### Title
U.S. Army Intelligence Center and School
USAICS, Software Analysis and Management
System Analysis of Geographic Transformation Algorithms.

### Authors
JET PROPULSION LABORATORY, California
Institute of Technology,
Pasadena, California

### Performing Organization Name and Address
Jet Propulsion Laboratory
ATTN: 171-209
California Institute of Technology
4800 Oak Grove, Pasadena, CA 91109

### Controlling Office Name and Address
Commander
USAICS
ATTN: ATSI-CD-SF
Ft Huachuca, AZ 85613

### Distribution Statement (of this Report)
Approved for Public dissemination

### Distribution Statement (of the abstract entered in Block 20, if different from Report)

### Supplementary Notes
Prepared by the Jet Propulsion Labs for the US Army Intelligence Center and School.

### Key Words
GEOPHIC TRANSFORMATION ALGORITHM.

### Abstract
This report describes the findings of JPL regarding geographic transformation algorithms used in MAGIC, Guardrail, Trailblazer and BETA systems. A set of parameters is developed to characterize and catalogue intelligence system algorithms in the 4 systems. Individual algorithms are also analyzed to determine if they are performing their functions properly.
Analysis of Geographic Transformation Algorithms

July 9, 1982

Approval:

H. F. Lesh, Manager
USAICS Software Analysis and Management System

H. Jeane, Manager
Software Development Section

John Radbill, Task Leader
USAICS Software Analysis and Management System

A. Ferrari, Manager
Defense Information Systems Program

Concur:

Major W. Akins, Chief
All Source Analysis System Office
U.S. Army Intelligence Center and School

This document has been approved for public release and sale; its distribution is unlimited.

JET PROPULSION LABORATORY
California Institute of Technology
Pasadena, California

UAA002
The following people contributed to this report

P. Babby
J. Gillis
A. Griesel
N. Palmer
J. Radbill
# Table of Contents

1. Introduction ............................................................................................................. 1
   1.1. Purpose .................................................................................................................... 1
   1.2. Background ............................................................................................................ 1
   1.3. User Benefits ......................................................................................................... 3

2. Analyzing the Algorithms ......................................................................................... 4
   2.1. Early Steps ............................................................................................................. 4
   2.2. Learning Military Mapping .................................................................................... 4
   2.3. Representing Algorithms in Standard Form .......................................................... 7

3. Characterizing and Cataloging the Algorithms ....................................................... 12

4. Algorithms in MAGIIC ............................................................................................... 14
   4.1. MAGIIC Lambert Constant Generation Algorithm ............................................... 14
   4.2. An Interrelated Set of MAGIIC Coordinate Conversion Algorithms ....................... 14
   4.3. MAGIIC Polar Grid to Universal Polar Sterographic Algorithm ................................ 16
   4.4. MAGIIC Northing and Easting to Universal Transverse Mercator Algorithm ......... 17

5. Comparing the Algorithms Across Systems .......................................................... 19
   5.1. Spheroid Models ..................................................................................................... 19
   5.2. Grid Zone Generation ............................................................................................ 19
   5.3. MAGIIC Grid Zone Generation Algorithm ............................................................ 20
   5.4. Guardrail Grid Zone Generation Algorithm .......................................................... 20
   5.5. Trailblazer Grid Zone Generation Algorithm ........................................................ 21
   5.6. BETA Grid Zone Generation Algorithm .................................................................. 21

6. Discussion and Conclusions ..................................................................................... 24
   6.1. Documentation of Algorithms ................................................................................ 24
   6.2. Similarity of Functions Across Systems ................................................................. 24
   6.3. Incompleteness of Algorithms for Global Applications ........................................... 24
   6.4. Robustness of Algorithms ..................................................................................... 24
   6.5. Selection/Consolidation of Algorithms .................................................................... 25
LIST OF FIGURES

2-1: Grid Zone Designations of the Military Grid Reference System (UTM) .......... 9
2-2: Grid Zone Designations of Universal Polar Stereographic (UPS) ............. 10
2-3: Basic Plan of the 100,000-meter Square Identifications .................. 11

3-1: Algorithm Level Help File ............................................. 13

6-1: Documentation of Algorithms ........................................... 26

LIST OF TABLES

5-1: Inconsistent Spheroid Usage ........................................... 23
1. Introduction

1.1. Purpose

This report describes the findings of the Algorithm Analysis Subtask group working on the U.S. Army Intelligence Center and School (USAICS) Software Analysis and Management System task (USAMS) regarding geographic transformation algorithms used in four of the intelligence-gathering systems under USAICS cognizance. In this report a set of parameters is developed to characterize and catalogue intelligence system algorithms in four specific systems. Individual algorithms are analyzed to determine whether they are performing their functions properly. Algorithms that perform the same function in different systems are compared to determine which ones are best according to various criteria.

The algorithms examined in this report are taken from the MAGIIC, Guardrail, Trailblazer, and BETA systems. They were chosen from the approximately 41 deployed intelligence systems for which USAICS is Combat Developer because their documentation was quickly accessible and because they represented a range of algorithm applications. Geographic transformation (mapping) algorithms were chosen for this report since all four systems contain position location/description functions and many of their algorithms are unclassified.

1.2. Background

Each of the about 41 intelligence systems under USAICS cognizance employs several types of algorithms to carry out its gathering and processing of intelligence data. Two important types of these algorithms, geographic transformation and correlation, have been chosen for analysis during this year. The former translates grid zone locations, for example, from latitude-longitude to Universal Transverse Mercator (UTM), while the latter resolves many individual sitings into militarily recognizable targets based chiefly on standard statistical procedures. It is important to develop a set of parameters to characterize these algorithms so that how they should be catalogued can be

---

"Two additional reports will be submitted in FY82: a correlation analysis report and a report on possible algorithm analysis methodologies."
determined. When these activities are completed, it becomes possible to compare algorithms that perform the same function in different systems.

To begin this process, the JPL Algorithm Analysis Sub-task group has examined the geographic transformation algorithms for four of the 41 systems, namely The Mobile Army Ground Imagery Interpretation Center (MAGIIC), Guardrail, Trailblazer, and Battlefield Exploitation and Target Acquisition (BETA). These four systems chosen for more detailed study represent several intelligence data analysis functions. MAGIIC is a ground-based analysis system to assist in interpreting hard-copy images from different airborne surveillance systems, including a capability for computerized mensuration on imagery; it can also receive and analyze data from Tactical Electronic Intelligence (TEREC) collection systems and provide emitter location estimates. Guardrail uses airborne sensor platforms to collect data on Direction Finding (DF) emitters; extensive ground-based software is then used to estimate the location of the units, such as command posts, associated with these emitters. Trailblazer also uses DF data to estimate emitter location. Its sensor platforms are essentially fixed and ground-based. BETA is a Test Bed program for correlating data received from several types of sensor systems and making target nominations. Both automatic correlation and aggregation techniques and interactive graphics are used in the operator's analysis. These systems would generally be employed at Division or Corps level or at an Air Force Tactical Air Control Element (TACE) or Allied Tactical Air Force (ATAF); target nominations and tactical situation reports would be available to commanders and their staffs from Brigade through Echelons Above Corps (EAC).

USAICS has cognizance of a large number of algorithms integral to intelligence-gathering systems in various stages of development and deployment. The state of "deployment" of algorithms in the USAICS inventory ranges from that of products of research contracts not yet implemented in any system to those in fielded systems such as Trailblazer or Guardrail. In the latter systems the algorithms are documented in design documents (narrative English and equations), and/or in machine readable design language, and in code. Often not all of these forms of documentation are available for any one system. For research algorithms not yet implemented actual code, or even detailed flow charts, may not be available; thus analysis must rely solely on mathematical descriptions.
"Algorithm" means any set of rules for carrying out a single con cep-
tual operation on a set of data, such as transforming latitude-longitude co-
ordinates to UTM or determining a position from a number of direction measure-
ments taken at known points. Algorithms are often hierarchical, lower-level
algorithms often being used to describe higher-level algorithms and thereby
illuminating their underlying logical structure. Thus, results from one algo-
rithm may be data for another. USAICS is interested in algorithms performing
intelligence data processing functions central to their systems' mission and
those performing crucial support functions, such as geographic location, common
to a number of systems. Data management or mathematical function algorithms,
although vital to the efficient functioning of the systems, are not being
treated in these first algorithm analyses.

1.3. User Benefits

These analyses can benefit users in several ways. First, a catalog of
existing algorithms will help USAICS avoid having algorithms redeveloped for
new systems from first principles. Second, analysis of individual algorithms
may, in a few cases, identify deficiencies worth correcting on the next system
revision. Third, and most important, the comparison of algorithms performing
the same function in different systems can lead to identifying guidelines for
developing and/or selecting algorithms to include in new and revised systems.
Selected algorithms from the systems studied will begin to form a library of
intelligence algorithms with associated computer subroutines that will be
analogous to the Collected Algorithms of the Association for Computing Mach-
inery (ACM). The creation of such a library is in the spirit of Ada*, the
Department of Defense language for embedded systems, and Ada's environment.

**These conceptual models should be describable, although their tech-
nical implementation is often significantly more complicated to present.

*Ada is a trademark of the Department of Defense
2. Analyzing the Algorithms

2.1. Early Steps

Since the Location and Movement Analysis System (LAMAS) system documentation was available first, our early analysis efforts were directed to that system. A preliminary analysis of a Shortest Path Algorithm was done and modeled in Pascal as an approach to standardizing representations. This algorithm was a variant of Dijkstra's Shortest Path Algorithm.

Later the sponsor decided that our first emphasis should be on the coordinate conversion algorithms of the MAGIC, Guardrail, Trailblazer and BETA systems. Three approaches to this analysis were tried and evaluated. The MAGIC system has been hierarchically analyzed for the interrelationship of the algorithms. The spheroid models of the Earth's oblateness have been examined for all the systems. The grid zone generation algorithms have been compared across all four systems.

2.2. Learning Military Mapping

To analyze the first type of algorithm required learning the military grid system. This discussion identifies the various map projections and military grid reference systems examined and how they are interrelated. The scope of this discussion is limited to only those map projections and grid reference systems pertinent to the MAGIC, BETA, Guardrail, and Trailblazer systems.

The map projections discussed are the Transverse Mercator, Polar Stereographic, Lambert Conformal Conic, and the Gnomonic. The grid reference systems used are the Universal Transverse Mercator (UTM), Universal Polar Stereographic (UPS), and Military Grid Reference System (MGR). The selection of a map projection is based on the properties it preserves in the transformation from a three-dimensional spheroid to a two-dimensional plane. These properties include orthogonality of latitude and longitude, equal area representation, distortion of shape, minimal change in scale factor in either east-west or north-south directions, and representation of great circles by straight lines. Since all map projections are from a spheroid model of the Earth, the parameters that the spheroid model use are very important. Various spheroid models are used for different portions of the Earth.
The selection of a grid reference system depends on the portion of the Earth examined and the resolution desired. The UTM and UPS coordinate systems were adopted as standards by the military to minimize coordination problems due to the proliferation of locally-used grid reference systems. These coordinate systems are most suitable for the representation of large geographic areas (greater than 9° in latitude and longitude). The MGR system provides greater resolution when representing smaller geographic areas (within 100,000-meter by 100,000-meter squares). The MGR system can be overlaid on the UTM and UPS coordinate systems to eliminate ambiguity due to repetitions of the 100,000-meter square identifiers. The geographic reference system is simply given as a longitude-latitude pair. However, this reference system of zones is cumbersome for representing locations in good resolution. Also, there is an inconsistency in the form of the coordinates: some applications use decimal degree notation while others use clock-like representations.

The UTM grid reference system is valid for all longitudes over latitudes between 84° North and 80° South. This area is divided into rectangles of 6° in longitude (zones) by 8° in latitude (bands), except for the 12° band from 72° to 84° latitude. There are 60 zones numbered from 1 through 60 for the zones from -180° to +180° longitude. There are 22 bands lettered C through X for the bands from -80° to 84° latitude. There are some subtle irregularities in this pattern beyond 56° latitude between 0° and 45° in longitude (see Figure 2-1). The UTM grid reference system is based on the Modified Transverse Mercator projection, but can be mathematically transformed for use with other types of projections.

The UPS grid reference system is valid for the North Polar (+84° longitude to the pole) and the South Polar regions (-80° longitude to the pole). These regions have a grid zone number of zero and consist only of a grid zone letter that is longitude-dependent. The North Polar region grid zone letters are Y (Western hemisphere) and Z (Eastern Hemisphere). The South Polar regions are A and B. The UPS grid reference system is based on the Polar Stereographic projection (see Figure 2-2).

The MGR grid reference system, illustrated for the UPS system in Figure 2-2 and for the UTM system in Figure 2-3, provides finer resolution than the UTM or UPS grid reference systems. It identifies 100,000-meter by 100,000-
meter squares by two letters, an Easting letter and a Northing letter. These letters are sequenced so as to provide at least 18° separation between similarly-lettered squares (within a given spheroid model area - otherwise the separation is 9°). These lettering sequences are biased and restarted at the boundaries of the underlying spheroid models.

These MGR system letter designations may be used without reference to the UTM or UPS designations when there is no likelihood of ambiguity, otherwise the UTM or UPS designation is included. Positions within these squares can be interpolated in tens of meters and are referred to as Easting and Northing terms representing distances rather than degrees.

The Transverse Mercator projection transforms the Earth's spheroid onto a cylinder secant to the Earth and perpendicular to its axis. This projection is used at latitudes within 84° North and 80° South. Scale linearity is correct at the two meridians cut by the cylinder (6° apart) and quite accurate in the band formed by them. Because of the vertical linearity this projection is particularly suitable to areas of interest in the North-South direction. This projection lends itself well to being overlaid with a rectangular grid reference system (such as the MGR system).

