

NASA-CR-178099
19880014547

NASA Contractor Report 178099

MANUAL FOR OBSCURATION CODE WITH
SPACE STATION APPLICATIONS

FOR REFERENCE

NOT TO BE TAKEN FROM THIS ROOM

R. J. Marhefka and L. Takacs

LIBRARY COPY

JUN 13 1988

LANGLEY RESEARCH CENTER
LIBRARY, NASA
HAMPTON, VIRGINIA

THE OHIO STATE UNIVERSITY
ElectroScience Laboratory
Columbus, Ohio

NOTICE

~~FOR EARLY DOMESTIC DISSEMINATION~~

~~Because of its significant early commercial potential, this information, which has been developed under a U.S. Government program, is being disseminated within the United States in advance of general publication. This information may be duplicated and used by the recipient with the express limitation that it not be published. Release of this information to other domestic parties by the recipient shall be made subject to these limitations.~~

~~Foreign release may be made only with prior NASA approval and appropriate export licenses. This legend shall be marked on any reproduction of this information in whole or in part.~~

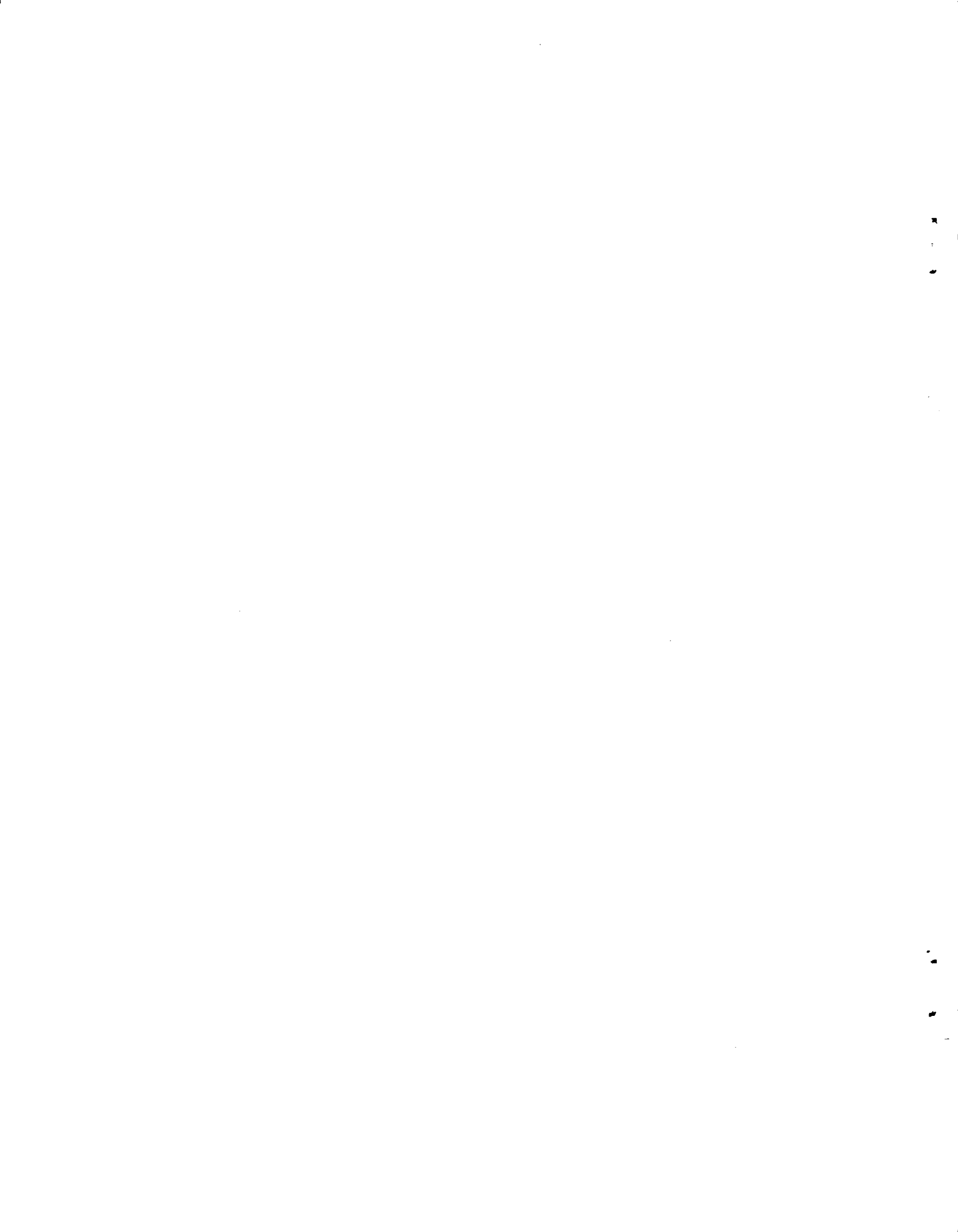
Date for general release May 31, 1988

Grant NSG-1498
May 1986



National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665



Contents

I User's Manual

1	Introduction	1
2	Method	4
3	Principle of Operation	13
3.1	Overview	13
3.2	Modeling the Structures	14
3.3	Running the Code	14
3.4	Non-Interactive Commands	15
3.5	Interactive Commands	18
3.6	Keypad Use	19
4	Non-Interactive Commands	22
4.1	Command CC: Cone Frustum Geometry	24
4.2	Command CG: Cylinder Geometry	27
4.3	Command CM: and CE: Comments	30
4.4	Command EN: End Program	30
4.5	Command GP: Ground Plane	31
4.6	Command NC: Next Set of Cylinders	32
4.7	Command NG: No Ground Plane	32
4.8	Command NP: Next Set of Plates	32
4.9	Command NS: Next Set of Sources	32
4.10	Command NX: Next Problem	32
4.11	Command PG: Plate Geometry	33
4.12	Command RT: Rotate-Translate Geometry	36
4.13	Command SG: Source Geometry	38
4.14	Command UF: Scale Factor	42
4.15	Command UN: Units of Geometry	42
4.16	Command US: Units of Source	42
4.17	Command VF: Far Zone Volumetric Pattern	44
4.18	Command XQ: Execute Code	47
5	Interactive Commands	48
5.1	Overview	48
5.2	Command Descriptions	48

5.2.1	EXIT	49
5.2.2	HELP	50
5.2.3	SHADOW	52
5.2.4	SPAWN	54
5.2.5	SET ANTENNA_LOCATION	56
5.2.6	SET COORDINATES	58
5.2.7	SET FILL_CHARACTER	59
5.2.8	SET INPUT_SET	62
5.2.9	SET KEYPAD_MODE	63
5.2.10	SET OUTPUT	64
5.2.11	SET PATTERN_CUT	66
5.2.12	SET SCALE_FACTOR	67
5.2.13	SET UNITS	68
5.2.14	SET WINDOW	69
5.2.15	SHOW ANTENNA_LOCATION	71
5.2.16	SHOW COORDINATES	72
5.2.17	SHOW FILL_CHARACTER	73
5.2.18	SHOW INPUT_SET	74
5.2.19	SHOW KEYPAD_MODE	75
5.2.20	SHOW OUTPUT	76
5.2.21	SHOW PATTERN_CUT	77
5.2.22	SHOW SCALE_FACTOR	78
5.2.23	SHOW UNITS	79
5.2.24	SHOW WINDOW	80
6	Interpretation of the Output	81
7	Examples	84
7.1	Example 1: A Plate	85
7.2	Example 2: A Different Plate	87
7.3	Example 3: The First Plate Revisited	89
7.4	Example 4: A Non-Interactive Version of Example 1	91
7.5	Example 5: An Elliptic Cylinder	92
7.6	Example 6: Two Elliptic Cylinders	97
7.7	Example 7: A Space Station Model	102
7.8	Example 8: Another Look at the Space Station	117
	References	119
II	Code Manual	121
8	Introduction	123
9	Code Organization	124

10 Listings of the Code	126
10.1 VAX/VMS Subroutines	126
10.1.1 MAIN PROGRAM	126
10.1.2 SUBROUTINE INIT	128
10.1.3 SUBROUTINE INTRAC	130
10.1.4 Interactive Service Routines	133
10.2 Non-VAX/VMS Subroutines	146
10.2.1 MAIN PROGRAM (non-interactive)	146
10.3 Subroutines common to both modes	148
10.3.1 SUBROUTINE ABSCIN	148
10.3.2 FUNCTION BABS	187
10.3.3 BLOCK DATA	188
10.3.4 FUNCTION BTAN2	189
10.3.5 SUBROUTINE CAPINT	190
10.3.6 SUBROUTINE CYLINT	192
10.3.7 SUBROUTINE CYLROT	194
10.3.8 SUBROUTINE DOCYLS	195
10.3.9 SUBROUTINE DOCYL	196
10.3.10 SUBROUTINE DOPLAS	200
10.3.11 SUBROUTINE DOPLA	201
10.3.12 SUBROUTINE GEOM	203
10.3.13 SUBROUTINE INITGF	205
10.3.14 SUBROUTINE PLAINT	206
10.3.15 SUBROUTINE SCAN	208
10.3.16 FUNCTION SCANC/SPANC	210
10.3.17 SUBROUTINE WRTOUT	211
10.3.18 Include file	213
10.4 Non-FORTRAN VAX/VMS source files	216
10.4.1 CDU Source file	216
10.4.2 Keypad initialization file	219
11 VAX Implementation	221
11.1 Assembling the Code	221
11.2 Running the Code	222
11.3 Modifying the code	223
12 Non-VAX Implementation	224
13 NCAR Plot Program	226

List of Tables

3.1	Table of non-interactive commands.	16
3.2	Table of interactive commands.	20
3.3	VT100 keypad for SHADOW interactive commands.	21

List of Figures

2.1	Geometry showing the projection of the plate edge onto the far zone sphere.	5
2.2	Geometry showing the projection of a cone frustum onto the far zone sphere.	7
2.3	Intersection of observation direction vector with plate.	8
2.4	The geometry for deciding whether a ray does or does not hit the plate. . .	10
2.5	Geometry illustrating the hit point on a cone frustum segment.	11
4.1	Definition of finite cylinder geometry composed of cone frustum segments with elliptic cross section.	26
4.2	Definition of finite elliptic cylinder geometry.	29
4.3	Definition of flat plate geometry.	35
4.4	Definition of rotate-translate coordinate system geometry.	37
4.5	Definition of source geometry for dipole antennas.	40
4.6	Definition of source geometry for aperture antennas.	41
4.7	Definition of volumetric pattern coordinate system.	46
7.1	Illustration of the Space Station	115
7.2	Three-axis view of the Space Station as modeled by the input set.	116
7.3	NCAR plot showing the shadow map of the space station model.	118



Part I

User's Manual



Chapter 1

Introduction

When siting antennas on large structures, it is desirable to be able to quickly determine the clear line of sight transmission or reception paths for the antennas. If the structure under consideration is a space station, there will be many antennas to consider in an environment composed of a very large and complex array of living and working module, solar panels, and support structures. The antennas will potentially need to communicate with systems anywhere around the near zone of the structure and the complete far zone sphere. In short, a challenging problem.

In order to aid the antenna design engineer in the prediction of the near and far zone antenna patterns, antenna to antenna coupling, and radiation hazard considerations, for high frequency antennas in a complex environment, a couple of user oriented computer codes have been developed: the NEC - Basic Scattering Code (NEC-BSC) [1,2] and the Aircraft Code (NEWAIR) [3]. Both codes are based on the Uniform Geometrical Theory of Diffraction (UTD) [4], which is a high frequency ray optical method with corrections at shadow boundaries. The UTD is ideal for construction of efficient computer codes, such as these, for modeling the scattering from large structures. The NEC-BSC and NEWAIR are complementary codes, that is, the NEC-BSC is used when the antennas are not mounted on a curved surface, and the NEWAIR is used when the antennas are mounted on a curved surface. Both codes use plates to model flat structures, the NEC-BSC presently uses finite elliptic cylinders to model curved surfaces, and the NEWAIR presently uses ellipsoids.

Although the two UTD codes are presently very useful for predicting the performance of antennas in a complex environment, such as a space station, there are a few important considerations that should be taken into account. First, the present versions of these codes were not specifically developed for a space station application. The NEC-BSC was developed for ships and the NEWAIR for aircraft. Second and most importantly, even though they run fast for large size structures in terms of a wavelength, as compared with computer codes using other theories, such as method of moments; a problem with as many structural pieces as a space station can take a very long time to calculate a volumetric pattern. This means that the problem of antenna siting in a large structural environment should be viewed as a multiple stepped procedure to optimize results for minimum time and cost.

The design procedure for antenna siting can be viewed as a three step process, as far as the computer codes are concerned. First, it can generally be assumed that a good antenna location will provide a clear line of sight path between transmitter and receiver over the desired range of operation. This can best be accomplished using an obscuration code, which

is the goal of this computer code and document. This code will provide a volumetric shadow map of the projected shadow of a structure onto the far zone sphere centered at the antenna location. It is very fast running on space station applications and can be run interactively providing nearly immediate answers depending on the overall useage of the computer.

Second, a worst case code could be developed that will predict not only the clear line of sight regions, but will also map out the maximum values of the various field terms, such as the reflected and diffracted lobes. These scattered fields can cause undesired lobes to show up in the region of interest. This type of code will not only provide an answer to the question of where the optimum location for an antenna system should be, but also how it should be oriented at that position and what the gain and side lobe levels would be optimum. It can be designed to run at a little additional time cost over the obscuration code.

The final step would be to run a field prediction code such as the NEC-BSC and NEWAIR codes or their future versions optimized for the space station. This would be the confirmation phase of the design procedure to make sure that no surprises occur in the volumetric patterns. At this stage, it does not matter that the codes take a little longer to run, especially for the wealth of information that they produce. Of course, these results can be used to compare with measured results on scale models to validate the measurements and vice versa.

This document is concerned with the obscuration code, referred to here as "SHADOW". It has been specifically design with space station applications in mind. It directly solves for a shadow map by projecting the border of multiple sided flat plates and composite cone frusturns of elliptic cross section onto the far zone sphere. It then fills between the borders based on a pixel resolution and window size specified by the user. The definition of the geometry is based on a subset of the command word input system used for the UTD codes. This means that as the engineer proceeds through a design scenario progressing through the different levels of codes, there will be a minimum amount of conversion of input information.

The obscuration code has proven to be so efficient, that it was felt that it could be of great benefit to the design engineer to be able to run it in an interactive mode. Unfortunately, interactive procedures are not generally transportable between different computer systems. Because of the wide availability of DEC VAX computers in the engineering environment, and because of the ease of developing an interactive system on a VAX, the interactive features have been developed using devise dependent software for the VAX. The non-interactive and interactive parts of the code have been kept separate, however, so that the code can be run non-interactively without much change.

This document is divided into two parts. Part I is a user manual, that treats the code more or less as a black box device. It is about all that will be need for the average user to get started and obtain results. Chapter 2 describes the method that is used to obtain the shadow. The overall view of the operation of the code is given in Chapter 3. It describes the non-interactive and interactive commands in a qualitative way. A dictionary of all the non-interactive commands needed in the SHADOW code is given in Chapter 4. It gives the details for inputting each command. Chapter 5 provides the details for the interactive commands. The output features are interpreted in Chapter 6. Examples on how to use the code are given in Chapter 7. When first learning how to use the code, it is essential to be able to reproduce some of these examples to be sure that the code is functioning properly

on your system.

Part II of this document is a code manual. It goes into more specific information about the coding itself. It is of importance primarily for people implementing the code on a new system, for debugging errors, or for making changes in how the code operates. An overview of how the code is organized is given in Chapter 9. A listing of the code is given in Chapter 10. It is broken up into three parts for the non-interactive, FORTRAN 77 subroutines and into the interactive VAX dependent subroutines. The implementation of the code on a VAX is given in Chapter 11 and a brief description of implementing the code on a non-VAX computer is given in Chapter 12. A listing of an NCAR plotting code for the shadow map is given in Chapter 13.

Chapter 2

Method

The first gauge of the ability of two antenna systems to communicate with one another at high frequencies is to determine if there is a clear line of sight path between them. This can be conveniently represented by a map of the projected shadow on the far zone sphere caused by the structures around an antenna's environment. One method of producing a shadow map is to choose an observation point on the far zone sphere and then determine if anything obscures the path and then move on to the next point. This method is slow, however, because there must be many repeated tests on the same blocking structures for the various observation point making up the shadow map.

In order to have quick turn around for antennas mounted on large structures, it is desirable to use a method that will directly produce the shadow projected onto the far zone sphere. This can be accomplished in a two step process. First the outside boundary of each individual piece making up the structure can be transformed from the x, y, z coordinate system into a sequence of lines in the θ and ϕ pattern coordinate system. The area of the shadow map between the boundary lines for each piece may then be filled by looking at the center location between the lines and a shadow check on that piece of the structure can be performed. This reduces the test on each piece of structure from once every observation point to a few tests every pattern cut line. The calculation time, in general, is reduced by about two orders of magnitude. For example, instead of taking two hours, a map can conservatively be produced in about one minutes or better. These numbers dependent on the geometry, the window size of the map, and the resolution desired.

There are two fundamental types of structural pieces presently available for modeling in this obscuration code, the multiple sided flat plate and the multiple rimmed composite cone frustum of elliptic cross section. More than one plate or cylinder can be specified to build up a complex structure. A plate can be defined by the location of its corners in a reference coordinate system. A cone frustum can be defined by the size of its major and minor radii for each rim making up the composite cylinder.

The boundary of the structures are traced onto the far zone sphere by defining a vector from the source position, \bar{R}_s , to some position along its outer boundary, \bar{R}_i , such that

$$\bar{R} = \bar{R}_i - \bar{R}_s.$$

In the case of the plate, the boundary is defined by some location along its edges, as illustrated in Figure 2.1. This vector can then be transformed into the pattern cut coordinate system, since the pattern may be defined relative to a different set of axes. The vector can

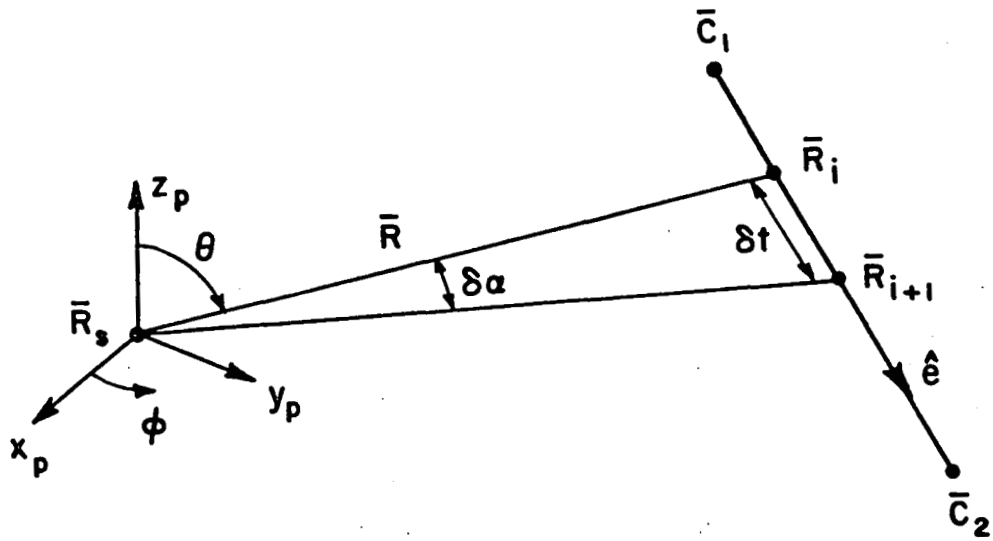


Figure 2.1: Geometry showing the projection of the plate edge onto the far zone sphere.

then be transformed onto the two dimensional far zone sphere by

$$\theta = \arctan \left(\hat{\rho} \cdot \bar{R} / \hat{z} \cdot \bar{R} \right)$$

and

$$\phi = \arctan \left(\hat{y} \cdot \bar{R} / \hat{x} \cdot \bar{R} \right).$$

The position of the vector along the edge is defined by starting at a corner and then incrementing the edge in steps of δt along the edge. In order to provide the most efficient performance and the best image of the shadow on the map, it is necessary to define δt as a function of the chosen resolution desired for the map, $\delta\alpha$, the distance, R , from the source to the edge point and the relative position of the projected shadow point with respect to the polar caps, *the Greenland effect*.

The resolution, $\delta\alpha$ is chosen to be the minimum of the two specified incremental values of θ and ϕ . The distance R is defined as $R = |\bar{R}_s - \bar{R}_i|$. Assuming that the resolution increment is small and the distance is relatively large, the value of the edge increment is given by

$$\delta t = \delta\alpha R \sin \theta.$$

The new edge point then becomes

$$\bar{R}_{i+1} = \bar{R}_i + \delta t \hat{e},$$

where \hat{e} is the edge vector pointing from the first corner to the second corner making up the edge.

The composite cone frustum can be done in the same way as the plate. In fact the end caps can be defined as plates with curved edges and the curved surfaces are added as edges whose corners are the tangent points illustrated in Figure 2.2.

Once a give plate or cylinder outer boundary is transformed onto the shadow map and stored in pixels of the desired resolution, the fill process can begin. The pixel array is considered one row at a time in a scanning operation from the one range of theta embodied in the pixel array to the other. The direction of the scan and the order in which rows are scanned is arbitrary. The fill process is the same for each scan line in the pixel array so that no logical interaction between lines takes place. The process is similar to the way in which a television paints pictures one row at a time on the screen. As the scan proceeds say from left to right, unlit pixels between object boundaries on the line which correspond to regions in the interior of the object are turned on creating an area fill. The decision to light a group of pixels on a given row is not made by testing each pixel individually for obscuration but by making a single test between the pixels which represent boundaries of the projected regions. In this way, only a single test is made to determine whether a whole group of pixels represent the interior or exterior of a region. This is one major key to the sizable reduction of processor time achieved.

The shadow test for a plate is made by first projecting the vector chosen at the mid point of the scan line, \hat{r} , onto the plane of the plate to find its intersection point, as shown in Figure 2.3, that is

$$\bar{R}_t = \bar{R}_s - \frac{[\hat{n} \cdot (\bar{R}_s - \bar{C}_1)] \hat{r}}{\hat{n} \cdot \hat{r}}.$$

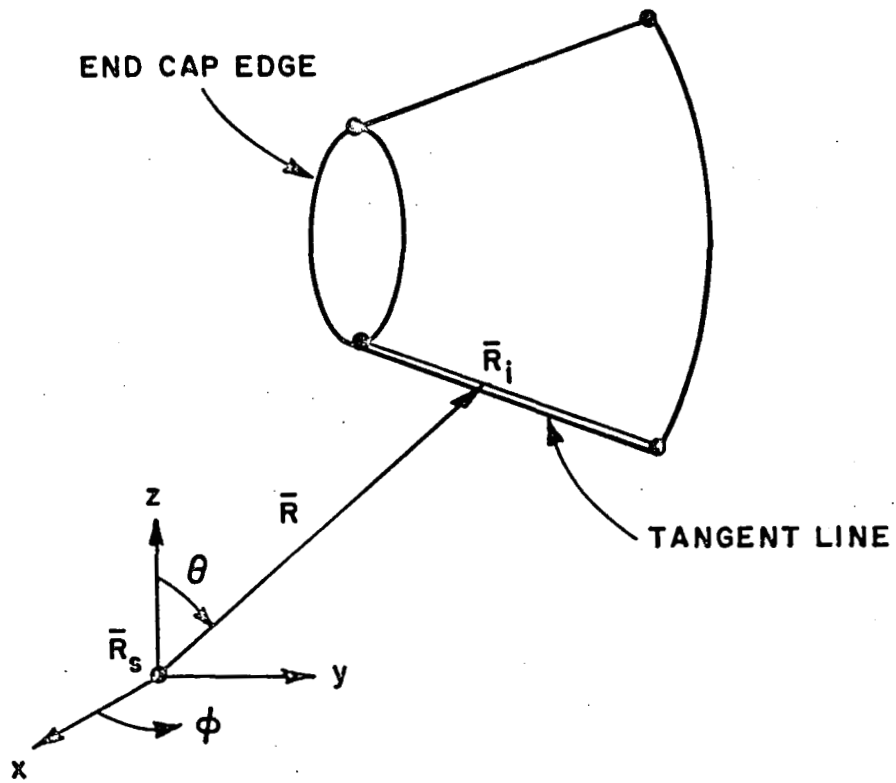


Figure 2.2: Geometry showing the projection of a cone frustum onto the far zone sphere.

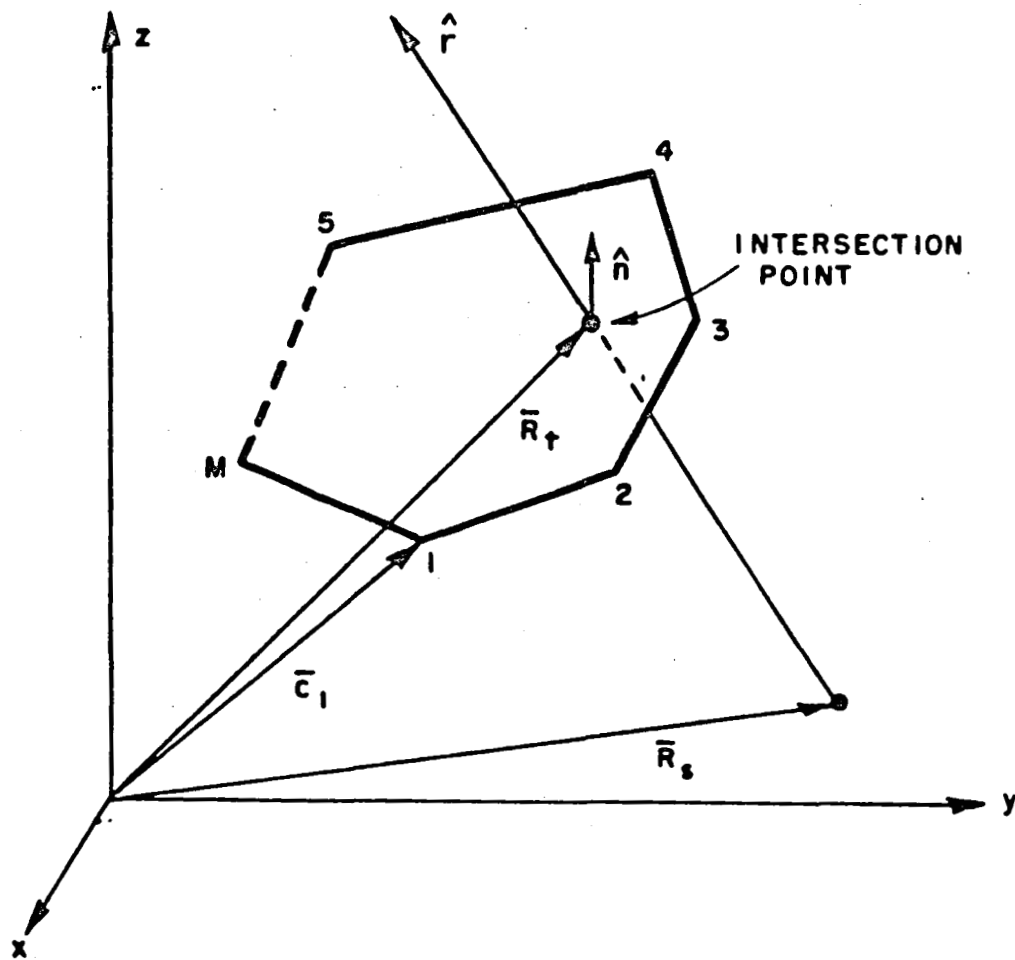


Figure 2.3: Intersection of observation direction vector with plate.

Now, using an idea based on Cauchy's formula from complex variables, that is,

$$\oint_C f(z) dz = \begin{cases} 0, & \text{no pole in } f(z) \\ 2\pi j, & \text{one pole in } f(z) \end{cases}$$

the intersection point can be tested to see whether or not it falls within the limits of the plate. This is illustrated in Figure 2.4.

It is easy to show that

$$\theta_m = \arctan \left[\frac{[(\bar{C}_m - \bar{R}_t) \times (\bar{C}_{m+1} - \bar{R}_t)] \cdot \hat{n}}{(\bar{C}_m - \bar{R}_t) \cdot (\bar{C}_{m+1} - \bar{R}_t)} \right]$$

which leads to the test, if

$$\left| \sum_{m=1}^M \theta_m \right| = \begin{cases} < \pi, & \text{no hit occurs} \\ > \pi, & \text{a hit occurs} \end{cases}$$

The end caps of the cone frustum cylinders can be done in the same way, by projecting the hit point in the plane of the end cap. The hit point distance can be tested from the center of the disk to see if it falls within the finite limits of a disk to simplify things a little. The curved surface test is a different matter, but still quite easy to accomplish. A vector on the surface of the cone frustum can be represented as

$$\bar{R}_c = \bar{R} + \bar{R}_s$$

or

$$\bar{R}_c = (R \cos \phi \sin \theta + x_s)^2 \hat{x} + (R \sin \phi \sin \theta + y_s)^2 \hat{y} + (R \cos \theta + z_s)^2 \hat{z}.$$

The geometry is illustrated in Figure 2.5. The point defined by \bar{R}_c should satisfy the equation for a cone, that is,

$$\frac{(R \cos \theta + x_s)^2}{a_j^2} + \frac{(R \sin \phi \sin \theta + y_s)^2}{b_j^2} - \lambda_j^2 (R \cos \theta + z_s) = 0$$

where

$$\lambda_j (R \cos \theta + z_s) = \left[1 + \frac{1}{a_j} \tan \theta_j (R \cos \theta + z_s - z_j) \right].$$

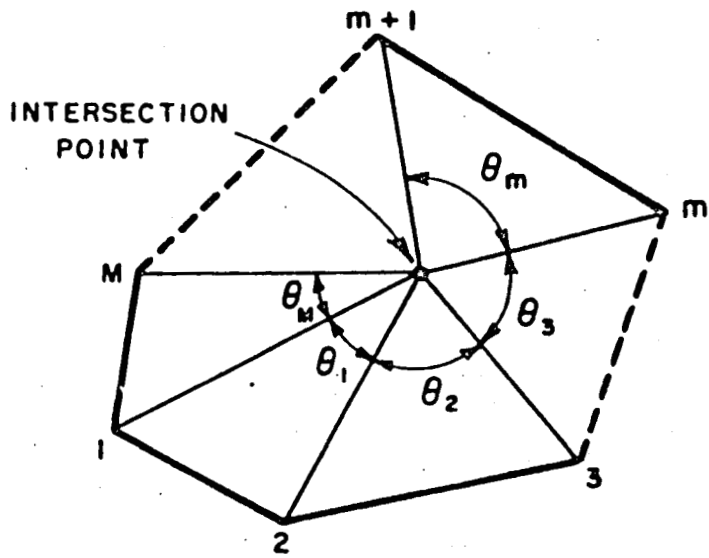
The distance R is unknown in this equation, since we know the direction to the observer, θ and ϕ , but not the distance to the hit point. We can solve for R, however, from the above equations using

$$\alpha R^2 + 2\beta R + \gamma = 0,$$

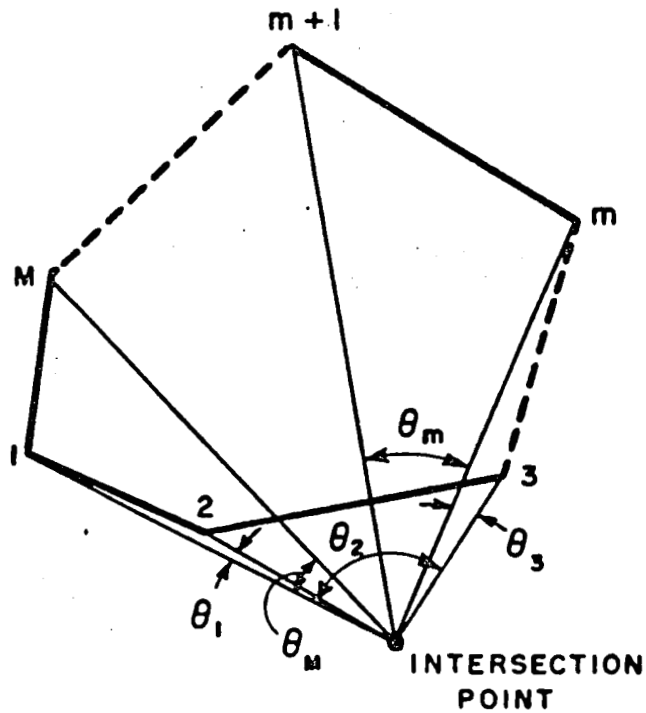
where

$$\alpha = \frac{\cos^2 \phi \sin^2 \theta}{a_j^2} + \frac{\sin^2 \phi \sin^2 \theta}{b_j^2} - \frac{\tan^2 \theta_j \cos^2 \theta}{a_j^2},$$

$$\beta = \frac{x_s \cos \phi \sin \theta}{a_j^2} + \frac{y_s \sin \phi \sin \theta}{b_j^2} - \tan \theta_j \cos \theta \lambda_j(z_s),$$



(a) RAY HITS PLATE



(b) RAY DOES NOT HIT PLATE

Figure 2.4: The geometry for deciding whether a ray does or does not hit the plate.

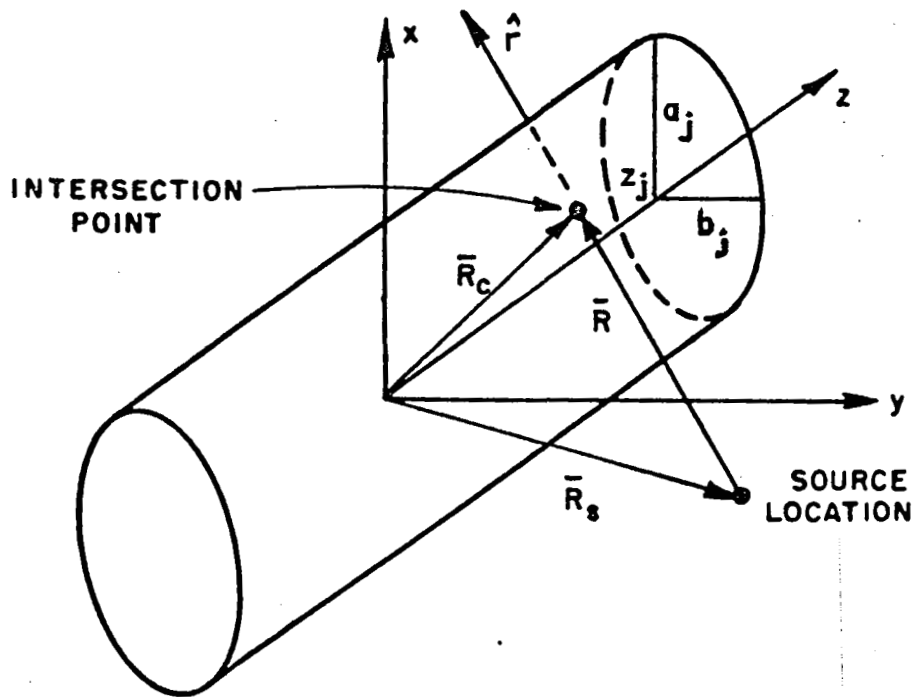


Figure 2.5: Geometry illustrating the hit point on a cone frustum segment.

and

$$\gamma = \frac{x_o^2}{a_j^2} + \frac{y_o^2}{b_j^2} - \lambda_j^2(z_o).$$

If the value of R is real, then the hit point is on the finite cone frustum and therefore the ray from the source to observer is shadowed. If the actual hit point is desired it should be noted that there are two values found from this equation, and that the right hit point can be found from the one representing the shortest distance. If the value of R is imaginary, however, this indicates that the hit point is off the real boundary of the cone frustum and therefore the ray is not shadowed. If R is real, an additional test must be made to decide whether the hit point is between the finite length bounds of the frustum.

The basic theory discussed here is rather straight forward. The implementation, of course, requires a lot of other considerations to be user friendly and as general purpose as possible. The next chapter will go into more detail about how the code interfaces with the operator.

Chapter 3

Principle of Operation

3.1 Overview

The Obscuration Code is intended to be an efficient means of determining the clear line of sight path for an antenna mounted in a complex environment. This code produces a shadow map of the geometry for a given source location. The configuration is defined using a command word system as discussed below. The geometry of the structure is defined by using plates and cylinders. It is thought that the obscuration code is just one step in a total evaluation scheme. The next step would be to either look at a "worst case" map that projects the location of the maximum lobes on to a volumetric map or to calculate the fields using a code like the NEC-BSC. In any case, the real fields should be calculated as the final step whether an intermediate one is used or not. For this reason, the geometry definition is based on the NEC-BSC code method of inputting information.

The obscuration code, however, is a very efficient means of providing a shadow map. It can be run in a matter of minutes or less for a given shadow map. It is, therefore, felt that it can be most efficiently run interactively, that is with the user sitting at a terminal changing antenna locations, looking at the resultant maps, deciding where to try the next antenna location until the desired optimum spot is found to achieve a given performance. For this reason the code has been developed in two pieces. One is a standard FORTRAN 77 part that does the essential shadowing calculations. The second is an interactive part that allows the user to change the source locations and window size without leaving the code. Unfortunately, this second part of the code is by nature device dependent. This part has been written for the DEC VAX series of computers using version 4 of VMS. It uses system handlers for defining the commands discussed in the sections below for the interactive commands and the keypad mode. The keys on the keypad of VT100 or VT200 series terminals can be used to represent the typed commands. This will simplify the use of the code by reducing the amount of typing necessary.

This chapter tries to give a brief overview of the specifics needed to run the code by treating it as a black box. It is intended to just get the user comfortable with the overall philosophy of the obscuration code.

3.2 Modeling the Structures

The building blocks available for the obscuration code are composed of pieces that are an extension of version 2 of the NEC-BSC [1]. Structures can be modeled using multiple sided flat plates and multiple rimmed cone frustum cylinders. The plates can be used individually to model things like solar panels or together to form box like structures to model things like the mast, etc. The cone frustums are a new feature here, and can be handy for modeling living modules, etc. Examples of space station models are given in Chapter 7.

Unlike the NEC-BSC, there are no real restrictions on how these structures are defined. Since the code just looks at each defined piece of the modeled structure individually, casts its shadow, then moves on to the next piece, it does not have to properly account for the wedge angles and other geometrical features needed in field calculations in the NEC-BSC. If one is setting up a model, however, it still might be useful to use the same modeling considerations as the NEC-BSC, such as defining the corners of a plate so the normals point in the region of space in which the source is located. It is assumed that the obscuration code phase of the design procedure will be followed by calculating the fields for the antenna on the structure using a code such as the NEC-BSC.

The number of plates and cylinders that can be used in the models is dictated only by the size of the dimensions implemented in the array for defining the geometry in the code. For convenience, these parameters are located in one file in the code so they only need to be changed in one spot. The details are given in Part II.

More information on how models are to be constructed are given in the section below on the non-interactive commands and in Chapter 4 where these commands are defined in more detail.

3.3 Running the Code

The first step of course is to get the code implemented on your system. The details of how to accomplish this are given in Part II. In order to use the full interactive features of the code, it is necessary to use the code on a DEC VAX. Many of the interactive features use VAX dependent implementations from version 4 of the VMS operating system. The code has been divided into standard FORTRAN 77 files and VAX dependent files, however, so that the code can be used without the interactive features on other systems. A slightly different main program needs to be used as provided in Part II. In addition, the non FORTRAN 77 INCLUDE statement has been used in the non interactive file. Many systems have this feature, so it was left in as a convenience. If the user system does not, it is easy to remove by hard wiring the lines in the appropriate file in place of the INCLUDE statement. Most of the information, here, will assume that the full features of the code will be able to be used.

The first step in using the code is to create a file that contains the basic structure definitions using the non-interactive commands discussed in the next section. The command defining the source location and window size of the shadow map can also be defined in this input file or added and changed in the interactive session. Of course, if you are running non-interactively, then all the data must be input from the input file.

Once the input file has been created or chosen from some stored files, the obscuration

code can be executed. It will read the input file from logical unit #5. An interactive command allows the user to connect the chosen input file to this logical unit number. The code then proceeds to read the input information and produce an output that is sent to the terminal (logical unit #6) representing how the code has interpreted the input. In this process, it is converting all the input into a standard reference coordinate system and into a common set of units which is meters. If there is a typographical error or other error in the input set the code will indicate so and stop execution at that point.

If the code completes the input, it will wait for the next instruction. For example, the output file name can be connect to the logical unit which is #7 for the line printer output and logical unit #10 for the plotter output. The antenna position can now be defined or modified and the desired window changed. The code can than be told to proceed to produce a shadow map.

When the code has completed the shadow calculation, the user can change the source location, the window of the map, or input another structure and run the code again; or he can print the map out. The map is an array of pixels (doubly dimensioned character array) that are in general either a blank representing a clear path or a character representing a blocked path. The character is normally a uniform character such as an "X". There is an option to tag a particular plate with a character that you define, or the code will letter each plate and cylinder separately. This is useful to determine which plates get in the way or for debugging purposes. More details on this will be given in Chapter 6.

3.4 Non-Interactive Commands

The non-interactive commands needed in this code are a subset and a slight extension of those used in version 2 of the NEC-BSC [1]. The total list of the available non-interactive commands are given in Table 3.1. Only the commands of interest to the obscuration code are defined in this manual. The rest can be found in reference [1] or in later reports and manuals for newer versions of the codes. This section is intended to give the user a brief overview of the specific commands of interest with the details coming in the next Chapter.

The input commands words are intended to make it convenient for the user to define the geometry of the structure without having to define information not needed or repeat information already defined. They are two letter pairs. The rest of the characters on a command word line can be used for comments, since only the first two letters are interpreted.

There is a place in the code to place default data that will be present without a call to the command. This is convenient when a specific resolution sized of the shadow map is desired as a default, for example. The default window can be initial theta angles of 0 to 180 in steps of 2 degrees and initial phi angles of 0 to 360 in steps of 2 degrees by defining the proper variables in this default section. A call to the VF command will over ride this data if it is specified in the input set.

The geometry information is by default assumed to be in meters in a definition coordinate system that is initially the reference coordinate system. The units can be changed using the UN command to either inches or feet or back to meters again. Once the UN command is specified all information after that command is assumed to be in those units unless changed by another call to UN. There is also provision for using any conversion fac-

COMMAND	DEFINITION	LOCATION
BP	back or bistatic scatter	[1]
CC	cone frustum geometry	pg 24
CE	last or only comment	pg 30 or [1]
CG	cylinder geometry	pg 27 or [1]
CM	comment card	pg 30 or [1]
EN	end execution	pg 30 or [1]
FM	swept frequencies	[1]
FR	frequency	[1]
GP	infinite ground plane	pg 31 or [1]
GR	range gate	not documented
LP	line printer output	[1]
NC	next set of cylinders	pg 32 or [1]
NG	no ground plane	pg 32 or [1]
NP	next set of plates	pg 32 or [1]
NR	next set of receivers	[1]
NS	next set of sources	pg 32 or [1]
NX	next problem	pg 32 or [1]
PD	far zone pattern cut	[1]
PF	far zone cut (non integer)	not documented
PG	plate geometry	pg 33 or [1]
PN	near zone pattern cut	[1]
PP	plotter output	[1]
PR	gain or coupling factors	[1]
RA	receiver array geometry	[1]
RD	far zone range	[1]
RG	receiver geometry	[1]
RM	NEC-MOM receiver input	[1]
RT	rotate-translate geometry	pg 36 or [1]
SA	source array geometry	[1]
SG	source geometry	pg 38 or [1]
SM	NEC-MOM source input	[1]
TO	test options	[1]
UF	model scale factor	pg 42 or [1]
UN	units of geometry	pg 42 or [1]
US	units of source size	[1]
VD	volumetric cut (integer)	not documented
VF	volumetric far zone cut	pg 44
VN	volumetric near zone cut	not documented
VP	volumetric plotter output	not documented
XQ	execute code	pg 47 or [1]

Table 3.1: Table of non-interactive commands.

tor desired. It is input using the UF command and is a scale factor multiplying times all the input dimensions in whatever unit have been defined. The code then takes the input information and changes it internally and stores it in meters, in order to have a uniform system in which to operate. The input dimensions and the internal dimensions are output in the feed back print out sent to a file so the user can see what happened. The dimensions of the source itself, that is length and width not its position, is handled with a default of wavelengths. This can be changed with the US. The length and width of the source is not important in this code so it can be ignored here.

The reference coordinate system is really whatever is convenient for the user. The definition coordinate system is the same as this initial reference system or it can be changed using the rotate translate command RT. The RT command allows the user to relocate the origin and orientation of the definition coordinate system with respect to the reference coordinate system. The definition system stays as defined for all subsequent geometry input until it is changed. The RT command's definition is always referenced to the reference coordinate system NOT to itself, that is, one does not put in inverse locations and angles to undo the command, but resets it to the zero position of the origin and the z-axis and x-axis of the reference coordinate system. Note that all angles are assumed to be input in degrees. The coordinate axes are input in a uniform way through out the code by treating the new axes vectors as if they were radial vectors in the system being used. That is the z-axis is defined using a theta and phi angle relative to the reference coordinate system in the RT command and likewise the x-axis is treated as a radial vector. The y-axis is defined by a cross product between the x and z axes. The code checks that the x and z axes were defined orthogonal to one another. If not an error message will result and the code will stop.

The geometry commands are the PG command for the plates, the GP command for the infinite ground plane, the CG command for an elliptic cylinder, and the CC command for the cone frustum cylinders. The plates are defined by inputting the number and location of their corners in the definition system. The ground plane is defined as a infinite plane lying in the x-y plane of the definition coordinate system. The elliptic cylinder definition is base on the location of its origin and the orientation of its z- and x-axes relative to the definition coordinate system. In addition, the radius along its cylinder x-axis and the radius along its cylinder y-axis, along with the z-axis position and angular orientation of its end caps are needed. The cone frustum's definition is similar except that the number of rims making up the cylinder need to be specified and the orientation of the rims does not, since they can't be cut at an angle as in the elliptic cylinder case. For the plates and cylinders the code automatically adds up the number of calls to the commands and counts that as the number of plates or cylinders specified. Only one infinite ground plane can be defined.

The location of the sources are specified by their location, type, orientation, and relative weights using the SG command. Only the location information is important to the obscuration code. Each source specified is automatically counted and remember as the number of sources. Unlike the plates and cylinders, the obscuration code only calculates one source at a time for a shadow map. In non-interactive mode, it does one source at a time producing a map for each. In interactive mode, it takes the first one as the default source and then each subsequent one needs to be interactively input. Receivers are not recognized by the shadow code, so if in reality you are studying a receiver, it must be input as a source not a

receiver for shadowing purposes.

In order to negate already defined commands for the geometry which is automatically increasing their number, a series of commands have been implemented. The plates can be reset to zero using the NP command. The ground plane with NG, the cylinders with NC, the sources with NS, and the entire run can be reset with the NX command.

The code is told to go and execute the interactive mode if it is available, or to go and execute the shadow calculations if the interactive mode has not been implemented using the XQ command. The EN command tells it to exit back to the operating system.

The next section will discuss an overview of the interactive commands and examples of these commands are given in Chapter 7.

3.5 Interactive Commands

The interactive commands provided by the code under VMS are designed to allow easy specification of commonly changed parameters with a syntax which is well-known to users of VMS, the DCL command interpreter syntax. To acquaint the reader with the appearance of these commands, they are summarized below. Detailed descriptions of each command complete with examples can be found in the Chapter 5 on interactive commands. A list of the available interactive commands are given in Table 3.2.

There are interactive commands to allow the user to control the operation of the code or to change or view the geometry. The SHADOW command produces the shadow map. The HELP command gives a descriptions of the commands. The EXIT command exits the user back to the operating system. The SPAWN command allows the use of DCL command while the user is still in the shadow code.

The rest of the commands either allow the user to change the geometry, with the SHOW commands, or see the present status of the geometry, with the SET commands. Most of them have a non-interactive command to which they are at least somewhat associated. The SET UNITS command allows the units of the antenna location to be chosen, similar to the UN command. The SET SCALE_FACTOR command is like the UF command, which allows an arbitrary scale factor for the geometry to be chosen. The SET COORDINATES command allows the definition coordinate system to be change, like in the RT command. The SET ANTENNA command enables the user to interactively specify the antenna location in the definition coordinate system. It is related to the SG command. The SET PATTERN_CUT command allow the user to specify the orientation of the pattern coordinate system in the reference coordinate system. The SET WINDOW command enables the initial, final and incremental angles of the shadow map to be specified. These two commands are related to the VF command.

The next four commands do not have non-interactive commands to which they are related. The SET INPUT command allows the user to specify what file containing the non-interactive commands is to be read. The SET OUTPUT command enables the specification of which output files are to be assigned and their names. The SET FILL_CHARACTER command allows the user to define the symbols that are used for the plate and cylinder shadows. The SET KEYPAD_MODE command enables the VT100 keypad to be used for command definitions as is discussed in the next section, otherwise, the keypad can be used for numerical input. These four commands are discussed much more thoroughly in

3.6 Keypad Use

The definable keypad functions are available for the interactive version of the code only. The keypad definitions are made possible through the use of an integrated VMS screen/terminal management package called SMG. It is a collection of runtime library routines which perform terminal I/O and intercept the special sequences transmitted by the keypad keys. When one of these keys are pressed, the text definition associated with the key is substituted onto the command line. All of this I/O is transparent to the user so that he need only worry about making the initial keypad definitions. For more information about SMG, the reader is referred to the VMS runtime library reference manual.

The keypad definitions are initialized by a text file containing suitable "DEFINE/KEY" commands. The file is called SHADOW.KPD and must reside in the default directory of the user running the code. There is a template file provided with the code which may be customized by the user. The predefined definitions of the VT100 keypad are shown in Table 3.3. Note that the "gold" enables the lower case action in the top row, that is, in most case the "SHOW" operation instead of the "SET" operation.

COMMAND	LOCATION
EXIT	Page 49
HELP	Page 50
SPAWN	Page 54
SET ANTENNA.LOCATION	Page 56
SET COORDINATES	Page 58
SET FILL.CHARACTER	Page 59
SET INPUT.SET	Page 62
SET KEYPAD.MODE	Page 63
SET OUTPUT	Page 64
SET PATTERN.CUT	Page 66
SET SCALE.FACTOR	Page 67
SET UNITS	Page 68
SET WINDOW	Page 69
SHADOW	Page 52
SHOW ANTENNA.LOCATION	Page 71
SHOW COORDINATES	Page 72
SHOW FILL.CHARACTER	Page 73
SHOW INPUT.SET	Page 74
SHOW KEYPAD.MODE	Page 75
SHOW OUTPUT	Page 76
SHOW PATTERN.CUT	Page 77
SHOW SCALE.FACTOR	Page 78
SHOW UNITS	Page 79
SHOW WINDOW	Page 80

Table 3.2: Table of interactive commands.

PF1 gold	PF2 HELP	PF3 SHADOW	PF3 nokeypad SET KEYPAD
7 show SET OUTPUT	8 show SET INPUT	9 show SET ANTENNA	- show SET WINDOW
4 show SET SCALE	5 show SET UNITS	6 show SET COORD	' show SET PATTERN
1 show SET FILL	2 /cylin FILL /PLATE	3 show FILL /SEQUEN	Enter RETURN
0 SPAWN		· EXIT	

Table 3.3: VT100 keypad for SHADOW interactive commands.

