TUTORIAL

Importing TIGER/Line Maps and STF1 Census Data to GRASS

by

Dr. James Hinthorne
GIS Laboratory
Central Washington University
Ellensburg, WA 98926
September 1992

Preface

This document addresses the use of a suite of GRASS programs to import and utilize the U.S. Census Bureau's TIGER/Line files, and the STF1 and PL94 Population and Housing demographic files. The programs and procedures described in this tutorial may be used in GRASS "locations" with UTM or latitude-longitude coordinate systems.

Programs discussed in this tutorial largely supercede the TIGER/Line import processes originally distributed with GRASS 4.0, although some of the original programs may be preferred for a few kinds of operations. The original programs (discussed in the tutorial "TIGER in GRASS with the RIM Database Manager") required the use of the *RIM* data base management system through the *v.db.rim* program and Shell-scripts (such as tig.rim.sh) which ran these programs indirectly. The programs discussed here do not require the use of *RIM* or *v.db.rim*, and do not require building a "Master" data base and vector map (done during the execution of v.in.tig.rim) prior to generating the TIGER/Line maps of interest in GRASS. Because *v.in.tig.basic* does not build a database and "Master" vector file, it runs much faster and uses less memory then *v.in.tig.rim*. The user should read through both tutorials and decide which version of the programs best meets his/her needs.

Organization of the Tutorial

Section I of this tutorial describes the TIGER/Line files and the use of the TIGER import programs, *v.in.tig.basic* and *v.in.tig.lndmk*, to make GRASS vector and site maps.

Section II describes the Census Demographic "STF1" and "PL94" data files, their extraction from tape and CD-ROM distribution media with *m.in.stf1.tape* and *m.in.stf1.db3*, and a program to integrate those data with the TIGER-based vector maps and as well as produce GRASS site maps and tabular reports of the demographic information, *s.in.stf1*.

Section III contains examples and information on the overall process of integrating the TIGER/Line and demographic data sets to create labelled vector maps.

Section IV discusses related programs and special cases.

Most of the examples in this tutorial use the Census data for Lawrence County, South Dakota, in which the *spearfish* area is located. Necessary parts of this data are distributed with GRASS 4.1 or can be obtained through the GRASS *ftp* site.

Section I. TIGER/Line Data and GRASS Programs

The TIGER/Line Data Files

The user should obtain the following document so that information on file formats and contents will be at hand for the TIGER/Line data:

TIGER/Line Census Files, 1990 Technical Documentation, prepared by Bureau of the Census.--Washington: The Bureau, 1991. This can be obtained by printing the files in the "document" directory on the TIGER CD-ROM distribution media, or from:

Data User Services Division
TIGER Staff
Bureau of the Census
Washington, D.C. 20233
Phone: (301) 763-1580

The TIGER/Line file tapes of data for each county, or CD-ROMs for each state can be obtained from:

Data User Services Division
Customer Services (Order Desk)
Bureau of the Census
Washington, D.C. 20233
Phone: (301)763-4100

TIGER File Names

Twelve (12) types of TIGER files are currently distributed for each county in the United States. TIGER files are named according to the format TGRssccc.Fvn where ss is the two digit FIPS State Code, the ccc is the three digit FIPS County Code, v is the Version number of the file (usually 4, but 5 should be released in 1993), and n is the File Type code. The twelve File Type codes for the version 4 files are 1,...,8,A,I,P and R. Examples of TIGER file names: TGR53037.F41 for File Type 1 for Kittitas County, Washington; TGR46081.F42 for File Type 2 for Lawrence County, South Dakota (which contains the "Spearfish" sample data base area). Depending on your operating system, the upper case letters in the file names may be converted automatically to lower case. [NOTE: Versions of the TIGER/Line files older than version 4 used several different naming conventions; they can usually be deciphered by careful inspection. Use of older versions is strongly discouraged due to their known data inaccuracies; they did not receive careful checking. Also, some of them were distributed in file formats incompatible with this set of programs.]

Only TIGER File Types 1 and 2 are used by the *v.in.tig.basic* import program to build most of the commonly used GRASS4.0 vector files, including those that will be later labelled with demographic data. Therefore, you only need access to File Types 1 and 2. On most systems TIGER files obtained on CD-ROM can be used directly from that distribution medium (the CD-ROM file system must be "mounted"), but much greater speed may be obtained by first copying the files to disk. Files for groups of counties (usually adjacent) may be concatenated using UNIX commands prior to making maps providing you have adequate disk space; typical files are 2 to 20 MB in size.

TIGER/Line "Landmark" records (Type 7) are converted to GRASS maps using the *v.in.tig.lndmk* program; seven TIGER/Line File Types are required: 1, 2, 7, 8, A, I and P. Landmark features may be point features (yeilding site maps), or area (polygon) features from which vector maps are constructed. Again, the files can be used directly from CD-ROM, or copied from tape or CD-ROM to disk.

TIGER/Line File Types 1, 2 and 7

Type 1 is the "Basic Data Record" (or Individual Feature Segment Record). It contains a single 228-character record for each unique line segment, and each record has 42 fields of information. Some important fields for creating vector files are: TLID ("TIGER/LINE ID," a unique 9 digit record ID number); and the fields giving the end point coordinates (nodes) of the line segment, FRLONG, FRLAT, TOLONG and TOLAT, whose values are given as integers with 6 decimal places implied. Other attributes or fields contain information which can be used to select subsets of the records to build maps of specific feature types (*e.g.*, permanent streams) or census counting units (*e.g.*, tracts, block groups, blocks).

The CFCC field in columns 56-58 is the Census Feature Classification Code for each line segment. The list of "CFCCs" is important and is reproduced in Appendix B. Review this list to determine the kinds of feature codes that have been assigned to the TIGER line segments.

An important concept to keep in mind is that the line segments represented by the Type 1 records never cross one another; they begin and end (have nodes) at all line intersection points, even if the intersecting lines represent totally different feature types. This results in a large number of line segments (Type 1 records) for a county; for example, 9,000 records for Lawrence Co., SD; 26,000 for Kittitas Co., WA; and 81,000 for Pima Co., AZ; many are larger. Also, all coordinate information in TIGER files is in latitude-longitude; the user's programs must convert to other coordinate systems, if necessary; the programs described in this tutorial do the conversion to UTM coordinates, if necessary.

Type 2 records are known as "Shape Coordinate Points" records and are 208 characters long. These are cross-referenced to Type 1 records by the TLID field, and contain up to 10 pairs of latitude-longitude coordinates for the intermediate points ("shape points") along each line segment, if any. If a line segment is straight, there will not be any Type 2 records for that line. (The only way to determine this is to search the entire Type 2 file!) Multiple Type 2 records may exist for one Type 1 record if more than 10 shape points are used to describe the line. There are approximately as many Type 2 records for a county as Type 1.