The Polar Stereographic projection transforms the Earth's spheroid onto a plane tangential to the Earth (at the pole, in our applications). This projection is used at latitudes beyond 84° North and from 80° South. Scale linearity decreases and equal area exaggeration increases as the distance from the pole increases. Latitude-longitude orthogonality is preserved at the meridian crossings. All circles of latitude are concentric, centered at the pole. Thus, this projection is useful for plotting radio waves and air navigation with a compass.

The Lambert Conformal Conic projection transforms the Earth's spheroid onto a cone parallel to the Earth's axis and secant to the earth at two latitudes referred to as the standard latitudes. The East-West scale linearity is correct at the two standard latitudes and is relatively accurate in the band between these latitudes. The projection preserves direction and shape quite well within and near the standard latitudes. Hence, the Lambert, Conformal Conic Projection is best suited to East-West measurements and is useful for air navigation. Also, all meridians are straight and intersect at the pole. This
projection is most applicable to the mid-latitude region where the cone is secant to the Earth.

The Gnomonic projection transforms the Earth's spheroid onto a plane tangent to the Earth's at the point of interest. This projection is valid over all latitudes and longitudes. It has the quality of representing all great circle arcs on the projection as straight lines. Since electromagnetic waves travel the shortest distance route (great circle arc), the Gnomonic projection is ideally suited for the presentation of direction-finding lines of bearing.

2.3. Representing Algorithms in Standard Form

To compare algorithms across systems, all algorithms analyzed must be translated into a standard format. Algorithms in the systems analyzed had been coded in such diverse languages as assembly and structured FORTRAN so that translation into a common Higher-Order Language (HOL) became essential to searching for common and diverse features. Publication ALGOL was seen as an attractive candidate because it has been used in the collected algorithms of the ACM for algorithm description. However, audiences outside the Applied Mathematics, Numerical Analysis, and Computer Sciences communities are generally unfamiliar with ALGOL; and compilers for ALGOL 60 or ALGOL 68 are not readily available in this country.

Pascal, an ALGOL-like language, has been chosen for the primary representation language for the algorithms because it has many of the properties of ALGOL (structure, strong typing, etc.), it has become familiar to a wide audience, and high quality compilers are available on many computers including the Digital Equipment Corp. VAX and many microcomputers with CPM operating systems. The last point is important because the VAX will be used by both USAICS and JPL, and the microcomputers are similar to the word processors and personal computers at JPL and USAICS. Among the features of Pascal that contribute to its clarity are the command structures, such as "if-then-else" and "case", and the user-defined data types. However, separate compilations of procedures to support hierarchical descriptions of algorithms are an implementation-dependent extension rather than a basic feature of the language. Because of this and other problems Pascal provides at best an interim solution to the algorithm description problem.
Ada offers a long-term solution. The Ada language avoids many of the shortcomings of Pascal and has many additional features. A stronger reason for using Ada is that the Army is likely to require all new systems initiated after 1984 to be programmed in it. While no complete compiler for Ada is currently available, there is an interpreter on the project VAX computer, although it is very slow. A compiler for an incomplete implementation available on Z80-based microcomputers with CPM operating systems is also available. Although this compiler is not completely satisfactory because of the lack of user-defined types, it is still useful for some simple examples and for comparison with Pascal.
Figure 2-1: Grid Zone Designations of the Military Grid Reference System (UTM)
Figure 2-3: Basic Plan of the 100,000-meter Square
Identifications of the U.S. Army Military
Grid Reference System, between 84° N. and 80° S
3. Characterizing and Cataloging the Algorithms

The set of properties stored for algorithms in the database should characterize an algorithm for military application purposes without requiring that the algorithm itself be retrieved and examined. The algorithm property selection is influenced by the following ways the database will be used. The user may wish a general summary of what algorithms are in the database. A more likely use is to look for algorithms that perform a specified function, such as position location. In another dimension, the user may ask "What algorithms do we have in MAGIC?". Cataloging properties should be independent for efficiency of description. Two of the properties chosen, performance and robustness, are not totally independent. Requirements performance is interpreted here as, "does the algorithm do what it says it does?". Lack of robustness is interpreted here as failure for special value or failure because of small errors in input data. Range checking of input variables is an important contributor to robustness. This is particularly important if the algorithm is to be used for more than one system where the calling programs can not be expected to protect it.

The Algorithm Level Help File; (Figure 3-1) taken from the Acquisition and Database Entry Prompting Tool (ADEPT) User's Guide, is examined and interpreted as a means for describing the present classification scheme. This information is provided for each system in which an algorithm is implemented. This set of parameters is likely to change as more experience is gained with its use.

Examples of PSA reports are shown in Appendix 7.1.
NAME is a name description of the algorithms function.

SYNONYM is the algorithms abbreviation (the same as its VAX file element name).

SOURCE:AUTHOR is the document from which the algorithm was taken and author, if known.

PROCESSING is what JPL has done with the algorithm, e.g. Pascal program tested.

MATH:FIELD is what mathematical field the algorithm is based on, e.g. least squares.

ROBUSTNESS is a measure of sensitivity to values of variables input to the algorithm, e.g. a transformation algorithm may produce an incorrect result with inputs of $\pm 180^\circ$ longitude.

TREE:LEVEL is a rough indication of location of the algorithm in the hierarchy of algorithms in a system.

REQUIREMENTS/PERFORMANCE Since the requirements documents for particular algorithms are generally not available, requirements must be derived from design documents or comments in code. This item describes how well the algorithm meets these requirements.

REFERENCE is a pointer to the VAX file containing the representation of the algorithm in standard form: Pascal and in some cases Ada. (These are given in Appendix 7.3).
4. Algorithms in MAgIIC

The MAgIIC System Lambert Constant Generation algorithm was analyzed and modeled in Pascal. This algorithm is required for analyzing and modeling the Geographic to Lambert/Polar Grid and Lambert/Polar Grid to Geographic conversions.

The MAgIIC system coordinate conversions were studied, and two interrelated algorithms were analyzed and modeled in Pascal. These were the Polar Grid to UPS and Northing and Easting to UTM conversions. Several inconsistencies have been noted (perhaps only because of the technical writing). Modeling these interrelated algorithms in Pascal led to the decision to use a non-standard Pascal feature - the VAX Pascal external procedure capability. This was considered necessary to best maintain structural integrity, accuracy and clarity in the algorithm representation.

4.1. MAgIIC Lambert Constant Generation Algorithm

The MAgIIC Lambert Constant Generation Algorithm, described in document CG108100A, dated 23 October 1978, paragraph 3.2.119, page 210, has insufficient input parameters and a lack of detail on setting the hemisphere flags to implement the algorithm in Pascal without making several assumptions. If we assume the availability of the underlying spheroid parameters, the Lambert Constant Generation Algorithm performs satisfactorily. The hemisphere flag issue remains unsettled: are they north-south hemispheres, east-west hemispheres, or both? - all three selections make sense in different contexts.

This algorithm has been modeled in Pascal on the VAX computer for uniformity of presentation and for comparison and analysis. All assumptions are included in comments in this Pascal representation (see Appendix 7.3).

4.2. An Interrelated Set of MAgIIC Coordinate Conversion Algorithms

There are a set of four interrelated coordinate conversion algorithms for MAgIIC that warrant a consolidated discussion because although they are pair-wise reciprocal and criss-cross "call" each other, their input and output parameters are specified inconsistently. These algorithms viewed as reciprocal pairs are:
They can also be viewed as "criss-cross" calling algorithms as follows:

1. Polar Grid to UPS
   (paragraph 3.2.118) each calls the other
2. Northing and Easting to UTM
   (paragraph 3.2.86) each calls the other
3. UPS to Polar Grid
   (paragraph 3.2.117) each calls the other
4. UTM to Northing and Easting
   (paragraph 3.2.85) each calls the other

Analysis of these algorithms and their interrelationships has revealed the following inconsistencies:

1. The Polar Grid to UPS algorithm should have an output list consistent with the UPS to Polar Grid algorithms input list since they are reciprocal functions. The lists are not consistent;

2. The Northing and Easting to UTM algorithm and the UTM to Northing and Easting algorithm, similarly, have inconsistent output and input lists;

3. In both of the above cases the same problems are true when the algorithms are used in the "criss-cross call" sense.

These points will be discussed in greater detail below in the independent analysis of the four algorithms.
4.3. MAGIIC Polar Grid to Universal Polar Sterographic Algorithm

The MAGIIC Polar Grid to Universal Polar Sterographic (UPS) Algorithm, described in document CG108100A, dated 23 October 1978, paragraph 3.2.118, page 209, fails to produce the correct output data.

This algorithm first determines if the UPS or UTM grid reference system is applicable based on the input grid zone designation. The algorithm that handles the UTM conversion is discussed as part of the Northing and Easting to Universal Transverse Mercator conversion algorithm (paragraph 3.2.86 of the previously referenced document).

In an attempt to render this algorithm amenable to analysis and modeling in Pascal several assumptions have been made (based on our interpretation of the composite of various inferences from paragraph 3.2.86, .87, .117, .118). These assumptions are:

1. The input/output lists consist of decimal and integer numbers, and alphabetic letters. There are no wholesale conversions of the input/output lists from and to ASCII representation,

2. The input/output lists are exchanged with the Northing and Easting to UTM algorithm for processing, when the input grid zone letter is from C through X (not in either polar region). Otherwise, this algorithm performs the processing (for the polar regions).

The following discussion only treats the UPS coverage area where a conversion from PS to UPS should be made. This conversion is handled within the Polar Grid to Universal Polar Sterographic conversion algorithm (paragraph 3.2.118). This algorithm is defective in producing disallowed 100,000 meter squares letter pairs.

The North Polar Area (≥ 84° N) and South Polar Area (> 80° S) are referenced using the UPS grid zone reference system with the grid zone number set to zero followed by one of the grid zone letters A, B, Y, or Z. Letter pairs represent the 100,000-meter squares which overlay the grid zone designations. It is in the lettering of these squares that this algorithm fails.
The descriptions of how to obtain the Easting letter $Le$ from the index $IE$, and the Northing letter $LN$ from the index $IN$ are merely statements that the functions should be performed without any details. Perhaps an elaborated description of the details of these functions would lead to satisfactory conversions, if these details were furnished.

4.4. MAGIIC Northing and Easting to Universal Transverse Mercator Algorithm

The MAGIIC Northing and Easting to Universal Transverse Mercator (UTM) Algorithm as described in document CG18100A, dated 23 October 1978, paragraph 3.2.86, page 156, appears to produce the correct output data on the UTM map segment available for reference. It will certainly fail in grid zones 31V, 32X, 34Z, and 36X and should be further evaluated.

This algorithm first determines, whether the UTM or UPS grid reference system is applicable based on the input grid zone designation. The algorithm that handles the UPS conversion is discussed as part of the Polar Grid to Universal Polar Sterographic conversion algorithm (paragraph 3.2.118 of the previously referenced document).

In an attempt to render the algorithm amenable to analysis and to model in Pascal several assumptions have been made (based on our interpretation of the composite of various inferences from paragraph 3.2.86, .87, .117, .118). These assumptions are:

1. The input lists consist of decimal and integer numbers and alphabetic letters. There is no wholesale conversion of the input list to ASCII representation. The output lists are entirely in ASCII representation.

2. The input/output lists are exchanged with the Polar Grid to UPS algorithm for processing, when the input grid zone letter is not from C through X (for the polar regions). Otherwise, this algorithm performs the processing (for the non-polar regions).

Only the conversion to the UTM coverage area, that is, the actual conversion handled within this Northing and Easting to Universal Transverse Mercator conversion algorithm (paragraph 3.2.86), is discussed here. This
algorithm is defective in producing grid zone designation 31V, which should be truncated at 3° E, and the three non-existent grid zone designations 32X, 34X, and 36X.
5. Comparing the Algorithms Across Systems

5.1. Spheroid Models

Because the Earth is not a perfect sphere, it is modeled as a spheroid. Since this underlying spheroid model of the Earth's oblatness spans most of the coordinate conversion algorithms, the use of various spheroid models was investigated for all four systems. It was found that twelve different spheroid models were used among or in the four systems raising the possibility of inconsistencies that may hamper inter-system communication. Of these spheroid models five were used in common by all four systems. The remaining seven spheroid models were not available in all systems, as illustrated in Table 1. Some of these partially-shared spheroid models may be equivalent, but this cannot be determined until the code is available for all four systems.

5.2. Grid Zone Generation

The Grid Zone Generation algorithms were analyzed for the MAGIIC, Guardrail, Trailblazer, and BETA systems. The Guardrail Grid Zone algorithm text description is consistent with the Trailblazer text description and computer code. The MAGIIC text description differs from the Guardrail and Trailblazer descriptions and would tend to produce less efficient runtime code, but would economize memory. All three of these systems' algorithms would produce the same flawed results, except BETA which fails badly at the upper latitudes. These three versions of the Grid Zone Generation algorithm have been modeled in Pascal.

The BETA Grid Zone Generation algorithm handles details of grid zone generation more completely. The grid zone number calculations handle input longitude beyond -180° and at 180° and beyond whereas the three other systems would fail. The grid zone letter calculations handle the special conditions of grid zone truncation for grid description 31V and the non-existence of grid designations 32X, 34X, and 36X.

Pascal implementations of these algorithms are given in Appendix 7.3. The underlying assumptions are given in the comments included in the code.
5.3. MAGIIC Grid Zone Generation Algorithm

The MAGIIC Grid Zone Generation Algorithm as described in document CG108100A, dated 23 October 1978, paragraph 3.2.90, page 167 has been analyzed and found to handle the following five areas incorrectly:

1. the upper limit of longitude (180° E),
2. the latitudes $\geq 80^\circ$ N (considerably below the upper limit, 84° N), where it provides erroneous data,
3. the truncated grid zone 31V,
4. the non-existent grid zones 32X, 34X, and 36X,
5. the regions beyond the stated longitude and latitude limits, where it fails catastrophically.