Chapter 4

Non-Interactive Commands

The non interactive commands discussed in this chapter are a subset of the commands used for the NEC-BSC2. The shadow code will recognize the entire set of NEC-BSC2 commands plus a few new ones. The new commands and some of the old ones that are pertinent to this code will be described here. The following sections define in detail each command word and the variables associated with them. This chapter is organized in alphabetical order of the commands. It is intended to be used as a reference for the user. Chapter 7 will give specific examples using this input method.

The method used to input data into the computer is presently based on a command word system. This is especially convenient when more than one problem is to be analyzed during a computer run. The code stores the previous input data such that one need only input that data which has to be changed from the previous execution. Also, there is a default list of data so for any given problem the amount of data that needs to be input has been shortened. The command word options presently available are listed in Table 3.1 on page 16. The colon after the command word is not necessary and is sometimes used just to illustrate the separation between the command word and the space where comments can be inserted.

In this system, all linear dimensions may be specified in either meters, inches, or feet and all angular dimensions are in degrees. All the dimensions are eventually referred to a fixed cartesian coordinate system used as a common reference for the source and scattering structures. There is, however, a geometry definition coordinate system that may be defined using the RT command. This command enables the user to rotate and translate the coordinate system to be used to input any selected data set into the best coordinate system for that particular geometry. Once the RT command is used all the input following the command will be in that rotated and translated coordinate system until the RT command is called again. See below for more details. There is also a separate coordinate system that can be used to define a pattern coordinate system. This is discussed in more detail in terms of the VF command.

It is felt that the maximum usefulness of the computer code can be achieved using it on an interactive computer system. As a consequence, all input data are defined in free format such that the operator need only put commas between the various inputs. This allows the user on an interactive terminal to avoid the problems associated with typing in the field length associated with a fixed format. This method also is useful on batch processing computers. Note that all read statements are made on unit #5, i.e., READ(5,*), where

the "*" symbol refers to free format. Other machines, however, may have different symbols representing free format.

In all the following discussions associated with logical variables a "T" will imply true, and an "F" will imply false. The complete words true and false need not be input since most compilers just consider the first character in determining the state of the logical variable.

4.1 Command CC: Cone Frustum Geometry

This command enables the user to define the geometry of the finite elliptic conical cylinder structures to be considered. The geometry is illustrated in Figure 4.1. One call to this command defines one cylinder. The number of cylinders in the structure are automatically counted by the number of calls to this command.

READ: (XCL(N,MC),N=1,3)

_____ where _____

XCL(N,MC) This is a doubly dimensioned real variable. It is used to specify the location of the origin of the MCth elliptic cylinder relative to the definition coordinate system. It is input on a single line with the real numbers being the x,y,z coordinates of the origin which correspond to N=1,2,3, respectively.

READ: TCLZ, PCLZ, TCLX, PCLX

_____ where _____

TCLZ,PCLZ These are real variables. They are input in degrees as spherical angles that define the z_c -axis of the cylinder coordinate system as if it was a radial vector in the definition coordinate system.

TCLX,PCLX These are real variables. They are input in degrees as spherical angles that define the x_c -axis of the cylinder coordinate system as if it was a radial vector in the definition coordinate system.

Note that the new x_c -axis and z_c -axis must be defined orthogonal to each other. The new y_c -axis is found from the cross product of the x_c - and z_c -axes. _____

READ: NEC(MC)

_____ where _____

NEC(MC) This is a dimensioned integer variable which defines the number of edges the conical cylinder has.

READ: AC(NC,MC), BC(NC,MC), ZC(NC,MC)

_____ where _____

- AC(NC,MC)** This is a double dimensioned real variable which defines the radius of the NCth rim on the x_c -axis of the MCth elliptic cylinder.
- BC(NC,MC)** This is a double dimensioned real variable which defines the radius of the NCth rim on the y_c -axis of the MCth elliptic cylinder.
- ZC(NC,MC)** This is a double dimensioned real variable which defines the z position of the NCth rim along the z_c -axis of the MCth elliptic cylinder.

Note that the program will keep increasing the number of cylinders in the solution by the number of calls to this command unless the NC or NX commands are called to reinitialize the cylinder geometry. Also, the ellipticity of a conical structure should remain the same for the entire length of that structure. The most positive rim should be defined first until all NC rims are defined in descending order.

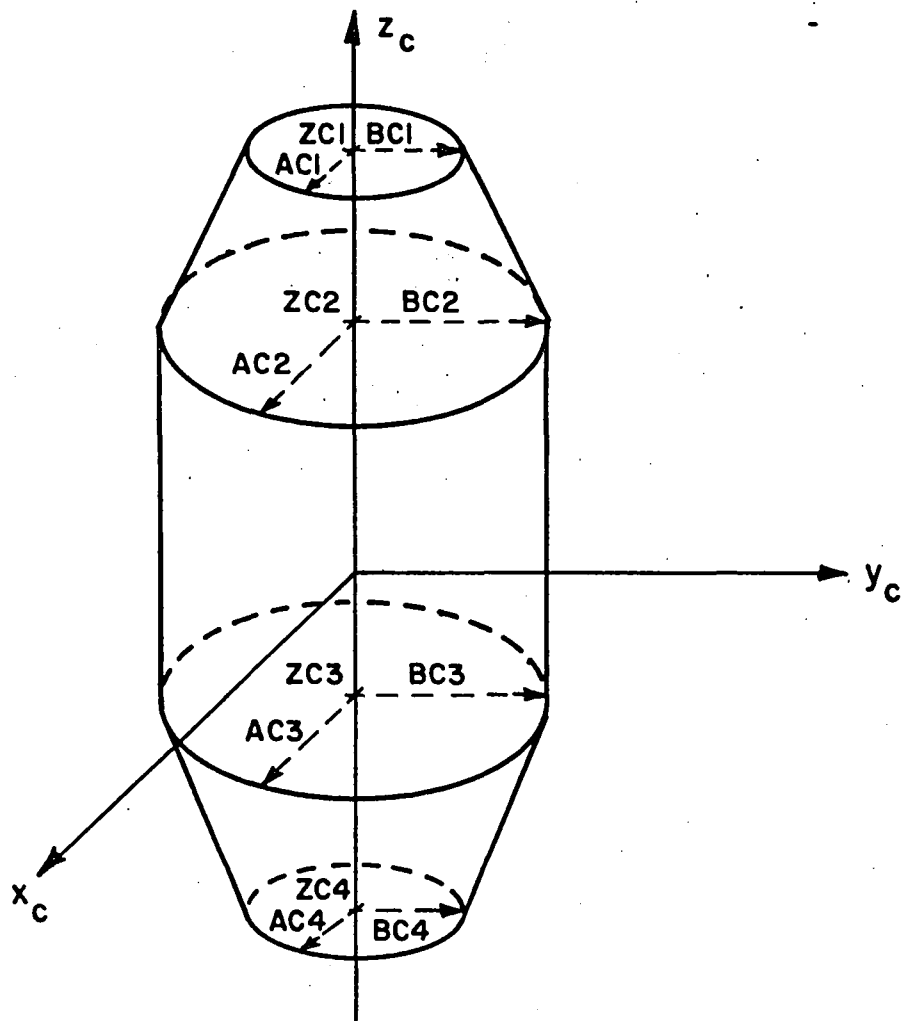


Figure 4.1: Definition of finite cylinder geometry composed of cone frustum segments with elliptic cross section.

4.2 Command CG: Cylinder Geometry

This command enables the user to define the geometry of the finite elliptic cylinder structures to be considered. The geometry is illustrated in Figure 4.2. One call to this command defines one cylinder. The number of cylinders in the structure are automatically counted by the number of calls to this command.

READ: (XCL(N,MC),N=1,3)

_____ where _____

XCL(N,MC) This is a doubly dimensioned real variable. It is used to specify the location of the origin of the MCth elliptic cylinder relative to the definition coordinate system. It is input on a single line with the real numbers being the x,y,z coordinates of the origin which correspond to N=1,2,3, respectively.

READ: TCLZ, PCLZ, TCLX, PCLX

_____ where _____

TCLZ,PCLZ These are real variables. They are input in degrees as spherical angles that define the z_c -axis of the cylinder coordinate system as if it was a radial vector in the definition coordinate system.

TCLX,PCLX These are real variables. They are input in degrees as spherical angles that define the x_c -axis of the cylinder coordinate system as if it was a radial vector in the definition coordinate system.

Note that the new x_c -axis and z_c -axis must be defined orthogonal to each other. The new y_c -axis is found from the cross product of the x_c - and z_c -axes. _____

READ: AC(1,MC), BC(1,MC)

_____ where _____

AC(1,MC) This is a double dimensioned real variable which defines the radius of the MCth elliptic cylinder on the x_c -axis of the cylinder.

BC(1,MC) This is a double dimensioned real variable which defines the radius of the MCth elliptic cylinder on the y_c -axis of the cylinder.

READ: ZCN, THTN, ZCP, THTP

_____ where _____

ZCN This is a real variable that defines the position the center of the most negative end cap on the z_c -axis of the cylinder.

THTN This is a real variable. It is input in degrees and defines the angle the surface of the most negative end cap makes with the positive z_c -axis in the x_c - z_c plane.

ZCP This is a real variable that defines the position of the center of the most positive end cap on the z_c -axis of the cylinder.

THTP This is a real variable. It is input in degrees and defines the angle the surface of the most positive end cap makes with the positive z_c -axis in the x_c - z_c plane.

Note that the program will keep increasing the number of cylinders in the solution by the number of calls to this command unless the NC or NX commands are called to reinitialize the cylinder geometry.

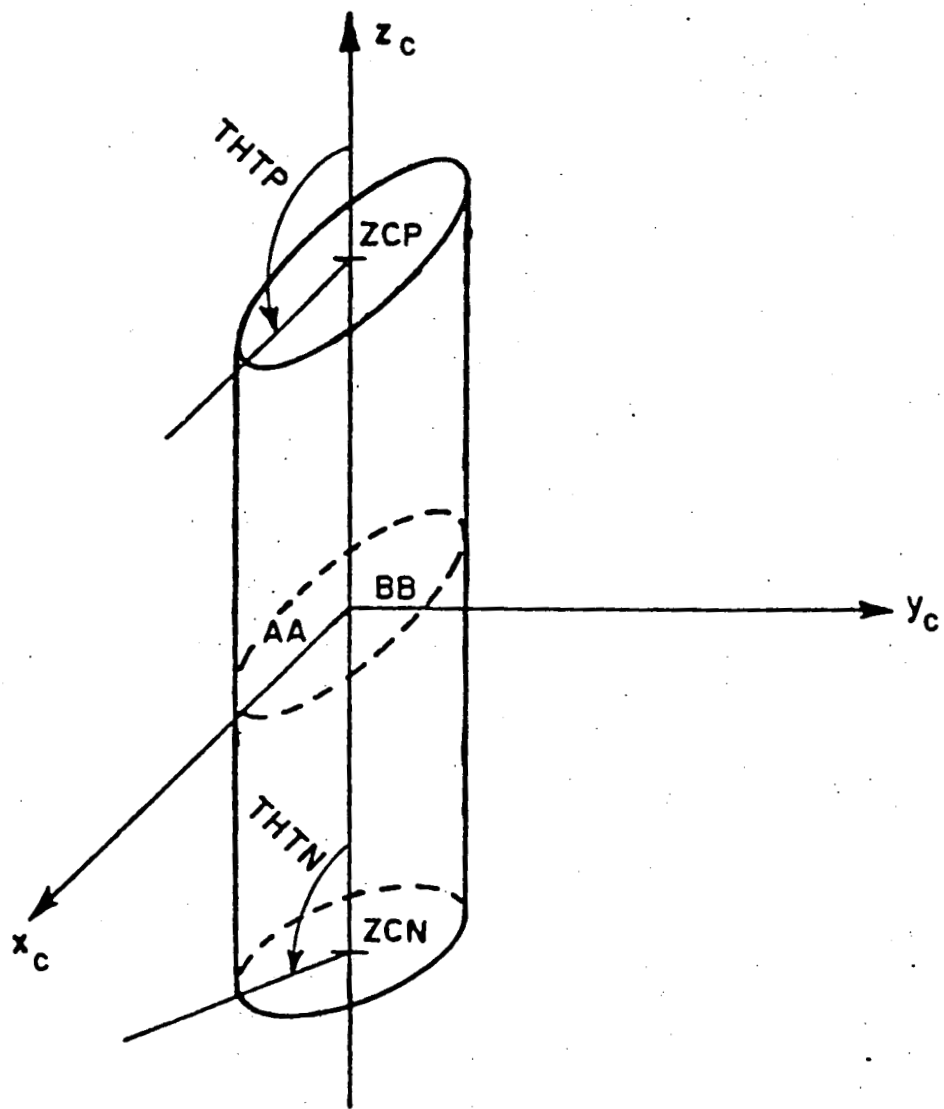


Figure 4.2: Definition of finite elliptic cylinder geometry.

4.3 Command CM: and CE: Comments

These commands enable the user to place comment cards in the input and output data in order to help identify the computer runs for present and future reference.

READ: (IR(I), I=1,36)

_____ where _____

IR(I) This is a CHARACTER*2 dimensioned array used to store the command word and comments. Each card should have CM or CE on them followed by an alphanumeric string of characters. The CM command implies that there will be another comment card following it. The last comment card must have the CE command on it. If there is only one comment card the CE command must be used.

Note that it is possible to place comments to the right of all the command words, if desired.

4.4 Command EN: End Program

This command enables the user to terminate the execution of the scattering code.

4.5 Command GP: Ground Plane

This command enables the user to specify an infinite ground plane in the x_t - y_t plane.

READ: LSLAB(MPDX)

_____ where _____

LSLAB(MPDX) This is a dimensioned integer variable. It is used to define the type of plate desired as follows:

0 = Perfectly conducting metallic plate

-3 = Dielectric half space

Note that if **LSLAB(MPDX)=0** the code will skip around the **READ** statement for the dielectric information, therefore, the next line defining the dielectric properties should not be placed in the input data set.

**READ: ERSLAB(1,MPDX), TESLAB(1,MPDX),
URSLAB(1,MPDX), TMSLAB(1,MPDX)**

_____ where _____

ERSLAB(1,MPDX) This is a doubly dimensioned variable. It is used to specify the relative dielectric constant of the half space.

TESLAB(1,MPDX) This is a doubly dimensioned variable. It is used to specify the dielectric loss tangent if the number is positive or the conductivity if the number is negative of the half space.

URSLAB(1,MPDX) This is a doubly dimensioned variable. It is used to specify the relative permeability constant of the half space.

TMSLAB(1,MPDX) This is a doubly dimensioned variable. It is used to specify the permeability loss tangent of the half space.

4.6 Command NC: Next Set of Cylinders

This command enables the user to initialize the cylinder data. All of the cylinders are removed from the problem unless they are respecified following this command.

4.7 Command NG: No Ground Plane

This command enables the user to initialize the infinite ground plane. The ground plane is removed from the problem unless it is respecified following this command.

4.8 Command NP: Next Set of Plates

This command enables the user to initialize the plate data. All of the plates are removed from the problem unless they are respecified following this command.

4.9 Command NS: Next Set of Sources

This command enables the user to initialize the source data. All of the sources are removed from the problem unless they are respecified following the command.

4.10 Command NX: Next Problem

This command enables the user to initialize the commands to their default conditions specified in the list at the beginning of the main program.

4.11 Command PG: Plate Geometry

This command enables the user to define the geometry of the flat plate structures to be considered. The geometry is illustrated in Figure 4.3. One call to this command defines one plate. The number of plates in the structure are automatically counted by the number of calls to this command.

READ: MEP(MP), LSLAB(MP)

_____ where _____

MEP(MP) This is a dimensioned integer variable. It is used to define the number of corners (or edges) on the MPth plate.

LSLAB(MP) This is a dimensioned integer variable. It is used to define the type of plate desired as follows:

- 1 = Transparent thin dielectric slab
- 0 = Perfectly conducting metallic plate
- 2 = Dielectric covered plate

Note that if LSLAB(MP)=0 the code will skip to the read statements associated with the corners XX(N,ME,MP). Therefore, the information for the different slab layers should not be put in the data list for the perfectly conducting plate.

READ: NSLAB(MP)

_____ where _____

NSLAB(MP) This is a dimensioned integer variable. It is used to define the number of dielectric layers on the MPth plate.

**READ: DSLAB(NS,MP), ERSLAB(NS,MP), TESLAB(NS,MP),
URSLAB(NS,MP), TMSLAB(NS,MP)**

_____ where _____

DSLAB(NS,MP) This is a doubly dimensioned variable. It is used to specify the thickness of the NSth layer.

ERSLAB(NS,MP) This is a doubly dimensioned variable. It is used to specify the relative dielectric constant of the NSth layer.

TESLAB(NS,MP) This is a doubly dimensioned variable. It is used to specify the dielectric loss tangent if the number is positive or the conductivity if the number is negative of the NSth layer.

URSLAB(NS,MP) This is a doubly dimensioned variable. It is used to specify the relative permeability constant of the NSth layer.

TMSLAB(NS,MP) This is a doubly dimensioned variable. It is used to specify the permeability loss tangent of the NSth layer.

Note there will be NSLAB(MP) number of lines of the above data.

READ: (XX(N,ME,MP),N=1,3)

_____ where _____

XX(N,ME,MP) This is a triply dimensioned real variable. It is used to specify the location of the Mth corner of the MPth plate. It is input on a single line with the real numbers being the x,y,z coordinates of the corner, in the specified coordinate system, which corresponds to N=1,2,3, respectively, in the array. For example, the array will contain the following for plate #1 and corner #2 located at x=2., y=4., z=6.:

XX(1,2,1)=2.

XX(2,2,1)=4.

XX(3,2,1)=6.

This data is input as: 2.,4.,6.

This read statement will be called MEP(MP) times so that all the corners are defined. As an example, the input data for the flat plate structure given in Figure 4.3, is given by

4,0	:corners and type of plate
1., 1., 0.	:corner #1
-1., 1., 0.	:corner #2
-1.,-1., 0.	:corner #3
1.,-1., 0.	:corner #4.

See elsewhere for further details on how to number the corners. Note that the program will keep increasing the number of plates in the solution by the number of calls to this command unless the NP or NX commands are called to reinitialize the plate geometry.

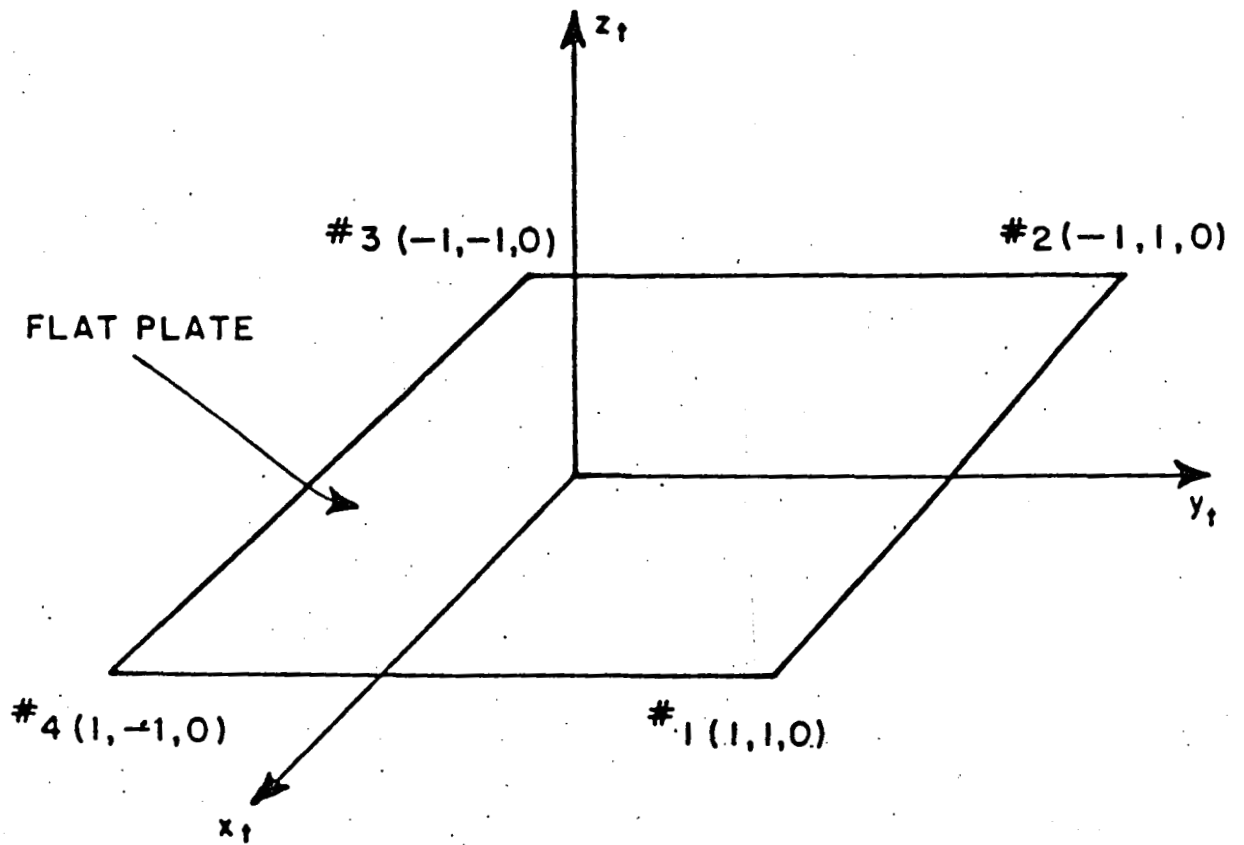


Figure 4.3: Definition of flat plate geometry.

4.12 Command RT: Rotate-Translate Geometry

This command enables the user to translate and/or rotate the coordinate system used to define the input data in order to simplify the specification of the plate, cylinder, and source geometries. The geometry is illustrated in Figure 4.4.

READ: (TR(N),N=1,3)

_____ where _____

TR(N) This is a dimensioned real variable. It is used to specify the origin of the new coordinate system to be used to input the data for the source or the scattering structures. It is input on a single line with the real numbers being the x,y,z coordinates of the new origin which corresponds to N=1,2,3, respectively.

READ: THZP, PHZP, THXP, PHXP

_____ where _____

THZP,PHZP These are real variables. They are input in degrees as spherical angles that define the z-axis of the new coordinate system as if it was a radial vector in the reference coordinate system.

THXP,PHXP These are real variables. They are input in degrees as spherical angles that define the x-axis of the new coordinate system as if it was a radial vector in the reference coordinate system.

The new x-axis and z-axis must be defined orthogonal to each other. The new y-axis is found from the cross product of the x- and z-axis. All the subsequent inputs will be made relative to this new coordinate system, which is shown as x_t, y_t, z_t , unless command RT is called again and redefined.

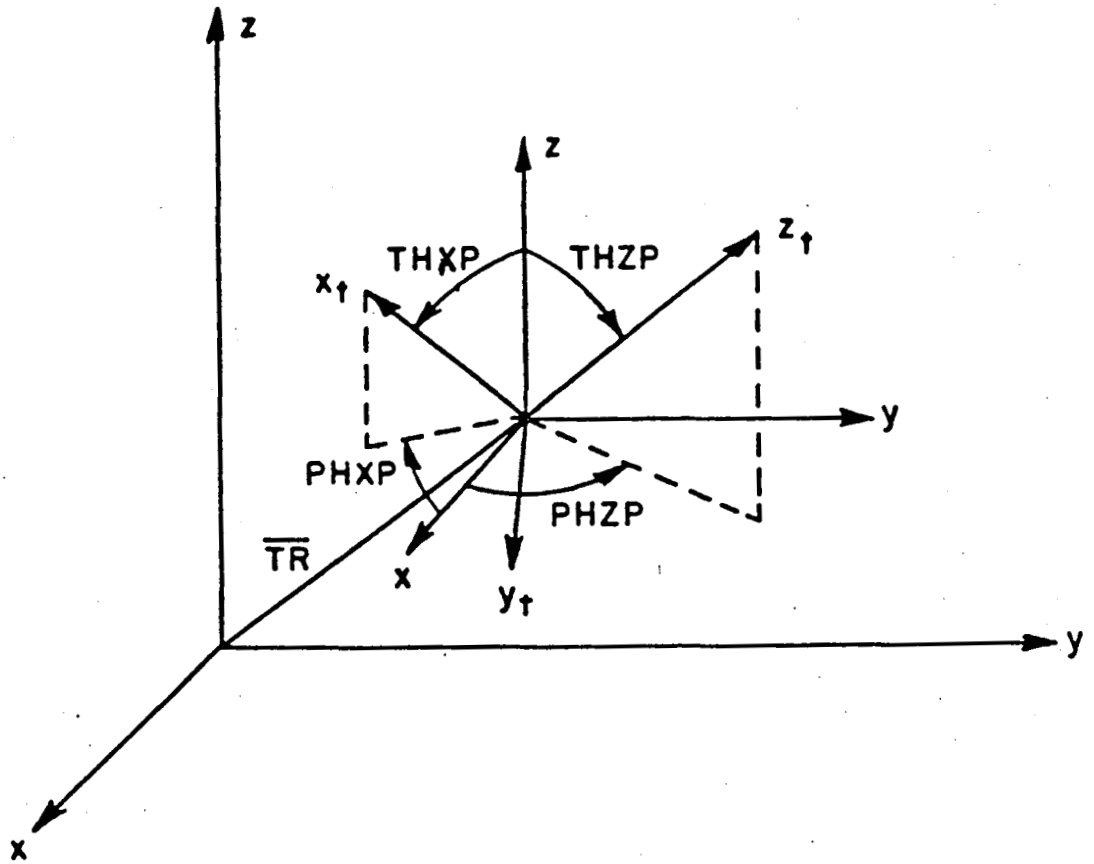


Figure 4.4: Definition of rotate-translate coordinate system geometry.

4.13 Command SG: Source Geometry

This command enables the user to specify the location and type of source to used. The geometry is illustrated in Figure 4.5 and 4.6. One call to this command defines one source. The number of sources in the problem are automatically counted by the number of calls to this command and the SA command.

READ: (XSS(N,MS),N=1,3)

_____ where _____

XSS(N,MS) This is a doubly dimensioned real array which is used to define the x,y,z location of the MSt^h element in the definition coordinate system. Again, a single line of data contains the x,y,z (N=1,2,3) locations.

READ: THSZ, PHSZ, THSX, PHSX

_____ where _____

THSZ,PHSZ These are real variables which are used to define the orientation of the MSt^h element in the definition coordinate system. They are input in degrees as spherical angles that define a radial direction which is parallel to the MSt^h current flow for a dipole antenna or which is parallel to the length of an aperture antenna.

THSX,PHSX These are real variables which are used to define the orientation of the MSt^h element in the definition coordinate system. They are input in degrees as spherical angles that define a radial direction which is parallel to the MSt^h elements aperture width or which is parallel to a slot's width. For a dipole antenna, these angles can be made in a convenient direction.

The x-axis and z-axis specified by these angles must be defined orthogonal to each other. The y-axis is found by the cross product of the x- and z-axes.

READ: IMS(MS), HS(MS), HAWS(MS)

_____ where _____

IMS(MS) This is an integer array which is used to define the MSt^h element's source type. The details of the different types of sources are given elsewhere. The designations are defined as follows:

IMS(MS)<0 for an electric element

IMS(MS)>0 for a magnetic element

$|IMS(MS)|=1$ for a uniform current distribution
 $|IMS(MS)|=2$ for a piece-wise sinusoidal distribution
 $|IMS(MS)|=3$ for a TE01 cosine current distribution

HS(MS) This is a real array which is used to input the length of the MSt^h element.

HAWS(MS) This is a real array which is used to input the width of the MSt^h element in the case of an aperture antenna. If $HAWS(MS)=0$, then it is assumed to be a dipole.

Note that the units of the variable $HS(MS)$ and $HAWS(MS)$ can be specified by the US command. If wavelength is chosen as the units then all the sources must be specified in wavelengths.

READ: WMS, WPS

_____ where _____

WMS, WPS These are real variables used to define the excitation associated with the MSt^h element. The magnitude is given by WMS and the phase in degrees by WPS.

Note that the program will keep increasing the number of sources in the solution by the number of calls to this command unless the NS or NX commands are called to reinitialize the source geometry.

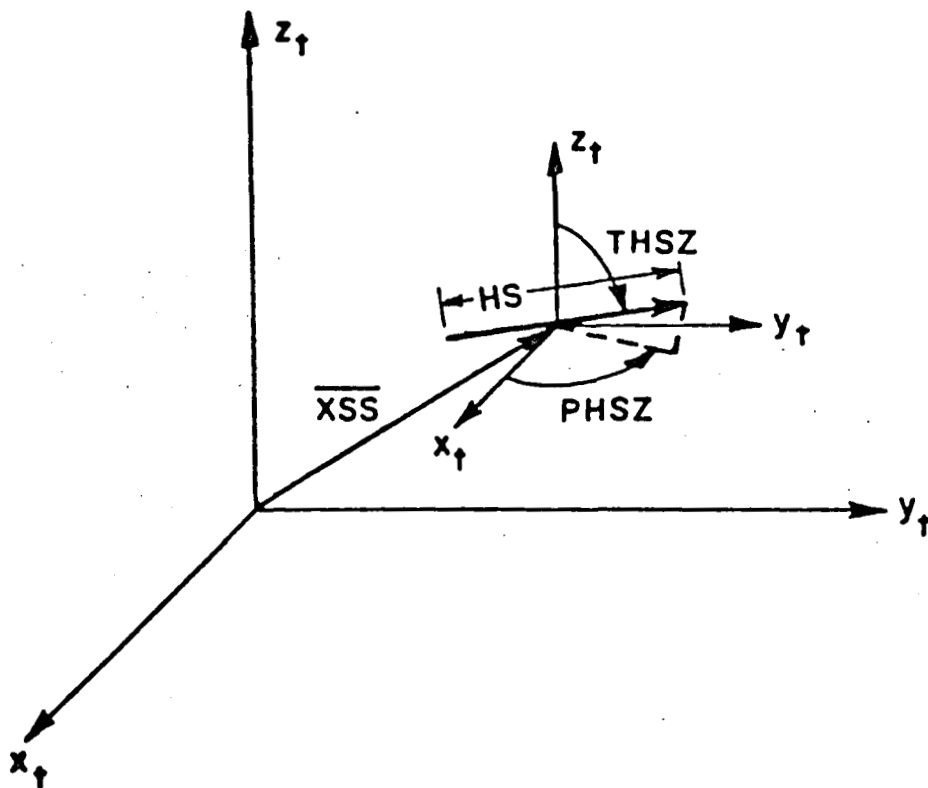


Figure 4.5: Definition of source geometry for dipole antennas.

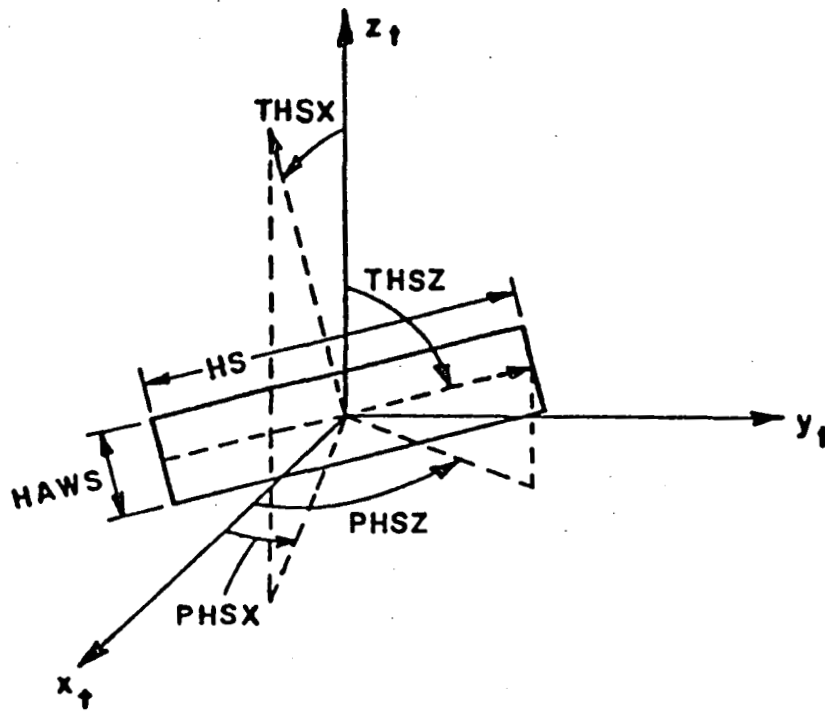


Figure 4.6: Definition of source geometry for aperture antennas.

4.14 Command UF: Scale Factor

This command enables the user to scale the linear dimensions that follow the command by the factor specified.

READ: UNITF

_____ where _____

UNITF This is a real variable that is used as a scale factor for all the linear dimensions that follow the command.

4.15 Command UN: Units of Geometry

This command enables the user to specify the units of all the linear dimensions to be input after the command is called. (The exceptions are the source length HS and width HAWS, and receiver length HR and width HAWR, see command US.)

READ: IUNIT

_____ where _____

IUNIT This is an integer variable that indicates the units for the input data that follows, such that

1= meters

2= feet

3= inches

4.16 Command US: Units of Source

This command enables the user to specify the units of the source length HS and width HAWS or receiver length HR and width HAWR to be input after the command is called. These variables are in the commands SG, SA, RG, and RA.

READ: IUNST

_____ where _____

IUNST This is an integer variable that indicates the units for the input data HS, HAWS, HR, HAWR that follows, such that if

0= wavelengths

1= meters

2= feet

3= inches

Note that if the units are specified to be wavelengths for one source it must be wavelengths for all the sources specified.

4.17 Command VF: Far Zone Volumetric Pattern

This command enables the user to define the far zone volumetric pattern coordinate system, the pattern cut, and the angular range that is desired. The geometry is illustrated in Figure 4.7.

READ: THCZ, PHCZ, THCX, PHCX

_____ where _____

THCZ,PHCZ These are real variables. They are input in degrees as spherical angles that define the z_p -axis of the pattern coordinate system as if it was a radial vector in the reference coordinate system.

THCX,PHCX These are real variables. They are input in degrees as spherical angles that define the x_p -axis of the pattern coordinate system as if it was a radial vector in the reference coordinate system.

Note that the new x_p -axis and z_p -axis must be defined orthogonal to each other. The new y_p -axis is found from the cross product of the x_p - and z_p -axes.

READ: LCNPAT, TPPD, TPPV, NPV

_____ where _____

LCNPAT This is a logical variable that defines the pattern cut desired, such that

T= The theta angle is held fixed while the phi angle is varied. The theta angle will then be incremented and another cut will be calculated.

F= The phi angle is held fixed while the theta angle is varied. The phi angle will then be incremented and another cut will be calculated.

TPPD This is a real variable. It defines the starting angle of the "fixed" angle specified by LCNPAT.

TPPV This is a real variable. It defines the incremental angle of the "fixed" angle specified by LCNPAT.

NPV This is a integer variable. It defines the number of pattern points of the "fixed" angle specified by LCNPAT.

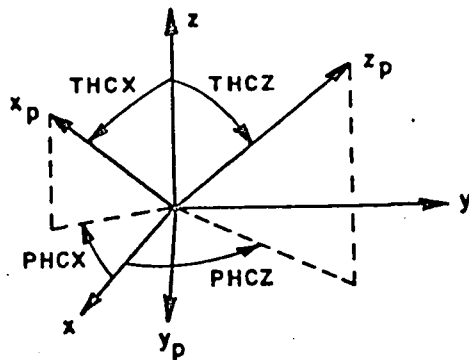
READ: TPPS, TPPI, NPN

_____ where _____

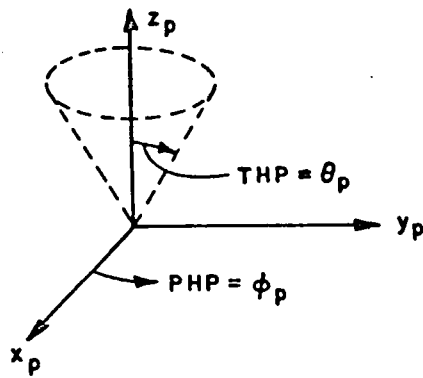
TPPS This is a real variable. It defines the starting angle of the "varying" angle specified by LCNPAT.

TPPI This is a real variable. It defines the incremental angle of the "varying" angle specified by LCNPAT.

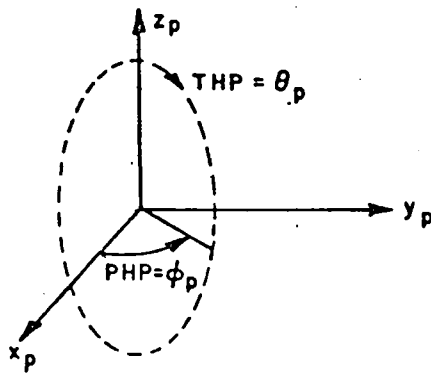
NPN This is a integer variable. It defines the anumber of pattern points of the "varying" angle specified by LCNPAT.



a. Definition of pattern coordinate system.



b. Conic pattern cut, LCNPAT=.TRUE., TPPD=THP.



c. Constant Phi pattern cut, LCNPAT=.FALSE., TPPD=PHP.

Figure 4.7: Definition of volumetric pattern coordinate system.

4.18 Command XQ: Execute Code

This command is used to execute the code so that the results may be computed. After execution the code returns for another possible command word.

Chapter 5

Interactive Commands

5.1 Overview

Facilities for interactive programs vary greatly from one operating system to the next with little or no standardization between systems. In spite of this, it was felt that the users of this code would benefit immensely from an interactive mode of operation. In order for the code to have interactive capability without an excessive amount of development time, the developers have used many features of the DEC VAX/VMS operating system. Since many engineers presently have access to the DEC VAX, it is felt that this will lead to reasonable transportability of the interactive mode for this code.

This decision has several ramifications for users of the SHADOW code. It means that the commands described in this chapter do not exist on computers that don't run VAX/VMS Version 4.0 or greater. It also means that this code has been separated into two parts, one standard FORTRAN 77 and the other VMS dependent containing the interactive facility, with a slightly different main program for the non-interactive code.

5.2 Command Descriptions

This section describes the interactive SHADOW commands in detail complete with examples for each. The syntax of the interactive commands is that of the Digital Command Language or DCL and for obvious reasons familiarity with the syntax of DCL is assumed throughout this chapter. For details about the utilities used to perform this DCL style command parsing, readers are referred to the VMS documentation concerning the Command Definition Utility or CDU.

EXIT

Causes the program to exit.

FORMAT

EXIT

Command Qualifiers
None.

Defaults
None.

restrictions

None.

prompts

None.

command parameters

None.

DESCRIPTION

All output files are closed, and control is returned to DCL.

COMMAND QUALIFIERS

None.

EXAMPLES

```
$ RUN SHADOW
SHADOW>
SHADOW> EXIT
$
```

This example shows how to exit the program.

HELP

Displays information about SHADOW commands or help text from any other library you specify.

FORMAT	HELP help-item	
	<hr/>	
	Command Qualifiers	Defaults
	/LIBRARY[=library-name]	/LIB=SYS\$DISK: []SHADOW

restrictions The indicated help files must exist.

prompts None.

command parameters help-item
The help-item is a keyword which is the item you want help on.

DESCRIPTION The SHADOW help command adheres to the conventions of VMS help libraries in form and content.

COMMAND QUALIFIERS /LIBRARY[=library-name]
/NOLIBRARY
Controls whether an alternate help library will be used in the search for topics. This qualifier must appear immediately after the HELP command or it will be interpreted as part of the help-item. If you specify /NOLIBRARY then no library will be searched.

EXAMPLES

SHADOW> HELP SET OUTPUT

. (SET OUTPUT help message)

Topic? EXIT

. (EXIT help message)

Topic? <RETURN>

SHADOW> HELP/LIBR=HELPLIB LOGOUT

. (LOGOUT help message from the system help library)

Topic? <RETURN>
SHADOW>

The above examples show how to get help about shadow topics and how to access other VMS help libraries with the HELP command

SHADOW

Initiates the obscuration calculation for the current antenna location and input geometry.

FORMAT SHADOW

Command Qualifiers	Defaults
None.	None.

restrictions None. Command may be abbreviated "S".

prompts None.

command parameters None.

DESCRIPTION The commands which alter parameters, such as SET WINDOW and SET ANTENNA do not initiate shadowing calculations automatically. This is to avoid redundant calculations when several parameters are changed at once. Once desired parameters are set, the SHADOW command performs the obscuration calculations and outputs the result.

COMMAND QUALIFIERS None.

EXAMPLES

SHADOW> SET ANTENNA

Input antenna location in meters: 11,22,32

Antenna in RCS (meters): 11.00000 22.00000 32.00000

Definit system (meters): 11.00000 22.00000 32.00000

SHADOW> SHAD

Working...

SHADOW> SET ANTENNA

Input antenna location in meters: 10,20,30

Antenna in RCS (meters): 10.00000 20.00000 30.00000

Definit system (meters): 10.00000 20.00000 30.00000

SHADOW> S

Working...

SHADOW>

The above commands all calculate the projected shadows for two different antenna locations on given input geometry. The results all go into the same output file, because no "SET OUTPUT" command was executed in between "SHADOW" commands.

SPAWN

Creates a subprocess for executing DCL commands without exiting the SHADOW program. This command is useful for executing DCL commands without reinitializing the context of a SHADOW program session.

FORMAT

SPAWN command-string

Command Qualifiers	Defaults
None.	None.

restrictions

A few restrictions are imposed by VMS.

- oThe RESOURCE_WAIT state must be enabled for the spawning process.
- oRequires TMPMBX or PRMMBX user privileges.
- oSPAWN does not manage terminal characteristics.

Command may be abbreviated "\$ ", where the blank after the \$ is necessary.

prompts

None.

command parameters

command-string

Specifies a DCL command string to be executed in the context of the subprocess. SHADOW will wait until the subprocess completes executing. If command-string is blank, the subprocess will prompt for commands repeatedly.

DESCRIPTION

The details of the spawn command are exactly as documented in the DCL dictionary, volume 2 of the VAX/VMS documentation set.

COMMAND QUALIFIERS

None.

EXAMPLES

```
SHADOW> SPAWN SHOW USERS
      VAX/VMS Interactive Users
      11-DEC-1985 08:34:18.72
      Total number of interactive users = 5
      Username      Process Name      PID      Terminal
```


CWP6148	CWP6148	00000891	VT1023:	TTA0:
DF6148	DF6148	00000AFA	VT1086:	TTC4:
EHN000	EHN000	000009F8	VT1085:	TTB7:
LT6199	LT6199	00000AEE	VT1081:	TTA7:
WE6148	WE6148	00000973	VT1082:	TTB2:

SHADOW> SPAWN
\$ SHO TIME
11-DEC-1985 08:36:13
\$ LOG
LT6199_1 job terminated at 11-DEC-1985 08:36:20.90
SHADOW>

The above spawn command illustrate how DCL commands may be executed without exiting the SHADOW program.

SET ANTENNA_LOCATION

Determines the location of the source point, or the center of the far-zone sphere for subsequent shadowing calculations.

FORMAT

SET ANTENNA

Command Qualifiers

None.

Defaults

None.

restrictions

Its recommended that the antenna not be placed in the interior of a cylinder. Unusual results may occur if this is done.

prompts

Input antenna location in meters:

Input antenna location in feet :

Input antenna location in inches:

command parameters

None.

DESCRIPTION

The antenna location consists of the (x,y,z) components of a vector in the current units and definition coordinate system, set by the SET UNITS and the SET COORDINATE commands, respectively. The command does NOT accept the antenna location on the command line, but prompts for it instead. The input syntax for the numbers is that of an unformatted FORTRAN read.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SET ANTENNA

Input antenna location in meters: 10,20,30

Antenna in RCS (meters): 10.00000 20.00000 30.00000

Definit system (meters): 10.00000 20.00000 30.00000

This example sets the antenna location to 10.,20.,30. (x,y,z) in the current units, which are meters.

EXAMPLES

SHADOW> SET UNIT FEET

SHADOW> SET ANT

Input antenna location in feet : 10,20,30

Antenna in RCS (meters):	3.04800	6.09600	9.14400
--------------------------	---------	---------	---------

Definit system (feet):	10.00000	20.00000	30.00000
-------------------------	----------	----------	----------

SHADOW>

This example shows how the antenna location is interpreted in the units of feet.

SET COORDINATES

Sets up a coordinate transformation to be applied to subsequent geometry.

FORMAT

SET COORDINATES

Command Qualifiers
None.

Defaults
None.

restrictions

The specified coordinate axes must be orthogonal to one another.

prompts

Please input a translation vector in feet:
Please input THZP,PHZP,THXP,PHXP in degrees:

command parameters

None.

DESCRIPTION

The antenna location may be specified relative to an alternative coordinate system. This coordinate system is established via the SET COORDINATES command. It does not affect the pattern cut coordinate system.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SET COOR

Please input a translation vector in feet : 100,200,300

Please input THZP,PHZP,THXP,PHXP in degrees: 0, -54, 265.5, 45

```
* The following rotations are used for ALL subsequent inputs: *
* VRS(1,1)= -0.70711 VRS(1,2)= -0.70711 VRS(1,3)= 0.00000 *
* VRS(2,1)= 0.70711 VRS(2,2)= -0.70711 VRS(2,3)= 0.00000 *
* VRS(3,1)= 0.00000 VRS(3,2)= 0.00000 VRS(3,3)= 1.00000 *
```

The above example shows how a default coordinate system may be established. The program echoes the established coordinate axes. These may be re-examined at any time with the SHOW COORDINATE command.

SET FILL_CHARACTER

Allows selection of the characters used to fill the output. Can be used to highlight particular elements of a geometry.

FORMAT	SET FILL [tag-character]								
	<hr/>								
	<table><thead><tr><th>Command Qualifiers</th><th>Defaults</th></tr></thead><tbody><tr><td>/SEQUENTIAL</td><td>None.</td></tr><tr><td>/PLATE=(num[,char])</td><td>None.</td></tr><tr><td>/CYLINDER=(num[,char])</td><td>None.</td></tr></tbody></table>	Command Qualifiers	Defaults	/SEQUENTIAL	None.	/PLATE=(num[,char])	None.	/CYLINDER=(num[,char])	None.
Command Qualifiers	Defaults								
/SEQUENTIAL	None.								
/PLATE=(num[,char])	None.								
/CYLINDER=(num[,char])	None.								
	<hr/>								
restrictions	None.								
	<hr/>								
prompts	None.								
	<hr/>								
command parameters	<p>tag-character</p> <p>Is any single ASCII character. If a lowercase letter is desired, enclose the letter in double quotes, i.e. "a". The default is "X".</p>								

DESCRIPTION

In order to better trace portions of a geometry through the shadowing process, the ability to tag a particular cylinder or plate has been added. The tag setting remains in effect until altered by a subsequent "SET FILL" command. The highlighted plate or cylinder appears in its entirety in the output regardless of its actual position in the hierarchy of obscuration. This allows the user to be absolutely certain of the shadowing caused by the particular highlighted geometry.

There are three tagging modes available. One is sequential tagging. In this mode, the code attempts to assign a unique character in the output to each plate/cylinder in the input. Plates are numbered beginning with "A" and increasing through the ASCII character sequence, and cylinders are treated the same way beginning with "1".

The second mode causes all parts of the geometry to be shaded with a single specified character such as "X". In this total obscuration mode, any one part of the input geometry is not easily identified — rather the total obscuration is presented homogeneously. It is specified using SET FILL without qualifiers. The third mode is the same as the second mode, but with the added feature of one single plate (or cylinder) highlighted with a different character. In this mode the relation of one part of the geometry to the rest is clearly

visible. This mode can be very helpful when isolating particular parts of a geometry that are shadowing the source.

**COMMAND
QUALIFIERS**

**/SEQUENTIAL
/SEQUENTIAL**

The **/SEQUENTIAL** qualifier selects the first mode of obscuration, sequential tagging of the input geometry. This qualifier may not be specified with a tag-character parameter nor with any of the other qualifiers.

**/PLATE=num
/PLATE=(num, char)**

The **/PLATE** qualifier selects the third mode of obscuration, homogenous tagging with highlighting of a particular plate.

The num argument is the number of the plate to be tagged. It is a required argument. The char argument is the ASCII character to be used when tagging the plate. It is optional, and defaults to "P" if unspecified.

This qualifier may not be specified in combination with other qualifiers. It is mutually exclusive with the **/CYLINDER** qualifier.

**/CYLINDER=num
/CYLINDER=(num, char)**

The **/CYLINDER** qualifier selects the third mode of obscuration, homogenous tagging with highlighting of a particular cylinder. It works exactly like the **/PLATE** qualifier.

The num argument is the number of the cylinder to be tagged. It is a required argument. The char argument is the ASCII character to be used when tagging the cylinder. It is optional, and defaults to "C" if unspecified.

This qualifier may not be specified in combination with other qualifiers. It is mutually exclusive with the **/PLATE** qualifier.

EXAMPLES

```
SHADOW> set fill
No individual plates/cylinders are tagged
All geometry marked by [X]
```

```
SHADOW> set fill $
No individual plates/cylinders are tagged
All geometry marked by [$]
```

```
SHADOW> set fill * /plate=6
Plate 6 is tagged with [P]
```

All other geometry tagged with [*]

```
SHADOW> set fill * /plate=(7,%)  
Plate 7 is tagged with [%]  
All other geometry tagged with [*]
```

```
SHADOW> set fill Q /cyl=(2,$)  
Cylinder 2 tagged with [$]  
All other geometry tagged with [Q]
```

```
SHADOW> set fill /plate=9 /cyl=4  
%CLI-W-CONFLICT, illegal combination of command elements
```

```
SHADOW> set fill Q /cyl=(2,$) /seq  
%CLI-W-MAXPARAM, too many parameters - reenter command with fewer parameters
```

```
SHADOW> set fill /seq ! Q /cyl=(2,$) /seq  
All cylinders/plates sequentially tagged
```

The above examples are obvious except possibly the last three. They show that the qualifiers are not allowed in combination, that the /SEQUENTIAL qualifier does not allow specification of a fill character, and that the DCL syntax ignores everything after an exclamation point.

SET INPUT_SET

Reads an input set from a named file

FORMAT SET INPUT_SET filename

Command Qualifiers	Defaults
None.	None.

restrictions The named input file must exist.

prompts input set:

command parameters filename
The name of the input set. It may be any valid VMS filename, including a logical name. The default filetype is .INP.

DESCRIPTION The set output command has the dual role of designating an input file and simultaneously causing that input set to be read and prepared for subsequent shadow commands. The current output files are NOT affected by this command so that several outputs may be concatenated. Normally though, this command would be entered after a SET OUTPUT command.

COMMAND QUALIFIERS None.