Type 7 records are the Landmark Feature records. These are point and polygon items the Census Bureau wished to include, but which cannot be specified within the structure of the Types 1 and 2 records. For many counties there are only a few (less than 100) Type 7 records. Types 8, A, I and P contain records which direct the construction of Landmark polygons from the Types 1 and 2 line records.

The documentation mentioned above gives the details of all the record types, as well as the fields by which they are cross-referenced for various useful purposes. Additional record types will be added for the version 5 TIGER/Line files expected in 1993.

Using the GRASS TIGER/Line Programs

The general process of importing a TIGER/Line map into GRASS is a simple two-step procedure. First, the Type 1 records of the line segments representing the desired features must be identified and (usually) extracted from the county-wide file. This first step is done as a separate operation prior to using *v.in.tig.basic*, but is automatically done by the *v.in.tig.lndmk* program. Second, the selected Type 1 lines (pairs of nodes) must be combined with the appropriate Type 2 shape points and written to the proper directories as a GRASS map.

Extracting Type 1 Records

Several methods exist for extracting a desired subset of Type 1 records from the original CD-ROM or

tape sources. The method you use will depend on your needs and experience. One way is not necessarily better than the others. This process is generally more demanding on the GRASS user than actually making the maps. For most of the methods you must know the location of the different fields within the Type 1 Basic Data Record; the Census Bureau's documentation clearly presents this. Very often selected subsets will be repeatedly used so they should be placed in a directory with file names indicating the contents for easy reference later. All the methods described below can extract directly from tape or CD-ROM as well as from disk files.

Method 1

Unix text filtering programs such as *grep* or *awk* can be used to select a subset of Type 1 lines by specify the original file as input, supplying appropriate **column** selection criteria, and directing the selected lines to a new file.

[*awk* is the most versatile of text filters and worth the time to learn if complex selections need to be accomplished. A useful publication is **The AWK Programming Language** by Aho, Kernighan and Weinberger; Addison-Wesley Publishing Co., 1988; ISBN 0-201-07981-X.]

Examples and comments:

```
# Extract the hydrography lines with a feature name of "Morrison".
# Columns 56-58 contain the Census Feature Class Codes (CFCC) and
#  hydrography line Codes begin with "H"; the FENAME field is in
#  columns 20-49.  Note: feature names are often missing in Type 1 records!
awk 'substr($0,56,1)=="H" && substr($0,20,8)=="Morrison"' < .../TGR46081.F41 > Morrison.River

# extract the county boundary line segments
# column 16 = 1 for a county boundary, blank otherwise.
awk 'substr($0,16,1)=="1"'
        or
grep '^.............1' < .../TGR46081.F41 > Lawrence.Co
```

Method 2

The new program *m.in.stf1.tape* can be used to select a subset of Type 1 lines (or lines from any file). Column numbers and their contents are specified with a special syntax. This program may have to be used if the Type 1 lines from the original sources contain RETURN (^M), as well as LINE FEED (^J), characters which upset some of the Unix utility programs.

Example:

```
#  Select all primary roads, CFCC A10-A19 in columns 56-58, except
#   those that are undivided with a rail line in the center (A14).
m.in.stf1.tape 56=A1 56!=A14 < .../TGR46081.F41 > roads.primary
```

(".../" represents an arbitray directory pathname.)

Method 3

Use the "line map" option in *v.in.tig.lndmk* and select Type 1 records according to their CFCC field values and/or the FENAME (feature name) field contents. *v.in.tig.basic* is automatically run by this program to create the requested map.

Method 4

The GRASS script GenVect may be used to build a special RIM data base and then the program v.db.rim may be used to extract the TLID value for lines selected with an SQL-like query. This was the only method available in the original GRASS 4.0 release, but is largely obsolete now because it is more convoluted, uses more disk space and is slower than the methods above. Refer to the "Tiger in Grass with the RIM Database Manager" tutorial for more information about database option. The GenVect script may have value as a model for importing Tiger data into other data base management systems.

Using v.in.tig.basic

The typical use of *v.in.tig.basic* to construct a GRASS vector file from the nodes and shape points in Type 1 and 2 files is:

```
v.in.tig.basic t1=subset.A.F41 t2=TGR46081.F42 out=newmap.A
```

where subset.A.F41 is a file containing a desired subset of Type 1 lines, TGR46081.F42 is the entire Type 2 file for a county, and A is the name of the vector map to be created in your current mapset. The Type 2 file is accessed randomly (after a pointer table is built) so its size has little effect on the processing speed. The TLID (TIGER/Line ID number) is attached as the attribute for each line; and each line is designated to be an "area boundary" in the attribute so that v.support can build areas, if desired. No area attributes can be generated simply from the Type 1 records, although complex scripts could be developed.

The current GRASS region is ignored by *v.in.tig.basic* in creating a vector map; and an artificially large scale (1:1000) is assigned to the vector map header so that the snapping threshold will be small and relatively few attributes will fail to be assigned by *v.support*, if the map's polygons are later labelled.

The *out=* parameter specifies the name of the new vector map to be created in your current mapset.

If the desired Type 1 record numbers (TLID field) have been extracted and listed in a separate text file (using *awk*, *grep* | *colrm*, an external data base manager, etc.), that file can be named on the *v.in.tig.basic* command line and only the listed records will be processed from the named Type 1 file. (*v.in.tig.lndmk* uses this method.) Generally the file containing all the Type 1 records for a county is given as the *t1=* parameter, but this file can be a subset also if it contains all the records listed in the *tlid=* file. In general, processing will be faster if the file list of needed TLIDs is sorted in ascending order. An example using an input file with TLID values is:

```
v.in.tig.basic t1=TGR46081.F41 t2=TGR46081.F42 tlid=$HOME/list.a out=A
```

Optional parameters for UTM spheroid and zone can be specified if the defaults are not desired. The default spheroid is "clark66."

Four command line flags can be used with this program:

-q (quiet) prevents printing of informative messages, including a "." for each record processed into the new map. Using this flag can significantly speed the process, especially when operating from a terminal in a "windows" environment.

-t causes the GRASS "topology" or "dig_plus" file to be automatically built for the new map (this can be slow for a large map).

-*p* causes a new Type 2 file pointer table to be created. This table is stored in the $LOCATION/.tmp directory and is retained when you exit GRASS (so that it does not have to be rebuilt when *v.in.tig.basic* is run again). The pointer file has an eight- or nine-digit file name, equivalent to the TLID number of the first record in the Type 2 file. The pointer file is automatically constructed if it cannot be found; this can take from 15 seconds to a few minutes.

