U.S. Department of the Interior
Geological Survey

DLG2ISM, a Fortran program to read DLG-3 Optional Format Digital Data Files into the VAX/VMS version of the Interactive Surface Modeling software package.

By

Gregory N. Green

Open-File Report
88-258-A Documentation (Paper Copy)
88-258-B Source Code DLG2ISM.FOR (Disk)

Denver, Colorado
March 1988

Fully topologically structured level 3 optional format Digital Line Graph (DLG) data files are available from the National Cartographic Information Center of the U.S. Geological Survey. These files are collections of significant points, lines, and polygons connected in spatial relationships. Early DLG-3 files contain political boundaries, hydrography and the BLM public land surveys. The DLG-3 files since 1985 may also contain major transportation, miscellaneous significant manmade structures, hypsography (topographic contour data), surface cover, some non-vegetative surface features and survey control and markers categories.

Polygons are not defined explicitly in a DLG-3 file. Although the larger and more powerful geographic information system (GIS) packages have routines to relate, link and build polygons from the DLG-3 line data files, more modest graphic analysis software programs usually can not construct polygons from just the line work. However, enough information is contained within a topologically structured DLG-3 optional format data file to rebuild the polygons.

Program DLG2ISM.FOR when compiled reads DLG-3 optional format 1:24,000 and 1:100,000 Digital Line Graph data files and creates surface annotation files for the VAX/VMS version of the Dynamic Graphics software program, Interactive Surface Modeling (ISM) Version 6.92A. Every DLG-3 point, line or polygon has a unique identification number. A line in the resulting ISM line file is assigned the same identifier as the DLG-3 line and is a projected attributed sequential list of the x,y pairs of that line.

Building ISM polygons from the DLG-3 optional line files can be difficult. In order to sequential arrange the lines required to construct a polygon, the lines that surround the polygon are isolated from the other lines. These lines are then sorted to match the end of one line to the start of the next line. DLG2ISM identifies internal polygons and treats them separately to prevent overprinting of patterns. Finally the DLG-3 optional format area attributes are attached. The resulting ISM polygon has the same DLG-3 area identifier and is a projected attributed sequential closed list of x,y pairs. Because an ISM annotation file is standard ASCII text, this software should work on a variety of systems.

The program prompts the user for the name of an input file, and for the names for the ISM line and polygon annotation files. If the input file is not found, DLG2ISM, will prompt the user again. If the data file is not DLG-3 optional format or if the file exceeds the current limits, the program should stop without creating line or polygon files.

There are two files on the attached diskette. DLG2ISM.DOC is a short description of the program. The source code for the DLG2ISM.FOR is a Fortran program written for minicomputers. To facilitate distribution the software and documentation are on a micro-computer MS-DOS five and one-quarter inch floppy disk. An example of a successful run of the DLG2ISM program is given below.

CBOUN.DLG is the DLG-3 optional political boundary input file for Hicks Dome, Illinois and Kentucky at 1:100,000. LINE.ANN and POLYGON.ANN are the annotation output files for ISM.

```
$ RUN DLG2ISM

DLG3 OPTIONAL to ISM Annotation Conversion Routine

Name of DLG3 Optional input file ? CBOUN.DLG

HICKS DOME, ILL. & KY. DLG DATA

BOUNDARY
PRODUCED 1988
AT A SCALE OF 1 :  100000
DLG COVERAGE IS CBOUN

NUMBER OF NODES -      39
NUMBER OF AREAS -       8
NUMBER OF LINES -      45
UTM PROJECTION

Name of ISM line output file? LINE.ANN

Name of ISM poly output file? POLYGON.ANN

 READING NODE FILE
 READING AREA FILE
 READING LINE FILE
 BUILDING POLYGONS
$
```

Because there are far more DLG-3 attributes than there are ISM line or area surface annotation attributes, most DLG attributes are converted to a simplified ISM format. Use the ISM graphic editor to modify the color or patterns in the simplified annotation file.

DLG2ISM was written in Fortran and compiles without error with the Digital Equipment VAX Fortran compiler. No other compiler was tested to insure compliance with the Fortran-77 standards. The current maximum limits are 750 nodes, 750 areas, 1000 lines, 1000 lines per area and 500 points per line. Because of the size of arrays some computer systems may not have enough memory. To change the limits of DLG2ISM, modify the PARAMETER statements. To tailor DLG2ISM for site specific applications or to support other graphic analysis programs modify the SETATTB subroutine.

Dynamic Graphics, Inc., 1986, ISM Interactive Surface Modeling User's Guide, Dynamic Graphics, Inc, Berkeley, California.

USGS, 1985, USGeoData Digital Line Graphs from 1:100,000-Scale Maps, Data Users Guide 2, U.S.Geological Survey, Reston Virginia.

