Geosoft eXecutables (GX's) developed by the U.S. Geological Survey, version 1.0, with a viewgraph tutorial on GX development

By Jeffrey D. Phillips[1], Joseph S. Duval[2], and R.W. Saltus[1]

[1]Denver, CO 80225
[2]Reston, VA 20192

Open-File Report 03-010

U.S. Department of the Interior
U.S. Geological Survey

Introduction

Geosoft executables (GX's) are custom software modules for use with the Geosoft OASIS/Montaj™ geophysical data processing system, which runs under the Microsoft Windows™ operating system. The U.S. Geological Survey (USGS) uses OASIS/Montaj, primarily for processing and display of airborne geophysical data. The ability to add custom software modules to the OASIS/Montaj system is a feature advantageous to the USGS, due to the large number of geophysical algorithms developed by the USGS during the past half century.

This report describes the initial set of GX's developed by the USGS or specifically for the USGS by contractors. They perform fairly basic operations that were missing or perceived to be inadequate in the built-in GX's that came with OASIS/Montaj version 4.2. Some of these GX's or GX's with similar functionality may be incorporated into later versions of OASIS/Montaj.

Download and Installation

The executable GX files, which will only work from within Geosoft OASIS/Montaj, and the source code files used to develop them can be downloaded using a web browser from **ftp://ftpext.usgs.gov/pub/cr/co/denver/musette**. The files are in the **/pub/gx** directory.

Each compiled GX consists of an executable file with the suffix **.gx**. In addition there is a global file containing error messages called **usgs.err** and a menu file called **usgs.omn**. Some GXs require a dynamically linked library with the suffix **.dll**. In the current release of OASIS/Montaj (version 5.1.5), compiled GXs are installed in the **gx** subdirectory of the **\OASISmontaj** directory; error files are installed in the **ger** subdirectory; menu files are installed in the **omn** subdirectory, and dynamic-link libraries are installed in the **bin** subdirectory.

The USGS Menu File and Short GX Descriptions

The GX's described here can be accessed from a USGS menu that can be added to the menu bar in OASIS/Montaj. The menu structure is defined in a text file called **usgs.omn**:

```
MENU "&USGS"
SUBMENU "&Grid Utilities"
SUBMENU "&Database Utilities"
SUBMENU "&Map Utilities"
SUBMENU "&Boundary"
MENU "&USGS/&Grid Utilities"
ITEM "&Test for dummy values"                  ,usgs_dvaltest.gx
ITEM "&Plug holes by regriding"                ,usgs_pluggrid.gx
ITEM "&Plug holes by iterating"                ,usgs_gridplug.gx
ITEM "&Mask a grid"                            ,usgs_gridmask.gx
ITEM "&Horizontal gradient"                    ,usgs_hgrad.gx
ITEM "&Filtered gradients"                     ,usgs_gradcomp.gx
ITEM "&Transpose"                              ,usgs_transpos.gx
ITEM "&Prepare grid for FFT"                   ,usgs_prep4.gx
ITEM "&Proximity of data points"               ,usgs_gridprox.gx
SEPARATOR
MENU "&USGS/&Database Utilities"
ITEM "&Renumber selected lines"                ,usgs_renumlns.gx
SEPARATOR
MENU "&USGS/&Map Utilities"
ITEM "&Digitize vectors"                       ,usgs_digvecs.gx
ITEM "&Draw polygons to a specified group"     ,usgs_pdraw.gx
SEPARATOR
MENU "&USGS/&Boundary"
ITEM "Calculate pseudogravity grid"            ,magmap1.gx
ITEM "Calculate horizontal gradient"           ,usgs_hgrad.gx
ITEM "Find grid peaks"                         ,gridpeak.gx
ITEM "Plot peak locations"                     ,symbols.gx
```

This file can be easily modified to add additional categories and custom GX's as they are developed. The magmap1.gx, gridpeak.gx, and symbols.gx referenced in the menu file are part of the OASIS/Montaj package.