-*v* is provided for compatibility with users of the GRASS 4.0 TIGER programs using *v.db.rim*. It causes a text file to be written in the current directory that can be processed using ".input filename" in *v.db.rim* after the "Master" county vector map is built. This file contains the representative point coordinates and other information to update the data base record for each Type 1 line.

Using v.in.tig.lndmk

Point and area (polygon) "Landmark" features in the Census TIGER/Line files are recorded in the Type 7 file for a county. For the most part, these records have a Dnn Census Feature Classification Code (CFCC), which are titled "Landmarks" in the documentation. However, other major CFCC categories, notably hydrography (Hnn), are referenced in the Type 7 records. In addition to being able to extract point and area Landmark features to GRASS maps, this program can directly access the Type 1 (line) file records and extract those which match specified CFCCs; this makes this program an alternative method of selecting subsets of Type 1 records.

Two modes of use for this program are provided. The "interactive," menu-driven, mode is entered by specifying the -*i* flag on the command line. The "command" mode may take input commands from the keyboard, or from stdin (with -*s* flag) or from a file named in the *input*= parameter and thus may be used to process "script" files.

Only one command line parameter is required, the path/name of the TIGER Type 1 file. The other necessary file types are assumed to follow the standard naming conventions and be in the same directory as the Type 1 file (on disk or CD-ROM); if this is not so, all the required files must be explicitly defined in command line parameters. Type 7 is the only necessary file if just "point" Landmark features are being extracted. Types 1 and 2 are required for "line" maps; and Types 1, 2, 7, 8, A, I, and P are required for "area" Landmark maps.

The steps in making a map with *v.in.lndmk* are:

1. Select the kind of map (point, area, or line)
2. Optionally review the available CFCCs
3. Enter the desired CFCC and/or the feature name strings

After choosing the type of map to be made (point, area or line), the program can list the available Landmark features by CFCC which are present in the Type 7 file, show the list on the screen, and send the list to a file. After reviewing the available CFCCs, you select a subset for inclusion in the map, enters up to 10 strings to match feature names, and waits for the results. In the command mode the CFCCs may be specified in groups, such as "D5" (D51-D59) or "H" (all hydrography records).

The character strings you may enter to match the Landmark names (the FENAME field in Type 1 records when extracting a line map) may be up to 30 characters long and they may be designated as exact case matching by having a "!" character begin a string. These user-specified strings are checked as substrings of the name field for each record, meaning that the string may occur anywhere within the name field. Example: If "hospital" is one of the specified strings, then records with these names would match: "Hospital", "Rogue Hospital", "County Clinic-hospital." Note: Many TIGER Landmark and Basic records do not have any information in the feature name field, so it is not good to rely solely on this field; you should always use the appropriate CFCC (*e.g.*, D31 for Hospitals), also.

Important Note: Some of the Type 1 records have Dnn CFCC Codes and represent linear "Landmark" features, *e.g.*, runways at small airports (CFCC=D51), and maps of these features may be extracted using *v.in.tig.basic* after suitable extraction of "D51" records from the Type 1 file. Other features of the same type, *e.g.*, large airports, may be recorded as Landmark area (polygon) features in the Type 7 file, and maps of both kinds of airports may have to be extracted and combined to have a complete map of airports for a county.

Examples of Making Vector Maps from TIGER Files:

1. Create a vector map of *all* TIGER lines in Lawrence County, SD, in a UTM location. [The entire Type 1 file is processed, UTM zone 13 and the *clark66* spheroid are used by default.]

("../" represents an arbitrary directory pathname)

```
v.in.tig.basic -q t1=.../TGR46081.F41 t2=.../TGR46081.F42 out=46081.all
```

2. Create a vector map of all interstate and limited access roads:

```
awk 'substr($0,56,2)=="A1"' < .../TGR46081.F41 > type1.tmp
v.in.tig.basic t1=type1.tmp t2=.../TGR46081.F42 out=roads.major
rm type1.tmp
```

3. Create a vector map of the Morrison River (compare with Method 1 under **Selecting Type 1 Records**). The TLID field is automatically written as the attribute for each line in the output map. Topology is constructed (*-t* flag) so that you may use *d.what.vect* to interactively retrieve the TIGER/Line ID number for each line segment, and thus have a linkage back to the original record.

```
m.in.stf1.tape 56 H 20 Morrison < .../TGR46081.F41 > type1.tmp
  [56=H selects all hydrographic features and 20=Morrison selects
   the lines with feature name Morrison]
v.in.tig.basic -t t1=type1.tmp t2=.../TGR46081.F42 out=Morrison.Riv
rm type1.tmp
```

4. Same goal as Example 3, but using *v.in.tig.lndmk*. (The program prompts are not shown.)

```
v.in.tig.lndmk -s t1=.../TGR46081.F41
[The following lines are input from stdin (the keyboard).]
.match_string
!Morrison
.end
.line vect=Morrison.Riv
H
.end
.exit
```

5. Extract maps of cemetaries (point Landmarks as a site list, and area Landmarks as a vector map).

```
v.in.tig.lndmk -s t1=.../TGR46081.F41
.point site=D83.points
D83
.end
.area vect=D83.areaS
D83
```

```
.end
.exit
```

Comments on the CFCCs and Creating GRASS Maps

As many of the above examples imply the CFCC field beginning in column 56 in the Type 1 files is an important one. The major categories of these codes include:

Road Features (A..)
Rail Features (B..)
Pipelines, Power Transmission Lines, and
Miscellaneous Transportation Features (C..)
Landmark Features (D..)
Other Physical Features (E..)
Nonvisible Boundaries (F..)
Hydrographic Features (H..)
Features Not Yet Classified (X..)

Each major category contains 5 to 60 subdivisions [some of which are interesting: A52 for "Jeep trail, passable only by four-wheel drive (4WD) vehicle, undivided, in tunnel"]. The user should study the list of CFCCs and try making simple maps first. Be aware that the CFCC may not be what each person would think a line segment should be labeled; roads may not always be properly classified (particularly A20's and A30's), and boundaries of large airports may be labeled "F40 - Property Line" rather than "D51 - Airport".

Note: For a large number of Type 1 records, especially outside urban areas, many of the data fields are blank; *e.g.*, Feature Name, Address Ranges and ZIP Code fields usually have no entries.

The GRASS vector topology (dig_plus) file can be created using the *v.support* program after running *v.in.tig.basic*, or during *v.in.tig.basic* by setting the *-t* flag. Since this process can take considerable time, you may wish to do it only on selected maps where topology is needed for programs like *v.digit* and *v.to.rast*. Be aware that *v.support* may not be able to build topology if the map has too many vectors (the limit on most modern workstations is probably well over 100,000).