USGS, 1986, USGeoData Digital Line Graphs from 1:24,000-Scale Maps, Data Users Guide 1, U.S.Geological Survey, Reston Virginia.

```
C=====================================================================C
C                                                                     C
C                              DLG2ISM                                C
C                                                                     C
C   DLG2ISM, a Fortran program to read DLG-3 Optional Format Digital  C
C   Data Files into the VAX/VMS version of the Interactive Surface    C
C   Modeling software package.                                        C
C                                                                     C
C                                 By                                  C
C                                                                     C
C                           Gregory N. Green                          C
C                                                                     C
C                           Open-File Report                          C
C                     88-258-A Documentation (Paper Copy)             C
C                     88-258-B Source Code DLG2ISM.FOR (Disk)         C
C                                                                     C
C                   United States Department of the Interior          C
C                            Geological Survey                        C
C                        Office of Mineral Resources                  C
C                            Denver, Colorado                         C
C                               March 1988                            C
C                                                                     C
C 19 Oct. 1987: Original Version                                      C
C 28 Jan. 1988: Bug fix, multiple islands within islands sequence.    C
C  8 Feb. 1988: Parameter statements added to adjust array sizes.     C
C 28 Mar. 1988: Publication release date                              C
C                                                                     C
C  This program converts U.S.G.S. DLG-3 optional files to ISM format. C
C                                                                     C
C  Although program tests have been made, no guarantee (expressed or  C
C  implied) is made regarding program correctness, accuracy, or proper C
C  execution on all computer systems.                                 C
C                                                                     C
C=====================================================================C
C                                                                     C
C Because there are far more DLG attributes than ISM attributes, most C
C DLG attributes are not converted to ISM form.  Use the ISM graphic  C
C editor to change color or pattern attributes.                       C
C                                                                     C
C DLG2ISM was written and compiled on a Digital Equipment Vax, VMS    C
C 4.6, with the DEC Fortran 77 version 4.7.                           C
C Some possible compile time problems:                                C
C   Single quotes are used around character strings.                  C
C   A few variables are longer than the standard 6 characters         C
C   Not all compilers support these OPENFILE subroutine statements.   C
C       INQUIRE(FILE=INFILE,EXIST=FOUND)                              C
C       OPEN (IOFILE,FILE=OUTFILE,STATUS='NEW')                       C
C       OPEN (DLG,FILE=INFILE,STATUS='OLD',READONLY)                  C
C       REWIND (UNIT=DLG)                                             C
C                                                                     C
C=====================================================================C
C References:                                                         C
C  Dynamic Graphics, Inc., 1986, ISM Interactive Surface Modeling     C
C       User's Guide, Dynamic Graphics, Inc, Berkeley, California.    C
C                                                                     C
```

```
C  USGS, 1985, USGeoData Digital Line Graphs from 1:100,000-Scale      C
C       Maps, Data Users Guide 2, U.S.Geological Survey, Reston       C
