UNITED STATES DEPARTMENT OF THE INTERIOR

GEOLOGICAL SURVEY

GLORIA Atlas Preparation: A basic application of the MAPGEN mapping system as a publishing aid

by

John C. Hampson,Jr.[1] and Evelyn L. Wright[2]

Open-File Report 88-287

1988

---

[1]CLM/SYSTEMS, Inc., 3654 Gandy Blvd., Tampa, FL 33611

[2]USGS, Woods Hole, MA 02543

CONTENTS

1. INTRODUCTION

The following material describes the preparation of computer graphics used in the GLORIA Atlas for the Gulf of Mexico and Eastern Caribbean (EEZ-Scan Scientific Staff, 1987). It is intended as an example of straightforward computer graphics providing a publishable final product. As support personnel dwindle and publishing pressures increase, these techniques can be increasingly useful.

The work was done on Pacific Microcomputer PM68D microcomputers, running a UNIX System V operating system. The graphics software is from the MAPGEN package (Evenden and Botbol, 1985). UNIX utilities and applications programs written in Woods Hole were used to process data for input to MAPGEN.

High-quality final plots were made on the Kongsberg flatbed plotter belonging to National Mapping Division in Reston. This plotter is critical to the process because it is difficult, and sometimes impossible, to get publishable quality from pen and ink plotters. The Kongsberg makes plots on positive or negative film with a photo head. Thanks to the flexibility of the MAPGEN plotting system, it is possible to export custom plot tapes to the Kongsberg with a 9-track magnetic tape.

2. BACKGROUND

In order to see how computer-generated products fit into a final product, it is useful to review typical map-publishing procedures. The final map products are customarily printed from a group of photographically etched inking plates, one plate for each color to appear on the final. Each inking plate is in turn derived from a color-separate negative containing only those portions of the sheet that are to be printed in a designated color. For example, the blue color-separate negative is a black-and-white negative photographed from a punch-registered stack comprising all the blue elements on the page. The "originals" in this stack do not have to be blue, and are better off being a color that photographs well, such as black or red. These "originals" can be computer-generated work, stick-on type, hand-scribing, photographically screened copy, or anything else, as long as it is all carefully registered with the rest of the stack. This registration is achieved by carefully aligning the copy on punch-registered carrier sheets that all share the same registration system. In National Mapping Division, Reston, a standard format is used throughout the process. It is called the Alldis punch system (after the machine used to create the inking plates).

MAPGEN is well suited for generating the "separates" that are stacked for multicolored finals. MAPGEN converts position data such as tracklines, core sites, bathymetric contours, or geologic boundaries

into scaled overlays, with or without labels, at any specified scale and map projection. Until final plotting, each of these overlays is held separately in a compact metagraphic overlay file. Metagraphic means plotter-independent graphic instructions that may be translated onto any plotting device. At final plotting, any or all of the individual overlay files may be combined with peripheral overlays (collar work) containing a grid or corner tic marks, scales, legends, and titles, or they may be plotted with no peripheral overlays at all.

3. MAP CREATION

The types of maps completed with MAPGEN for the Gulf of Mexico and Caribbean atlases are shown in Figure 1. Each map comprises several overlays containing a mix of computer drafting, stick-up type, and hand-drafting.

Map generation begins by defining an appropriate scale, projection, and map bounds in a control file, and running that control file through the 'mapdef' program to create a map definition file. For this atlas, all MAPGEN control files were named with the suffix ".c" and all map definition files were named "map.def" and kept in a separate directory for each map area.

The map definition file is used with the programs 'grid', 'legend', 'lines', 'points', and 'coast' to create overlays of latitude-longitude coordinates, scales and titles, tracklines and bathymetric contours, core sites, and coastlines, respectively. By splitting the input to these programs, the map information can be contained in as many separate overlay files as desired. Each of these metagraphic overlay files was named with a ".ov" suffix.

Data input files to MAPGEN routines were preprocessed using UNIX utilities and applications software. A great deal of editing was done to the bathymetric contour data to smooth boundary faults acquired from merging separate bathymetric sources and to correct minor errors introduced in digitizing. The trackline data were processed to remove errors, and special input files were created to permit marking lines every hour with arrows that show trackline direction. To permit ease of editing, the trackline labels were kept in a separate file and plotted as a separate overlay.

As mentioned, the computer overlay files prepared with MAPGEN can be plotted on any display device. This is accomplished using the 'plotter' command. Drafting can be assigned to any pen number on the plotter at the plotter stage of map preparation, as well as earlier in the process during creation of the metagraphic overlay files. On the Kongsberg plotter equipped with a photo head, these pen assignments provide a range of line weights from .002 to .027 inches (See Appendix). Most of

the pen specifications in the following examples create overlays with pen 0. Pen assignments for the Kongsberg plotter are made (or changed) in the final plot using the pen mapping facility in 'plotter'. This is accomplished with the command line:

```
plotter -d kong -o dev/(tape drive) -p0:(new pen number) overlay1
 -p0:(another new pen number) overlay2 -p0:etc.
```

