED FEATURES (OPTIONAL)**

10000 + the normal feature code causes lines to be displayed as dashed for editing purposes. Codes are:

1XXXX

where XXXX is the normal feature code.

DATA SOURCES FOR THE MAPREQUEST PROGRAM

1. Already Digitized USGS 7.5', 1:24,000 maps (KDB).
2. TO BE Digitized USGS 7.5', 1:24,000 maps.
3. TO BE Digitized Hand-drawn Additions to 7.5' maps.
4. Existing GIMMAP form DDF files.
5. TO BE PROJECTED Lat/Long file, with NO Z values.
6. TO BE PROJECTED Lat/Long file, with Z values.
7. Other existing GIMMAP form Database.
8. TO BE DIGITIZED from other maps.
9. From other digital files in other forms.
0. Unknown.

SYMBOL TYPES AND SPECIFICATIONS FOR THE MAPREQUEST PROGRAM

A. POINT SYMBOLS

CODE	SYMBOL	
0	Unknown	All point symbols are centered at the location specified. The specification is for the size of the symbol in plotter units. If the size is negative, then it is used as a scale factor with the feature code.*
10	Plus sign (+)	
11	X mark (X)	
12	Asterisk (*)	
13	Square	
14	Circle	
15	Triangle	
16	Fiducial	
17	Hexagon	
21	Oil Well	
22	Gas Well	
23	Oil and Gas Well	
24	Dry Hole	
25	Miscellaneous Well	
31	Inverted Triangle	
41	Solid Circle	
42 - 99	Other	

* To calculate the symbol size when the size is specified as a negative number, let

SCALEFACTOR = absolutevalue (size specification)

Then, the symbol size, which will vary with the feature code of the arc is:

SYMBOLSIZE = (FEATURE CODE - BASE) * SCALEFACTOR

where BASE = user-selected feature code base (usually 1000).

B. LINE TYPES

CODE	LINE TYPE	SPEC1	SPEC2
0	Unknown	-	-
20	Solid Line (Default)	-	-
21	Dashed Line	Dash length	Gap Length
22	Double Line	Separation	-
23	Hashed Line	Hash Width	Hash Spacing
24	Double Dashed Line **	Dash Length	Gap Length
25	Long-Short Dashed	Long Dash	Short/Gap
26	Long-Short-Short	Long Dash	Short/Gap
27-29	Other	-	-

** When the double dashed line type is selected, the user must supply a third specification - the separation between lines.

XYNETICS PEN SIZE CODES FOR MAPREQUEST

A. LIQUID INK PENS

CODE	PEN SIZE	
1	0000	Smallest
2	000	
3	00	
4	0	
5	1	
6	2	
7	3	Largest

B. SCRIBE POINTS

CODE	POINT SIZE	
1	2	Smallest
2	3	
3	4	
4	5	
5	6	
6	7	
7	8	
8	9	Largest

Note: Pens for ballpoint come in three sizes (bold, medium, fine), but are not easily distinguishable. Therefore, no sizes are specified. Also, some of the scribe points may not be available, but the closest available size will be substituted.

MAP STATUS VALUES FOR THE MAPREQUEST PROGRAM

1. Planning
2. Major Digitizing
3. Database Construction
4. Interactive Graphical Edit
5. Zone Generation
6. Interactive Zone Edit
7. User Verification of Edit Completion
8. User Verification of Zone Colors
9. Interactive Symbology Placement
10. User Verification of Symbology
11. User Verification of Completion
12. Production (Plots, Scribes, Files,..)
13. Archival (Finished, but saved for future).
14. Release (Finished, and not saved).

KANSAS COUNTY NUMBERS FOR MAPREQUEST

1. ALLEN
2. ANDERSON
3. ATCHISON
4. BARBER
5. BARTON
6. BOURBON
7. BROWN
8. BUTLER
9. CHASE
10. CHAUTAUQUA
11. CHEROKEE
12. CHEYENNE
13. CLARK
14. CLAY
15. CLOUD
16. COFFEY
17. COMANCHE
18. COWLEY
19. CRAWFORD
20. DECATUR
21. DICKINSON
22. DONIPHAN
23. DOUGLAS
24. EDWARDS
25. ELK
26. ELLIS
27. ELLSWORTH
28. FINNEY
29. FORD
30. FRANKLIN
31. GEARY
32. GOVE
33. GRAHAM
34. GRANT
35. GRAY
36. GREELEY
37. GREENWOOD
38. HAMILTON
39. HARPER
40. HARVEY
41. HASKELL
42. HODGEMAN
43. JACKSON
44. JEFFERSON
45. JEWELL
46. JOHNSON
47. KEARNY
48. KINGMAN
49. KIOWA
50. LABETTE
51. LANE
52. LEAVENWORTH
53. LINCOLN
54. LINN
55. LOGAN
56. LYON
57. MARION
58. MARSHALL
59. MCPHERSON
60. MEADE
61. MIAMI
62. MITCHELL
63. MONTGOMERY
64. MORRIS
65. MORTON
66. NEMAHA
67. NEOSHO
68. NESS
69. NORTON
70. OSAGE
71. OSBORNE
72. OTTAWA
73. PAWNEE
74. PHILLIPS
75. POTTAWATOMIE
76. PRATT
77. RAWLINS
78. RENO
79. REPUBLIC
80. RICE
81. RILEY
82. ROOKS
83. RUSH
84. RUSSELL
85. SALINE
86. SCOTT
87. SEDGWICK
88. SEWARD
89. SHAWNEE
90. SHERIDAN
91. SHERMAN
92. SMITH
93. STAFFORD
94. STANTON
95. STEVENS
96. SUMNER
97. THOMAS
98. TREGO
99. WABAUNSEE
100. WALLACE
101. WASHINGTON
102. WICHITA
103. WILSON
104. WOODSON
105. WYANDOTTE