In general, perhaps due to technical writing, there are many errors of omission and/or commission where the criteria for certain algorithm parts are left unstated; for example, a text states, "If the latitude is 84° north or greater, or 80° south or greater, the grid zone number (sic) shall be set to Y or Z or to A or B, respectively. The grid zone number shall be set to zero."

5.4. Guardrail Grid Zone Generation Algorithm

The Guardrail Grid Zone Generation Algorithm as described in document ESL-TM928, dated 15 September 1979, paragraph 16.6.2.1, page 16-192 fails in five areas:

1. at longitudes equal to and beyond 180° E and beyond 180° W,
2. at latitudes equal to and beyond 80° N and beyond 80° S,
3. at the truncated grid zone 3IV,
4. at the non-existent grid zones 32X, 34X, and 36X,
5. beyond the stated longitude and latitude limits, where it fails catastrophically.

5.5. Trailblazer Grid Zone Generation Algorithm

The Trailblazer Grid Zone Generation Algorithm is described in the well-commented ROLM assembly language listings. This algorithm, extracted from the code for the GP2UM subprogram dated 20 February 1981, fails in five areas:

1. at longitudes equal to and beyond 180° E and beyond 180° N,
2. at latitudes equal to and beyond 80° N and beyond 80° S,
3. at the truncated grid zone 3IV,
4. at the non-existent grid zones 32X, 34X, and 36X,
5. beyond the stated longitude and latitude limits.

Because the hierarchical program structure is not yet available for Trailblazer, it is possible that some or all of these problems are handled adequately in higher levels of the program structure.

5.6. BETA Grid Zone Generation Algorithm

The BETA Grid Zone Generation Algorithm, described in document SS22-43, Appendix IV, page II-474 for the ADSONU subprogram, and page II-45D for the ADSCCM subprogram, was in Structured FORTRAN with in-line coding. The "INCLUDE" subprogram ZDBPRO was missing so some "reasonable" assumptions were made about it.

This algorithm performs as specified and effectively handles the following:

1. Grid zone wrap-around (longitudes >180° W or ≥ 180° E),
2. North and South Polar Regions (latitudes ≥ 84° N or > 80° S),

-21-
3. Truncating grid zone 31V,

4. The non-existent grid zones 32X, 34X, and 36X.
<table>
<thead>
<tr>
<th>Spheroid Model</th>
<th>System</th>
<th>Technical Manual</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BETA</td>
<td>MAGIIC</td>
</tr>
<tr>
<td>Clark 1866</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>International</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Clark 1880</td>
<td>X</td>
<td>x</td>
</tr>
<tr>
<td>Everest</td>
<td>X</td>
<td>x</td>
</tr>
<tr>
<td>Bessel</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Australian</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Walbeck</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fisher</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Krasovsky</td>
<td></td>
<td></td>
</tr>
<tr>
<td>World Geodetic</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Airy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Malayan</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reference

1. DD2642, dtd 20 Feb 81, pg 263
2. CG1808100A, dtd 23 Oct 78, pg 168
3. ESL-TM929, dtd 15 Sep 79, pg 15-158
4. TM32-5811-022-10-0
6. Discussion and Conclusions

6.1. Documentation of Algorithms

Only a design document has been available for the MAGIIC system; and the code has not been available, as shown in Figure 6-1. Therefore, some of the apparent deficiencies in the algorithms may be due to poor technical writing and may not exist in the code itself. For the Guardrail system, tapes containing code have been available, but the printouts obtained to date have been largely unreadable so that suppositions made from the documentation could not be confirmed from the code. In the case of Trailblazer, code is available, but the structured overview expected from documentation is not available.

6.2. Similarity of Functions Across Systems

The functions performed by the geographic transformation algorithms are found to be basically the same across the four systems examined, although the functions are implemented in slightly different ways.

6.3. Incompleteness of Algorithms for Global Applications

The MAGIIC, Guardrail, and Trailblazer transformation algorithms do not account for all the vagaries of the military grid system. Only the BETA algorithms account for all regions and boundaries. The former systems may ensure that "bad" arguments are never passed to these algorithms, so that no anomalies would occur in overall system performance. However, to develop a library of algorithms shared by many systems requires that algorithms internally protect themselves from "bad" input data.

6.4. Robustness of Algorithms

All systems but BETA fail to check for limits of latitude and longitude. This may arise from a common tendency to focus attention on certain areas of the world, e.g. Western Europe. This tendency is especially inappropriate when developing software that may well outlive any given political or geographical constraints.
6.5. Selection/Consolidation of Algorithms

The BETA grid zone generation algorithm is superior to those in the other three systems. Selection of a spheroid model for the library is not possible on the basis of the presently available data and may eventually require developing algorithms based on our experience with many different systems.
### Fig. 6-1: Documentation of Algorithms

<table>
<thead>
<tr>
<th>System</th>
<th>Documentation</th>
<th>Code Available</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAGIIC</td>
<td>Yes</td>
<td>No</td>
<td>Bad technical writing: Errors of omission and bad-quality copy. Portions of some pages unreadable. TERE task similar to Guardrail.</td>
</tr>
<tr>
<td></td>
<td>(Barely usable)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guardrail</td>
<td>Yes</td>
<td>No</td>
<td>Multiple Tasks: Program structure &quot;implied&quot; by document's structure. Flowcharts with verbal description. At least one significant technique covered by math description only, entirely included in one box at the flowchart level.</td>
</tr>
<tr>
<td></td>
<td>Good</td>
<td>Glimpses of code appear to be structured FORTRAN; most sections are missing</td>
<td></td>
</tr>
<tr>
<td>Trailblazer</td>
<td>No</td>
<td>Yes</td>
<td>Well commented code. Many similarities to Guardrail.</td>
</tr>
<tr>
<td></td>
<td>Not separately published, but basic documentation included at the beginning of each code segment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BETA</td>
<td>Yes</td>
<td>Comprehensive</td>
<td>Program structure explicitly included in documents. Some &quot;key&quot; charts not readable due to photo reductions. Code is in-line commented from Program Design Language (PDL), but we don't have the PDL</td>
</tr>
<tr>
<td></td>
<td>Good</td>
<td>Structured FORTRAN</td>
<td></td>
</tr>
</tbody>
</table>
7. Appendices
7.1. Database Entries and Products (PSL/PSA)

The three attached PSA Reports were found to be useful to our analysis. The first report is essentially the PSA Report representing our PSL input data descriptions. The second is the PSA Data Activity Interaction Matrix. It shows the interrelationships between the algorithms and their input (R) and output (D) data items. The third is part of the PSA Structure Chart and is in indented hierarchy chart for the algorithms in our PSL database.
7.2. Algorithm Hierarchy Charts
### Structure Report

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>MG Algorithms</td>
</tr>
<tr>
<td>2.</td>
<td>MG IIFLMITZN</td>
</tr>
<tr>
<td>3.</td>
<td>MG NAPCUR2NE</td>
</tr>
<tr>
<td>4.</td>
<td>MG STERDOMT</td>
</tr>
<tr>
<td>5.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>6.</td>
<td>MG IIFLMITZN</td>
</tr>
<tr>
<td>7.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>8.</td>
<td>MG IIFLMITZN</td>
</tr>
<tr>
<td>9.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>10.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>11.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>12.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>13.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>14.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>15.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>16.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>17.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>18.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>19.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>20.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>21.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>22.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>23.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>24.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>25.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>26.</td>
<td>MG NAVCORN</td>
</tr>
<tr>
<td>27.</td>
<td>MG NAVCORN</td>
</tr>
</tbody>
</table>

### Diagram

The diagram shows the structure report with various steps and their corresponding descriptions. Each step is represented by a box with a specific type of operation or algorithm.
<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MG.1ER.Sureen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>MG.BUIL.CD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MG.IIAR.BLOC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MG.BUILENARIO</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MG.HAPCUR2ME</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>6</td>
<td>MG.2P2ME</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>7</td>
<td>MG.UE2OP</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>8</td>
<td>MG.FRAME2CH</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>9</td>
<td>MG.UTR2ME</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>10</td>
<td>MG.U2ME</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>11</td>
<td>MG.PQ2UPS</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>12</td>
<td>MG.IIIMVL</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>13</td>
<td>MG.IIICOMV</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>14</td>
<td>MG.LORANCN</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>15</td>
<td>MG.NAPITIZ</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>16</td>
<td>MG.NAVCORN</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>17</td>
<td>MG.PRECSTFLE</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>18</td>
<td>MG.PRIPTDET</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>19</td>
<td>MG.ME2HAPCUR</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>20</td>
<td>MG.PRXSKRCH</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>21</td>
<td>MG.STEROMON</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
</tbody>
</table>

Version AS.2RO

SEMAPH - TEST

Utilizes Matrix

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MG.1ER.Sureen</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>MG.BUIL.CD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MG.IIAR.BLOC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MG.BUILENARIO</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MG.HAPCUR2ME</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>6</td>
<td>MG.2P2ME</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>7</td>
<td>MG.UE2OP</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>8</td>
<td>MG.FRAME2CH</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>9</td>
<td>MG.UTR2ME</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>10</td>
<td>MG.U2ME</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>11</td>
<td>MG.PQ2UPS</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>12</td>
<td>MG.IIIMVL</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>13</td>
<td>MG.IIICOMV</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>14</td>
<td>MG.LORANCN</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>15</td>
<td>MG.NAPITIZ</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>16</td>
<td>MG.NAVCORN</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>17</td>
<td>MG.PRECSTFLE</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>18</td>
<td>MG.PRIPTDET</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>19</td>
<td>MG.ME2HAPCUR</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>20</td>
<td>MG.PRXSKRCH</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>21</td>
<td>MG.STEROMON</td>
<td></td>
<td></td>
<td>U</td>
<td>U</td>
<td>U</td>
</tr>
</tbody>
</table>
Data Activity Interaction Matrix

<table>
<thead>
<tr>
<th>Data Activity</th>
<th>Interaction Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>34 MG_UTM2NE</td>
<td></td>
</tr>
<tr>
<td>33 MG_UPS2FG</td>
<td></td>
</tr>
<tr>
<td>32 MG_STEROMGHT</td>
<td></td>
</tr>
<tr>
<td>31 MG_RSTARGLOC</td>
<td></td>
</tr>
<tr>
<td>30 MG_RHSINVTL</td>
<td></td>
</tr>
<tr>
<td>29 MG_RMSFLMLTN</td>
<td></td>
</tr>
<tr>
<td>28 MG_RMSCOORDCNV</td>
<td></td>
</tr>
<tr>
<td>27 MG_RAD2DEG</td>
<td></td>
</tr>
<tr>
<td>26 MG_PROX3RCH</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data Activity</th>
<th>Interaction Matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 MG_Angle_in_Degrees/Mins/Secs</td>
<td>D</td>
</tr>
<tr>
<td>2 MG_Angle_in_Scaled_Fi_Radians</td>
<td>R</td>
</tr>
<tr>
<td>3 MG_F75T2</td>
<td></td>
</tr>
<tr>
<td>4 MG_Spheroid</td>
<td>D</td>
</tr>
<tr>
<td>5 MG_F171TS</td>
<td></td>
</tr>
<tr>
<td>6 MG_Latitude/Longitude</td>
<td></td>
</tr>
<tr>
<td>7 MG_Northings/Easting</td>
<td>D</td>
</tr>
<tr>
<td>8 MG_Grid_ZONE_No./Letter</td>
<td>D</td>
</tr>
<tr>
<td>9 MG_Map_Cursor_Coordinate</td>
<td></td>
</tr>
<tr>
<td>10 MG_UTM_Northings/Easting_Set</td>
<td></td>
</tr>
<tr>
<td>11 MG_Grid_ZONE_Letter</td>
<td>R</td>
</tr>
<tr>
<td>12 MG_N/E_Letters</td>
<td>R</td>
</tr>
<tr>
<td>13 MG_N/E_Numbers</td>
<td>R</td>
</tr>
<tr>
<td>14 MG_100K_Meter_Square_ID</td>
<td></td>
</tr>
<tr>
<td>Data Activity Interaction Matrix</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>25 MG.FRIF1BE1</td>
<td></td>
</tr>
<tr>
<td>24 MG.FREEFILE</td>
<td></td>
</tr>
<tr>
<td>23 MG.FEO2HFS</td>
<td></td>
</tr>
<tr>
<td>22 MG.MF2UPH</td>
<td></td>
</tr>
<tr>
<td>21 MG.MEN2NTR1UR</td>
<td></td>
</tr>
<tr>
<td>20 MG.MEN20F</td>
<td></td>
</tr>
<tr>
<td>19 MG.NAVCORDH</td>
<td></td>
</tr>
<tr>
<td>18 MG.MAPITZNM</td>
<td></td>
</tr>
<tr>
<td>17 MG.MARCCUR2NE</td>
<td></td>
</tr>
<tr>
<td>16 MG.LORANCNV</td>
<td></td>
</tr>
<tr>
<td>15 MG.LAMBERTCG</td>
<td></td>
</tr>
<tr>
<td>14 MG.L/FQ2OF</td>
<td></td>
</tr>
<tr>
<td>13 MG.LJAK6GLOC</td>
<td></td>
</tr>
<tr>
<td>12 MG.LJLVMTL</td>
<td></td>
</tr>
<tr>
<td>11 MG.LFLMITHZ</td>
<td></td>
</tr>
<tr>
<td>10 MG.LIIC0ODCMV</td>
<td></td>
</tr>
<tr>
<td>9 MG.GRDZONEC MV</td>
<td></td>
</tr>
<tr>
<td>8 MG.GEO2CNE</td>
<td></td>
</tr>
<tr>
<td>7 MG.GEO2/L/FG</td>
<td></td>
</tr>
<tr>
<td>6 MG.DEHSPHERE</td>
<td></td>
</tr>
<tr>
<td>5 MG.FRAMSHC</td>
<td></td>
</tr>
<tr>
<td>4 MG.DEC2RAD</td>
<td></td>
</tr>
<tr>
<td>3 MG.BUILD1C</td>
<td></td>
</tr>
<tr>
<td>2 MG.BUILD2C</td>
<td></td>
</tr>
<tr>
<td>1 MG.PEACONF</td>
<td></td>
</tr>
</tbody>
</table>

| 1 MG.Angle in Degrees/Minutes/Seconds: R |
| 2 MG.Angle in Scaled Pi, Radians: D   |
| 3 MG.F/55C: R                      |
| 4 MG.Spheroid: R                   |
| 5 MG.F/17TS: R                    |
| 6 MG.Latitude/Latitude: R         |
| 7 MG.Meridian/Easting: R            |
| 9 MG.Grid Zone No./Letter: R            |
| 8 MG.Meridian Cursor Coordinate: B | |
| 10 MG.UHR.Meridian/Easting Set: B  |
| 11 MG.Grid Zone Letter: B          |
| 12 MG.M.H/V Letters: B             |
| 13 MG.H/V Numbers: B              |
| 14 MG.100h Meter Sensor ID: B    |