EXAMPLES

```
SHADOW> SET OUT AN5S1
Plotting file is: USER1:[RJM.NAS]AN5S1.PLT;1
Printer file is: _NLAO:[ ]FOR007.DAT;
Input echo file: USER1:[RJM.NAS]AN5S1.LIS;1
SHADOW> SET INPUT AN5S1
The current input set is
USER1:[RJM.NAS]AN5S1.INP;1
```

The SET OUTPUT command is used to set the output files – the printer output is discarded by default. The input set AN5S1.INP is then read and processed by the SET INPUT command.

SET KEYPAD_MODE

Causes the keypad state to change to non-numeric.

FORMAT	SET [NO]KEYPAD				
	<hr/>				
	<table><tr><td>Command Qualifiers</td><td>Defaults</td></tr><tr><td>None.</td><td>None.</td></tr></table>	Command Qualifiers	Defaults	None.	None.
Command Qualifiers	Defaults				
None.	None.				
	<hr/>				
restrictions	None.				
	<hr/>				
prompts	None.				
	<hr/>				
command parameters	None.				

DESCRIPTION The keypad of most DEC terminals can be in one of two states, numeric mode or keypad mode. In numeric mode, the keypad buttons represent the numbers and symbols printed on the keys. In keypad mode, the keys may be defined to provide functions, in much the same way as they do in DCL.

SET KEYPAD enables the defined-key feature of SHADOW, and SET NOKEYPAD returns the keypad to numeric-entry mode.

The keypad definitions are made in a session startup file called SHADOW.KPD; in the current default directory.

COMMAND QUALIFIERS	None.
---------------------------	-------

EXAMPLES

```
SHADOW> SET KEYPAD
The keyboard is in keypad mode.
SHADOW> SET NOKEYPAD
The keyboard is not in keypad mode.
```

SET OUTPUT

Determines the names of new output files and closes current output files.

FORMAT

SET OUTPUT filename

Command Qualifiers

/PLOTTABLE
/PRINTABLE
/ECHOING

Defaults

/PLOTTABLE
/NOPRINTABLE
/ECHOING

restrictions

The filename must be a valid VMS filename.

prompts

filename:

command parameters

filename

The name(s) of the newly created output file(s).

DESCRIPTION

There are three different outputs from the shadow program. One is an echo of the input set from the input processor. Another is a line printer output of the shadow map. The third is an output suitable for input to a separate plotting program. The set output command opens these files for the code. The name of the file opened is specified as the filename parameter. The filetypes are set by the command automatically, so that only the filename need be specified.

COMMAND QUALIFIERS

/PLOTTABLE
/NOPLOTTABLE

Causes a plottable output file to be produced. This is the default. Specifying /NOPLOT will override this default.

/PRINTABLE
/NOPRINTABLE

Causes an output file to be produced which is suitable for printing on a standard line printer. /NOPRINT is the default. Specifying /PRINT will override this default.

/ECHOING
/NOECHOING

Causes the input echo to be saved in a file when a new input set is

read. /ECHOING is the default. Specifying /NOECHO will override this default.

EXAMPLES

```
SHADOW> SET OUT AN5S1
Plotting file is: USER1:[RJM.NAS]AN5S1.PLT;1
Printer file is: _NLAO:[]FOR007.DAT;
Input echo file: USER1:[RJM.NAS]AN5S1.LIS;1
SHADOW> SET OUT AN5S1 /PRINT
Plotting file is: USER1:[RJM.NAS]AN5S1.PLT;2
Printer file is: USER1:[RJM.NAS]AN5S1.PRT;1
Input echo file: USER1:[RJM.NAS]AN5S1.LIS;2
SHADOW> SET OUT AN5S1 /NO PLOT /NOECHO /PRINT
Plotting file is: _NLAO:[]FOR010.DAT;
Printer file is: USER1:[RJM.NAS]AN5S1.PRT;2
Input echo file: _NLAO:[]FOR006.DAT;
SHADOW> SET OUT AN5S1
Plotting file is: USER1:[RJM.NAS]AN5S1.PLT;3
Printer file is: _NLAO:[]FOR007.DAT;
Input echo file: USER1:[RJM.NAS]AN5S1.LIS;3
```

The above examples show the operation of the SET OUTPUT command. Note that the printer file is not produced by default, and the device NLAO: (the null device) is where the output is discarded:

SET PATTERN_CUT

Specifies the pattern cut coordinate system.

FORMAT

SET PATTERN_CUT

Command Qualifiers

Defaults

None.

None.

restrictions

The specified coordinate axes must be orthogonal.

prompts

Please input THZP,PHZP,THXP,PHXP in degrees:

command parameters

None.

DESCRIPTION

The shadow map window is specified relative to the pattern-cut coordinate system. This system can be changed to facilitate easier specification of this window relative to the blocking object coordinate system.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SET PAT

Please input THZP,PHZP,THXP,PHXP in degrees: 0, -54, 265.5, 45, 0, 135

```
*   The following rotations are used for ALL subsequent inputs:   *
*   VPC(1,1)= -0.70711  VPC(1,2)= -0.70711  VPC(1,3)=  0.00000   *
*   VPC(2,1)=  0.70711  VPC(2,2)= -0.70711  VPC(2,3)=  0.00000   *
*   VPC(3,1)=  0.00000  VPC(3,2)=  0.00000  VPC(3,3)=  1.00000   *
```

The pattern-cut coordinate system shown has been set up.

SET SCALE_FACTOR

Sets a new value for the uniform scale factor.

FORMAT

SET SCALE_FACTOR

Command Qualifiers
None.

Defaults
None.

restrictions

The scale factor may not be specified on the command line.

prompts

Please input a uniform scale factor:

command parameters

None.

DESCRIPTION

In order to allow for more flexibility in specifying input, an additional scale factor may be applied to numerical inputs. The default value of this command is 1.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SET SCALE

Please input a uniform scale factor: 5.5

The uniform scale factor is 5.5000000

The uniform scale factor has been changed to 5.5.

SET UNITS

Sets the default units for the entry of numeric values. Allowable units are Meters, Feet, Inches.

FORMAT

SET UNITS keyword

Command Qualifiers

None.

Defaults

None.

restrictions

None.

prompts

inches, feet, or meters:

command parameters

Keyword may be one of the following:

- o METERS
- o FEET
- o INCHES

DESCRIPTION

When the antenna location is set, these are the units applied to the specified position. Internal calculations are always done in meters.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SET UNI FEET

This example sets the default units to feet.

SET WINDOW

Sets parameters for windowing of the output.

FORMAT

SET WINDOW

Command Qualifiers
None.

Defaults
None.

restrictions

The maximum span of theta must be less than 180 degrees. The maximum span of phi must be less than 360 degrees. The maximum resolution is a function of the specified range for both theta and phi. None of these parameters is specified on the command line.

prompts

Please enter a new range for theta (lower,higher):
Please enter a new THETA resolution in degrees/pixel:
Please enter a new range for phi (lower,higher):
Please enter a new PHI resolution in degrees/pixel:

command parameters

None.

DESCRIPTION

In order to be more flexible on the presentation of the output, a windowing feature was included so that portions of theta-phi space may be mapped onto a larger output surface. The set window command does this by prompting for the desired range of displayed theta and phi, and the desired levels of resolution. The default window displays the entire range of theta and phi at a resolution of 2 degrees/pixel in both directions.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SET WINDOW

The current range of theta in degrees is 0.000000E+00 to 180.0000
with a resolution of 2.000000 degrees/pixel.

The current range of phi in degrees is 0.000000E+00 to 360.0000
with a resolution of 2.000000 degrees/pixel.

Please enter a new range for theta (lower,higher): 30,40

Please enter a new THETA resolution in degrees/pixel: .5

Please enter a new range for phi (lower,higher): 45,90

Please enter a new PHI resolution in degrees/pixel: .5
The current range of theta in degrees is 30.00000 to 40.00000
with a resolution of 0.500000 degrees/pixel.
The current range of phi in degrees is 45.00000 to 90.00000
with a resolution of 0.500000 degrees/pixel.

The set window command above first displays the current window settings (which also happen to be the default settings), then prompts for new values. The new values are then also shown.

SHOW ANTENNA_LOCATION

Display the current antenna position.

FORMAT

SHOW ANTENNA_LOCATION

Command Qualifiers
None.

Defaults
None.

restrictions

None.

prompts

None.

command parameters

None.

DESCRIPTION

The antenna location is displayed in both the current default units and the Reference Coordinate System.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SHO ANT

Antenna in RCS (meters):	2.00000	3.00000	4.00000
Definit system (meters):	2.00000	3.00000	4.00000

This command displays the current antenna location in both the reference coordinate systems (RCS) and the current default units, which are also meters in this example.

SHOW COORDINATES

Displays the default transformation applied to antenna placement commands.

FORMAT	SHOW COORDINATES				
	<table><tr><td>Command Qualifiers</td><td>Defaults</td></tr><tr><td>None.</td><td>None.</td></tr></table>	Command Qualifiers	Defaults	None.	None.
Command Qualifiers	Defaults				
None.	None.				
restrictions	None.				
prompts	None.				
command parameters	None.				
DESCRIPTION	The antenna location is input in terms of an antenna coordinate system. This command displays the orientation of this system.				
COMMAND QUALIFIERS	None.				

EXAMPLES

SHADOW> SHO COORD

```
* VRS(1,1)= 1.00000 VRS(1,2)= 0.00000 VRS(1,3)= 0.00000 *
```

```
* VRS(2,1)= 0.00000 VRS(2,2)= 1.00000 VRS(2,3)= 0.00000 *
```

```
* VRS(3,1)= 0.00000 VRS(3,2)= 0.00000 VRS(3,3)= 1.00000 *
```

In this example, the antenna coordinate system is coincident with the reference coordinate system.

SHOW FILL_CHARACTER

Displays the current output fill modes.

FORMAT

SHOW FILL

Command Qualifiers
None.

Defaults
None.

restrictions

None.

prompts

None.

command parameters

None.

DESCRIPTION

The output may be generated in one of three modes. For a detailed description of the possible modes, see the SET FILL command.

COMMAND QUALIFIERS

None.

EXAMPLES

```
SHADOW> SHOW FILL
```

```
Plate 6 is tagged with [P]
```

```
All other geometry tagged with [*]
```

In the above example, the sixth plate of the input set is tagged with the ASCII character "P". The SET FILL command has many more examples.

SHOW INPUT_SET

Displays the name of the file from which the current geometry was defined.

FORMAT

SHOW INPUT_SET

Command Qualifiers
None.

Defaults
None.

restrictions

None.

prompts

None.

command parameters

None.

DESCRIPTION

The input set is determined with the SET INPUT command. The SHOW INPUT command echoes this input set filename.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SHOW INPUT

The current input set is

USER1: [RJM.NAS]AN5S1.INP;1

SHOW KEYPAD_MODE

Displays the current state of the keyboard.

FORMAT

SHOW KEYPAD_MODE

Command Qualifiers	Defaults
None.	None.

restrictions

None.

prompts

None.

command parameters

None.

DESCRIPTION

The keypad of most DEC terminals can be in one of two states, numeric mode or keypad mode. In numeric mode, the keypad buttons represent the numbers and symbols printed on the keys. In keypad mode, the keys may be defined to provide functions, in much the same way as they do in DCL. The keypad definitions are established by a startup file called SHADOW.KPD in the current default directory.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SHOW KEYPAD

The keyboard is not in keypad mode.

The keypad was not in keypad mode in this example.

SHOW OUTPUT

Displays the names of the current output files.

FORMAT	SHOW OUTPUT				
	<table><tr><td>Command Qualifiers</td><td>Defaults</td></tr><tr><td>None.</td><td>None.</td></tr></table>	Command Qualifiers	Defaults	None.	None.
Command Qualifiers	Defaults				
None.	None.				
restrictions	None.				
prompts	None.				
command parameters	None.				
DESCRIPTION	There are three possible output files produced by the shadow program. One is for plotting with a separate plotting program and has a filetype of .PLT. The second is a line-printer formatted output with a filetype of .PRT. The third is the input set listing echo, which may be redirected into a file. Its filetype is .LIS.				
COMMAND QUALIFIERS	None.				

EXAMPLES

```
SHADOW> SET OUTPUT EXAMPLE3 /PRINT
SHADOW> SHOW OUTPUT
Plotting file is: USER1:[RJM.NAS]EXAMPLE3.PLT;1
Printer file is: USER1:[RJM.NAS]EXAMPLE3.PRT;1
Input echo file: USER1:[RJM.NAS]EXAMPLE3.LIS;1
```

This example shows how a SET OUTPUT command creates the names shown for output files. See the SET OUTPUT command description for more details.

SHOW PATTERN_CUT

Displays the pattern-cut coordinate system transformation matrix.

FORMAT

SHOW PATTERN_CUT

Command Qualifiers
None.

Defaults
None.

restrictions

None.

prompts

None.

command parameters

None.

DESCRIPTION

The shadow map window is specified relative to the pattern-cut coordinate system. This system can be changed to facilitate easier specification of this window relative to the blocking object's coordinate system, that is, the reference coordinate system. For more information, see the SET PATTERN command on page 66.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SHOW PATT

```
*   The following rotations are used for ALL subsequent inputs:   *
*   VPC(1,1)= -0.70711  VPC(1,2)= -0.70711  VPC(1,3)=  0.00000   *
*   VPC(2,1)=  0.70711  VPC(2,2)= -0.70711  VPC(2,3)=  0.00000   *
*   VPC(3,1)=  0.00000  VPC(3,2)=  0.00000  VPC(3,3)=  1.00000   *
```

The pattern-cut coordinate system shown has been set up.

SHOW SCALE_FACTOR

Displays the uniform scale factor currently in effect.

FORMAT

SHOW SCALE_FACTOR

Command Qualifiers

None.

Defaults

None.

restrictions

None.

prompts

None.

command parameters

None.

DESCRIPTION

The SET SCALE_FACTOR command can set a uniform scale factor on subsequent antenna inputs. It allows an extra scaling on the inputs.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SHOW SCALE

The uniform scale factor is 1.00000000

The above scale factor is the default. It has not been changed with SET SCALE.

SHOW UNITS

Displays the current units in effect. Valid units are meters, feet, and inches.

FORMAT

SHOW UNITS

Command Qualifiers	Defaults
None.	None.

restrictions

None.

prompts

None.

command parameters

None.

DESCRIPTION

There are three different units in which antenna locations may be specified. This command displays the units currently in effect. The SET UNITS command changes the default units.

COMMAND QUALIFIERS

None.

EXAMPLES

SHADOW> SHOW UNITS

The current units are feet

SHOW WINDOW

Displays the current window parameters.

FORMAT	SHOW WINDOW				
	<table><tr><td>Command Qualifiers</td><td>Defaults</td></tr><tr><td>None.</td><td>None.</td></tr></table>	Command Qualifiers	Defaults	None.	None.
Command Qualifiers	Defaults				
None.	None.				
restrictions	None.				
prompts	None.				
command parameters	None.				
DESCRIPTION	The output can be windowed onto a smaller range of theta or phi, with any desired resolution. The parameters for this windowing are established by the SET WINDOW command.				
COMMAND QUALIFIERS	None.				

EXAMPLES

SHADOW> SHOW WIND

The current range of theta in degrees is 0.000000E+00 to 180.0000
with a resolution of 2.000000 degrees/pixel.
The current range of phi in degrees is 0.000000E+00 to 360.0000
with a resolution of 2.000000 degrees/pixel.

In this case, the window is set to its default range with a resolution of two degrees/pixel.

Chapter 6

Interpretation of the Output

The final product of the obscuration code, SHADOW, is a map of the projected shadow of a defined object onto the far zone sphere with its center at the antenna location. The map is composed of pixels with the size and range specified by the user. The obscuration code provides complete control over the parameters needed to define the map and provides a line printer output or a plottable file that can be used by an external plotting code. This chapter outlines the details of defining, obtaining, and interpreting the shadow map.

For this discussion, the far zone sphere can be viewed as being ironed out into a flat plane, that is, a Mercator's projection with the angle phi along the x axis and the angle theta along the y axis. Using the VF non-interactive command or the SET WINDOW interactive command, the user can choose starting angles, incremental step size which is the resolution of the map, and the total number of steps or pixels for both the theta and phi angles. This, of course, also dictates the stopping angles of the map which it computes. The default is for theta to vary from 0 to 180 degrees in steps of 2 degrees for a total of 91 pixels, and for phi to vary from 0 to 360 in steps of 2 for a total of 181 pixels. The interactive command SET WINDOW allows these parameters to be changed at any time during a session. It asks for the starting and stopping angle and the resolution which is the step size and it computes the number of pixels for each angle. These angles are defined with respect to the pattern coordinate system, which is specified by the first set of angles in the VF command or by the SET PATTERN command. The default is for the pattern coordinate system to be the same as the reference coordinate system.

As discussed in Chapter 2, the code computes the shadow by first projecting the objects border onto the far zone sphere and then filling in between the borders. A pixel is considered to be filled if the border at least passes through more than half the distance to the center of a pixel. It determines this by rounding the theta and phi angles defining the border to the nearest integer with respect to the resolution size of the pixel, which is the step size. This sometimes appears to produce a ragged border around the edges of the shadow if the border is very curved. Note that a straight edged plate projects a shadow that is curved in border. In addition, this is dependent on the coordinate system in which the shadow is viewed. Chapter 7 presents specific examples of these types of maps.

The shadow is represented by an ASCII character being placed in an array corresponding to the integerized theta and phi angles. A clear viewing point is left blank. The choice of the character that is placed in the pixel can be controlled by the user. The default is for an "X" to be used as a fill character. Interactively this can be changed using the SET FILL

command. Noninteractively, these are hard-wired into the source code.

For debugging purposes or so that the user can get a feel for which plates and cylinders are shadowing which regions of space a highlighting feature has been provided. The SET FILL/SEQUENTIAL command tags each plate and cylinder with its own unique fill character. The first plate starts with "A" and each succeeding plate is incremented up by one ASCII character. The first cylinder starts with "1" and each succeeding cylinder is incremented up by one ASCII character. Note that if there are a lot of plates and/or cylinders, the fill characters will eventually get into some of the more seldom used ASCII characters. Also note that in this mode of filling, the code superimposes the latest calculated shadow for a plate or cylinder on top of the shadow map. This means that the character in a pixel for a finished map will represent the last object that the code calculated a shadow for and not the object that is located closest to the observer.

In order to get around the ambiguous behavior of highlighting the plates and cylinders by order of processing rather than by location, the user can instead use the standard fill character for all plates and cylinders and highlight one particular specified object. The command SET FILL/PLATE = (number,character) or SET FILL/CYLINDER = (number,character) will highlight the chosen plate or cylinder against the regular fill character. The plate or cylinder options are mutually exclusive. It represents the shadow of the whole plate or cylinder that is tagged. A non interactive command has not been provided for these fill features. The user can change the fill characters and mode in the INIT subroutine.

The output that the user sees can come in three forms. The first type of output comes from an echo of the command set that is read from the input file on logical unit #5. The output is sent to logical unit #6, which is normally assigned to a default file type of .LIS on a VAX in the interactive mode. An ASCII file of the shadow map is written to logical unit #7, which is normally assigned to a default file type of .PRT on a VAX. A binary file of the shadow map that can be used to transfer information to another code to plot the map is sent to logical unit #10, which is normally assigned to a default file type of .PLT on a VAX.

In the interactive mode, the input set can be opened using the SET OUTPUT command. The output files can be opened and closed using the SET OUTPUT /ECHOING, /PRINTABLE, /PLOTTABLE commands, respectively. In the non-interactive mode, they can be controlled by using system commands, such as ASSIGN on the VAX. In the interactive mode, the output files should generally be set first, so that the code will have the desired information as to where to sent the echo back information. In addition, once the code is run and it is desired to see the results, it is possible to print or plot the results using the SPAWN ("\$") command. The files that are desired to be printed or plotted, however, must be closed first, that is, the SET OUTPUT command should be given again reassigning the files to another name, a null device, or the printing device. This will close the files and allow them to be accessed. Of course, it is important to remember to reopen them after the user is finished and wants to run more results. Presently, the echo, printable, and plottable map files will accumulate information until they are closed.

Generally, the code will be used to produce plottable files of the shadow maps with the printable file being used for debug purposes. Plotted maps are small and nicer to look at. Unfortunately, graphical routines are presently system dependent. A plotting code for a NCAR [5], has been provided, however, in Chapter 13. This is one example of how the

data of the shadow map can be plotted. Examples of both the printed and plotted maps are illustrated in the examples of Chapter 7. It should be noted that due to the limited amount of space across the width of a line printer, a printed map will be broken up into widths that will fit onto the width of the paper if it is too wide. The map will come out in as many strips as necessary to produce the whole map. Plotted maps should not have this problem since the individual pixels can be graphed very close together.

Chapter 7

Examples

The following examples are used to illustrate the various features of the SHADOW computer code. Each example is designed to show how a set of non-interactive and interactive commands can be put together to solve a problem. The beginner can use the examples in this chapter to learn more about the code. In addition, these examples can be used to ensure that the code is operating correctly on your system. These examples were run on a DEC VAX 11/780 computer using version 4 of the VMS operating system.

The shadow maps shown here are presented mostly with the line printer output, since this is generally the most transportable. Plotted output would normally be preferred in a design situation. A few examples of this type of output are also given.

7.1 Example 1: A Plate

The first example is a four-cornered plate centered at the origin and situated in the X-Y plane. The antenna is located on the positive Z axis. It was generated with the following input files and commands. The commands were:

```
$ RUN SHADOW
SHADOW> SET OUT PLAEX1/NO PLOT/PRINT
SHADOW> SET INP PLAEX
SHADOW> SET UNI METERS
SHADOW> SET WIND
          90, 180
          1.0
          0., 360
          5.
SHADOW> SET ANT
SHADOW> 0,0,8
SHADOW> SHADOW
SHADOW> EXIT
$
```

The input set defining the plate was the following:

```
CM: SIMPLE PLATE TEST SET
CE: RCS INPUT SET
UN:
1
PG: THE PLATE IS 400 SQUARE-METERS.
4,0
-10.0, +10.0, 0.0
-10.0, -10.0, 0.0
+10.0, -10.0, 0.0
+10.0, +10.0, 0.0
XQ:
EN:
```

The output this produced was the following:

ANTENNA (RCS) = (0.0000, 0.0000, 8.0000) IN METERS INPUT SET: USER1:[RJM.WAS.MAN]PLAEX.INP;6

PHI	THETA (DEGREES)									
	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00	180.00
0.00										
5.00										
10.00										
15.00										
20.00										
25.00										
30.00										
35.00										
40.00										
45.00										
50.00										
55.00										
60.00										
65.00										
70.00										
75.00										
80.00										
85.00										
90.00										
95.00										
100.00										
105.00										
110.00										
115.00										
120.00										
125.00										
130.00										
135.00										
140.00										
145.00										
150.00										
155.00										
160.00										
165.00										
170.00										
175.00										
180.00										
185.00										
190.00										
195.00										
200.00										
205.00										
210.00										
215.00										
220.00										
225.00										
230.00										
235.00										
240.00										
245.00										
250.00										
255.00										
260.00										
265.00										
270.00										
275.00										
280.00										
285.00										
290.00										
295.00										
300.00										
305.00										
310.00										
315.00										
320.00										
325.00										
330.00										
335.00										
340.00										
345.00										
350.00										
355.00										
360.00										

7.2 Example 2: A Different Plate

This example is another four-cornered plate, but this time the antenna is located at the origin, and the plate is centered along the positive Y axis and is normal to it.

The commands were:

```
$ RUN SHADOW
SHADOW> SET OUT PLAEX2/NO PLOT/PRINT
SHADOW> SET INP PLAEX2
SHADOW> SET UNI METERS
SHADOW> SET WIND
                0, 180
                2.0
                0., 180
                5.
SHADOW> SET ANT
                0,0,0
SHADOW> SHADOW
SHADOW> EXIT
$ EXIT
```

The input set defining the plate was the following:

```
CM: SIMPLE PLATE TEST SET
CE: RCS INPUT SET
UN:
1
PG: THE PLATE IS 400 SQUARE-METERS.
4 0
-10.0, 8, +10.0
-10.0, 8, -10.0
+10.0, 8, -10.0
+10.0, 8, +10.0
XQ:
EN:
```


7.3 Example 3: The First Plate Revisited

The current example is deceptive. Both the input geometry and the source location are identical with the first plate example, but the obscuration output is identical to the second example! A closer of the input sets reveals the the two examples are really the same geometry, but defined in different orientations with respect to the Reference Coordinate System. The third example takes advantage of this fact and uses the SET PATTERN.CUT command to reorient the coordinate system of the antenna. The result is that while the geometry is defined the same as the first example, the output resembles the second example. The commands to generate the example were:

```
$ RUN SHADOW
SHADOW> SET OUT PLAEX3/NO PLOT/PRINT
SHADOW> SET INP PLAEX
SHADOW> SET UNI METERS
SHADOW> SET WIND
      0,180
      2.0
      0.,180
      5.
SHADOW> SET ANT
      0,0,8
SHADOW> SET PATT
      90., +90., 90., 0.
SHADOW> SHADOW
SHADOW> EXIT
$ EXIT
```

The input set defining the plate was the same one used in example one. It is:

```
CM: SIMPLE PLATE TEST SET
CE: RCS INPUT SET
UN:
1
PG: THE PLATE IS 400 SQUARE-METERS.
4 0
-10.0, +10.0, 0.0
-10.0, -10.0, 0.0
+10.0, -10.0, 0.0
+10.0, +10.0, 0.0
XQ:
EH:
```

The output generated by the code was the following:

ANTENNA (RCS) = (0.0000, 0.0000, 8.0000) IN METERS INPUT SET: USER1:[RJM,NAS,MAN]PLAEX.INP;8

	THETA (DEGREES)									
PHI	0.00	20.00	40.00	60.00	80.00	100.00	120.00	140.00	160.00	180.00
	+	+	+	+	+	+	+	+	+	+
0.00										
5.00										
10.00										
15.00										
20.00										
25.00										
30.00										
35.00										
40.00										
45.00										
50.00										
55.00										
60.00										
65.00										
70.00										
75.00										
80.00										
85.00										
90.00										
95.00										
100.00										
105.00										
110.00										
115.00										
120.00										
125.00										
130.00										
135.00										
140.00										
145.00										
150.00										
155.00										
160.00										
165.00										
170.00										
175.00										
180.00										

7.4 Example 4: A Non-Interactive Version of Example 1

This example illustrates an input set for non-interactive use of the code. The main program has been changed to the non-interactive version and non-interactive subroutines were not linked into the code. The input set is the same as Example 1, except that the source and window have been define using the SG and VF commands, respectively. Note that these commands can also be used in the interactive mode also to hard wire the antenna location and window as a default case. The output is not shown here because it is identical to that of Example 1.

The input set defining the plate is the following:

```
CM: SIMPLE PLATE TEST SET
CE: RCS INPUT SET
UN:
1
PG: THE PLATE IS 400 SQUARE-METERS.
4 0
-10.0, +10.0, 0.0
-10.0, -10.0, 0.0
+10.0, -10.0, 0.0
+10.0, +10.0, 0.0
SG: THE SOURCE LOCATION
0.,0.,8.
0.,0.,90.,0.
-1,0.5,0.
1.,0.
VF: WINDOW SIZE
0.,0.,90.,0.
T,0.,2.,91
0.,2.,181
XQ:
EN:
```

7.5 Example 5: An Elliptic Cylinder

This example consists of one elliptic cylinder centered on the origin with its axis directed along the Y axis. Three different source locations are presented with this single example.

The commands were:

```
$ RUN SHADOW
SHADOW> SET OUT  CYLEX1/NO PLOT/PRINT
SHADOW> SET INP  CYLEX1
SHADOW> SET UNI  METERS
SHADOW> SET WIND
130,180
0.55555556
0.,360
5.
!
! An overhead view of the cylinder, which is centered on the origin,
! with radii of 1 and 1, with a length of 1 meter.
!
SHADOW> SET ANT
0,0,4
SHADOW> SHADOW
!
! A broadside look at the cylinder.
!
SHADOW> SET WIND
45,135
1.0
220.,310
1.25
SHADOW> SET ANT
0,4,0
SHADOW> SHADOW
!
! Now a look at the same geometry along the axis of the cylinder.
!
SHADOW> SET ANT
4,0,0
SHADOW> SET WIND
45,135
1.0
130.,220
1.25
SHADOW> SHADOW
SHADOW> EXIT
$ EXIT
```

The input set defining the plate was the following:

```
CM:  SIMPLE AIRCRAFT
CE:  RCS INPUT SET
UN:
```

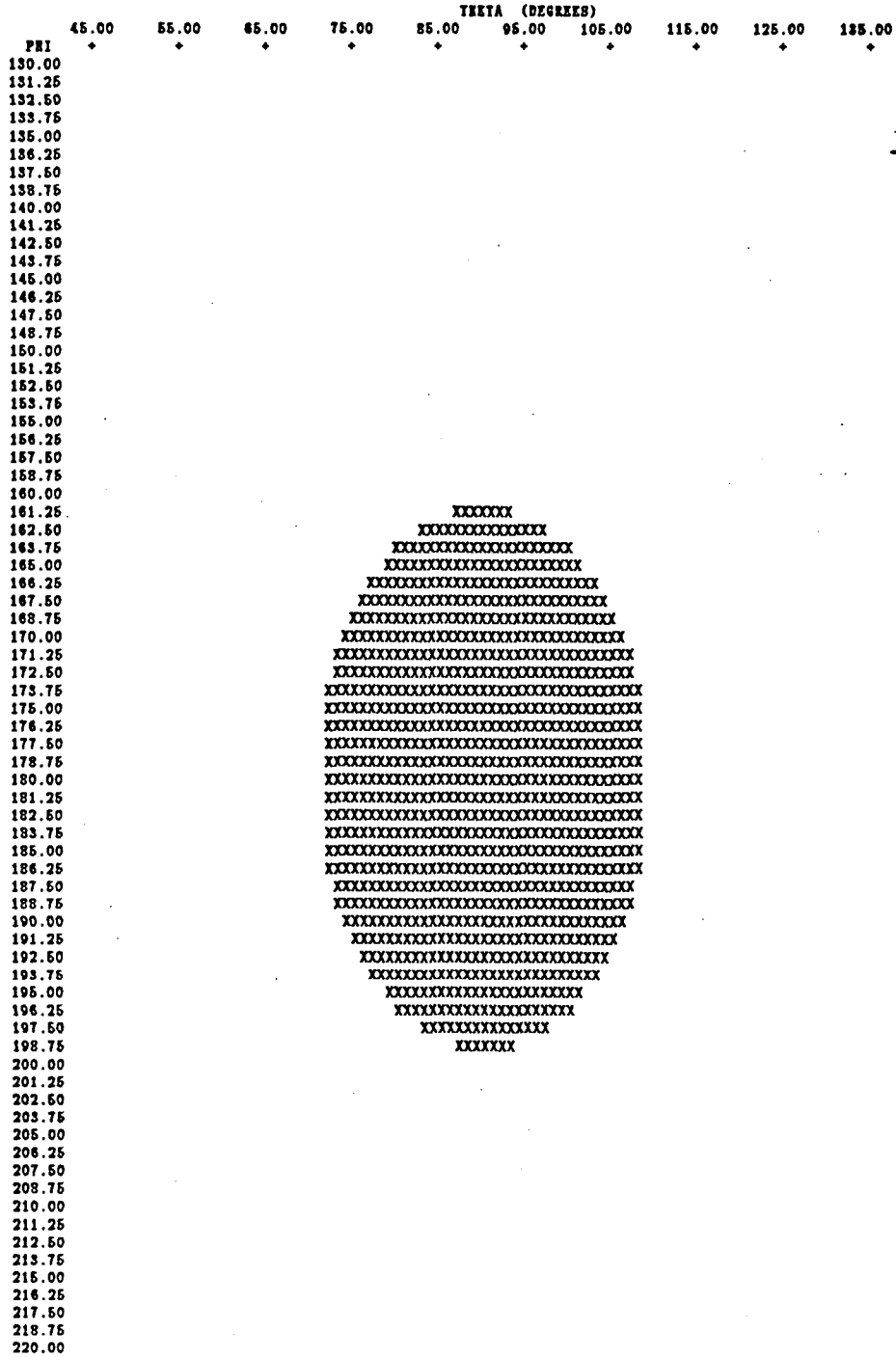
1
CC: FIRST CYLINDER
0.,0.,0.
90.,0.,0.,0.
2
1.,1., 1.
1.,1., -1.
XQ:
EN:

The output generated by the code was the following:

ANTENNA (RCS) = (0.0000, 0.0000, 4.0000) IN METERS INPUT EXT: USER1:[RJM.NAS.MAN]CYLEX1.INP:4

PHI	THETA (DEGREES)									
	130.00	135.56	141.11	146.67	152.22	157.78	163.33	168.89	174.44	180.00
0.00	+	+	+	+	+	+	+	+	+	+
5.00										
10.00										
15.00										
20.00										
25.00										
30.00										
35.00										
40.00										
45.00										
50.00										
55.00										
60.00										
65.00										
70.00										
75.00										
80.00										
85.00										
90.00										
95.00										
100.00										
105.00										
110.00										
115.00										
120.00										
125.00										
130.00										
135.00										
140.00										
145.00										
150.00										
155.00										
160.00										
165.00										
170.00										
175.00										
180.00										
185.00										
190.00										
195.00										
200.00										
205.00										
210.00										
215.00										
220.00										
225.00										
230.00										
235.00										
240.00										
245.00										
250.00										
255.00										
260.00										
265.00										
270.00										
275.00										
280.00										
285.00										
290.00										
295.00										
300.00										
305.00										
310.00										
315.00										
320.00										
325.00										
330.00										
335.00										
340.00										
345.00										
350.00										
355.00										
360.00										

ANTENNA (RCS) = (4.0000, 0.0000, 0.0000) IN METERS INPUT SET: USER1:[RJM.NAS.MAN]CYLEX1.INP:4



7.6 Example 6: Two Elliptic Cylinders

This example consists of two elliptic cylinders equidistant from the origin with axes coincident and directed along the Y axis. Three different source locations are presented with this single example.

The commands were:

```
$ RUN SHADOW
SHADOW> SET OUT CYLEX2/NO PLOT/PRINT
SHADOW> SET INP CYLEX2
SHADOW> SET UNI METERS
SHADOW> SET WIND
130,180
0.55555556
0.,360
5.
!
! An overhead view of the 2 cylinders with radii of 1 and 1,
! with a length of 1 meter each.
!
SHADOW> SET FILL /CYL=1
SHADOW> SET ANT
0,0,4
SHADOW> SHADOW
!
SHADOW> SET WIND
45,135
1.0
220.,310
1.25
SHADOW> SET ANT
0,4,0
SHADOW> SHOW FILL
SHADOW> SHADOW
!
SHADOW> SET ANT
4,0,0
SHADOW> SET WIND
45,135
1.0
130.,220
1.25
SHADOW> SHOW FILL
SHADOW> SHADOW
SHADOW> EXIT
$ EXIT
```

The input set defining the plate was the following:

```
CM: SIMPLE AIRCRAFT
CE: RCS INPUT SET
UN:
```

1

CC: FIRST CYLINDER

0.,-2.,0.

90.,0.,0.,0.

2

1.,1., 1.

1.,1., -1.

CC: SECOND CYLINDER

0.,+2.,0.

90.,0.,0.,0.

2

1.,1., 1.

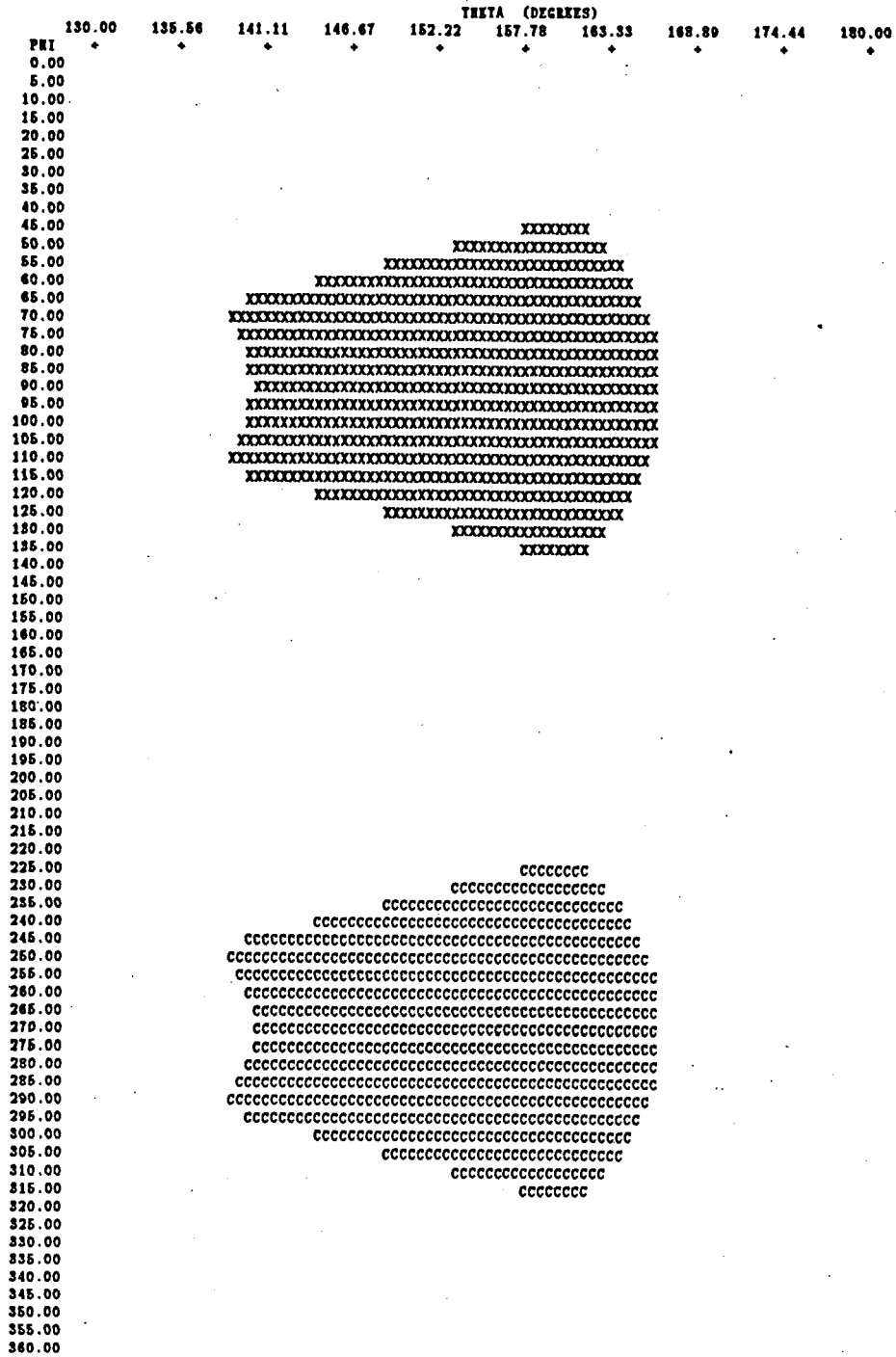
1.,1., -1.

XQ:

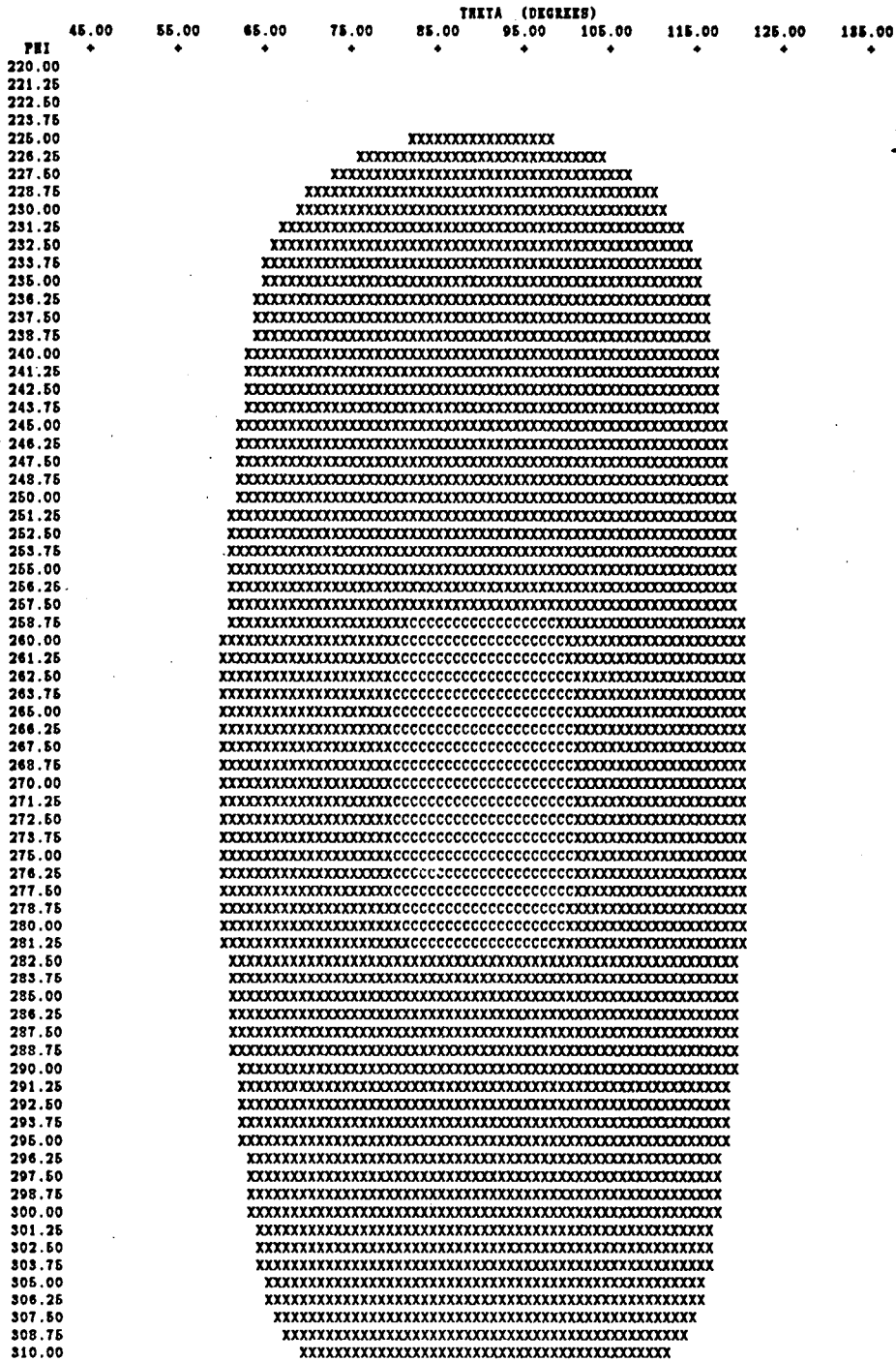
EN:

The output generated by the code was the following:

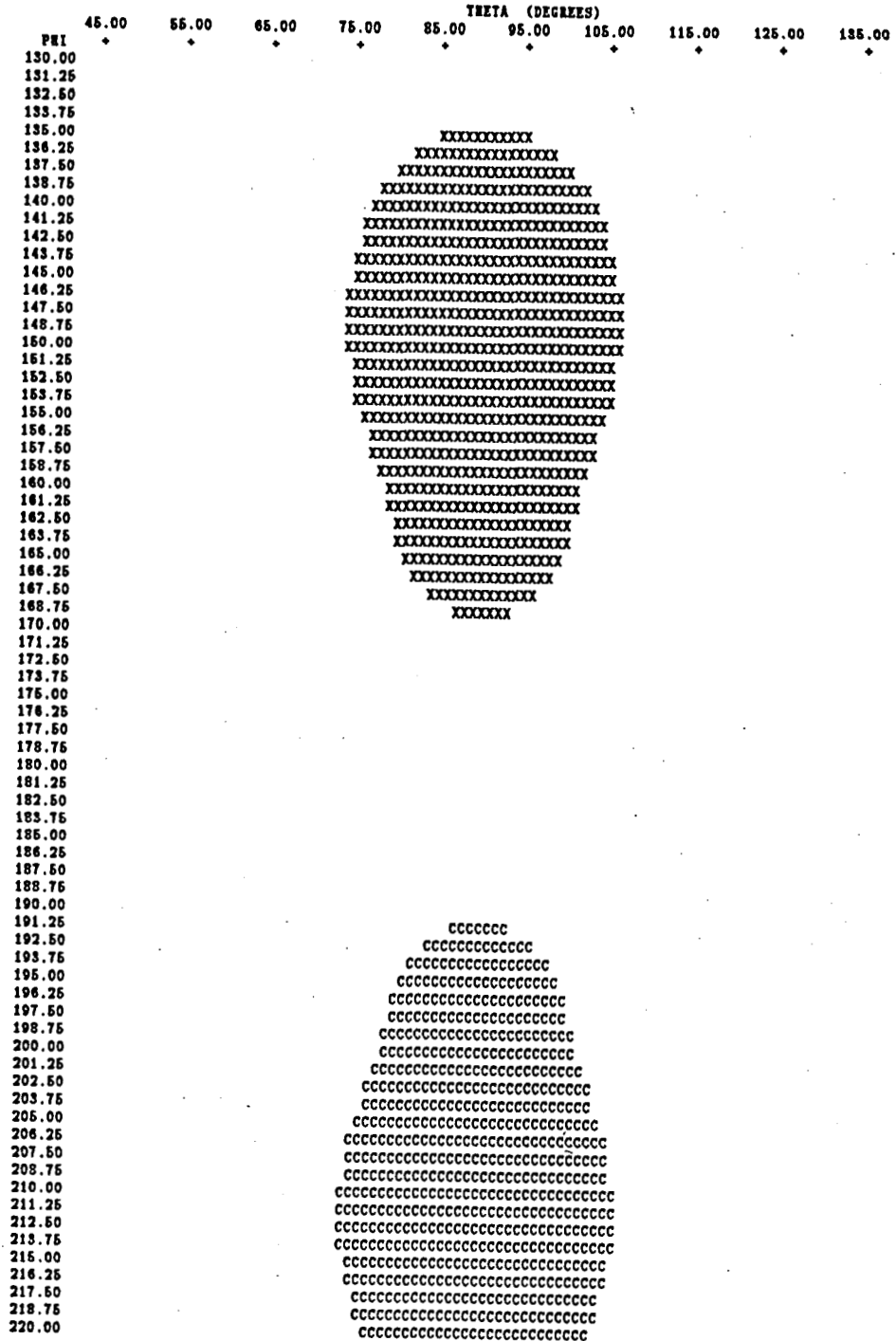
ANTENNA (RCS) = (0.0000, 0.0000, 4.0000) IN METERS INPUT SET: USER1:[LJM.NAS.MAN]CYLEX2.INP;2



ANTENNA (RCS) = (0.0000, 4.0000, 0.0000) IN METERS INPUT SKT: USER1:[BJN.NAS.MAN]CYLEX2.IMP;2



ANTENNA (RCS) = (4.0000, 0.0000, 0.0000) IN METERS INPUT SET: USER1:[RJM.NAS.MAN]CYLEX2.INP;2



7.7 Example 7: A Space Station Model

This example uses a space station, shown in Figure 7.1, that has been provided by NASA, Langley. The computer model is illustrated in Figure 7.2. It is an demonstrates how to use the windowing and highlighting commands (SET FILL) to effectively show obscuration.

The commands were:

```
$ RUN SHADOW
SHADOW> SET OUT AN5S1 /PRINT/NOECHO
SHADOW> SET INP AN5S1
SHADOW> SET UNI FEET
SHADOW> SET WIND
0.180
2.0
20.,290
2.5
!
! Display ONLY plate 6.
!
SHADOW> SET FILL " " /PLATE=6
SHADOW> SET ANT
25. 15. 256.5
SHADOW> SHADOW
!
! Now make plate 6 stand out from the crowd.
!
SHADOW> SET FILL "!" /PLATE=(6,$)
SHADOW> SET ANT
25. 15. 256.5
SHADOW> SHADOW
$ EXIT
```

The input set defining the plate was the following:

```
CM: *****CASE AN5S1*****
CM: *****OBSCURATION*****
CE:
LP:
F
UN: UNITS IN FEET
2
CM: UPPER BOOM
CE:
PG: BOTTOM
4 0
4.5 49.5 387.
4.5 -49.5 387.
-4.5 -49.5 387.
-4.5 49.5 387.
PG: +X SIDE
4 0
4.5 49.5 396.
```


4.5 -49.5 396.
4.5 -49.5 387.
4.5 49.5 387.
CM: UPPER KEEL
CE:
PG: -Y #1
4 0
4.5 -4.5 270.
4.5 -4.5 387.
-4.5 -4.5 387.
-4.5 -4.5 270.
PG: +Y #1
4 0
4.5 4.5 270.
-4.5 4.5 270.
-4.5 4.5 387.
4.5 4.5 387.
PG: +X SIDE
4 0
4.5 4.5 387.
4.5 -4.5 387.
4.5 -4.5 270.
4.5 4.5 270.
CM: LOWER KEEL & EXTENSION
CE:
PG: +X SIDE
12 0
4.5 22.5 0.
4.5 22.5 99.
4.5 4.5 99.
4.5 4.5 261.
4.5 -4.5 261.
4.5 -4.5 99.
4.5 -22.5 99.
4.5 -22.5 0.
4.5 -13.5 0.
4.5 -13.5 54.
4.5 13.5 54.
4.5 13.5 0.
PG: -Y #1
4 0
4.5 -22.5 0.
4.5 -22.5 99.
-4.5 -22.5 99.
-4.5 -22.5 0.
PG: -Y #2
4 0
4.5 -22.5 99.
4.5 -4.5 99.
-4.5 -4.5 99.
-4.5 -22.5 99.

PG: -Y #3
4 0
4.5 -4.5 99.
4.5 -4.5 261.
-4.5 -4.5 261.
-4.5 -4.5 99.
PG: +Y #1
4 0
4.5 22.5 0.
-4.5 22.5 0.
-4.5 22.5 99.
4.5 22.5 99.
PG: +Y #2
4 0
4.5 22.5 99.
-4.5 22.5 99.
-4.5 4.5 99.
4.5 4.5 99.
PG: +Y #3
4 0
4.5 4.5 99.
-4.5 4.5 99.
-4.5 4.5 261.
4.5 4.5 261.
CM: NON-ROTATING SECTION
CM: OF SOLAR PANEL BOOM
CE:
PG: BOTTOM
4 0
4.5 49.5 261.
4.5 -49.5 261.
-4.5 -49.5 261.
-4.5 49.5 261.
PG: +X SIDE
4 0
4.5 49.5 270.
4.5 -49.5 270.
4.5 -49.5 261.
4.5 49.5 261.
CM: ROTATING SECTION OF
CM: SOLAR PANEL BOOM
CE:
RT: -Y SIDE
0. -54. 265.5
0. 0. 90. 0.
PG: TOP
4 0
4.5 4.5 4.5
-4.5 4.5 4.5
-4.5 -76.5 4.5
4.5 -76.5 4.5

PG: BOTTOM

4 0

4.5 4.5 -4.5
4.5 -76.5 -4.5
-4.5 -76.5 -4.5
-4.5 4.5 -4.5

PG: +X SIDE

4 0

4.5 4.5 4.5
4.5 -76.5 4.5
4.5 -76.5 -4.5
4.5 4.5 -4.5

PG: -X SIDE

4 0

-4.5 4.5 4.5
-4.5 4.5 -4.5
-4.5 -76.5 -4.5
-4.5 -76.5 4.5

CM: UPPER OUTBOARD SOLAR PANEL

CE:

RT: -Y OUTBOARD

0. -132. 265.5

0. 0. 90. -52.