C       Virginia.                                                     C
C                                                                     C
C  USGS, 1986, USGeoData Digital Line Graphs from 1:24,000-Scale Maps, C
C       Data Users Guide 1, U.S.Geological Survey, Reston Virginia.   C
C                                                                     C
C  Any user of trade names and trademarks is for identification       C
C  purposes only and does not constitute endorsement by the U.S.      C
C  Geological Survey.                                                 C
C                                                                     C
C=====================================================================C
C the current array limits:                                           C
C                                                                     C
C ( MAXIMUM NUMBER OF NODES )            = MXNODE =  750              C
C ( MAXIMUM NUMBER OF AREAS )            = MXAREA =  750              C
C ( MAXIMUM NUMBER OF LINES )            = MXLINE = 1000              C
C ( MAXIMUM NUMBER OF LINES / AREA )     = MXARCS = 1000              C
C ( MAXIMUM NUMBER OF X,Y PAIRS / LINE ) = MXPAIR =  500              C
C=====================================================================C
      PARAMETER MXNODE = 750
      PARAMETER MXAREA = 750
      PARAMETER MXLINE = 1000
      PARAMETER MXARCS = 1000
      PARAMETER MXPAIR = 500
C *
      DIMENSION HCODE(21),PARMS(4)
      DIMENSION AID(MXAREA),ALINK(MXARCS,MXAREA),ACODE(6,MXAREA)
      DIMENSION LID(MXLINE),LCODE(7,MXLINE)
      DIMENSION LN(MXNODE),SN(MXNODE),EN(MXNODE),ATTB(2,MXAREA)
      DIMENSION X(MXPAIR,MXLINE),Y(MXPAIR,MXLINE)
      DIMENSION INFX(4),INFY(4)
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 HCODE,AID,ACODE,ALINK,LCODE,LID,ATTB
      REAL*4 X,Y,PARMS,INFX,INFY
      DATA IN/5/,SCREEN/6/,DLG/10/,OUTN/11/,OUTA/12/,OUTL/13/
      DATA STRUCTURE/0/
C *
C * Open the DLG input file
      CALL OPENFILE (DLG)
C * Type to screen dlg header information
      CALL HEADER (HCODE,PARMS,INFX,INFY)
C *
C * Are there more points, lines or areas than allowed ?
      IF (HCODE(11).GT.MXNODE) THEN
       WRITE (SCREEN,601) MXNODE
       STOP
      ENDIF
      IF (HCODE(20).GT.MXLINE) THEN
       WRITE (SCREEN,602) MXLINE
       STOP
      ENDIF
      IF (HCODE(15).GT.MXAREA) THEN
       WRITE (SCREEN,603) MXAREA
       STOP
      ENDIF
```

```
C * open output files
C     IF (HCODE(11).GT.0) CALL OPENFILE (OUTN)
      IF (HCODE(20).GT.0) CALL OPENFILE (OUTL)
      IF (HCODE(15).GT.0) CALL OPENFILE (OUTA)
C *
C * read the node, area and line files from the DLG input files
      IF (HCODE(11).GT.0) CALL NODES (HCODE)
      IF (HCODE(15).GT.0) CALL AREAS (HCODE,AID,ALINK,ACODE,ATTB
     + ,MXARCS,MXAREA)
      IF (HCODE(20).GT.0) CALL LINES (HCODE,LCODE,X,Y,PARMS
     + ,MXPAIR,LID,MXLINE)

      DO J=1,HCODE(15)
      STRUCTURE = STRUCTURE + ACODE(2,J)
      ENDDO
C * Build output polygons
      IF (STRUCTURE.GT.0) THEN
       CALL POLYS(HCODE,ACODE,LCODE,AID,ALINK,X,Y,ATTB
     + ,MXARCS,MXPAIR,MXLINE,MXAREA,MXNODE,LN,SN,EN)
      ELSE
       WRITE (SCREEN,604)
       STOP
      ENDIF
C *
C * close up shop
      CLOSE (DLG)
C     CLOSE (OUTN)
      CLOSE (OUTA)
      CLOSE (OUTL)
C *
601   FORMAT (' FATAL ERROR: MORE THAN ',I5,' NODES')
602   FORMAT (' FATAL ERROR: MORE THAN ',I5,' LINES')
603   FORMAT (' FATAL ERROR: MORE THAN ',I5,' AREAS')
604   FORMAT (' FATAL ERROR: INPUT FILE NOT DLG-3 OPTIONAL FORMAT')
      END
C
C ==================== SUBROUTINE HEADER ============================
      SUBROUTINE HEADER (HCODE,PARMS,INFX,INFY)
C * read the DLG-3 optional header
      CHARACTER DESCRIBE*72,NAMES*40,YEAR*10,NETNAM*20,TIC*2
      DIMENSION HCODE(21),PARMS(4),INFX(4),INFY(4)
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*4 SCALE
      INTEGER*2 HCODE
      REAL*4 PARMS,INFX,INFY
      DATA DLG/10/
C *
C * this could be expanded to go off and read DLG-3 standard format,
C * or DLG-3 optional 1:2,000,000.  On err spin off to different read.
C *
      READ (DLG,1001,ERR=1) DESCRIBE
1     READ (DLG,1002,ERR=2) NAMES,YEAR,SCALE
2     READ (DLG,1003,ERR=3) (HCODE(I),I=1,8)
3     READ (DLG,1004,ERR=4)