The following GX's are described in this report:

usgs_digvecs.gx - digitize vectors from a map and place them in an ASCII XYZ file.
usgs_dvaltest.gx - test for dummy values in a grid.
usgs_gradcomp.gx - compute filtered horizontal derivative grids and the horizontal gradient magnitude grid using the gradient-component method of Thurston and Brown (1994).
usgs_gridmask.gx - restore holes to a grid from a masking grid.
usgs_gridplug.gx - plug holes in a grid by using minimum curvature iterations.
usgs_gridprox.gx - create a grid containing distance to the nearest data point.
usgs_hgrad.gx - calculate the magnitude of the horizontal gradient of a grid.
usgs_pdraw.gx - draw, redraw, or append polygons from a .ply file to a specified map group.
usgs_pluggrid.gx - plug holes in a grid by regriding.
usgs_prep4.gx - prepare a grid for Fourier transform by plugging holes (if any), and extending the rows and columns.
usgs_renumlns.gx - renumber selected lines in a database.
usgs_transpos.gx - transpose a grid.

Help Files for the Individual GX's

USGS_DIGVECS.HLP

```
USGS_DIGVECS GX

Digitize vectors from a map and place them in an ASCII XYZ file.

INTERACTIVE PARAMETERS

  Output ASCII XYZ File
  Append or Overwrite
  Grid name
  Line Thickness
  Line Color

BATCH PARAMETERS

USGS_DIGVECS.XYZ             output ASCII XYZ file name
USGS_DIGVECS.APPEND          overwrite file = 0; append to output = 1
USGS_DIGVECS.GRID            grid name
USGS_DIGVECS.LINETHICK       line thickness (mm)
USGS_DIGVECS.LINECOLOR       line color

APPLICATION NOTES

If the output file does not exist, you must specify overwrite mode.  If
the file does exist and you specify overwrite mode, you will be asked to
confirm the overwrite.

If no grid is specified, only LINE,X,Y coordinates will be placed in the
output file. If a grid is specified, LINE,X,Y and Z values will be placed
in the output file.  A header record in the output file identifies the
channel names.  To use this header record when importing the XYZ file into
a database, blank out the 'Import template' field.

A map group corresponding to the output file name prefix will be used to
hold the vectors drawn on the map.

Multi-point vectors are digitized using the left mouse button.  The right
mouse button is used to terminate a vector (using the "Done" menu item)
and start a new vector.  Digitizing is terminated by using the right mouse
button and selecting "Done" twice in a row.  The "Cancel" menu item is
equivalent to the "Done" menu item, it does not remove the digitized
vector from the file or the map.  The "Exit Menu" menu item will cancel
the menu and return to digitizing the current vector.

Because the map cannot be redrawn inside a GX, the digitized vectors will
not be drawn until the GX is exited.  For the same reason, the remaining
right mouse menu items won't work properly.

Written by Jeff Phillips (jeff@usgs.gov) 2/9/99, modified 3/21/02.
```

USGS_DVALTEST.HLP

```
USGS_DVALTEST GX

Tests for dummy values in a grid.

INTERACTIVE PARAMETERS

  Input grid file

BATCH PARAMETERS

  USGS_DVALTEST.INGRID           input grid

Written by Jeff Phillips (jeff@usgs.gov) 01/07/00.
```