The *v.in.tig.basic* map building process does not respect the current GRASS region; *i.e.*, all appropriate lines in the Type 1 input file included in the vector map produced.

The limit to the utility of this import process is how creative you can be in identifying/extracting meaningful subsets of Type 1 lines based on the 42 fields of data, and Type 1 lines selected by reference to other TIGER file types such as Landmark records. Some installations may want to write specialized database processes for doing this selection. Future extensions to this import process should include the extraction of line segments cross-referenced through File Type 3 (Additional Geographic Area Codes), especially for maps of voting districts.

Section II. 1990 Census STF1 Data and Import Programs

About the 1990 STF1 and PL94 Data and Files

A portion of the demographic data distributed by the Bureau of the Census consists of the "1990 Census of Population and Housing" Summary Tape File 1, known hereafter as STF1. Four different STF1 file types are described in the Census Bureau's documentation: STF1A files contain data down to the "block group" level and are distributed as state files; 150 MB is a typical file size. STF1B files contain most of the STF1A data plus records for each census "block" and are distributed as state files; 1500 MB is a typical file size for a state. The STF1C file contains records for the entire United States down to the "county" level in a 300 MB file of about 31,000 records. STF1D provides data for the Congressional Districts of the 103rd Congress in a single file of unknown size. The GRASS import programs described below have been tested on tape and CD-ROM files for all types except STF1D at the time of this writing.

Many kinds of data are repeated in two or three of these files. The Technical Documentation can help you decide which file(s) to obtain.

Users of the programs described below should obtain and study the following document:

> *Census of Population and Housing, 1990: Summary Tape File 1 Technical Documentation*, prepared by the Bureau of the Census.--Washington: The Bureau. 1991.
> This can be obtained by printing the chapters from several files in the "document" subdirectory on the CD-ROM used for data distribution, or from:
>
> Data User Services Division
> Data Access and Use Staff
> Bureau of the Census
> Washington, D.C. 20233
> Phone: (301)763-2074
>
> Additionally, several critical portions of the document may be printed by setting suitable flags in the *m.in.stf1.tape* program.

The STF1A file for each **state** can be obtained from the following address. The CD-ROM distribution media is recommended for those installations with limited amounts of disk space, although routine operation from CD-ROM is slow.

> Data User Services Division
> Customer Services Branch
> Bureau of the Census
> Washington, D.C. 20233
> Phone: (301)763-4100

Caution should be exercised in copying and using the STF1 tape files because they contain extremely long text records (4806 characters) and most Unix text filter utilities (*e.g., grep* or *sort*) will not properly deal with these records. Also, each logical record (9612 characters!) consists of two physical records in which the first 300 characters are duplicated. On CD-ROM the STF1 records are split across 10 files (compatible with dBase III); if these are copied to disk, be certain to treat them as binary files and copy all of them to a single directory without changing the filenames.

The PL94 (Public Law 94-171) file in use at some installations consists of the first 300 characters for each record from the STF1B file plus a few selected demographic fields (508 characters total)

and, thus, has useful data to the census "block" level. The "PL94" file can be used in place of the STF1 file by the import programs *m.in.stf1.tape* and *s.in.stf1* discussed here. Data field information for PL94 files is described in the document:

Census of Population and Housing, 1990: Public Law (P.L.) 94-171 Data Technical Documentation, prepared by the Bureau of the Census.--Washington: The Bureau. 1991.
Data User Services Division
Data Access and Use Staff
Bureau of the Census
Washington, D.C. 20233
Phone: (301)763-2074
Additionally, data field information may be printed by setting a suitable flag in the *m.in.stf1.tape* program.

In the STF and PL94 file records there are two subdivisions of data field types. The IDENTIFICATION SECTION consists of 67 fields in the first 300 characters; these fields serve to identify the census counting unit and give some useful geographic information such as location and area of the counting unit. The MATRIX SECTION consists of the remainder of the record, over 900 fields for the STF files and 23 fields for the PL94 records. All ID SECTION fields can be referenced by their position in the record or by their field names. The STF1 MATRIX SECTION fields can be referenced by their position in the record or by the CD-ROM matrix identification codes (except the PL94 matrix fields which must be referenced by position). Lists of these names and codes may be printed with suitable flags in the *m.in.stf1.tape* program. These reference methods apply to data obtained from tape or CD-ROM sources.

The census counting units are also called Summary Levels; and the Summary Level for each record is a 3-digit value in columns 11-13, also known as the SUMLEV field. Most extractions will be based on a SUMLEV value. The Summary Level charts for the four STF1 file types are presented in Chapter 6 of the Technical Documentation, or may be printed by setting a flag in the *s.in.stf1* program (see below). A few of the commonly used ones are:

040 State
050 County
140 Census Tract/Block Numbering Area
150 Block Group
100 Block

Demographic Data Usage Outlined

Two steps are normally necessary to import and use the STF1 or PL94 data.

First, the records of interest are extracted and collected in a disk file using *m.in.stf1.db3* or *m.in.stf1.tape*, depending on whether the CD-ROM or tape distribution format is being used. Both of these programs write the extracted records with the "tape format" for consistency of later processing. Additional extractions may be done on the files of extracted records with *m.in.stf1.tape*, as desired.

Next, the program *s.in.stf1* is run to process a file of selected records and create a variety of output types, including tabular reports, GRASS site maps, or GRASS vector attribute files. This will be the most used program of this set.

Using m.in.stf1.db3 to Extract STF1 Records from CD-ROM

This program must be run in command mode only. The command line parameters: (1) define the records to be extracted by specifying the contents of specific columns in desired records, and (2) give the path/file name of the first of the set of CD-ROM files desired. The output is sent to stdout and can be redirected to a file of your choice.

Examples:

```
m.in.stf1.db3 -p SUMLEV=050 in=/cdrom/stf1a0wa.dbf > $HOME/wa.050
m.in.stf1.db3 SUMLEV=140 CNTY=081 in=/cdrom/STF1A0SD.DBF > sd.140.081
m.in.stf1.db3 11=140 72=007 in=/cdrom/stf1a0az.dbf > az.140.007
```

Example 1 extracts all 39 of the county level records from the STF1A CD-ROM files for Washington State and reformats them to the tape file format and writes them to the file *wa.050* in your home directory. The *-p* flag displays information about each of the 10 associated ".DBF" files as they are opened.

Examples 2 and 3 perform identical operations (on files for different states). They extract the tract records for a specified county. Example 2 has the starting columns expressed as field names, and Example 3 uses explicit column numbers. The program converts the field names to column numbers prior to processing.