---


1 MG_Algorithms
2 MG_BUILDCD
3 MG_IITARGLOC
4 MG_GENSHPERE
2 MG_BUILDSKETCH
3 MG_IITARGLOC
4 MG_GENSHPERE
3 MG_IFLMITZN
3 MG_MAPCUR2NE
4 MG_GF2NE
5 MG_GENSHPERE
5 MG_GF2L/FG
5 MG_GRDZONECNV
4 MG_GRDZONECNV
4 MG_NE2GP
5 MG_GF2NE
6 MG_GENSHPERE
6 MG_GF2L/FG
6 MG_GRDZONECNV
3 MG_FRAMESRCH
3 MG_NE2GP
4 MG_GF2NE
5 MG_GENSHPERE
5 MG_GF2L/FG
5 MG_GRDZONECNV
3 MG_UTM2NE
4 MG_GENSHPERE
4 MG_NE2GP
5 MG_GF2NE
6 MG_GENSHPERE
6 MG_GF2L/FG
6 MG_GRDZONECNV
4 MG_NE2UTM
5 MG_FG2UPS
6 MG_NE2UTM

PXA168: Loop detected (see level 4) - Structure truncated.
4 MG_FG2UPS
5 MG_NE2UTM
6 MG_FG2UPS

PXA168: Loop detected (see level 4) - Structure truncated.
4 MG_UFS2FG
3 MG_IINVTL
4 MG_IITARGLOC
5 MG_GENSHPERE
3 MG_RMSTNL
2 MG_IICORDCNV
3 MG_GF2NE
4 MG_GENSHPERE
4 MG_GF2L/FG
4 MG_GRDZONECNV
3 MG_NE2GP
4 MG_GF2NE
5 MG_GENSHPERE
5 MG_GF2L/FG
5 MG_GRDZONECNV
3 MG_UTM2NE
4 MG_GENSHPERE
4 MG_NE2GP
PSA168: Loop detected (see level 4) - Structure truncated.
4  MG_FG2UPS
5  MG_NE2UTM
6  MG_FG2UPS

PSA168: Loop detected (see level 4) - Structure truncated.
4  MG_UPS2PG
3  MG_NE2UTM
4  MG_FG2UPS
5  MG_NE2UTM

PSA168: Loop detected (see level 3) - Structure truncated.
2  MG_LORANCNV
3  MG_GENSPHERE
2  MG_MAPITZN
3  MG_LAMBERTCG
2  MG_NAVCORN
3  MG_IITARGLOC
4  MG_GENSPHERE
3  MG_IIFLHITZN
3  MG_MAP_JRONE
4  MG_GF2NE
5  MG_GENSPHERE
5  MG_GF2L_PG
5  MG_GRDZONECNV
4  MG_GRDZONECNV
4  MG_NE2GP
5  MG_GF2NE
6  MG_GENSPHERE
6  MG_GF2L_PG
6  MG_GRDZONECNV
3  MG_GF2NE
4  MG_GENSPHERE
4  MG_GF2L_PG
4  MG_GRDZONECNV
3  MG_NE2GP
4  MG_GF2NE
5  MG_GENSPHERE
5  MG_GF2L_PG
5  MG_GRDZONECNV
3  MG_UTM2NE
4  MG_GENSPHERE
4  MG_NE2GP
5  MG_GF2NE
6  MG_GENSPHERE
6  MG_GF2L_PG
6  MG_GRDZONECNV
4  MG_NE2UTM
5  MG_FG2UPS
6  MG_NE2UTM

PSA168: Loop detected (see level 4) - Structure truncated.
4  MG_FG2UPS
5  MG_NE2UTM
6  MG_FG2UPS

29-7
PSA168: Loop detected (see level 4) - Structure truncated.
4  MG_UPS2PG
3  MG_NE2UTM
4  MG_PG2UPS
5  MG_NE2UTM

PSA168: Loop detected (see level 3) - Structure truncated.
2  MG_PRECPTFLE
3  MG_RMSTARGLOC
2  MG_PRIPTDET
3  MG_IITARGLOC
4  MG_GENSPHERE
3  MG_IIFLMITZN
3  MG_MAPCUR2NE
4  MG_GP2NE
5  MG_GENSPHERE
5  MG_GP2L/PG
5  MG_GRDZONECNV
4  MG_GRDZONECNV
4  MG_NE2GP
5  MG_GP2NE
6  MG_GENSPHERE
6  MG_GP2L/PG
6  MG_GRDZONECNV
3  MG_GP2NE
4  MG_GENSPHERE
4  MG_GP2L/PG
4  MG_GRDZONECNV
3  MG_NE2GP
4  MG_GP2NE
5  MG_GENSPHERE
5  MG_GP2L/PG
5  MG_GRDZONECNV
3  MG_UTM2NE
4  MG_GENSPHERE
4  MG_NE2GP
5  MG_GP2NE
6  MG_GENSPHERE
6  MG_GP2L/PG
6  MG_GRDZONECNV
4  MG_NE2UTM
5  MG_PG2UPS
6  MG_NE2UTM

PSA168: Loop detected (see level 4) - Structure truncated.
4  MG_PG2UPS
5  MG_NE2UTM
6  MG_PG2UPS

PSA168: Loop detected (see level 4) - Structure truncated.
4  MG_UPS2PG
3  MG_NE2UTM
4  MG_PG2UPS
5  MG_NE2UTM

PSA168: Loop detected (see level 3) - Structure truncated.
3  MG_IINVTTL
4  MG_IITARGLOC
5  MG_GENSPHERE
4  MG_GENSPHERE
3  MG_RMSINVTTL
3  MG_RMSTARGLOC
3  MG_NE2MAPCUR
PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.

PSA168: Loop detected (see level 4) - Structure truncated.
**DESCRIPTION:**
This algorithm converts angle in degrees/minutes/seconds into its equivalent angle in scaled Pi radians.

**SOURCES:** CG108100A/Part-I

**SECURITY:** U

**RESP PD:** JWG

**ATTRIBUTE:**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>07/01/32</td>
</tr>
<tr>
<td>processing_done</td>
<td>none</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>trisometry</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>TRG</td>
</tr>
</tbody>
</table>
DESCRIPTION:

This algorithm determines the appropriate spheroid number corresponding to the input latitude/longitude by scanning a specially constructed map database which contains the relationship between longitude bands/latitude strips and spheroid numbers.

KEYWORDS: algorithm

SOURCES: 3.2.91 CG108100A/Part_I

SECURITY: U

RESP PD: JWG

ATTRIBUTE:

<table>
<thead>
<tr>
<th>attribute</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviation</td>
<td>Spheroid_Generation</td>
</tr>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/01/32'</td>
</tr>
<tr>
<td>processing_done</td>
<td>none</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>cartography</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>TBD</td>
</tr>
</tbody>
</table>
DESCRIPTION: This algorithm converts an input geographic coordinate latitude/longitude pair to the equivalent Lambert/Polar grid northing/easting coordinate set.

KEYWORDS: algorithm

SOURCES: 3.2.115 CG108100A/Part_1

SECURITY: U

RESP FD: JWG

ATTRIBUTE:

<table>
<thead>
<tr>
<th>ATTRIBUTE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviation</td>
<td>Lat/Long_2_Lambert/Polar_Grid</td>
</tr>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/01/82'</td>
</tr>
<tr>
<td>processing_done</td>
<td>none</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>cartography</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>TBD</td>
</tr>
</tbody>
</table>
DESCRIPTION: This algorithm converts and inputs latitude, longitude pair into its equivalent northing and easting coordinate set.

KEYWORDS: algorithm

SOURCES: 3.2.63 CG108100A/Part_I

SECURITY: U

RESP PD: JWG

ATTRIBUTE: VALUE:
abbreviation Geodetic_2_Northing/Easting
type_of_source document
date_acquired '07/01/82'
processing_done none
mathematical_field cartography
level leaf
requirements_performance TBD
DESCRIPTION:
This algorithm converts input longitude and latitude to UTM/UPS Grid Zone number and letter. It fails at the upper (closed) limits of both longitude and latitude.

KEYWORDS: algorithm

SOURCES: 3.2.90 CG106109A/Part_I

SECURITY: U

RESP PD: JWG

<table>
<thead>
<tr>
<th>ATTRIBUTE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviation</td>
<td>Grid_Zone_Generation</td>
</tr>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/01/82'</td>
</tr>
<tr>
<td>Processing_done</td>
<td>analyzed/Pascalized</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>cartography</td>
</tr>
<tr>
<td>robustness</td>
<td>3</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>generally_satisfactory</td>
</tr>
<tr>
<td>references</td>
<td>[JWG106109A/Part_I/GRIDZONE]</td>
</tr>
</tbody>
</table>
DESCRIPTION:
This algorithm converts Lambert/Polar Grid input coordinates into equivalent geographic coordinates latitude/longitude.

KEYWORDS: algorithm

SOURCES: 3.2.116

SECURITY: U

RESP PD: JWG

ATTRIBUTE:

<table>
<thead>
<tr>
<th>attribute</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviation</td>
<td>Lambert/Polar_Grid_2_Lat/Lon</td>
</tr>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/01/82'</td>
</tr>
<tr>
<td>processing_done</td>
<td>none</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>cartography</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>TBD</td>
</tr>
</tbody>
</table>
DESCRIPTION:

This algorithm calculates the Lambert constants required for use with the Lambert/Polar Grid to UPS and the UPS to Lambert/Polar Grid conversion algorithms.

KEYWORDS: algorithm

SOURCES: 3.2.119

SECURITY: U

RESP PD: JWG

ATTRIBUTE:

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>type_of_source</td>
<td>Lambert/Constant_Generation</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/31/92'</td>
</tr>
<tr>
<td>processing_done</td>
<td>analyzed/normalized</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>charts</td>
</tr>
<tr>
<td>robustness</td>
<td>leaf</td>
</tr>
<tr>
<td>tree_level</td>
<td>satisfactory</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>'EJ283LC03PR02'</td>
</tr>
<tr>
<td>references</td>
<td></td>
</tr>
</tbody>
</table>
DESCRIPTION:
This algorithm converts an NY map cursor position to its equivalent northing/easting or latitude/longitude or 3rd zone number/letter and spheroid output coordinate.

KEYWORDS: algorithm

SOURCES: 3.2.69, CG108100A/Part_I

SECURITY: U

RESP FD: JWG

ATTRIBUTE: VALUE:
abbreviation
Map_Cursor2Northing_Easting
type_of_source
document
date_acquired
'07/01/82'
processing_done
none
mathematical_field
cartography
tree_level
middle
requirements_performance
TBD
DESCRIPTION:
This algorithm converts and inputs northings/easting set into a latitude/longitude pair.

KEYWORDS: algorithm

SOURCES: 3.2.84
CG108100A/Part_I

SECURITY: U

RESP PD: JWG

ATTRIBUTE:

<table>
<thead>
<tr>
<th>attribute</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviation</td>
<td>Northings/Eastings/Lat/Lon</td>
</tr>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/01/82'</td>
</tr>
<tr>
<td>processing_done</td>
<td>none</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>cartography</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>TBD</td>
</tr>
</tbody>
</table>
DESCRIPTION:
This algorithm converts either northing/easting or latitude/longitude or grid zone number/letter pairs into equivalent map cursor position pairs for display.

KEYWORDS: algorithm

SOURCES: 3.2.38

SECURITY: U

RESP PD: JWB

ATTRIBUTE: value:
abbreviation: Northing/Easting/Zone
type_of_document: 07/01/82
acquired:
document:
processing:
none:
field:
cartography:
level:
middle:
requirements:
TED:
DESCRIPTION: This algorithm converts northing and easting to a composite UTM pair.

KEYWORDS: algorithm

SOURCES: 3.2.86 CG108100A/Part-I

SECURITY: U

RESP FD: JWG

ATTRIBUTE: VALUE:
abbreviation Northings/Eastings2UTM
type_of_source document
date_acquired '07/01/82'
processing_done none
mathematical_field cartography
level leaf
requirements_performance TBD
DESCRIPTION:
This algorithm converts Polar Grid northing/easting coordinates into equivalent Universal Polar Stereographic or Universal Transverse Mercator (utilizing the NE2UTM algorithm).

KEYWORDS: algorithm

SOURCES: 3.2.118  CG108100A/Part_1

SECURITY: U

RESP PD: JWG

ATTRIBUTE:  VALUE:
abbreviation  Polar_Grid_2_UPS
type_of_source  document
date_acquired  '07/01/92'
processing_done  analyzed/Pascalized
calculations  3
mathematical_field  leaf
robustness  TBD
tree_level  TBD
requirements_performance  TBD
references  TBD
This algorithm converts an angle in scaled \( \pi \) radians into an equivalent angle in degrees/minutes/seconds.