PG: -X 82X33

4 0

-1. 16.5 89.
-1. 16.5 7.
-1. -16.5 7.
-1. -16.5 89.

PG: UPPER 33

4 0

1. 16.5 89.
-1. 16.5 89.
-1. -16.5 89.
1. -16.5 89.

PG: LOWER 33

4 0

1. 16.5 7.
1. -16.5 7.
-1. -16.5 7.
-1. 16.5 7.

PG: INSIDE 82

4 0

1. 16.5 89.
1. 16.5 7.
-1. 16.5 7.
-1. 16.5 89.

CM: LOWER OUTBOARD SOLAR PANEL

CE:

PG: -X 82X33

4 0

-1. 16.5 -89.
-1. -16.5 -89.
-1. -16.5 -7.
-1. 16.5 -7.

PG: LOWER 33

4 0

1. 16.5 -89.
1. -16.5 -89.
-1. -16.5 -89.
-1. 16.5 -89.

PG: UPPER 33

4 0

1. 16.5 -7.
-1. 16.5 -7.
-1. -16.5 -7.
1. -16.5 -7.

PG: INSIDE 82

4 0

1. 16.5 -89.
-1. 16.5 -89.
-1. 16.5 -7.
1. 16.5 -7.

CM: UPPER INBOARD SOLAR PANEL

CE:

RT: -Y INBOARD

O. -78. 265.5

O. O. 90. -52.

PG: -X 82X33

4 0

-1. 16.5 89.
-1. 16.5 7.
-1. -16.5 7.
-1. -16.5 89.

PG: UPPER 33

4 0

1. 16.5 89.
-1. 16.5 89.
-1. -16.5 89.
1. -16.5 89.

PG: LOWER 33

4 0

1. 16.5 7.
1. -16.5 7.
-1. -16.5 7.
-1. 16.5 7.

PG: 82 INSIDE

4 0

1. 16.5 89.
1. 16.5 7.
-1. 16.5 7.
-1. 16.5 89.

PG: 82 OUTSIDE
4 0
1. -16.5 89.
-1. -16.5 89.
-1. -16.5 7.
1. -16.5 7.
CM: LOWER INBOARD SOLAR PANEL
CE:
PG: -X 82X33
4 0
-1. 16.5 -89.
-1. -16.5 -89.
-1. -16.5 -7.
-1. 16.5 -7.
PG: LOWER 33
4 0
1. 16.5 -89.
1. -16.5 -89.
-1. -16.5 -89.
-1. 16.5 -89.
PG: UPPER 33
4 0
1. 16.5 -7.
-1. 16.5 -7.
-1. -16.5 -7.
1. -16.5 -7.
PG: 82 INSIDE
4 0
1. 16.5 -89.
-1. 16.5 -89.
-1. 16.5 -7.
1. 16.5 -7.
PG: 82 OUTSIDE
4 0
1. -16.5 -89.
1. -16.5 -7.
-1. -16.5 -7.
-1. -16.5 -89.
CM: ROTATING SECTION OF
CM: SOLAR PANEL BOOM
CE:
RT: +Y SIDE
O. 54. 265.5
O. O. 90. O.
PG: TOP
4 0
4.5 -4.5 4.5
4.5 76.5 4.5
-4.5 76.5 4.5
-4.5 -4.5 4.5
PG: BOTTOM

4 0
4.5 -4.5 -4.5
-4.5 -4.5 -4.5
-4.5 76.5 -4.5
4.5 76.5 -4.5
PG: +X SIDE
4 0
4.5 -4.5 4.5
4.5 -4.5 -4.5
4.5 76.5 -4.5
4.5 76.5 4.5
PG: -X SIDE
4 0
-4.5 -4.5 4.5
-4.5 76.5 4.5
-4.5 76.5 -4.5
-4.5 -4.5 -4.5
CM: UPPER OUTBOARD SOLAR PANEL
CE:
RT: +Y OUTBOARD
0. 132. 265.5
0. 0. 90. -52.
PG: -X 82X33
4 0
-1. 16.5 89.
-1. 16.5 7.
-1. -16.5 7.
-1. -16.5 89.
PG: UPPER 33
4 0
1. 16.5 89.
-1. 16.5 89.
-1. -16.5 89.
1. -16.5 89.
PG: LOWER 33
4 0
1. 16.5 7.
1. -16.5 7.
-1. -16.5 7.
-1. 16.5 7.
PG: INSIDE 82
4 0
1. -16.5 89.
-1. -16.5 89.
-1. -16.5 7.
1. -16.5 7.
CM: LOWER OUTBOARD SOLAR PANEL
CE:
PG: -X 82X33
4 0
-1. 16.5 -89.

-1. -16.5 -89.
-1. -16.5 -7.
-1. 16.5 -7.
PG: LOWER 33
4 0
1. 16.5 -89.
1. -16.5 -89.
-1. -16.5 -89.
-1. 16.5 -89.
PG: UPPER 33
4 0
1. 16.5 -7.
-1. 16.5 -7.
-1. -16.5 -7.
1. -16.5 -7.
PG: INSIDE 82
4 0
1. -16.5 -89.
1. -16.5 -7.
-1. -16.5 -7.
-1. -16.5 -89.
CM: UPPER INBOARD SOLAR PANEL
CE:
RT: +Y INBOARD
O. 78. 265.5
O. O. 90. -52.
PG: -X 82X33
4 0
-1. 16.5 89.
-1. 16.5 7.
-1. -16.5 7.
-1. -16.5 89.
PG: UPPER 33
4 0
1. 16.5 89.
-1. 16.5 89.
-1. -16.5 89.
1. -16.5 89.
PG: LOWER 33
4 0
1. 16.5 7.
1. -16.5 7.
-1. -16.5 7.
-1. 16.5 7.
PG: 82 OUTSIDE
4 0
1. 16.5 89.
1. 16.5 7.
-1. 16.5 7.
-1. 16.5 89.
PG: 82 INSIDE

```

4 0
1. -16.5 89.
-1. -16.5 89.
-1. -16.5 7.
1. -16.5 7.
CM: LOWER INBOARD SOLAR PANEL
CE:
PG: -X 82X33
4 0
-1. 16.5 -89.
-1. -16.5 -89.
-1. -16.5 -7.
-1. 16.5 -7.
PG: LOWER 33
4 0
1. 16.5 -89.
1. -16.5 -89.
-1. -16.5 -89.
-1. 16.5 -89.
PG: UPPER 33
4 0
1. 16.5 -7.
-1. 16.5 -7.
-1. -16.5 -7.
1. -16.5 -7.
PG: 82 OUTSIDE
4 0
1. 16.5 -89.
-1. 16.5 -89.
-1. 16.5 -7.
1. 16.5 -7.
PG: 82 INSIDE
4 0
1. -16.5 -89.
1. -16.5 -7.
-1. -16.5 -7.
-1. -16.5 -89.
PP:
T
T 8.186 4.87
180. -180. -30.
-40 40. 4.
XQ: EXECUTE CODE
EN: END CODE

```

The output generated by the code was the following:

```

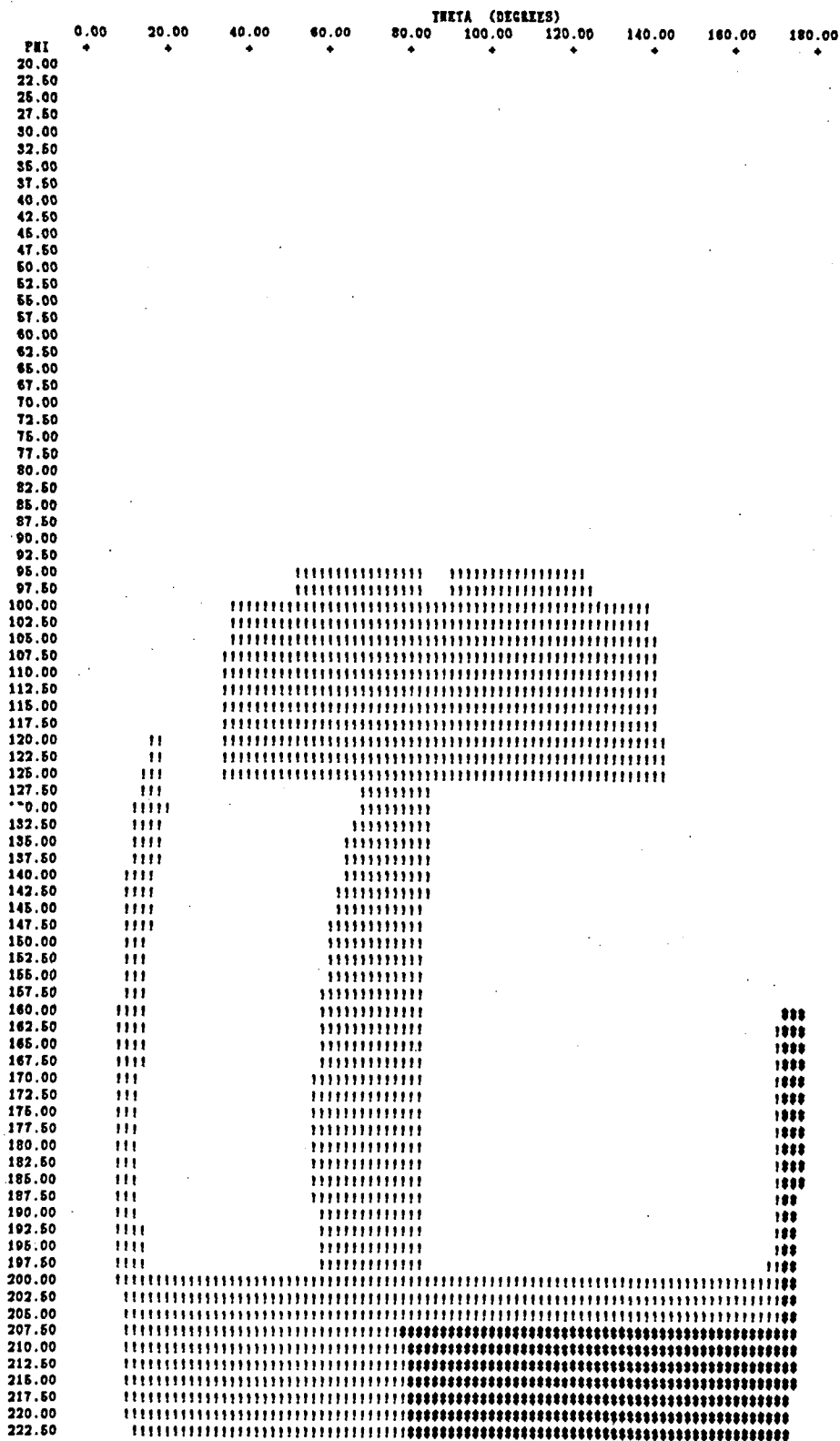
ANTENNA (RCS) = ( 7.6200, 4.6720, 78.1812 ) IN METERS   INPUT SET: USER1:[RJM.NAS.MAN]ANGS1.INP:1

```

PHI	THETA (DEGREES)									
	0.00	20.00	40.00	60.00	80.00	100.00	120.00	140.00	160.00	180.00
20.00	+	+	+	+	+	+	+	+	+	+
22.50										
25.00										
27.50										

247.50
250.00
252.50
255.00
257.50
260.00
262.50
265.00
267.50
270.00
272.50
275.00
277.50
280.00
282.50
285.00
287.50
290.00

ANTENNA (RCS) = (7.6200, 4.5720, 78.1812) IN METERS INPUT SKI: USER1:[LJM.NAS.MAN]AN651.INP:1



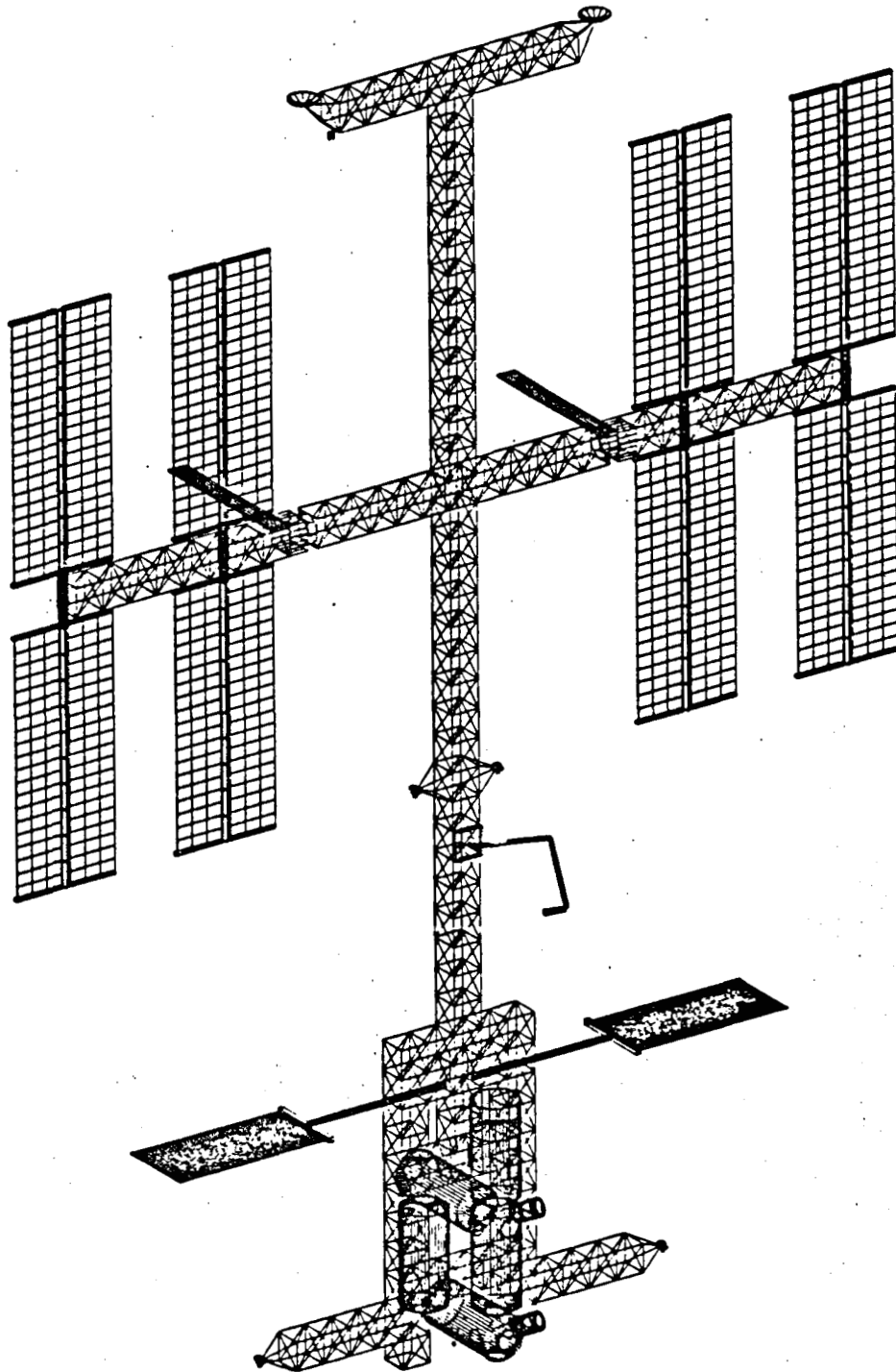
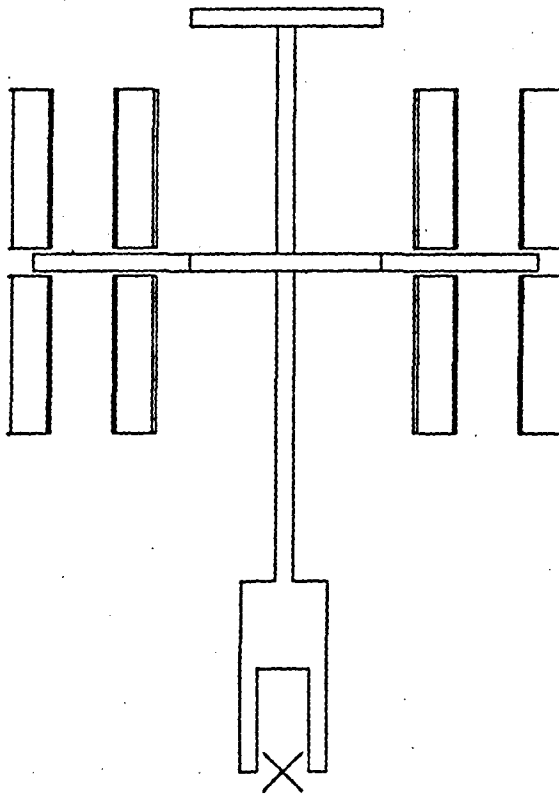


Figure 7.1: Illustration of the Space Station



TOP



FRONT



SIDE

Figure 7.2: Three-axis view of the Space Station as modeled by the input set.

7.8 Example 8: Another Look at the Space Station

This example presents a full view of the space station in the previous, except that the output is generated with the NCAR graphics interface. The non-interactive input is the same. The standard fill character procedure is used and a complete window is displayed with two degree resolution in theta and phi. The NCAR plot has been obtained using the plotting code in Chapter 13. The shadow map produced is shown in Figure 7.3

SHADØW TEST FØR CASE ANSS1
ANTENNA LØCATED AT 25.0, 15.0, 256.5
ANTENNA ØRIENTATIØN 0.0, 0.0, 90.0, 0.0
SØLAR PANELS RØTATED 0.0, -52.0
THERMAL RADIATØRS RØTATED 0.0

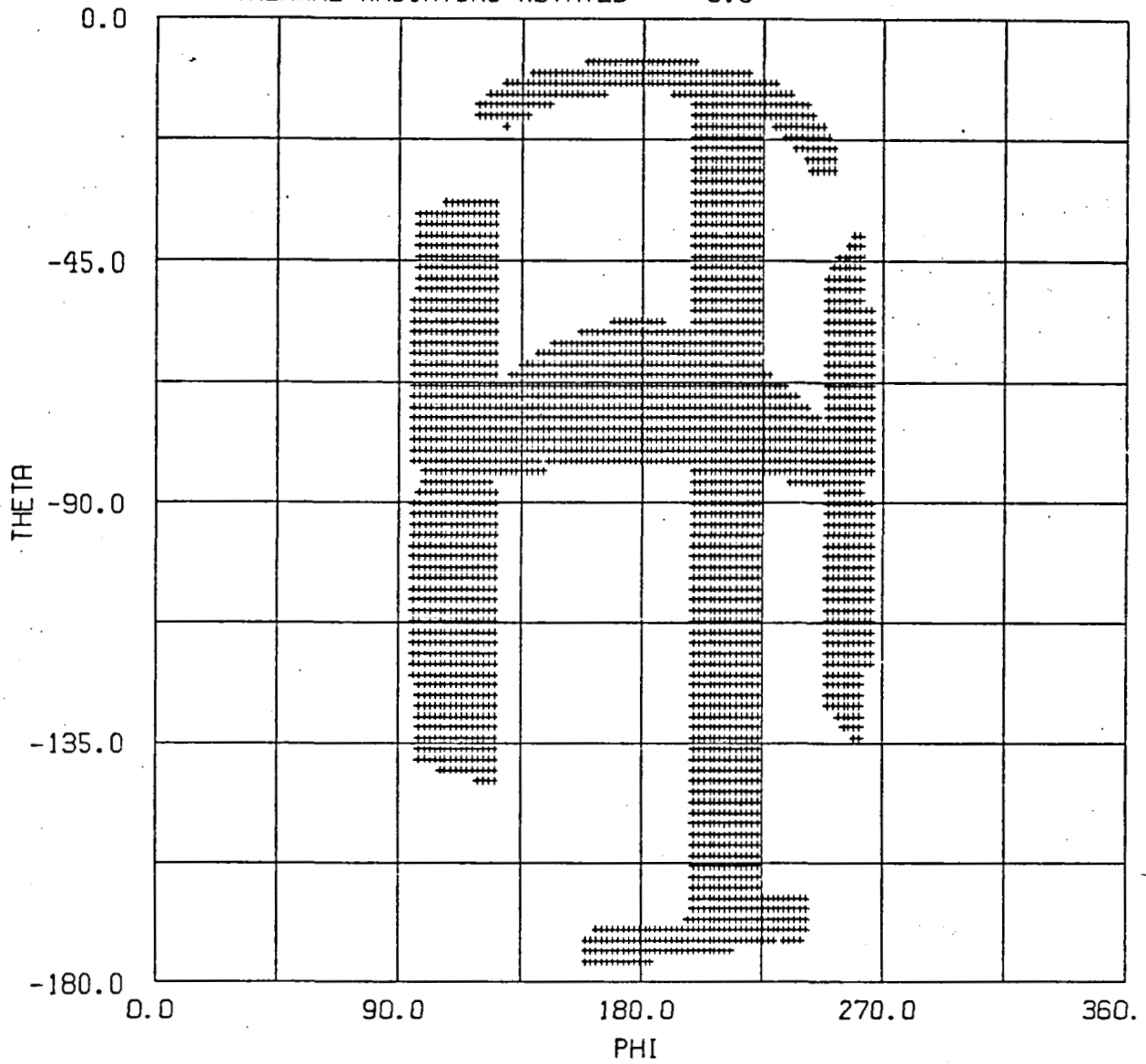


Figure 7.3: NCAR plot showing the shadow map of the space station model.

References

- [1] R. J. Marhefka and W. D. Burnside, "Numerical Electromagnetic Code - Basic Scattering Code, NEC-BSC (Version 2), Part I: User's Manual," Technical Report 712242-14, December 1982, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract No. N00123-79-C-1469 for Naval Regional Contracting Office.
- [2] R. J. Marhefka, "Numerical Electromagnetic Code - Basic Scattering Code, NEC-BSC (Version 2), Part II: Code Manual," Technical Report 712242-15, December 1982, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract No. N00123-79-C-1469 for Naval Regional Contracting Office.
- [3] H. H. Chung and W. D. Burnside, "General 3D Airborne Antenna Radiation Pattern Code User's Manual," Technical Report 711679-10, July 1982, The Ohio State University ElectroScience Laboratory, Department of Electrical Engineering; prepared under Contract No. F30602-79-C-0068 for Air Force Systems Command.
- [4] R. G. Kouyoumjian and P. H. Pathak, "A Uniform Geometrical Theory of Diffraction for an Edge in a Perfectly Conducting Surface," Proc. IEEE, Vol. 62, pp. 1448-1461, November 1974.
- [5] G. R. McArthur, editor, "The System Plot Package," NCAR Technical Note - 162+IA, January 1981, National Center for Atmospheric Research, Boulder, Colorado.

Part II
Code Manual

Chapter 8

Introduction

The obscuration code SHADOW, is designed to produce a projected shadow map onto the far zone radiation sphere of an antenna in a complex environment. The map is efficiently calculated by directly tracing the outer boundaries of the multisided flat plates and composite cone frustum cylinders onto theta - phi space and then filling between the boundaries along raster lines. The code has been developed to be interactively run on a DEC VAX computer. It can, also, be run non-interactively on any other computer by simply substituting the small main program and leaving out the interactive subroutines.

Part I of this manual is a user's guide which treats the code from the users standpoint without much particular details about the coding. Part II, given here, is intended to give some details about the internal workings of the code. It gives more specific information about the coding itself. It is of importance primarily for people implementing the code on a new system, for debugging errors, or for making changes in how the code operates. An overview of how the code is organized is given in Chapter 9. A listing of the code is given in Chapter 10. It is broken up into three parts for the non-interactive, FORTRAN 77 subroutines and into the interactive VAX dependent subroutines. The implementation of the code on a VAX is given in Chapter 11 and a brief description of implementing the code on a non-VAX computer is given in Chapter 12. A listing of an NCAR plotting code for the shadow map is given in Chapter 13.

Chapter 9

Code Organization

The obscuration code SHADOW is designed to produce a projected shadow map onto the far zone radiation sphere of an antenna in a complex environment. The map is efficiently calculated by directly tracing the outer boundaries of the multisided flat plates and composite cone frustum cylinders onto theta - phi space and then filling between the boundaries along raster lines.

The code has been developed with efficiency and ease of use as primary considerations. Often with other similar codes the engineer is not part of a tight interactive design loop. In order to facilitate this capability, while maintaining necessary transportability, the code has been split into two versions so that it can be run in two different modes, interactively or non-interactively depending on the computer being used. In both versions the flow of program control is basically the same. The main program either accepts interactive commands from the terminal and acts on those commands, or reads a different set of non-interactive commands from the input file and processes those. In both cases, the main program loops on input commands and calls appropriate subroutines for the creation and output of the shadow map.

The map creation is broken down into separate phases for each class of geometry being processed. Plates and elliptic cylinders are the two phases currently implemented. Each processing phase works by projecting each member of each class of geometry onto the far-zone sphere. The code implements the shadow map by mapping the far zone sphere in theta-phi space into a rectangular character array. The size of the array and hence the angular resolution of the resulting map is determined by the user at run time. After a member is projected, the far-zone grid (array) is processed in a raster-scan fashion to implement an area fill for the member. In this way every geometric entity is processed and included in the array. After all items of all classes have been processed, the output routines format and display/dump the resulting map. The main program then readies itself to execute yet another command or commands.

The source code is also organized into two groups of files depending upon the desired mode of operation. The code is organized this way so that minimum source modification is necessary in order to run in either interactive (in the case of a VAX computer) or non-interactive modes. The chapter on Non-VAX implementation describes the conversion of the source to non-interactive mode in detail.

Since the map computation and display routines are identical for both modes of operation, the transportability of generated results depends on the numerical behavior of the

target machine and not on implementational differences between the interactive and non-interactive versions.

Chapter 10

Listings of the Code

This chapter describes the operation of the routines and functions used by the program. Each listing is presented in alphabetical order and is preceded where appropriate by a short explanation of methods used.

10.1 VAX/VMS Subroutines

The following routines are for the interactive implementations of the code. They are used in conjunction with the routines in this chapter that are common to both versions.

MAIN PROGRAM

This is the main routine for the interactive versions of the program. It calls a one-time initialization routine and then executes commands until finished. There is another slightly different main program for the non-interactive code.

```
0001      PROGRAM SHADOW
0002      C!!!+
0003      C!!! This program was written at the ohio state university
0004      C!!! electroscience laboratory. any problems or comments
0005      C!!! can be referred to:
0006      C!!!
0007      C!!!      LASZLO A. TAKACS OR RONALD J. MARHEFKA
0008      C!!!      ELECTROSCIENCE LABORATORY
0009      C!!!      1320 KINNEAR RD.
0010      C!!!      COLUMBUS, OHIO 43212
0011      C!!!      PHONE: (614) 422-5752 OR 422-5848
0012      C!!!
0013      C!!! This program provides a printer output of the geometrical
0014      C!!! shadow boundries of a structure of plates and cylinders input
0015      C!!! as valid input sets to the numerical code.
0016      C!!!
0017      C!!! This program was written 15-JUN-1984.
0018      C!!! The latest modification occurred 18-DEC-1985.
0019      C!!!--
0020      C!!!
0021      C!!! Beginning of the main routine.
```



```
0022 C111 Initialize any SHADOW data structures.
0023 C111
0024 CALL INIT
0025 C111
0026 C111 Call the interactive terminal interface. This routine calls all
0027 C111 other subroutines.
0028 C111
0029 CALL INTRAC
0030 C111
0031 C111 Finished.
0032 C111
0033 END
```

SUBROUTINE INIT

```
0001
0002 C-----
0003     SUBROUTINE INIT
0004     INCLUDE 'SHACOM.FOR'
0170 C!!!
0171 C!!! This subroutine initializes the main routine.
0172 C!!! It is meant to be called once, at the start of the program.
0173 C!!!
0174 C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
0175 C!!!
0176 C!!! NOTICE:
0177 C!!! This routine performs actions which do not apply to the
0178 C!!! non-interactive mode of operation. In particular, the variables
0179 C!!! which are initialized here may be reinitialized elsewhere in both
0180 C!!! interactive and non-interactive versions. Altering these
0181 C!!! parameters may or may not achieve the expected results.
0182 C!!!
0183 C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
0184 C!!!
0185 C!!! Initialize variables to their default values.
0186 C!!!
0187 C!!! The lower/higher theta end of the range.
0188 C!!!
0189         THET1 =      0.0      * RPD
0190         THET2 =     180.0      * RPD
0191 C!!!
0192 C!!! The lower/higher phi end of the range.
0193 C!!!
0194         PH1   =      0.0      * RPD
0195         PH2   =     360.0      * RPD
0196 C!!!
0197 C!!! The desired theta/phi resolution in units of radians/pixel.
0198 C!!!
0199         RESTH =      2.      * RPD
0200         RESPH =      2.      * RPD
0201 C$$$
0202 C$$$ Rotate translate default data RT:
0203 C$$$
0204         THZP  =  0.
0205         PHZP  =  0.
0206         THXP  =  90.
0207         PHXP  =  0.
0208
0209         TRS( 1 ) = 0.
0210         TRS( 2 ) = 0.
0211         TRS( 3 ) = 0.
0212
0213         VRS( 1, 1 ) = 1.
0214         VRS( 1, 2 ) = 0.
0215         VRS( 1, 3 ) = 0.
0216
0217         VRS( 2, 1 ) = 0.
0218         VRS( 2, 2 ) = 1.
0219         VRS( 2, 3 ) = 0.
0220
0221         VRS( 3, 1 ) = 0.
0222         VRS( 3, 2 ) = 0.
0223         VRS( 3, 3 ) = 1.
0224 C$$$
0225 C$$$ Units default data UN:, UF:
```

```

0226 C$$$
0227 IUNIT = 1
0228 UNIFF = 1.0
0229 UNITH = UNIT( IUNIT )
0230 UNIIS = UNITH * UNIFF
0231 C$$$
0232 C$$$ Pattern cut orientation data VF:
0233 C$$$
0234 VPC( 1, 1 ) = 1.
0235 VPC( 1, 2 ) = 0.
0236 VPC( 1, 3 ) = 0.
0237
0238 VPC( 2, 1 ) = 0.
0239 VPC( 2, 2 ) = 1.
0240 VPC( 2, 3 ) = 0.
0241
0242 VPC( 3, 1 ) = 0.
0243 VPC( 3, 2 ) = 0.
0244 VPC( 3, 3 ) = 1.
0245 C!!!
0246 C!!! Open some standard input/output files for the VMS support routines.
0247 C!!! Units 5 and 6 are reserved for input set reading and echoing by the
0248 C!!! input set processor. NOTE: This is operating system dependent
0249 C!!! stuff. This is the natural place for it since it is initialized
0250 C!!! at the start.
0251 C!!!
0252 OPEN( UNIT=1,FILE='SYS$INPUT',TYPE='OLD' )
0253 OPEN( UNIT=2,FILE='SYS$OUTPUT',TYPE='UNKNOWN' )
0254 C!!!
0255 C!!! End of program initialization.
0256 C!!!
0257 RETURN
0258 END

```

SUBROUTINE INTRAC

This is the interactive commands subroutine called by the main routine. It fields commands typed by the user and executes the appropriate service routines. Also listed here are two I/O function subprograms which are indirectly invoked by INTRAC. They are called GET_INPUT and PUT_OUTPUT.

```

0001 C-----
0002 SUBROUTINE INTRAC
0003 |
0004 |++
0005 | FACILITY: INTERACTIVE TERMINAL COMMAND INTERFACE
0006 |
0007 | ABSTRACT:
0008 |
0009 | This procedure prompts a terminal for input and parses/dispatches
0010 | through CLI$ routines.
0011 |
0012 | ENVIRONMENT: VAX/VMS Version 4.x
0013 |
0014 | AUTHOR: Laszlo Takacs CREATION DATE: 20-AUG-1985
0015 |
0016 | MODIFIED BY:
0017 |
0018 | 1-001 - Original, LAT 20-AUG-1985
0019 |--
0020 IMPLICIT NONE
0021 INCLUDE '($RMSDEF)'
0463 INCLUDE '($SMGDEF)'
0774 INCLUDE 'SHACOM.FOR'
0940 EXTERNAL
0941 + COMMAND_TABLES, ! User-defined com
0942 + GET_INPUT ! The I/O routine at the b
0943 |
0944 INTEGER*4
0945 + STS,
0946 + READ_STIS,
0947 + CLI$PRESENT,
0948 + CLI$DISPATCH,
0949 + CLI$DCL_PARSE,
0950 + CLI$GET_VALUE,
0951 + SMG$LOAD_KEY_DEFS,
0952 + SMG$CREATE_KEY_TABLE,
0953 + SMG$DELETE_VIRTUAL_KEYBOARD,
0954 + SMG$CREATE_VIRTUAL_KEYBOARD
0955 |
0956 | Make a key definition table.
0957 |
0958 STS = SMG$CREATE_KEY_TABLE( KEYTBL )
0959 IF (.NOT. STS) CALL LIB$SIGNAL( %VAL(STS) )
0960 |
0961 | Load the definitions from the key definition file. Ignore "file not f
0962 |
0963 STS = SMG$LOAD_KEY_DEFS( KEYTBL, 'SHADOW.KPD' )
0964 IF ((.NOT. STS) .AND. (STS .NE. RMS$_FNF))
0965 + CALL LIB$SIGNAL( %VAL(STS) )
0966 |
0967 | Get a handle on SYS$INPUT.
0968 |
0969 READ_STIS = SMG$CREATE_VIRTUAL_KEYBOARD( KBDID )

```

```

0970 |
0971 | The main processing loop. Keep reading input until the user types EOF
0972 |
0973 |     DO WHILE ( READ_STS .NE. RMS$EOF )
0974 |
0975 | Read from input and parse the command.
0976 |
0977 |     READ_STS = CLI$DCL_PARSE(
0978 |         +
0979 |             COMMAND_TABLES,
0980 |             GET_INPUT,
0981 |             GET_INPUT,
0982 |             'SHADOW>' )
0983 |
0984 | If the command parse was successful, execute the command-routine.
0985 |
0986 |     IF ( .NOT. (.NOT. READ_STS) ) CALL CLI$DISPATCH()
0987 |
0988 |     END DO
0989 |
0990 | Get rid of the virtual keyboard.
0991 |
0992 |     STS = SMG$DELETE_VIRTUAL_KEYBOARD( KBDID )
0993 |     IF ( .NOT. STS ) CALL LIB$SIGNAL( %VAL(STS) )
0994 |
0995 | Return
0996 |
0997 |     RETURN
0998 |     END
0999 |
-----
0001 C-----
0002 |     INTEGER*4      FUNCTION  GET_INPUT( COMMAND, PROMPT, LENGTH )
0003 C!!!
0004 C!!! This routine does all the reading for the terminal interface.
0005 C!!! It has the same calling format as LIB$GET_INPUT except that optiona
0006 C!!! parameters may not be omitted.
0007 C!!!
0008 |     INCLUDE '($RMSDEF)'
0009 |     INCLUDE 'SHACOM.FOR'
0010 |     EXTERNAL
0011 |         + SMG$EOF
0012 |         CHARACTER*(*)
0013 |         +
0014 |         COMMAND,
0015 |         PROMPT
0016 |         +
0017 |         INTEGER
0018 |         +
0019 |         LENGTH*2,
0020 |         SMG$READ_COMPOSED_LINE*4
0021 |
0022 | Read a (composed) line and return the status to CLI$ stuff.
0023 |
0024 |     GET_INPUT = SMG$READ_COMPOSED_LINE (
0025 |         +
0026 |         KBDID,
0027 |         KEYTAB,
0028 |         COMMAND,
0029 |         PROMPT,
0030 |         LENGTH )
0031 |
0032 |     IF ( GET_INPUT .EQ. %LOC( SMG$EOF ) ) GET_INPUT = RMS$EOF
0033 |
0034 |     RETURN
0035 |     END
0036 |
-----
0001 C-----
0002 |     INTEGER*4      FUNCTION  PUT_OUTPUT ( STRING )
0003 C!!!
0004 C!!! This routine does all the writing for the terminal interface.
0005 C!!! It has the same calling format as LIB$PUT_OUTPUT.
0006 C!!!

```

```

0007      INCLUDE      'SHACOM.FOR'
0173      CHARACTER*(*)
0174      +                STRING
0175      INTEGER*4
0176      +                LIB$PUT_OUTPUT
0177      !
0178      ! Read a line.
0179      !
0180      PUT_OUTPUT = LIB$PUT_OUTPUT ( STRING )
0181      !
0182      ! There should be no errors here.  Signal if there are any.
0183      !
0184      IF (.NOT. PUT_OUTPUT) CALL LIB$SIGNAL( %val(PUT_OUTPUT) )
0185      !
0186      ! Return.
0187      !
0188      RETURN
0189      END

```

Interactive Service Routines

The following routines are used ONLY in the interactive version of the code and are operating system dependent. They provide functions and service routines for the interactive commands.

```

0001 C|!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
0002 C|
0003 C| The system-dependent stuff goes below here.
0004 C|
0005 C|!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
0006 C|
0007 C|++
0008 C|
0009 C| FUNCTIONAL DESCRIPTION:
0010 C|
0011 C|   These functions are the action routines invoked by the VERB which
0012 C|   follows from the on each routine.
0013 C|
0014 C| CALLING SEQUENCE:
0015 C|
0016 C|   ret-status.wlc.da = routine ( )
0017 C|
0018 C| FORMAL PARAMETERS:
0019 C|
0020 C|   NONE
0021 C|
0022 C| IMPLICIT INPUTS:
0023 C|
0024 C|   FUNCTION SPECIFIC
0025 C|
0026 C| IMPLICIT OUTPUTS:
0027 C|
0028 C|   FUNCTION SPECIFIC
0029 C|
0030 C| COMPLETION STATUS:
0031 C|
0032 C|   FUNCTION SPECIFIC
0033 C|
0034 C|   SS$NORMAL      Success, or
0035 C|   fac$status     some other status
0036 C|
0037 C| SIDE EFFECTS:
0038 C|
0039 C|   VARIABLE
0040 C|--
0041 C|   INTEGER        FUNCTION      SERVICE_ROUTINES
0042 C|   IMPLICIT       NONE
0043 C|   PARAMETER      SUCCESS = 1
0044 C|   INCLUDE '$SSDEF)/NOLIST'      ! Include system status definitions
1035 C|   INCLUDE 'SHACOM.FOR/LIST'     ! Include SHADOW common block
1036 1 C!!!
1037 1 C!!! COMMON declerations...
1038 1 C!!!
1039 1   COMMON /PIS/
1040 1   +     PI,
1041 1   +     TPI,
1042 1   +     DPR,
1043 1   +     RPD
1044 1 C!!!

```

```

1045 1 C+++ MAXIMUM DIMENSION FOR PLATES
1046 1     INTEGER      NPX
1047 1     PARAMETER (NPX=75)
1048 1 C+++ MAXIMUM DIMENSION FOR PLATE EDGES
1049 1     INTEGER      NEX
1050 1     PARAMETER (NEX=12)
1051 1 C+++ MAXIMUM DIMENSION FOR CYLINDERS
1052 1     INTEGER      NCX
1053 1     PARAMETER (NCX=5)
1054 1 C+++ MAXIMUM DIMENSION FOR CYLINDER RIMS
1055 1     INTEGER      NNX
1056 1     PARAMETER (NNX=10)
1057 1 C+++ MAXIMUM DIMENSION FOR ROWS (PHI)
1058 1     INTEGER      MAXROW
1059 1     PARAMETER (MAXROW=361)
1060 1 C+++ MAXIMUM DIMENSION FOR COLUMNS (THETA)
1061 1     INTEGER      MAXCOL
1062 1     PARAMETER (MAXCOL=181)
1063 1 C!!!
1064 1     COMMON /GEOPLA/
1065 1     +      XX      (3,NEX,NPX),
1066 1     +      V      (3,NEX,NPX),
1067 1     +      VP     (3,NEX,NPX),
1068 1     +      VN     (3,NPX),
1069 1     +      MEP    (NPX),
1070 1     +      MPX
1071 1 C!!!
1072 1     COMMON /GEOMEL/
1073 1     +      AC     (NNX,NCX),
1074 1     +      BC     (NNX,NCX),
1075 1     +      ZC     (NNX,NCX),
1076 1     +      TCR   (NNX,NCX),
1077 1     +      XCL   (3,NCX),
1078 1     +      VCL   (3,3,NCX),
1079 1     +      NEC   (NCX),
1080 1     +      MCX
1081 1 C!!!
1082 1     COMMON /EDMAG/ VMAG(NEX,NPX)
1083 1 C!!!
1084 1     COMMON /SHADWN/ COLS, ROWS, ANTENN(3),CTROID(3),
1085 1     +      MP,ME,NEXTME,MC,
1086 1     +      THET1,THET2,PH1,PH2,RESTH,RESPH,ALPH,
1087 1     +      UNIT(3),TRS(3),VRS(3,3),IUNIT,UNITF,UNITS,UN
1088 1     +      THZP,PHZP,THXP,PHXP,FILPNM,FILCNM
1089 1     COMMON /SHADWC/ INPFIL,OUTBUF(MAXCOL,MAXROW),
1090 1     +      FILCHC,FILCHP,FILCHR
1091 1 C!!!
1092 1     COMMON /PATCUI/ VPC(3,3)
1093 1 C!!!
1094 1 C!!! The first set of declarations is the stuff in /SHADOW/ common bloc
1095 1 C!!!
1096 1     INTEGER
1097 1     +      MP, ME, NEXTME, MC,
1098 1 C! Plate#/edge#/cyl# variables.
1099 1     +      FILPNM, FILCNM,
1100 1 C! Plate and cyl numbers for special filling
1101 1     +      COLS,
1102 1 C! The size of the array subsection determined
1103 1     +      ROWS
1104 1 C! by internal resolution requirements.
1105 1
1106 1     REAL
1107 1     +      CTROID,
1108 1 C! A geometric center of the object in question.

```



```

1109 1 + ANTENN,
1110 1 C! The antenna location in Ref Coord. System.
1111 1 + THEY1,
1112 1 C! The lower theta end of the range.
1113 1 + THEY2,
1114 1 C! The higher theta end of the range.
1115 1 + PHI,
1116 1 C! The lower phi end of the range.
1117 1 + PH2,
1118 1 C! The higher phi end of the range.
1119 1 + RESTH,
1120 1 C! The desired theta/phi resolution
1121 1 + RESPH,
1122 1 C! in units of radians/pixel.
1123 1 + ALPH
1124 1 C! Maximum allowed angular excursion.
1125 1
1126 1 CHARACTER
1127 1 + OUTBUF+1,
1128 1 C! The output buffer which is displayed.
1129 1 + INPFIL+63,
1130 1 C! The filename of the input set.
1131 1 + FILCHC,
1132 1 C! special fill character for cylinders
1133 1 + FILCHP,
1134 1 C! special fill character for everything else
1135 1 + FILCHR
1136 1 C! special fill character for plates.
1137 1 DATA FILCHC, FILCHP, FILCHR / 'C', 'P', 'X' /
1138 1 C!!!
1139 1 C!!! From the /PIS/ COMMON block...
1140 1 C!!!
1141 1 REAL PI, TPI, DPR, RPD
1142 1 C!!!
1143 1 C!!! From the /GEOPLA/ COMMON block...
1144 1 C!!!
1145 1 INTEGER
1146 1 + MEP,
1147 1 C! Number of edges per plate
1148 1 + MPI
1149 1 C! Total number of plates
1150 1 REAL
1151 1 + XX,
1152 1 C! The array of plate corners
1153 1 + V,
1154 1 C! Edge unit vectors
1155 1 + VP,
1156 1 C! Edge unit binormals
1157 1 + VN
1158 1 C! Unit normal for each plate
1159 1 C!!!
1160 1 C!!! From the /GEOMEL/ COMMON block...
1161 1 C!!!
1162 1 INTEGER
1163 1 + NEC,
1164 1 C! Number of sections per cylinder
1165 1 + MCX
1166 1 C! Total number of cylinders
1167 1 REAL
1168 1 + AC,
1169 1 C! Elliptic parameter along x-axis
1170 1 + BC,
1171 1 C! Elliptic parameter along y-axis
1172 1 + ZC,

```

```

1173 1 C! Cylinder endcaps in cyl coord sys
1174 1 + TCR,
1175 1 C! Angle endcap makes with positive z axis
1176 1 + ICL,
1177 1 C! Cyl coord sys origin
1178 1 + VCL
1179 1 C! Definition of cyl coord sys
1180 1 C!
1181 1 INTEGER
1182 1 + IUNIT
1183 1 REAL
1184 1 + UNITF,
1185 1 + UNITS,
1186 1 + UNITN,
1187 1 + UNIT,
1188 1 + TRS,
1189 1 + THZP,PHZP,THXP,PHXP,
1190 1 + VRS,
1191 1 + VPC,
1192 1 + VMAG
1193 1 DATA UNIT/1...3048,0.0254/
1194 1 C!
1195 1 C!!!+
1196 1 C!!! The following common block is for VMS/SMG$ software only.
1197 1 C!!!
1198 1 INTEGER KBDID, KEYIBL
1199 1 COMMON /TERCOM/ KBDID, KEYIBL
1200 1 C!!!-

1201 EXTERNAL
1202 + PUT_OUTPUT, GET_INPUT, ! My own $SMG-type I/O routines
1203 + CLI$PRESENT, !
1204 + CLI$NEGATED, !
1205 + CLI$LOCPRES, ! locally present
1206 + CLI$LOCNEG, ! locally negated
1207 + CLI$DEFAULTED,
1208 + CLI$ABSENT,
1209 + CLI$IVVALU
1210
1211 CHARACTER
1212 + P1*80, ! Command line variable
1213 + P2*80, !
1214 + UNCHAR*1, ! A character
1215 + LIBRARY*64, ! Name of the help library is defa
1216 + LABEL(3)*6 ! Units label
1217 + /'meters', 'feet ', 'inches'/,
1218 + FILE *50, ! Temporary file variable
1219 + PRTFIL*50, ! Printable file
1220 + LISFIL*50, ! Input echo listing
1221 + OUTFIL*50 ! "Plottable" output file
1222 !
1223 DATA IUNIT/1/
1224
1225 LOGICAL*4
1226 + VALID_INPUT, ! A loop control variable
1227 + CLI$PRESENT, ! CLI interface to get info about
1228 + CLI$GET_VALUE ! CLI interface to get info about
1229
1230 REAL*4
1231 + DOT,DZX,XQ(3)
1232 INTEGER*4
1233 + N,NI,NJ,STS, ! sordid variables...
1234 + KEYPAD, ! Keypad condition flag
1235 !
1236 ! General library routines

```

```

1237 |
1238 | + LIB$SPAWN, | Executes a subprocess
1239 | + LBR$OUTPUT_HELP, | The librarian help routine
1240 | + SMC$SET_KEYPAD_MODE, | Screen management package
1241 |
1242 | "SET/SHOW" routines
1243 |
1244 | + SET_ANT, SET_OUT, SET_COO,
1245 | + SET_PAT, SET_SCA, SET_WIN,
1246 | + SET_KEY, SET_INP, SET_UNI_ME
1247 | + SET_UNI_INCHES, SET_UNI_FEET,
1248 | + SHOW_ANT, SHOW_OUT, SHOW_COO,
1249 | + SHOW_PAT, SHOW_SCA, SHOW_WIN,
1250 | + SHOW_KEY, SHOW_INP,
1251 | + SHOW_UNI,
1252 |
1253 | various command routines
1254 |
1255 | + EXIT_COMMAND, HELP_COMMAND, DCL_COMMAND, SHADOW_COM
1256 |
1257 C|!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1258 C|
1259 C| This routine sets the current fill characters being used for plates
1260 C| or cylinders.
1261 C|
1262 ENTRY SET_FIL
1263 IF ( CLI$PRESENT( 'SEQUENTIAL' ) ) THEN
1264 C|
1265 C| Reset things to their default state.
1266 C|
1267 FILPNM = -1
1268 FILCNM = -1
1269 FILCHP = 'P'
1270 FILCHC = 'C'
1271 FILCHR = 'X'
1272 C|
1273 C| To avoid screwing up the test in SCAN, use a character that will
1274 C| not be used by the fill process, like char 7.
1275 C| Set a plate up for tagging.
1276 C|
1277 ELSEIF ( CLI$PRESENT( 'PLATE' ) ) THEN
1278 C|
1279 C| Clear any cylinder tagging residue.
1280 C|
1281 FILCNM = 0
1282 FILCHC = 'C'
1283 C|
1284 C| Get the master fill character.
1285 C|
1286 CALL CLI$GET_VALUE( 'P2', FILCHR )
1287 C|
1288 C| Get the qualifier numeral value. STS is being used for the length of
1289 C| and the status of the decode.
1290 C|
1291 IF ( CLI$GET_VALUE( 'PLATE', P2, STS ) ) THEN
1292 DECODE (STS,1,P2,IOSTAT=STS) FILPNM
1293 ELSE
1294 STS = -1
1295 ENDIF
1296 C|
1297 C| Get the fill character for that plate. Use a 'P' if none is given.
1298 C|
1299 IF ( STS .NE. 0 ) THEN
1300 SET_FIL = %LOC( CLI$_IVALU )

```

```

1301         ELSE
1302             IF ( .NOT. CLI$GET_VALUE('PLATE',FILCHP) ) THEN
1303                 FILCHP = 'P'
1304             ENDIF
1305         ENDIF
1306     C!
1307     C! Set a cylinder up for tagging.
1308     C!
1309         ELSEIF ( CLI$PRESENT( 'CYLINDER' ) ) THEN
1310     C!
1311     C! Clear any cylinder tagging residue.
1312     C!
1313             FILPHM = 0
1314             FILCHP = 'P'
1315     C!
1316     C! Get the master fill character.
1317     C!
1318             CALL CLI$GET_VALUE( 'P2', FILCHR )
1319     C!
1320     C! Get the qualifier numeral value. STS is being used for the length of
1321     C! and the status of the decode.
1322     C!
1323             IF ( CLI$GET_VALUE( 'CYLINDER', P2, STS ) ) THEN
1324                 DECODE (STS,1,P2,IDSTAT=STS) FILCNM
1325             ELSE
1326                 STS = -1
1327             ENDIF
1328     C!
1329     C! Get the fill character for that cylinder. Use a 'C' if none is given
1330     C!
1331             IF ( STS .NE. 0 ) THEN
1332                 SET_FIL = %LOC( CLI$_IVVALU )
1333             ELSE
1334                 IF ( .NOT. CLI$GET_VALUE( 'CYLINDER', FILCHC ) ) THEN
1335                     FILCHC = 'C'
1336                 ENDIF
1337             ENDIF
1338     C!
1339     C! The else here is for a "SET FILL [x]" command.
1340     C!
1341             ELSE
1342     C!
1343     C! Get the master fill character.
1344     C!
1345             CALL CLI$GET_VALUE( 'P2', FILCHR )
1346     C!
1347     C! End of cases.
1348     C!
1349             ENDIF
1350
1351             GOTO 3
1352     1  FORMAT( I )
1353
1354     C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1355     C!
1356     C! This routine displays the current fill characters being used for plat
1357     C! or cylinders.
1358     C!
1359             ENTRY SHOW_FIL
1360     C!
1361     C! Assume success only when the SHOW command is being executed.
1362     C!
1363             SHOW_FIL = SUCCESS
1364     C!

