REUSE MANUAL

ELLIPSOID

10xxxxxx.1

Implementation

TABLE OF CONTENTS

SECTION 1. INTRODUCTION

1.1 PURPOSE OF THE REUSE MANUAL

This document describes the characteristics of the ELLIPSOID reusable software component and provides instructions on its installation and operation. The manual is a self-contained reference for the software engineer intending to incorporate the component in another software system. This manual was written with the assumption that the user has a basic working knowledge of C and is familiar with fundamental C concepts and terminology.

1.2 PURPOSE OF THE REUSABLE SOFTWARE COMPONENT

The purpose of ELLIPSOID is to provide access to ellipsoid parameters for a collection of common ellipsoids. A particular ellipsoid can be accessed by using its standard 2-letter code to find its index in the ellipsoid table. The index can then be used to retrieve the ellipsoid name and parameters.

By sequentially retrieving all of the ellipsoid codes and/or names, a menu of the available ellipsoids can be constructed. The index values resulting from selections from this menu can then be used to access the parameters of the selected ellipsoid.

This component depends on a data file named “ellips.dat”, which contains the ellipsoid parameter values. A copy of this file must be located in the directory specified by the value of the environment variable "ELLIPSOID_DATA", if defined, or else in the current directory, whenever a program containing this component is executed.

Additional ellipsoids can be added to this file, either manually or using the Create_Ellipsoid function. However, if a large number of ellipsoids are added, the ellipsoid table array size in this component will have to be increased.

1.3 GENERAL INFORMATION

1.3.1 POINT OF CONTACT

U.S. Army Topographic Engineering Center (USATEC)

Geospatial Information Division (GID)

ATTN: CETEC-GD-A (Dan Specht)

7701 Telegraph Road

Alexandria, VA 22315-3864

Dan Specht (703) 428 - 6761 Project Manager

1.3.2 CERTIFICATION LEVEL

This RSC has been certified at level 4. A level 4 component satisfies the criteria for reliability, testing, and documentation for the Army Reuse Center (ARC). The component comes with test materials and a Reuse Manual that aids in integrating the component into a software system.

1.3.3 LEGAL RESTRICTIONS

This Reusable Software Component (RSC) contains data with Unlimited Government Rights.

SECTION 2. INSTALLATION

The following is a list of the files which make up the ELLIPSOID component:

Source Code Files:

```
ellipse.c
```

Header Files :

```
ellipse.h
```

Data Files :

```
ellips.dat
```

The compilation instructions for the ELLIPSOID component are as follows:

DOS Makefile (Uses Microsoft C):

```
cl /nologo /W3 /FR /G2 /DNDEBUG /Gs /Ox /AM /D_DOS /c ellipse.c
```

UNIX Makefile (Uses gcc compiler):

```
cc -g -O -ansi -Wall -c ellipse.c
```

The compilation order of the ELLIPSOID component relative to other components is unconstrained.

2.1 PARTIAL REUSE

The ELLIPSOID component does not allow for partial reuse.

2.2 MODIFICATIONS

The ELLIPSOID component does not permit modifications.

SECTION 3. ENVIRONMENT

This section provides details on the environment under which ELLIPSOID was developed, tested, and executed.

3.1 HARDWARE

3.1.1 DEVELOPMENT

The following is a list of hardware configurations under which ELLIPSOID was developed and tested.

- SUN SparcStation 20
- IBM compatible Pentium PC

3.1.2 TARGET

The following is a list of hardware configurations under which ELLIPSOID was executed.

- SUN SparcStation 20
- IBM compatible Pentium PC

3.2 SOFTWARE

3.2.1 OPERATING SYSTEM

The following is a list of operating systems under which ELLIPSOID was executed and tested.

- Solaris 2.5
- Windows 95

3.2.2 COMPILERS

The following is a list of compilers on which ELLIPSOID was compiled successfully.

- GCC version 2.8.1
- Microsoft Visual C++ version 6

3.3 ASSUMPTIONS AND PERFORMANCE LIMITATIONS

There are no hardware or environment constraints. There are no limitations.

This RSC is written in ANSI C.