4     READ (DLG,1005,ERR=5) (PARMS(I),I=1,4)
```

```
5     DO I=1,4
       READ (DLG,1006,ERR=6) TIC,INFX(I),INFY(I)
6     ENDDO
      DO I=1,(HCODE(7)-4)
       READ (DLG,1007,ERR=7) TIC
7     ENDDO
      READ (DLG,1008,ERR=8) NETNAM,(HCODE(I),I=9,21)
8     CALL TEXT (DESCRIBE,NAMES,YEAR,SCALE,NETNAM,HCODE)
      RETURN
C *
1001  FORMAT (A72)
1002  FORMAT (A40,1X,A10,1X,I8)
1003  FORMAT (/,4I6,18X,4I6)
1004  FORMAT (////)
1005  FORMAT (4D18.11)
1006  FORMAT (A2,34X,2F12.2)
1007  FORMAT (A2)
1008  FORMAT (A20,I4,2I6,1X,2I1,1X,2I6,1X,3I1,2I6,3X,I1)
      END
C
C ==================== SUBROUTINE NODES ================================
      SUBROUTINE NODES (HCODE)
C * read the DLG-3 optional nodes and ignore
      CHARACTER*1 NAL
      DIMENSION HCODE(21),NCODE(6),NLINK(12)
      DIMENSION NMAJOR(12),NMINOR(12)
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 HCODE,NCODE,NLINK,NMAJOR,NMINOR
C *
      WRITE (SCREEN,1000)
      DO J=1,HCODE(11)
       READ (DLG,1001) NAL,(NCODE(I),I=1,6)
       IF (NAL.NE.'N') RETURN
       IF (NCODE(2).NE.0) READ(DLG,1002) (NLINK(I),I=1,NCODE(2))
       IF (NCODE(4).NE.0) READ (DLG,1003)
     +  (NMAJOR(I),NMINOR(I),I=1,NCODE(4))
      ENDDO
      RETURN
C *
1000  FORMAT ('  READING NODE FILE')
1001  FORMAT (A1,29X,6I6)
1002  FORMAT (12I6)
1003  FORMAT (24I6)
      END
C
C ==================== SUBROUTINE AREAS ================================
      SUBROUTINE AREAS (HCODE,AID,ALINK,ACODE,ATTB
     + ,MXARCS,MXAREA)
C * read the DLG-3 optional area data array for subroutine poly
C
      CHARACTER*1 NAL
      DIMENSION HCODE(21)
      DIMENSION AID(MXAREA),ALINK(MXARCS,MXAREA),ACODE(6,MXAREA)
      DIMENSION AMAJOR(12),AMINOR(12),ATTB(2,MXAREA)
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 HCODE,ACODE,AID,ALINK,AMAJOR,AMINOR,ATTB
```

```
C *
      WRITE (SCREEN,1000)
      DO J=1,HCODE(15)
       READ (DLG,1001) NAL,AID(J),(ACODE(I,J),I=1,6)
       IF (NAL.NE.'A') RETURN
       IF (ACODE(2,J).GT.MXARCS) THEN
        WRITE (SCREEN,601) MXARCS
        STOP
       ENDIF
       IF (ACODE(2,J).NE.0) READ(DLG,1002) (ALINK(I,J),I=1,ACODE(2,J))
       IF (ACODE(4,J).NE.0) THEN
        READ (DLG,1003) (AMAJOR(I),AMINOR(I),I=1,ACODE(4,J))
C *
C * load into ATTB only the first attributes... this MUST be expanded
C * to correctly deal with every DLG attributes.
C *
        ATTB(1,J)=AMAJOR(1)
        ATTB(2,J)=AMINOR(1)
       ENDIF
      ENDDO
      RETURN
C *
 601  FORMAT (' FATAL ERROR: MORE THAN ',I5,' LINES / AREA ')
1000  FORMAT ('  READING AREA FILE')
1001  FORMAT (A1,I5,24X,6I6)
1002  FORMAT (12I6)
1003  FORMAT (24I6)
      END
C
C ==================== SUBROUTINE LINES ==============================
      SUBROUTINE LINES (HCODE,LCODE,X,Y,PARMS
     + ,MXPAIR,LID,MXLINE)
C * read the DLG-3 optional line data array for subroutine poly
C * dump the line work into annotation file
      CHARACTER*1 NAL
      DIMENSION HCODE(21),PARMS(4),LMAJOR(12),LMINOR(12)
      DIMENSION LID(MXLINE),LCODE(7,MXLINE)
      DIMENSION X(MXPAIR,MXLINE),Y(MXPAIR,MXLINE)
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 HCODE,LCODE,LID,LMAJOR,LMINOR
      REAL*4 X,Y,PARMS
C *
      WRITE (SCREEN,1000)
      DO J=1,HCODE(20)
       READ (DLG,1001) NAL,LID(J),(LCODE(I,J),I=1,7)
       IF (NAL.NE.'L') RETURN
       READ (DLG,1002,ERR=100) (X(I,J),Y(I,J),I=1,LCODE(5,J))
       IF (LCODE(6,J).NE.0)
     +  READ (DLG,1003) (LMAJOR(I),LMINOR(I),I=1,LCODE(6,J))
       DO I=1,LCODE(5,J)
        X(I,J)=(PARMS(1)*X(I,J))+(PARMS(2)*Y(I,J))+(PARMS(3))
        Y(I,J)=(PARMS(1)*Y(I,J))-(PARMS(2)*X(I,J))+(PARMS(4))
       ENDDO
        IF (LMAJOR(1).EQ.0) LMAJOR(1) = 1
        CALL SETATTB (OUTL,LMAJOR(1),LMINOR(1))
```

```
        WRITE (OUTL,1301)
     +   X(1,J),Y(1,J),LID(J),(X(I,J),Y(I,J),I=2,LCODE(5,J))
       ENDDO
       RETURN
100    WRITE (SCREEN,601) MXPAIR
       STOP
C *
 601   FORMAT (' FATAL ERROR: MORE THAN ',I5,' PAIRS / LINE')
1000   FORMAT ('  READING LINE FILE')
1001   FORMAT (A1,I5,4I6,12X,3I6)
1002   FORMAT ((6F12.2))
1003   FORMAT (24I6)
1301   FORMAT (2F12.2,' "',I5,'"',/,(2F12.2))
       END
C
C ==================== SUBROUTINE POLYS ==============================
       SUBROUTINE POLYS (HCODE,ACODE,LCODE,AID,ALINK,X,Y,ATTB