The *-f* flag can be used to print a list of IDENTIFICATION SECTION field names and their column positions and widths. The *-n* flag can be used to output only the sequential number of each record that matches the selection criteria (this is for general information).

Multiple selection criteria on a single command line are logically "anded" to form a composite test. The tests are made as string, not numeric, comparisons; so SUMLEV=050 is not the same as SUMLEV=50. The following are valid selection criteria examples; note the use of "!=" for "not equals":

```
SUMLEV=150 CNTY!=037    (Block Group records for all counties except 037)
CNTY=004                (All records for county 004)
SUMLEV=050 192!=C       (All counties, except for those beginning with 'C')
```

Using m.in.stf1.tape

Review the discussion of *m.in.stf1.db3*, above. The only differences in using these two programs are: the *-p* flag is not recognized, and the input for *m.in.stf1.tape* comes from a single file redirected through stdin, or from records "piped" to this program. For example, the input text records may come directly from magnetic tape.

Examples:

```
#  Select all records for county 051 directly from tape
m.in.stf1.tape CNTY=051 < /dev/rst0 > tmp.file.1

#  Select all tract records, except tract 9753
m.in.stf1.tape SUMLEV=140 52!=9753 < tmp.file.1 > tracts

#  Select all lines from an arbitrary text file which contain "T450"
#    beginning in column and "Bu??s" starting in column 7; note
#    the ? single-character wild cards and the quotes so they will not
```

```
#    be recognized by the Shell command interpreter.
m.in.stf1.tape 1=T450 '7=Bu??s' < textfile > outfile

#  Select all non-blank lines
m.in.stf1.tape '1=?' < textfile > outfile
```

Note: This program contains special code to accomodate the very long STF1 records and PL94 records which may be distributed with a variety of terminating characters for each record. As they are processed, the selected records are output with the Unix-normal Control-J character as the terminator. A by-product of this code is that blank lines are removed, even if all input lines are selected. This program can be used as a filter on any text file; specifically, it may be used to select TIGER/Line Type 1 records for input to *v.in.tig.basic*. Also, it will handle input lines up to 10,000 characters long; whereas, most of the Unix text filter programs will not properly process lines longer than 500-2000 characters and **should not** be used to process STF1 tape format records (4806 characters).

Using s.in.stf1

This is a powerful, multi-purpose program for processing files of PL94 and STF1 (tape format) records. Normally, files created by *m.in.stf1.tape* or *m.in.stf1.db3* are used as input.

Several concepts are crucial to understand when using this program. First, there are several available **output choices** which expand the potential applications of this program beyond creating maps. Second, a specified algebraic **formula** is applied to each input record to create a resulting value. Third, two fields in the IDENTIFICATION SECTION of each record contain the latitude and longitude of a **representative point** located within the boundaries of the census counting unit the record represents (these are converted to UTM coordinates by the program, when necessary).

An explanation of the program's parameters will help you get started.

Command syntax:

```
s.in.stf1 [-dplu] input_stf1=name [out=name] [name_field=name]
      [zone=value] [spheroid=name] ef=name | formula=expr
```

Four flags are available for printing documents useful when running the program. Only one flag can be used at a time, and the program exits after sending the document to stdout. These flags are:

```
-d   Output IDENTIFICATION SECTION to stdout (20 pages)
-p   Output STF1 MATRIX TABLE to stdout (30 pages)
-l   Output PL94 MATRIX TABLE to stdout (1 page)
-u   Output STF1 SUMLEV TABLE to stdout (4 pages)
```

input_stf1 is the path/name of the file containg the STF1 or PL94 records to to processed.

out specifies the type and destination of output (default is the - or stdout option). The choices are:

site, writes a grass site list in the current mapset containing the coordinates and the value calculated from the formula as the site "description" for each input record processed.

atts, writes a GRASS vector attribute file containing the representative point coordinates and the value calculated from the formula as the "attribute."

table:Lxxx:Lxxx. . ., sends to stdout a table of results with the formula value preceeded by the contents of the field(s) specified by the ":Lxxx" items; see below for the meaning of "Lxxx".

- *(stdout)*, sends to stdout the representative point coordinates and the formula value for each of the input records.

ef is the path/name of a text file with formula expression(s) and optional output choices.

formula is a statement of the form **map=expression**; see below.

Note: Only one of the **ef** and **formula** parameters may be used.

name_field, zone, spheroid: See the reference manual page for information.

Construction of the Formula and "Lxxx" Column Designations

The "formula" consists of two parts: "mapname=expression" where "mapname" is the name of the sites map or vector attributes file being generated, and the "expression" consists of operators and operands. The "mapname" is required, even though it is ignored when output is sent to stdout. Operators in the expression may be any of the standard arithmetic ones (+, -, *, /, %) plus the functions allowed by the UNIX *bc* calculator, including parentheses used in the normal manner. The operands may be integer or real constants, and numeric field values from the IDENTIFICATION and MATRIX SECTIONS of the STF1 or PL94 records. Specification of these numeric field values can take three forms:

Lxxx, where L is an upper case alphabetic letter, and xxx is a number designating the column in which the required field starts. The width of the data field is denoted by the letter used: A=1, B=2, ..., I=9, J=10, etc. Most population and housing data fields are nine (9) characters wide, so the most common usage is Ixxx; example: *I301* for the total (100%) population field.

Pnnnnnn or Hnnnnnn are the column designation strings listed for the MATRIX SECTION fields in the CD-ROM STF1 documentation. They are the data field names when using the CD-ROM with dBase III. When used here, they are automatically translated to the Lxxx format.

The field names from the IDENTIFICATION SECTION may also be used as operands; they, too, are translated to the Lxxx form. Remember, these are the same for the STF1 and PL94 records.

The column numbers, column designation strings, and ID SECTION field names can be found in the Technical Documentation or the documents which can be printed by the "flags" in this program.

Examples of Using s.in.stf1

For the following examples and comments we assume that a file of STF1A records representing the census tracts in Lawrence County, SD, has been extracted to a file *081.140* in your home directory.

Note the use of quotations to protect special characters from Shell interpretation, and $HOME for the user's home directory. Of course, tabular output to stdout may be redirected to a file or piped to other processes.

Example 1. Create a table giving the representative point coordinates and the percent Hispanic persons in each tract:

```
s.in.stf1 in=$HOME/081.140 out=- formula='pct.hisp=100*I652/I301'
```

The percent Hispanic population was generated by dividing the total number of Hispanics by the total population and multiplying by 100. Note the quotations to protect the expression from Shell interpretation. The map name is informative, but ignored for stdout.