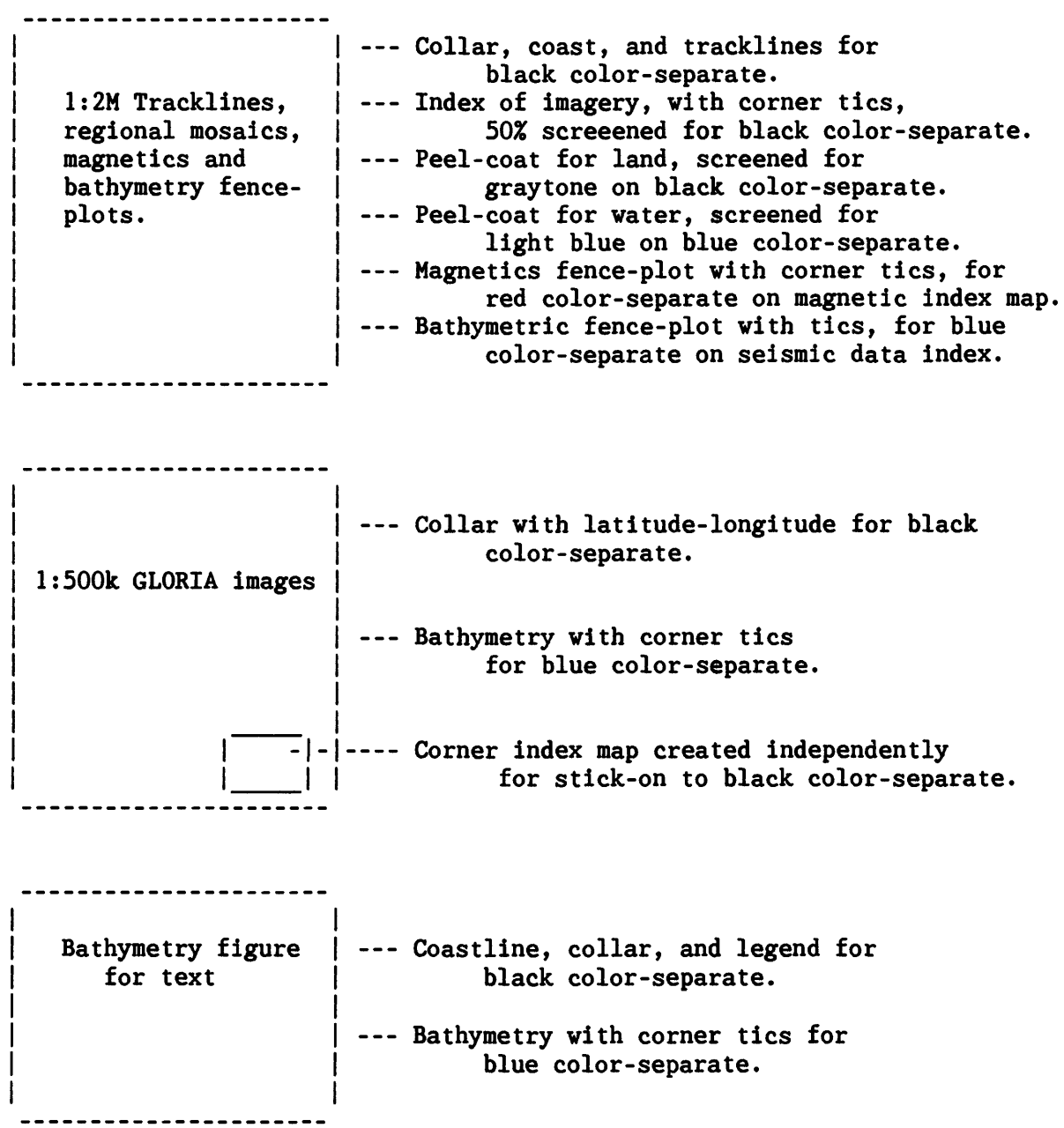
For example:

```
plotter -d kong -o /dev/nrmt1 -p0:4 grid*.ov -p0:7 mapdex.ov mapnum.ov
```

will create a tape for a Kongsberg plotter, with grid*.ov plotted with pen 4, while mapdex.ov and mapnum.ov are plotted with pen 7. In the control files that follow, the pen numbers and nominal line size assigned in the final plot are noted next to the pen weight commands.

The focus of the following sections is on the control files, which can be copied and modified for many mapping applications. The user is referred to Evenden and Botbol (1985) for a complete description of MAPGEN and the programs that system comprises.

Figure 1. Computer-generated overlays for GLORIA Atlas. Atlas includes twelve 1:2,000,000 maps, forty 1:500,000 maps of GLORIA images, and a bathymetry figure.



4. 1:2,000,000 TRACKLINE MAPS AND FENCE PLOTS

The 1:2,000,000 trackline maps served as reference maps at the beginning of each section of the atlas. Those map definition files, with grid and legend overlays, served as the bases for all twelve of these maps, which were printed with various combinations of data overlays. The procedure used to create a map definition file and each of the overlays in Figure 1 is described here.

4.1 1:2,000,000 Map definition file.

All the map definition files for the atlas were designed to approximately center the map on a 24"x30" sheet of film, with an additional 10 cm at the bottom and left margins to allow the film to be positioned 10 cm from the edge of the Kongsberg plotter.

A sample control file is shown here for the eastern half of the Gulf of Mexico.

```
::::::::::::::
map.c
::::::::::::::
90dw 82dw 23dn 31dn 86dw    #bounds and central meridian
 +proj=aea +lon0=86dw       #Albers equal area proj, central meridian.
2000000                     #scale = 1:2,000,000
0                           #no rotation of axes
19.5 25.5                   #left margin 19.5 cm, bottom margin 25.5 cm
9.5 15.5                    #right margin 9.5 cm, top margin 15.5 cm
```

In the system used for the atlas, each map area is given a separate UNIX directory; within each directory, 'mapdef' input (control) files are always named "map.c". Thus a script file can be called up each time a new map is to be created:

```
::::::::::::::
mkmap.cs
::::::::::::::
#
mapdef -mcvs map.def < map.c >> ../mapstats
```

and run from within the directory for the new map. With the above script, the map statistics and creation date are appended to a file called "mapstats". Mapstats is in the directory above the individual map directories and becomes a record of all maps created.

4.2 1:2,000,000 Collar.

The collar consists of a labeled latitude-longitude grid, created with the 'grid' program, as well as bar scales and a numeric map scale created with 'legend'. The page headings and projection information, which were stick-on type in this atlas, could be approximated using MAPGEN fonts. It should be noted, however, that the page headings can most easily be located in the same place on each page by putting them on an independent, reusable film overlay, sized to fit the largest map. This sheet can be composited with each black color-separate.