     + ,MXARCS,MXPAIR,MXLINE,MXAREA,MXNODE,LN,SN,EN)
C * structure the line and area data array
C * dump the polygons into annotation file
C *
       DIMENSION HCODE(21),LCODE(7,MXLINE)
       DIMENSION AID(MXAREA),ALINK(MXARCS,MXAREA),ACODE(6,MXAREA)
       DIMENSION X(MXPAIR,MXLINE),Y(MXPAIR,MXLINE)
       DIMENSION LN(MXNODE),SN(MXNODE),EN(MXNODE),ATTB(2,MXAREA)
       COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
       INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
       INTEGER*2 HCODE,LCODE,ACODE,AID,ALINK,ATTB
       INTEGER*2 PID,NL,LN,SN,EN,CN
       REAL*4 X,Y
C *
       WRITE (SCREEN,1200)
C * sort, set and dump the lines
       DO I=1,HCODE(15)
        PID = AID(I)
        NL = ACODE(2,PID)
         DO J=1,NL
         K = ABS(ALINK(J,PID))
         LN(J) = ALINK(J,PID)
         SN(J) = LCODE(1,K)
         EN(J) = LCODE(2,K)
        ENDDO
C * check if first polygon is an island
       IF (LN(1).EQ.0) THEN
        DO J=2,NL
         LN(J-1) = LN(J)
         SN(J-1) = SN(J)
         EN(J-1) = EN(J)
        ENDDO
        NL = NL - 1
       ENDIF
C * sort the lines in line number (ln) array
C * the idea for how to select and sort nodes came from
C * Gary Selner of the Office of Mineral Resources, U.S.G.S.
       J = 1
       CN = EN(J)
```

```
1234    DO K = J+1,NL
         IF ((SN(K).EQ.CN).OR.(EN(K).EQ.CN)) THEN
          L       = LN(J+1)
          LN(J+1) = LN(K)
          LN(K)   = L
          L       = SN(J+1)
          SN(J+1) = SN(K)
          SN(K)   = L
          L       = EN(J+1)
          EN(J+1) = EN(K)
          EN(K)   = L
          IF (SN(J+1).EQ.CN) THEN
           CN = EN(J+1)
          ELSE
           CN = SN(J+1)
          ENDIF
          J = J+1
          GOTO 1234
         ENDIF
        ENDDO
C * set do loop counters to dump each polygon in sequence.
C *
C * special case: no attributes
        IF (ATTB(1,PID).GT.0) THEN
        CALL SETATTB (OUTA,ATTB(1,PID),ATTB(2,PID))
        DO J=1,NL
         K = ABS(LN(J))
C * special case: line number zero, (island flag)
         IF (K.NE.0) THEN
C * special case: 1st arc of polygon
         IF (J.EQ.1) THEN
          WRITE (OUTA,1201) X(1,K),Y(1,K),PID
           IS = 2
           IE = LCODE(5,K)
           ID = 1
           CN = EN(J)
C * special case: 1st arc of island
         ELSEIF (LN(J-1).EQ.0) THEN
          IF (LN(J).GT.0) THEN
           IS = 1
           IE = LCODE(5,K)
           ID = 1
           CN = EN(J)
          ELSE
           IS = LCODE(5,K)
           IE = 1
           ID = - 1
           CN = SN(J)
          ENDIF
C * normal case: positive arc
         ELSEIF (SN(J).EQ.CN) THEN
          IS = 1
          IE = LCODE(5,K)
          ID = 1
          CN = EN(J)
C * normal case: negative arc
```

```
        ELSEIF (EN(J).EQ.CN) THEN
         IS = LCODE(5,K)
         IE = 1
         ID = - 1
         CN = SN(J)
        ELSE
         IS = LCODE(5,K)
         IE = 1
         ID = - 1
         CN = SN(J)
        ENDIF
C *
C * dump the x,y pairs for each line
        DO L = IS,IE,ID
          WRITE (OUTA,1202) X(L,K),Y(L,K)
         ENDDO
        ELSE
C * flag start of island
         WRITE (OUTA,1203)
        ENDIF
       ENDDO
      ENDIF
      ENDDO
      RETURN
C *
1200  FORMAT ('  BUILDING POLYGONS')
1201  FORMAT (2F12.2,' "',I5,'"')
1202  FORMAT (2F12.2)
1203  FORMAT (' .1000000E21 .1000000E21')
      END
C
C ==================== SUBROUTINE TEXT ============================
      SUBROUTINE TEXT (DESCRIBE,NAMES,YEAR,SCALE,NETNAM,HCODE)
C * print general text information about the dlg sheet
      CHARACTER DESCRIBE*72,NAMES*40,YEAR*10,NETNAM*20
      DIMENSION HCODE(21)
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*4 SCALE
      INTEGER*2 HCODE
C *
      WRITE (SCREEN,600) DESCRIBE
      WRITE (SCREEN,601) NAMES,YEAR,SCALE
      WRITE (SCREEN,602) NETNAM
      WRITE (SCREEN,603) HCODE(11),HCODE(15),HCODE(20)
C *
      IF (HCODE(2) .EQ. 0) WRITE (SCREEN,620)
      IF (HCODE(2) .EQ. 1) WRITE (SCREEN,621) HCODE(3)
      IF (HCODE(2) .EQ. 2) WRITE (SCREEN,622) HCODE(3)
      IF (HCODE(2) .EQ. 3) WRITE (SCREEN,623)
      RETURN
C *
 600  FORMAT (/,1X,A72,/)
 601  FORMAT (1X,A40,/,' PRODUCED ',A10,/,' AT A SCALE OF 1 :'I8)
 602  FORMAT (' DLG COVERAGE IS ',A20)