Example 2. Create a table giving the tract number, the number of males, the number of females and the ratio of males to females.

```
s.in.stf1 in=$HOME/081.140 out=table:TRACTBNA:I364:I373
        formula='m = I364 / I373'
```

"TRACTBNA" is the ID SECTION field name for the tract number; the male and female population counts are in columns 364-372 and 373-382, respectively. Note the use of a "dummy" map name in the expression and the optional spaces in the expression.

Example 3. Create a GRASS site map in the current mapset with the site descriptions having the value of the number of people per square mile. The description fields in the site file will have a "#" preceeding the value calculated from the formula. The site map will be named "pop.density".

```
s.in.stf1 in=$HOME/081.140 out=site
              f='pop.density=2.59 * I301 /(AREALAND/1000)'
```

Typical site list entry: 601230.55|4721349.00|#345.473509

AREALAND is the ID SECTION field that gives the land area in the counting unit in units of thousandths of square kilometers. This is divided by 1000 to yield square kilometers of area. The 2.59 factor converts square kilometers to square miles. "AREALAND" in the expression could have been replaced by "J172" because that field is 10 characters long and begins in column 172. The area occupied by water in a counting unit is reeferenced by "J182" or "AREAWAT"; of course the total area in the counting unit is the sum of AREALAND and AREAWAT.

Example 4. Create a GRASS vector map attributes file giving the ratio of "owner occupied" to "renter occupied" housing units for each census tract. It is recommended that this be done after the vector map of tract boundaries has been created or copied to the current mapset with a name appropriate to the attributes being calculated, in this case "own.to.rent". The attribute file generated **replaces** the current one, if any.

```
s.in.stf1 in=$HOME/081.140 out=att f='own.to.rent=H0030001/H0030002'
```

Typical attribute file entry: A 601230.55 4721349.00 3.9089832

The CD-ROM style data field names have been used here; "I6609/I6618" could have been used as the expression. The values calculated by this and many other expressions may be small values, and will have many decimal places because of processing by the *bc* calculator program. The decimal places will be truncated off by the v.support program when it is run to (re)create the topology (dig_plus) file after installation of the new attributes. A scaling factor, say 10 or 100,

may be included in the expression so that the integer attribute values remaining after truncation of decimal places have an appropriate spread of values; the program may be run with the *out*=- mode to preview the calculated values and determine an appropriate scaling factor. Attribute values less that 1.0 will be truncated to zero, and v.support will discard attributes of zero.

SectionIII. Integrating TIGER/Line and STF1 Data in GRASS 4 Maps

Complete Example Using TIGER and STF1A Files--The Usual Sequence of Steps

For this example, the final map desired is a raster map of Census Tracts (or "Block Numbering Areas") for that portion of Lawrence County, SD, within the default region of the "Spearfish" database with category values representing the total number of persons between the ages of 21 and 29 in each tract. The TIGER and STF1A records are assumed to be available on CD-ROM. ["Mounting" the CD-ROM will require slightly different procedures on different workstations, see your "system administrator." Also, file names on the CD-ROM may be either upper or lower case, depending on your workstation.]

Step 1. Install the TIGER CD-ROM for South Dakota and "mount" the file system it contains as "/cdrom". Extract the appropriate Type 1 TIGER records. *awk* is used because this is a complex selection involving the comparison of values in two fields. All TIGER Type 1 records where the tract number on the left side of the line segment is not equal to the tract number on the right side of the line are selected. By definition, these lines are tract boundaries; and include all the county bounday line segments as these are also tract boundaries.

```
awk 'substr($0,171,4)!=substr($0,177,4)' < /cdrom/TGR46081.F41 > tr.tmp
  [Left and Right side tract numbers are in columns 171-174 and 177-180]
```

Step 2. Make the vector map of tract boundaries and display it.

```
v.in.tig.basic t1=tr.tmp t2=/cdrom/TGR46081.F42 out=tracts.all
rm tracts.tmp
g.region vect=tracts.all; d.erase
d.vect tracts.all
```

Step 3A: "Mount" the STF1A CD-ROM which includes South Dakota and extract all the Lawrence County STF1A records from the CD-ROM to a disk file. All the county records are extracted because it may take up to 3 hours to search all the CD-ROM records, and we don't want to do this too often--so get them all now. Remember, this program writes the selected records in the "STF1 Tape Format."

```
m.in.stf1.db3 CNTY=081  in=/cdrom/stf1a0sd.dbf > 081.all
 [The county code for Lawrence Co. is 081]
```

Step 3B: Select the STF1A tract summary records from the file of all Lawrence County STF1A records. This subset, too, should be saved for future use in making other demographic tract maps for Lawrence County. There will be fewer than 10 records selected because there are not many census tracts in this sparsely populated region.

```
m.in.stf1.tape SUMLEV=140 < 081.all > tracts.081
```

Step 4. Make a copy of the tract vector map with an appropriate name. Keep the original so you will not ever have to do steps 1 and 2 again as you make additional tract maps.

```
g.copy vect=tracts.all,tracts.21.29
```

Step 5A. Create the polygon attribute file from the STF1A tract records. The expression sums the various required age subdivision fields.

```
s.in.stf1 in=tracts.081 out=att f='tracts.21.29=I931+I940+I949'
```

Step 5B. Generate a table of these results on your terminal to contemplate.

```
s.in.stf1 in=tracts.081 out=table:TRACTBNA f='m=I931+I940+I949'
```

Step 6. Build the vector topology file to attach the attributes to the tract polygons.)

```
v.support tracts.21.29 option=build
```

Step 7. Make the raster map. Set to the default region first, and note that none of the preceeding steps respected the "current region."

```
g.region -d
v.to.rast in=tracts.21.29 out=tracts.21.29
```

Finished! Display the new raster map with *d.3d*

Steps 4, 5, and 6 may be repeated (rather quickly) to make as many demographically labeled vector tract maps the user desires.

Step 7 can be executed whenever a raster map is desired. Adjust the region boundaries and resolution to suit your needs. *r.mapcalc* may be particularly useful for making additional maps by arithmetically combining a variety of raster demographic maps produced by the steps listed above, or demographic maps in combination with other thematic maps in your location.

Producing a Block Group Map

The census counting unit smaller than a tract is a block group. The STF1A data has this as the smallest subdivision. Your goal here is to make a vector map with the total population in each block group as the polygon attributes.

Step 1. Extract the TIGER line segments which are coded as block group boundaries. This is fairly complex, as the block group numbers are repeated in each tract. First check each line to see if it is a tract bounday line, as above. If it is, accept it. If not, then check to see if it is a block group boundary line (different block group number on each side. If so, accept it. [The "||" symbol means "or". A complex *awk* expression may be put in a file and executed with *awk -f filename < input > output*]

```
awk 'substr($0,171,4)!=substr($0,177,4) ||
     substr($0,183,1)!=substr($0,187,1)' < /cdrom/TGR46081.F41 > bg.all
```

The tract numbers on the left an right sides of each line segment are in four character fields beginning in columns 171 and 177; columns 183 and 187 contain the single digit block group numbers.