USGS_GRADCOMP.HLP

```
USGS_GRADCOMP GX

  Computes filtered horizontal derivative and horizontal gradient
  magnitude (HGM) grids using the gradient-component method of
  Thurston and Brown (1994).

INTERACTIVE PARAMETERS

  Input grid file
  Output HGM grid file
  Output x(east)-derivative grid file
  Output y(north)-derivative grid file
  Window size (odd and < 22)
  Polynomial order (< Window size)

BATCH PARAMETERS

USGS_GRADCOMP.GRID           input grid
USGS_GRADCOMP.HGRID          output HGM grid
USGS_GRADCOMP.XGRID          output x-derivative grid
USGS_GRADCOMP.YGRID          output y-derivative grid
USGS_GRADCOMP.WINDOW         window size
USGS_GRADCOMP.POLY           polynomial order

APPLICATION NOTES

  Computes filtered horizontal derivative and horizontal gradient
  magnitude grids using the gradient-component method of Thurston and
  Brown (1994).  The user must specify a window size w and a polynomial
  order n < w.  The standard first-difference operator corresponds to
  w = 3, n = 2.  For this and similar operators having n = w-1, there is
  no attenuation of short wavelengths.  For operators having n < w-1,
  short wavelength features will be attenuated.

  Source code modified from Thurston and Brown (1994) by Jeff Phillips.
  The maximum number of columns in the grid is limited to 8000.

  Reference:

  Thurston, J.B., and Brown, R.J., 1994, Automated source-edge location
       with a new variable pass-band horizontal-gradient operator:
       Geophysics, v.59, no.4, p.546-554.

Written by Jeff Phillips (jeff@usgs.gov) 10/13/99.
```

USGS_GRIDMASK.HLP

```
USGS_GRIDMASK GX

Masks a grid file by inserting dummy values from a second grid file
(normal masking) or by inserting the non-dummy values from the second
grid file as dummy values in the masked grid file (inverse masking).

INTERACTIVE PARAMETERS
```

```
  Grid  file to be masked
  Grid file containing the masking values
  Output masked grid file
  Normal or inverse masking

BATCH PARAMETERS

USGS_GRIDMASK.INGRID1       input grid file to be masked
USGS_GRIDMASK.INGRID2       input masking grid file
USGS_GRIDMASK.OUTGRID       output masked grid file
USGS_GRIDMASK.MTYPE         type of masking (Normal or Inverse)

APPLICATION NOTES

The maximum column dimension is 16384 for all grids.

Written by Jeff Phillips (jeff@usgs.gov) 01/18/00.
```

USGS_GRIDPLUG.HLP

```
USGS_GRIDPLUG GX

Plugs dummy values in a grid file by using polynomial or local
median initialization followed by minimum curvature iterations.

INTERACTIVE PARAMETERS

  Grid  file to be plugged
  Output plugged grid file
  Polynomial order (0 to 3, or -1 for local median replacement)
  Number of minimum curvature iterations

BATCH PARAMETERS

USGS_GRIDPLUG.INGRID        input grid file to be plugged
USGS_GRIDPLUG.OUTGRID       output plugged grid file
USGS_GRIDPLUG.IORDER        polynomial order for initialization
                              (0 to 3 or -1 for local median)
USGS_GRIDPLUG.NITER         number of minimum curvature iterations
                              (e.g., 100)

APPLICATION NOTES

The maximum column dimension of the grids is 10,000.

Written by Jeff Phillips (jeff@usgs.gov) 12/16/2002.
```

USGS_GRIDPROX.HLP

USGS_GRIDPROX GX

Use this GX to create a grid containing distance to the nearest data point.

INTERACTIVE PARAMETERS

  X channel
  Y channel
  Input grid (used to determine size and spacing of output grid)
  Output grid

BATCH PARAMETERS

USGS_GRIDPROX.DB
USGS_GRIDPROX.XCH
USGS_GRIDPROX.YCH
USGS_GRIDPROX.INGRID
USGS_GRIDPROX.OUTGRID

APPLICATION NOTES

This GX is written entirely in native gxc code - it is slow.

Written by Rick Saltus, USGS (saltus@usgs.gov).

USGS_HGRAD.HLP

USGS_HGRAD.GX

This GX calculates the magnitude of the horizontal gradient of a grid.

INTERACTIVE PARAMETERS

        Input grid
        Output horizontal gradient magnitude grid

BATCH PARAMETERS

        USGS_HGRAD.IN = Input grid
        USGS_HGRAD.OUT = Output horizontal gradient magnitude grid

APPLICATION NOTES

The HORIZONTAL GRADIENT MAGNITUDE is the square root of the sum of the squares of the derivatives in the x, and y directions:

        hgrad = sqrt (dx*dx + dy*dy)

Written by Northwest Geophysical Associates for the USGS.