INFORMATION IN MAP PROJECT REQUEST DATABASE

*** **RECORD 1** ***

ITEM		Variable	Type/Size*
1	User's Name	UNAME	C30
2	User's section or organization	SECTION	C30
3	Original Request Date	RDATE(3)	I4
4	Last Modification Date	MDATE(3)	I4
5	Map Form (1=Final, 0=Not Final)	FORM	I2
6	Deadline Date	DLDATE(3)	I4
7	Preferred Completion Date	PDATE(3)	I4
8	Map Project Name	PROJCT	C30
9	Map Name (Can be same as 8)	MAPNAM	C30
10	Number of Products	NUMMAP	I2
11	Publication (1=Yes, 2=4-color)	PUBLIC	I2
12	Media (+1 paper,+2 mylar,+4 scribe, +8 plot files, +16 DDF files)	MEDIA	I2
13	Password for Request	PW	C20
14	Unused	4 Bytes	

* Types are: C=character, I=integer, R=real (floating-point)

*** **RECORD 2** ***

ITEM		Variable	Type/Size
1	Ink (1=ballpoint,2=liquid,3=both, 4=scribe,0=Unknown)	INK	I2
2	Preferred Meeting Time, Line 1	MEET1	C60
3	Preferred Meeting Time, Line 2	MEET2	C60
4	Map Projection (0=NO, 1=MP, 2=LC, 3=Other, 4=Unknown)	MAPROJ	I2
5	Map Scale Factor (-1. if unknown)	MSCALE	R4
6	South Standard Parallel (-1.=Unused)	SPARAL	R4
7	North Standard Parallel (-1.=Unused)	NPARAL	R4
8	Projection Area Minimum Latitude	PLATMN	R4
9	Projection Area Maximum Latitude	PLATMX	R4
10	Projection Area Minimum Longitude	PLONMN	R4
11	Projection Area Maximum Longitude	PLONMX	R4
12	Overlay With Other Computer Plots (0=NO, 1=SURFACEII, 2=OTHER, 3=?)	OVRLAY	I2
13	SURFACE EXTREMES values (if used)	SEXTR(4)	R4
14	SURFACE BXTREMES values (if used)	SBXEX(4)	R4
15	Unused	14 Bytes	

*** **RECORD 3** ***

ITEM		Variable	Type/Size
1	Type of Coverage: 1 = multi-state 2 = state 3 = Lat/Long rectangle 4 = Quadrangle Row/Column rectangle 5 = county list 6 = Non-Rectangular area 7 = Other	COVRGE	I2
2	Minimum Latitude	MINLAT	R4
3	Minimum Longitude	MINLON	R4
4	Maximum Latitude	MAXLAT	R4
5	Maximum Longitude (Items 2-5 are in decimal degrees, if used)	MAXLON	R4
6	Counties Completely Included Entry I = 1, if County I is completely included in the map coverage. (See County list). Entries for counties 1-55 are in this record. The rest are in record 4.	CCI(105)	I2

*** **RECORD 4** ***

ITEM		Variable	Type/Size
1	Counties Completely Included (Counties 56-105)	CCI(105)	I2
2	Minimum Quad Row (if used)	MINROW	I2
3	Minimum Quad Column (if used)	MINCOL	I2
4	Maximum Quad Row (if used)	MAXROW	I2
5	Maximum Quad Column (if used)	MAXCOL	I2
6	Description of Boundary, Line 1	BNDRY(4)	C60

*** **RECORD 5** ***

ITEM		Variable	Type/Size
1	Descripion of Boundary, Lines 2-4	BNDRY(4)	C60

*** **RECORDS 6-25** ***

Information for 1-20 features. To map between records and indices:

RECORDNUMBER = INDEX + 5

ITEM		Variable	Type/Size
0	Number of Features (only in record 6)	NFEAT	I2

ITEM		Variable	Type/Size
1	Feature Name (alphanumeric)	FEATUR(I)	C20
2	Feature Code	FCODE(I)	I2
3	Feature Source (see Source List)	FSOURC(I)	I2
4	Symbol/Line Type (see Symbol List)	SYMTYP(I)	I2
5	Spec. 1 (see Symbol Specs. Chart)	SPEC1(I)	R4
6	Spec. 2 (for line type only)	SPEC2(I)	R4
7	Spec. 3 (for some line types only)	SPEC3(I)	R4
8	Pen Size (see Pen Size Code List)	PENSIZ(I)	I2
9	Name of Plot File (path name)	PLTFIL(I)	C60
10	Feature Labelling 0 = None, 1 = Bold, 2 = Subtle, 3 = Medium, 4 = Unknown	FLABEL(I)	I2
11	Source Map name (source = Other)	SOURCN(I)	C60
12	Source Map Projection if Other	SOURCP(I)	I2
13	Source Map Scale (source = other)	SOURCS(I)	R4
14	Unused - record 6: 34 bytes, records 7-25: 36 bytes		

***** RECORD 26 *****

ITEM		Variable	Type/Size
1	Title for Map	MAPTTL	C60
2	Location of Map Title 0 = None, 1 = Top, 2 = Bot., 3 = Left, 4 = Right, 5 = ? (Unknown)	LOCTTL	I2
3	Map Scale Bar 0 = None, 1 = Miles, 2 = Km., 3 = ?	SCLBAR	I2
4	Scale in Numbers (1:scale) 0 = None, 1 = Yes, 2 = ?	SCLNUM	I2
5	Map Border 0 = None, 1 = Single, 2 = Double, 3 = ?	BORDER	I2
6	Border Margin (inches, -1, if unused)	MARGIN	R4
7	Lat/Long Grid 0 = None, 1 = Labeled grid, 2 = Labeled tick marks, 3 = Unknown, -1 = Unlabeled grid, -2 = Unlabeled ticks	LATLON	I2
8	Interval for Lat/Long Grid in degrees	INTRVL	R4
9	Number of lines of Other Text (0-5)	OTEXTL	I2
10	Other Text, Lines 1 and 2	OTEXT(5)	C60
11	Unused	None	

***** RECORD 27 *****

ITEM		Variable	Type/Size
1	Other Text, Lines 3-5	OTEXT(5)	C60
2	Unused	20 Bytes	

*** **RECORD 28** ***

ITEM		Variable	Type/Size
1	User Comments	COMENT(3)	C60
2	Unused	20 Bytes	

*** **RECORD 29** ***

ITEM		Variable	Type/Size
1	Output offset for X in inches	XOFF	R4
2	Output offset for Y in inches	YOFF	R4
3	MAPGEN file name for plot production	MGPFIL	C60
4	(Secondary) Base name for titling	TTLBAS	C10
5	MAPGEN secondary (titling) base	MGTFIL	C60
6	X coordinates of Registraion marks	XREG(6)	R4
7	Y coords. for same (-1. if unused)	YREG(6)	R4
8	Registration Symbol (see Chart, 16 = Fiducial)	REGSYM	I2
9	Size of Registraion Symbol in inches	REGSIZ	R4
10	Map Status (see Map Status List)	MPSTV	I2
11	Initials of operator giving Map Status	MPSTI	C3
12	Unused	3 Bytes	

*** **RECORD 30** ***

ITEM		Variable	Type/Size
1	Map (Outer) Border X Minimum	BRDXMN	R4
2	" " " Y "	BRDYMN	R4
3	" " " X Maximum	BRDXMX	R4
4	" " " Y "	BRDYMX	R4
	(Items 1-4 are set to -999 if unused)		
5	Separation for double border (-1. if unused)	BRDSEP	R4
6	Special (plot,etc.) Instructions	INSTR(3)	C60
7	Unused	None	

*** **RECORDS 31-40** ***

Titling information for up to 20 title groups. Two groups in each record. To map between the records and the groups:

```
RECORD = (GROUP-1)/2 +31
```

and

```
GROUP = (RECORD-31)*2+1
```

ITEM		Variable	Type/Size
1	Feature type for titles in group 1	TTLTYP(I)	C20
2	Feature code for titles in group 1	TTLCOD(I)	I2
3	Height for group I titles (inches)	TTLHGT(I)	R4
4	Hershey Alphabet for group 1 titles	TTLSET(I)	I2
5	Plot pathname for group 1 titles	TTLFIL(I)	C60
6	Same as 1 above but for group I+1	TTLTYP(I+1)	
7	Same as 2 above but for group I+1	TTLCOD(I+1)	
8	Same as 3 above but for group I+1	TTLHGT(I+1)	
9	Same as 4 above but for group I+1	TTLSET(I+1)	
10	Same as 5 above but for group I+1	TTLFIL(I+1)	
11	Unused	14 Bytes	