**SOURCES:** CG108100A/Part-1

**SECURITY:** U

**RESP PD:** JWG

**ATTRIBUTE:**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/01/92'</td>
</tr>
<tr>
<td>processing_done</td>
<td>none</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>trigonometry</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>TBD</td>
</tr>
</tbody>
</table>
DESCRIPTION: This algorithm converts Universal Polar Stereographic into equivalent Polar grid coordinates.

KEYWORDS: algorithm

SOURCES: 3.2.117, CG108100A/Part_I

SECURITY: U

RESP PD: JWG

ATTRIBUTE: VALUE:

<table>
<thead>
<tr>
<th>attribute</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviation</td>
<td>UF3_2_Polar_Grid</td>
</tr>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/01/82'</td>
</tr>
<tr>
<td>processing_done</td>
<td>none</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>cartography</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>TBD</td>
</tr>
</tbody>
</table>
This algorithm converts a UTM coordinate set into the equivalent composite northing and easting pair.

**KEYWORDS:** algorithm

**SOURCES:** 3.2.65

**SECURITY:** U

**RESP PD:** JWG

**ATTRIBUTE:**

<table>
<thead>
<tr>
<th>Value</th>
<th>VALUE:</th>
</tr>
</thead>
<tbody>
<tr>
<td>abbreviation</td>
<td>UTM_2_Northing/Easting</td>
</tr>
<tr>
<td>type_of_source</td>
<td>document</td>
</tr>
<tr>
<td>date_acquired</td>
<td>'07/31/82'</td>
</tr>
<tr>
<td>processing_done</td>
<td>none</td>
</tr>
<tr>
<td>mathematical_field</td>
<td>cartography</td>
</tr>
<tr>
<td>tree_level</td>
<td>leaf</td>
</tr>
<tr>
<td>requirements_performance</td>
<td>Ted</td>
</tr>
</tbody>
</table>
7.3. Algorithms in Standard Form
PROGRAM DriverLambertConstantGeneration (INPUT;OUTPUT);

DriverLambertConstantGeneration provides an environment for

Drivem the LambertConstantGeneration procedure

Interfaces with the LambertConstantGeneration procedure are:

GLOBAL VARIABLES
  - Eccentricity
  - SemiMajorAxis

INPUT PARAMETERS
  - LambdaL
  - LambdaR
  - PhiU
  - PhiL
  - Phi1
  - Phi2

OUTPUT PARAMETERS
  - Kuppa
  - Iota
  - ProjectionConeRadius
  - LambdaC
  - SquaredEccentricity
  - Hemisphere

Declarations

TYPE
  PiRadians = REAL;
  ZeroTuOne = REAL;
  Flas = (northern,southern);

VAR
  Eccentricity : ZeroTuOne;
  SemiMajorAxis : REAL;
  LambdaL : PiRadians;
  LambdaR : PiRadians;
  PhiU : PiRadians;
  PhiL : PiRadians;
  Phi1 : PiRadians;
  Phi2 : PiRadians;
  Kuppa : REAL;
  Iota : REAL;
  ProjectionConeRadius : REAL;
  LambdaC : PiRadians;
  SquaredEccentricity : ZeroTuOne;
  Hemisphere : Flas;

PROCEDURE LambertConstantGeneration

<GLOBAL> Eccentricity : ZeroTuOne;
<GLOBAL> SemiMajorAxis : REAL;
<IN> LambdaL : PiRadians;
<IN> LambdaR : PiRadians;
<IN> PhiU : PiRadians;
<IN> PhiL : PiRadians;
<IN> Phi1 : PiRadians;
<IN> Phi2 : PiRadians;
<OUT> VAR Kuppa : REAL;
<OUT> VAR Iota : REAL;
<OUT> VAR ProjectionConeRadius : REAL;
<OUT> VAR LambdaC : PiRadians;
<OUT> VAR SquaredEccentricity : ZeroTuOne;
<OUT> VAR Hemisphere : Flas;

EXTERN
procedure LambertConstantGeneration models the Lambert Constant

Generation algorithm described in reference 3.2.119 of document

CG108100a dated 23 October 1978.

J.W. Gillis 4-22-82

This procedure ASSUMES that certain data are available as
required by the algorithm but not described as input in the
reference document. These data are: Eccentricity Semimajor Axis

This procedure DOES NOT perform data validation checks that are
not specified in the algorithm description. This is to allow the
algorithm features to be presented more clearly.

executables

BEGIN

establish GLOBAL variables

Eccentricity := 0.51
SemimajorAxis := 10.01

establish INPUT PARAMETERS

LambdaL := 0.251
LambdaR := 0.01
PhiL := 0.1251
PhiL := 0.01
Phi1 := 0.06751
Phi2 := 0.3751

value INPUT PARAMETERS

WRITELN

WRITELN ('setup Eccentricity is 'Eccentricity')
WRITELN ('setup SemimajorAxis is 'SemimajorAxis')
WRITELN ('setup LambdaL is 'LambdaL')
WRITELN ('setup LambdaR is 'LambdaR')
WRITELN ('setup PhiL is 'PhiL')
WRITELN ('setup PhiL is 'PhiL')
WRITELN ('setup Phi1 is 'Phi1')
WRITELN ('setup Phi2 is 'Phi2')


list output parameters from IOL
12700 { WRITELN;
12800 WRITELN ('Kappa is ' Kappa);
12900 WRITELN ('Iota is ' Iota);
13000 WRITELN ('ProjectionConeRadius is ' ProjectionConeRadius);
13100 WRITELN ('LambdaC is ' LambdaC);
13200 WRITELN ('SquareEccentricity is ' SquareEccentricity);
13300 WRITELN ('Hemisphere is ' Hemisphere);
13400 { WRITELN ('DriverLambertConstantGeneration PROCEDURE
13500 { END. ('DriverLambertConstantGeneration PROCEDURE
13600 }
MODULE LambertConstantGeneration;

200  <
300  <
400  <

500  TYPE
600      zeroToOne = REAL;
700      PiRadians = REAL;
800      Flag = (northern,southern);
900  <
1000  <

1100  PROCEDURE LambertConstantGeneration
1200 < GLOBAL> Eccentricity : REAL;
1300 < GLOBAL> SemiMajorAxis : REAL;
1400 < (IN) Lambda : PiRadians;
1500 < (IN) LambdaR : PiRadians;
1600 < (IN) Phi1 : PiRadians;
1700 < (IN) Phi2 : PiRadians;
1800 < (OUT) VAR Kappa : REAL;
1900 < (OUT) VAR Iota : REAL;
2000 < (OUT) VAR ProjectionConeRadius : REAL;
2100 < (OUT) VAR LambdaC : PiRadians;
2200 < (OUT) VAR SquaredEccentricity : ZeroToOne;
2300 < (OUT) VAR Hemisphere : Flag
2400 <
2500 <
2600 <
2700 <

2800 < Generation algorithm described in paragraph 3.2.119 of document
2900 < dated 23 October 1978.
3000 <
3100 < J.W. Gillis 4-27-82
3200 <
3300 < This procedure ASSUMES that certain data are available as
3400 < required by the algorithm but not described as inputs in the
3500 < reference document. These data are: Eccentricity
3600 < SemiMajorAxis
3700 <
3800 < This procedure DOES NOT perform data validation checks that are
3900 < not specified in the algorithm description. This is to allow the
4000 < algorithm features to be presented more clearly.
4100 <
4200 <
4300 <
4400 < CONST
4500 PiOver2 = 1.57079;
4600 <
4700 VAR
4800 Phi : PiRadius;
4900 PhiPrime : PiRadius;
5000 Zeta : REAL;
5100 Curvature : REAL;
5200 Index : INTEGER;
5300 BEGIN
5400 <
5500 Phi[1] := Phi1;
5600 Phi[2] := Phi2;
5700 Phi[3] := (Phi1+Phi2)/2.0;
5800 <
5900 LambdaC := (Lambda + LambdaR)/2.0;
6000 <
6100 SquaredEccentricity := SQRT(Eccentricity);
6200 <
6300 FOR Index := 1 TO 3 DO 30-4
PhiPrime(Index) := ARCTAN((1.0-SquaredEccentricity) * 
(SIN(ABS(Phi[Index])) / 
COS(ABS(Phi[Index]))));

FOR Index := 1 TO 2 DO
  Curvature[Index] := SemimajorAxis * 
(SQR((1.0-SquaredEccentricity) * 
SQR(ABS(ABS(Phi[Index])))));

FOR Index := 1 TO 3 DO
  Zeta[Index] := PiOver2-PiPrime[Index];

Iota := (LN(COS(ABS(Phi[1]))) - LN(COS(ABS(Phi[2])))) + 
LN(Curvature[1]) - LN(Curvature[2]);

Iota := (LN(SIN(Zeta[1]/2.0)/COS(Zeta[1]/2.0)) - 
LN(SIN(Zeta[2]/2.0)/COS(Zeta[2]/2.0)));

Kappa := (Curvature[1]/COS(ABS(Phi[1]))) * 
(Iota*(SIN(Zeta[1]/2.0)/COS(Zeta[1]/2.0))**Iota));

ProjectionConeRadius := Kappa*(SIN(Zeta[3]/2.0)) / 
COS(Zeta[3]/2.0)**Iota));

< NO ALGORITHM is available for the Hemispheric Focal set >

< list intermediate values >

FOR Index := 1 TO 3 DO
  WRITELN('
  WRITELN('computed Phi[3] in 'rPhi[3]);
  FOR Index := 1 TO 3 DO
    WRITELN('computed PhiPrime[Index] is 'rPhiPrime[Index]);
  FOR Index := 1 TO 2 DO
    WRITELN('computed Curvature[Index] is 'rCurvature[Index]);
  FOR Index := 1 TO 3 DO
    WRITELN('computed Zeta[Index] is 'rZeta[Index]);

END;  < LambertConstantGeneration PROCEDURE >

END.  < Legendre MODULE >
PROGRAM Drvit (INPUT,OUTPUT);  

TYPE  
  Meters = REAL;  
  Decameters = REAL;  
  Letter = 'A'..'Z';  
  Spheres = INTEGER;  

VAR  
  FGNorthiriCoord : Meters;  
  PGEastingCoord : Meters;  
  FGZoneLetter : Letter;  
  FGZoneNumber : INTEGER;  
  SpheroidNumber : Spheres;  
  UPSNorthingLetter : Letter;  
  UPEastingLetter : Letter;  
  UPSNorthingNumber : INTEGER;  

PROCEDURE PolarToUPS  
  (IN  
    FGNorthiriCoord : Meters;  
    PGEastingCoord : Meters;  
    FGZoneLetter : Letter;  
    FGZoneNumber : INTEGER;  
    SpheroidNumber : Spheres;  
  )  
  (OUT  
    UPSNorthingLetter : Letter;  
    UPEastingLetter : Letter;  
    UPSNorthingNumber : Decameters;  
    UPSZoneLetter : Letter;  
    UPSEastingsLetter : Letter;  
    UPSNorthingNumber : Decameters);  
BEGIN  
  WRITELN (' ENTER PG NORTHING COORD ');  
  READLN (PGNorthiriCoord);  
  WRITELN (' ENTER PG EASTING COORD ');  
  READLN (PGEastingCoord);  
  WRITELN (' ENTER PG ZONE NUMBER ');  
  READLN (FGZoneNumber);  
  WRITELN (' ENTER PG ZONE LETTER ');  
  READLN (FGZoneLetter);  
  WRITELN (' ENTER SPHEROID NUMBER ');  
  READLN (SpheroidNumber);  
  PolarToUPS (FGNorthiriCoord,  
               PGEastingCoord,  
               FGZoneLetter,  
               FGZoneNumber,  
               SpheroidNumber,  
               UPSNorthingLetter,  
               UPSEastingsLetter,  
               UPSNorthingNumber);  
END.  

3D-6
6400 END. (of PROGRAM Dr vit)
PROGRAM POL2UPS (INPUT,OUTPUT);

TYPE
  Meters = REAL;
  Decimeters = REAL;
  Letter = 'A'..'Z';
  Spheres = INTEGER;

VAR
  UTMZoneLetter : Letter;
  UTEastingsLetter : Letter;
  UTMNorthingsLetter : Letter;
  UTEastingsNumber : Decameters;

PROCEDURE PolarToUPS
  (IN ) PGNorthingsCoord : Meters;
  (IN ) PGEastingsCoord : Meters;
  (IN ) PGZoneLetter : Letter;
  (IN ) PGZoneNumber : INTEGER;
  (IN ) SpheroidNumber : Spheres;
  (OUT) VAR UPSZoneLetter : Letter;
  (OUT) VAR UPEastingsLetter : Letter;
  (OUT) VAR UPSNorthingsLetter : Letter;
  (OUT) VAR UPEastingsNumber : Decameters;

PROCEDURE PolarToUPS models the Polar Grid to Universal Polar
Steriographic conversion algorithm described in "The Universal Polar
This procedure ASSUMES that certain data are available as
required by the algorithm but not adequately described in the
algorithm description.
This procedure DOES NOT perform data validity checks that are
not specified in the algorithm description. This is to allow
the algorithm features to be presented more clearly.
PROCEDURE PolarToUPS accepts northings and eastings coordinates;
tests whether the UPS or UTM grid system is a suitable target
system. If the UPS grid system is appropriate, the conversion is performed by this procedure. If the UTM grid system is appropriate, then this procedure CALLs the
"PolarToUTH" procedure (algorithm 3.2.36) to perform the conversion.