```

```

1365 C| Examine the plate situation.
1366 C|
1367 3 IF ( FILPNM .GT. 0 ) THEN
1368 WRITE(2,FMT='(' Plate ',I3,' is tagged with ['',A,'']')')
1369 + FILPNM, FILCHP
1370 WRITE(2,FMT='(' All other geometry tagged with ['',A,'']')')
1371 + FILCHR
1372 ENDIF
1373 C|
1374 C| Examine the cylinder situation.
1375 C|
1376 IF ( FILCNM .GT. 0 ) THEN
1377 WRITE(2,FMT='(' Cylinder ',I3,' tagged with ['',A,'']')')
1378 + FILCNM, FILCHC
1379 WRITE(2,FMT='(' All other geometry tagged with ['',A,'']')')
1380 + FILCHR
1381 ENDIF
1382 C|
1383 C| Check on a no-tag background character situation.
1384 C|
1385 IF ( (FILCNM .EQ. 0) .AND. (FILPNM .EQ. 0) ) THEN
1386 WRITE(2,FMT='(' No individual plates/cylinders are tagged')')
1387 WRITE(2,FMT='(' All geometry marked by ['',A,'']')') FILCHR
1388 ENDIF
1389 C|
1390 C| Report the sequential numbering case.
1391 C|
1392 IF ( ( FILCNM .LT. 0 ) .AND. ( FILPNM .LT. 0 ) )
1393 + WRITE(2,FMT='(' All cylinders/plates sequentially tagged')')
1394 C|
1395 RETURN
1396
1397 C|!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1398 C|
1399 C| This routine sets the antenna location.
1400 C|
1401 ENTRY SET_ANT
1402 WRITE (2,FMT='(1X,'Input antenna location in ',A6,'': ',I4,'')')
1403 + LABEL(IUNIT)
1404 READ (1,*) ANTENN(1), ANTENN(2), ANTENN(3)
1405 C|
1406 C| Perform appropriate units conversion here.
1407 C|
1408 DD 3424 N=1,3
1409 3424 IQ(N)=ANTENN(N)
1410
1411 DD 3425 N=1,3
1412 3425 ANTENN(N)=UNITS*
1413 + ( IQ(1)*VRS(1,N) + IQ(2)*VRS(2,N) + IQ(3)*VRS(3,N) ) + TRS(N)
1414
1415 SET_ANT = SUCCESS
1416 C|
1417 C| This routine displays the current antenna position.
1418 C|
1419 ENTRY SHOW_ANT
1420 C|
1421 C| Transform the antenna back
1422 C|
1423 DD N=1,3
1424 IQ(N) = ( ( ANTENN(1)-TRS(1)) * VRS(N,1) +
1425 + ( ANTENN(2)-TRS(2)) * VRS(N,2) +
1426 + ( ANTENN(3)-TRS(3)) * VRS(N,3) ) / UNITS
1427 END DO
1428 C|

```

```

1429     WRITE(2,FMT='(' Antenna in RCS (meters): ',3F12.5)') ANTENN
1430     WRITE(2,FMT='(' Definit system ('',A,''): ',3F12.5)')
1431     + LABEL(IUNIT), XQ
1432     SHOW_ANT = SUCCESS
1433     RETURN
1434
1435 C! ~~~~~
1436 C!
1437 C! Process a new input set.  Inquire about the full name.
1438 C!
1439     ENTRY SET_INP
1440     CALL CLI$GET_VALUE( 'P2', FILE )
1441     OPEN ( UNIT=5, FILE=FILE, DEFAULTFILE='.INP', STATUS='OLD')
1442     CALL ABSCIN
1443     SET_INP = SUCCESS
1444 C!
1445 C! This routine displays the current input set name.
1446 C!
1447     ENTRY SHOW_INP
1448     INQUIRE ( UNIT=5, NAME=INPFIL )
1449     TYPE +, 'The current input set is ', INPFIL
1450     SHOW_INP = SUCCESS
1451     RETURN
1452
1453 C! ~~~~~
1454 C!
1455 C! This routine toggles/report keypad mode.
1456 C!
1457     ENTRY SET_KEY
1458     IF ( .NOT. CLI$PRESENT( 'KEYPAD_MODE' ) ) THEN
1459     KEYPAD = 0
1460     ELSE
1461     KEYPAD = 1
1462     END IF
1463     SET_KEY = SMG$SET_KEYPAD_MODE( KBDID, KEYPAD )
1464 C!
1465 C! This routine displays the current keypad mode.
1466 C!
1467     ENTRY SHOW_KEY
1468     IF ( KEYPAD .EQ. 0 ) THEN
1469     WRITE(2,*) 'The keyboard is not in keypad mode.'
1470     ELSE
1471     WRITE(2,*) 'The keyboard is in keypad mode.'
1472     END IF
1473     RETURN
1474
1475 C! ~~~~~
1476 C!
1477 C! Set up a coordinate system.
1478 C!
1479     ENTRY SET_COO
1480 C$$$
1481 C$$$ TRS(N)=LINEAR TRANSLATION OF COORDINATES FROM THE FIXED
1482 C$$$ COORDINATES WHICH IS ORIGINALLY SET UP BY OPERATOR.
1483 C$$$
1484     TYPE 3921,LABEL(IUNIT)
1485     3921 FORMAT(' Please input a translation vector in ',A6,' : ')
1486     accept+, (TRS(N),N=1,3)
1487     DO 3920 N=1,3
1488     3920 TRS(N)=TRS(N)*UNITS
1489 C$$$
1490 C$$$ THZP,PHZP=ORIENTATION OF THE VRS(3,N) AXIS RELATIVE TO THE
1491 C$$$ FIXED COORDINATE SYSTEM.
1492 C$$$

```

```

1493 C$$$ THXP,PHXP=ORIENTATION OF THE VRS(1,N) AXIS RELATIVE TO THE
1494 C$$$ FIXED COORDINATE SYSTEM.
1495 C$$$
1496 123 continue
1497 type*, 'Please input THZP,PHZP,THXP,PHXP in degrees:'
1498 accept*, THZP,PHZP,THXP,PHXP
1499 VRS(3,1)=SIN(THZP*RPD)*COS(PHZP*RPD)
1500 VRS(3,2)=SIN(THZP*RPD)*SIN(PHZP*RPD)
1501 VRS(3,3)=COS(THZP*RPD)
1502 VRS(1,1)=SIN(THXP*RPD)*COS(PHXP*RPD)
1503 VRS(1,2)=SIN(THXP*RPD)*SIN(PHXP*RPD)
1504 VRS(1,3)=COS(THXP*RPD)
1505 C111 INSURE VRS(1,N) IS PERPENDICULAR TO VRS(3,N)
1506 DZX=VRS(3,1)*VRS(1,1)+VRS(3,2)*VRS(1,2)+VRS(3,3)*VRS(1,3)
1507 IF(ABS(DZX).GT.0.1) THEN
1508 TYPE*, 'The coordinates are NOT orthogonal - Respecify.'
1509 goto 123
1510 ELSE
1511 VRS(1,1)=VRS(1,1)-VRS(3,1)*DZX
1512 VRS(1,2)=VRS(1,2)-VRS(3,2)*DZX
1513 VRS(1,3)=VRS(1,3)-VRS(3,3)*DZX
1514 DOT=VRS(1,1)*VRS(1,1)+VRS(1,2)*VRS(1,2)+VRS(1,3)*VRS(1,3)
1515 DOT=SQRT(DOT)
1516 VRS(1,1)=VRS(1,1)/DOT
1517 VRS(1,2)=VRS(1,2)/DOT
1518 VRS(1,3)=VRS(1,3)/DOT
1519 VRS(2,1)=VRS(3,2)*VRS(1,3)-VRS(3,3)*VRS(1,2)
1520 VRS(2,2)=VRS(3,3)*VRS(1,1)-VRS(3,1)*VRS(1,3)
1521 VRS(2,3)=VRS(3,1)*VRS(1,2)-VRS(3,2)*VRS(1,1)
1522 WRITE(6,3931)
1523 END IF
1524 C1
1525 C1 Display the coordinates
1526 C1
1527 ENTRY SHOW_COO
1528 C1
1529 3931 FORMAT(2H *,5X,'The following rotations are used for ALL',
1530 2' subsequent inputs:',T79,1H*)
1531 DO 3932 NI=1,3
1532 3932 WRITE(6,3933) (NI,NJ,VRS(NI,NJ),NJ=1,3)
1533 3933 FORMAT(2H *,1X,3(2X,'VRS(',I1,',',I1,')=' ,F9.6),T79,1H*)
1534 C1
1535 RETURN
1536
1537 C1!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1538 C1
1539 C1 set up pattern cut coordinate system
1540 C1
1541 ENTRY SET_PAT
1542 C$$$
1543 C$$$ THZP,PHZP=ORIENTATION OF THE VPC(3,N) AXIS RELATIVE TO THE
1544 C$$$ FIXED COORDINATE SYSTEM.
1545 C$$$
1546 C$$$ THXP,PHXP=ORIENTATION OF THE VPC(1,N) AXIS RELATIVE TO THE
1547 C$$$ FIXED COORDINATE SYSTEM.
1548 C$$$
1549 1234 continue
1550 type*, 'Please input THZP,PHZP,THXP,PHXP in degrees:'
1551 accept*, THZP,PHZP,THXP,PHXP
1552 VPC(3,1)=SIN(THZP*RPD)*COS(PHZP*RPD)
1553 VPC(3,2)=SIN(THZP*RPD)*SIN(PHZP*RPD)
1554 VPC(3,3)=COS(THZP*RPD)
1555 VPC(1,1)=SIN(THXP*RPD)*COS(PHXP*RPD)
1556 VPC(1,2)=SIN(THXP*RPD)*SIN(PHXP*RPD)

```

```

1557     VPC(1,3)=COS(THXP*RPD)
1558 C!!! INSURE VPC(1,N) IS PERPENDICULAR TO VPC(3,N)
1559     DZX=VPC(3,1)*VPC(1,1)+VPC(3,2)*VPC(1,2)+VPC(3,3)*VPC(1,3)
1560     IF(ABS(DZX).GT.0.1) THEN
1561         TYPE*, 'The coordinates are NOT orthogonal - Respecify.'
1562         goto 1234
1563     ELSE
1564         VPC(1,1)=VPC(1,1)-VPC(3,1)*DZX
1565         VPC(1,2)=VPC(1,2)-VPC(3,2)*DZX
1566         VPC(1,3)=VPC(1,3)-VPC(3,3)*DZX
1567         DOT=VPC(1,1)*VPC(1,1)+VPC(1,2)*VPC(1,2)+VPC(1,3)*VPC(1,3)
1568         DOT=SQRT(DOT)
1569         VPC(1,1)=VPC(1,1)/DOT
1570         VPC(1,2)=VPC(1,2)/DOT
1571         VPC(1,3)=VPC(1,3)/DOT
1572         VPC(2,1)=VPC(3,2)*VPC(1,3)-VPC(3,3)*VPC(1,2)
1573         VPC(2,2)=VPC(3,3)*VPC(1,1)-VPC(3,1)*VPC(1,3)
1574         VPC(2,3)=VPC(3,1)*VPC(1,2)-VPC(3,2)*VPC(1,1)
1575         WRITE(6,3931)
1576     END IF
1577 C!
1578 C! re-display the pattern cut system
1579 C!
1580     ENTRY SHOW_PAT
1581     DO NI=1,3
1582         WRITE(6,4933) (NI,NJ,VPC(NI,NJ),NJ=1,3)
1583     END DO
1584 4933 FORMAT(2H *,1X,3(2X,'VPC(',I1,',',I1,')=',F9.5),179,1H*)
1585     RETURN
1586
1587 C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1588 C!
1589 C! This routine sets/displays a scale factor.
1590 C!
1591     ENTRY SET_SCA
1592         WRITE (2,*) ' Please input a uniform scale factor:'
1593         READ (1,*) UNITF
1594         UNITS = UNITN * UNITF
1595 C!
1596 C! This entry displays the uniform scale factor.
1597 C!
1598     ENTRY SHOW_SCA
1599         WRITE (2,FMT='(' The uniform scale factor is ',F10.8)') UNITF
1600     RETURN
1601
1602 C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1603 C!
1604 C! This routine sets the units for the program.
1605 C!
1606 C! IUNIT = Indicator of units used for input data.
1607 C!
1608 C!         1=METERS, 2=FEET, 3=INCHES
1609 C!
1610     ENTRY         SET_UNI_METERS
1611         IUNIT = 1
1612         GOTO 2
1613     ENTRY         SET_UNI_FEET
1614         IUNIT = 2
1615         GOTO 2
1616     ENTRY         SET_UNI_INCHES
1617         IUNIT = 3
1618 2     UNITH = UNIT( IUNIT )
1619         UNITS = UNITN * UNITF
1620     RETURN

```



```

1621
1622 C!
1623 C!
1624 C! This entry shows the current units.
1625 C!
1626     ENTRY SHOW_UNI
1627     WRITE (2,FMT='(' The current units are ',A6') LABEL( IUNIT )
1628     RETURN
1629
1630 C!
1631 C!
1632 C! This routine sets the window.
1633 C!
1634     ENTRY SET_WIN
1635
1636     VALID_INPUT = .FALSE.
1637     DO WHILE ( .NOT. VALID_INPUT )
1638     |
1639         TYPE*, 'The current range of theta in degrees is ', THET1+dpr,
1640         +      ' to ', THET2*DPR
1641         TYPE*, 'with a resolution of ', RESTH*DPR, ' degrees/pixel.'
1642         TYPE*, 'The current range of phi in degrees is ', PH1+dpr,
1643         +      ' to ', PH2*DPR
1644         TYPE*, 'with a resolution of ', RESPH*DPR, ' degrees/pixel.'
1645     |
1646         TYPE*, 'Please enter a new range for theta (lower,higher):'
1647         ACCEPT*, THET1, THET2
1648         THET1 = THET1 * RPD
1649         THET2 = THET2 * RPD
1650     |
1651         TYPE*, 'Please enter a new THETA resolution in degrees/pixel:'
1652         ACCEPT*, RESTH
1653         RESTH = RESTH * RPD
1654     |
1655         TYPE*, 'Please enter a new range for phi (lower,higher):'
1656         ACCEPT*, PH1, PH2
1657         PH1 = PH1 * RPD
1658         PH2 = PH2 * RPD
1659     |
1660         TYPE*, 'Please enter a new PHI resolution in degrees/pixel:'
1661         ACCEPT*, RESPH
1662         RESPH = RESPH * RPD
1663     |
1664         ROWS = INT( (PH2 - PH1) / RESPH + 0.5 ) + 1
1665         COLS = INT( (THET2 - THET1) / RESTH + 0.5 ) + 1
1666
1667         VALID_INPUT = (.NOT. (ROWS.GT.MAXROW) ).OR.
1668         +      (.NOT. (COLS.GT.MAXCOL) )
1669         IF ( .NOT. VALID_INPUT ) WRITE(2,*)
1670         +      ' Insufficient dimensions for specified resolution.'
1671         END DO
1672 C!
1673 C! Show the window parameters
1674 C!
1675     ENTRY SHOW_WIN
1676     TYPE*, 'The current range of theta in degrees is ', THET1+dpr,
1677     +      ' to ', THET2*DPR
1678     TYPE*, 'with a resolution of ', RESTH*DPR, ' degrees/pixel.'
1679     TYPE*, 'The current range of phi in degrees is ', PH1+dpr,
1680     +      ' to ', PH2*DPR
1681     TYPE*, 'with a resolution of ', RESPH*DPR, ' degrees/pixel.'
1682     RETURN
1683
1684 C!

```

```

1685 C!
1686 C! This routine determines names of output files. Here are the current
1687 C! assignments.
1688 C!
1689 C! Unit          Meaning          Default file n
1690 C! 1             interactive input      sys$input
1691 C! 2             interactive output     sys$output
1692 C! 5             input processor input   FILE.INP
1693 C! 6             input processor (echo) output FILE.LIS
1694 C! 7             printable output file  FILE.PRT
1695 C! 10            "plot" data output file  FILE.PLT
1696 C!
1697 ENTRY SET_OUT
1698 CALL CLI$GET_VALUE( 'P2', FILE )
1699 C!
1700 C! Only if /NO PLOT is specified, then discard all output written to unit
1701 C! The user should always get plottable output by default.
1702 C!
1703 IF ( .NOT. CLI$PRESENT('PLOTTABLE') ) THEN
1704 OPEN( UNIT=10, FILE='_NL:', STATUS='OLD', FORM='UNFORMATTED' )
1705 ELSE
1706 OPEN( UNIT=10, FILE='.PLT', DEFAULTFILE=FILE, STATUS='NEW',
1707 + FORM='UNFORMATTED' )
1708 ENDIF
1709 C!
1710 C! If /PRINT is not specified, discard all output written to unit 7.
1711 C! The user only wants to see the line printer if he asks for it.
1712 C!
1713 OPEN( UNIT=7, STATUS='OLD', FILE='_NL:' )
1714 IF ( CLI$PRESENT( 'PRINTABLE' ) ) THEN
1715 OPEN( UNIT=7, DEFAULTFILE=FILE, STATUS='NEW', FILE='.PRT' )
1716 ENDIF
1717 C!
1718 C! If /NOECHO is specified, the input echo is discarded.
1719 C! The user should get an echo file by default, just like a .PLT file.
1720 C!
1721 IF ( .NOT. CLI$PRESENT( 'ECHOING' ) ) THEN
1722 OPEN( UNIT=6, FILE='_NL:', STATUS='OLD' )
1723 ELSE
1724 OPEN( UNIT=6, FILE='.LIS', DEFAULTFILE=FILE, STATUS='NEW' )
1725 ENDIF
1726 C!
1727 C! Now retrieve the full filenames for future reference.
1728 C!
1729 INQUIRE ( UNIT = 10, NAME = OUTFIL )
1730 INQUIRE ( UNIT = 7, NAME = PRIFIL )
1731 INQUIRE ( UNIT = 6, NAME = LISFIL )
1732 SET_OUT = SUCCESS
1733 C!
1734 C! This routine displays the current output files.
1735 C!
1736 ENTRY SHOW_OUT
1737 TYPE *, 'Plotting file is: ', OUTFIL
1738 TYPE *, 'Printer file is: ', PRIFIL
1739 TYPE *, 'Input echo file: ', LISFIL
1740 SHOW_OUT = SUCCESS
1741 RETURN
1742
1743 C!
1744 C!
1745 C! This routine stops the program.
1746 C!
1747 ENTRY EXIT_COMMAND
1748 CALL EXIT

```

```

1749         RETURN
1750
1751 C)!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1752 C!
1753 C! This routine services online help requests.
1754 C!
1755         ENTRY  HELP_COMMAND
1756         LIBRARY = ' '
1757         P1 = ' '
1758         CALL CLI$GET_VALUE( 'P1', P1 )
1759         CALL CLI$GET_VALUE( 'HELPLIB', LIBRARY )
1760         HELP_COMMAND = LBR$OUTPUT_HELP(
1761         +         PUT_OUTPUT,,                ! Help output rout
1762         +         P1,                          ! Help key descrip
1763         +         LIBRARY,,                    ! Help library nam
1764         +         GET_INPUT                    ) ! The prompting in
1765         RETURN
1766
1767 C)!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1768 C!
1769 C! This routine calls the routines which do the shadowing.
1770 C!
1771         ENTRY  SHADOW_COMMAND
1772         TYPE*, 'Working...'                ! Type an informational me
1773         CALL INITGF                          ! Initialize next plot
1774         CALL DOPLAS                          ! Draw the plates
1775         CALL DOCYLS                          ! Draw the cylinders
1776         CALL WRTOUT                          ! Write the output buffer
1777         SHADOW_COMMAND = SUCCESS            ! Return a normal
1778         RETURN
1779
1780 C)!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
1781 C!
1782 C! This routine executes a DCL command as a subprocess. Add a test for
1783 C! better behavior with blank Pis.
1784 C!
1785         ENTRY  DCL_COMMAND
1786         CALL CLI$GET_VALUE( 'P1', P1 )
1787         IF ( P1 .EQ. ' ' ) THEN
1788             DCL_COMMAND = LIB$SPAWN()
1789         ELSE
1790             DCL_COMMAND = LIB$SPAWN( P1 )
1791         ENDIF
1792         RETURN
1793 C!
1794 C! End of action routines.
1795 C!
1796         END

```

10.2 Non-VAX/VMS Subroutines

The following routines are for the non-interactive implementations of the code. They are used in conjunction with the routines in this chapter that are common to both versions.

MAIN PROGRAM (non-interactive)

This is the main routine to be used with the non-interactive code.

```
0001      PROGRAM SHADOW
0002      C!!!
0003      C!!! THIS COMPUTE CODE WAS WRITTEN AT THE OHIO STATE UNIVERSITY
0004      C!!! ELECTROSCIENCE LABORATORY. ANY PROBLEMS OR COMMENTS
0005      C!!! CAN BE REFERRED TO:
0006      C!!!
0007      C!!! RONALD J. MARHEFKA OR LASZLO A. TAKACS
0008      C!!! ELECTROSCIENCE LABORATORY
0009      C!!! 1320 KINNEAR RD.
0010      C!!! COLUMBUS, OHIO 43212
0011      C!!! PHONE: (614) 422-5752 OR 422-5648
0012      C!!!
0013      C!!! THIS COMPUTER CODE CALCULATES SHADOWING OF AN ANTENNA
0014      C!!! USING THE NEC-BSC INPUTS NON-INTERACTIVELY.
0015      C!!! IT SHOULD BE USED IN PLACE OF INTERACTIVE MAIN PROGRAM
0016      C!!! WHEN THE SHADOW CODE IS USED ON NON VAX COMPUTERS.
0017      C!!!
0018      INCLUDE 'SHACOM.FOR'
0184      PARAMETER (NSX=30)
0185      COMPLEX WS
0186      LOGICAL LRET
0187      COMMON/SORARY/WS(NSX),XSS(3,NSX),MSA(2,NSX),MSX,MSP,MSPP
0188      C!!! Initialize fill tags
0189      C!!! FILPNN and FILCMM < 0 is sequential tagging
0190      C!!! FILPHM or FILCMM > 0 that object is tagged with
0191      C!!! FILCHP or FILCHC
0192      C!!! FILPHM or FILCMM = 0 everything tagged with FILCHR
0193      FILPHM=0
0194      FILCMM=0
0195      C!!! Initialize fill characters
0196      FILCHP='P'
0197      FILCHC='C'
0198      FILCHR='X'
0199      C!!! Initialize return flag
0200      LRET=.TRUE.
0201      C!!! Initialize and read command information.
0202      CALL ABSCIN
0203      100 CONTINUE
0204      C!!! Choose a source location from stored positions.
0205      DO 1200 MS=1,MSX
0206      DO 1000 N=1,3
0207      1000 ANTENN(N)=XSS(N,MS)
0208      C!!! Initialize graphics information.
0209      CALL INITGF
0210      C!!! Calculate shadow of plates.
0211      CALL DOPLAS
0212      C!!! Calculate shadow of cylinders.
0213      CALL DOCYLS
0214      C!!! Write out maps to printer and plotter files.
0215      CALL WRTOUT
0216      1200 CONTINUE
```

0217 C!!! Read more command information.
0218 CALL ABSCRE
0219 C!!! Return to execute next shadow map.
0220 IF(LRET) GO TO 100
0221 STOP
0222 END

10.3 Subroutines common to both modes

The following routines are used by both the interactive and non-interactive implementations of the code. They are written in transportable FORTRAN-77.

SUBROUTINE ABSCIN

This is the input-set processor routine. It reads commands from the input file which define the input geometry.

```
0001 C-----
0002 SUBROUTINE ABSCIN
0003 C!!!
0004 C!!! THE NEC - BASIC SCATTERING CODE (NEC-BSC) WAS WRITTEN
0005 C!!! AT THE OHIO STATE UNIVERSITY ELECTROSCIENCE LABORATORY.
0006 C!!! ANY PROBLEMS OR COMMENTS CAN BE REFERRED TO:
0007 C!!!
0008 C!!! RONALD J. MARHEFKA
0009 C!!! ELECTROSCIENCE LABORATORY
0010 C!!! 1320 KINNEAR RD.
0011 C!!! COLUMBUS, OHIO 43212
0012 C!!! PHONE: (614) 422-6762
0013 C!!!
0014 C!!! THIS IS A PORTION OF THE MAIN PROGRAM OF THE NEC-BSC.
0015 C!!! IT READS IN THE INPUT AND PASSES THE GEOMETRY INFORMATION
0016 C!!! TO THE SHADOW CALCULATION PART OF THIS OBSCURATION CODE.
0017 C!!! IT READS LOCATIONS OF SOURCES A NUMBER OF FINITE
0018 C!!! PLATES AND/OR A SET OF FINITE
0019 C!!! ELLIPTIC CYLINDERS AND CONE FRUSTUM SECTIONS.
0020 C!!! THE PLATES ARE DEFINED
0021 C!!! BY THEIR CORNER LOCATIONS. THEY CAN BE PERFECTLY
0022 C!!! CONDUCTING, MULTI LAYERED DIELECTRIC SLABS, OR COATED
0023 C!!! METAL PLATES. AN INFINITE GROUND PLANE CAN ALSO BE
0024 C!!! ADDED. THE CYLINDERS ARE DEFINED BY THEIR ORIGIN,
0025 C!!! AXES DIRECTIONS, AND BY THE RADIUS ON THEIR MAJOR
0026 C!!! AND MINOR AXES AND THE ENDCAPS AND FRUSTUM RIMS ARE DEFINED BY
0027 C!!! THEIR POSITION ON THE CYLINDER AXIS AND THE ANGLE
0028 C!!! OF THEIR SURFACES WITH THE CYLINDER AXIS IN THE X-Z
0029 C!!! CYLINDER PLANE. THE CYLINDERS MUST BE PERFECTLY
0030 C!!! CONDUCTING. AS DIMENSIONED, IT CAN HANDLE 76 PLATES
0031 C!!! WITH A MAXIMUM OF 12 CORNERS PER PLATE, WITH 6 LAYERS
0032 C!!! OF DIELECTRIC AND 6 CYLINDERS, WITH 10 RIMS
0033 C!!! ALSO 30 TRANSMITTING
0034 C!!! ELEMENTS AND 30 RECEIVING ELEMENTS CAN BE INPUT.
0035 C!!! NOTE THAT THE LIMITS ON THE NUMBER OF PLATES,
0036 C!!! CORNERS, LAYERS, CYLINDERS, SOURCES, AND RECEIVERS
0037 C!!! ARE ONLY DUE TO THE SIZE OF THE ARRAYS.
0038 C!!! THE LINEAR DIMENSIONS ARE INPUT IN METERS UNLESS
0039 C!!! SPECIFIED OTHERWISE. THE ANGULAR DIMENSIONS
0040 C!!! ARE IN DEGREES.
0041 C!!!
0042 C!!! NOTE THAT COMMENTS ARE INDICATED IN DIFFERENT FORMS:
0043 C!!! C!!! IMPLIES EXPLANATION OF PROGRAM SECTION
0044 C!!! C$$$ IMPLIES DESCRIPTION OF INPUT DATA
0045 C!!! C=== IMPLIES COMMAND INPUT SECTION
0046 C!!! C--- IMPLIES BEGINNING OF SUBROUTINE
0047 C!!! C+++ IMPLIES SPECIFICATION OF MAXIMUM DIMENSIONS
0048 C!!! CXXX means lines were not needed for SHADOW program
0049 C!!! CFFF means lines were not implemented for current version
0050 C!!!
```

0051 C111 NEC-BSC VERSION 2.6-1 (UPDATED 8/16/85)
0052 C111
0053 C111 MAJOR VERSION CHANGES ARE DENOTED BY THE FIRST DIGIT
0054 C111 MINOR CHANGES IN CAPABILITY ARE DENOTED BY THE DECIMAL
0055 C111 POINTS AND MINOR CHANGES THAT DO NOT NEED ADDED
0056 C111 DOCUMENTATION ARE SHOWN AFTER THE DASH.
0057 C111
0058 C111 NOTE ON VERSION 2.2
0059 C111 1) THE PLATE - CYLINDER TERMS ARE NOT PRESENTLY INCLUDED.
0060 C111 2) THE CYLINDER - CYLINDER INTERACTION TERMS WORK ONLY
0061 C111 FOR PARALLEL CYLINDERS WITH THE PATTERN CUT
0062 C111 PERPENDICULAR TO THE CYLINDER AXES.
0063 C111
0064 C111 NOTE ON VERSION 2.3
0065 C111 RANGE GATING HAS BEEN ADDED IN THE NEAR ZONE
0066 C111
0067 C111 NOTE ON VERSION 2.4
0068 C111 VOLUMETRIC PATTERN CAPABILITY HAS BEEN ADDED
0069 C111
0070 C111 NOTE ON VERSION 2.5
0071 C111 PARAMETER STATEMENTS FOR DIMENSIONS ADDED
0072 C111 ARRAY INDICES CHANGED FOR MORE EFFICIENCY
0073 C111
0074 C111 NOTE OF VERSION 2.6
0075 C111 CONE FRUSTUM INPUT ADDED
0076 C111
0077 C+++
0078 C+++ SPECIFICATION OF MAXIMUM DIMENSION SIZES
0079 C+++
0080 C+++ MAXIMUM DIMENSION FOR OBSERVATION POINTS
0081 C+++ PARAMETER (NOX=1801)
0082 C+++ MAXIMUM DIMENSION FOR PLATE DIELECTRIC LAYERS
0083 C+++ PARAMETER (NLX=5)
0084 C+++ MAXIMUM DIMENSION FOR SOURCES
0085 C+++ PARAMETER (NSX=30)
0086 C+++ MAXIMUM DIMENSION FOR RECEIVERS
0087 C+++ PARAMETER (NRX=30)
0088 C+++
0089 C+++ INCLUDE 'SHACOM.FOR'
0255 C111 COMPLEX CJ, CPI4, WS, WR
0256 C111 COMPLEX CI11, CI22, Z11, Z22
0257 C111 CHARACTER*2 IT(40), IR(36), LABEL(3)*6
0258 C111 CHARACTER RUNDAT*9, RUNTIM*8
0259 C111 DIMENSION IMS(NSX), HS(NSX), HAWS(NSX), VISS(3,3,NSX)
0260 C111 DIMENSION IMR(NRX), HR(NRX), HAWR(NRX), VIRR(3,3,NRX)
0261 C111 DIMENSION XRR(3,NRX), VXRP(3,3,NRX)
0262 C111 DIMENSION IPC(3), VRT(3,3), TR(3)
0263 C111 DIMENSION JMI(4), DR(3), DT(3), DP(3), RDR(3)
0264 C111 DIMENSION IQR(3), IQ(3)
0265 C111 LOGICAL LKJ(4,6), LFQG, LWARN, LSCAT, LPPREC
0266 C111 LOGICAL LSOR, LOUT, LSRFC, LSURF, LSHD, LCYL, LPLA
0267 C111 LOGICAL LIHD, LDEBUG, LTEST, LSLOPE, LCONR, LDC
0268 C111 LOGICAL LWRITE, LPLT, LGRND, LSMP, LRMP, LPRAD, LRANG, LCPAT
0269 C111 LOGICAL LNEAR, LRCVR, LRECT, LVOLP, LVPLT, LFARN
0270 C111 COMMON/SORDAT/IM, H, HAW, FACTOR
0271 C111 COMMON/SORARY/WS(NSX), XSS(3,NSX), MSA(2,NSX), MSX, MSP, MSPP
0272 C111 COMMON/TEST/LDEBUG, LTEST, LWARN
0273 C111 COMMON/SORINF/IS(3), VXS(3,3)
0274 C111 COMMON/IMAINF/II(3,NPX,NPX), VKI(3,3,NPX)
0275 C111 COMMON/RECINF/WR(NRX), IMRP, HRPP, HAWRP, VXR(3,3), MR
0276 C111 COMMON/RECAR/IRP(3,NRX), MRA(2,NRX), DRP(3), DTP(3), DPP(3)
0277 C111 COMMON/LIMIT/SML, SMLR, SMLT, BIG
0278 C111 COMMON/DIR/RD(3), D(3), LNEAR, LRCVR
0279 C111 COMMON/WAVE/WK, WL

```

0280      COMMON/COMP/CJ,CPI4
0281      COMMON/FNANG/FNP(NEX,NPX)
0282      COMMON/LPLCY/LPLA,LCYL
0283      COMMON/GROUND/LGRND,MPXR
0284      COMMON/OUTPFZ/TPPD,PRAD,RANG,LCNPAT,LPRAD,LRANG
0285      COMMON/OUTPNZ/RXS,RXI,TYS,TYI,PZS,PZI,LRECT
0286      COMMON/OUTPNV/IVPN,IV,LVOLF
0287      COMMON/TRANDT/LSLAB(NPX),NSLAB(NPX),DSLAB(NLX,NPX)
0288      2,ERSLAB(NLX,NPX),TESLAB(NLX,NPX),URSLAB(NLX,NPX)
0289      3,TMSLAB(NLX,NPX)
0290      DATA LABEL/'METERS','FEET ','INCHES'/
0291      DATA IT/'IO','PD','PG','SG','LP','PP','GP','XQ','RT','CG'
0292      2,'SM','RD','CM','CE','BP','UF','RM','UN','FR','NX'
0293      3,'EN','NP','NC','NG','NS','PR','US','PN','RG','NR'
0294      4,'SA','FM','RA','GR','VD','VN','VP','PF','VF','CC'/
0295      C111 MAX. DIMENSION OF SOURCES,RECEIVERS,CYLINDERS,RIMS,PLATES,EDGES,
0296      C111 LAYERS, AND OBSERVATION POINTS.
0297      MSDX=NSX
0298      MRDX=NRX
0299      MCDX=NCX
0300      NCDX=NNX
0301      MPDX=NPX
0302      MEDX=NEX
0303      MLDX=NLX
0304      MODX=NOX
0305      C111 NOTE: IN SUB. RFPCTL THE VARIABLES IVD,PHOR,PHORP,AND VRO
0306      C111 MUST BE DIMENSIONED 2*MPDX+1
0307      C111
0308      C111 SET TIME FLAGS TO ZERO
0309      IATIM=0
0310      IBTIM=0
0311      ICTIM=0
0312      GO TO 2701
0313      2700 CONTINUE
0314      WRITE(6,3006)
0315      WRITE(6,3005)
0316      2701 CONTINUE
0317      C111 INITIALIZE DATA TO DEFAULT VALUES.
0318      C$$$ TEST OUTPUT DEFAULT DATA TO:
0319      LDEBUG=.FALSE.
0320      LTEST=.FALSE.
0321      LOUT=.FALSE.
0322      LWARN=.TRUE.
0323      LSLOPE=.TRUE.
0324      Lcornr=.TRUE.
0325      LSOR=.FALSE.
0326      JMX(1)=1
0327      JMX(2)=6
0328      JMX(3)=5
0329      JMX(4)=4
0330      DO 2705 J=1,6
0331      DO 2705 K=1,4
0332      LKJ(K,J)=.FALSE.
0333      IF(J.LE.JMX(K)) LKJ(K,J)=.TRUE.
0334      2705 CONTINUE
0335      LKJ(3,4)=.FALSE.
0336      LKJ(3,5)=.FALSE.
0337      C$$$ FAR ZONE RANGE DEFAULT DATA RD:
0338      LRANG=.FALSE.
0339      RANG=1.
0340      C$$$ RANGE GATE DATA GR:
0341      RNIN=SMLT
0342      RMAX=BIG
0343      C$$$ POWER RADIATED DEFAULT DATA PR:

```


0344 LPRAD=.FALSE.
 0345 PRAD=0.
 0346 IPRAD=1
 0347 C\$\$\$ PATTERN DEFAULT DATA PD:, PN:, PF:, VD:, VF:, & VN:
 0348 LVOLP=.TRUE.
 0349 LFARN=.TRUE.
 0350 LNEAR=.FALSE.
 0351 LRECT=.FALSE.
 0352 LCNPAT=.TRUE.
 0353 TPPD=0.
 0354 TPPV=2.
 0355 TPPS=0.
 0356 TPP1=2.
 0357 THCZ=0.
 0358 PHCZ=0.
 0359 THCX=90.
 0360 PHCX=0.
 0361 VPC(1,1)=1.
 0362 VPC(1,2)=0.
 0363 VPC(1,3)=0.
 0364 VPC(2,1)=0.
 0365 VPC(2,2)=1.
 0366 VPC(2,3)=0.
 0367 VPC(1,3)=0.
 0368 VPC(2,3)=0.
 0369 VPC(3,3)=1.
 0370 XPC(1)=0.
 0371 XPC(2)=0.
 0372 XPC(3)=0.
 0373 RIS=1.
 0374 RXI=0.
 0375 TYS=0.
 0376 TYI=2.
 0377 PZS=0.
 0378 PZI=2.
 0379 IVPN=3
 0380 NPN=181
 0381 NPV=91
 0382 C\$\$\$ BACK OR BISTATIC NEAR ZONE SCATTERING DEFAULT DATA BP:
 0383 LSCAT=.FALSE.
 0384 C\$\$\$ FREQUENCY DEFAULT DATA FR: & FM:
 0385 FRQG=.2997925
 0386 LFQG=.FALSE.
 0387 FQGS=.2997925
 0388 FQGI=0.
 0389 NFQG=1
 0390 C\$\$\$ PLATE DEFAULT DATA PG:
 0391 LPLA=.FALSE.
 0392 MPX=0
 0393 MEP(1)=4
 0394 LSLAB(1)=0
 0395 XX(1,1,1)=1.
 0396 XX(2,1,1)=1.
 0397 XX(3,1,1)=0.
 0398 XX(1,2,1)=-1.
 0399 XX(2,2,1)=1.
 0400 XX(3,2,1)=0.
 0401 XX(1,3,1)=-1.
 0402 XX(2,3,1)=-1.
 0403 XX(3,3,1)=0.
 0404 XX(1,4,1)=1.
 0405 XX(2,4,1)=-1.
 0406 XX(3,4,1)=0.
 0407 C\$\$\$ GROUND PLANE DEFAULT DATA GP:

0408 LGRND=.FALSE.
 0409 MPXR=MPX
 0410 C\$\$\$ SOURCE DEFAULT DATA SG: ,SA: ,& SM:
 0411 LSMP=.FALSE.
 0412 MSX=0
 0413 MSAT=0
 0414 MSA(1,1)=0
 0415 MSA(2,1)=0
 0416 ISS(1,1)=0.
 0417 ISS(2,1)=0.
 0418 ISS(3,1)=0.
 0419 IMS(1)=-1
 0420 HS(1)=0.5
 0421 HAWS(1)=0.
 0422 THSZ=0.
 0423 PHSZ=0.
 0424 THSI=90.
 0425 PHSI=0.
 0426 VISS(1,1,1)=1.
 0427 VISS(1,2,1)=0.
 0428 VISS(1,3,1)=0.
 0429 VISS(2,1,1)=0.
 0430 VISS(2,2,1)=1.
 0431 VISS(2,3,1)=0.
 0432 VISS(3,1,1)=0.
 0433 VISS(3,2,1)=0.
 0434 VISS(3,3,1)=1.
 0435 WS(1)=(1.,0.)
 0436 C\$\$\$ RECEIVER DEFAULT DATA RG: ,RA: ,& RM:
 0437 LRCVR=.FALSE.
 0438 LRMP=.FALSE.
 0439 MRX=0
 0440 MRAT=0
 0441 MRA(1,1)=0
 0442 MRA(2,1)=0
 0443 XRR(1,1)=0.
 0444 XRR(2,1)=0.
 0445 XRR(3,1)=0.
 0446 IMR(1)=-1
 0447 HR(1)=0.5
 0448 HAWR(1)=0.
 0449 THRZ=0.
 0450 PHRZ=0.
 0451 THRX=90.
 0452 PHRX=0.
 0453 VIRR(1,1,1)=1.
 0454 VIRR(1,2,1)=0.
 0455 VIRR(1,3,1)=0.
 0456 VIRR(2,1,1)=0.
 0457 VIRR(2,2,1)=1.
 0458 VIRR(2,3,1)=0.
 0459 VIRR(3,1,1)=0.
 0460 VIRR(3,2,1)=0.
 0461 VIRR(3,3,1)=1.
 0462 WR(1)=(0.,0.)
 0463 C\$\$\$ LINE PRINTER DEFAULT DATA LP:
 0464 LWRITE=.FALSE.
 0465 C\$\$\$ PLOTTER DEFAULT DATA PP: & VP:
 0466 LVPLT=.FALSE.
 0467 LPLT=.FALSE.
 0468 LPPREC=.FALSE.
 0469 PPXL=0.
 0470 PPYL=3.
 0471 PPIXB=0.

```

0472      PPXE=360.
0473      PPXS=30.
0474      PPYB=-40.
0475      PPYE=0.
0476      PPYS=10.
0477  C$$$ ROTATE TRANSLATE DEFAULT DATA RT:
0478      THZP=0.
0479      PHZP=0.
0480      THXP=90.
0481      PHXP=0.
0482      TR(1)=0.
0483      TR(2)=0.
0484      TR(3)=0.
0485      VRT(1,1)=1.
0486      VRT(1,2)=0.
0487      VRT(1,3)=0.
0488      VRT(2,1)=0.
0489      VRT(2,2)=1.
0490      VRT(2,3)=0.
0491      VRT(3,1)=0.
0492      VRT(3,2)=0.
0493      VRT(3,3)=1.
0494  C$$$ CYLINDER DEFAULT DATA CO: & CC:
0495      MDC=0
0496      LCYL=.FALSE.
0497      MCX=0
0498      NEC(1)=2
0499      AC(1,1)=1.
0500      BC(1,1)=1.

0501      AC(2,1)=1.
0502      BC(2,1)=1.
0503      ZC(2,1)=-3.
0504      TCR(2,1)=1.570796
0505      ZC(1,1)=3.
0506      TCR(1,1)=1.570796
0507      VCL(1,1,1)=1.
0508      VCL(1,2,1)=0.
0509      VCL(1,3,1)=0.
0510      VCL(2,1,1)=0.
0511      VCL(2,2,1)=1.
0512      VCL(2,3,1)=0.
0513      VCL(3,1,1)=0.
0514      VCL(3,2,1)=0.
0515      VCL(3,3,1)=1.
0516      XCL(1,1)=0.
0517      XCL(2,1)=0.
0518      XCL(3,1)=0.
0519  C$$$ UNITS DEFAULT DATA UN: ,UF: & US:
0520      IUNIT=1
0521      UNITN=UNIT(IUNIT)
0522      UNITF=1.
0523      UNITS=UNITN+UNITF
0524      IUNST=0
0525      IUNSP=IUNST
0526      GO TO 2999
0527      ENTRY ABSCRE
0528  3000 CONTINUE
0529      WRITE(6,3006)
0530  3006 FORMAT(1X,1H+,76X,1H+)
0531      WRITE(6,3006)
0532      WRITE(6,3005)
0533  3005 FORMAT(1X,26(3H***))
0534  C111 READ IN VARIOUS COMMAND OPTIONS.
0535  2999 READ(5,3001,END=3004) (IR(I),I=1,36)

```

```

0536 3001 FORMAT(36A2)
0537 WRITE(6,3002)
0538 3002 FORMAT(1H ,////////,1X,26(3H**))
0539 WRITE(6,3006)
0540 WRITE(6,3003) (IR(I),I=1,36)
0541 3003 FORMAT(1X,1H*,2X,36A2,2X,1H*)
0542 C!!!
0543 C!!! CHECK AGAINST STORED OPTIONS
0544 C!!!
0545 C*** CM: COMMENT CARD
0546 IF(IR(1).EQ.IT(13)) GO TO 3090
0547 C*** CE: LAST COMMENT CARD
0548 IF(IR(1).EQ.IT(14)) GO TO 3000
0549 WRITE(6,3006)
0550 WRITE(6,3006)
0551 C*** TD: TEST DATA GENERATION OPTION.
0552 IF(IR(1).EQ.IT(1)) GO TO 3100
0553 C*** PD: FAR ZONE PATTERN INTEGER ANGLES
0554 IF(IR(1).EQ.IT(2)) GO TO 3200
0555 C*** RD: FAR ZONE RANGE INPUT
0556 IF(IR(1).EQ.IT(12)) GO TO 3250
0557 C*** PG: PLATE GEOMETRY INPUT
0558 IF(IR(1).EQ.IT(3)) GO TO 3300
0559 C*** SG: SOURCE GEOMETRY INPUT
0560 IF(IR(1).EQ.IT(4)) GO TO 3400
0561 C*** SM: SOURCE NEC DR AMP INPUT
0562 IF(IR(1).EQ.IT(11)) GO TO 3450
0563 C*** LP: LINE PRINTER LISTING OF RESULTS
0564 IF(IR(1).EQ.IT(5)) GO TO 3500
0565 C*** PP: PEN PLOT OF RESULTS
0566 IF(IR(1).EQ.IT(6)) GO TO 3600
0567 C*** GP: INCLUDE INFINITE GROUND PLANE
0568 IF(IR(1).EQ.IT(7)) GO TO 3700
0569 C*** XQ: EXECUTE PROGRAM
0570 IF(IR(1).EQ.IT(8)) GO TO 3800
0571 C*** RT: TRANSLATE AND/OR ROTATE COORDINATES
0572 IF(IR(1).EQ.IT(9)) GO TO 3900
0573 C*** CG: CYLINDER GEOMETRY INPUT
0574 IF(IR(1).EQ.IT(10)) GO TO 4000
0575 C*** CC: CONE GEOMETRY INPUT
0576 IF(IR(1).EQ.IT(40)) GO TO 4000
0577 C*** BP: BACK OR BISTATIC NEAR ZONE SCATTERING
0578 IF(IR(1).EQ.IT(15)) GO TO 6240
0579 C*** UF: SCALE FACTOR FOR INPUT
0580 IF(IR(1).EQ.IT(16)) GO TO 4120
0581 C*** UN: UNITS OF INPUT
0582 IF(IR(1).EQ.IT(18)) GO TO 4100
0583 C*** FR: FREQUENCY
0584 IF(IR(1).EQ.IT(19)) GO TO 4200
0585 C*** NX: NEXT PROBLEM
0586 IF(IR(1).EQ.IT(20)) GO TO 2700
0587 C*** EN: END PROGRAM
0588 IF(IR(1).EQ.IT(21)) GO TO 997
0589 C*** NP: NEXT SET OF PLATES
0590 IF(IR(1).EQ.IT(22)) GO TO 3350
0591 C*** NC: NEXT SET OF CYLINDERS
0592 IF(IR(1).EQ.IT(23)) GO TO 4050
0593 C*** NG: NO GROUND PLANE
0594 IF(IR(1).EQ.IT(24)) GO TO 3750
0595 C*** NS: NEXT SET OF SOURCES
0596 IF(IR(1).EQ.IT(25)) GO TO 3490
0597 C*** PR: POWER RADIATED INPUT
0598 IF(IR(1).EQ.IT(26)) GO TO 3440
0599 C*** US: UNITS OF HS AND HAWS IN SG: , SA: , RG: ,* RA:

```

```

0600         IF(IR(1).EQ.IT(27)) GO TO 4110
0601 C### PN: NEAR ZONE PATTERN DESIRED
0602         IF(IR(1).EQ.IT(28)) GO TO 3260
0603 C### RG: RECEIVER GEOMETRY INPUT
0604         IF(IR(1).EQ.IT(29)) GO TO 4400
0605 C### RM: RECEIVER NEC OR AMP INPUT
0606         IF(IR(1).EQ.IT(17)) GO TO 4450
0607 C### NR: NEXT SET OF RECEIVERS
0608         IF(IR(1).EQ.IT(30)) GO TO 3495
0609 C### SA: SOURCE ARRAY GEOMETRY INPUT
0610         IF(IR(1).EQ.IT(31)) GO TO 3810
0611 C### FM: MULTIPLE FREQUENCY INPUT
0612         IF(IR(1).EQ.IT(32)) GO TO 4250
0613 C### RA: RECEIVER ARRAY GEOMETRY INPUT
0614         IF(IR(1).EQ.IT(33)) GO TO 4810
0615 C### GR: RANGE GATE INPUT
0616         IF(IR(1).EQ.IT(34)) GO TO 5260
0617 C### VD: FAR ZONE VOLUMETRIC PATTERN INTEGER ANGLES
0618         IF(IR(1).EQ.IT(35)) GO TO 3210
0619 C### VN: NEAR ZONE VOLUMETRIC PATTERN
0620         IF(IR(1).EQ.IT(36)) GO TO 3270
0621 C### VP: VOLUMETRIC DUMP OF RESULTS FOR PLOTTING
0622         IF(IR(1).EQ.IT(37)) GO TO 3650
0623 C### PF: FAR ZONE NON INTEGER ANGLES
0624         IF(IR(1).EQ.IT(38)) GO TO 3220
0625 C### VF: FAR ZONE VOLUMETRIC PATTERN NON INTEGER ANGLES
0626         IF(IR(1).EQ.IT(39)) GO TO 3230
0627 C###
0628         WRITE(6,3021)
0629 3021 FORMAT(' *** PROGRAM ABORTS!!! COMMAND INPUT IS NOT PART',
0630 2' OF STORED COMMAND LIST ***')
0631 3004 STOP
0632 C=====
0633 3090 CONTINUE
0634 C=== CM: CE: COMMANDS =====
0635 C###
0636 C### IR(I)=CM: OR CE: FOLLOWED BY AN ALPHANUMERIC STRING OF
0637 C### CHARACTERS. THE CM: COMMAND IMPLIES THAT THERE WILL BE
0638 C### ANOTHER COMMENT CARD FOLLOWING IT. THE LAST COMMENT CARD
0639 C### MUST HAVE THE CE: COMMAND ON IT. IF THERE IS ONLY ONE
0640 C### COMMENT CARD THE CE: COMMAND SHOULD BE USED.
0641 C###
0642         READ(5,3001) (IR(I),I=1,36)
0643         WRITE(6,3003) (IR(I),I=1,36)
0644         IF(IR(1).EQ.IT(14)) GO TO 3000
0645         IF(IR(1).EQ.IT(13)) GO TO 3090
0646         WRITE(6,3091)
0647 3091 FORMAT(' *** PROGRAM ABORTS!!! CE: COMMAND MUST BE',
0648 2' USED TO END COMMENTS. ***')
0649         STOP
0650 C=====
0651 3100 CONTINUE
0652 C=== TO: COMMAND =====
0653 C###
0654 C### LDEBUG=DEBUG DATA OUTPUT ON LINE PRINTER(TRUE OR FALSE)
0655 C###
0656 C### LTEST=TEST DATA TO INSURE PROGRAM OPERATION(TRUE OR FALSE)
0657 C###
0658 C### LOUT=OUTPUT MAIN PROGRAM DATA ON LINE PRINTER(TRUE OR FALSE)
0659 C###
0660 C### LWARN=WARNING DATA OUTPUT ON LINE PRINTER(TRUE OR FALSE)
0661 C###
0662         READ(5,*) LDEBUG,LTEST,LOUT,LWARN
0663         WRITE(6,3101) LDEBUG,LTEST,LOUT,LWARN