SECTION 4. GLOBAL RSC ENVIRONMENT

4.1 TYPES

Not applicable. All types defined in this component are private.

4.2 CONSTANTS

The following is a list of significant visible constants declared globally in ELLIPSOID with their descriptions.

```
ELLIPSE_NO_ERROR                : No errors occured in function
ELLIPSE_FILE_OPEN_ERROR         : Ellipsoid file opening error
ELLIPSE_INITIALIZE_ERROR        : Ellipsoid table can not initialize
ELLIPSE_TABLE_OVERFLOW_ERROR    : Ellipsoid table overflow
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid table not initialized properly
ELLIPSE_INVALID_INDEX_ERROR     : Index is an invalid value
ELLIPSE_INVALID_CODE_ERROR      : Code was not found in table
ELLIPSE_A_ERROR                 : Semi-major axis less than or equal to zero
ELLIPSE_B_ERROR                 : Semi-minor axis less than or equal to zero
ELLIPSE_A_LESS_B_ERROR          : Semi-major axis less than semi-minor axis
```

4.3 VARIABLES

Not applicable. All variables defined in this component are private.

4.4 INCLUDE FILES

```
ctype.h                         : Standard C character handling library
stdio.h                         : Standard C input/output library
stdlib.h                        : Standard C general utility library
string.h                        : Standard C string handling library
ellipse.h                       : Prototype error checking and error codes
```

4.5 DEPENDENCIES

None, other than the standard ANSI C character handling, input/output, general utility, and string handling libraries.

SECTION 5. FUNCTIONS

5.1 INITIALIZE_ELLIPSOIDS

5.1.1 DESCRIPTION

This function initializes the module by reading “ellips.dat”, which must be in the current directory. It uses the data from this file to create an ellipsoid table.

5.1.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Initialize_Ellipsoids();
```

Not applicable.

5.1.3 DECLARATIONS

5.1.3.1 TYPES

Not applicable.

5.1.3.2 CONSTANTS

Not applicable.

5.1.3.3 VARIABLES

Not applicable.

5.1.4 DEPENDENCIES

Assign_Ellipsoid_Row – used to copy a table entry.

5.1.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occured in function
```

```
ELLIPSE_FILE_OPEN_ERROR          : Ellipsoid file opening error
ELLIPSE_INITIALIZE_ERROR         : Ellipsoid table can not initialize
ELLIPSE_TABLE_OVERFLOW_ERROR     : Ellipsoid table overflow
```

5.2 CREATE_ELLIPSOID

5.2.1 DESCRIPTION

This function creates a new ellipsoid with the specified code, name, and axes, adds it to the ellipsoid table, and updates the ellipsoid data file. The specified code must not already exist in the ellipsoid table.

5.2.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Create_Ellipsoid (const char *code,
                       const char *name,
                       double a,
                       double b);
```

code	The 2-letter code for the new ellipsoid (input),
name	The name for the new ellipsoid (input),
a	The semi-major axis, in meters, of the new ellipsoid (input),
b	The semi-minor axis, in meters, of the new ellipsoid (input).

Example:

```
status = Create_Ellipsoid(code, name, a, b)
```

Inputs:

code	"XX"
name	"Experimental ellipsoid"
a	6377295.664
b	6356094.668

5.2.3 DECLARATIONS

5.2.3.1 TYPES

Not applicable.

5.2.3.2 CONSTANTS

Not applicable.

5.2.3.3 VARIABLES

Not applicable.

5.2.4 DEPENDENCIES

None.

5.2.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occured in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid table not initialized properly
ELLIPSE_TABLE_OVERFLOW_ERROR    : Ellipsoid table overflow
ELLIPSE_INVALID_CODE_ERROR      : Code is already used in table
ELLIPSE_A_ERROR                 : Semi-major axis less than or equal to zero
ELLIPSE_B_ERROR                 : Semi-minor axis less than or equal to zero
ELLIPSE_A_LESS_B_ERROR          : Semi-major axis less than semi-minor axis
ELLIPSE_FILE_OPEN_ERROR         : Ellipsoid file opening error
```

5.3 ELLIPSOID_COUNT

5.3.1 DESCRIPTION

This function returns the number of ellipsoids in the ellipsoid table.