 603  FORMAT (/,' NUMBER OF NODES - ',I6,/,' NUMBER OF AREAS - ',I6,
     + /,' NUMBER OF LINES - ',I6)
```

```
C *
 620  FORMAT (' NO PROJECTION')
 621  FORMAT (' UTM PROJECTION, ZONE = ',I6)
 622  FORMAT (' STATE PLANE, ZONE = ',I6)
 623  FORMAT (' ALBERS CONICAL EQUAL AREA PROJECTION')
      END
C
C ==================== SUBROUTINE OPENFILE ==============================
      SUBROUTINE OPENFILE (IOFILE)
C * open input and outoput files
C * openfile was written on a dec vax/vms version 4.6
      CHARACTER INFILE*75,OUTFILE*75
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IOFILE
      LOGICAL FOUND
      FOUND=.FALSE.
C *
      IF (IOFILE.EQ.DLG) THEN
       WRITE (SCREEN,600)
1      WRITE (SCREEN,601)
       READ(IN,501,ERR=1) INFILE
C * PANIC CHECK
       IF ((INFILE.EQ.'EXIT').OR.(INFILE.EQ.'exit')) STOP
       IF ((INFILE.EQ.'QUIT').OR.(INFILE.EQ.'quit')) STOP
       INQUIRE(FILE=INFILE,EXIST=FOUND)
       IF (FOUND) THEN
        REWIND (UNIT=DLG)
        OPEN (DLG,FILE=INFILE,STATUS='OLD',READONLY)
       ELSE
        WRITE (SCREEN,602) INFILE
        GOTO 1
       ENDIF
C *
      ELSE
       IF (IOFILE .EQ. OUTN) WRITE(SCREEN,603)
       IF (IOFILE .EQ. OUTA) WRITE(SCREEN,604)
       IF (IOFILE .EQ. OUTL) WRITE(SCREEN,605)
       READ(IN,501) OUTFILE
       OPEN (IOFILE,FILE=OUTFILE,STATUS='NEW')
      ENDIF
      RETURN
C *
 501  FORMAT(A75)
 600  FORMAT (/' DLG3 OPTIONAL to ISM Annotation Conversion Routine ')
 601  FORMAT (/' Name of DLG3 Optional input file ? ',$)
 602  FORMAT (/' ERROR:  FILE NOT FOUND - ',A75)
 603  FORMAT (/' Name of ISM node output file? ',$)
 604  FORMAT (/' Name of ISM poly output file? ',$)
 605  FORMAT (/' Name of ISM line output file? ',$)
      END
C
C ==================== SUBROUTINE SETATTB ==============================
      SUBROUTINE SETATTB (IOFILE,MAJOR,MINOR)
      INTEGER*2 IOFILE,MAJOR,MINOR
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
```

```
C
C Considerable work should go into improving this attribute system.  At
C this time only the simplest features are matched and converted to ISM
C format.  For instance, the DLG major code for hydrography is 50, all
C code 50 polygons will be blue with the same pattern.  No hydrography
C minor codes are tested, except minor code 610 (Intermitttent).
C Cavet Emptor
C The major minor pairs are passed to these routines, however almost
C none of the minor attributes are used.
C
C ------------------------------------------------------------------ C
C MAJOR CLASS                                                        C
C 020 HYPSOGRAPHY                                                    C
C 050 HYDROGRAPHY                                                    C
C 070 SURFACE COVER                                                  C
C 080 NONVEGETATIVE SURFACE FEATURES                                 C
C 090 BOUNDRIES                                                      C
C 150 SURVEY CONTROL AND MARKERS                                     C
C 170 TRANSPORTATION, ROADS AND TRAILS                               C
C 180 TRANSPORTATION, RAILROADS                                      C
C 190 TRANSPORTATION, PIPELINES, TRANSMISSION AND MISC. FEATURES     C
C 200 OTHER SIGNIFICANT MANMADE STRUCTURES                           C
C 300 U.S. PUBLIC LAND SURVEY SYSTEM                                 C
C 500 GEOLOGY                                                        C
C ------------------------------------------------------------------ C
C
C  PEN   COLOR
C   1    BLACK
C   2    GREEN
C   3    BLUE
C   4    RED
C   5    BROWN
C   6    VIOLET
C   7    YELLOW
C   8    MAGENTA
C
C  LINE         TYPE
C    1  _____________________
C    2  _____________________ BOLD
C    3  _ _ _ _ _ _ _ _ _ _ _ _
C    4  __ __ __ __ __ __ __ __
C    5  ___ ___ ___ ___ ___ ___
C    6  ___ _ ___ _ ___ _ ___ _
C    7  ____ _ ____ _ ____ _ __
C    8  ___ _ _ ___ _ _ ___ _ _
C    9  ___ __ __ ___ __ __ ____
C    10 +-+-+-+-+-+-+-+-+-+-+-+-
C
C    POLYGON FILL TYPE
C
C   1    ---   HORIZONTAL BARS
C
C   2    |  |  VERTICAL BARS
C
C   3    \\\\  LEFT SLANT BARS
C
```

```
C   4   ////  RIGHT SLANT BARS
C
C   5   ++++  CROSS-HATCH
C
C   6   /\/\ SLANT CROSS-HATCH
C * NODES