```

```

0664 3101 FORMAT(2H *,5X,'LDEBUG= ',L3,5X,'LTEST= ',L3,5X,'LOUT=',L3
0665 2,5X,'LWARN=',L3,T79,1H+)
0666 WRITE(6,3006)
0667 C###
0668 C### LSLOPE=SLOPE DIFFRACTED FIELD DESIRED (T OR F)
0669 C###
0670 C### LCCORNR=CORNER DIFFRACTED FIELD DESIRED (T OR F)
0671 C###
0672 C### LSOR=ANTENNA SHADOW ALONE(TRUE OR FALSE)
0673 C###
0674 READ(6,*) LSLOPE,LCCORNR,LSOR
0675 WRITE(6,3102) LSLOPE,LCCORNR,LSOR
0676 3102 FORMAT(2H *,5X,'LSLOPE= ',L3,5X,'LCCORNR= ',L3,5X,'LSOR= ',L3,
0677 2T79,1H+)
0678 WRITE(6,3006)
0679 IF(LSOR) WRITE(6,3402)
0680 3402 FORMAT(2H *,5X,'SOURCE SHADOW ALONE IS COMPUTED!!!',T79,1H+)
0681 IF(LSOR) WRITE(6,3006)
0682 C###
0683 C### K=1,J=OPTION TO RUN DIRECT RAY TERM:
0684 C### 1=DIRECT FIELD
0685 C### NOTE: NORMALLY LKJ(1,1)=.TRUE. THIS COMPUTES THE INCIDENT FIELD.
0686 C###
0687 C### K=2,J=OPTION TO RUN VARIOUS RAY TERMS FOR PLATES:
0688 C### 1=SINGLE REFLECTED FIELD
0689 C### 2=DOUBLE REFLECTED FIELD
0690 C### 3=SINGLE DIFFRACTED FIELD
0691 C### 4=REFLECTED/DIFFRACTED FIELD
0692 C### 5=DIFFRACTED/REFLECTED FIELD
0693 C### 6=DOUBLE DIFFRACTION IDENTIFICATION
0694 C### NOTE: NORMALLY LKJ(2,1 TO 6)=.TRUE. THIS COMPUTES ALL FIELD
0695 C### VALUES INCLUDING IDENTIFYING DOUBLE DIFFRACTION PROBLEM AREAS
0696 C### FOR A CONVEX OR CONCAVE PLATE STRUCTURE.
0697 C###
0698 C### K=3,J=OPTION TO RUN VARIOUS RAY TERMS FOR CYLINDER:
0699 C### 1=REFLECTED,TRANSITION,AND CREEPING WAVE FIELDS
0700 C### 2=SINGLE REFLECTED FIELDS FROM ENDCAPS
0701 C### 3=SINGLE DIFFRACTED FIELDS FROM ENDCAP RIMS
0702 C### 4=REFLECTED-SCATTERED FIELDS FROM TWO PARALLEL CYLINDERS
0703 C### 5=DIFFRACTED-SCATTERED FIELDS FROM TWO PARALLEL CYLINDERS
0704 C### NOTE: NORMALLY LKJ(3,1 TO 5)=.TRUE. THIS COMPUTES ALL FIELD
0705 C### VALUES FOR A FINITE ELLIPTIC CYLINDER.
0706 C###
0707 C### K=4,J=OPTION TO RUN VARIOUS RAY TERMS FOR
0708 C### PLATE-CYLINDER INTERACTIONS:
0709 C### 1=FIELDS REFLECTED FROM THE PLATES THEN REFLECTED OR
0710 C### DIFFRACTED FROM THE CYLINDER
0711 C### 2=FIELDS REFLECTED OR DIFFRACTED FROM THE CYLINDER THEN
0712 C### REFLECTED FROM THE PLATES
0713 C### 3=FIELDS REFLECTED FROM THE CYLINDER THEN DIFFRACTED
0714 C### FROM THE PLATES
0715 C### 4=FIELDS DIFFRACTED FROM THE PLATES THEN REFLECTED
0716 C### FROM THE CYLINDER
0717 C### NOTE: NORMALLY LKJ(4,1 TO 4)=.TRUE. THIS COMPUTES ALL FIELD
0718 C### VALUES THAT INTERACT BETWEEN THE PLATES AND CYLINDERS.
0719 C###
0720 DO 3104 K=1,4
0721 JK=JMX(K)
0722 READ(6,*) (LKJ(K,J),J=1,JK)
0723 3104 WRITE(6,3103) K,(LKJ(K,J),J=1,JK)
0724 3103 FORMAT(2H *,T79,1H+,T8,'LKJ(',I1,',',J)= ',6L2)
0725 GO TO 3000
0726 C=====
0727 4100 CONTINUE

```

```

0728 C=== UN:      COMMAND      *****
0729 C###
0730 C### IUNIT=INDICATOR OF UNITS USED FOR INPUT DATA.
0731 C###          1=METERS
0732 C###          2=FEET
0733 C###          3=INCHES
0734 C###
0735 READ(6,*) IUNIT
0736 UNITN=UNIT(IUNIT)
0737 UNITS=UNITN*UNITF
0738 WRITE(6,4101) LABEL(IUNIT)
0739 4101 FORMAT(2H *,5X,'ALL THE LINEAR DIMENSIONS BELOW ARE'
0740 2,' ASSUMED TO BE IN ',A6,T99,1H*)
0741 GO TO 3000
0742 C=====
0743 4120 CONTINUE
0744 C=== UF:      COMMAND *****
0745 C###
0746 C### UNITF = SCALE FACTOR FOR GEOMETRY
0747 C###
0748 READ(6,*) UNITF
0749 UNITS=UNITN*UNITF
0750 WRITE(6,4121) UNITF

0751 4121 FORMAT(2H *,5X,'ALL THE LINEAR DIMENSIONS BELOW ARE SCALED BY'
0752 2,' A FACTOR OF ',F12.5,T99,1H*)
0753 GO TO 3000
0754 C=====
0755 4110 CONTINUE
0756 C=== US:      COMMAND      *****
0757 C###
0758 C### IUNST=INDICATOR OF UNITS USED FOR HS AND HAWS IN THE
0759 C### SG: COMMAND.
0760 C###          0=WAVELENGTHS
0761 C###          1=METERS
0762 C###          2=FEET
0763 C###          3=INCHES
0764 C###
0765 C### NOTE: IF ONE SOURCE IS SPECIFIED IN WAVELENGTHS, THEY ALL
0766 C### MUST BE IN WAVELENGTHS.
0767 READ(6,*) IUNST
0768 IF(MSX.EQ.0) GO TO 4112
0769 IF(IUNST.EQ.0.AND.IUNSP.EQ.0) GO TO 4112
0770 IF(IUNST.NE.0.AND.IUNSP.NE.0) GO TO 4112
0771 WRITE(6,4111)
0772 4111 FORMAT(' *** PROGRAM ABORTS IN SOURCE UNITS. ALL UNITS NOT'
0773 2,' SPECIFIED IN WAVELENGTHS!!! ***')
0774 STOP
0775 4112 CONTINUE
0776 IF(IUNST.EQ.0) GO TO 4114
0777 WRITE(6,4113) LABEL(IUNST)
0778 4113 FORMAT(2H *,5X,'THE SOURCE LENGTH HS AND WIDTH HAWS ARE'
0779 2,' ASSUMED TO BE IN ',A6,T99,1H*)
0780 GO TO 4116
0781 4114 WRITE(6,4115)
0782 4115 FORMAT(2H *,5X,'THE SOURCE LENGTH HS AND WIDTH HAWS ARE'
0783 2,' ASSUMED TO BE IN WAVELENGTHS',T99,1H*)
0784 4116 IUNSP=IUNST
0785 GO TO 3000
0786 C=====
0787 4200 CONTINUE
0788 C=== FR:      COMMAND      *****
0789 C###
0790 C### FRQG=FREQUENCY IN GIGAHERTZ.
0791 C###

```

```

0792         LFQG=.FALSE.
0793         NFQG=1
0794         READ(6,*) FRQG
0795         WL=.2997925/FRQG
0796         WRITE(6,4201) FRQG
0797 4201  FORMAT(2H *,5X,'FREQUENCY= ',F7.3,' GIGAHERTZ',T79,1H*)
0798         WRITE(6,3006)
0799         WRITE(6,4202) WL
0800 4202  FORMAT(2H *,5X,'WAVELENGTH= ',F10.6,' METERS',T79,1H*)
0801         GO TO 3000
0802 C=====
0803 4250  CONTINUE
0804 C=== FM:      COMMAND      =====
0805 C$$$
0806 C$$$ NFQG=NUMBER OF FREQUENCIES DESIRED
0807 C$$$
0808 C$$$ FQGS=STARTING FREQUENCY IN GIGAHERTZ
0809 C$$$
0810 C$$$ FQGI=INCREMENTAL FREQUENCY CHANGE IN GIGAHERTZ
0811 C$$$
0812 C$$$ NOTE:   THE SOURCE LENGTH AND WIDTH MUST NOT BE SPECIFIED
0813 C$$$         IN WAVELENGTHS.  ALSO ONLY ONE PATTERN LOCATION
0814 C$$$         CAN BE SPECIFIED.
0815 C$$$
0816         LFQG=.TRUE.
0817         READ(6,*) NFQG,FQGS,FQGI
0818         WRITE(6,4251) NFQG
0819 4251  FORMAT(2H *,5X,I3,' FREQUENCIES ARE SPECIFIED',T79,1H*)
0820         IF(NFQG.GT.MODX) WRITE(6,3286) NFQG
0821         IF(NFQG.GT.MODX) STOP
0822         WRITE(6,3006)
0823         WRITE(6,4252) FQGS,FQGI
0824 4252  FORMAT(2H *,5X,'STARTING FREQ.= ',F10.6,' IN STEPS OF ',F10.6
0825         2,' GHZ.',T79,1H*)
0826 C111  CALCULATE MID-FREQUENCY
0827         FRQG=FQGS+0.5*FQGI*(NFQG-1)
0828         WL=.2997925/FRQG
0829         GO TO 3000
0830 C=====
0831 3230  CONTINUE
0832 C=== VF:      COMMAND      =====
0833 C$$$
0834 C$$$ FAR ZONE VOLUMETRIC PATTERN NON INTEGER ANGLES
0835 C$$$
0836         LVOLP=.TRUE.
0837         LFARN=.TRUE.
0838         GO TO 3211
0839 C=====
0840 3210  CONTINUE
0841 C=== VD:      COMMAND      =====
0842 C$$$
0843 C$$$ FAR ZONE VOLUMETRIC PATTERN INTEGER ANGLES
0844 C$$$
0845         LVOLP=.TRUE.
0846         LFARN=.FALSE.
0847         GO TO 3211
0848 C=====
0849 3220  CONTINUE

0850 C=== PF:      COMMAND      =====
0851 C$$$
0852 C$$$ FAR ZONE PATTERN NON INTEGER ANGLES
0853 C$$$
0854         LVOLP=.FALSE.
0855         LFARN=.TRUE.

```



```

0856          GO TO 3211
0857          C-----
0858          3200 CONTINUE
0859          C--- PD:      COMMAND      -----
0860          C$$$
0861          C$$$ FAR ZONE PATTERN INTEGER ANGLES
0862          C$$$
0863          C$$$ THCZ,PHCZ=ORIENTATION OF THE Z AXIS RELATIVE TO THE
0864          C$$$ FIXED COORDINATE SYSTEM.
0865          C$$$
0866          C$$$ THCX,PHCX=ORIENTATION OF THE X AXIS RELATIVE TO THE
0867          C$$$ FIXED COORDINATE SYSTEM
0868          C$$$
0869          LVDLP=.FALSE.
0870          3211 LNEAR=.FALSE.
0871          READ(6,*) THCZ,PHCZ,THCX,PHCX
0872          VPC(3,1)=SIN(THCZ*RPD)*COS(PHCZ*RPD)
0873          VPC(3,2)=SIN(THCZ*RPD)*SIN(PHCZ*RPD)
0874          VPC(3,3)=COS(THCZ*RPD)
0875          VPC(1,1)=SIN(THCX*RPD)*COS(PHCX*RPD)
0876          VPC(1,2)=SIN(THCX*RPD)*SIN(PHCX*RPD)
0877          VPC(1,3)=COS(THCX*RPD)
0878          C!!! INSURE VPC(1,N) IS PERPENDICULAR TO VPI(3,N)
0879          DZX=VPC(3,1)*VPC(1,1)+VPC(3,2)*VPC(1,2)+VPC(3,3)*VPC(1,3)
0880          IF(ABS(DZX).GT.0.1) WRITE(6,3201)
0881          3201 FORMAT(' *** PROGRAM ABORTS IN PATTERN CUT SECTION.'
0882          2,' THE COORDINATES ARE NOT ORTHOGONAL!!! ***')
0883          IF(ABS(DZX).GT.0.1) STOP
0884          VPC(1,1)=VPC(1,1)-VPC(3,1)*DZX
0885          VPC(1,2)=VPC(1,2)-VPC(3,2)*DZX
0886          VPC(1,3)=VPC(1,3)-VPC(3,3)*DZX
0887          DOT=VPC(1,1)*VPC(1,1)+VPC(1,2)*VPC(1,2)+VPC(1,3)*VPC(1,3)
0888          DOT=SQRT(DOT)
0889          VPC(1,1)=VPC(1,1)/DOT
0890          VPC(1,2)=VPC(1,2)/DOT
0891          VPC(1,3)=VPC(1,3)/DOT
0892          VPC(2,1)=VPC(3,2)*VPC(1,3)-VPC(3,3)*VPC(1,2)
0893          VPC(2,2)=VPC(3,3)*VPC(1,1)-VPC(3,1)*VPC(1,3)
0894          VPC(2,3)=VPC(3,1)*VPC(1,2)-VPC(3,2)*VPC(1,1)
0895          WRITE(6,3202)
0896          3202 FORMAT(2H *,5X,'THE PATTERN AXES ARE AS FOLLOWS:',T79,1H*)
0897          DO 3204 NI=1,3
0898          WRITE(6,3006)
0899          3204 WRITE(6,3205) (NI,NJ,VPC(NI,NJ),NJ=1,3)
0900          3205 FORMAT(2H *,1X,3(2X,'VPC(',I1,',',',I1,')='),F9.5),T79,1H*)
0901          DO 3203 N=1,3
0902          3203 IPC(N)=0.
0903          C$$$
0904          C$$$ LCNPAT=IS PATTERN CONIC CUT(T OR F)?
0905          C$$$ T=THETA CUT(CONIC CUT)
0906          C$$$ F=PHI CUT(PHI CONSTANT)
0907          C$$$
0908          C$$$ TPPD=PATTERN ANGLE THAT IS CONSTANT
0909          C$$$ IF LCNPAT=T: TPPD=THP CONSTANT
0910          C$$$ IF LCNPAT=F: TPPD=PHP CONSTANT
0911          C$$$
0912          IF(LVOLDP) GO TO 3212
0913          TPPV=0.
0914          NPV=1
0915          READ(6,*) LCNPAT,TPPD
0916          WRITE(6,3006)
0917          IF(.NOT.LCNPAT) WRITE(6,3206) TPPD
0918          3206 FORMAT(2H *,5X,'THETA IS BEING VARIED WITH PHI='),F10.6
0919          2,T79,1H*)

```

```

0920      IF(LCNPAT) WRITE(6,3207) TPPD
0921 3207  FORMAT(2H *,5X,'PHI IS BEING VARIED WITH THETA= ',F10.5
0922      2,I79,1H*)
0923      WRITE(6,3006)
0924      GO TO 3216
0925  C###
0926  C###  TPPD=START OF VOLUMETRIC PATTERN ANGLE
0927  C###  TPPV=INCREMENT FOR VOLUMETRIC PATTERN ANGLE
0928  C###  NPV=NUMBER OF VOLUMETRIC PATTERN ANGLES
0929  C###
0930 3212  READ(5,*) LCNPAT,TPPD,TPPV,NPV
0931      WRITE(6,3006)
0932      IF(LCNPAT) WRITE(6,3213)
0933 3213  FORMAT(2H *,5X,'FOR THETA ANGLE:',I79,1H*)
0934      IF(.NOT.LCNPAT) WRITE(6,3214)
0935 3214  FORMAT(2H *,5X,'FOR PHI ANGLE:',I79,1H*)
0936      WRITE(6,3215) TPPD,TPPV,NPV
0937 3215  FORMAT(2H *,5X,'START= ',F10.5,' STEP= 'F10.5,' NUMBER= ',I4
0938      2,I79,1H*)
0939      WRITE(6,3006)
0940      IF(LCNPAT) WRITE(6,3214)
0941      IF(.NOT.LCNPAT) WRITE(6,3213)
0942 3216  CONTINUE
0943      IF(LFARN) GO TO 3217
0944  C###
0945  C###  IB,IE,IS=BEGIN,END,STEP
0946  C###
0947      READ(5,*) IB,IE,IS
0948      IF(IB.LT.0) IB=0
0949      IF(IE.GT.360) IE=360
0950      IF(IS.LE.0) IS=1
0951      TPPS=IB
0952      TPPI=IS
0953      NPN=(IE-IB)/IS+1
0954      WRITE(6,3208) IB,IE,IS
0955 3208  FORMAT(2H *,5X,'THE RANGE OF PATTERN ANGLE INDICES FOR THIS'
0956      2,' RUN ARE: ',I3,2(' ',I3),I79,1H*)
0957      GO TO 3218
0958 3217  CONTINUE
0959  C###
0960  C###  TPPS=START OF PATTERN
0961  C###  TPPI=PATTERN INCREMENT
0962  C###  NPN=NUMBER OF PATTERN POINTS
0963  C###
0964      READ(5,*) TPPS,TPPI,NPN
0965      WRITE(6,3215) TPPS,TPPI,NPN
0966 3218  CONTINUE
0967      RXS=1.
0968      RXI=0.
0969      TYS=TPPD
0970      TYI=TPPV
0971      PZS=TPPS
0972      PZI=TPPI
0973      IVPH=3
0974      IF(LCNPAT) GO TO 3209
0975      TYS=TPPS
0976      TYI=TPPI
0977      PZS=TPPD
0978      PZI=TPPV
0979      IVPH=-3
0980 3209  CONTINUE
0981      GO TO 3000
0982  C=====
0983 3250  CONTINUE

```

```

0984 C=== RD:    COMMAND    =====
0985 C$$$
0986 C$$$ RANGS=FAR FIELD RANGE DISTANCE
0987 C$$$
0988 C$$$ NOTE IF RANGS IS GREATER THAN OR EQUAL TO 1.E30
0989 C$$$ THAN LRANG WILL BE SET FALSE
0990 C$$$
0991 LRANG=.TRUE.
0992 READ(5,*) RANGS
0993 IF(RANGS.GT.9.9E29) GO TO 3252
0994 RANG=UNITS*RANGS
0995 WRITE(6,3251) RANGS,LABEL(IUNIT),RANG
0996 3251 FORMAT(2H *,5X,'THE FAR FIELD RANGE SPECIFIED IS ',E12.6,
0997 2' IN ',A6,I79,1H*,/2H *,5X,'THE RANGE SPECIFIED IN METERS'
0998 3,' IS ',E12.6,I79,1H*)
0999 GO TO 3000
1000 3252 CONTINUE
1001 LRANG=.FALSE.
1002 RANG=1.
1003 WRITE(6,3253)
1004 3253 FORMAT(2H *,5X,'NO FAR FIELD RANGE SPECIFIED.',I79,1H*)
1005 GO TO 3000
1006 C=====
1007 3270 CONTINUE
1008 C=== VN:    COMMAND    =====
1009 C$$$
1010 C$$$ NEAR ZONE VOLUMETRIC PATTERN
1011 C$$$
1012 LVOLP=.TRUE.
1013 GO TO 3271
1014 C=====
1015 3260 CONTINUE
1016 C=== PN:    COMMAND    =====
1017 C$$$
1018 C$$$ XPC(N)=XYZ LOCATION OF THE NEAR ZONE PATTERN ORIGIN
1019 C$$$
1020 LVOLP=.FALSE.
1021 3271 LNEAR=.TRUE.
1022 READ(5,*) (XPC(N),N=1,3)
1023 WRITE(6,3254) LABEL(IUNIT),(XPC(N),N=1,3)
1024 3254 FORMAT(2H *,1X,'PATTERN ORIGIN IN ',A6,' : XPC(1)=' ,F8.3
1025 2,' XPC(2)=' ,F8.3,' XPC(3)=' ,F8.3,I79,1H*)
1026 WRITE(6,3006)
1027 DO 3263 N=1,3
1028 3263 XPC(N)=UNITS*XPC(N)
1029 IF(IUNIT.NE.1) WRITE(6,3254) LABEL(1),(XPC(N),N=1,3)
1030 IF(IUNIT.NE.1) WRITE(6,3006)
1031 WRITE(6,3006)
1032 C$$$
1033 C$$$ THCZ,PHCZ=ORIENTATION OF THE Z-AXIS OF THE PATTERN AXES
1034 C$$$ RELATIVE TO THE FIXED COORDINATE SYSTEM
1035 C$$$
1036 C$$$ THCX,PHCX=ORIENTATION OF THE X-AXIS OF THE PATTERN AXES
1037 C$$$ RELATIVE TO THE FIXED COORDINATE SYSTEM
1038 C$$$
1039 READ(5,*) THCZ,PHCZ,THCX,PHCX
1040 VPC(3,1)=SIN(THCZ*RPD)*COS(PHCZ*RPD)
1041 VPC(3,2)=SIN(THCZ*RPD)*SIN(PHCZ*RPD)
1042 VPC(3,3)=COS(THCZ*RPD)
1043 VPC(1,1)=SIN(THCX*RPD)*COS(PHCX*RPD)
1044 VPC(1,2)=SIN(THCX*RPD)*SIN(PHCX*RPD)
1045 VPC(1,3)=COS(THCX*RPD)
1046 C!!! INSURE VPC(1,N) IS PERPENDICULAR TO VPC(3,N)
1047 DZI=VPC(3,1)*VPC(1,1)+VPC(3,2)*VPC(1,2)+VPC(3,3)*VPC(1,3)

```

```

1048      IF(ABS(DZX).GT.0.1) WRITE(6,3201)
1049      IF(ABS(DZX).GT.0.1) STOP
1050      VPC(1,1)=VPC(1,1)-VPC(3,1)*DZX
1051      VPC(1,2)=VPC(1,2)-VPC(3,2)*DZX
1052      VPC(1,3)=VPC(1,3)-VPC(3,3)*DZX
1053      DOT=VPC(1,1)*VPC(1,1)+VPC(1,2)*VPC(1,2)+VPC(1,3)*VPC(1,3)
1054      DOT=SQRT(DOT)
1055      VPC(1,1)=VPC(1,1)/DOT
1056      VPC(1,2)=VPC(1,2)/DOT
1057      VPC(1,3)=VPC(1,3)/DOT
1058      VPC(2,1)=VPC(3,2)*VPC(1,3)-VPC(3,3)*VPC(1,2)
1059      VPC(2,2)=VPC(3,3)*VPC(1,1)-VPC(3,1)*VPC(1,3)
1060      VPC(2,3)=VPC(3,1)*VPC(1,2)-VPC(3,2)*VPC(1,1)
1061      WRITE(6,3202)
1062      DO 3264 NI=1,3
1063      WRITE(6,3006)
1064      3264 WRITE(6,3205) (NI,NJ,VPC(NI,NJ),NJ=1,3)
1065      WRITE(6,3006)
1066      WRITE(6,3006)
1067      C$$$
1068      C$$$ LRECT=F, SPHERICAL PATTERN CUT
1069      C$$$ LRECT=T, LINEAR PATTERN CUT
1070      C$$$
1071      C$$$ RXS,TYS,PZS=STARTING LOCATION OF PATTERN
1072      C$$$           LRECT=F: RADIAL,THETA,PHI
1073      C$$$           LRECT=T: X,Y,Z
1074      C$$$
1075      C$$$ RXI,TYI,PZI=SIZE OF INCREMENTAL STEPS
1076      C$$$           LRECT=F: RADIAL,THETA,PHI
1077      C$$$           LRECT=T: X,Y,Z
1078      C$$$
1079      READ(6,*) LRECT
1080      READ(6,*) RXS,TYS,PZS
1081      READ(6,*) RXI,TYI,PZI
1082      IF(LRECT) WRITE(6,3261) RXS,TYS,PZS,LABEL(IUNIT)
1083      3261 FORMAT(2H *,2X,'STARTING XYZ=',F10.5,2(' ',F10.5),1X,A6
1084      2,T79,1H*)
1085      IF(LRECT) WRITE(6,3262) RXI,TYI,PZI,LABEL(IUNIT)
1086      3262 FORMAT(2H *,2X,'STEP XYZ=',F10.5,2(' ',F10.5),1X,A6,T79,1H*)
1087      IF(.NOT.LRECT) WRITE(6,3267) RXS,TYS,PZS,LABEL(IUNIT)
1088      3267 FORMAT(2H *,2X,'STARTING R,THETA,PHI=',F10.5
1089      2,2(' ',F10.5),1X,A6,' AND DEG.',T79,1H*)
1090      IF(.NOT.LRECT) WRITE(6,3268) RXI,TYI,PZI,LABEL(IUNIT)
1091      3268 FORMAT(2H *,2X,'STEP R,THETA,PHI=',F10.5,2(' ',F10.5),1X,A6
1092      2,' AND DEG.',T79,1H*)
1093      WRITE(6,3006)
1094      RXS=UNITS*RXS
1095      RXI=UNITS*RXI
1096      IF(.NOT.LRECT) GO TO 3265
1097      TYS=UNITS*TYS
1098      PZS=UNITS*PZS
1099      TYI=UNITS*TYI
1100      PZI=UNITS*PZI
1101      3265 CONTINUE
1102      IF(LRECT.AND.IUNIT.NE.1) WRITE(6,3261) RXS,TYS,PZS,LABEL(1)
1103      IF(LRECT.AND.IUNIT.NE.1) WRITE(6,3262) RXI,TYI,PZI,LABEL(1)
1104      IF(.NOT.LRECT.AND.IUNIT.NE.1) WRITE(6,3267) RXS,TYS,PZS,LABEL(1)
1105      IF(.NOT.LRECT.AND.IUNIT.NE.1) WRITE(6,3268) RXI,TYI,PZI,LABEL(1)
1106      IF(.NOT.LRECT.AND.IUNIT.NE.1) WRITE(6,3006)
1107      IF(LVOLP) GO TO 3272
1108      C$$$
1109      C$$$ NPN=NUMBER OF PATTERN POINTS
1110      C$$$
1111      READ(6,*) NPN

```

```

1112 WRITE(6,3269) NPN
1113 3269 FORMAT(2H *,5X,'NUMBER OF PATTERN POINTS= ',I4,T79.1H*)
1114 IVPN=3
1115 IF(ABS(PZI).LT.SMLR) IVPN=-3
1116 IF(LRECT) IVPN=0
1117 GO TO 3276
1118 C###
1119 C### IVPN=1 FOR R-THETA OR X-Y VARYING
1120 C### NPV=NUMBER OF R OR X AND NPN=NUMBER OF THETA OR Y
1121 C###
1122 C### IVPN=2 FOR R-PHI OR X-Z VARYING
1123 C### NPV=NUMBER OF R OR X AND NPN=NUMBER OF PHI OR Z
1124 C###
1125 C### IVPN=3 FOR THETA-PHI OR Y-Z VARYING
1126 C### NPV=NUMBER OF THETA OR Y AND NPN=NUMBER OF PHI OR Z
1127 C###
1128 C### IF IVPN IS LESS THAN ZERO THE ORDER IS REVERSED
1129 C### I.E. IVPN=-1 FOR THETA-R OR Y-X VARYING
1130 C###
1131 3272 READ(6,*) IVPN,NPV,NPN
1132 IF(IVPN.EQ.1) WRITE(6,3273) NPV,NPN
1133 IF(IVPN.EQ.-1) WRITE(6,3273) NPN,NPV
1134 3273 FORMAT(2H *,5X,'NUMBER OF POINTS FOR R OR X= ',I4
1135 2,' AND THETA OR Y= ',I4)
1136 IF(IVPN.EQ.2) WRITE(6,3274) NPV,NPN
1137 IF(IVPN.EQ.-2) WRITE(6,3274) NPN,NPV
1138 3274 FORMAT(2H *,5X,'NUMBER OF POINTS FOR R OR X= ',I4
1139 2,' AND PHI OR Z= ',I4)
1140 IF(IVPN.EQ.3) WRITE(6,3275) NPV,NPN
1141 IF(IVPN.EQ.-3) WRITE(6,3275) NPN,NPV
1142 3275 FORMAT(2H *,5X,'NUMBER OF POINTS FOR THETA OR Y= ',I4
1143 2,' AND PHI OR Z= ',I4)
1144 3276 CONTINUE
1145 IF(NPN.GI.MODX) WRITE(6,3266) NPN
1146 3266 FORMAT(' ***** NUMBER OF POINTS= ',I3,' PROGRAM ABORTS'
1147 2,'PATTERN STORAGE DIMENSION IS EXCEEDED *****')
1148 IF(NPN.GI.MODX) STOP
1149 GO TO 3000
1150 C=====
1151 5240 CONTINUE
1152 C=== BP: COMMAND =====
1153 C###
1154 C### BACK OR BISTATIC NEAR ZONE SCATTERING
1155 C###
1156 C### THE SG:, RG:, AND PN: COMMANDS MUST BE SPECIFIED
1157 C### TO USE THIS OPTION.
1158 LSCAT=.TRUE.
1159 GO TO 3000
1160 C=====
1161 5260 CONTINUE
1162 C=== GR: COMMAND =====
1163 C###
1164 C### RANGE GATE INPUT
1165 C###
1166 C### RMIN=THE MINIMUM DISTANCE FROM TRANSMITTER TO RECEIVER
1167 C### RMAX=THE MAXIMUM DISTANCE FROM TRANSMITTER TO RECEIVER
1168 C###
1169 C### THE PN: COMMAND MUST BE USED
1170 C###
1171 READ(6,*) RMIN,RMAX
1172 WRITE(6,5261) RMIN,RMAX,LABEL(IUNIT)
1173 5261 FORMAT(2H *,2X,'RMIN= ',F10.5,'RMAX= ',F10.5,' IN ',A6,T79.1H*)
1174 RMIN=UNITS*RMIN
1175 RMAX=UNITS*RMAX

```

```

1176 WRITE(6,5261) RMIN,RMAX,LABEL(1)
1177 GO TO 3000
1178 C=====
1179 3300 CONTINUE
1180 C=== PG: COMMAND =====
1181 C$$$
1182 C$$$ PLATE GEOMETRY INPUT
1183 C$$$
1184 LPLA=.TRUE.
1185 MPX=MPX+1
1186 IF(MPX.GT.MPDX) WRITE(6,901) MPX
1187 901 FORMAT(' ***** NUMBER OF PLATES= ',I3,' PROGRAM ABORTS',
1188 2' SINCE MAX. PLATE DIMENSION IS EXCEEDED. *****')
1189 IF(MPX.GT.MPDX) STOP
1190 WRITE(6,3301) MPX
1191 3301 FORMAT(2H *,5X,'THIS IS PLATE NO. ',I3,' IN THIS ',
1192 2'SIMULATION.',I79,1H*)
1193 MP=MPX
1194 WRITE(6,3006)
1195 WRITE(6,3006)
1196 WRITE(6,3006)
1197 C$$$
1198 C$$$ MEP(MP)=NUMBER OF CORNERS ON THE MP-TH PLATE.
1199 C$$$

1200 C$$$ LSLAB= 1 IMPLIES TRANSPARENT THIN DIELECTRIC SLAB
1201 C$$$ = 0 IMPLIES METAL PLATE, AND
1202 C$$$ =-2 IMPLIES DIELECTRIC COVERED PLATE ON BOTH SIDES
1203 C$$$ =-4 IMPLIES DIELECTRIC COVERED PLATE ON SIDE OF NORMAL
1204 C$$$
1205 C$$$ NOTE: IF DIELECTRIC COVERED, ONE MUST READ DIELECTRIC DATA.
1206 C$$$
1207 C$$$
1208 READ(6,*) MEP(MP), LSLAB(MP)
1209 IF(LSLAB(MP).EQ.0) WRITE(6,3392)
1210 3392 FORMAT(2H *,5X,'METAL PLATE USED IN THIS SIMULATION',I79,1H*)
1211 IF(LSLAB(MP).EQ.1) WRITE(6,3393)
1212 3393 FORMAT(2H *,5X,'TRANSPARENT THIN DIELECTRIC LAYER USED IN THIS',
1213 2'SIMULATION',I79,1H*)
1214 IF(LSLAB(MP).EQ.-2) WRITE(6,3394)
1215 3394 FORMAT(2H *,5X,'DIELECTRIC COVERED PLATE USED IN THIS',
1216 2' SIMULATION',I79,1H*)
1217 WRITE(6,3006)
1218 IF(LSLAB(MP).EQ.0) GO TO 3313
1219 C$$$
1220 C$$$ NSLAB(MP)=NUMBER OF DIELECTRIC LAYERS ON THE MP PLATE
1221 C$$$
1222 READ(6,*) NSLAB(MP)
1223 NSS=NSLAB(MP)
1224 IF(NSS.GT.MLDX) STOP
1225 WRITE(6,3391)
1226 3391 FORMAT(2H *,13X,'THICKNESS',2X,'DIELECTRIC',3X,'LOSS',4X,
1227 2'PERMITTIVITY',3X,'LOSS',I79,1H+,'/',
1228 32H *,5X,'LAYER#',2X,'IN METERS',3X,'CONSTANT',3X,'TANGENT',
1229 44X,'CONSTANT',3X,'TANGENT',I79,1H+,'/',
1230 52H *,5X,'-----',2X,'-----',2X,'-----',2X,'-----',
1231 62X,'-----',2X,'-----',I79,1H+)
1232 C$$$
1233 C$$$ DSLAB(NS,MP)=THICKNESS OF NS LAYER
1234 C$$$
1235 C$$$ ERSLAB(NS,MP)=RELATIVE DIELECTRIC CONSTANT OF THE NS LAYER
1236 C$$$
1237 C$$$ TESLAB(NS,MP)=DIELECTRIC LOSS TANGENT OF THE NS LAYER
1238 C$$$
1239 C$$$ URSLAB(NS,MP)=RELATIVE PERMEABILITY CONSTANT OF THE NS LAYER

```

```

1240 C###
1241 C### TMSLAB(NS,MP)=PERMEABILITY LOSS TANGENT OF THE NS LAYER
1242 C###
1243 DO 3312 NS=1,NSS
1244 READ(6,*) DSLAB(NS,MP),ERSLAB(NS,MP),TESLAB(NS,MP),
1245 2URSLAB(NS,MP),TMSLAB(NS,MP)
1246 DSLAB(NS,MP)=DSLAB(NS,MP)*UNITS
1247 3312 WRITE(6,3399) NS,DSLAB(NS,MP),ERSLAB(NS,MP),TESLAB(NS,MP),
1248 2URSLAB(NS,MP),TMSLAB(NS,MP)
1249 3399 FORMAT(2H *,6X,I3,4X,F9.4,2X,F10.4,2X,F7.4,2X,F11.4,2X,
1250 2F7.4,T79,1H*)
1251 WRITE(6,3006)
1252 WRITE(6,3006)
1253 WRITE(6,3006)
1254 3313 MEX=MEP(MP)
1255 IF(MEX.GT.MEDX) WRITE(6,903) MP,MEX
1256 903 FORMAT(' ***** PLATE #',I3,' HAS ',I3,' EDGES.',
1257 2' PROGRAM ABORTS SINCE MAX. EDGE DIMENSION IS EXCEEDED.'
1258 3,' *****')
1259 IF(MEX.GT.MEDX) STOP
1260 DO 5 ME=1,MEX
1261 C###
1262 C### IX(N,ME,MP)=X,Y,Z COMPONENTS OF CORNER #ME OF PLATE #MP.
1263 C### N=1(X),N=2(Y),N=3(Z). INPUT CORNER DATA AS FOLLOWS:
1264 C### 1.,1.,0.
1265 C### -1.,1.,0.
1266 C### -1.,-1.,0.
1267 C### 1.,-1.,0.
1268 C### THIS IS THE INPUT FOR A 2 METER SQUARE PLATE.
1269 C### NOTE THAT IF THERE IS MORE THAN ONE PLATE, THEN THE CORNER
1270 C### DATA FOR EACH PLATE WOULD FOLLOW SEQUENTIALLY.
1271 C###
1272 READ(6,*) (IX(N,ME,MP),N=1,3)
1273 5 CONTINUE
1274 WRITE(6,3302) LABEL(IUNIT)
1275 3302 FORMAT(2H *,2X,'PLATE#',2X,'CORNER#',3X,'INPUT LOCATION IN ',
1276 2A6,4X,'ACTUAL LOCATION IN METERS',T79,1H*)
1277 WRITE(6,3303)
1278 3303 FORMAT(2H *,2X,'-----',2X,'-----'
1279 2,2(2X,2('-----')),T79,1H*)
1280 DO 3304 ME=1,MEX
1281 WRITE(6,3006)
1282 DO 3310 N=1,3
1283 3310 IQ(N)=XX(N,ME,MP)
1284 DO 3311 N=1,3
1285 3311 IX(N,ME,MP)=UNITS*(IQ(1)*VRT(1,N)+IQ(2)*VRT(2,N)
1286 2*IQ(3)*VRT(3,N))+TR(N)
1287 WRITE(6,3305) MP,ME,(IQ(N),N=1,3),(IX(N,ME,MP),N=1,3)
1288 3305 FORMAT(2H *,4X,I3,6X,I2,2X,2(2X,F8.3,2(' ',F8.3)),T79,1H*)
1289 3304 CONTINUE
1290 GO TO 3000
1291 C=====
1292 3350 CONTINUE
1293 C=== NP: COMMAND =====
1294 C###
1295 C### INITIALIZE PLATE DATA.
1296 C###
1297 LPLA=.FALSE.
1298 MPX=0
1299 WRITE(6,3351)
1300 3351 FORMAT(2H *,5X,' THE PLATE DATA IS INITIALIZED. ',T79,1H*/
1301 2,2H *,5X,' NO PLATES ARE PRESENTLY IN THE PROBLEM. ',T79,1H*)
1302 GO TO 3000
1303 C=====

```

```

1304 3400 CONTINUE
1305 C=== SC:      COMMAND      =====
1306 C###
1307 C### MSX=NUMBER OF ANTENNA ELEMENTS.
1308 C###
1309 LSMP=.FALSE.
1310 MSX=MSX+1
1311 MSXAT=MSAT+MSX
1312 IF(MSXAT.GT.MSDX) WRITE(6,904) MSXAT
1313 904  FORMAT(' ***** NUMBER OF SOURCES= ',I3,' PROGRAM',
1314 2' ABORTS SINCE MAX. SOURCE DIMENSION IS EXCEEDED. *****')
1315 IF(MSXAT.GT.MSDX) STOP
1316 WRITE(6,3401) MSX
1317 3401 FORMAT(2H *,5X,'THIS IS SOURCE NO. ',I3,' IN THIS',
1318 2' COMPUTATION.',I79,1H*)
1319 WRITE(6,3008)
1320 WRITE(6,3006)
1321 C###
1322 C### XSS(N,MS)=XYZ LOCATION OF MS-TH ANTENNA ELEMENT.
1323 C###
1324 C### IMS(MS)=TYPE OF LINEAR ANTENNA
1325 C### .LT.0: ELECTRIC LINEAR ELEMENT
1326 C### .GT.0: MAGNETIC LINEAR ELEMENT
1327 C### ABS(IMB)=1: UNIFORM CURRENT DISTRIBUTION
1328 C### =2: STANDARD DIPOLE CURRENT DISTRIBUTION
1329 C### =3: CAVITY BACKED SLOT CURRENT DISTRIBUTION (TE01)
1330 C###
1331 C### HAWS(MS)=APERTURE WIDTH IN WAVELENGTHS (NOTE: IF
1332 C### HAWS(MS) IS LESS THAN .1 LAMBDA, SOURCE IS
1333 C### CONSIDERED TO BE DIPOLE SOURCE
1334 C### HS(MS)=LENGTH OF LINEAR ELEMENT IN WAVELENGTHS
1335 C###
1336 C### THSZ,PHSZ=ORIENTATION ANGLES USED TO DEFINE LINEAR
1337 C### ELEMENT AXIS.
1338 C###
1339 C### THSX,PHSX=ORIENTATION ANGLES USED TO DEFINE APERTURE
1340 C### PLANE OR DIPOLE X-AXIS.
1341 C###
1342 C### WMS,WPS=MAGNITUDE AND PHASE OF EXCITATION OF
1343 C### MS-TH ELEMENT.
1344 C###
1345 MS=MSX
1346 MSA(1,MS)=0
1347 MSA(2,MS)=0
1348 READ(6,*) (XSS(N,MS),N=1,3)
1349 READ(6,*) THSZ,PHSZ,THSX,PHSX
1350 READ(6,*) IMS(MS),HS(MS),HAWS(MS)
1351 READ(6,*) WMS,WPS
1352 IF(IMB(MS).LT.0) WRITE(6,3411) IMS(MS)
1353 3411 FORMAT(2H *,5X,'THIS IS AN ELECTRIC SOURCE OF TYPE ',I3,I79,1H*)
1354 IF(IMB(MS).GE.0) WRITE(6,3412) IMS(MS)
1355 3412 FORMAT(2H *,5X,'THIS IS A MAGNETIC SOURCE OF TYPE ',I3,I79,1H*)
1356 WRITE(6,3008)
1357 IF(IUNST.EQ.0) GO TO 3414
1358 UNSTS=UNIT(IUNST)
1359
1360 3413 WRITE(6,3413) HS(MS),HAWS(MS),LABEL(IUNST)
1361 3413 FORMAT(2H *,5X,'SOURCE LENGTH=',F10.5,' AND WIDTH='
1362 2,F10.5,1X,A6,I79,1H*)
1363 HS(MS)=UNSTS*UNITF+HS(MS)
1364 HAWS(MS)=UNSTS*UNITF+HAWS(MS)
1364 IF(IUNST.NE.1) WRITE(6,3008)
1365 IF(IUNST.NE.1) WRITE(6,3413) HS(MS),HAWS(MS),LABEL(1)
1366 GO TO 3416
1367 3414 WRITE(6,3415) HS(MS),HAWS(MS)

```



```

1368 3416 FORMAT(2H *,5X,'SOURCE LENGTH=',F10.5,' AND WIDTH='
1369 2,F10.5,' WAVELENGTHS',T79,1H*)
1370 3416 WRITE(6,3006)
1371 WS(MS)=WMS*CEXP(CJ*WPS*RPD)
1372 WRITE(6,3417) WMS,WPS
1373 3417 FORMAT(2H *,5X,'THE SOURCE WEIGHT HAS MAGNITUDE='
1374 2,F10.5,' AND PHASE=',F10.5,T79,1H*)
1375 WRITE(6,3006)
1376 WRITE(6,3006)
1377 WRITE(6,3421) LABEL(IUNIT)
1378 3421 FORMAT(2H *,T6,'SOURCE#',T17,'INPUT LOCATION IN ',A6,T46,
1379 2'ACTUAL LOCATION IN METERS',T79,1H*)
1380 WRITE(6,3422)
1381 3422 FORMAT(2H *,T6,7('-'),T16,27('-'),T46,27('-'),
1382 2T79,1H*)
1383 WRITE(6,3006)
1384 DO 3424 N=1,3
1385 3424 IQ(N)=XSS(N,MS)
1386 DO 3425 N=1,3
1387 3425 XSS(N,MS)=UNITS*(IQ(1)*VRT(1,N)+IQ(2)*VRT(2,N)
1388 2+IQ(3)*VRT(3,N))+TR(N)
1389 WRITE(6,3426) MS,(IQ(N),N=1,3),(XSS(N,MS),N=1,3)
1390 3426 FORMAT(2H *,T8,I3,T16,F8.3,2(' ',F8.3),T44,F8.3,2(' ',F8.3)
1391 2,T79,1H*)
1392 TQR=THSZ*RPD
1393 PQR=PHSZ*RPD
1394 IQ(1)=SIN(TQR)*COS(PQR)
1395 IQ(2)=SIN(TQR)*SIN(PQR)
1396 IQ(3)=COS(TQR)
1397 DO 3431 N=1,3
1398 3431 VISS(3,N,MS)=IQ(1)*VRT(1,N)+IQ(2)*VRT(2,N)+IQ(3)*VRT(3,N)
1399 TQR=THSX*RPD
1400 PQR=PHSX*RPD
1401 IQ(1)=SIN(TQR)*COS(PQR)
1402 IQ(2)=SIN(TQR)*SIN(PQR)
1403 IQ(3)=COS(TQR)
1404 DO 3432 N=1,3
1405 3432 VISS(1,N,MS)=IQ(1)*VRT(1,N)+IQ(2)*VRT(2,N)+IQ(3)*VRT(3,N)
1406 DZI=VISS(1,1,MS)+VISS(3,1,MS)+VISS(1,2,MS)+VISS(3,2,MS)
1407 2+VISS(1,3,MS)+VISS(3,3,MS)
1408 IF(ABS(DZI).GT.0.1) WRITE(6,3436)
1409 3436 FORMAT(' *** PROGRAM ABORTS IN SOURCE SECTION IN THAT THE',
1410 2' COORDINATES ARE NOT ORTHOGONAL !!! ***)
1411 IF(ABS(DZI).GT.0.1) STOP
1412 VISS(1,1,MS)=VISS(1,1,MS)-VISS(3,1,MS)*DZI
1413 VISS(1,2,MS)=VISS(1,2,MS)-VISS(3,2,MS)*DZI
1414 VISS(1,3,MS)=VISS(1,3,MS)-VISS(3,3,MS)*DZI
1415 DOT=VISS(1,1,MS)*VISS(1,1,MS)+VISS(1,2,MS)*VISS(1,2,MS)
1416 2+VISS(1,3,MS)*VISS(1,3,MS)
1417 DOT=SQRT(DOT)
1418 VISS(1,1,MS)=VISS(1,1,MS)/DOT
1419 VISS(1,2,MS)=VISS(1,2,MS)/DOT
1420 VISS(1,3,MS)=VISS(1,3,MS)/DOT
1421 VISS(2,1,MS)=VISS(3,2,MS)+VISS(1,3,MS)-VISS(3,3,MS)+VISS(1,2,MS)
1422 VISS(2,2,MS)=VISS(3,3,MS)+VISS(1,1,MS)-VISS(3,1,MS)+VISS(1,3,MS)
1423 VISS(2,3,MS)=VISS(3,1,MS)+VISS(1,2,MS)-VISS(3,2,MS)+VISS(1,1,MS)
1424 WRITE(6,3006)
1425 WRITE(6,3006)
1426 WRITE(6,3437)
1427 3437 FORMAT(2H *,5X,'THE FOLLOWING SOURCE ALIGNMENT IS USED:'
1428 2,T79,1H*)
1429 DO 3433 NI=1,3
1430 WRITE(6,3006)
1431 3433 WRITE(6,3434) (NI,NJ,MS,VISS(NI,NJ,MS),NJ=1,3)

```

```

1432 3434 FORMAT(2H *,1X,3(2X,'VXSS(',I1,',',I1,',',I2,')=',F9.5)
1433 2,T79,1H+)
1434 GO TO 3000
1435 C=====
1436 3810 CONTINUE
1437 C=== SA: COMMAND =====
1438 C$$$
1439 C$$$ MSX=NUMBER OF ANTENNA ARRAY GROUPINGS.
1440 C$$$
1441 C$$$ MSAX=NUMBER OF ELEMENTS PER GROUPING.
1442 C$$$
1443 LSMP=.FALSE.
1444 MSX=MSX+1
1445 READ(6,*) MSAX
1446 MSAT=MSAT+MSAX
1447 MSXAT=MSAT+MSX
1448 IF(MSXAT.GT.MSDX) WRITE(6,904) MSXAT
1449 IF(MSXAT.GT.MSDX) STOP
1450 WRITE(6,3805) MSX,MSAX
1451 3805 FORMAT(2H *,5X,'THIS IS SOURCE NO. ',I3,' IN THIS',
1452 2' COMPUTATION.',T79,1H*/2H *,5X,'THERE ARE ',
1453 3I3,' SOURCES ARRANGED TOGETHER.',T79,1H+)
1454 WRITE(6,3006)
1455 WRITE(6,3006)
1456 C$$$
1457 C$$$ XSS(N,MA)=XYZ LOCATION OF MA-TH ANTENNA ELEMENT.
1458 C$$$
1459 C$$$ XSS(N,MS)=XYZ LOCATION OF MS-TH WEIGHTED CENTER OF THE
1460 C$$$ ARRAY GROUPING.
1461 C$$$
1462 C$$$ THE ARRAY ELEMENTS ARE ASSUMED TO HAVE THE SAME LENGTH,
1463 C$$$ WIDTH, AND ORIENTATION. ALSO, THEY ARE ASSUMED TO BE
1464 C$$$ EITHER ALL MOUNTED AND OR ALL OFF A PLATE.
1465 C$$$ IMS(MS)=TYPE OF LINEAR ANTENNA
1466 C$$$ .LT.O: ELECTRIC LINEAR ELEMENT
1467 C$$$ .GT.O: MAGNETIC LINEAR ELEMENT
1468 C$$$ ABS(IMB)=1: UNIFORM CURRENT DISTRIBUTION
1469 C$$$ =2: STANDARD DIPOLE CURRENT DISTRIBUTION
1470 C$$$ =3: CAVITY BACKED SLOT CURRENT DISTRIBUTION (TE01)
1471 C$$$
1472 C$$$ HAWS(MS)=APERTURE WIDTH IN WAVELENGTHS (NOTE: IF
1473 C$$$ HAWS(MS) IS LESS THAN .1 LAMBDA, SOURCE IS
1474 C$$$ CONSIDERED TO BE DIPOLE SOURCE
1475 C$$$ HS(MS)=LENGTH OF LINEAR ELEMENT IN WAVELENGTHS
1476 C$$$
1477 C$$$ THSZ,PHSZ=ORIENTATION ANGLES USED TO DEFINE LINEAR
1478 C$$$ ELEMENT AXIS.
1479 C$$$
1480 C$$$ THSX,PHSX=ORIENTATION ANGLES USED TO DEFINE APERTURE
1481 C$$$ PLANE OR DIPOLE X-AXIS.
1482 C$$$
1483 C$$$ WMS,WPS=MAGNITUDE AND PHASE OF EXCITATION OF
1484 C$$$ MA-TH ELEMENT.
1485 C$$$
1486 MS=MSX
1487 MAI=MSDX-MSAT+1
1488 MAF=MAI+MSAX-1
1489 MSA(1,MS)=MAI
1490 MSA(2,MS)=MAF
1491 DO 3841 MA=MAI,MAF
1492 3841 READ(6,*) (XSS(N,MA),N=1,3)
1493 READ(6,*) THSZ,PHSZ,THSX,PHSX
1494 READ(6,*) IMS(MS),HS(MS),HAWS(MS)
1495 IF(IMB(MS).LT.O) WRITE(6,3411) IMS(MS)