USGS_PDRAW.HLP

The USGS_PDRAW GX draws the polygons from a polygon file on the map.

Draw polygons from a polygon file dialog box options

Polygon file name (.ply)	The name of the polygon file. Script Parameter: USGS_PDRAW.FILE
Map view	Map view in which to draw, select from the list. Script Parameter: USGS_PDRAW.VIEW
line thickness (mm)	The polygon outline line thickness in mm. Script Parameter: USGS_PDRAW.LINETHICK
Line colour	Enter a line colour of form RxxxGxxxBxxx. Script Parameter: USGS_PDRAW.LINECOLO
Fill colour	Enter a fill colour of form RxxxGxxxBxxx. Script Parameter: USGS_PDRAW.FILLCOLO
Group Name	Enter the group name Script Parameter: USGS_PDRAW.GROUPNAME
Group Action	Enter a zero (0) to replace the group (i.e. Erase and replot) and a one (1) to append polygons to the group. Script Parameter: USGS_PDRAW.GROUPACTN

Application Notes

Polygons are ASCII files with default extension .ply. A polygon file contains a list of X,Y coordinates that define one or more polygons. The file may contain any number of polygons, and each polygon may have any number of vertices. The first and last points in each polygon are assumed to connect. If the file will contain more than one polygon, each polygon must start with a line 'poly #' ('p' or 'P' in column 1). Comment lines are indicated by a '/' in column 1. Please note that the polygon coordinates are assumed to be in the same coordinate system as that of the map view on which the polygons will be drawn.

Following is an example of a single polygon file:

```
/
/  Sample single polygon file
/----------------------------
poly 1
    1376027.6061    6178025.9399
    1382846.3129    6178025.9399
    1382846.3129    6182748.8182
    1376027.6061    6182748.8182
    1376027.6061    6178025.9399
```

Following is an example of a three-polygon file:

```
/
/ Sample three polygon file
/--------------------------
poly 1
    1375594.6724    6181961.6718
    1378706.3838    6181961.6718
    1378706.3838    6184716.6842
    1375594.6724    6184716.6842
    1375594.6724    6181961.6718
poly 2
    1375757.0225    6178629.4188
    1379896.9516    6178629.4188
    1379896.9516    6181174.5254
    1375757.0225    6181174.5254
    1375757.0225    6178629.4188
poly 3
    1380546.3523    6175926.8828
    1384550.9896    6175926.8828
    1384550.9896     6179022.992
    1380546.3523     6179022.992
    1380546.3523    6175926.8828
```

TOPIC MODIFIED BY JOE DUVAL, USGS (03/04/2002)

USGS_PLUGGRID.HLP

```
USGS_PLUGGRID GX

Plugs holes in a grid using minimum curvature.

INTERACTIVE PARAMETERS

  Input grid file
  Output plugged grid file

BATCH PARAMETERS

  USGS_PLUGGRID.INGRID           input grid
  USGS_PLUGGRID.PLUGGRID         output plugged grid

APPLICATION NOTES

  Tests input grid for dummy values.  If dummy values are
  found, runs GRIDGDB.GX to copy valid grid values to a
  temporary database, then runs RANGRID.GX to create the
  plugged grid.

Written by Jeff Phillips (jeff@usgs.gov) 01/07/00.
```

USGS_PREP4.HLP

```
USGS_PREP4 GX

Prepares a grid for Fourier transform by plugging holes, and extending the
rows and columns.

INTERACTIVE PARAMETERS

  Input grid file
  Output plugged and extended grid file
  Percent expansion

BATCH PARAMETERS

USGS_PREP4.INGRID           input grid
USGS_PREP4.PREPGRID         output plugged and extended grid
USGS_PREP4.PCT              percent expansion

APPLICATION NOTES

Maximum dimensions of the extended grid are 16384 x 16384.  The input
grid should be smaller than this.

Initial processing tests the input grid for dummy values; if any exist,
the valid grid values are copied into a temporary database called
Prep4.gdb and regridded using minimum curvature into a new grid called
Prep4.plg.

The input grid (or Prep4.plg) is then processed through the
following 8 steps:
1. Get the min/max of the grid and the extended dimensions.
2. Extend the grid to the right using prediction filtering.
3. Transpose the extended grid.
4. Extend the transposed grid to the right.
5. Smooth across the first extension.
6. Transpose back.
7. Smooth across the second extension.
8. Center the data in the grid.

Written by Jeff Phillips (jeff@usgs.gov) 01/18/00.
```