      IF (IOFILE.EQ.OUTN) THEN
       CALL SETNODE (IOFILE,MAJOR,MINOR)
C * LINES
      ELSEIF (IOFILE.EQ.OUTL) THEN
       CALL SETLINE (IOFILE,MAJOR,MINOR)
C * AREAS
      ELSE
       CALL SETAREA (IOFILE,MAJOR,MINOR)
      ENDIF
      RETURN
      END
C
C ==================== SUBROUTINE SETNODE ============================
      SUBROUTINE SETNODE (IOFILE,MAJOR,MINOR)
      INTEGER*2 IOFILE,MAJOR,MINOR
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
C * NODES
      RETURN
      END
C
C ==================== SUBROUTINE SETLINE ============================
      SUBROUTINE SETLINE (IOFILE,MAJOR,MINOR)
      INTEGER*2 IOFILE,MAJOR,MINOR
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
C *
C zero length line features are not handled YET
C * LINES
C 000 NO FEATURE
      IF ((MAJOR.EQ.' ').OR.(MAJOR.LT. 20).OR.(MAJOR.GE.600)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,101)
C 020 HYPSOGRAPHY
      ELSEIF ((MAJOR.GE. 20).AND.(MAJOR.LT. 29)) THEN
       WRITE (OUTL,  5)
       WRITE (OUTL,101)
C 050 HYDROGRAPHY
      ELSEIF (MAJOR.EQ. 50) THEN
       WRITE (OUTL,  3)
       IF (MINOR.EQ.610) WRITE (OUTL,107)
       IF (MINOR.NE.610) WRITE (OUTL,101)
C 070 SURFACE COVER
      ELSEIF ((MAJOR.GE. 70).AND.(MAJOR.LT. 79)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,101)
C 080 NONVEGETATIVE SURFACE FEATURES
      ELSEIF ((MAJOR.GE. 80).AND.(MAJOR.LT. 89)) THEN
C 090 BOUNDRIES
```

```
      ELSEIF ((MAJOR.GE. 90).AND.(MAJOR.LT. 99)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,101)
C 150 SURVEY CONTROL AND MARKERS
      ELSEIF ((MAJOR.GE.150).AND.(MAJOR.LT.159)) THEN
C 170 TRANSPORTATION, ROADS AND TRAILS
      ELSEIF ((MAJOR.GE.170).AND.(MAJOR.LT.179)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,101)
C 180 TRANSPORTATION, RAILROADS
      ELSEIF ((MAJOR.GE.180).AND.(MAJOR.LT.189)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,110)
C 190 TRANSPORTATION, PIPELINES, TRANSMISSION AND MISC. FEATURES
      ELSEIF ((MAJOR.GE.190).AND.(MAJOR.LT.199)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,101)
C 200 OTHER SIGNIFICANT MANMADE STRUCTURES
      ELSEIF ((MAJOR.GE.200).AND.(MAJOR.LT.209)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,101)
C 300 U.S. PUBLIC LAND SURVEY SYSTEM
      ELSEIF ((MAJOR.GE.300).AND.(MAJOR.LT.309)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,101)
c expand for minor
C 500 GEOLOGY
      ELSEIF ((MAJOR.GE.500).AND.(MAJOR.LT.599)) THEN
       WRITE (OUTL,  1)
       WRITE (OUTL,101)
c expand for minor
C 000 NO FEATURE
      ELSE
       RETURN
      ENDIF
      RETURN
C *
1     FORMAT ('SETPEN  1,   2, 1')
2     FORMAT ('SETPEN  2,   2, 1')
3     FORMAT ('SETPEN  3,   2, 1')
4     FORMAT ('SETPEN  4,   2, 1')
5     FORMAT ('SETPEN  5,   2, 1')
6     FORMAT ('SETPEN  6,   2, 1')
7     FORMAT ('SETPEN  7,   2, 1')
8     FORMAT ('SETPEN  8,   2, 1')
101   FORMAT ('SRFLNE  1,   1, 0')
102   FORMAT ('SRFLNE  2,   1, 0')
103   FORMAT ('SRFLNE  3,   1, 0')
104   FORMAT ('SRFLNE  4,   1, 0')
105   FORMAT ('SRFLNE  5,   1, 0')
106   FORMAT ('SRFLNE  6,   1, 0')
107   FORMAT ('SRFLNE  7,   1, 0')
108   FORMAT ('SRFLNE  8,   1, 0')
109   FORMAT ('SRFLNE  9,   1, 0')
110   FORMAT ('SRFLNE 10,   1, 0')
C *
      END
```

```
C
C ==================== SUBROUTINE SETAREA ============================
      SUBROUTINE SETAREA (IOFILE,MAJOR,MINOR)
      INTEGER*2 IOFILE,MAJOR,MINOR
      COMMON /CHANNEL/ IN,SCREEN,DLG,OUTN,OUTA,OUTL
      INTEGER*2 IN,SCREEN,DLG,OUTN,OUTA,OUTL
C * AREAS
C 000 NO FEATURE
       IF (MAJOR.EQ.' ') RETURN
       IF (MAJOR.LT. 20) RETURN
       IF (MAJOR.GE.600) RETURN
C 020 HYPSOGRAPHY
      IF ((MAJOR.GE. 20).AND.(MAJOR.LT. 29)) THEN
C 050 HYDROGRAPHY
      ELSEIF ((MAJOR.GE. 50).AND.(MAJOR.LT. 59)) THEN
        WRITE (OUTA,  3)
        WRITE (OUTA,205)
C 070 SURFACE COVER
      ELSEIF ((MAJOR.GE. 70).AND.(MAJOR.LT. 79)) THEN
       WRITE (OUTA,  2)
       WRITE (OUTA,210)
C 080 NONVEGETATIVE SURFACE FEATURES
      ELSEIF ((MAJOR.GE. 80).AND.(MAJOR.LT. 89)) THEN
C 090 BOUNDRIES
      ELSEIF ((MAJOR.GE. 90).AND.(MAJOR.LT. 99)) THEN
       WRITE (OUTA,  1)
       WRITE (OUTA, 10)
C 150 SURVEY CONTROL AND MARKERS
      ELSEIF ((MAJOR.GE.150).AND.(MAJOR.LT.159)) THEN
C 170 TRANSPORTATION, ROADS AND TRAILS