```

```

1496      IF(IMS(MS).GE.0) WRITE(6,3412) IMS(MS)
1497      WRITE(6,3006)
1498      IF(IUNST.EQ.0) GO TO 3814
1499      UNSTS=UNIT(IUNST)
1500      WRITE(6,3413) HS(MS),HAWS(MS),LABEL(IUNST)

1501      HS(MS)=UNSTS*UNITF+HS(MS)
1502      HAWS(MS)=UNSTS*UNIIF+HAWS(MS)
1503      IF(IUNST.NE.1) WRITE(6,3006)
1504      IF(IUNST.NE.1) WRITE(6,3413) HS(MS),HAWS(MS),LABEL(1)
1505      GO TO 3816
1506 3814 WRITE(6,3415) HS(MS),HAWS(MS)
1507 3816 WRITE(6,3006)
1508      WS(MS)=(1.,0.)
1509      WMSA=0.
1510      XSAX=0.
1511      XSAY=0.
1512      XSAZ=0.
1513      DO 3843 MA=MAI,MAF
1514      READ(6,*) WMS,WPS
1515      WRITE(6,3817) MA,WMS,WPS
1516 3817 FORMAT(2H *,5X,'SOURCE ',I3,' HAS MAGNITUDE='
1517 2,F10.6,' AND PHASE=',F10.6,I79,1H*)
1518      WS(MA)=WMS*CEXP(CJ+WPS*RPD)
1519      WMSA=WMSA+WMS
1520      XSAX=XSAX+WMS*XSS(1,MA)
1521      XSAY=XSAY+WMS*XSS(2,MA)
1522 3843 XSAZ=XSAZ+WMS*XSS(3,MA)
1523      XSS(1,MS)=XSAX/WMSA
1524      XSS(2,MS)=XSAY/WMSA
1525      XSS(3,MS)=XSAZ/WMSA
1526      WRITE(6,3006)
1527      WRITE(6,3006)
1528      WRITE(6,3421) LABEL(IUNIT)
1529      WRITE(6,3422)
1530      WRITE(6,3006)
1531      DO 3824 N=1,3
1532 3824 IQ(N)=XSS(N,MS)
1533      DO 3825 N=1,3
1534 3825 XSS(N,MS)=UNITS*(IQ(1)+VRT(1,N)+IQ(2)+VRT(2,N)
1535 2*IQ(3)+VRT(3,N))+TR(N)
1536      WRITE(6,3426) MS,(IQ(N),N=1,3),(XSS(N,MS),N=1,3)
1537      DO 3829 MA=MAI,MAF
1538      DO 3827 N=1,3
1539 3827 IQ(N)=XSS(N,MA)
1540      DO 3828 N=1,3
1541 3828 XSS(N,MA)=UNITS*(IQ(1)+VRT(1,N)+IQ(2)+VRT(2,N)
1542 2*IQ(3)+VRT(3,N))+TR(N)
1543 3829 WRITE(6,3426) MA,(IQ(N),N=1,3),(XSS(N,MA),N=1,3)
1544      TQR=THSZ+RPD
1545      PQR=PHSZ+RPD
1546      XQ(1)=SIN(TQR)+COS(PQR)
1547      XQ(2)=SIN(TQR)*SIN(PQR)
1548      XQ(3)=COS(TQR)
1549      DO 3831 N=1,3
1550 3831 VXSS(3,N,MS)=XQ(1)+VRT(1,N)+XQ(2)+VRT(2,N)+XQ(3)+VRT(3,N)
1551      TQR=THSX+RPD
1552      PQR=PHSX+RPD
1553      XQ(1)=SIN(TQR)+COS(PQR)
1554      XQ(2)=SIN(TQR)*SIN(PQR)
1555      XQ(3)=COS(TQR)
1556      DO 3832 N=1,3
1557 3832 VISS(1,N,MS)=XQ(1)+VRT(1,N)+XQ(2)+VRT(2,N)+XQ(3)+VRT(3,N)
1558      DZX=VISS(1,1,MS)*VISS(3,1,MS)+VISS(1,2,MS)*VISS(3,2,MS)
1559      2*VISS(1,3,MS)*VISS(3,3,MS)

```

```

1560      IF(ABS(DZX).GT.0.1) WRITE(6,3436)
1561      IF(ABS(DZX).GT.0.1) STOP
1562      VXSS(1,1,MS)=VXSS(1,1,MS)-VXSS(3,1,MS)*DZX
1563      VXSS(1,2,MS)=VXSS(1,2,MS)-VXSS(3,2,MS)*DZX
1564      VXSS(1,3,MS)=VXSS(1,3,MS)-VXSS(3,3,MS)*DZX
1565      DOT=VXSS(1,1,MS)*VXSS(1,1,MS)+VXSS(1,2,MS)*VXSS(1,2,MS)
1566      +VXSS(1,3,MS)*VXSS(1,3,MS)
1567      DOT=SQRT(DOT)
1568      VXSS(1,1,MS)=VXSS(1,1,MS)/DOT
1569      VXSS(1,2,MS)=VXSS(1,2,MS)/DOT
1570      VXSS(1,3,MS)=VXSS(1,3,MS)/DOT
1571      VXSS(2,1,MS)=VXSS(3,2,MS)*VXSS(1,3,MS)-VXSS(3,3,MS)*VXSS(1,2,MS)
1572      VXSS(2,2,MS)=VXSS(3,3,MS)*VXSS(1,1,MS)-VXSS(3,1,MS)*VXSS(1,3,MS)
1573      VXSS(2,3,MS)=VXSS(3,1,MS)*VXSS(1,2,MS)-VXSS(3,2,MS)*VXSS(1,1,MS)
1574      WRITE(6,3006)
1575      WRITE(6,3006)
1576      WRITE(6,3437)
1577      DO 3833 NI=1,3
1578      WRITE(6,3006)
1579      3833 WRITE(6,3434) (NI,NJ,MS,VXSS(NI,NJ,MS),NJ=1,3)
1580      GO TO 3000
1581      C=====
1582      3440 CONTINUE
1583      C=== PR:      COMMAND      =====
1584      C$$$
1585      C$$$ IPRAD= 1 =NORMALIZATION FOR FAR ZONE AS FOLLOWS
1586      C$$$
1587      C$$$ PRAD=TOTAL POWER RADIATED IN WATTS.
1588      C$$$
1589      C$$$ PRAD CAN ALSO BE SPECIFIED AS THE POWER INPUT IN WATTS.
1590      C$$$
1591      C$$$ NOTE IF PRAD IS LESS THAN OR EQUAL TO 1.E-30
1592      C$$$ THAN LPRAD WILL BE SET FALSE
1593      C$$$
1594      LPRAD=.TRUE.
1595      READ(6,*) IPRAD
1596      IF(IPRAD.GT.4) STOP
1597      GO TO (3444,3445,3446,3447),IPRAD
1598      3444 READ(6,*) PRAD
1599      IF(PRAD.LT.1.1E-30) GO TO 3442
1600      WRITE(6,3441) PRAD
1601      3441 FORMAT(2H *,5X,'TOTAL POWER RADIATED IN WATTS= ',E12.6
1602      2,T9,1H*)
1603      GO TO 3000
1604      3442 CONTINUE
1605      LPRAD=.FALSE.
1606      PRAD=0.
1607      WRITE(6,3443)
1608      3443 FORMAT(2H *,5X,'NO POWER RADIATED IS SPECIFIED',T9,1H*)
1609      GO TO 3000
1610      3445 CONTINUE
1611      C$$$
1612      C$$$ IPRAD = 2 =MUTUAL IMPEDANCE CALCULATION Z12 = Z21
1613      C$$$
1614      C$$$ CI11 = SOURCE TERMINAL CURRENT (REAL AND IMAGINARY)
1615      C$$$ CI22 = RECEIVER TERMINAL CURRENT (REAL AND IMAGINARY)
1616      C$$$
1617      READ(6,*) CI11,CI22
1618      WRITE(6,4445) CI11,CI22
1619      4445 FORMAT(2H *,5X,'SOURCE TERMINAL CURRENT= ',2E12.6,T9,1H*/
1620      2,2H *,5X,'RECEIVER TERMINAL CURRENT= ',2E12.6,T9,1H*)
1621      GO TO 3000
1622      3446 CONTINUE
1623      C$$$

```

```

1624 C### IPRAD = 3 =COUPLING VIA THE REACTION THEORY
1625 C### THIS GIVES A MODIFIED FRII'S TRANSMISSION TYPE RESULT
1626 C###
1627 C### PRAD = POWER RADIATED BY THE SOURCE
1628 C### PRADR = POWER RADIATED BY THE RECEIVER AS IF IT WERE A SOURCE
1629 C###
1630 READ(5,*) PRAD,PRADR
1631 WRITE(6,4447) PRAD,PRADR
1632 4447 FORMAT(2H *,5X,' SOURCE POWER RADIATED= ',E12.6,T79,1H+/  

1633 2,2H *,5X,'RECEIVER POWER RADIATED= ',E12.6,T79,1H+)  

1634 GO TO 3000
1635 3447 CONTINUE
1636 C###
1637 C### IPRAD= 4 =COUPLING BY THE LINVILLE METHOD
1638 C###
1639 C### CI11 = SOURCE TERMINAL CURRENT (REAL AND IMAGINARY)
1640 C### CI22 = RECEIVER TERMINAL CURRENT (REAL AND IMAGINARY)
1641 C###
1642 C### Z11 = SOURCE TERMINAL IMPEDANCE (REAL AND IMAGINARY)
1643 C### Z22 = RECEIVER TERMINAL IMPEDANCE (REAL AND IMAGINARY)
1644 C###
1645 READ(5,*) CI11,CI22
1646 WRITE(6,4445) CI11,CI22
1647 READ(5,*) Z11,Z22
1648 WRITE(6,4446) Z11,Z22
1649 4446 FORMAT(2H *,5X,' SOURCE TERMINAL IMPEDANCE= ',2E12.6,T79,1H+/  

1650 2,2H *,5X,'RECEIVER TERMINAL IMPEDANCE= ',2E12.6,T79,1H+)  

1651 GO TO 3000
1652 C=====
1653 4400 CONTINUE
1654 C=== RG: COMMAND =====
1655 C###
1656 C### MRX=NUMBER OF ANTENNA ELEMENTS.
1657 C###
1658 LRCVR=. TRUE.
1659 LRMP=. FALSE.
1660 MRX=MRX+1
1661 MRXAT=MRAT+MRX
1662 IF(MRXAT.GT.MRDX) WRITE(6,4404) MRXAT
1663 4404 FORMAT(' ***** NUMBER OF RECEIVERS= ',I3,' PROGRAM',  

1664 2' ABORTS SINCE MAX. RECEIVER DIMENSION IS EXCEEDED. *****')
1665 IF(MRXAT.GT.MRDX) STOP
1666 WRITE(6,4401) MRX
1667 4401 FORMAT(2H *,5X,'THIS IS RECEIVER NO. ',I3,' IN THIS',  

1668 2' COMPUTATION.',T79,1H+)  

1669 WRITE(6,3006)  

1670 WRITE(6,3006)
1671 C###
1672 C### XRR(N,MR)=XYZ LOCATION OF MR-TH ANTENNA ELEMENT.
1673 C###
1674 C### IMR(MR)=TYPE OF LINEAR ANTENNA
1675 C### .LT.O: ELECTRIC LINEAR ELEMENT
1676 C### .GT.O: MAGNETIC LINEAR ELEMENT
1677 C### ABS(IMR)=1: UNIFORM CURRENT DISTRIBUTION
1678 C### =2: STANDARD DIPOLE CURRENT DISTRIBUTION
1679 C### =3: CAVITY BACKED SLOT CURRENT DISTRIBUTION (TE01)
1680 C###
1681 C### HAWR(MR)=APERTURE WIDTH IN WAVELENGTHS (NOTE: IF
1682 C### HAWR(MR) IS LESS THAN .1 LAMBDA, RECEIVER IS
1683 C### CONSIDERED TO BE DIPOLE RECEIVER
1684 C### HR(MR)=LENGTH OF LINEAR ELEMENT IN WAVELENGTHS
1685 C###
1686 C### THRZ,PHRZ=ORIENTATION ANGLES USED TO DEFINE LINEAR
1687 C### ELEMENT AXIS.

```

```

1688 C$$$
1689 C$$$ THRX,PHRX=ORIENTATION ANGLES USED TO DEFINE APERTURE
1690 C$$$ PLANE OR DIPOLE X-AXIS.
1691 C$$$
1692 C$$$ WNR,WPR=MAGNITUDE AND PHASE OF EXCITATION OF
1693 C$$$ MR-IH ELEMENT.
1694 C$$$
1695 MR=MRX
1696 MRA(1,MR)=0
1697 MRA(2,MR)=0
1698 READ(5,*) (XRR(N,MR),N=1,3)
1699 READ(5,*) THRX,PHRX,THRX,PHRX
1700 READ(5,*) IMR(MR),HR(MR),HAWR(MR)
1701 READ(5,*) WNR,WPR
1702 IF(IMR(MR).LT.0) WRITE(6,4411) IMR(MR)
1703 4411 FORMAT(2H *,5X,'THIS IS AN ELECTRIC RECEIVER OF TYPE ',I3
1704 2,T79,1H*)
1705 IF(IMR(MR).GE.0) WRITE(6,4412) IMR(MR)
1706 4412 FORMAT(2H *,5X,'THIS IS A MAGNETIC RECEIVER OF TYPE ',I3
1707 2,T79,1H*)
1708 WRITE(6,3006)
1709 IF(IUNST.EQ.0) GO TO 4414
1710 UNSTS=UNIT(IUNST)
1711 WRITE(6,4413) HR(MR),HAWR(MR),LABEL(IUNST)
1712 4413 FORMAT(2H *,5X,'RECEIVER LENGTH=',F10.5,' AND WIDTH='
1713 2,F10.5,1X,A6,T79,1H*)
1714 HR(MR)=UNSTS*UNITF*HR(MR)
1715 HAWR(MR)=UNSTS*UNITF*HAWR(MR)
1716 IF(IUNST.NE.1) WRITE(6,3006)
1717 IF(IUNST.NE.1) WRITE(6,4413) HR(MR),HAWR(MR),LABEL(1)
1718 GO TO 4416
1719 4414 WRITE(6,4415) HR(MR),HAWR(MR)
1720 4415 FORMAT(2H *,5X,'RECEIVER LENGTH=',F10.5,' AND WIDTH='
1721 2,F10.5,' WAVELENGTHS',T79,1H*)
1722 4416 WRITE(6,3006)
1723 WR(MR)=WNR*CEXP(CJ+WPR*RPD)
1724 WRITE(6,4417) WNR,WPR
1725 4417 FORMAT(2H *,5X,'THE RECEIVER WEIGHT HAS MAGNITUDE='
1726 2,F10.5,' AND PHASE=',F10.5,T79,1H*)
1727 WRITE(6,3006)
1728 WRITE(6,3006)
1729 WRITE(6,4421) LABEL(IUNIT)
1730 4421 FORMAT(2H *,76,'RECEIVER#',I17,' INPUT LOCATION IN ',A6,I46,
1731 2'ACTUAL LOCATION IN METERS',T79,1H*)
1732 WRITE(6,4422)
1733 4422 FORMAT(2H *,T6,7('-'),I16,27('-'),I45,27('-'),
1734 2T79,1H*)
1735 WRITE(6,3006)
1736 DO 4424 N=1,3
1737 4424 XQ(N)=XRR(N,MR)
1738 DO 4425 N=1,3
1739 4425 XRR(N,MR)=UNITS*(XQ(1)*VRT(1,N)+XQ(2)*VRT(2,N)
1740 2+XQ(3)*VRT(3,N))+TR(N)
1741 WRITE(6,4426) MR,(XQ(N),N=1,3),(XRR(N,MR),N=1,3)
1742 4426 FORMAT(2H *,T8,I3,I15,F8.3,2(' ',F8.3),I44,F8.3,2(' ',F8.3)
1743 2,T79,1H*)
1744 TQR=THRX*RPD
1745 PQR=PHRX*RPD
1746 IQ(1)=SIN(TQR)*COS(PQR)
1747 IQ(2)=SIN(TQR)*SIN(PQR)
1748 IQ(3)=COS(TQR)
1749 DO 4431 N=1,3
1750 4431 VXRR(3,N,MR)=IQ(1)*VRT(1,N)+IQ(2)*VRT(2,N)+IQ(3)*VRT(3,N)
1751 TQR=THRX*RPD

```

```

1752      PQR=PHRX*RPD
1753      IQ(1)=SIN(TQR)*COS(PQR)
1754      IQ(2)=SIN(TQR)*SIN(PQR)
1755      IQ(3)=COS(TQR)
1756      DO 4432 N=1,3
1757 4432  VIRR(1,N,MR)=IQ(1)*VRT(1,N)+IQ(2)*VRT(2,N)+IQ(3)*VRT(3,N)
1758      DZX=VIRR(1,1,MR)*VIRR(3,1,MR)+VIRR(1,2,MR)*VIRR(3,2,MR)
1759      2*VIRR(1,3,MR)+VIRR(3,3,MR)
1760      IF(ABS(DZX).GT.0.1) WRITE(6,4436)
1761 4436  FORMAT(' *** PROGRAM ABORTS IN RECEIVER SECTION IN THAT THE',
1762      2' COORDINATES ARE NOT ORTHOGONAL !!! ***')
1763      IF(ABS(DZX).GT.0.1) STOP
1764      VIRR(1,1,MR)=VIRR(1,1,MR)-VIRR(3,1,MR)*DZX
1765      VIRR(1,2,MR)=VIRR(1,2,MR)-VIRR(3,2,MR)*DZX
1766      VIRR(1,3,MR)=VIRR(1,3,MR)-VIRR(3,3,MR)*DZX
1767      DOT=VIRR(1,1,MR)*VIRR(1,1,MR)+VIRR(1,2,MR)*VIRR(1,2,MR)
1768      2*VIRR(1,3,MR)*VIRR(1,3,MR)
1769      DOT=SQRT(DOT)
1770      VIRR(1,1,MR)=VIRR(1,1,MR)/DOT
1771      VIRR(1,2,MR)=VIRR(1,2,MR)/DOT
1772      VIRR(1,3,MR)=VIRR(1,3,MR)/DOT
1773      VIRR(2,1,MR)=VIRR(3,2,MR)*VIRR(1,3,MR)-VIRR(3,3,MR)*VIRR(1,2,MR)
1774      VIRR(2,2,MR)=VIRR(3,3,MR)*VIRR(1,1,MR)-VIRR(3,1,MR)*VIRR(1,3,MR)
1775      VIRR(2,3,MR)=VIRR(3,1,MR)*VIRR(1,2,MR)-VIRR(3,2,MR)*VIRR(1,1,MR)
1776      WRITE(6,3006)
1777      WRITE(6,3006)
1778      WRITE(6,4437)
1779 4437  FORMAT(2H * ,5X, 'THE FOLLOWING RECEIVER ALIGNMENT IS USED:'
1780      2 ,T79,1H*)
1781      DO 4433 NI=1,3
1782      WRITE(6,3006)
1783 4433  WRITE(6,4434) (NI,NJ,MR,VIRR(NI,NJ,MR),NJ=1,3)
1784 4434  FORMAT(2H * ,1X,3(2X, 'VIRR(',I1,',',I1,',',I2,')=' ,F9.5)
1785      2 ,T79,1H*)
1786      GO TO 3000
1787  C=====
1788  4810  CONTINUE
1789  C=== RA:      COMMAND      =====
1790  C$$$
1791  C$$$ MRX=NUMBER OF ANTENNA ARRAY GROUPINGS.
1792  C$$$
1793  C$$$ MRAX=NUMBER OF ELEMENTS PER GROUPING.
1794  C$$$
1795      LRCVR=.TRUE.
1796      LRMP=.FALSE.
1797      MRX=MRX+1
1798      READ(5,*) MRAX
1799      MRAT=MRAT+MRAX
1800      MRXAT=MRAT+MRX
1801      IF(MRXAT.GT.MRDX) WRITE(6,4404) MRXAT
1802      IF(MRXAT.GT.MRDX) STOP
1803      WRITE(6,4805) MRX,MRAX
1804 4805  FORMAT(2H * ,5X, 'THIS IS RECEIVER NO. ',I3, ' IN THIS',
1805      2' COMPUTATION.',T79,1H+/2H * ,5X, 'THERE ARE ',
1806      3I3, ' RECEIVERS ARRAYED TOGETHER.',T79,1H+)
1807      WRITE(6,3006)
1808      WRITE(6,3006)
1809  C$$$
1810  C$$$ XRR(N,MA)=XYZ LOCATION OF MA-TH ANTENNA ELEMENT.
1811  C$$$
1812  C$$$ XRR(N,MR)=XYZ LOCATION OF MR-TH WEIGHTED CENTER OF THE
1813  C$$$      ARRAY GROUPING.
1814  C$$$
1815  C$$$ THE ARRAY ELEMENTS ARE ASSUMED TO HAVE THE SAME LENGTH.

```

```

1816 C### WIDTH, AND ORIENTATION. ALSO, THEY ARE ASSUMED TO BE
1817 C### EITHER ALL MOUNTED AND OR ALL OFF A PLATE.
1818 C### IMR(MR)=TYPE OF LINEAR ANTENNA
1819 C### .LT.O: ELECTRIC LINEAR ELEMENT
1820 C### .GT.O: MAGNETIC LINEAR ELEMENT
1821 C### ABS(IMR)=1: UNIFORM CURRENT DISTRIBUTION
1822 C### =2: STANDARD DIPOLE CURRENT DISTRIBUTION
1823 C### =3: CAVITY BACKED SLOT CURRENT DISTRIBUTION (TE01)
1824 C###
1825 C### HAWR(MR)=APERTURE WIDTH IN WAVELENGTHS (NOTE: IF
1826 C### HAWR(MR) IS LESS THAN .1 LAMBDA, RECEIVER IS
1827 C### CONSIDERED TO BE DIPOLE RECEIVER
1828 C### HR(MR)=LENGTH OF LINEAR ELEMENT IN WAVELENGTHS
1829 C###
1830 C### THRZ,PHRZ=ORIENTATION ANGLES USED TO DEFINE LINEAR
1831 C### ELEMENT AXIS.
1832 C###
1833 C### THRX,PHRX=ORIENTATION ANGLES USED TO DEFINE APERTURE
1834 C### PLANE OR DIPOLE X-AXIS.
1835 C###
1836 C### WMR,WPR=MAGNITUDE AND PHASE OF EXCITATION OF
1837 C### MA-TH ELEMENT.
1838 C###
1839 MR=MRX
1840 MAI=MRDX-MRAT+1
1841 MAF=MAI+MRAX-1
1842 MRA(1,MR)=MAI
1843 MRA(2,MR)=MAF
1844 DO 4841 MA=MAI,MAF
1845 4841 READ(5,*) (XRR(N,MA),N=1,3)
1846 READ(5,*) THRZ,PHRZ,THRX,PHRX
1847 READ(5,*) IMR(MR),HR(MR),HAWR(MR)
1848 IF(IMR(MR).LT.O) WRITE(6,4411) IMR(MR)
1849 IF(IMR(MR).GE.O) WRITE(6,4412) IMR(MR)
1850 WRITE(6,3006)
1851 IF(IUNST.EQ.O) GO TO 4814
1852 UNSTS=UNIT(IUNST)
1853 WRITE(6,4413) HR(MR),HAWR(MR),LABEL(IUNST)
1854 HR(MR)=UNSTS*UNITF*HR(MR)
1855 HAWR(MR)=UNSTS*UNITF*HAWR(MR)
1856 IF(IUNST.NE.1) WRITE(6,3006)
1857 IF(IUNST.NE.1) WRITE(6,4413) HR(MR),HAWR(MR),LABEL(1)
1858 GO TO 4816
1859 4814 WRITE(6,4415) HR(MR),HAWR(MR)
1860 4816 WRITE(6,3006)
1861 WR(MR)=(1.,0.)
1862 WMRA=0.
1863 XRAX=0.
1864 XRAY=0.
1865 XRAZ=0.
1866 DO 4843 MA=MAI,MAF
1867 READ(5,*) WMR,WPR
1868 WRITE(6,4817) MA,WMR,WPR
1869 4817 FORMAT(2H *,5X,'RECEIVER ',I3,' HAS MAGNITUDE='
1870 2,F10.6,' AND PHASE=',F10.6,T99,1H*)
1871 WR(MA)=WMR*CEXP(CJ*WPR*RPD)
1872 WMRA=WMRA+WMR
1873 XRAX=XRAX+WMR*XRR(1,MA)
1874 XRAY=XRAY+WMR*XRR(2,MA)
1875 4843 XRAZ=XRAZ+WMR*XRR(3,MA)
1876 XRR(1,MR)=XRAX/WMRA
1877 XRR(2,MR)=XRAY/WMRA
1878 XRR(3,MR)=XRAZ/WMRA
1879 WRITE(6,3006)

```



```

1880      WRITE(6,3006)
1881      WRITE(6,4421) LABEL(IUNIT)
1882      WRITE(6,4422)
1883      WRITE(6,3006)
1884      DO 4824 N=1,3
1885 4824  IQ(N)=XRR(N,MR)
1886      DO 4825 N=1,3
1887 4825  XRR(N,MR)=UNITS*(IQ(1)+VRT(1,N)+IQ(2)+VRT(2,N)
1888      2+IQ(3)+VRT(3,N))+TR(N)
1889      WRITE(6,4426) MR, (IQ(N),N=1,3), (XRR(N,MR),N=1,3)
1890      DO 4829 MA=MAI,MAF
1891      DO 4827 N=1,3
1892 4827  IQ(N)=XRR(N,MA)
1893      DO 4828 N=1,3
1894 4828  XRR(N,MA)=UNITS*(IQ(1)+VRT(1,N)+IQ(2)+VRT(2,N)
1895      2+IQ(3)+VRT(3,N))+TR(N)
1896 4829  WRITE(6,4426) MA, (IQ(N),N=1,3), (XRR(N,MA),N=1,3)
1897      TQR=THRZ+RPD
1898      PQR=PHRZ+RPD
1899      IQ(1)=SIN(TQR)*COS(PQR)
1900      IQ(2)=SIN(TQR)*SIN(PQR)
1901      IQ(3)=COS(TQR)
1902      DO 4831 N=1,3
1903 4831  VXRR(3,N,MR)=IQ(1)+VRT(1,N)+IQ(2)+VRT(2,N)+IQ(3)+VRT(3,N)
1904      TQR=THRX+RPD
1905      PQR=PHRX+RPD
1906      IQ(1)=SIN(TQR)*COS(PQR)
1907      IQ(2)=SIN(TQR)*SIN(PQR)
1908      IQ(3)=COS(TQR)
1909      DO 4832 N=1,3
1910 4832  VXRR(1,N,MR)=IQ(1)+VRT(1,N)+IQ(2)+VRT(2,N)+IQ(3)+VRT(3,N)
1911      DZX=VXRR(1,1,MR)+VXRR(3,1,MR)+VXRR(1,2,MR)+VXRR(3,2,MR)
1912      2+VXRR(1,3,MR)+VXRR(3,3,MR)
1913      IF (ABS(DZX).GT.0.1) WRITE(6,4436)
1914      IF (ABS(DZX).GT.0.1) STOP
1915      VXRR(1,1,MR)=VXRR(1,1,MR)-VXRR(3,1,MR)+DZX
1916      VXRR(1,2,MR)=VXRR(1,2,MR)-VXRR(3,2,MR)+DZX
1917      VXRR(1,3,MR)=VXRR(1,3,MR)-VXRR(3,3,MR)+DZX
1918      DOT=VXRR(1,1,MR)*VXRR(1,1,MR)+VXRR(1,2,MR)*VXRR(1,2,MR)
1919      2+VXRR(1,3,MR)*VXRR(1,3,MR)
1920      DOT=SQRT(DOT)
1921      VXRR(1,1,MR)=VXRR(1,1,MR)/DOT
1922      VXRR(1,2,MR)=VXRR(1,2,MR)/DOT
1923      VXRR(1,3,MR)=VXRR(1,3,MR)/DOT
1924      VXRR(2,1,MR)=VXRR(3,2,MR)+VXRR(1,3,MR)-VXRR(3,3,MR)+VXRR(1,2,MR)
1925      VXRR(2,2,MR)=VXRR(3,3,MR)+VXRR(1,1,MR)-VXRR(3,1,MR)+VXRR(1,3,MR)
1926      VXRR(2,3,MR)=VXRR(3,1,MR)+VXRR(1,2,MR)-VXRR(3,2,MR)+VXRR(1,1,MR)
1927      WRITE(6,3006)
1928      WRITE(6,3006)
1929      WRITE(6,4437)
1930      DO 4833 NI=1,3
1931      WRITE(6,3006)
1932 4833  WRITE(6,4434) (NI,NJ,MR,VXRR(NI,NJ,MR),NJ=1,3)
1933      GO TO 3000
1934      C=====
1935      3450 CONTINUE
1936      C=== SM:      COMMAND      =====
1937      C$$$
1938      C$$$ PRAD=TOTAL POWER RADIATED IN WATTS
1939      C$$$
1940      LPRAD=.TRUE.
1941      READ(5,*) PRAD
1942      WRITE(6,3441) PRAD
1943      WRITE(6,3006)

```

```

1944 C###
1945 C### MSX=NUMBER OF ANTENNA SEGMENTS
1946 C###
1947 LSMP=.TRUE.
1948 READ(6,*) MSX
1949 IF(MSX.GT.MSDX) WRITE(6,3477) MSX
1950 3477 FORMAT(' ***** NUMBER OF SEGMENTS= ',I3,
1951 2' PROGRAM ABORTS SINCE MAX. SOURCE DIMENSION'
1952 3,' IS EXCEEDED. *****')
1953 IF(MSX.GT.MSDX) STOP
1954 WRITE(6,3451) MSX
1955 3451 FORMAT(2H *,5X,'THERE ARE ',I3,' SEGMENTS IN THIS',
1956 2' COMPUTATION.',I79,1H+)
1957 WRITE(6,3006)
1958 WRITE(6,3006)
1959 C###
1960 C### XS(MS,N)=XYZ LOCATION OF MS-TH ANTENNA SEGMENT
1961 C###
1962 C### IMS(MS)=-1=ELECTRIC LINEAR ELEMENT WITH A UNIFORM DISTRIBUTION
1963 C###
1964 C### HS(MS)=LENGTH OF LINEAR ELEMENT
1965 C###
1966 C### THSZ,PHSZ=ORIENTATION ANGLES USED TO DEFINE
1967 C### LINEAR ELEMENT AXIS.
1968 C###
1969 C### WMS,WPS=REAL AND IMAGINARY CURRENT WEIGHT.
1970 C###
1971 WRITE(6,3458) LABEL(IUNIT)
1972 3458 FORMAT(2H *,I7,'MS',I13,'HS:',A6,I23,'HS:METERS',
1973 2I41,'INPUT: THS,PHS',I60,'ACTUAL: THS,PHS',I79,1H+)
1974 WRITE(6,3459)
1975 3459 FORMAT(2H *,I6,3('-'),I12,20('-'),I40,16('-'),I59,
1976 217('-'),I79,1H+)
1977 WRITE(6,3006)
1978 DO 3463 MS=1,MSX
1979 READ(6,*) (XSS(N,MS),N=1,3),HS(MS),THSZ,PHSZ
1980 MSA(1,MS)=0
1981 MSA(2,MS)=0
1982 IMS(MS)=-1
1983 HAWS(MS)=0.
1984 HSQ=HS(MS)
1985 HS(MS)=UNITS*HSQ
1986 TQ=90.-THSZ
1987 PQ=PHSZ
1988 XQ(1)=SIN(TQ+RPD)*COS(PQ+RPD)
1989 XQ(2)=SIN(TQ+RPD)*SIN(PQ+RPD)
1990 XQ(3)=COS(TQ+RPD)
1991 DO 3481 N=1,3
1992 3481 XQR(N)=XQ(1)*VRT(1,N)+XQ(2)*VRT(2,N)+XQ(3)*VRT(3,N)
1993 THSZ=DPR*BTAN2(SQRT(XQR(1)*XQR(1)+XQR(2)*XQR(2)),XQR(3))
1994 PHSZ=DPR*BTAN2(XQR(2),XQR(1))
1995 WRITE(6,3464) MS,HSQ,HS(MS),TQ,PQ,THSZ,PHSZ
1996 3464 FORMAT(2H *,I6,I3,3X,2(2X,F8.4),5X,2(2X,F8.3),',',F8.3)
1997 2,I79,1H+)
1998 DO 3484 N=1,3
1999 3484 VXSS(3,N,MS)=XQR(N)

2000 VXSS(1,1,MS)=COS(THSZ+RPD)*COS(PHSZ+RPD)
2001 VXSS(1,2,MS)=COS(THSZ+RPD)*SIN(PHSZ+RPD)
2002 VXSS(1,3,MS)=-SIN(THSZ+RPD)
2003 VXSS(2,1,MS)=-SIN(PHSZ+RPD)
2004 VXSS(2,2,MS)=COS(PHSZ+RPD)
2005 VXSS(2,3,MS)=0.
2006 3463 CONTINUE
2007 WRITE(6,3006)

```

```

2008 WRITE(6,3006)
2009 WRITE(6,3006)
2010 WRITE(6,3454)
2011 3454 FORMAT(2H *,T31,'SEGMENT COORDINATES',T79,'*')
2012 WRITE(6,3006)
2013 WRITE(6,3006)
2014 WRITE(6,3456) LABEL(IUNIT)
2015 3456 FORMAT(2H *,T7,'MS',T14,'INPUT LOCATION IN ',A6,
2016 2T43,'ACTUAL LOCATION IN METERS',T79,'*')
2017 WRITE(6,3457)
2018 3457 FORMAT(2H *,T6,3('-',),T13,26('-',),T42,27('-',),T79,'*')
2019 WRITE(6,3006)
2020 DO 3473 MS=1,MSX
2021 DO 3474 N=1,3
2022 3474 IQ(N)=ISS(N,MS)
2023 DO 3475 N=1,3
2024 3475 ISS(N,MS)=UNITS*(IQ(1)*VRT(1,N)+IQ(2)*VRT(2,N)
2025 2*IQ(3)*VRT(3,N))+TR(N)
2026 WRITE(6,3476) MS,(IQ(N),N=1,3),(ISS(N,MS),N=1,3)
2027 3476 FORMAT(2H *,T6,I3,T13,F8.3,2(' ',F8.3),T42,F8.3,
2028 22(' ',F8.3),T79,1H*)
2029 3473 CONTINUE
2030 WRITE(6,3006)
2031 WRITE(6,3006)
2032 WRITE(6,3485)
2033 3485 FORMAT(2H *,T33,'CURRENT WEIGHTS',T79,1H*,/2H *,T7,'MS',T18,
2034 2'REAL',T31,'IMAG.',T46,'MAG.',T57,'PHASE',T79,1H*)
2035 WRITE(6,3486)
2036 3486 FORMAT(2H *,T6,3('-',),T17,6('-',),T30,7('-',),T45,6('-',),
2037 2T56,7('-',),T79,1H*)
2038 DO 3465 MS=1,MSX
2039 READ(5,*) WMS,WPS
2040 WS(MS)=CMPLX(WMS,WPS)
2041 WMM=BABS(CMPLX(WMS,WPS))
2042 WPP=DPR*BTAN2(WPS,WMS)
2043 WRITE(6,3466) MS,WMS,WPS,WMM,WPP
2044 3466 FORMAT(2H *,T6,I3,5X,E11.4,2X,E11.4,4X,E11.4,2X,F8.3,T79,1H*)
2045 3465 CONTINUE
2046 WRITE(6,3006)
2047 GO TO 3000
2048 C=====
2049 4450 CONTINUE
2050 C=== RM: COMMAND =====
2051 C$$$
2052 C$$$ PRADR=TOTAL POWER RADIATED IN WATTS
2053 C$$$
2054 READ(6,*) PRADR
2055 WRITE(6,3441) PRADR
2056 WRITE(6,3006)
2057 C$$$
2058 C$$$ MRX=NUMBER OF ANTENNA SEGMENTS
2059 C$$$
2060 LRCVR=.TRUE.
2061 LRMP=.TRUE.
2062 READ(5,*) MRX
2063 IF(MRX.GT.MRDX) WRITE(6,4477) MRX
2064 4477 FORMAT(' ***** NUMBER OF SEGMENTS= ',I3,
2065 2' PROGRAM ABORTS SINCE MAX. RECEIVER DIMENSION'
2066 3,' IS EXCEEDED. *****')
2067 IF(MRX.GT.MRDX) STOP
2068 WRITE(6,3451) MRX
2069 WRITE(6,3006)
2070 WRITE(6,3006)
2071 C$$$

```

```

2072 C### XRR(N,MR)=XYZ LOCATION OF MR-TH ANTENNA SEGMENT
2073 C###
2074 C### IMR(MR)=-1=ELECTRIC LINEAR ELEMENT WITH A UNIFORM DISTRIBUTION
2075 C###
2076 C### HR(MR)=LENGTH OF LINEAR ELEMENT
2077 C###
2078 C### THRZ,PHRZ=ORIENTATION ANGLES USED TO DEFINE
2079 C### LINEAR ELEMENT AXIS.
2080 C###
2081 C### YMR,WPR=REAL AND IMAGINARY CURRENT WEIGHT.
2082 C###
2083 WRITE(6,4458) LABEL(IUNIT)
2084 4458 FORMAT(2H *,T7,'MR',T13,'HR:',A6,T23,'HR:METERS',
2085 2I41,'INPUT: THR,PHR',T60,'ACTUAL: THR,PHR',T79,1H*)
2086 WRITE(6,3459)
2087 4459 FORMAT(2H *,T6,3('-'),T12,20('-'),T40,16('-'),T59,
2088 2I7('-'),T79,1H*)
2089 WRITE(6,3006)
2090 DO 4463 MR=1,MRX
2091 READ(5,*) (XRR(N,MR),N=1,3),HR(MR),THRZ,PHRZ
2092 MRA(1,MR)=0
2093 MRA(2,MR)=0
2094 IMR(MR)=-1
2095 HAWR(MR)=0.
2096 HRQ=HR(MR)
2097 HR(MR)=UNITS*HRQ
2098 TQ=90.-THRZ
2099 PQ=PHRZ
2100 XQ(1)=SIN(TQ*RPD)*COS(PQ*RPD)
2101 XQ(2)=SIN(TQ*RPD)*SIN(PQ*RPD)
2102 XQ(3)=COS(TQ*RPD)
2103 DO 4481 N=1,3
2104 4481 XQR(N)=XQ(1)*VRT(1,N)+XQ(2)*VRT(2,N)+XQ(3)*VRT(3,N)
2105 THRZ=DPR*BTAN2(SQRT(XQR(1)*XQR(1)+XQR(2)*XQR(2)),XQR(3))
2106 PHRZ=DPR*BTAN2(XQR(2),XQR(1))
2107 WRITE(6,3464) MR,HRQ,HR(MR),TQ,PQ,THRZ,PHRZ
2108 DO 4484 N=1,3
2109 4484 VXRR(3,N,MR)=XQR(N)
2110 VXRR(1,1,MR)=COS(THRZ*RPD)*COS(PHRZ*RPD)
2111 VXRR(1,2,MR)=COS(THRZ*RPD)*SIN(PHRZ*RPD)
2112 VXRR(1,3,MR)=-SIN(THRZ*RPD)
2113 VXRR(2,1,MR)=-SIN(PHRZ*RPD)
2114 VXRR(2,2,MR)=COS(PHRZ*RPD)
2115 VXRR(2,3,MR)=0.
2116 4463 CONTINUE
2117 WRITE(6,3006)
2118 WRITE(6,3006)
2119 WRITE(6,3006)
2120 WRITE(6,3464)
2121 3 WRITE(6,3006)
2122 WRITE(6,3006)
2123 WRITE(6,4456) LABEL(IUNIT)
2124 4456 FORMAT(2H *,T7,'MR',T14,'INPUT LOCATION IN ',A6,
2125 2I43,'ACTUAL LOCATION IN METERS',T79,'*')
2126 WRITE(6,3457)
2127 WRITE(6,3006)
2128 DO 4473 MR=1,MRX
2129 DO 4474 N=1,3
2130 4474 XQ(N)=XRR(N,MR)
2131 DO 4475 N=1,3
2132 4475 XRR(N,MR)=UNITS*(XQ(1)*VRT(1,N)+XQ(2)*VRT(2,N)
2133 2*XQ(3)*VRT(3,N))+TR(N)
2134 WRITE(6,3476) MR,(XQ(N),N=1,3),(XRR(N,MR),N=1,3)
2135 4473 CONTINUE

```

```

2136 WRITE(6,3006)
2137 WRITE(6,3006)
2138 WRITE(6,3485)
2139 WRITE(6,3486)
2140 DD 4465 MR=1,MRX
2141 READ(5,*) WMR,WPR
2142 WR(MR)=CMPLX(WMR,WPR)
2143 WMM=BABS(CMPLX(WMR,WPR))
2144 WPP=DPR+BTAN2(WPR,WMR)
2145 WRITE(6,3466) MR,WMR,WPR,WMM,WPP
2146 4465 CONTINUE
2147 WRITE(6,3006)
2148 GO TO 3000
2149 C=====
2150 3490 CONTINUE
2151 C=== NS: COMMAND =====
2152 C$$$
2153 C$$$ INITIALIZE SOURCE DATA.
2154 C$$$
2155 LSMP=.FALSE.
2156 MSX=0
2157 MSAT=0
2158 WRITE(6,3491)
2159 3491 FORMAT(2H *,5X,' THE SOURCE DATA IS INITIALIZED. ',T79,1H*/
2160 2,2H *,5X,' NO SOURCES ARE PRESENTLY IN THE PROBLEM. '
2161 3,T79,1H*)
2162 GO TO 3000
2163 C=====
2164 3495 CONTINUE
2165 C=== NR: COMMAND =====
2166 C$$$
2167 C$$$ INITIALIZE RECEIVER DATA
2168 C$$$
2169 LRCVR=.FALSE.
2170 LRMP=.FALSE.
2171 MRX=0
2172 WRITE(6,3496)
2173 3496 FORMAT(2H *,5X,' THE RECEIVER DATA IS INITIALIZED. '
2174 2,T79,1H*/2H *,5X,' NO RECEIVERS ARE PRESENTLY IN THE'
2175 3,' PROBLEM. ',T79,1H*)
2176 GO TO 3000
2177 C=====
2178 3500 CONTINUE
2179 C=== LP: COMMAND =====
2180 C$$$
2181 C$$$ LWRITE=TRUE IF LINE PRINTER OUTPUT OF DATA IS DESIRED
2182 C$$$
2183 READ(5,*) LWRITE
2184 IF(.NOT.LWRITE) WRITE(6,6505)
2185 6505 FORMAT(2H *,5X,'NO LINE PRINTER OUTPUT',T79,1H*)
2186 IF(.NOT.LWRITE) GO TO 3000
2187 WRITE(6,3501)
2188 3501 FORMAT(2H *,5X,' DATA WILL BE OUTPUT ON LINE PRINTER !!!',
2189 2T79,1H*)
2190 GO TO 3000
2191 C=====
2192 3650 CONTINUE
2193 C=== VP: COMMAND =====
2194 C$$$
2195 C$$$ VOLUMETRIC DUMP FOR PLOTS
2196 C$$$
2197 READ(5,*) LVPLT
2198 IF(.NOT.LVPLT) WRITE(6,6606)
2199 IF(.NOT.LVPLT) GO TO 3000

```

```

2200          GO TO 3651
2201 C-----
2202 3600 CONTINUE
2203 C== PP:  COMMAND  -----
2204 C$$$
2205 C$$$ LPLT=TRUE IF PEN PLOTTER OUTPUT IS DESIRED
2206 C$$$
2207          READ(5,*) LPLT
2208          IF(.NOT.LPLT) WRITE(6,6606)
2209 6606  FORMAT(2H *,5X,'NO PEN PLOT DESIRED',T79,1H*)
2210          IF(.NOT.LPLT) GO TO 3000
2211 C$$$
2212 C$$$ IF LPLT=TRUE READ IN DIMENSIONS
2213 C$$$
2214 C$$$ LPPREC = TRUE IMPLIES RECTANGULAR PLOT
2215 C$$$ PPXL = LENGTH OF X-AXIS (ANGLE AXIS)
2216 C$$$ PPYL = LENGTH OF Y-AXIS (DB AXIS)
2217 C$$$
2218 C$$$ LPPREC = FALSE IMPLIES POLAR PLOT
2219 C$$$ PPXL = ANGULAR POSITION OF X-AXIS
2220 C$$$ PPYL = RADIUS OF GRID
2221 C$$$
2222 3651  READ(5,*) LPPREC,PPXL,PPYL
2223 C$$$
2224 C$$$ PPXB = BEGINNING VALUE OF X-AXIS
2225 C$$$ PPXE = END VALUE OF X-AXIS
2226 C$$$ PPXS = STEP SIZE OF X-AXIS GRID MARKS
2227 C$$$
2228          READ(5,*) PPXB,PPXE,PPXS
2229 C$$$
2230 C$$$ PPYB = BEGINNING VALUE OF Y-AXIS
2231 C$$$ PPYE = END VALUE OF Y-AXIS
2232 C$$$ PPYS = STEP SIZE OF Y-AXIS GRID MARKS
2233 C$$$
2234          READ(5,*) PPYB,PPYE,PPYS
2235          WRITE(6,3602)
2236 3602  FORMAT(2H *,5X,'DATA WILL BE OUTPUT FOR A PLOT !!!'
2237        2,T79,1H*)
2238          WRITE(6,3006)
2239          WRITE(6,3603) LPPREC,PPXL,PPYL
2240 3603  FORMAT(2H *,5X,' LPPREC= ',L2,5X,'PPXL= ',F10.5,5X,
2241        2'PPYL= ',F10.5,T79,1H*)
2242          WRITE(6,3604) PPXB,PPXE,PPXS
2243 3604  FORMAT(2H *,5X,' PPXB= ',F10.5,5X,'PPXE= ',F10.5,5X,
2244        2'PPXS= ',F10.5,T79,1H*)
2245          WRITE(6,3605) PPYB,PPYE,PPYS
2246 3605  FORMAT(2H *,5X,' PPYB= ',F10.5,5X,'PPYE= ',F10.5,5X,
2247        2'PPYS= ',F10.5,T79,1H*)
2248          IF(LPLT) GO TO 3000
2249          WRITE(6,3006)

2250 C$$$
2251 C$$$ IVIYP=TYPE OF RESULTS OUTPUT
2252 C$$$ IVIYP=1 ELECTRIC FIELD OUTPUT
2253 C$$$ IVIYP=2 MAGNETIC FIELD OUTPUT
2254 C$$$ IVIYP=3 BOTH ELECTRIC AND MAGNETIC FIELDS OUTPUT
2255 C$$$ COUPLING IS OUTPUT IF RECEIVER IS DEFINED FOR ANY IVIYP
2256 C$$$
2257 C$$$ IVPOL=POLARIZATION OF RESULTS OUTPUT
2258 C$$$ IVPOL=1,2,3 THEN R,THETA,PHI OR X,Y,Z RESPECTIVELY IS OUTPUT
2259 C$$$ IVPOL=4 THEN R-THETA OR X-Y ARE OUTPUT
2260 C$$$ IVPOL=5 THEN R-PHI OR X-Z ARE OUTPUT
2261 C$$$ IVPOL=6 THEN THETA-PHI OR Y-Z ARE OUTPUT
2262 C$$$ IVPOL=7 THEN R, THETA AND PHI OR X, Y AND Z ARE OUTPUT
2263 C$$$ COUPLING HAS NO POLARIZATION

```

```

2264 C###
2265 READ(5,*) IVIYP,IVPOL
2266 WRITE(6,3655) IVIYP,IVPOL
2267 3655 FORMAT(2H *,5X,' IVIYP= ',I2,' IVPOL= ',I2,T79,1H*)
2268 GO TO 3000
2269 C=====
2270 3700 CONTINUE
2271 C=== GP:      COMMAND      =====
2272 C###
2273 C### INFINITE GROUND PLANE EFFECT INCLUDED.
2274 C###
2275 LGRND=.TRUE.
2276 DO 3702 N=1,3
2277 XX(N,1,MPDX)=1.E5*(VRT(1,N)+VRT(2,N))+TR(N)
2278 XX(N,2,MPDX)=1.E5*(-VRT(1,N)+VRT(2,N))+TR(N)
2279 XX(N,3,MPDX)=1.E5*(-VRT(1,N)-VRT(2,N))+TR(N)
2280 3702 XX(N,4,MPDX)=1.E5*(VRT(1,N)-VRT(2,N))+TR(N)
2281 C###
2282 C### LSLAB= 0  IMPLIES METAL PLATE, AND
2283 C###      =-3  IMPLIES DIELECTRIC HALF SPACE
2284 C###
2285 C### NOTE:  IF DIELECTRIC COVERED, ONE MUST READ DIELECTRIC DATA.
2286 C###
2287 READ(5,*) LSLAB(MPDX)
2288 IF(LSLAB(MPDX).EQ.0) WRITE(6,3706)
2289 3706 FORMAT(2H *,5X,'PERFECTLY CONDUCTING',T79,1H*)
2290 IF(LSLAB(MPDX).NE.0) WRITE(6,3707)
2291 3707 FORMAT(2H *,5X,'SEMI-INFINITELY THICK DIELECTRIC',T79,1H*)
2292 WRITE(6,3701)
2293 3701 FORMAT(2H *,5X,'INFINITE GROUND PLANE INSERTED IN',
2294 2' STRUCTURE !!!',T79,1H*)
2295 WRITE(6,3006)
2296 WRITE(6,3703) (TR(N),N=1,3)
2297 3703 FORMAT(2H *,5X,'THE ORIGIN IS AT ',F12.6,',',F12.6
2298 2,',',F12.6,' METERS',T79,1H*)
2299 WRITE(6,3006)
2300 WRITE(6,3704) (VRT(3,N),N=1,3)
2301 3704 FORMAT(2H *,5X,'THE NORMAL IS ',F12.6,',',F12.6,',',
2302 2,F12.6,T79,1H*)
2303 IF(LSLAB(MPDX).EQ.0) GO TO 3000
2304 NSLAB(1)=1
2305 DSLAB(1,MPDX)=0.
2306 LSLAB(MPDX)=-3
2307 C###
2308 C### ERLAB(1,MPDX)=RELATIVE DIELECTRIC CONSTANT
2309 C###
2310 C### TESLAB(1,MPDX)=DIELECTRIC LOSS TANGENT
2311 C###
2312 C### URSLAB(1,MPDX)=RELATIVE PERMEABILITY CONSTANT
2313 C###
2314 C### TMSLAB(1,MPDX)=PERMEABILITY LOSS TANGENT
2315 C###
2316 READ(5,*) ERLAB(1,MPDX),TESLAB(1,MPDX)
2317 2,URSLAB(1,MPDX),TMSLAB(1,MPDX)
2318 WRITE(6,3006)
2319 WRITE(6,3708)
2320 3708 FORMAT(2H *,5X,'DIELECTRIC',3X,'LOSS',4X,
2321 2'PERMITTIVITY',3X,'LOSS',T79,1H*./,
2322 32H *,6X,'CONSTANT',3X,'TANGENT',
2323 44X,'CONSTANT',3X,'TANGENT',T79,1H*./,
2324 52H *,5X,'-----',2X,'-----',
2325 62X,'-----',2X,'-----',T79,1H*)
2326 WRITE(6,3709) ERLAB(1,MPDX),TESLAB(1,MPDX)
2327 2,URSLAB(1,MPDX),TMSLAB(1,MPDX)