5.3.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Ellipsoid_Count (long *count);
```

count Number of ellipsoids in the ellipsoid table (output).

5.3.3 DECLARATIONS

5.3.3.1 TYPES

Not applicable.

5.3.3.2 CONSTANTS

Not applicable.

5.3.3.3 VARIABLES

Not applicable.

5.3.4 DEPENDENCIES

None.

5.3.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occured in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid table not initialized properly
```

5.4 ELLIPSOID_INDEX

5.4.1 DESCRIPTION

This function returns the index in the ellipsoid table of the ellipsoid with the specified 2-letter code.

5.4.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Ellipsoid_Index (const char *code,
                      long *index);
```

code The 2-letter code for the desired ellipsoid (input),

index The index of the ellipsoid in the table with the specified code (output).

Example:

```
status = Ellipsoid_Index(code, index)
```

Inputs:

code "ED"

Outputs:

index 11

5.4.3 DECLARATIONS

5.4.3.1 TYPES

Not applicable.

5.4.3.2 CONSTANTS

Not applicable.

5.4.3.3 VARIABLES

Not applicable.

5.4.4 DEPENDENCIES

None.

5.4.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occured in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid table not initialized properly
ELLIPSE_INVALID_CODE_ERROR      : Code was not found in table
```

5.5 ELLIPSOID_NAME

5.5.1 DESCRIPTION

This function returns the name of the ellipsoid referenced by index.

5.5.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Ellipsoid_Name (const long index,
            char *name);
```

index	The index of a given ellipsoid in the ellipsoid table (input),
name	The name of the ellipsoid referenced by index (output).

Example:

```
status = Ellipsoid_Name(index, name)
```

Inputs:

index:	11

Outputs:

name:	"Everest 1969 (West Malasia)"

5.5.3 DECLARATIONS

5.5.3.1 TYPES

Not applicable.

5.5.3.2 CONSTANTS

Not applicable.

5.5.3.3 VARIABLES

Not applicable.

5.5.4 DEPENDENCIES

None.

5.5.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occured in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid table not initialized properly
ELLIPSE_INVALID_INDEX_ERROR     : Index is an invalid value
```

5.6 ELLIPSOID_CODE

5.6.1 DESCRIPTION

This function returns the 2-letter code of the ellipsoid referenced by index.

5.6.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Ellipsoid_Code (const long index,
            char *code);
```

index	The index of a given ellipsoid in the ellipsoid table (input),
code	The 2-letter code of the ellipsoid referenced by index (output).

5.6.3 DECLARATIONS

5.6.3.1 TYPES

Not applicable.

5.6.3.2 CONSTANTS

Not applicable.

5.6.3.3 VARIABLES

Not applicable.

5.6.4 DEPENDENCIES

None.

5.6.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occured in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid table not initialized properly
ELLIPSE_INVALID_INDEX_ERROR     : Index is an invalid value
```

5.7 ELLIPSOID_AXES

5.7.1 DESCRIPTION

This function returns the lengths of the semi-major and semi-minor axes of the ellipsoid referenced by index.

5.7.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Ellipsoid_Axes (const long index,
            double *a,
            double *b);
```

index	The index of a given ellipsoid in the ellipsoid table (input),
a	The semi-major axis, in meters, of the ellipsoid referenced by index (output),
b	The semi-minor axis, in meters, of the ellipsoid referenced by index (output).

Example:

```
status = Ellipsoid_Axes(index, a,b)
```

Inputs:

index	11

Outputs:

a	6377295.664
b	6356094.668

5.7.3 DECLARATIONS

5.7.3.1 TYPES

Not applicable.

5.7.3.2 CONSTANTS

Not applicable.

5.7.3.3 VARIABLES

Not applicable.

5.7.4 DEPENDENCIES

None.

5.7.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
ELLIPSE_INVALID_INDEX_ERROR     : Index out of valid range
```

5.8 ELLIPSOID_ECCENTRICITY2

5.8.1 DESCRIPTION

This function returns the square of the eccentricity of the ellipsoid referenced by index.