USGS_RENUMLNS.HLP

```
Use the USGS_RENUMLNS GX to renumber all selected lines.  This is useful for changing line
numbers prior to merging databases.
```

Renumber all selected lines dialog box options

Starting Line Number — Enter the starting line number.
Script Parameter: USGS_RENMULNS.STRTLN

Line number Increment — Enter the increment to be used for numbering sequential lines.
Script Parameter: USGS_RENUMLNS.INCRLN

Application Notes

If any parameters are blank, the default parameter value is set to 1.
The process will stop if the new name of a selected line already exists.
All lines processed up to that point will be changed.

TOPIC WRITTEN BY JOE DUVAL, USGS (11/21/2000)

USGS_TRANSPOS.HLP

USGS_TRANSPOS GX

Transpose a grid.

INTERACTIVE PARAMETERS

Input grid file
Output grid file

BATCH PARAMETERS

USGS_TRANSPOS.INGRID	input grid
.OUTGRID	output grid

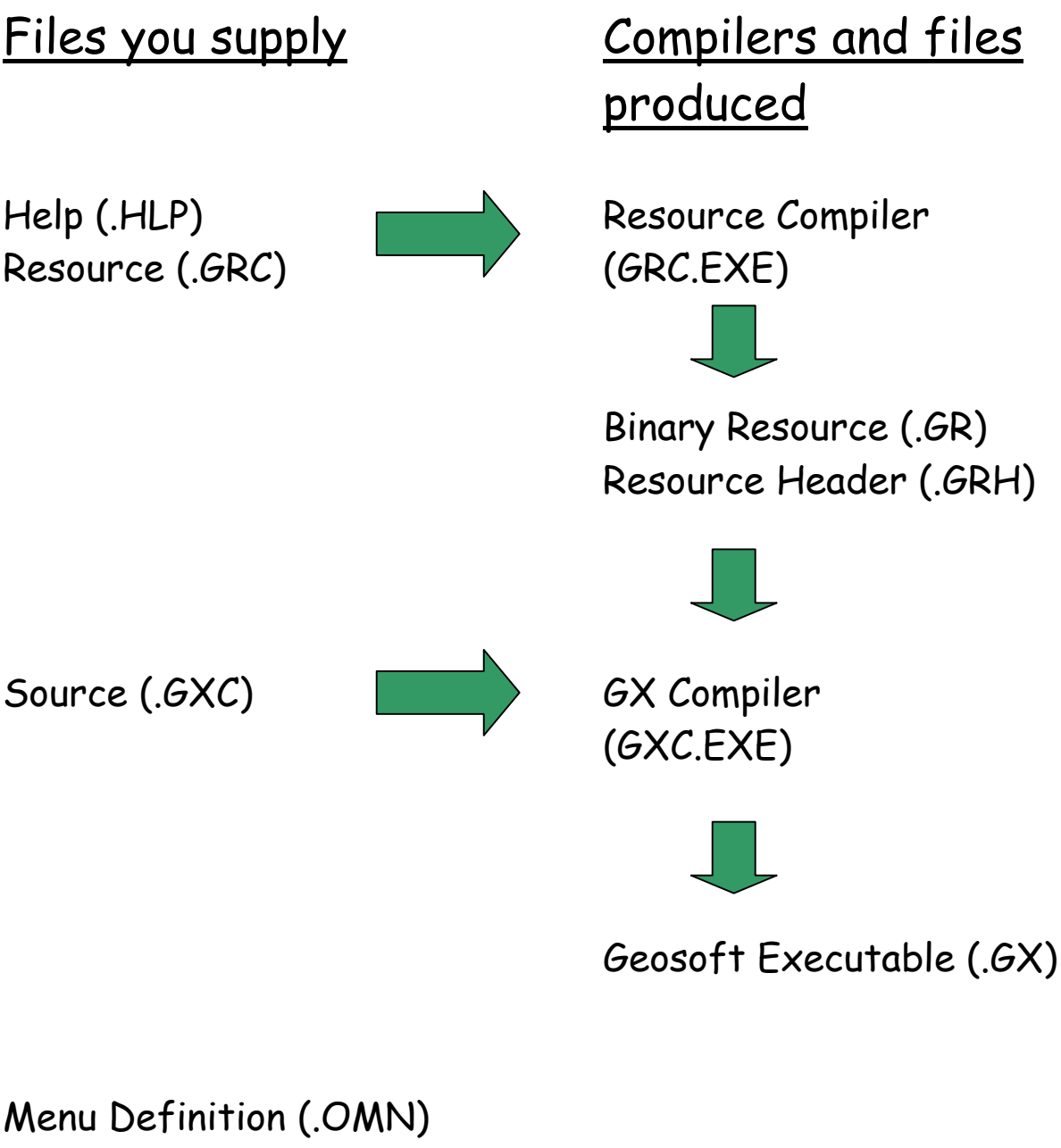
APPLICATION NOTES

Written by Jeff Phillips (jeff@usgs.gov) 1/11/00.

Source Code Files and Tutorials

Each GX is built from a number of source code files. These include, at a minimum, an ASCII help file with the suffix **.hlp**, a GX source code file with the suffix **.gxc**, and a resource file with the suffix **.grc**. Some GXs use a dynamic link library compiled from Fortran code. In addition to the three source code files described above, these GXs require the Fortran source code file with a suffix of **.f**, which is converted to C code using Geosoft's version of **f2c.exe**; a C code wrapper function for the Fortran subroutine calls, usually called **gxx_*.c**; the generic wrapper functions **wfuncs.c**; the C header files **f2c.h** and **wrappers.h**, and a **.gxh** prototype file. The interrelationships of these files are described in the following tutorial on GX development, which is suitable for printing in viewgraph format.