The first set of overlay files for the collar are the latitude-longitude grid. The control files for input to the 'grid' program were as follows:

```
::::::::::::::
grid1.c (latitude-longitude graticules on even degrees, labelled.)
::::::::::::::
-mi 2 -pi 2     #-meridian interval = 2 deg. -parallel intvl = 2 deg.
-s 0.25 -d 0.3  # -size of annotation = .25 cm -label offset = 0.3cm
-mu 0.3 -pu 0.3 # -merid. as 0.3 cm tics -parallels as 0.3 cm tics
-c 0 -b 0       # -char. assigned pen 0  -grid lines assigned pen 0
# # final pen for lines and labels = Kongsberg pen 4 (.006 in.)
# # if different line and char. weights desired:  -c 0 -b 1 can be
# # given in the control file, and "-p0:n -p1:m" set in the final plot.
```

```
::::::::::::::
grid2.c (label corners and draw tics along odd
parallel at top of map)
::::::::::::::
-mi 82 -pi 31    # meridian at 82d, parallel at 31d
-mu .01 -pu .01  # don't draw meridian or parallel line.
-a lr            # only label left and right margins (grid1.c labels
# #                   top and bottom margins with even meridians.)
-mj 41           # 82/41 degrees = minor interval (unlabeled merid.
# #                  tics along top parallel)
-mv 0.3 -pv 0.3  # tic size for minor interval = 0.3 cm
-s 0.25 -d 0.3
-c 0 -b 0        # map to pen 4 (.006 in) in final plot
```

```
::::::::::::::
grid3.c (label corners and draw tics along odd
parallel at bottom of map)
::::::::::::::
-mi 90 -pi 23      # left and bottom of longitude-latitude range.
-mu .01 -pu .01    # don't draw meridian or parallel line.
-a lr              # only label left and right margins
-mj 45             # 90/45 degrees = minor interval
-mv 0.3 -pv 0.3    # tic size for minor interval = 0.3 cm
-s 0.25 -d 0.3
-c 0 -b 0          # map to pen 4 (.006 in) in final plot



::::::::::::::
grid4.c (draw top and right neat lines)
::::::::::::::
-mi 82 -pi 31
-b 0               # try mapping to pen 6 (.010 in)



::::::::::::::
grid5.c (draw bottom and left neat lines)
::::::::::::::
-mi 90 -pi 23
-b 0               # try mapping to pen 6 (.010 in)
```

Each control file must be input separately to create a separate metagraphic overlay file. A script file such as the following can be used for this repetitive process:

```
::::::::::
mkgrid.cs
::::::::::
#
grid -mo map.def grid1.ov grid1.c
grid -mo map.def grid2.ov grid2.c
grid -mo map.def grid3.ov grid3.c
grid -mo map.def grid4.ov grid4.c
grid -mo map.def grid5.ov grid5.c
```

When plotting, the grid overlays can be referred to collectively in the 'plotter' command as grid*.ov.

The legend control file to create a bar scale and map scale looks like:

```
::::::::::::::
legend.c
::::::::::::::
-s .23        # char size .23 cm
-f -          # default system font
-p 0          # pen 0; use Kongsberg 5 (.007 in) in final
-w d          # measure positions from data window (excludes margins)
-x |-5.6      # position the following information 5.6cm left of center
-y -3.5       # position 3.5cm below bottom of data window (in margin)
-b 50,km,5,5 # plot scale bar with 5 50-km units right of 0, and 1 50-km
              # # unit divided into 5 parts to the left of 0
-y -2.5       # next scale bar 2.5cm below data window, x is unchanged
-b 50,kn,3,5 # 3 50-naut. mi. units rght of 0, 1 50-nm unit/5 to lft of 0
-j c          # center justify the following
-x |          # position at center of data window in x-direction
-y -1.5       # position 1.5 cm below bottom of data window (in margin)
-t            # print text
SCALE 1:2 000 000
.             # no more text to enter
```

The command line for legend is:

```
::::::::::::::
mkleg.cs
::::::::::::::
# Usage: csh mkleg.cs  (run in directory with map.def and legend.c
#          files; creates overlay legend.ov)
legend -mo map.def legend.ov legend.c
# final plot of legend.ov is run with Kongsberg pen 4 (.006 in)
```

The above script takes a map definition file "map.def" and a control file "legend.c" and creates a metagraphic overlay file, here called "legend.ov".

4.3 1:2,000,000 Coast and State Boundaries.

4.3.1 Drafted coastlines and boundaries. The coastline as well as state and international boundaries were created using the MAPGEN program 'coast'. A sample script file follows:

```
::::::::::::::
mkcoast.cs
::::::::::::::
# Usage:  csh mkcoast.cs     (run in directory with map.def file)
# file to create coastline overlay, international bounds, and state bounds.
coast -mo map.def coast.ov -c '-m 0' /usr/coast/na/cil /usr/coast/sa/cil
coast -mo map.def intbound.ov -c '-m 0 -d .1,43690' /usr/coast/sa/bdy
coast -mo map.def statebound.ov -c '-m 0 -d .1,0x6666' /usr/coast/na/pby
#  -m 0 is pen #, .1 is dash size in cm, 43690 and 0x6666 are decimal
#   and hexadecimal representations of binary number dash masks (1=dash
#   0=no dash).  Coast files are mapped to Kongsberg pen 5 (~.007 in)
```

4.3.2 Peel-coat of coastline for blue water. In addition to the above coast files (coast.ov, intbound.ov, and statebound.ov), each map had a subdirectory called blucoast, containing coast.ov and a special grid overlay with neatlines only. These two files were plotted with no pen mapping in order to drive a cutting head on the Kongsberg plotter across a 24"x30" sheet of peel-coat. This created a coastline mask overlay, permitting the blue coloring of the water in the trackline map.