5.8.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Ellipsoid_Eccentricity2 (const long index,
             double *e2);
```

index The index of a given ellipsoid in the ellipsoid table (input),

e2	The square of the eccentricity of the ellipsoid referenced by index (output).

Example:

```
status = Ellipsoid_Eccentricity2(index, e2)
```

Inputs:

index	11

Outputs:

e2	0.00663784660369520500

5.8.3 DECLARATIONS

5.8.3.1 TYPES

Not applicable.

5.8.3.2 CONSTANTS

Not applicable.

5.8.3.3 VARIABLES

Not applicable.

5.8.4 DEPENDENCIES

None.

5.8.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
ELLIPSE_INVALID_INDEX_ERROR     : Index out of valid range
```

5.9 ELLIPSOID_FLATTENING

5.9.1 DESCRIPTION

This function returns the flattening of the ellipsoid referenced by index.

5.9.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long Ellipsoid_Flattening (const long index,
             double *f);
```

index	The index of a given ellipsoid in the ellipsoid table (input),
f	The flattening of the ellipsoid referenced by index (output).

Example:

```
status = Ellipsoid_Flattening(index, f)
```

Inputs:

index	11

Outputs:

f	0.00332444929666288500

5.9.3 DECLARATIONS

5.9.3.1 TYPES

Not applicable.

5.9.3.2 CONSTANTS

Not applicable.

5.9.3.3 VARIABLES

Not applicable.

5.9.4 DEPENDENCIES

None.

5.9.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
ELLIPSE_INVALID_INDEX_ERROR     : Index out of valid range
```

5.10 WGS84_AXES

5.10.1 DESCRIPTION

This function returns the semi-major and semi-minor axis values of the WGS 84 ellipsoid.

5.10.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long WGS84_Axes (double *a,
                 double *b);
```

a The semi-major axis, in meters, of the WGS84 ellipsoid (output),

b The semi-minor axis, in meters, of the WGS84 ellipsoid (output).

5.10.3 DECLARATIONS

5.10.3.1 TYPES

Not applicable.

5.10.3.2 CONSTANTS

Not applicable.

5.10.3.3 VARIABLES

Not applicable.

5.10.4 DEPENDENCIES

None.

5.10.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
```

5.11 WGS84_ECCENTRICITY2

5.11.1 DESCRIPTION

This function returns the square of the eccentricity of the WGS 84 ellipsoid.

5.11.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long WGS84_Eccentricity2 (double *e2);
```

e2	The square of the eccentricity of the WGS84 ellipsoid (output).

5.11.3 DECLARATIONS

5.11.3.1 TYPES

Not applicable.

5.11.3.2 CONSTANTS

Not applicable.

5.11.3.3 VARIABLES

Not applicable.

5.11.4 DEPENDENCIES

None.

5.11.5 ERROR HANDLING

This function checks that the table has been initialized. The possible error codes are:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
ELLIPSE_INVALID_INDEX_ERROR     : Index out of valid range
```

5.12 WGS84_FLATTENING

5.12.1 DESCRIPTION

This function returns the flattening of the WGS 84 ellipsoid.

5.12.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long WGS84_Flattening (double *f);
```

f The flattening of the WGS84 ellipsoid (output).

5.12.3 DECLARATIONS

5.12.3.1 TYPES

Not applicable.

5.12.3.2 CONSTANTS

Not applicable.

5.12.3.3 VARIABLES

Not applicable.

5.12.4 DEPENDENCIES

None.

5.12.5 ERROR HANDLING

This function checks that the table has been initialized. The possible error codes are:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
ELLIPSE_INVALID_INDEX_ERROR     : Index out of valid range
```

5.13 WGS72_AXES

5.13.1 DESCRIPTION

This function returns the semi-major and semi-minor axis values of the WGS 72 ellipsoid.

5.13.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long WGS72_Axes (double *a,
                 double *b);
```

a	The semi-major axis, in meters, of the WGS72 ellipsoid (output),
b	The semi-minor axis, in meters, of the WGS72 ellipsoid (output).

5.13.3 DECLARATIONS

5.13.3.1 TYPES

Not applicable.

5.13.3.2 CONSTANTS

Not applicable.

5.13.3.3 VARIABLES

Not applicable.

5.13.4 DEPENDENCIES

None.

5.13.5 ERROR HANDLING

This function returns the following status codes:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
```

5.14 WGS72_ECCENTRICITY2

5.14.1 DESCRIPTION

This function returns the square of the eccentricity of the WGS 72 ellipsoid.