Step 2. Create and display GRASS vector map of block group polygons.

```
v.in.tig.basic t1=bg.all t2=/cdrom/TGR46081.F42 out=bg.all
g.region vect=bg.all; d.erase
d.vect bg.all
```

Step 3. Extract the STF1A block group records.

```
m.in.stf1.tape SUMLEV=150 < 081.all > bg.081
```

Step 4. Make a copy of the block group vector map with the proper name for the demographic map you want.

```
g.copy vect=bg.all,bg.pop100
```

Step 5. Attach the desired demographic attribute (total population) to each polygon and build the topology file.

```
s.in.stf1 in=bg.081 out=att f='bg.pop100=I301'
v.support bg.pop100 option=build
```

Done! Repeat Steps 4 and 5 to make additional block group demographic vector maps.

Other Possibilities

If a raster map of single cells with the population density values were required, the process is shorter.

```
# generate site list map with calculated attributes scaled by * 10
s.in.stf1 in=stf.140 out=site f='popden=10*(I301/((J172+J182)/1000))/2.59'
# convert to cell map
s.menu  . . . (option 7)
# done
```

The user can experiment with a host of labor saving devices to improve productivity with these tools. For example, path/filename and/or a complex formulae may be defined as Shell variables and used repeatedly:

```
setenv S1 /home/me/stf.data/stf.140
setenv POPDEN '10*(I291/((J172+J182)/1000))/2.59'
s.in.stf1 in=$S1 out=site f=popden="$POPDEN"
s.in.stf1 in=$S1 f=popden="$POPDEN"
s.in.stf1 in=$S1 out=table:D52 f=popden="$POPDEN"
```

Appendix A. Manual Pages for Programs

[The manual pages for *v.in.tig.basic, s.in.stf1,* and *m.in.stf1.tape* follow the comments below.]

Comments on **v.in.tig.basic**

This is a C program which does the actual integration of coordinates from a file of Type 1 records (usually selected with *m.in.stf1.tape* or a Unix utility) and the county-wide file of Type 2 records, to produce a GRASS4.0 vector file and attribute file in the user's current mapset.

A list of steps performed by this program will help explain what it does.

1. Verify that GRASS4.0 is running and that the current "location" has a latitude-longitude or UTM coordinate system.

2. Read the Type 2 file and build a table of pointers to each record, and sort the pointers for faster access. If small enough, the pointer table will be kept in memory to increase execution speed, otherwise it is used from a disk file. (The maximum size to keep in memory is determined at run time; see source code.)

3. For each Type 1 record do steps a through h:

    a. Read the record.

    b. Extract the TLID, and end point lat-long coordinates (nodes).

    c. Find and insert the shape points from Type 2 records, if any. (Finding is the time-consuming step.)

    d. If necessary, convert the coordinate pairs to UTM. The current zone and the "clark66" spheroid are used unless changed by the *zone* and/or *spheroid* command line parameters.

    e. Write the line to the vector spaghetti file (in the "dig" directory).

    f. Calculate the representative point for the line (half-way between the nodes if only two points, otherwise use point (n+1)/2), and convert to UTM, if necessary.

    g. Write the representative point coordinates and the TLID number as the attribute for that line with a 'L' or 'line' vector type designation (in the "dig_att" directory).

    h. If the *-v* flag is set, write the information to a temporary file for updating a variety of fields for each record in a special *v.db.rim* data base.

4. Write an appropriate header for the vector file.

5. Run v.support to build the topology file, if requested by the *-t* flag.

Timing results for **v.in.tig.basic**

Test runs building a vector file for all lines for two counties were made on a SUN4/280 (a "10 MIPS, 16MB" machine) running an X-windows user interface. The results:

For Santa Cruz Co., AZ, with about 10,500 Type 1 records and 10,400 Type 2 records, 3 minutes and 20 seconds were required to build the map. Input files were on the same disk as the GRASS data base. The -q (quiet) flag was set. Pointer file was kept in memory.

For Pima Co., AZ, with about 81,000 Type 1 records and 54,000 Type 2 records, about 40 minutes were required. Input files were used directly from CD-ROM. No flags were set. Pointer file was kept in memory.

If the input files are on disk and the pointer file is small enough to fit in memory, the execution speed is limited by the processor speed, and should improve approximately as the machine speed increases.

Comments on **s.in.stf1**

This C program allows reading, computation and output of locational and demographic data contained in Census Bureau STF1 and PL94-171 files.

This program performs the following steps:

1. Opens file in first parameter for reading (STF1 or PL94-171 file, or subset).

2. Reads map=formula parameters and concatenates them into a string (formula).

3. If necessary, reads stdin for record numbers to process.

4. Reads the two-line STF1 or one-line PL94-171 record into buffer.

5. Extracts "east" and "north" coordinates.

6. Substitutes fields in formula with values from buffer.

7. Sends coordinates and formula to "bc -l | paste - - - > file or stdout".

8. Loops for all standard input or STF1 records.

Execution speed is quite rapid for reasonable numbers (hundreds) of input records.