The grid control files to place neat lines only are:

```
::::::::::::::
grid1.c (draw lines along 90d meridian and 23d parallel)
::::::::::::::
-mi 90 -pi 23
-b 0

::::::::::::::
grid2.c (draw lines along 82d meridian and 31d parallel)
::::::::::::::
-mi 82 -pi 31
-b 0
```

4.4 1:2,000,000 Tracklines.

The tracklines consist of three sets of overlay files: lines, arrows, and labels. The creation of these files required the MAPGEN routines 'lines' and 'points'. Applications software and UNIX utilities were combined to process the raw data for input to MAPGEN routines.

4.4.1 Basic lines. The basic tracklines can be created using 'lines' on the unedited navigation data, or, for aesthetic purposes, on an edited version, with "holidays" (unexpected deviations due to storms,

reruns, collision avoidance, etc.) removed from the trackline. Note that the more holidays removed, the more editing and placing of time labels to be done at each break.

A sample 'lines' command for raw data in NGSDC merge-merge format is:

```
::::::::::::::
mktrack.cs
::::::::::::::
# Usage: csh mktrack.cs input(NGSDC data) output(MAPGEN overlay)
lines map.def -o $2 -c '-p 0 -d .34.8,.26.7' $1
# final tracks plotted with Kongsberg pen 4 (.006 in).
```

where the "-d .34.8,.26.7" means that longitude starts after column 34 and the longitude field is 8 characters, latitude starts after column 26 and the latitude field is 7 characters. The -p 0 is for pen 0; the final tracklines were plotted with pen 5 (.007 in).

4.4.2 Arrows every hour. The data were reformatted using 'mrg3rfmt' (Paskevich, 1986) and fed into a special script called arrowscript (p. 11). Arrowscript calculates the direction of the track and intersperses MAPGEN commands to change the plotting angle of the arrows so they point in the direction of the track. The input file must contain all the data to permit accurate track-direction calculations. The output is piped through the UNIX stream editor so that the final file contains only the change-plotting-angle commands and the position at every hour along the track. This output file can then be run through the MAPGEN routine 'points' to create a final overlay with all the arrows pointing along the direction of the trackline.

The arrowscript routine calls several undocumented programs, listed in the Appendix, that are not standard to UNIX or MAPGEN. These programs are currently available on the Branch of Atantic Marine Geology computers in Woods Hole. A simpler script that does not call any special programs is also shown in the Appendix. It can be used with 'points' to place tic marks every hour instead of arrows.

A script to call arrowscript, create the 'points' input file, and create the 'points' overlay file is:

```
::::::::::::::
mkarrows.cs
::::::::::::::
# Usage: csh mkarrows.cs input output.c output.ov
#         input is NGSDC formatted data, output.c is the control
#         and data file, output.ov is the points output overlay.
#
# Script to take NGSDC format GLORIA data and place arrows every hour:
mrg3rfmt < $1 | arrowscript 66w 500 0 angles_out | sed -f sedlh.c > $2
points map.def -o $3 -c '-p 0 -d .14.11,.3.10 -sf -sr -sc 62 -ss .16 -sj c' $2
#  Final plot of overlay file mapped to Kongsberg pen 4 (.006 in)
```

The points command above includes a list of graphic options as follows:

```
-d .14.11,.3.10 # longitude starts after column 14, 11 char's in field,
     latitude starts after col. 3, 10 char's in field;
-p 0 # select pen number (for final use Kongsberg 4 [.006 in]);
-sf -sr #select font;  -sc 62 # define symbol(>);
-ss .16 # symbol size= 0.16cm;  -sj c # center symbol on point.
```

Mkarrows.cs above shows specific values following the 'arrowscript' command. These are variables explained in the body of arrowscript (shown below). The stream editor takes the output from arrowscript and selects all command lines plus data values every hour. The control file for the stream editor (sed) is:

```
::::::::::::
sedlh.c
::::::::::::
/#/ p
/ 198. .. .. ..  0/!d
```

Sedlh.c resides in the directory where mkarrows.cs is executed. The text of arrowscript follows. Tab characters have been replaced with the explicit "tab". In order to run, the script requires that a true tab be substituted for "tab". Arrowscript, and the programs it calls, are available through E. Wright.

```
# ARROWSCRIPT
# Prepares navigation-related data for putting arrows along the
#      navigation track lines.
#
# Usage:
#   arrowscript c_merid max_track_var angle_constant angles_out <input >output
#
#      where c_merid is the central meridian of the plot area,
#            max_track_var is the maximum allowable variation in the
#                          navigation track line in meters for purposes of
```

```
#                       identifying turning points in the track line
#                       (500 works pretty well),
#           angle_constant is the counterclockwise angle from horizontal in
#                       degrees to be added to the computed angle of the
#                       navigation track lines
#           angles_out is an output file containing the turning points of
#                       the track line and the perpendicular angles to
#                       the track-line segments,
#           input is an input data file of records containing latitude and
#                       longitude as the first two fields and any number of
#                       additional fields with spaces separating all fields,
#           output is the output data file of input data records with
#                      "# -b" and "# -sr track_angle" records inserted
#                      at each track-line turning point.
#
# Limit the number of input data records to 4000
head -4000 >tempin$$
#
# Compute sinusoidal projection & change field delimiter from space to comma
sed -e 's/   //' -e 's/  */,/g' <tempin$$ | proj -d',' -r2,1 -s +proj=sinu \
+lon_0=$1 | sed -e 's/"tab"/,/' >tempp$$
#
# Find track line turning points
sed -e 's/,/"tab"/' < tempp$$ | red1 $2 | sed -e 's/"tab"/,/g' | corner \
tempp$$ >tempr$$
#
# Fit least squares straight lines to the track segments, compute angles from
#      the horizontal (counterclockwise) of the line segments.
#
segfit tempp$$ <tempr$$ | segang $3 tempp$$ $4
#
# Merge  "# -b" and "# -sr track_angle" records with original data
cat tempin$$ | angmerge2 y $4
#
# Remove temporary files
/bin/rm -f tempin$$
/bin/rm -f tempp$$
/bin/rm -f tempr$$
```