SIMPLE GX DEVELOPMENT



STEPS IN DEVELOPMENT

1. Decide exactly what the GX will do. Choose a name for the GX. Determine the input required from the user.

2. Create a new GX project directory and copy an existing, somewhat similar GX source code (.GXC) file, resource (.GRC) file, and help (.HLP) file to be used as prototypes. Rename the files to the name of the new GX.

3. Edit the resource (.GRC) file and the help (.HLP) file to reflect your design. Run the GRC resource compiler, which is in the \OASISmontaj\bin directory on the PC. Correct syntax errors and warnings.

4. Edit the source code (.GXC) file to perform the required function. Remember to correct the GX name in the description and the names of the included resource files. Note: library routines are poorly documented; see the header (.GXH) files in the INCLUDE directory on the CD-ROM and the source code examples under the SRC directory. Compile using the GXC source compiler, also in the \OASISmontaj\bin directory on the PC. Correct errors.

5. Move the compiled GX to the \OASISmontaj\gx directory. Start Oasis and test the GX. Correct any errors you see at run time.

6. Include the GX in a new or existing menu definition (.OMN) file in the \OASISmontaj\omn directory for easier access from the Oasis menu bar.

WHAT'S IN THE USER-SUPPLIED FILES?

Help File (mygx.HLP) - text only; should describe variables and the function of the entire GX.