TYPE
  Meters = REAL;
  Decimeters = REAL;
  Letter = 'A'..'Z';
  Spheres = INTEGER;

VAR
  UTMZoneLetter : Letter;
  UTEastingsLetter : Letter;
  UTMNorthingsLetter : Letter;
  UTEastingsNumber : Decameters;

PROCEDURE PolarToUPS CALLs: PROCEDURE ConvertToUPS
  PROCEDURE ConvertToUTH

Programmed by J.W.Silvis 5-5-82

This procedure assumes that certain data are available as
required by the algorithm but not adequately described in the
algorithm description.
This procedure does not perform data validity checks that are
not specified in the algorithm description. This is to allow
the algorithm features to be presented more clearly.

PROCEDURE ConvertToUPS accepts northings and eastings coordinates;
tests whether the UPS or UTM grid system is a suitable target
system. If the UPS grid system is appropriate, the conversion is performed by this procedure. If the UTM grid system is appropriate, then this procedure CALLs the
"PolarToUTH" procedure (algorithm 3.2.36) to perform the conversion.

TYPE
  Meters = REAL;
  Decimeters = REAL;
  Letter = 'A'..'Z';
  Spheres = INTEGER;

VAR
  UTMZoneLetter : Letter;
  UTEastingsLetter : Letter;
  UTMNorthingsLetter : Letter;
  UTEastingsNumber : Decameters;
PROCEDURE ConvertToUTM
0000 ((IN ) FGNorthingsCoord :Meters;
0000 (IN ) FGEastingsCoord :Meters;
0000 (IN ) FGZoneLetter :Letter;
0000 (IN ) FGZoneNumber :INTEGER;
0000 (OUT) VAR UTMZoneLetter :Letter;
0000 (OUT) VAR UTMEastingsLetter :Letter;
0000 (OUT) VAR UTMNorthingsLetter :Letter;
0000 (OUT) VAR UTMEastingsNumber :Decameters;
0000 (OUT) VAR UTMNorthingsNumber :Decameters);EXTERN;
0000 (IN ) PGZoneNumber :INTEGER;
0000 (IN ) PGEastingsCoord :Meters;
0000 (IN ) PGNorthingsCoord :Meters;
0000 (IN ) PGZoneLetter :Letter;
0000 (OUT) VAR UPSZoneLetter :Letter;
0000 (OUT) VAR UPSEastingsLetter :Letter;
0000 (OUT) VAR UPSNorthingsLetter :Letter;
0000 (OUT) VAR UPSEastingsNumber :Decameters;
0000 (OUT) VAR UPSNorthingsNumber :Decameters);EXTERN;

PROCEDURE ConvertToUPS
0000 ((IN ) FGNorthingsCoord :Meters;
0000 (IN ) FGEastingsCoord :Meters;
0000 (IN ) FGZoneLetter :Letter;
0000 (IN ) FGZoneNumber :INTEGER;
0000 (OUT) VAR UPSZoneLetter :Letter;
0000 (OUT) VAR UPSEastingsLetter :Letter;
0000 (OUT) VAR UPSNorthingsLetter :Letter;
0000 (OUT) VAR UPSEastingsNumber :Decameters;
0000 (OUT) VAR UPSNorthingsNumber :Decameters);EXTERN;

BEGIN
9000 WRITELN ('FG2UPS ALL HOOKED UP');
9100 IF PGZoneNumber = 0
9200 THEN ConvertToUPS ((OUT) FGNorthingsCoord,
9300 (OUT) FGEastingsCoord,
9400 (OUT) FGZoneLetter,
9500 (OUT) FGZoneNumber,
9600 (IN ) UPSZoneLetter,
9700 (IN ) UPSEastingsLetter,
9800 (IN ) UPSNorthingsLetter,
9900 (IN ) UPSEastingsNumber,
10000 (IN ) UPSNorthingsNumber)
10100 ELSE ConvertToUTM ((OUT) FGNorthingsCoord,
10200 (OUT) FGEastingsCoord,
10300 (OUT) FGZoneLetter,
10400 (OUT) FGZoneNumber,
10500 (OUT) SpheroidNumber,
10600 (IN ) UTMZoneLetter,
10700 (IN ) UTMEastingsLetter,
10800 (IN ) UTMNorthingsLetter,
10900 (IN ) UTMNorthingsNumber)
11000 END; (of PROCEDURE PolarToUPS)
11100 (of MODULE FG2UPS
PROCEDURE ConvertToUPS

PROCEDURE ConvertToUPS performs the Polar Grid to Universal Polar Steriographic conversion algorithm described in Paragraph 3.2.118 of CG108100A dated 23 October 1978.

PROCEDURE ConvertToUPS is CALLED by: PROCEDURE PolarToUPS

PROCEDURE PolarToUTM

Programmed by J.W. Gillis 5-6-82

This procedure ASSUMES that certain data are available as required by the algorithm but not adequately described in the algorithm description.

This procedure DOES NOT perform data validity checks that are not specified in the algorithm description. This is to allow the algorithm features to be presented more clearly.

PROCEDURE ConvertToUPS accepts polar grid northing and easting coordinates and converts them to UPS coordinates.

FUNCTION aMODb (IN a,b:REAL):INTEGER;

VAR Ainteger INTEGER;
Binteger INTEGER;

BEGIN
Ainteger:=TRUNC(a);
Binteger:=TRUNC(b);
aMODb:=Ainteger MOD Binteger
END; (of FUNCTION aMODb)
BEGIN
  1. Initialize the GridLetter array.
  2. FOR Index := 1 TO 26 DO
     GridLetter[Index] := CHR$(ORD('A') + Index - 1);
  3. Calculate the UPSEastingsNumber & UPSEastingsLetter Index.
  4. IF PGZoneLetter > 'M'
      THEN BEGIN
        UPSEastingsNumber := aMODb(FGEastingsCoord - 200000),
        100000/10;
        Index := aMODb((FGEastingsCoord - 200000)/100000 + 20)
      END; (of IF)
  5. IF PGZoneLetter < 'N'
      THEN BEGIN
        UPSEastingsNumber := aMODb(FGEastingsCoord, 100000)/10;
        Index := aMODb(FGEastingsCoord/100000 + 18)
      END; (of IF)
  6. WRITELN;
  7. WRITELN ('UPSEASTINGNUMBER IS ', UPSEASTINGNUMBER);
  8. WRITELN ('AND ITS LETTER INDEX IS ', INDEX);
  9. Calculate the UPSEastingsLetter from its index.
  10. IF Index > 16
      THEN Index := Index + 2;
  11. IF (Index > 2) AND (Index < 17)
      THEN Index := Index + 1;
  12. UPSEastingsLetter := GridLetter[Index];
  13. WRITELN;
  14. WRITELN ('UPSEASTINGLETTER IS ', UPSEASTINGLETTER);
  15. WRITELN ('AND ITS CORRECTED INDEX IS ', INDEX);
  16. Calculate the UFSNorthingNumber & NorthingLetter Index.
  17. IF PGZoneLetter > 'M'
      THEN BEGIN
        UFSNorthingNumber := aMODb(FGNorthingCoord - 1300000),
        100000/10;
        Index := aMODb((FGNorthingCoord - 1300000)/100000),
        24)
      END; (of IF)
  18. IF PGZoneLetter < 'N'
      THEN BEGIN
        UFSNorthingNumber := aMODb(FGNorthingCoord - 3000000),
        100000/10;
        Index := aMODb((FGNorthingCoord - 3000000)/100000),
        24)
      END; (of IF)
  19. WRITELN;
WRITELN('UFSNORTHINGNUMBER IS 'UPSNORTHINGNUMBER);
WRITELN('AND ITS LETTER INDEX IS 'INDEX);
(Calculate the UFSNORTHINGLetter from its Index)

IF Index > 16
    THEN Index:=Index+2;
IF Index < 17
    THEN INDEX:=INDEX+1;
UPSNORTHINGLetter:=GridLetter[Index];

WRITELN('UPSNORTHINGLETTER IS 'UPSNORTHINGLETTER);
WRITELN('AND ITS CORRECTED INDEX IS 'INDEX);

END; (of PROCEDURE ConvertToUPS)

END. (of MODULE CNV2UPS)
TEMPORARY

STUB PROCEDURE

100 <
200 MODULE CNV2UTM (INPUT, OUTPUT);
300 <
400 <
500 TYPE
600 Meters = REAL;
700 Decameters = REAL;
800 Letter = SET OF CHAR;
900 Spheres = INTEGER;
1000 <
1100 <
1200 PROCEDURE ConvertToUTM
1300 ({IN} PGNorthinCoord : Meters;
1400 {IN} FGEastingsCoord : Meters;
1500 {IN} FGZoneLetter : Letter;
1600 {IN} FGZoneNumber : INTEGER;
1700 {IN} SpheroidNumber : Spheres;
1800 {OUT} VAR UTMZoieLetter : Letter;
1900 {OUT} VAR UTMEasttingsLetter : Letter;
2000 {OUT} VAR UTMNorthinLetter : Letter;
2100 {OUT} VAR UTMEasttingsNumber : Decameters;
2200 {OUT} VAR UTMNorthinNumber : Decameters);
2300 <
2400 <
2500 { PROCEDURE ConvertToUTM performs the Polar Grid to Universal
2600 { Transverse Mercator conversion algorithm described in
2700 { Paragraph 3.2.86 of CG108100A dated 23 October 1978.
2800 <
2900 { PROCEDURE ConvertToUTM is CALLED by: PROCEDURE PolarToUTM
3000 { PROCEDURE PolarToUPS
3100 { Programmed by J.W. Gillis 5-6-82
3200 { This procedure ASSUMES that certain data are available as
3300 { required by the algorithm but not adequately described in the
3400 { algorithm description.
3500 { This procedure DOES NOT perform data validity checks that are
3600 { not specified in the algorithm description. This is to allow
3700 { the algorithm features to be presented more clearly.
3800 { PROCEDURE ConvertToUTM accepts polar grid northing and eastings
3900 { coordinates and converts them to UTM coordinates.
4000 { TYPE
4100 TYPE Meters = REAL;
4200 Decameters = REAL;
4300 Letter = SET OF CHAR;
4400 Spheres = INTEGER;
4500 <
4600 <
4700 BEGIN
4800 IF <
4900 THEN
5000 WRITELN ('CNV2UTM HOOKED UP OK')
5100 <
5200 END; (of PROCEDURE ConvertToUTM)
5300 END; (of MODULE CNV2UTM)
PROGRAM ConvertToUTM (INPUT, OUTPUT); TYPE
  ASCIIArray = ARRAY [1..4] OF CHAR;
  Letter = CHAR;
VAR
  UTMCoordinate : RECORD
    UTMZoneNumber : ARRAY[1..2] OF INTEGER;
    UTMZoneLetter : Letter;
    UTMEastingLetter : Letter;
    UTMNorthingLetter : Letter;
    UTMEastingNumber : ARRAY[1..4] OF CHAR;
    UTMNorthingNumber : ARRAY[1..4] OF CHAR;
END; (UTMC)
PROCEDURE ConvertToUTM is called by: PROCEDURE PolarToUTM
PROCEDURE PolarToUTM
PROCEDURE ConvertToUTM accepts polar grid northing and easting coordinates and converts them to UTM coordinates.
PROCEDURE ConvertToUTM performs the Polar Grid to Universal
  Transverse Mercator conversion algorithm described in
  Paragraph 3.2.8a of US1510A dated 23 October 1978.

This procedure ASSUMES that certain data are available as
required by the algorithm but not adequately described in the
algorithm description.

This procedure DOES NOT perform data validity checks that are
not specified in the algorithm description. This is to allow
the algorithm features to be presented more clearly.

PROCEDURE ConvertToUTM accepts polar grid northing and easting
coordinates and converts them to UTM coordinates.

PROCEDURE ConvertToASCII: Number: INTEGER (in);
VAR AlphaNum:ASCIIArray;
CONVERT an integer number to its ASCII representation in base 10
This procedure not detailed in source