4.4.3 Labels every 12 hours. The tracklines were labelled every 12 hours with day of year and Greenwich Mean Time. The day of year format was chosen to remain consistent with the magnetics, bathymetry, and seismic profile annotations. Day of year is also easier to fit on a trackline. The script used to process NGSDC GLORIA data is shown below:

```
::::::::::::::
p12hr.s
::::::::::::::

# Note: maintain blank line above (signals csh that this is a shell script).
# Usage:  sh p12hr.s  input  output
cat charspec.c > $2
#
# awk script to create 12-hr, year,mo,day file:  tmp0
# Selects even 12 hours, and reformats NGSDC GLORIA data.
# Sample line of input follows; field separator is one or more spaces:
#FARN   1/85   850809 0000 23.8317 -82.1700          0150201503040
#If input has no space between fields one and two, change '$3' through '$6' to
# '$2' through '$5' in the following 6 lines.
awk < $1 '
$4 ~ /0000/ || $4 ~ /1200/ {
       yr = substr($3,1,2)
       mo = substr($3,3,2)
       dy = substr($3,5,2)
       printf "%f\t%f\t%s,%s,%s\t%s\n",$5, $6, yr, mo, dy, $4
       }
' > tmp0
#
# Replace yr,mo,dy field in tmp0 with day of year, write to output file.
cut -f3 < tmp0 | doy | paste tmp0 - | awk -F"tab" -e '{print $1FS$2FS$5FS$4}' >>$2
#
# Remove temporary file
/bin/rm tmp0
```

The file tmp0 contains the 12 hour annotations with year, month and day. To save tmp0, place a #-sign in front of last line (#/bin/rm tmp0). The script file processes tmp0 and replaces yr, mo, day with day-of-year in the output file. This is accomplished with the Fortran routine 'doy' shown in the Appendix.

The cat program on line 4 of p12hr.s inserts a separate file of character-plotting control-statements in the output file. These character attributes for the trackline map are as follows:

```
::::::::::::
charspec.c
::::::::::::
# -c        # select character posting
# # define longitude/latitude (x/y) fields
# -d 2,1
#  # define location of character string
# -f 3,4
# -cf -     # select system default for character font
# -cr 0     #  horizontal character plotting (rotation = zero)
# -cj 1     # left justified
# -cs .15   # character size
# -cy -.16  #offset in y-direction
# -cx 0     #offset in x direction
# -cl 12    # interline leading=12/8 times ch. size
# #           (distance from bottom to bottom of 2 lines)
# -p 0      # pen 0; use pen 5 (.007 in) for final plot.
```

The output of p12hr.s is a combined control and data file, ready to run through 'points':

```
points -mo map.def output.ov p12hr.control
```

This control/data file can have statements inserted among the data to move over-writing labels. The -cj, -cy and -cx options shown in charspec.c can be changed by inserting a new command before the label to be moved. The new option remains in force for all subsequent data lines until a new control statement is inserted. [Note that this only works for 'points'. It is a peculiarity of 'lines' that embedding a control statement in the data affects the one previous as well as subsequent data lines.]

For example:

```
-cx 0     #set character x-offset to 0 cm from latitude-longitude.
-cy .16   #set y-offset to .16 cm up.
23.5432dn  84.9876dw  230  0000
23.5332dn  84.9332dw  230  1200
-cx -.10  # offset all following labels left .1 cm.
-cy .20   # offset all following labels up .2 cm.
23.5222dn  84.9000dw  230  0000
23.5232dn  84.9100dw  230  1200
-cy .16   # change y-offset back to .16, (leave x at -.1 cm).
23.5252dn  84.9200dw  230  0000
23.5272dn  84.9300dw  230  1200
```

4.5 1:2,000,000 Index of 2-degree sheets (1:500,000 images)

The construction of the overlay files for the index was straightforward. Simply use map corner positions to plot line segments. The size and positioning of sheet numbers is shown in the second control file below.

```
::::::::::::::
mapdex.c
::::::::::::::
# -p 0        #plot final with Kongsberg pen 6 (.010 in)
29.5     -90.0
29.5     -88.0
29.5     -86.0
# -b         # break line here
28.0     -96.0
28.0     -94.0
28.0     -92.0
28.0     -90.0
28.0     -88.0
28.0     -86.0
28.0     -84.5
# -b         # break line here
29.5     -90.0
et cetera
```

The mapdex.c file is run through 'lines'.

```
::::::::::::::
mapnum.c
::::::::::::::
# -d 2,1   #longitude in field 2, latitude in field 1 (tab
# #          is default field separator).
# -f 3     #read data from field 3
# -p 0     #pen number, plot final w/ Kongsberg pen 6 (.010 cm)
# -cr 0    #char. rotation = 0 (horizontal)
# -cf -sr  #use font -sr
# -c       #enable character plotting
# -cj 1    #left justify
#-cx -1.   #plot char's 1 cm left of latitude-longitude
#-cy -.9   #plot char's 1 cm below latitude-longitude
# -cs .6   #make char's 0.6 cm tall
28      -94     9
# -cx -1.5          #plot 10,12,13,14,16,15 1.5 cm to left
28      -92     10
28      -88     12
28      -86     13
28      -84.5   14
29.5    -88     16
29.5    -86     15
# -cx -1.         #plot 8,7,5,4,3,2,1 1 cm to left
# -cy .9          # "          "        0.9 cm above latitude
24      -94     8
24      -92     7
25      -88     5
24      -86     4
24      -84     3
23      -84     2
23      -82     1
```

The mapnum.c file is run through 'points'.