Resource File (mygx.GRC) - Defines the user interface. Components include:

// - comment lines

FORM - A dialog box with one or more of:

- EDIT - text entry fields
- FEDIT - filename entry fields
- LEDIT - drop-down lists
- EBUT - pushbuttons
- HBUT - help pushbuttons

LIST - drop-down list definition for each LEDIT with two or more:

- ITEM - list items

HELP - pointer to the help file

Source File (mygx.GXC) - pseudo-C code with the following sections:

Description

- NAME = "My GX"
- VERSION = "v1.00a US Geological Survey"
- DESCRIPTION = "A multi-line text description"

Resources

- RESOUCE = "mygx.gr"
- #include "mygx.grh"

Includes

- #include <all32.grh>

Variable definitions

- strings, reals, ints, DGW (dialog objects), etc.

```
Code
        {
        // Interactive Parameter Block specification
            if (iInteractive_SYS()) {          // Are we running
                                                    interactively?
            Diag = Create_DGW("MYGX");    // Create the dialog
                SetInfoSYS_DGW(Diag,...);// Set default
                                                    parameters
                i = iRunDialog_DGW(Diag);  // Run dialog
                GetInfoSYS_DGW(Diag,...);// Load new
                                                    parameters
            }
        // Read Parameter Block
            GetString_SYS("MYGX",...);       // Get strings
            iVar = iGetInt_SYS("MYGX",...);  // Get integers
            rVar = rGetReal_SYS("MYGX",...);       // Get reals
        // Initialize objects
        // Process the data
        // Cleanup and exit
        }
```

Menu Definition File (USGS.OMN)

```
/
/ USGS
/----------
MENU "&USGS"
SUBMENU "&Digitizing"
MENU "&USGS/&Digitizing"
ITEM "&digitize vectors"        ,mygx.gx
SEPARATOR
```

CONVERTING THE EXAMPLE FORTRAN CODE TO A GX USING GEOSOFT'S F2C AND LIBRARIES

REQUIRED: Microsoft Visual C++ version 6.0

1. Start Visual C++ and create a new Project/Workspace using
   File / New / Win32 Dynamic-Link Library.
   Give the new project a name that corresponds to the DLL name in the .gxh file ("example" in this case) and a location. Note that OASIS/Montaj doesn't like DLL names with prefixes longer than 11 characters.
   Create an empty DLL project.
   The result will be a new directory (example) containing files example.dsp, example.dsw, and example.ncb

2. Close or iconify Visual C++.

3. Copy the Fortran source code (example.f) to the new directory.
   (If this were your code, you would need to edit it to:
   a. replace all user I/O with variables passed through subroutine calls.
   b. replace all file I/O with calls to wrapper functions.)

4. Run Geosoft's f2c.exe as:
   f2c -A example.f
   This will create example.c in your directory.

5. Copy wfuncs.c, gxx_example.c, wrappers.h, and f2c.h to your directory. (If this were your code, you would need to create or modify the first three files.)

6. Restart Visual C++ and build the project using:
   Project / Add to Project / Files - add all files.
   Project / Settings / Link - add geogx.lib to the end of the object/library modules list.
   Tools / Options / Directories*
      include files – add C:\OASISmontaj\GxDev\Fortran\f2c
                        C:\OASISmontaj\GxDev\C\include
      library files - add C:\OASISmontaj\GxDev\C\lib
   Build / Build example.dll

7. Close or iconify Visual C++

8. Copy the new dynamic -link library (example.dll) from the Debug subdirectory to the working directory.

9. Copy example.rtf, example.grc, example.gxc, and example.gxh to your working directory.

10. Run grc example.

11. Setup your environment:
   set include=C:\OASISmontaj\GxDev\include

12. Run gxc example.

13. Copy example.gx to the C:\OASISmontaj\gx directory, and example.dll to the C:\OASISmontaj\bin directory.

14. Test it from within Oasis.

* You only need to do this once.

CONVERTING YOUR FORTRAN CODE TO A GX USING GEOSOFT'S F2C AND LIBRARIES

1. Follow the example (example.f), and use the supplied wrapper functions (example\wfuncs.c and wrappers.h).

2. You need the following files to create the DLL:
   - yourcode.f - edited to remove or replace I/O with parameters passed through subroutine calls or with calls to wrapper functions.
   - yourcode.c - the result of f2c -A yourcode.f
   - f2c.h - copy it from the example directory.
   - gxx_yourcode.c - wrapper function for IUSGS_yourcode.
   - wfuncs.c - the _WF wrapper functions.
   - wrappers.h - wrapper include files.