Appendix B. LIST OF CENSUS FEATURE CLASSIFICATION CODES

A00 Road, Class Unknown or Not Elsewhere Classified
A01 Road, undivided
A02 Road, undivided, in tunnel
A03 Road, undivided, underpassing
A04 Road, undivided, rail line in center
A05 Road, divided
A06 Road, divided, in tunnel
A07 Road, divided, underpassing
A08 Road, divided, rail line in center
A10 Pri. rd, interstate hwy and lmtd acc. road
A11 Pri. rd, interstate hwy and lmtd acc. road, undiv.
A12 Pri. rd, interstate hwy and lmtd acc. road, undiv., in tunnel
A13 Pri. rd, interstate hwy and lmtd acc. road, undiv., underpassing
A14 Pri. rd, interstate hwy and lmtd acc. road, undiv., rail in center
A15 Pri. rd, interstate hwy and lmtd acc. road, divided
A16 Pri. rd, interstate hwy and lmtd acc. road, divided, in tunnel
A17 Pri. rd, interstate hwy and lmtd acc. road, divided, underpassing
A18 Pri. rd, interstate hwy and lmtd acc. road, divided, rail in center
A20 Sec. rd, U.S. hwy not A10, state rds
A21 Sec. rd, U.S. hwy not A10, state rds, undiv.
A22 Sec. rd, U.S. hwy not A10, state rds, undiv., in tunnel
A23 Sec. rd, U.S. hwy not A10, state rds, undiv., underpassing
A24 Sec. rd, U.S. hwy not A10, state rds, undiv., rail in center
A25 Sec. rd, U.S. hwy not A10, state rds, divided
A26 Sec. rd, U.S. hwy not A10, state rds, divided, in tunnel
A27 Sec. rd, U.S. hwy not A10, state rds, divided, underpassing
A28 Sec. rd, U.S. hwy not A10, state rds, divided, rail in center
A30 Connect. rd, cnty rds, not A10 or A20
A31 Connect. rd, cnty rds, not A10 or A20, undiv.
A32 Connect. rd, cnty rds, not A10 or A20, undiv., in tunnel
A33 Connect. rd, cnty rds, not A10 or A20, undiv., underpassing
A34 Connect. rd, cnty rds, not A10 or A20, undiv., rail in center
A35 Connect. rd, cnty rds, not A10 or A20, divided
A36 Connect. rd, cnty rds, not A10 or A20, divided, in tunnel
A37 Connect. rd, cnty rds, not A10 or A20, divided, underpassing
A38 Connect. rd, cnty rds, not A10 or A20, divided, rail in center
A40 Neighborhood rds, city st., unimproved rds
A41 Neighborhood rds, city st., unimproved rds, undiv.
A42 Neighborhood rds, city st., unimproved rds, undiv., in tunnel
A43 Neighborhood rds, city st., unimproved rds, undiv., underpassing
A44 Neighborhood rds, city st., unimproved rds, undiv., rail in center
A45 Neighborhood rds, city st., unimproved rds, divided
A46 Neighborhood rds, city st., unimproved rds, divided, in tunnel
A47 Neighborhood rds, city st., unimproved rds, divided, underpassing
A48 Neighborhood rds, city st., unimproved rds, divided, rail in center
A50 Jeep trail (4WD only)
A51 Jeep trail (4WD only), undivided
A52 Jeep trail (4WD only), undivided, in tunnel
A53 Jeep trail (4WD only), undivided, underpassing
A60 Special Road Feature
A61 Cul-de-sac
A62 Traffic circle

A63 Cloverleaf or interchange
A64 Service drive
A65 Ferry crossing
A70 Other Thoroughfare
A71 Walkway
A72 Stairway
A73 Alley

B00 Railroad, Classification Unknown or Not Elsewhere Classified
B01 Railroad track
B02 Railroad track, in tunnel
B03 Railroad track, underpassing
B10 Railroad Main Track
B11 Railroad main track
B12 Railroad main track, in tunnel
B13 Railroad main track, underpassing
B20 Railroad Spur Track
B21 Railroad spur track
B22 Railroad spur track, in tunnel
B23 Railroad spur track, underpassing
B30 Railroad Yard
B31 Railroad yard
B32 Railroad yard, in tunnel
B33 Railroad yard, underpassing
B40 Railroad Ferry Crossing
B50 Other Rail Feature
B51 Carline
B52 Cog railroad, incline railway, or logging tram

C00 Spec. Transport. Feature, Class Unknown or Not Elsewhere Classified
C10 Pipeline
C20 Power Transmission Line
C30 Other Special Transportation Feature
C31 Aerial tramway, monorail, or ski lift

D00 Landmark Feature, Class Unknown or Not Elsewhere Classified
D10 Military installation
D20 Multihousehold and transient quarters
D21 Apartment building or complex
D22 Rooming or boarding house
D23 Trailer court or mobile home park
D24 Marina
D25 Crew of vessel
D26 Housing facility for workers
D27 Hotel, motel, resort, spa, YMCA, or YWCA
D28 Campground
D29 Shelter or mission
D30 Custodial facility
D31 Hospital
D32 Halfway house
D33 Nursing home, retirement home, or home for the aged
D34 County home or poor farm
D35 Orphanage
D36 Jail or detention center

D37 Federal penitentiary, state prison, or prison farm
D40 Educational or religious institution
D41 Sorority or fraternity
D42 Convent or monastery
D43 Educational institution
D44 Religious institution
D50 Transportation terminal
D51 Airport or airfield
D52 Train station
D53 Bus terminal
D54 Marine terminal
D55 Seaplane anchorage
D60 Employment center
D61 Shopping center or major retail center
D62 Industrial building or industrial park
D63 Office building or office park
D64 Amusement center
D65 Government center
D66 Other employment center
D70 Tower
D71 Lookout tower
D80 Open space
D81 Golf course
D82 Cemetery
D83 National park or forest
D84 Other federal land
D85 State or local park or forest
D90 Special purpose landmark
D91 Post office box ZIP code

E00 Physical Feature, Class Unknown or Not Elsewhere Classified
E10 Fence line
E20 Topographic feature
E21 Ridge line
E22 Mountain peak

F00 Nonvisible Boundary, Class Unknown or not Elsewhere Classified
F10 Nonvisible Political Boundary
F11 Offset corporate boundary
F12 Corporate Corridor
F13 Nonvis. interpolated boundary, polygon def. line for hydro. areas
F20 Feature Extension, Not Otherwise Classified
F21 Automated extension
F22 Irregular block extension
F23 Closure extension
F24 Nonvisible separation line
F25 Nonvisible corporate corridor centerline
F30 Point-to-Point Line
F40 Property Line
F50 ZIP Code Boundary
F60 Map Edge
F70 Statistical Boundary
F71 1980 Statistical Boundary
F72 1990 Block Boundary

F73 1990 Statistical Boundary
F74 1990 Statistical Boundary, Tabulation Areas Only

H00 Water Feature, Class Unknown or Not Elsewhere Classified
H01 Shoreline of perennial water feature
H02 Shoreline of intermittent water feature
H10 Stream
H11 Perennial stream
H12 Intermittent stream or wash
H13 Braided stream
H20 Canal, Ditch, or Aqueduct
H21 Perennial canal, ditch, or aqueduct
H22 Intermittent canal, ditch, or aqueduct
H30 Lake or pond
H31 Perennial lake or pond
H32 Intermittent lake or pond
H40 Reservoir
H41 Perennial reservoir
H42 Intermittent reservoir
H50 Bay, Estuary, Gulf, Sound, Sea, or Ocean
H51 Bay, estuary, gulf, or sound
H53 Sea or ocean
H60 Gravel Pit or Quarry Filled With Water
H70 Nonvisible Water Area Definition Boundary
H71 USGS closure line
H72 Census computed center line
H73 Census international water boundary, 10-mi. limit, area measure line
H74 Census water boundary, inland or coastal or Great Lakes
H75 3-mile limit water boundary
H80 Special Water Feature
H81 Glacier

X00 Feature Not Yet Classified