The two overlays (mapdex.ov and mapnum.ov) were printed on a separate sheet with unlabeled corner tics. This sheet was then screened 50% before compositing with the trackline map.

4.6 1:2,000,000 Fence diagrams

The fence diagrams at the beginning of both the seismic section and the bathymetric and magnetics section comprised a separate 'fence' overlay printed with unlabeled corner tics. This separate was mounted on a punch registered carrier to be composited as a color separate with the trackline map.

Most of the work involved with creating this graphic lies in editing the data files to instruct the pen to pick up around corners where the fence

over-wrote the map bounds or too many other fences. This, of course, requires several iterations of editing and plotting. The basic command line for 'fence' is:

```
fence def_file -o overlay_file control_file data_file
```

This is the standard MAPGEN input sequence. The control files should look like:

```
::::::::::::::
batfen.c
::::::::::::::
#-p 0      # pen 0, final plot Kongsberg pen 5 (.007 in).
#-d .14.8,.3.7 #longitude starts after column 14 of data file,
# # field is 8 characters; latitude column 3, 7 characters.
#-f .47.4      # bathymetry data starts after column 47, 4 char.
#-l            # start line drafting of selected data field.
#-v 0          # do not draw fence posts
#-S 0,4000,0,-1 # data range 0-4000(m), equivalent fence
# # scale = 0 to -1 cm.(-1 cm is one cm below geograph. coord.)

::::::::::::::
magfen.c
::::::::::::::
#-p 0         # pen 0; final plot Kongsberg pen 5 (.007 in).
#-d .14.8,.3.7 #longitude after col.14, 8 char.; latitude col 3, 7 char.
#-f .54.4      # mag. data after col 54. 4-char. field.
#-l            # start line drafting for current data field.
#-v 0          # no fence posts
#-S -600,600,-1.2,1.2 # range -600 to +600 gamma; plot in
# #                     -1.2 to 1.2 cm range.
```

5. OVERLAYS FOR 1:500,000 SHEETS OF GLORIA IMAGES

The map definition file for each 1:500,000 map is in a directory that is named for the map. The example here is sheet 1 in the Gulf of Mexico. The directory "gom01" contains the following control files and sub-directories:

map.c: the control file for input to mapdef
gridl.c - gridn.c: the grid control files to make the collar
bat: a subdirectory with bathymetric contour overlay files and its own set of grid files for corner tics

5.1 Map definition file

The map-definition control-file for gom01 looks like:

```
::::::::::
map.c
::::::::::
84w 82w 23n 25n 83w
 +proj=aea +lon_0=83w
500000
0
19.5 26.
9.5 16.
```

The borders in the last two lines are designed to approximately center the largest (furthest south) of the 1:500,000 collars on a 24" x 30" piece of film. An additional 10 cm is added to the left and bottom border so the Kongsberg can start at 0,0 but the film can be taped down 10 cm up and in from 0,0 (which is right at the edge of the plotter).

5.2 Map collar

The computer graphics in the map collar consisted of the latitude-longitude grid and the inset map in the corner. The inset map shows the regional index of 1:500,000 GLORIA images. It was created separately and stuck on the final map, with zip-tone shading the appropriate block.

The grid lines were all created with the 'grid' program. This provided a more reliable neatline around the map bounds than the alternate technique using 'legend'. In addition, this technique ensures that the map corners are labelled.

5.2.1 1:500,000 latitude-longitude collar The following grid control files were run through 'grid' with the script file:

```
::::::::::::::
grid.cs
::::::::::::::
# script to make grid overlay
# usage grid.cs n  -where n is the digit in the control file name,
# #                    e.g., 2 for grid2.c
grid map.def -o grid$1.ov grid$1.c
```

The grid control files for gom01 are shown below; other examples can be found in section 4.2.

```
::::::::::
grid1.c
::::::::::
-pi 23 -mi 82  #parallel at 23 d, meridian every 82 deg.
-pu .01 -mu .01 # don't draw lines for major parallels.
-s 0.25        #char size in cm
-f -sr         #font "-sr"
-d 0.3         #offset distance for char. annotation in cm
-pj 46 -mj 164  #minor parallel interval = major/46 (23/46)=.5 deg.
# # minor meridian interval = 82/164 = .5 deg.
-mv .4 -pv .4 #draw minor intervals as tic marks, 0.4 cm tall
-c 0 -b 0      # char pen 0, line pen 0; map pen -p0:4 (.006 in).

::::::::::
grid2.c
::::::::::
-pi 25 -mi 84      # draw other parallel and meridian at map border.
-pu .01 -mu .01 # don't draw lines for major parallels.
-s 0.25        #char size in cm
-f -sr         #font "-sr"
-d 0.3         #offset distance for char. annotation in cm
-pj 50 -mj 168
-mv .4 -pv .4 #draw minor intervals as tic marks, 0.4 cm tall
-c 0 -b 0      # char pen 0, line pen 0; map pen -p0:4 (.006 in).

::::::::::
grid3.c
::::::::::
-pi 24 -mi 83  # set parallel interval for every 24 parallels, and every
# #  83rd meridian (this file is to label tic marks halfway up map)
-s 0.25
-f -sr
-d 0.3
-pu 0.4 -mu 0.4 #draw major parallels and meridians as tic marks 0.4 cm.
-c 0 -b 0    # remap pen 0 to pen 4 (.006 in).

::::::::::
grid4.c
::::::::::
-pi 23 -mi 82  #parallel at 23 d, meridian every 82 deg.
-b 1      # line pen 1; map pen -p1:7 (.010 in).

::::::::::
grid5.c
::::::::::
-pi 25 -mi 84  #parallel at 25 d, meridian every 84 deg.
-b 1      # line pen 1; map pen -p1:7 (.010 in).
```