3. You need the following files to create the gx:
   - yourcode.dll - produced by Visual C++ 6.0.
   - yourcode.grc - resource source.
   - yourcode.gxc - gx source.
   - yourcode.gxh - gx include file – defines the DLL name.
   - yourcode.err or yourcode.ger - optional error messages.

4. Once the gx is compiled, the following files need to be copied to the C:\OASISmontaj\gx \bin and \ger subdirectories respectively:
   - yourcode.gx - the compiled gx
   - yourcode.dll - the dynamic link library
   - yourcode.err or yourcode.ger - optional error messages

WHAT TO DO IF IT DOESN'T WORK

1. Check the Fortran code:
    a. Make sure you have the correct number of file I/O calls to wrapper functions.
    b. Make sure the variables in the wrapper function calls are passed in the correct order.
    c. Check the f2c output for correct typing of variables. F2c seems to require implicit Fortran typing for integers. For example, this doesn't work:

```
Subroutine mysub(order)
Integer*4 order
```

    But this does:

```
Subroutine mysub(norder)
```

2. Make sure that you have included yourcode.gxh in yourcode.gxc with the statement:

```
#include "yourcode.gxh"
```

and that the library name in yourcode.gxh is correct and not longer than 11 characters.

3. Make sure the calls in gxx_yourcode.c are correct. See the following table.

yourcode.f	yourcode.c	gxx_yourcode.c	
		definition	call
character*(*) infile	char *infile	const char *pcInfile	const char *pcInfile
	ftnlen infile_len	long *plInfile_len	*plInfile_len
integer*4 ntotal	integer *ntotal	long *plNotal	plNtotal
Real*4 blat	real *blat	double *pdBlat	&fBlat
Real*8 ave	doublereal *ave	double *pdAve	pdAve

So the Fortran code (in yourcode.f):

```
subroutine mysub(infile, ntotal, blat, ave,ierr)
character*(*) infile
real*8 ave
```

becomes the C code (in yourcode.c):

```
int mysub_(char *infile, integer *ntotal, real *blat, doublereal *ave,
                integer *ierr, ftnlen infile_len)
```

which is referenced (in gxx_yourcode.c) via the wrapper function:

```
GX_WRAPPER_FUNC GX_LONG GX_WRAPPER_CALL
IUSGS_Mysub(void      *pGeo,
            const char *pcInfile,
            long *plInfile_len,
```

```
                double *pdblat,
                double *pdave)
{
  const char *modn = "IUSGS_Mysub";

  float fBlat;
  long lErr=0;

  // --- Load the global structure ---

  InitGlobals(pGeo);

  // --- Convert doubles to floats --

  fBlat = (float)*pdBlat;

  mysub_((char *)pcInfile, plNtotal, &fBlat, pdAve, &lErr,
*plInfile_len);

  // --- Transfer returned values back to doubles --

  *pdBlat = fBlat;

  return 0;
}
```

4. Use debugging options

A. Debug the gxc code using assert and abort:

```
Assert_SYS(i);  // will display a mesage if i is zero or
                FALSE.
```

Abort_SYS("mesage"); // will stop execution and display the message.

B. Debug the Fortran code using wrapper functions. For example, you can check that parameters are being passed correctly to the DLL by:

1. placing the following calls near the top of your Fortran code:

```
Call RegisterErr_WF(1000,'yourcode')
Call SetErrParmS_WF(1,infile,*)
Call SetErrParmI_WF(2,ntotal)
Call SetErrParmR_WF(3,blat)
```

2. adding an error file yourcode.err to the C:\OASISmontaj\ger directory containing:

```
#1000
! infile = %1
! ntotal = %2, blat = %3
```

3. and registering the error file in the InitGlobals section of wfuncs.c:

```
IStrcpy_STR(pGeo, Globals.szErrFile,"yourcode.err",
                    &lSize);
```

FORTRAN CONVERSION SUMMARY