      ELSEIF ((MAJOR.GE.170).AND.(MAJOR.LT.179)) THEN
C 180 TRANSPORTATION, RAILROADS
      ELSEIF ((MAJOR.GE.180).AND.(MAJOR.LT.189)) THEN
C 190 TRANSPORTATION, PIPELINES, TRANSMISSION AND MISC. FEATURES
      ELSEIF ((MAJOR.GE.190).AND.(MAJOR.LT.199)) THEN
C 200 OTHER SIGNIFICANT MANMADE STRUCTURES
      ELSEIF ((MAJOR.GE.200).AND.(MAJOR.LT.209)) THEN
       WRITE (OUTA,  1)
       WRITE (OUTA, 10)
C 300 U.S. PUBLIC LAND SURVEY SYSTEM
      ELSEIF ((MAJOR.GE.300).AND.(MAJOR.LT.309)) THEN
       WRITE (OUTA,  1)
       WRITE (OUTA, 10)
C 500 GEOLOGY
      ELSEIF ((MAJOR.GE.500).AND.(MAJOR.LT.599)) THEN
C SET POLY FILL COLOR
C
C if major gt 500 then pattern from ARC/INFO arcplot symbols
       ICOLOR = MINOR/4.0
       COLOR  = MINOR/4.0
       COLOR = COLOR - ICOLOR
       IF (COLOR.EQ.0.25) WRITE (OUTA,1)
       IF (COLOR.EQ.0.50) WRITE (OUTA,4)
       IF (COLOR.EQ.0.75) WRITE (OUTA,2)
       IF (COLOR.EQ.0.0)  WRITE (OUTA,3)
C *
```

```
       IF ((MINOR.GE. 1).AND.(MINOR.LT. 5)) WRITE (OUTA,601)
       IF ((MINOR.GE. 5).AND.(MINOR.LT. 9)) WRITE (OUTA,115)
       IF ((MINOR.GE. 9).AND.(MINOR.LT.13)) WRITE (OUTA,110)
       IF ((MINOR.GE.13).AND.(MINOR.LT.17)) WRITE (OUTA,105)
       IF ((MINOR.GE.17).AND.(MINOR.LT.21)) WRITE (OUTA,215)
       IF ((MINOR.GE.21).AND.(MINOR.LT.25)) WRITE (OUTA,210)
       IF ((MINOR.GE.25).AND.(MINOR.LT.29)) WRITE (OUTA,205)
       IF ((MINOR.GE.29).AND.(MINOR.LT.33)) WRITE (OUTA,315)
       IF ((MINOR.GE.33).AND.(MINOR.LT.37)) WRITE (OUTA,310)
       IF ((MINOR.GE.37).AND.(MINOR.LT.41)) WRITE (OUTA,305)
       IF ((MINOR.GE.41).AND.(MINOR.LT.45)) WRITE (OUTA,415)
       IF ((MINOR.GE.45).AND.(MINOR.LT.49)) WRITE (OUTA,410)
       IF ((MINOR.GE.49).AND.(MINOR.LT.53)) WRITE (OUTA,405)
       IF ((MINOR.GE.53).AND.(MINOR.LT.57)) WRITE (OUTA,515)
       IF ((MINOR.GE.57).AND.(MINOR.LT.61)) WRITE (OUTA,510)
       IF ((MINOR.GE.61).AND.(MINOR.LT.65)) WRITE (OUTA,505)
       IF ((MINOR.GE.65).AND.(MINOR.LT.69)) WRITE (OUTA,615)
       IF ((MINOR.GE.69).AND.(MINOR.LT.73)) WRITE (OUTA,610)
       IF ((MINOR.GE.73).AND.(MINOR.LT.77)) WRITE (OUTA,605)
       IF ((MINOR.GE.77).AND.(MINOR.LT.81)) WRITE (OUTA,611)
       IF ((MINOR.GE.81).AND.(MINOR.LT.85)) WRITE (OUTA,608)
       IF ((MINOR.GE.85).AND.(MINOR.LT.89)) WRITE (OUTA,603)
       IF ((MINOR.GE.89).AND.(MINOR.LT.93)) WRITE (OUTA,620)
       IF ((MINOR.GE.93).AND.(MINOR.LT.97)) WRITE (OUTA,609)
       IF ((MINOR.GE.97).AND.(MINOR.LT.101)) WRITE (OUTA,612)
      ELSE
       RETURN
      ENDIF
      RETURN
C *
1     FORMAT ('SETPEN  1,   2, 1')
2     FORMAT ('SETPEN  2,   2, 1')
3     FORMAT ('SETPEN  3,   2, 1')
4     FORMAT ('SETPEN  4,   2, 1')
5     FORMAT ('SETPEN  5,   2, 1')
6     FORMAT ('SETPEN  6,   2, 1')
7     FORMAT ('SETPEN  7,   2, 1')
8     FORMAT ('SETPEN  8,   2, 1')
C *
 10   FORMAT ('SRFPLY  0,   1, 0')
105   FORMAT ('SRFPLY  1,  .5, 0')
110   FORMAT ('SRFPLY  1, 1.0, 0')
115   FORMAT ('SRFPLY  1, 1.5, 0')
120   FORMAT ('SRFPLY  1, 2.0, 0')
130   FORMAT ('SRFPLY  1, 3.0, 0')
C *
205   FORMAT ('SRFPLY  2,  .5, 0')
210   FORMAT ('SRFPLY  2, 1.0, 0')
215   FORMAT ('SRFPLY  2, 1.5, 0')
220   FORMAT ('SRFPLY  2, 2.0, 0')
230   FORMAT ('SRFPLY  2, 3.0, 0')
C *
305   FORMAT ('SRFPLY  3,  .5, 0')
310   FORMAT ('SRFPLY  3, 1.0, 0')
315   FORMAT ('SRFPLY  3, 1.5, 0')
320   FORMAT ('SRFPLY  3, 2.0, 0')
330   FORMAT ('SRFPLY  3, 3.0, 0')
```

```
C *
405   FORMAT ('SRFPLY  4,  .5, 0')
410   FORMAT ('SRFPLY  4, 1.0, 0')
415   FORMAT ('SRFPLY  4, 1.5, 0')
420   FORMAT ('SRFPLY  4, 2.0, 0')
430   FORMAT ('SRFPLY  4, 3.0, 0')
C *
505   FORMAT ('SRFPLY  5,  .5, 0')
510   FORMAT ('SRFPLY  5, 1.0, 0')
515   FORMAT ('SRFPLY  5, 1.5, 0')
520   FORMAT ('SRFPLY  5, 2.0, 0')
530   FORMAT ('SRFPLY  5, 3.0, 0')
C *
601   FORMAT ('SRFPLY  6,  .1, 0')
603   FORMAT ('SRFPLY  6,  .3, 0')
605   FORMAT ('SRFPLY  6,  .5, 0')
608   FORMAT ('SRFPLY  6,  .8, 0')
609   FORMAT ('SRFPLY  6,  .9, 0')
610   FORMAT ('SRFPLY  6, 1.0, 0')
611   FORMAT ('SRFPLY  6, 1.1, 0')
612   FORMAT ('SRFPLY  6, 1.2, 0')
615   FORMAT ('SRFPLY  6, 1.5, 0')
620   FORMAT ('SRFPLY  6, 2.0, 0')
630   FORMAT ('SRFPLY  6, 3.0, 0')
      END
C ========================== FINI ================================
```