5.2.2 Inset map: regional index The regional index map was created at 1:11,125,000 (page size), but plotted on the Kongsberg at 40% of original size. The control files were put in a directory called "gomdex" on the same level with all the 1:500,000 map directories.

```
::::::::::::::
gomdex/map.c
::::::::::::::
100w 80w 22n 31n 90w
 +proj=aea +lon_0=90w
11125000
0
13 13
3 3

::::::::::::::
gomdex/grid1.c
::::::::::::::
-pi 22 -mi 20
-pu .01 -mu .01
-pj 22 -mj 20
-mv .22 -pv .22
-b 1    # map pen -p1:4(.006 in) for final

::::::::::::::
gomdex/grid2.c
::::::::::::::
-pi 31 -mi 20
-pu .01 -mu .01
-mj 20
-mv .22 -pv .22
-b 1    # map pen -p1:4(.006 in) for final

::::::::::::::
gomdex/grid3.c
::::::::::::::
-pi 22 -mi 20
-b 1    # map pen -p1:4(.006 in) for final
-c 0    # map pen -p0:2 (.004 in) for final
-f -sr
-s 0.4
-d 0.5
```

```
::::::::::::::
gomdex/grid4.c
::::::::::::::
-pi 31 -mi 90
-pu .01
-s 0.4
-d 0.5
-f -sr
-b 1    # map pen -pl:4(.006 in) for final
-c 0    # map pen -p0:2 (.004 in) for final
```

The control files to create the index of 1:500,000 sheets look similar to those used for the 1:2,000,000 track map (Section 4.5). Each column must be separated by a tab character.

```
:::::::::::::::
gomdex/mapdex.c (run thru 'lines' to draw boundaries of 1:500,000 sheets)
:::::::::::::::
# -p 0    #pen 0; use Kongsberg pen 4 (.006 in) in final plot.
# -b
29.5    -90.0
29.5    -88.0
29.5    -86.0
# -b
28.0    -96.0
28.0    -94.0
28.0    -92.0
28.0    -90.0
28.0    -88.0
28.0    -86.0
28.0    -84.5
# -b
26.0    -96.0
26.0    -94.0
26.0    -92.0
26.0    -90.0
26.0    -88.0
26.0    -86.0
26.0    -84.0
# -b
etc.
```

```
::::::::::::::
gomdex/mapnum.c (run thru 'points')
::::::::::::::
# -d 2,1   #longitude in field 2, latitude in field 1 (tab
# #          is default field separator).
# -f 3     #read data from field 3
# -p 0     #pen number, plot final w/ Kongsberg pen 4 (.006 cm)
# -cr 0    #char. rotation = 0 (horizontal)
# -cf -sr  #use font -sr
# -c       #enable character plotting
# -cj c    #center justify
# -cs .45  #make char's 0.45 cm tall
# # Note:  Separate each column with tab.
23.5    -85     2
25.5    -89     5
# -cs .55  # make rest of characters 0.55 cm tall
24      -83     1
25      -85     3
25      -87     4
25      -91     6
25      -93     7
25      -95     8
27      -95     9
# -cx -.325   #slight offset centers two-digit numbers in boxes.
27      -93     10
27      -91     11
27      -89     12
27      -87     13
27      -85.25  14
28.75   -87     15
# -j l       #left justify to avoid landmass
# -cx 0      #reset x-offset to 0 cm.
28.75   -89     16
```

The coastline and international boundary were derived from the MAPGEN 'coast' program using the world map file to avoid too much detail:

```
coast -mo map.def coast.ov -c '-m 0' /usr/coast/old/W_I_500
coast -mo map.def intbound.ov -c '-m 0 -d .1,43690' /usr/coast/sa/bdy
#  -m 0 is pen #, .1 is dash size in cm, 43690 is decimal
#   representation of binary number dash mask (1=dash 0=space).
#  final plot map pen 0 to pen 2 (.004 in).
```

5.3 Bathymetric Contours

Raw bathymetric data are kept in their own directory elsewhere on the system. They are organized with one contour per file, in the format: latitude "tab" longitude; with breaks in the data separated by the #-b symbol. This facilitates creation of the overlay files, permitting the bathymetric data to be run directly into 'lines'. The bathymetry overlay files are kept in a subdirectory of each map called "bat". The "bat" directory also contains a special set of grid overlays that just provide corner tics.

With a script file like "mkbat.cs", the bathymetry overlays were created in subdirectories of each 1:500,000 map as background runs during the night.

```
::::::::::::
mkbat.cs
::::::::::::
# script to be run from subdirectory "bat"
lines -mo ../map.def mgf.0250.ov /usr8/gulfbathy/mgf.0250
lines -mo ../map.def mgf.0500.ov /usr8/gulfbathy/mgf.0500
lines -mo ../map.def mgf.0750.ov /usr8/gulfbathy/mgf.0750
lines -mo ../map.def mgf.1000.ov /usr8/gulfbathy/mgf.1000
lines -mo ../map.def mgf.1250.ov /usr8/gulfbathy/mgf.1250
lines -mo ../map.def mgf.1500.ov /usr8/gulfbathy/mgf.1500
lines -mo ../map.def mgf.1750.ov /usr8/gulfbathy/mgf.1750
lines -mo ../map.def mgf.2000.ov /usr8/gulfbathy/mgf.2000
lines -mo ../map.def mgf.2250.ov /usr8/gulfbathy/mgf.2250
lines -mo ../map.def mgf.2500.ov /usr8/gulfbathy/mgf.2500
lines -mo ../map.def mgf.2750.ov /usr8/gulfbathy/mgf.2750
#
lines ../map.def -o jh.1000.ov /usr31/jh/gloria/gombat/jh.1000
lines ../map.def -o jh.1250.ov /usr31/jh/gloria/gombat/jh.1250
lines ../map.def -o jh.1500.ov /usr31/jh/gloria/gombat/jh.1500
lines ../map.def -o jh.1750.ov /usr31/jh/gloria/gombat/jh.1750
lines ../map.def -o jh.2000.ov /usr31/jh/gloria/gombat/jh.2000
lines ../map.def -o jh.2250.ov /usr31/jh/gloria/gombat/jh.2250
lines ../map.def -o jh.2500.ov /usr31/jh/gloria/gombat/jh.2500
lines ../map.def -o jh.2750.ov /usr31/jh/gloria/gombat/jh.2750
lines ../map.def -o jh.3000.ov /usr31/jh/gloria/gombat/jh.3000
lines ../map.def -o jh.3250.ov /usr31/jh/gloria/gombat/jh.3250
```