VAR i: INTEGER;
END;
BEGIN i := 1 DOWNT0 1 DO
FOR i := 1 DOWN TO 1 DO
BEGIN
  Adj := (Number MOD 10) + ORD('0') +
  AlphaNum[i] := CHR(Adj);
  Number := Number DIV 10
END
5S4;")0
END;
(COnvert.ToA;-C7iI>
0 05
LEGIN
(COnvefrToUTM)>
WR'CTELN
('FG Nos~rinl Cuod(inte-eer)
U
READLN (FGeastdit.Coonv6)
Z<
G
Zone Letter (Letter)
7-00
READLN (FG7oreLtAtLer )
G
o ne N ia e
r
790
?EDL C
OZog
Nurziber
WRI
4TELN
('psh
p ro iid N
uiiibe
r
i
t e
r))
~
C endof' incu 13-
3500 WITH UTilCoordin-ate
DO
600~
BEGIN
30 0{
Find EastinA Letter
(A2)
I-
I
=: FGZurieNuiibe r MOD
3
0C
EastinsA Letter Offs-et Factor
900
CASE I OF
9100
0: ELetterOffset := 24
9200
1: ELetterOffset := 6
9300
2: ELetterOffset := 15
9400
END;
9500
Tel := (FGeastins Coo rd
5300000
)DIV 10-3000 +
EMEttie-0ffF1'
E'v
Fbi 1owirA
letter con-Ver-sion'
not
ELCO niinY
tu-uITJm
t3t
i3n
3n
PO
{ F'
C
rucedure in document;ation fails at
lt-smt
-A,~'
HersIt:id.
FRGu?
10000 -L which i-5 in
32U
, arid th-is one succeeds at lea- I ti .
1C)2
C.0
IF Tel
'-13
THEN Tel
z.
Tel%-
+2
10300
ELSE IF Tel
=: 13
THEN Tel
:= Tel
-1
10400
ELSE Tel := Tel
- 2
;
10500
Tel := Tel + ORD('A') - 1
10600
UTMEastingsLetter := Chr(Tel ) (A2);4
10700
<
10800
< Find Northing Number >
10900
IF NOT EQD( FGZoneNumber) THEN
11000
FGNorthingCoord := FGNorthingCoord + 500000
11100
Tp := FGNorthingCoord MOD 2000000
11200
Tn1 := Tp DIV 100000 + 1
11300
< Make Spheroid Adjustment >
11400
CASE SpheroidNumber OF
11500
< Clark 1866 >
11600
1: IF (FGZoneNumber > 31) OR (FGZoneNumber > 58) THEN Tn1 := Tn1 + 10
11700
ELSE IF (FGZoneNumber <= 31) AND (FGZoneNumber > 3) THEN Tn1 := Tn1 - 10;
11800
< International >
11900
2: IF (FGZoneNUMBER >43) AND (FGZoneNumber <52)
12000
THEN Tn1 := Tn1 + 10;
12100
< Clark 1880 >
12200
3: Tn1 := Tn1 + 10;
12300
<
BEGIN {Everest} 
4: IF FGZoneNumber < 46 THEN Tnl := Tnl + 10; 
BEGIN {Bessel} 
5: IF (FGZoneNumber mod 52) AND (FGZoneLetter = 'R') THEN Tnl := Tnl + 10; 
BEGIN {Australian} 
6: IF ODD(FGZoneNumber) THEN Tnl := Tnl + 5 
ELSE Tnl := Tnl + 5 
END {CASE OF SpheroidNumber} 
BEGIN {Calculate Final Northings Letter Index} 
IF Tnl > 14 THEN Tnl := Tnl + 2 
ELSE IF Tnl > 9 THEN Tnl := Tnl + 1; 
Tnl := Tnl + ORD('A') - 1; 
UTMNorthingsLetter := CHR(Tnl); 
BEGIN {Format for Output} 
ConvertToASCII(FGNorthingsCoord mod 100000, UTMNorthingsNumber); 
ConvertToASCII(FGEastingsCoord mod 100000, UTEastingsNumber); 
UTMZoneNumber[I] := FGZoneNumber DIV 10; 
UTMZoneNumber[I] := FGZoneNumber MOD 10; 
UTMZoneLetter := FGZoneLetter; 
BEGIN {WRITE OUTPUT} 
WRITELN; 
WRITELN; 
WRITELN; 
FOR I := 1 TO 2 DO WRITELN ('UTM Zone Number ',I,UTMZoneNumber[I]); 
WRITELN ('UTM Zone Letter  ',UTMZoneLetter); 
WRITELN ('UTM Eastings Letter ',UTMEastingsLetter); 
WRITELN ('UTM Northings Letter ',UTMNorthingsLetter); 
FOR I := 1 TO 4 DO BEGIN 
WRITELN ('UTM Eastings Number ',I,UTMEastingsNumber[I]); 
WRITELN ('UTM Northings Number ',I,UTMNorthingsNumber[I]); 
END; 
BEGIN {End of Output} 
BEGIN {of WITH UTM} 
END; 
BEGIN {ConvertToUTM} 
END.
PROCEDURE ConvertToUTM IS

--
TYPE ShortArray IS ARRAY(1..4) OF CHARACTER;

PGNorthingCoord :INTEGER; --IN
PGEastingCoord :INTEGER; --IN
PGZoneLetter :CHARACTER; --IN
PGZoneNumber :INTEGER; --IN
SpheroidNumber :INTEGER; --IN

--
UTMZoneNumber :ARRAY(1..2) OF INTEGER;
UTMZoneLetter :CHARACTER;
UTMEastingLetter :CHARACTER;
UTMNorthingLetter :CHARACTER;
UTMEastingNumber :ARRAY(1..4) OF CHARACTER;
UTMNorthingNumber :ARRAY(1..4) OF CHARACTER;

--
I, EletterOffset, Tel, Tp, tnl, ASCIIPos: INTEGER; --Local

PROCEDURE ConvertToASCII( Number: in INTEGER;
AlphaNum: out ShortArray) IS

-- Convert an integer number to its ASCII representation in base 10
-- This procedure not detailed in source
BEGIN
FOR I in reverse 1 .. 4 LOOP
  ASCIIPos := Number MOD 10 + CHARACTER.POS('0');
  AlphaNum(i) := CHARACTER'VAL( ASCIIPos);
  Number := Number / 10;
END LOOP;
END ConvertToASCII;

FUNCTION ODD( I: INTEGER ) return BOOLEAN is
BEGIN
  IF I = 2 * ( I/2 ) THEN return false;
  Else return true;
  END IF;
END ODD;

-- Find Easting Letter (A2)
I := PGZoneNumber MOD 3;
-- Easting Letter Offset Factor
CASE I IS
  WHEN 0 =>ELetterOffset := 24;
  WHEN 1 =>ELetterOffset := 6;
  WHEN 2 =>ELetterOffset := 15;
  WHEN OTHERS => NULL;
END CASE;

--
Tel := ( PGEastingCoord - 500000 ) / 100000 + ELetterOffset;
IF Tel > 14 THEN Tel := Tel + 2;
ELSIF Tel > 9 THEN Tel := Tel + 1;
END IF;
Tel := Tel + CHARACTER.POS('a') - 1;
UTMEastingLetter := CHARACTER'VAL( Tel ); --A2

-- Find Northing Number
IF NOT ODD( PGZoneNumber ) THEN
  PGNorthingCoord := PGNorthingCoord + 500000;
END IF;
Tp := PGNorthingCoord MOD 2000000;
Tn1 := Tp / 100000 + 1;
-- Make Spheroid Number Adjustment
CASE SpheroidNumber IS
  -- Clark 1866
  WHEN 1=>IF PGGridzoneNum < 31 OR PGZoneNumber > 58 THEN Tn1 := Tn1 + 10;
  ELSIF PGZoneNumber > 51 AND PGZoneNumber < 59 THEN Tn1 := Tn1 - 10;
  END IF;
  -- International
  WHEN 2=>IF PGZoneNumber > 46 AND PGZoneNumber < 52 THEN Tn1 := Tn1 + 10;
  END IF;
  -- Clark 1880
  WHEN 3=>Tn1 := Tn1 + 10;
  -- Everest
  WHEN 4=>IF PGZoneNumber < 46 THEN Tn1 := Tn1 + 10; END IF;
  -- Bessel
  WHEN 5=>IF PGZoneNumber < 52 AND PGZoneLetter < 'R' THEN Tn1 := Tn1 + 10;
  END IF;
  -- Australian
  WHEN 6=>IF ODD( PGZoneNumber ) THEN Tn1 := Tn1 + 15;
  END IF;
  WHEN OTHERS => NULL; -- Not in source
END CASE;

-- Calculate Final Northing Letter Index
If Tn1 > 14 Then Tn1 := Tn1 + 2;
ELSIF Tn1 > 9 THEN Tn1 := Tn1 + 1;
END IF;

Tn1 := Tn1 + CHARACTER'POS('A') - 1;
UTMNorthingLetter := CHARACTER'VAL( Tn1 );
-- format for output
ConvertToASCII( PGNorthingCoord MOD 100000, UTMNorthingNumber );
ConvertToASCII( PGEastingCoord MOD 180088, UTMNorthingNumber );
UTMZoneNum(1) := PGZoneNumber / 10;
UTMZoneNum(2) := PGZoneNumber MOD 10;
UTMZoneLetter := PGZoneLetter;
END ConvertToUTM;
PROGRAM DRVGZTB (INPUT,OUTPUT);

< PROGRAM DRVGZTB provides a test driver capability for testing. >
< GridZoneGeneration procedures for TRAILBLAZER. >

TYPE

DegreesReal = Real;

ZoneRange = 1..60;

Letters = 'A'..'Z';

VAR

Longitude : DegreesReal;

Latitude : DegreesReal;

GridZoneNumber : ZoneRange;

GridZoneLetter : Letters;

PROCEDURE TBGridZoneGeneration

< (OUT) Longitude : DegreesReal;

< (OUT) Latitude : DegreesReal;

< (IN ) VAR GridZoneNumber : ZoneRange;

< (IN ) VAR GridZoneLetter : Letters ) EXTERN;

< PROCEDURE TBGridZoneGeneration models the TRAILBLAZER conversion of geographic coordinates to Universal Transverse Mercator. >
< (UTM) coordinates grid zone designator number, letter. >

BEGIN

WRITELN ('ENTER Longitude');

READLN (Longitude);

WRITELN ('ENTER Latitude');

READLN (Latitude);

TBGridZoneGeneration ( (OUT) Longitude, Latitude,

< (IN ) gridzonenumber, gridzonelatter);
```pascal
100 MODULE TBGZDG ( INPUT,OUTPUT );
200 {
300 TYPE
400 DegreesReal = REAL;
500 ZoneRange = 1..60;
600 Letters = 'A'..'Z';
700 {
800 PROCEDURE TBGridZoneGeneration
900 (IN ) Longitude : DegreesReal;
1000 (IN ) Latitude : DegreesReal;
1100 (OUT) VAR GridZoneIndex : ZonRange;
1200 (OUT) VAR GridZoneLetter : Letters;
1300 {
1400 PROCEDURE TRGridZoneGeneration procues the TRAILBLAZER conversion
1500 of geographic coordinates to Universal Transverse Mercator
1600 (UTM) coordinates - grid zone designator number, letter.
1700 Documentation used was the GP20K subroutine did. 20 Feb 81
1800 from the listings provided for the TRAILBLAZER system.
1900 {
2000 PROCEDURE TBGridZoneGeneration is referenced by:
2100 PROGRAM DRUGZTR;
2200 {
2300 PROCEDURE TBGridZoneGeneration makes no references.
2400 {
2500 This procedure DOES NOT perform any data validity checks
2600 that are not explicitly specified in the algorithm.
2700 description. This is to allow the algorithm features to be
2800 represented more clearly.
2900 {
3200 TYPE
3300 Letters = 'A'..'Z';
3400 IndexRange = 1..24;
3500 {
3600 VAR
3700 GridZoneIndex : IndexRange;
3800 GridZoneLetterList : ARRAY[1..24] OF LETTERS;
3900 {
4000 BEGIN
4100 Initialize allowable eharacters array
4200 {
5000 GridZoneLetterList[8] := 'H';
5400 GridZoneLetterList[12] := 'M';
5800 GridZoneLetterList[16] := 'R';
5900 GridZoneLetterList[17] := 'S';
6000 GridZoneLetterList[18] := 'T';
6100 GridZoneLetterList[19] := 'U';
6300 GridZoneLetterList[21] := 'W';
6400 GridZoneLetterList[22] := 'X';
6500 GridZoneLetterList[23] := 'Y';
```
GridZoneLetterList [22] := 'X';
GridZoneLetterList [23] := 'Y';

GridZoneNumber := TRUNC (31.0+(Longitude/6.0));
GridZoneIndex := TRUNC (13.0+(Latitude/8.0));
GridZoneLetter := GridZoneLetterList[GridZoneIndex];

END;  

END;  

END.  

END.  

END; 

PROCEDURE TGridZoneGeneration 

MODULE SGRZTB
100 <
200 PROGRAM DRUGZHG ( INPUT, OUTPUT );
300 <
400 < PROGRAM DRUGZHG provides a test driver capability for testing >
500 < GridZoneGeneration procedures for MAGIC. >
600 <
700 TYPE
800 Radians: REAL;
900 ZoneRange = 1..60;
1000 Letters = 'A'..'Z';
1100 VAR
1200 Longitude: Radians;
1300 Latitude: Radians;
1400 GridZoneNumber: ZoneRange;
1500 GridZoneLetter: Letters;
1600 <
1700 PROCEDURE MagicGridZoneGeneration
1800 ((OUT) Longitude : Radians;
1900 (OUT) Latitude : Radians;
2000 (IN) VAR GridZoneNumber : ZoneRange;
2100 (IN) VAR GridZoneLetter : Letters; EXTERN;
2200 <
2300 < PROCEDURE MagicGridZoneGeneration models the MAGIC conversion >
2400 < of geographic coordinates to Universal Transverse Mercator >
2500 < (UTM) coordinates grid zone designation number & letter. >
2600 <
2700 BEGIN
2800 <
2900 WRITELN ( 'ENTER Longitude' );
3000 READLN ( Longitude );
3100 WRITELN ( 'ENTER Latitude' );
3200 READLN ( Latitude );
3300 MagicGridZoneGeneration ( <OUT> Longitude, Latitude,
3400 (IN) GridZoneNumber, GridZoneLetter );
3500 WRITELN;
3600 WRITELN ( 'GridZoneNumber is ' , GridZoneNumber );
3700 WRITELN;
3800 WRITELN ( 'GridZoneLetter is ' , GridZoneLetter );
3900 <
4000 END. < of PROGRAM DRUGZHG >
6100 <  
6200 MODULE MGGZDG ( INPUT, OUTPUT ) ;  
6300 <  
6400 TYPE  
6500 Radians = REAL ;  
6600 ZoneRange = 1..60 ;  
6700 Letters = 'A'..'Z' ;  
6800 <  
6900 PROCEDURE MGGGridZoneGeneration  
7000 ( IN ) Longitude : Radians ;  
7100 ( IN ) Latitude : Radians ;  
7200 ( OUT ) VAR GridZoneNumber : ZoneRange ;  
7300 ( OUT ) VAR GridZoneLetter : Letters ;  
7400 <  
7500 PROCEDURE MGGGridZoneGeneration models the MAGIIC conversion of geographic coordinates to Universal Transverse Mercator (UTM) coordinates - grid zone designator number & letter.  
7600 Documentation used was source code listings from the MAGIIC document CAI08100A dtd. 23 Oct 1978, Par. 3.2.90, pg. 167.  
7700 PROCEDURE MGGGridZoneGeneration is referenced by:  
7800 PROGRAM DRVGSZMG  
7900 <  
8000 PROCEDURE MGGGridZoneGeneration makes no references.  
8100 <  
8200 This procedure DOES NOT perform any data validity checks that are not explicitly specified in the algorithm description. This is to allow the algorithm features to be represented more clearly.  
8300 <  
8400 CONST  
8500 Pi = 3.1415926 ;  
8600 <  
8700 TYPE  
8800 IndexRange = 1..26 ;  
8900 <  
9000 VAR  
9100 GridZoneIndex : IndexRange ;  
9200 <  
9300 BEGIN  
9400 <  
9500 Calculate the grid zone number  
9600 <  
9700 GridZoneNumber := TRUNC(((180.0/Pi)*Longitude+180.0)/6.0)+1 ;  
9800 <  
9900 NO compensation for wrap around of grid zone numbers  
1000 <  
10100 <  
10200 Determine the grid zone letter  
10300 <  
10400 <  
10500 Since no details are provided in the referenced documentation  
10600 it is ASSUMED that we know how to assign A or B for latitudes  
10700 equal to or over 84 degrees North and Y or Z for latitudes  
10800 equal to or over 80 degrees South.  
10900 <  
11000 TRUNCATION to integer is ASSUMED since it is necessary at this point in order to use the GridZoneIndex as a pointer.  
11100 <  
11200 STATED LATITUDE RANGE IS 80<=LATITUDE<=84 IN DEGREES  
11300 ;
GridZoneIndex := TRUNC (((180.0/\pi)*Latitude+30.0)/8.0);