5.14.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long WGS72_Eccentricity2 (double *e2);
```

e2 The square of the eccentricity of the WGS72 ellipsoid (output).

5.14.3 DECLARATIONS

5.14.3.1 TYPES

Not applicable.

5.14.3.2 CONSTANTS

Not applicable.

5.14.3.3 VARIABLES

Not applicable.

5.14.4 DEPENDENCIES

None.

5.14.5 ERROR HANDLING

This function checks that the table has been initialized. The possible error codes are:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
ELLIPSE_INVALID_INDEX_ERROR     : Index out of valid range
```

5.15 WGS72_FLATTENING

5.15.1 DESCRIPTION

This function returns the flattening of the WGS 72 ellipsoid.

5.15.2 INTERFACES AND EXAMPLES

The following is a list of the formal arguments required to use this function.

```
long WGS72_Flattening (double *f);
```

f	The flattening of the WGS72 ellipsoid (output).

5.15.3 DECLARATIONS

5.15.3.1 TYPES

Not applicable.

5.15.3.2 CONSTANTS

Not applicable.

5.15.3.3 VARIABLES

Not applicable.

5.15.4 DEPENDENCIES

None.

5.15.5 ERROR HANDLING

This function checks that the table has been initialized. The possible error codes are:

```
ELLIPSE_NO_ERROR                : No errors occurred in function
ELLIPSE_NOT_INITIALIZED_ERROR   : Ellipsoid module has not been initialized
                                  properly
ELLIPSE_INVALID_INDEX_ERROR     : Index out of valid range
```

APPENDIX A STRUCTURE/DEPENDENCY DIAGRAMS

This component consists of a single compilation unit and depends only on the standard ANSI C character handling, input/output, general utility, and string handling libraries.

APPENDIX B DEFINITIONS/GLOSSARY

Eccentricity – The ratio of the distance between the center and a focus of an ellipse to the length of its semimajor axis.

Ellipsoid – The surface generated by an ellipse rotating about one of its axes.

Flattening – The ratio of the difference between the equatorial and polar radii of the Earth (semi-major and semi-minor axes) to its equatorial radius (semi-major axis).

Geoid – The equipotential surface of the earth's gravity field approximated by undisturbed mean sea level fo the oceans. The direction of gravity passing through a given point on the geoid is perpendicular to this equipotential surface.

Geoid Separation – The distance between the geoid and the mathematical reference ellipsoid as measured along the ellipsoidal normal. This distance is positive outside, or negative inside, the reference ellipsoid. Also called geoidal height; undulation of the geoid.

Reference Ellipsoid – An ellipsoid whose dimensions closely approach the dimensions of the geoid; the exact dimensions are determined by various considerations of the section of the earth's surface concerned. Usually a bi-axial ellipsoid of revolution.

Semi-Major Axis – One-half the longest diameter (equatorial axis) of an ellipsoid.

Semi-Minor Axis – One-half the shortest diameter (polar axis) of an ellipsoid.

WGS 72 – World Geodetic System 1972. WGS 72 was the previous DoD standard earth-centered, earth-fixed world geodetic system. It was superceded by WGS 84.

WGS 84 – World Geodetic System 1984. A world geodetic system provides the basic reference frame, geometric figure and gravimetric model for the earth, and provides the means for relating positions on various local geodetic systems to an earth-centered, earth-fixed coordinate system. WGS 84 is the current DoD standard earth-centered, earth-fixed world geodetic system.

APPENDIX C REFERENCES

(1) Topographic Engineering Center, TEC-SR-7, **Handbook for transformation of DATUMS, PROJECTIONS, GRIDS, AND COMMON COORDINATE SYSTEMS**, January 1996.

(2) Department of Defense, MIL-HDBK-850, **Military Handbook – Glossary of Mapping, Charting, and Geodetic Terms**, 21 January 1994.