These files are mapped to pen 0 by default. They were plotted on the Kongsberg using pen 4 (.006 in) for the intermediate contours and pen 7 (.010 in) for the 1000-m contours. We used a command line to run the plot such as:

```
plotter -d kong -o /dev/"tape" grid*.ov -p0:4 jh.?[1-9]*.ov mgf.?[1-9]*
-p0:7 jh.?000.ov mgf.?000.ov
```

Another alternative is to put your contour overlays in separate directories such as "index" and "inter" and run a command like this:

```
plotter -d kong -o /dev/"tape" grid*.ov -p0:4 inter/*.ov -p0:7 index/*.ov
```

The bat/grid files require one file for each corner tic, of the form:

```
::::::::::::
bat/grid1.c
::::::::::::
-pi 25 -mi 82 # select 25th parallel and 82nd meridian.
-mu .4 -pu .4 # make tics 0.4 cm.
-b 2          # use pen 2 (.003 in)
```

5.4 Geologic interpretations

The geological interpretations were made on, or transcribed to, the 1:500,000 images on copies of the digital bathymetry. The interpretations themselves were not digitized. They were hand scribed on punch-registered overlays for compositing with the digital bathymetry.

6. REFERENCES

Evenden, G.I., and Botbol, J.M., 1985, MAPGEN (UNIX version) Users Manual: U.S. Geological Survey Open File Report 85-706, 58 pp. and appendices.

Paskevich, V.F., 1986, Mrg3rfmt system documentation: Unpublished administrative report on file at U.S. Geological Survey, Woods Hole, Mass.

EEZ-Scan Scientific Staff, 1987, Atlas of the U.S. Exclusive Economic Zone, Gulf of Mexico and Eastern Caribbean Areas: U.S. Geological Survey Miscellaneous Investigations Series I-1864A-B, 162 pp.

7. APPENDIX

A. Code for Fortran routine to convert to day of year.

```
::::::::::::
doy.for
::::::::::::
c
c*******************
c
c
c *   Converts from day, month and year to Julian day for
c        that year.
c
c *   R. Groman
c
c *   Date: 17 July 1975
c
c************************
c
      dimension month(12)
c
      data month(1),month(3),month(4),month(5)/31,31,30,31/
      data month(6),month(7),month(8),month(9)/30,31,31,30/
      data month(10),month(11),month(12)/31,30,31/
c
    3 read(*,5,END=40)iyear,imonth,iday
    5 format(i4,1x,i2,1x,i2)
      jday=iday
      month(2)=28
      if(mod(iyear,4) .eq. 0) month(2)=29
      do 10 i=1,(imonth-1)
   10 jday=jday + month(i)
      write(*,18)jday
   18 format(i3)
      goto 3
   40 continue
      end
```

B. List of Kongsberg plotter pen specifications, National Mapping Division Eastern Mapping Center, Reston, VA.

```
        Kongsberg              Nominal      MAPGEN       Actual size on new film,
        instruction            line wt.     pen #         photo head, disk 1(old)

penmap[] = {
        "D11@M21@",     /*   2/1000 in, pen:  0      .002 in */
        "D12@M21@",     /*   3/1000 in, pen:  1      .003 in */
        "D11@M22@",     /*   4/1000 in, pen:  2      .004 in */
        "D13@M21@",     /*   5/1000 in, pen:  3      .005 in */
        "D11@M23@",     /*   6/1000 in, pen:  4      .007 in */
        "D14@M21@",     /*   7/1000 in, pen:  5      .007 in */
        "D12@M23@",     /*   9/1000 in, pen:  6      .011 in */
        "D13@M22@",     /*  10/1000 in, pen:  7      .010 in */
        "D14@M22@",     /*  14/1000 in, pen:  8      .014 in */
        "D13@M23@",     /*  15/1000 in, pen:  9      .020 in */
        "D14@M23@",     /*  21/1000 in, pen: 10      .027 in */
};

Updated 11/4/87 by Ed Moser;

Magnification 23 (3x), designated M23, should be avoided if possible;
it makes the plot run slower and tends to blur line edges.  For example,
use pen 5 instead of pen 4, use 7 instead of 6.
```

C. Special programs called by arrowscript:

red1
corner
segfit
segang
angmerge2

D. Alternate script to arrowscript with tics every hour:

```
::::::::::::::
plhr.s
::::::::::::::

# Note: maintain blank line above (signals csh that this is a shell script).
# Usage:  sh plhr.s  input  output
#
# awk script to create 1-hr, latitude-longitude file.
# Selects every 1 hour, and reformats NGSDC GLORIA data for input to 'points'.
# Sample line of input follows; fields are separated by one or more spaces:
#FARN   1/85   850809 0000 23.8317 -82.1700          0150201503040
#If input has no space between fields one and two, change '$3' through '$6' to
# '$2' through '$5' in the following lines.
awk < $1 '
$4 ~ /..00/  {
       printf "%f\t%f\t%s\t%s\n",$5, $6, $3, $4
       }
' > $2
```