5400 \{ Compute midrange grid zone letters \}
5500 \}
5600 \{ GridZoneIndex \leq 5 \}
5700 \{ Here we handle the 'i' which is not used \}
5800 \{ GridZoneIndex \geq 6 \) AND \}
5900 \{ GridZoneIndex \leq 10 \)
6000 \{ The rest of the GridZoneIndex are biased off by ORD('C') \}
6100 \{ GridZoneIndex \geq 11 \) AND \}
6200 \{ GridZoneIndex \leq 19 \)
6300 \{ GridZoneIndex \geq 19 \)
6400 \{ Assign Y or Z to the North Polar Zone according as Western \}
6500 \{ or Eastern Hemisphere respectively. \}
6600 \{ Assign A or B to the South Polar Zone according as Western \}
6700 \{ or Eastern Hemisphere respectively. \}
6800 \{ NO correction for the four irregular zones - \}
6900 \{ 32X, 34X, and 36X do not exist \}
7000 \{ 31V is truncated \}
7100 \{ of PROCEDURE MGGGridZoneGeneration \}
7200 \{ of MODULE MGGZDG \}
7300 \}
PROGRAM DRVGZBT (INPUT,OUTPUT);

PROGRAM DRVGZBT provides a test driver capability for testing.

GridZoneGeneration procedures for BETA.

TYPE
  Radians = REAL;
  ZoneRange = 1..60;
  Letters = 'A'..'Z';

VAR
  Longitude : Radians;
  Latitude : Radians;
  GridZoneNumber : ZoneRange;
  GridZoneLetter : Letters;
  CenterMeridian : Radians;

PROCEDURE BTGridZoneGeneration
  (OUT) Longitude : Radians;
  (OUT) Latitude : Radians;
  (IN)VAR GridZoneNumber : ZoneRange;
  (IN)VAR GridZoneLetter : Letters;
  (IN)VAR CenterMeridian : Radians); EXTERN;

PROCEDURE BTGridZoneGeneration models the BETA conversion
of geographic coordinates to Universal Transverse Mercator
(UTM) coordinates grid zone designator number, letter and the
central meridian of the rectangle.

BEGIN
  WRITELN (ENTER Longitude);
  READLN (Longitude);
  WRITELN (ENTER Latitude);
  READLN (Latitude);
  BTGridZoneGeneration (OUT) Longitude, Latitude,
  (IN) GridZoneNumber, GridZoneLetter,
  CenterMeridian);
  WRITELN;
  WRITELN (GridZoneNumber is GridZoneNumber);
  WRITELN;
  WRITELN (GridZoneLetter is GridZoneLetter);
  WRITELN;
  WRITELN (CenterMeridian is CenterMeridian);

END. (of PROGRAM DRVGZBT)
MODULE BTGZDG (INPUT,OUTPUT);

TYPE
  Radians = REAL;
  ZoneRange = 1..60;
  Letters = 'A'..'Z';

PROCEDURE BTGridZoneGeneration
  <(IN) Latitude : Radians;>
  <(IN) Longitude : Radians;>
  <(OUT) VAR GridZoneNumber : ZoneRange;>
  <(OUT) VAR GridZoneLetter : Letters;>
  <(OUT) VAR CenterMeridian : Radians;>

PROCEDURE BTGridZoneGeneration makes no references.

PROCEDURE BTGridZoneGeneration makes no references.

This procedure DOES NOT perform any data validity checks
that are not explicitly specified in the algorithm
description. This is to allow the algorithm features to be
represented more clearly.

Since the included 'ZDBPRO.COM' is not available to us at
this time we assume implicit typing in the source FORTRAN
code.

CONST
  Pi = 3.1415926;

TYPE
  Letters = 'A'..'Z';
  IndexRange = 1..24;

VAR
  GridZoneLtrList : ARRAY[1..24] OF LETTERS;
  GridZoneIndex : IndexRange;

BEGIN
  Initialize allowable characters array
  GridZoneLtrList[1] := 'A';
  GridZoneLtrList[3] := 'C';
  GridZoneLtrList[7] := 'G';
  GridZoneLtrList[8] := 'H';
  GridZoneLtrList[9] := 'I';
  GridZoneLtrList[10] := 'J';
6400 GridZoneLtrList[12] := 'N'
6500 GridZoneLtrList[13] := 'N'
6600 GridZoneLtrList[14] := 'P'
6700 GridZoneLtrList[15] := 'Q'
6800 GridZoneLtrList[16] := 'R'
6900 GridZoneLtrList[17] := 'S'
7000 GridZoneLtrList[18] := 'T'
7100 GridZoneLtrList[19] := 'U'
7200 GridZoneLtrList[20] := 'V'
7300 GridZoneLtrList[21] := 'W'
7400 GridZoneLtrList[22] := 'X'
7500 GridZoneLtrList[23] := 'Y'
7600 GridZoneLtrList[24] := 'Z'
7700 < Calculate the grid zone number
7800 <
7900 GridZoneNumber := TRUNC(((18000.0/PI)*Latitude) +18600.0) DIV 6001
8000 <
8100 < Compensate for error around of grid zone numbers
8200 <
8300 IF GridZoneNumber > 60 THEN GridZoneNumber := GridZoneNumber - 60
8400 IF GridZoneNumber < 1 THEN GridZoneNumber := GridZoneNumber + 60
8500 <
8600 < Determine the grid zone letter
8700 <
8800 GridZoneIndex := TRUNC((Latitude*(180.0/PI)+104.0)/8.0))
8900 <
9000 Test for and lock out North Polar Zones
9100 IF GridZoneIndex > 22 THEN GridZoneIndex := 22
9200 <
9300 < NOTE that no such test is needed for the South Polar Zone
9400 < because the altitude limit was given as <= 80 South
9500 <
9600 GridZoneLetter := GridZoneLtrList[GridZoneIndex]
9700 <
9800 < Correct for the four irregular zones -
9900 < 32X, 34X, and 36X do not exist
1000 31V is truncated
1010 <
1020 < Truncate grid zone 31V
1030 <
1040 IF (GridZoneIndex = 20) AND ((GridZoneNumber -31) AND (Longitude >= 3.0*(PI/180.0))) THEN GridZoneNumber := 32;
1050 <
1060 Correct for grid zones 32X, 34X, and 36X
1070 <
1080 IF (GridZoneIndex = 22) AND (GridZoneNumber = 32) THEN IF Longitude >= 9.0*(PI/180.0) THEN GridZoneNumber := 33 ELSE GridZoneNumber := 31;
1090 <
1100 Correct for grid zones 32X, 34X, and 36X
1110 <
1120 IF (GridZoneIndex = 22) AND (GridZoneNumber = 34) THEN GridZoneNumber := 30.27
12700 THEN IF Longitude >= 21.0*(Pi/180.0)
12800 THEN GridZoneNumber := 35
12900 ELSE GridZoneNumber := 33
13000 IF (GridZoneIndex - 22) AND
13100 (GridZoneNumber = 36)
13200 THEN IF Longitude >= 33.0*(Pi/180.0)
13300 THEN GridZoneNumber := 37
13400 ELSE GridZoneNumber := 35
13500 CenterMeridian := (4*GridZoneNumber-183)*(Pi/180.0)
13600 END \ of PROCEDURE BTGridZoneGeneration
13700 END \ of MODULE BTGZDG
100 { 
200 PROGRAM DRVGZGR (INPUT,OUTPUT); 
300 { 
400 PROGRAM DRVGZGR provides a test driver capability for testing 
500 GridZoneGeneration procedures for GUARDRAIL. 
} 
800 TYPE 
900 DegreesReal = REAL; 
1000 DegreesInteger = INTEGER; 
1100 ZoneRange = 1..60; 
1200 Letters = 'A'..'Z'; 
1300 VAR 
1400 Longitude; 
1500 Latitude; 
1600 GridZoneNumber; 
1700 GridZoneLetter; 
1750 centerMeridian; 
1800 
1900 PROCEDURE GridZoneGeneration 
2000 (OUT) Longitude; 
2100 (OUT) Latitude; 
2200 (IN) VAR GridZoneNumber; 
2300 (IN) VAR GridZoneLetter; 
2400 (IN) VAR centerMeridian; 
2500 { 
2600 PROCEDURE GridZoneGeneration models the GUARDRAIL conversion 
2604 of geographic coordinates to Universal Transverse Mercator 
2606 (UTH) coordinates grid zone designator number, letter, and the 
2608 central meridian of the rectangle. 
2690 } 
2900 BEGIN 
3200 
3400 WRITELN ( 'ENTER Longitude'); 
3500 READLN ( Longitude ); 
3600 WRITELN ( 'ENTER Latitude' ); 
3700 READLN ( Latitude ); 
3800 GridZoneGeneration ( OUT) Longitude,Latitude, 
3900 (IN) GridZoneNumber,GridZoneLetter, 
3950 centerMeridian; 
4000 WRITELN; 
4100 WRITELN ( 'GridZoneNumber is ' GridZoneNumber ); 
4200 WRITELN; 
4300 WRITELN ( 'GridZoneLetter is ' GridZoneLetter ); 
4400 WRITELN; 
4500 WRITELN ( 'CenterMeridian is ' CenterMeridian ); 
4550 } 
4600 END. ( of PROGRAM DRVGZGR )
100  
200 MODULE GRGZDG ( INPUT,OUTPUT );
300  
400 TYPE
500  DegreesReal       = REAL;
600  DegreesInteger    = INTEGER;
700  ZoneRange         = 1..60;
800  Letters           = 'A'..'Z';
900  
1000 PROCEDURE ORGridZoneGeneration
1100      (IN )  Longitude  : DegreesReal;
1200      (IN )  Latitude   : DegreesReal;
1300      (OUT) VAR GridZoneNumber : ZoneRange;
1400      (OUT) VAR GridZoneLetter  : Letters;
1500      (OUT) VAR CenterMeridian : DegreesInteger;
1600  
1700  PROCEDURE ORGridZoneGeneration models the GUARDRAIL conversion
1800  of geographic coordinates to Universal Transverse Mercator
1900  (UTM) coordinates - grid zone designator numbers, letter and
2000  (the central meridian.
2100  
2200  PROCEDURE ORGridZoneGeneration is referenced by:
2300  PROGRAM DRVGZGR
2400  
2500  PROCEDURE ORGridZoneGeneration makes no references.
2600  
2700  This procedure DOES NOT perform any data validity checks
2800  that are not explicitly specified in the algorithm
2900  description. This is to allow the algorithm features to be
3000  represented more clearly.
3100  
3200  TYPE
3300  Letters           = 'A'..'Z';
3400  IndexRange        = 1..24;
3500  
3600  VAR
3700  GridZoneListList : ARRAY[1..24] OF LETTERS;
3800  GridZoneIndex     : IndexRange;
3900  
4000 BEGIN
4100  INITIALIZE ALLOWABLE CHARACTERS ARRAY
4200  
4300  GridZoneListList [1] := 'A';
5000  GridZoneListList [8] := 'H';
5100  GridZoneListList [9] := 'I';
5200  GridZoneListList [10] := 'K';
5400  GridZoneListList [12] := 'M';
5600  GridZoneListList [14] := 'P';
5700  GridZoneListList [15] := 'Q';
5800  GridZoneListList [16] := 'R';
5900  GridZoneListList [17] := 'S';
6000  GridZoneListList [18] := 'T';
6100  GridZoneListList [19] := 'U';
6200  GridZoneListList [20] := 'V';
6300  GridZoneListList [21] := 'W';
6400  
6500 }
GridZoneLetterList [22] := 'X';
GridZoneLetterList [23] := 'Y';

GridZoneNumber := TRUNC (31.0+(Longitude/6.0));
GridZoneIndex := TRUNC (13.0+(Latitude/6.0));
GridZoneLetter := GridZoneLetterList[GridZoneIndex];
CenterMeridian := 6*GridZoneNumber-183;

END; < of PROCEDURE GRIDZoneGeneration

END. < of MODULE GRIDZDG
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A
SUPPLEMENTARY

INFORMATION
This report describes the findings of JPL regarding geographic transformation algorithms used in MAGIIC, GUARDRAIL, TRAILBLAZER, and BETA systems. A set of parameters is developed to characterize and catalogue intelligence system algorithms in the four systems. Individual algorithms are also analyzed to determine if they are performing their functions properly.