```

```

2328 3709 FORMAT(2H *,6X,F10.4,2X,F7.4,2X,F11.4,2X,
2329 2F7.4,T79,1H+)
2330 GO TO 3000
2331 C=====
2332 3750 CONTINUE
2333 C=== NG: COMMAND =====
2334 C$$$
2335 C$$$ INITIALIZE GROUND PLANE DATA.
2336 C$$$
2337 LGRND=.FALSE.
2338 WRITE(6,3751)
2339 3751 FORMAT(2H *,5X,' GROUND PLANE DATA IS INITIALIZED. ',T79,1H+/
2340 2,2H *,5X,' NO GROUND PLANE IS PRESENTLY IN THE PROBLEM. '
2341 3,T79,1H+)
2342 GO TO 3000
2343 C=====
2344 3900 CONTINUE
2345 C=== RI: COMMAND =====
2346 C$$$
2347 C$$$ TR(N)=LINEAR TRANSLATION OF COORDINATES FROM THE FIXED
2348 C$$$ COORDINATES WHICH IS ORIGINALLY SET UP BY OPERATOR.
2349 C$$$
2350 READ(5,*) (TR(N),N=1,3)
2351 WRITE(6,3901) LABEL(IUNIT),(TR(N),N=1,3)
2352 3901 FORMAT(2H *,5X,'TRANSLATION IN ',A6,': TR(1)=' ,F8.3,
2353 2' TR(2)=' ,F8.3, ' TR(3)=' ,F8.3,T79,1H+)
2354 DO 3920 N=1,3
2355 3920 TR(N)=TR(N)*UNITS
2356 WRITE(6,3006)
2357 IF(IUNIT.NE.1) WRITE(6,3901) LABEL(1),(TR(N),N=1,3)
2358 IF(IUNIT.NE.1) WRITE(6,3006)
2359 WRITE(6,3006)
2360 C$$$
2361 C$$$ THZP,PHZP=ORIENTATION OF THE VRI(3,N) AXIS RELATIVE TO THE
2362 C$$$ FIXED COORDINATE SYSTEM.
2363 C$$$
2364 C$$$ THXP,PHXP=ORIENTATION OF THE VRI(1,N) AXIS RELATIVE TO THE
2365 C$$$ FIXED COORDINATE SYSTEM.
2366 C$$$
2367 READ(5,*) THZP,PHZP,THXP,PHXP
2368 VRI(3,1)=SIN(THZP*RPD)*COS(PHZP*RPD)
2369 VRI(3,2)=SIN(THZP*RPD)*SIN(PHZP*RPD)
2370 VRI(3,3)=COS(THZP*RPD)
2371 VRI(1,1)=SIN(THXP*RPD)*COS(PHXP*RPD)
2372 VRI(1,2)=SIN(THXP*RPD)*SIN(PHXP*RPD)
2373 VRI(1,3)=COS(THXP*RPD)
2374 C!!! INSURE VRI(1,N) IS PERPENDICULAR TO VRI(3,N)
2375 DZX=VRI(3,1)*VRI(1,1)+VRI(3,2)*VRI(1,2)+VRI(3,3)*VRI(1,3)
2376 IF(ABS(DZX).GT.0.1) WRITE(6,3903)
2377 3903 FORMAT(' *** PROGRAM ABORTS IN ROTATE SECTION IN THAT THE',
2378 2' COORDINATES ARE NOT ORTHOGONAL!!! ***')
2379 IF(ABS(DZX).GT.0.1) STOP
2380 VRI(1,1)=VRI(1,1)-VRI(3,1)*DZX
2381 VRI(1,2)=VRI(1,2)-VRI(3,2)*DZX
2382 VRI(1,3)=VRI(1,3)-VRI(3,3)*DZX
2383 DOT=VRI(1,1)*VRI(1,1)+VRI(1,2)*VRI(1,2)+VRI(1,3)*VRI(1,3)
2384 DOT=SQRT(DOT)
2385 VRI(1,1)=VRI(1,1)/DOT
2386 VRI(1,2)=VRI(1,2)/DOT
2387 VRI(1,3)=VRI(1,3)/DOT
2388 VRI(2,1)=VRI(3,2)*VRI(1,3)-VRI(3,3)*VRI(1,2)
2389 VRI(2,2)=VRI(3,3)*VRI(1,1)-VRI(3,1)*VRI(1,3)
2390 VRI(2,3)=VRI(3,1)*VRI(1,2)-VRI(3,2)*VRI(1,1)
2391 WRITE(6,3931)

```



```

2392 3931 FORMAT(2H *,5X,'THE FOLLOWING ROTATIONS ARE USED FOR ALL',
2393 2' SUBSEQUENT INPUTS:',T79,1H*)
2394 DO 3932 NI=1,3
2395 WRITE(6,3006)
2396 3932 WRITE(6,3933) (NI,NJ,VRT(NI,NJ),NJ=1,3)
2397 3933 FORMAT(2H *,1X,3(2X,'VRT(',I1,',',I1,')=' ,F9.5),T79,1H*)
2398 GO TO 3000
2399 C=====
2400 4000 CONTINUE
2401 C=== CG: AND CC: COMMAND =====
2402 C$$$
2403 C$$$ CYLINDER GEOMETRY INPUT
2404 C$$$
2405 LCYL=.TRUE.
2406 MCX=MCX+1
2407 MC=MCX
2408 IF(MCX.GT.MCDX) WRITE(6,6311) MCX
2409 6311 FORMAT(' ***** NUMBER OF CYLINDERS= ',I3,'PROGRAM',
2410 2' ABORTS SINCE MAX. CYLINDER DIMENSION IS EXCEEDED. *****')
2411 IF(MCX.GT.MCDX) STOP
2412 C$$$
2413 C$$$ XCL(N,MC)=XYZ LOCATION OF THE ORIGIN OF THE MC-TH CYLINDER
2414 C$$$
2415 READ(5,*) (XCL(N,MC),N=1,3)
2416 DO 6301 N=1,3
2417 6301 XQ(N)=XCL(N,MC)
2418 DO 6302 N=1,3
2419 6302 XCL(N,MC)=UNITS*(XQ(1)*VRT(1,N)+XQ(2)*VRT(2,N)
2420 2*XQ(3)*VRT(3,N))+TR(N)
2421 WRITE(6,6308) LABEL(IUNIT)
2422 6308 FORMAT(2H *,I5,'CYLINDER#',T17,'INPUT LOCATION IN ',A6
2423 2,I46,'ACTUAL LOCATION IN METERS',T79,1H*)
2424 WRITE(6,3422)
2425 WRITE(6,3006)
2426 WRITE(6,3426) MC, (XQ(N),N=1,3), (XCL(N,MC),N=1,3)
2427 WRITE(6,3006)
2428 WRITE(6,3006)
2429 C$$$
2430 C$$$ TCLZ,PCLZ=ORIENTATION OF THE CYLINDER AXIS
2431 C$$$
2432 C$$$ TCLX,PCLX=ORIENTATION OF THE CYLINDER'S X-AXIS
2433 C$$$
2434 READ(5,*) TCLZ,PCLZ,TCLX,PCLX
2435 XQ(1)=SIN(TCLZ*RPD)*COS(PCLZ*RPD)
2436 XQ(2)=SIN(TCLZ*RPD)*SIN(PCLZ*RPD)
2437 XQ(3)=COS(TCLZ*RPD)
2438 DO 6303 N=1,3
2439 6303 VCL(3,N,MC)=XQ(1)*VRT(1,N)+XQ(2)*VRT(2,N)+XQ(3)*VRT(3,N)
2440 XQ(1)=SIN(TCLX*RPD)*COS(PCLX*RPD)
2441 XQ(2)=SIN(TCLX*RPD)*SIN(PCLX*RPD)
2442 XQ(3)=COS(TCLX*RPD)
2443 DO 6304 N=1,3
2444 6304 VCL(1,N,MC)=XQ(1)*VRT(1,N)+XQ(2)*VRT(2,N)+XQ(3)*VRT(3,N)
2445 DZX=VCL(1,1,MC)+VCL(3,1,MC)+VCL(1,2,MC)+VCL(3,2,MC)
2446 2+VCL(1,3,MC)+VCL(3,3,MC)
2447 IF(ABS(DZX).GT.0.1) WRITE(6,6305)
2448 6305 FORMAT(' *** PROGRAM ABORTS THE COORDINATES ARE NOT',
2449 2' ORTHOGONAL!!! ***')
2450 IF(ABS(DZX).GT.0.1) STOP
2451 VCL(1,1,MC)=VCL(1,1,MC)-VCL(3,1,MC)*DZX
2452 VCL(1,2,MC)=VCL(1,2,MC)-VCL(3,2,MC)*DZX
2453 VCL(1,3,MC)=VCL(1,3,MC)-VCL(3,3,MC)*DZX
2454 DOT=VCL(1,1,MC)+VCL(1,1,MC)+VCL(1,2,MC)+VCL(1,2,MC)
2455 2+VCL(1,3,MC)+VCL(1,3,MC)

```

```

2456      DOT=SQRT(DOT)
2457      VCL(1,1,MC)=VCL(1,1,MC)/DOT
2458      VCL(1,2,MC)=VCL(1,2,MC)/DOT
2459      VCL(1,3,MC)=VCL(1,3,MC)/DOT
2460      VCL(2,1,MC)=VCL(3,2,MC)*VCL(1,3,MC)-VCL(3,3,MC)*VCL(1,2,MC)
2461      VCL(2,2,MC)=VCL(3,3,MC)*VCL(1,1,MC)-VCL(3,1,MC)*VCL(1,3,MC)
2462      VCL(2,3,MC)=VCL(3,1,MC)*VCL(1,2,MC)-VCL(3,2,MC)*VCL(1,1,MC)
2463      WRITE(6,6309)
2464  6309  FORMAT(2H *,5X,'THE FOLLOWING CYLINDER ALIGNMENT IS USED:'
2465      2,T79,1H*)
2466      DD 6308 NI=1,3
2467      WRITE(6,3006)
2468  6306  WRITE(6,6307) (NI,NJ,VCL(NI,NJ,MC),NJ=1,3)
2469  6307  FORMAT(2H *.1X,3(2X,'VCL(',I1,'.',I1,')=' ,F9.6),T79,1H*)
2470      C###
2471      C###  AC=RADIUS OF ELLIPSE ON X CYLINDER AXIS
2472      C###  BC=RADIUS OF ELLIPSE ON Y CYLINDER AXIS
2473      C###
2474      C###  ZCN,THIN=MOST NEGATIVE ENDCAP'S Z COMPONENT
2475      C###  AND ANGLE OF SURFACE WITH THE CYLINDER AXIS
2476      C###  ZCP,THIP=MOST POSITIVE ENDCAP'S Z COMPONENT
2477      C###  AND ANGLE OF SURFACE WITH THE CYLINDER AXIS
2478      C###
2479      IF(IR(1).EQ.II(40)) GO TO 6400
2480      NEC(MC)=2
2481      READ(5,*) AC(1,MC),BC(1,MC)
2482      READ(5,*) ZCN,THIN,ZCP,THIP
2483      AAD=AC(1,MC)
2484      BBO=BC(1,MC)
2485      AC(1,MC)=AC(1,MC)*UNITS
2486      BC(1,MC)=BC(1,MC)*UNITS
2487      AC(2,MC)=AC(1,MC)
2488      BC(2,MC)=BC(1,MC)
2489      ZC(2,MC)=ZCN*UNITS
2490      TCR(2,MC)=THIN*RPD
2491      ZC(1,MC)=ZCP*UNITS
2492      TCR(1,MC)=THIP*RPD
2493      WRITE(6,3006)
2494      WRITE(6,3006)
2495      WRITE(6,6310) LABEL(IUNIT),AAD
2496  6310  FORMAT(2H *,5X,'X AXIS DIMENSION IN ',
2497      2A6,'=',F8.3,T79,1H*)
2498      WRITE(6,3006)
2499      IF(IUNIT.NE.1) WRITE(6,6310) LABEL(1),AC(1,MC)

2500      IF(IUNIT.NE.1) WRITE(6,3006)
2501      WRITE(6,3006)
2502      WRITE(6,6320) LABEL(IUNIT),BBO
2503  6320  FORMAT(2H *,5X,'Y AXIS DIMENSION IN ',
2504      2A6,'=',F8.3,T79,1H*)
2505      WRITE(6,3006)
2506      IF(IUNIT.NE.1) WRITE(6,6320) LABEL(1),BC(1,MC)
2507      IF(IUNIT.NE.1) WRITE(6,3006)
2508      WRITE(6,3006)
2509      WRITE(6,6330) LABEL(IUNIT),ZCN
2510  6330  FORMAT(2H *,5X,'MOST NEGATIVE END CAP Z COMPONENT IN ',
2511      2A6,'=',F8.3,T79,1H*)
2512      WRITE(6,3006)
2513      IF(IUNIT.NE.1) WRITE(6,6330) LABEL(1),ZC(2,MC)
2514      IF(IUNIT.NE.1) WRITE(6,3006)
2515      WRITE(6,6340) THIN
2516  6340  FORMAT(2H *,5X,'ANGLE OF NEG. END CAP SURFACE WITH NEG.',
2517      2' CYL. AXIS ', '=',F8.3,T79,1H*)
2518      WRITE(6,3006)
2519      WRITE(6,3006)

```

```

2520      WRITE(6,6350) LABEL(IUNIT),ZCP
2521 6350  FORMAT(2H *,5X,'MOST POSITIVE END CAP Z COMPONENT IN ',
2522      2A6,'=',F8.3,I79,1H+)
2523      WRITE(6,3006)
2524      IF(IUNIT.NE.1) WRITE(6,6350) LABEL(1),ZC(1,MC)
2525      IF(IUNIT.NE.1) WRITE(6,3006)
2526      WRITE(6,6360) THTP
2527 6360  FORMAT(2H *,5X,'ANGLE OF POS. END CAP SURFACE WITH POS. ',
2528      2' CYL. AXIS ',',F8.3,I79,1H+)
2529      GO TO 3000
2530 6400  CONTINUE
2531      READ(5,*) NEC(MC)
2532      IF(NEC(MC).GT.NCDX) STOP
2533      NECX=NEC(MC)
2534      DO 6410 MC=1,NECX
2535      READ(5,*) AC(NC,MC),BC(NC,MC),ZC(NC,MC)
2536      AAO=AC(NC,MC)
2537      BBD=BC(NC,MC)
2538      ZCN=ZC(NC,MC)
2539      AC(NC,MC)=AC(NC,MC)*UNITS
2540      BC(NC,MC)=BC(NC,MC)*UNITS
2541      ZC(NC,MC)=ZC(NC,MC)*UNITS
2542      TCR(NC,MC)=0.5*PI
2543      WRITE(6,3305) MC,NC,AAO,BBD,ZCN,AC(NC,MC),BC(NC,MC),ZC(NC,MC)
2544 6410  CONTINUE
2545      GO TO 3000
2546  C=====
2547  4050  CONTINUE
2548  C===  NC:      COMMAND      =====
2549  C###
2550  C###  INITIALIZE CYLINDER DATA.
2551  C###
2552      LCYL=.FALSE.
2553      MCX=0
2554      WRITE(6,4051)
2555  4051  FORMAT(2H *,5X,' CYLINDER DATA IS INITIALIZED. ',I79,1H+/
2556      2,2H *,5X,' NO CYLINDER IS PRESENTLY IN THE PROBLEM. '
2557      3,I79,1H+)
2558      GO TO 3000
2559  C=====
2560  997  CONTINUE
2561  C===  EN:      COMMAND      =====
2562  C###
2563  C###  END PROGRAM
2564  C###
2565      WRITE(6,3006)
2566      WRITE(6,3006)
2567      WRITE(6,3005)
2568      STOP
2569  C=====
2570  3800  CONTINUE
2571  C===  IQ:      COMMAND      =====
2572  C###
2573  C###  EXECUTE PROGRAM
2574  C###
2575      WRITE(6,3006)
2576      WRITE(6,3006)
2577  C111
2578  C111  2.  INITIALIZATION SECTION
2579  C111
2580      WL=.2997925/FRQG
2581      WK=TPI/WL
2582      WRITE(6,3005)
2583      MPXR=MPX

```

```

2584 C111 GROUND PLANE IS ANOTHER PLATE IN SOLUTION.
2585 IF(LGRND) MPXR=MPX+1
2586 IF(MPXR.GT.MPDX) WRITE(6,501) MPXR
2587 IF(MPXR.GT.MPDX) STOP
2588 IF(.NOT.LGRND) GO TO 3801
2589 LPLA=.TRUE.
2590 MEP(MPXR)=4
2591 DO 3802 I=1,4
2592 DO 3802 N=1,3
2593 3802 XX(N,I,MPXR)=XX(N,I,MPDX)
2594 LSLAB(MPXR)=LSLAB(MPDX)
2595 DSLAB(1,MPXR)=DSLAB(1,MPDX)
2596 ERSLAB(1,MPXR)=ERSLAB(1,MPDX)
2597 TESLAB(1,MPXR)=TESLAB(1,MPDX)
2598 URSLAB(1,MPXR)=URSLAB(1,MPDX)
2599 TMSLAB(1,MPXR)=TMSLAB(1,MPDX)
2600 3801 CONTINUE
2601 IF(MPXR.EQ.0) LPLA=.FALSE.
2602 IF(LPLA) CALL GEOM
2603 C111 MAKE PATTERN INFORMATION MATCH WITH SHADOW CODE
2604 IF(IVPN.GT.0) THEN
2605     NVFT=NPV
2606     NVFP=NPN
2607 ELSE
2608     NVFT=NPN
2609     NVFP=NPV
2610 ENDIF
2611 COLS=NVFT
2612 THET1=IYS*RPD
2613 RESTH=IYI*RPD
2614 THET2=THET1+(COLS-1)*RESTH
2615 ROWS=NVFP
2616 PH1=PZS*RPD
2617 RESPH=PZI*RPD
2618 PH2=PH1+(ROWS-1)*RESPH
2619 C111 MAKE SOURCE INFORMATION FOR THE FIRST ONE MATCH WITH
2620 C111 SHADOW CODE
2621 DO 3806 N=1,3
2622 3806 ANTENN(N)=XSS(N,1)
2623 RETURN
2624 END

```

FUNCTION BABS

This is function BABS. It is used to obtain complex absolute values without runtime numerical errors.

```
0001 C-----  
0002     FUNCTION BABS(Z)  
0003 C!!!  
0004 C!!! THIS ROUTINE IS USED TO GIVE COMPLEX ABSOLUTE VALUES. IT IS  
0005 C!!! USED RATHER THAN STANDARD ROUTINES TO AVOID EXECUTION  
0006 C!!! ERRORS.  
0007 C!!!  
0008     COMPLEX Z  
0009     COMMON/LIMIT/SML,SMLR,SMLT,BIG  
0010     X=ABS(REAL(Z))  
0011     Y=ABS(AIMAG(Z))  
0012     IF(X.LT.SMLT.AND.Y.LT.SMLT) GO TO 10  
0013     BABS=CABS(Z)  
0014     RETURN  
0015 10    BABS=SMLT  
0016     RETURN  
0017     END
```

BLOCK DATA

This is contant block data.

```
0001  C-----  
0002      BLOCK DATA  
0003  C!!!  
0004  C!!! LOAD COMMONLY USED DATA INTO COMMON AREA.  
0005  C!!!  
0006      COMPLEX CJ,CPI4  
0007      COMMON/PIS/PI,TPI,DPR,RPD  
0008      COMMON/COMP/CJ,CPI4  
0009      COMMON/LIMIT/SML,SMLR,SMLT,BIG  
0010      DATA PI,TPI,DPR,RPD/3.14159265,6.28318531,57.2957795,  
0011      20.0174532925/  
0012      DATA CJ,CPI4/(0.,1.),(.70710678,-.70710678)/  
0013      DATA SML,SMLR,SMLT,BIG/1.E-3,1.E-5,1.E-10,1.E30/  
0014      END
```

FUNCTION BTAN2

This function is identical to the intrinsic fortran ATAN2 function, except it avoids runtime numerical errors.

```
0001 C-----  
0002 FUNCTION BTAN2(Y,X)  
0003 C!!!  
0004 C!!! THIS ROUTINE IS USED TO COMPUTE THE ARCTANGENT. IT IS  
0005 C!!! SIMILAR TO ATAN2 EXCEPT IT AVOIDS THE RUN TIME ERRORS.  
0006 C!!!  
0007 COMMON/PIS/PI,TPI,DPR,RPD  
0008 COMMON/LIMIT/SML,SMLR,SMLI,BIG  
0009 IF(ABS(X).GT.SMLI) GO TO 50  
0010 IF(ABS(Y).GT.SMLT) GO TO 10  
0011 BTAN2=0.  
0012 RETURN  
0013 10 BTAN2=0.5*PI  
0014 IF(Y.LT.0.) BTAN2=-BTAN2  
0015 RETURN  
0016 50 BTAN2=ATAN2(Y,X)  
0017 RETURN  
0018 END
```

SUBROUTINE CAPINT

This routine is used to determine if a ray strikes an elliptic cylinder endcap.

```

0001  C-----
0002      SUBROUTINE CAPINT(XIS,D,DHIT,MD,LHIT,MH)
0003  C!!!
0004  C!!! DOES RAY HIT ENDCAP?
0005  C!!!
0006      INCLUDE 'SHACOM.FOR'
0172      DIMENSION XIS(3),D(3),XI(3),XISC(3),DC(3)
0173      LOGICAL LHIT,LDEBUG,LTEST,LWARN
0174      COMMON/TEST/LDEBUG,LTEST,LWARN
0175      COMMON/LIMIT/SML,SMLR,SMLT,BIG
0176      COMMON/WAVE/WX,WL
0177      LHIT=.FALSE.
0178      DHIT=0.
0179  C!!! STEP THROUGH CYLINDERS
0180      DO 40 MCC=1,MCX
0181      IF(MH.LT.0 .AND. IABS(MH).NE.MCC) GO TO 40
0182      IF(MH.GT.0 .AND. MH.EQ.MCC) GO TO 40
0183      CALL CYLROD(D,DC,1,MCC)
0184      CALL CYLROD(XIS,XISC,2,MCC)
0185  C!!! STEP THRU ENDCAPS
0186      NECX=NEC(MCC)
0187      DO 40 MN=1,NECX
0188      IF(MD.LT.0 .AND. IABS(MD).NE.MN) GO TO 40
0189      IF(MD.GT.0 .AND. MD.EQ.MN) GO TO 40
0190      A=AC(MN,MCC)
0191      B=BC(MN,MCC)
0192      CNC=COS(OCR(MN,MCC))
0193      SNC=SIN(OCR(MN,MCC))
0194      AN=-XISC(1)*CNC+(XISC(3)-ZC(MN,MCC))*SNC
0195      DN=-CNC*DC(1)+SNC*DC(3)
0196  C!!! DOES RAY HIT ENDCAP PLANE?
0197      IF(AN*DN.GE.0.) GO TO 40
0198  C!!! COMPUTE POINT XI, WHERE RAY HITS ENDCAP PLANE
0199      DO 10 N=1,3
0200      XI(N)=XISC(N)-AN*DC(N)/DN
0201      RHOT=XI(1)*XI(1)+XI(2)*XI(2)
0202      2+(XI(3)-ZC(MN,MCC))*(XI(3)-ZC(MN,MCC))
0203      RHOT=SQRT(RHOT)
0204      AE=A/SNC
0205  C!!! IS HIT POINT ON ENDCAP?
0206      IF(RHOT.GT.AE .AND. RHOT.GT.B) GO TO 40
0207      IF(RHOT.LT.AE .AND. RHOT.LT.B) GO TO 20
0208      VE=BTAN2(A*XI(2),B*XI(1))
0209      CVE=COS(VE)
0210      SVE=SIN(VE)
0211      RHO=SQRT(AE*AE*CVE*CVE+B*B*SVE*SVE)
0212      IF(RHOT.GT.RHO) GO TO 40
0213      20 CONTINUE
0214  C!!! CALCULATE DHT, THE DISTANCE FROM SOURCE TO HIT POINT
0215      DHT=0.
0216      DO 30 N=1,3
0217      DHT=DHT+(XI(N)-XISC(N))*(XI(N)-XISC(N))
0218      DHT=SQRT(DHT)+SMLR*WL
0219      IF(LHIT .AND. (DHT.GT.DHIT)) GO TO 40
0220      LHIT=.TRUE.
0221      DHIT=DHT
0222      IF(MD.LE.0) GO TO 40

```



```
0223      CALL CYLROT(IIS,IT,-2,MCC)
0224      40      CONTINUE
0225      IF(LTEST) THEN
0226          WRITE(6,900)
0227      900      FORMAT(/,' TESTING CAPINT SUBROUTINE')
0228          WRITE(6,*) IIS
0229          WRITE(6,*) D
0230          WRITE(6,*) DHIT,MD,LHIT,MH
0231      ENDIF
0232      RETURN
0233      END
```

SUBROUTINE CYLINT

This routine is used to determine if a ray strikes an elliptic cylinder.

```

0001 C-----
0002     SUBROUTINE CYLINT(XS,D,DHIT,LHIT,LBDF,MH)
0003     C!!!
0004     C!!! DOES RAY HIT CYLINDER?
0005     C!!!
0006     INCLUDE 'SHACOM.FOR'
0172     DIMENSION D(3),XS(3),VID(2),BTD(4),CTC(2),DC(3),XSC(3)
0173     LOGICAL LHIT,LBDF,LPLA,LCYL,LDEBUG,LTEST,LHT,LWARN
0174     COMMON/LPLCY/LPLA,LCYL
0175     COMMON/TEST/LDEBUG,LTEST,LWARN
0176     COMMON/LIMIT/SML,SMLR,SMLT,BIG
0177     COMMON/WAVE/WK,WL
0178     LHIT=.FALSE.
0179     DHIT=0.
0180     IF(.NOT.LCYL) GO TO 100
0181     C!!! STEP THRU CYLINDETS
0182     DO 50 MCC=1,MCX
0183     IF(MH.LT.0 .AND. IABS(MH).NE.MCC) GO TO 50
0184     IF(MH.GT.0 .AND. MH.EQ.MCC) GO TO 50
0185     CALL CYLROT(XS,XSC,2,MCC)
0186     CALL CYLROT(D,DC,1,MCC)
0187     C!!! DOES RAY HIT CYLINDER SURFACE SECTION?
0188     PHSR=BTAN2(DC(2),DC(1))
0189     CPS=COS(PHSR)
0190     SPS=SIN(PHSR)
0191     RHOS=SQRT(XSC(1)*XSC(1)+XSC(2)*XSC(2))
0192     C!!! STEP THRU CYLINDER SECTIONS
0193     NECX=NEC(MCC)-1
0194     DO 40 NC=1,NECX
0195     NCP=NC+1
0196     A=AC(NC,MCC)
0197     B=BC(NC,MCC)
0198     C!!! PARAMETER FOR ELLIPTIC CYLINDER
0199     CTC(1)=COS(ICR(NC,MCC))/SIN(ICR(NC,MCC))
0200     CTC(2)=COS(ICR(NCP,MCC))/SIN(ICR(NCP,MCC))
0201     C!!! PARAMETERS FOR CONE FRUSTUMS SECTION
0202     ZC=ZC(NCP,MCC)-ZC(NC,MCC)
0203     IF(ABS(ZC).LT.SML*WL) GO TO 40
0204     TNJ=(AC(NCP,MCC)-AC(NC,MCC))/ZC
0205     FL=TNJ*(XSC(3)-ZC(NC,MCC))/A+1.
0206     C!!! RADII AT SOURCE LOCATION
0207     AL=A*FL
0208     BL=B*FL
0209     C!!! IS SOURCE INSIDE OF INFINITE CYLINDER?
0210     IF(RHOS.GT.AL .AND. RHOS.GT.BL) GO TO 5
0211     IF(RHOS.LT.AL .AND. RHOS.LT.BL) GO TO 30
0212     VE=BTAN2(AL*XSC(2),BL*XSC(1))
0213     CVE=COS(VE)
0214     SVE=SIN(VE)
0215     RHOE=SQRT(AL*AL+CVE*CVE+BL*BL+SVE*SVE)
0216     IF(RHOS.GT.RHOE) GO TO 5
0217     30 CONTINUE
0218     C!!! IS SOURCE INSIDE OF FINITE CYLINDER SECTION?
0219     IF(XSC(3).GT.(ZC(NC,MCC)+XSC(1)+CTC(1))) GO TO 40
0220     IF(XSC(3).LT.(ZC(NCP,MCC)+XSC(1)+CTC(2))) GO TO 40
0221     LHIT=.TRUE.
0222     GO TO 100

```

```

0223 6 CONTINUE
0224 C111 FIND COEFFICIENT OF EQUATION TO DETERMINE HIT POINT
0225 AA=A*A
0226 BB=B*B
0227 C111 PARTS FOR ALL ELLIPTIC CROSS SECTION TYPES
0228 CA=DC(1)*DC(1)/AA+DC(2)*DC(2)/BB
0229 CB=ISC(1)*DC(1)/AA+ISC(2)*DC(2)/BB
0230 CC=ISC(1)*ISC(1)/AA+ISC(2)*ISC(2)/BB
0231 C111 PARTS FOR CONE FRUSTUM SECTIONS
0232 CA=CA-TNJ*TNJ+DC(3)+DC(3)/AA
0233 CB=CB-TNJ*FL+DC(3)/A
0234 CC=CC-FL*FL
0235 C111 IS QUADRATIC SOLVABLE IN REAL SPACE?
0236 C111 IF NOT, NO HIT POINT ON CYLINDER SURFACE SECTION
0237 CT=CB*CB-CA*CC
0238 IF(CT.LE.O.) GO TO 40
0239 C111 DETERMINE TWO POSSIBLE HIT DISTANCES
0240 SCT=SQRT(CT)
0241 RHP=(-CB+SCT)/CA
0242 RHM=(-CB-SCT)/CA
0243 C111 NEAREST POSITIVE ONE IS TRUE HIT POINT
0244 IF(RHP.LT.O..AND.RHM.LT.O.) THEN
0245 GO TO 40
0246 ELSE
0247 IF(RHP.LT.O..OR.RHM.LT.O.) THEN
0248 RH=AMAX1(RHP,RHM)
0249 ELSE
0250 RH=AMIN1(RHP,RHM)
0251 ENDIF
0252 ENDF
0253 XPM=RH*DC(1)+ISC(1)
0254 ZPM=RH*DC(3)+ISC(3)
0255 C111 IS HIT POINT ON FINITE CYLINDER SECTION?
0256 IF(ZPM.GT.ZC(NC,MCC)+XPM+CTC(1) .OR.
0257 2ZPM.LT.ZC(NCP,MCC)+XPM+CTC(2)) GO TO 40
0258 C111 DISTANCE FROM SOURCE TO HIT
0259 DHT=RH+SMLR*WL
0260 C111 CHECK FOR NEAREST HIT POINT FOR DIFFERENT SECTIONS
0261 IF(LHIT .AND. (DHT.GT.DHIT)) GO TO 40
0262 LHIT=.TRUE.
0263 DHIT=DHT
0264 40 CONTINUE
0265 C111 CHECK TO SEE IF RAY HITS ENDCAPS
0266 CALL CAPINT(XS,D,DHT,O,LHT,-MCC)
0267 IF(.NOT.LHT) GO TO 50
0268 IF(LHIT .AND. (DHT.GT.DHIT)) GO TO 50
0269 LHIT=.TRUE.
0270 DHIT=DHT
0271 50 CONTINUE
0272 100 IF(LTEST) THEN
0273 WRITE(6,900)
0274 900 FORMAT(/,' TESTING CYLINT SUBROUTINE')
0275 WRITE(6,*) XS
0276 WRITE(6,*) D
0277 WRITE(6,*) DHIT,LHIT,LBDF,MH
0278 ENDF
0279 RETURN
0280 END

```

SUBROUTINE CYLROT

This routine performs vector transformations between the various cylinder coordinate systems and the reference coordinate system.

```
0001 C-----
0002 SUBROUTINE CYLROT(XREF,XCYL,N,MCL)
0003 C***
0004 C*** ROTATES AND OR TRANSLATES VECTORS IN OR OUT OF THE
0005 C*** CYLINDER COORDINATE SYSTEMS
0006 C***
0007 INCLUDE 'SHACOM.FOR'
0173 DIMENSION XREF(3),XCYL(3)
0174 IF(N.LT.0) GO TO 100
0175 XC=0.
0176 DO 50 I=1,3
0177 XCYL(I)=0.
0178 DO 50 J=1,3
0179 IF(N.EQ.2) XC=XCL(J,MCL)
0180 XCYL(I)=XCYL(I)+VCL(I,J,MCL)*(XREF(J)-XC)
0181 50 CONTINUE
0182 RETURN
0183 100 DO 200 I=1,3
0184 XREF(I)=0.
0185 DO 150 J=1,3
0186 150 XREF(I)=XREF(I)+VCL(J,I,MCL)*XCYL(J)
0187 IF(N.EQ.-2) XREF(I)=XREF(I)+XCL(I,MCL)
0188 200 CONTINUE
0189 RETURN
0190 END
```

SUBROUTINE DOCYLS

This procedure determines which mode of mapping has been selected by the user and calls the appropriate cylinder processing routines.

```
0001 C-----
0002 SUBROUTINE DOCYLS
0003 INCLUDE 'SHACOM.FOR'
0169 C!!!
0170 C!!! This subroutine processes all the cylinders one at a time.
0171 C!!! Do any special cylinders last.
0172 C!!!
0173 IF ( FILCNM .GT. 0 ) THEN
0174 DO 1 MC=1, MCX
0175 IF ( MC .NE. FILCNM ) CALL DOCYL( MC, FILCHR )
0176 1 CONTINUE
0177 CALL DOCYL( FILCNM, FILCHR )
0178 C!!!
0179 C!!! Fill with a different character for each cylinder.
0180 C!!!
0181 ELSEIF ( FILCNM .LT. 0 ) THEN
0182 DO 2 MC=1, MCX
0183 CALL DOCYL( MC, CHAR( MC+ICHR( '0' ) ) )
0184 2 CONTINUE
0185 C!!!
0186 C!!! Fill with the main background fill character.
0187 C!!!
0188 ELSE
0189 DO 3 MC=1, MCX
0190 CALL DOCYL( MC, FILCHR )
0191 3 CONTINUE
0192 ENDIF
0193 RETURN
0194 END
```

SUBROUTINE DOCYL

This routine projects the shadow boundary of a single cylinder onto the far-zone sphere and fills the area of the cylinder with the FILL argument.

```
0001 C-----
0002     SUBROUTINE DOCYL( IC, FILL )
0003     INCLUDE 'SHACOM.FOR'
0169 C!!!
0170 C!!! This subroutine processes a single cylinder.
0171 C!!!
0172     CHARACTER FILL
0173     INTEGER
0174     +     J, K, IC
0175 C! Loop control variables.
0176     +     INT,
0177 C! Truncate to integer.
0178     +     THETA1, PHI1
0179
0180     REAL
0181     +     THETA2, PHI2,
0182 C! Theta & phi in radians.
0183     +     I,
0184 C! The parametric loop parameter.
0185     +     MAGNE,
0186 C! Length of a pseudo-side
0187     +     XPY,
0188 C! Scratch variable.
0189     +     DOT, LSIDOT,
0190 C! Dot product variables
0191     +     BTAN2, SQRT, ABS,
0192 C! Miscellaneous functions
0193     +     XYZ( 3 ),
0194 C! temporary vector
0195     +     XPQ( 3 ),
0196 C! Source to edge in ref coords
0197     +     XPC( 3 ),
0198 C! Source to edge in pat coords
0199     +     RIM( 3 ),
0200 C! Point along cap in cyl coords
0201     +     RIM1( 3 ),
0202 C! Use for dotmin points
0203     +     RIM2( 3 ), ANCYL( 3 ),
0204 C! Antenna location in cylinder coords
0205     +     DOTMIN( 2, 10 )
0206 C! The two angles where dot is minimum
0207     LOGICAL
0208     +     FNDONE
0209 C! Found one of the zero dots
0210 C!
0211 C!!! Loop through endcaps, and incrementally on edges.
0212 C!!! Transform the antenna to cyl coords (include a translation).
0213 C!!!
0214     CALL CYLROT( ANTENN, ANCYL, +2, IC )
0215 C!!!
0216 C!!! Do the endcaps one at a time.
0217 C!!!
0218     DO 200 J=1, NEC(IC)
0219 C!!!
0220 C!!! Loop around the endcap and remember where the dot products are zero
0221 C!!! between the vector looking at the point and the radial vector on
```

```

0222 C!!! the endcap to the point. The cryptic parameters on the loop say:
0223 C!!! "Loop from zero to 2*PI in one-degree steps."
0224 C!!!
0225 DO 300 I=0.0, TPI+(TPI/360.0), (TPI/360.0)
0226 C!!!
0227 C!!! Calculate the dot product and remember the two smallest ones.
0228 C!!!
0229 RIM( 1 ) = COS(T) * AC(J,IC)
0230 RIM( 2 ) = SIN(T) * BC(J,IC)
0231 RIM( 3 ) = ZC(J,IC)
0232 DOT = RIM(1) * ( RIM(1) - ANCYL(1) )
0233 + RIM(2) * ( RIM(2) - ANCYL(2) )
0234 C!!!
0235 C!!! If (the last dot product) * (this dot product) < 0, then that is
0236 C!!! where our dot sign goes through zero.
0237 C!!!
0238 IF ( T .EQ. 0.0 ) THEN
0239 LSTDOT = DOT
0240 FNDONE = .FALSE.
0241 ENDIF
0242 IF ( SIGN(1.0,DOT) * SIGN(1.0,LSTDOT) .LT. 0.0 ) THEN
0243 IF( .NOT. FNDONE ) THEN
0244 DOTMIN( 1, J ) = T
0245 FNDONE = .TRUE.
0246 ELSE
0247 DOTMIN( 2, J ) = T
0248 ENDIF
0249 ENDIF
0250 LSTDOT = DOT
0251 C!!!
0252 C!!! Calculate theta & phi as we go around the rim.
0253 C!!! Transform the rim point into ref. coord.system.
0254 C!!! Find vector from source to rim.
0255 C!!!
0256 CALL CYLROT( XYZ, RIM, -2, IC )
0257 C!!!
0258 C!!! Convert from the reference coordinate system to the pattern
0259 C!!! coordinate system.
0260 C!!!
0261 XPQ(1) = XYZ(1) - ANTENN(1)
0262 XPQ(2) = XYZ(2) - ANTENN(2)
0263 XPQ(3) = XYZ(3) - ANTENN(3)
0264
0265 XPC(1) = XPQ(1)*VPC(1,1) + XPQ(2)*VPC(1,2) + XPQ(3)*VPC(1,3)
0266 XPC(2) = XPQ(1)*VPC(2,1) + XPQ(2)*VPC(2,2) + XPQ(3)*VPC(2,3)
0267 XPC(3) = XPQ(1)*VPC(3,1) + XPQ(2)*VPC(3,2) + XPQ(3)*VPC(3,3)
0268
0269 XPY = SQRT( XPC(1)*XPC(1) + XPC(2)*XPC(2) )
0270 C!!!
0271 C!!! Calculate angles representing border of rim and do branch test
0272 C!!! on the phi angle.
0273 C!!!
0274 THETAR = BTAN2( XPY, XPC(3) )
0275 PHIR = BTAN2( XPC(2), XPC(1) )
0276 IF ( PHIR .LT. PH1-0.5*RESPH ) PHIR = TPI + PHIR
0277 C!!!
0278 C!!! Define pixel location.
0279 C!!!
0280 THETA1 = INT( (THETAR - THET1) / RESTH + 0.5 ) + 1
0281 PHII = INT( (PHIR - PH1) / RESPH + 0.5 ) + 1
0282 C!!!
0283 C!!! Put the character into the output buffer at the proper position.
0284 C!!! Test if indices fall within specified window.
0285 C!!!

```

```

0286         IF ( (THETA1 .GE. 1) .AND. (THETA1 .LE. COLS) ) THEN
0287         IF ( (PHI1 .GE. 1) .AND. (PHI1 .LE. ROWS) ) THEN
0288             OUTBUF( THETA1, PHI1 ) = CHAR(7)
0289         ENDIF
0290     ENDIF
0291 C!!!
0292 C!!! Reduplicate a wrapped-around character.
0293 C!!!
0294         IF ( (PHI1 .EQ. 1) .AND. ABS(PH2-PH1-TPI) .LE. RESPH) THEN
0295             OUTBUF( THETA1, ROWS ) = CHAR(7)
0296         ENDIF
0297     300 CONTINUE
0298     200 CONTINUE
0299 C!!!
0300 C!!! Before rasterizing, connect the "dotmins".
0301 C!!! A sneaky trick is pulled here. Instead of transforming every
0302 C!!! increment of the dotmin points, only the two end points are
0303 C!!! transformed, then theta & phi are calculated for each increment.
0304 C!!! This is valid because the line which connects the two points on
0305 C!!! the rims of the cylinders are straight lines in both RCS and cyl
0306 C!!! coord systems. Note that this gizmo assumes that you are never
0307 C!!! inside of a cylinder, or your dotmins(K,) probably get crossed
0308 C!!! resulting in an inside-out or bowtie-shaped cylinder.
0309 C!!!
0310     DO 400 K=1, 2
0311         DO 500 J=1, NEC(IC)-1
0312             RIM( 1 ) = COS( DOTMIN(K,J) ) * AC(J,IC)
0313             RIM( 2 ) = SIN( DOTMIN(K,J) ) * BC(J,IC)
0314             RIM( 3 ) = ZC(J,IC)
0315             CALL CYLROT( RIM1, RIM, -2, IC )
0316             RIM( 1 ) = COS( DOTMIN(K,J+1) ) * AC(J+1,IC)
0317             RIM( 2 ) = SIN( DOTMIN(K,J+1) ) * BC(J+1,IC)
0318             RIM( 3 ) = ZC(J+1,IC)
0319             CALL CYLROT( RIM2, RIM, -2, IC )
0320 C!!!
0321 C!!! This MAGME is analogous to the one in DOPLA except it
0322 C!!! works with psuedo-sides, so the name is somewhat misleading.
0323 C!!!
0324             MAGME = SQRT(
0325 +                 ( RIM2(1) - RIM1(1) )**2 +
0326 +                 ( RIM2(2) - RIM1(2) )**2 +
0327 +                 ( RIM2(3) - RIM1(3) )**2 )
0328             T = 0.0
0329     50     IF ( T .GT. 1.0 ) GOTO 600
0330 C!!!
0331 C!!!
0332 C!!! These functions compute the theta/phi associated with a given point
0333 C!!! along a cylinder psuedo-edge as a function of T (See DOPLA.)
0334 C!!! The variables XYZ and RIM are re-used for multiple purposes here.
0335 C!!!
0336 C!!! Find vector from source to rim.
0337 C!!! Convert from the reference coordinate system to the pattern
0338 C!!! coordinate system
0339 C!!!
0340             XPQ(1) = ( RIM2(1)-RIM1(1) )*T + RIM1(1)-ANTEHN(1)
0341             XPQ(2) = ( RIM2(2)-RIM1(2) )*T + RIM1(2)-ANTEHN(2)
0342             XPQ(3) = ( RIM2(3)-RIM1(3) )*T + RIM1(3)-ANTEHN(3)
0343
0344             XPC(1) = XPQ(1)*VPC(1,1) + XPQ(2)*VPC(1,2) + XPQ(3)*VPC(1,3)
0345             XPC(2) = XPQ(1)*VPC(2,1) + XPQ(2)*VPC(2,2) + XPQ(3)*VPC(2,3)
0346             XPC(3) = XPQ(1)*VPC(3,1) + XPQ(2)*VPC(3,2) + XPQ(3)*VPC(3,3)
0347             XPY = SQRT(XPC(1)*XPC(1) + XPC(2)*XPC(2))
0348 C!!!
0349 C!!! Define the angles representing the projection of the curved sides

```



```

0350 C!!! and do a branch cut test on phi.
0351 C!!!
0352 THETAR = BTAN2( XPY, IPC(3) )
0353 PHIR = BTAN2( IPC(2), IPC(1) )
0354 IF ( PHIR .LT. PHI-0.5*RESPH ) PHIR = TPI + PHIR
0355 C!!!
0356 C!!! Define pixel location and put character in appropriate spot.
0357 C!!!
0358 THETAI = INT( (THETAR - THET1) / RESTH + 0.5 ) + 1
0359 PHII = INT( (PHIR - PHI) / RESPH + 0.5 ) + 1
0360 C!!!
0361 C!!! Check if angles fall within window.
0362 C!!!
0363 IF ( (THETAI .GE. 1) .AND. (THETAI .LE. COLS) ) THEN
0364 IF ( (PHII .GE. 1) .AND. (PHII .LE. ROWS) ) THEN
0365 OUTBUF( THETAI, PHII ) = CHAR(7)
0366 ENDF
0367 ENDF
0368 C!!!
0369 C!!! Reduplicate a wrapped-around character.
0370 C!!!
0371 IF( (PHII .EQ. 1) .AND. ABS(PH2-PH1-TPI) .LE. RESPH ) THEN
0372 OUTBUF( THETAI, ROWS ) = CHAR(7)
0373 ENDF
0374 T = T + MIN( 0.99, (XPY*ALPH/MAGNE + 1.E-7) )
0375 GOTO 50
0376 600 CONTINUE
0377 500 CONTINUE
0378 400 CONTINUE
0379 C!!!
0380 C!!! Now do an area fill on the object just outlined.
0381 C!!! Tell SCAN that this is a CYLINDER by using a "2".
0382 C!!!
0383 DO 700 PHII = 1, ROWS
0384 CALL SCAN( IC, OUTBUF(1,PHII), PHII, FILL, 2 )
0385 700 CONTINUE
0386
0387 RETURN
0388 END

```

SUBROUTINE DOPLAS

This routine determines which mapping options the user has selected and calls the appropriate plate processing routines.

```
0001 C-----
0002 SUBROUTINE DOPLAS
0003 INCLUDE 'SHACOM.FOR'
0169 C!!!
0170 C!!! This subroutine processes each plate one at a time. The
0171 C!!! highlighting logic is contained here.
0172 C!!!
0173 C!!! Do the plates one at a time, then do the plate that was supposed to
0174 C!!! be highlighted last.
0175 C!!!
0176 IF ( FILPNM .GT. 0 ) THEN
0177 DO 1 MP = 1, MPX
0178 IF ( MP .NE. FILPNM ) CALL DOPLA( MP, FILCHR )
0179 1 CONTINUE
0180 CALL DOPLA( FILPNM, FILCHP )
0181 C!!!
0182 C!!! Fill with a different character for each plate.
0183 C!!!
0184 ELSEIF ( FILPNM .LT. 0 ) THEN
0185 DO 2 MP = 1, MPX
0186 CALL DOPLA( MP, CHAR( MP+ICHAR( 'a' ) ) )
0187 2 CONTINUE
0188 C!!!
0189 C!!! Fill everything with the main background character.
0190 C!!!
0191 ELSE
0192 DO 3 MP = 1, MPX
0193 CALL DOPLA( MP, FILCHR )
0194 3 CONTINUE
0195 ENDIF
0196
0197 RETURN
0198 END
```

SUBROUTINE DOPLA

This routine computes the shadow map for a single cylinder by projecting its boundaries onto the far-zone sphere and then filling in its area in the map array.

```

0001  C-----
0002      SUBROUTINE DOPLA( IP, FILL )
0003      INCLUDE 'SHACOM.FOR'
0169      CHARACTER FILL
0170      INTEGER
0171      + IP, INT,
0172      C! Truncate to the nearest integer.
0173
0174      + THETA1, PHI1
0175      C! Local indicies into char array.
0176
0177      REAL
0178      + T,
0179      C! Parametric increment parameter.
0180
0181      + THETA1R, PHI1R,
0182      C! Theta & phi in radians.
0183
0184      + MAGME,
0185      C! Length of side ME.
0186
0187      + IPY,
0188      C! temporary variable
0189
0190      + XPQ(3),
0191      C! Source to edge in ref coords
0192
0193      + XPC(3),
0194      C! Source to edge in pat coords
0195
0196      + BTAN2, SQRT, ABS
0197      C! Miscellaneous functions.
0198      C!!!
0199      C!!! Loop through incrementally along edges.
0200      C!!!
0201      DO 200 ME=1, MEP( IP )
0202          NEXTME = MOD( ME, MEP(IP) ) + 1
0203          MAGME = VMAG( ME, IP )
0204          T = 0.0
0205      50      IF ( T .GT. 1.0 ) GOTO 100
0206      C!!!
0207      C!!! These functions compute the theta/phi associated with a given
0208      C!!! point along an edge between two corners ME and NEXTME as a
0209      C!!! function of T. T varies from 0 to 1 and is adjusted to keep
0210      C!!! within a safe and efficient excursion at all times.
0211      C!!!
0212      C!!! Convert from the reference coordinate system to the pattern
0213      C!!! coordinate system
0214      C!!!
0215      XPQ(1)=(XX(1,NEXTME,IP)-XX(1,ME,IP))+T*XX(1,ME,IP)-ANTENN(1)
0216      XPQ(2)=(XX(2,NEXTME,IP)-XX(2,ME,IP))+T*XX(2,ME,IP)-ANTENN(2)
0217      XPQ(3)=(XX(3,NEXTME,IP)-XX(3,ME,IP))+T*XX(3,ME,IP)-ANTENN(3)
0218
0219      IPC(1)=XPQ(1)*VPC(1,1) + XPQ(2)*VPC(1,2) + XPQ(3)*VPC(1,3)
0220      IPC(2)=XPQ(1)*VPC(2,1) + XPQ(2)*VPC(2,2) + XPQ(3)*VPC(2,3)
0221      IPC(3)=XPQ(1)*VPC(3,1) + XPQ(2)*VPC(3,2) + XPQ(3)*VPC(3,3)

```

```

0222
0223       XPY = SQRT( XPC(1)*XPC(1) + XPC(2)*XPC(2) )
0224     C!!!
0225     C!!! Define the angles representing the projection of the curved sides
0226     C!!! and do a branch cut test on phi.
0227     C!!!
0228       THETAR = BIAN2( XPY,   XPC(3) )
0229       PHIR   = BIAN2( XPC(2), XPC(1) )
0230
0231       IF ( PHIR .LT. PH1-0.5*RESPH ) PHIR = TPI + PHIR
0232     C!!!
0233     C!!! Define pixel location and put the a character in the appropriate
0234     C!!! spot.
0235     C!!!
0236       THETAI = INT( (THETAR - THET1) / RESTH + 0.5 ) + 1
0237       PHII   = INT( (PHIR - PH1)   / RESPH + 0.5 ) + 1
0238     C!!!
0239     C!!! Check if angles fall within window.
0240     C!!!
0241       IF ( (THETAI .GE. 1) .AND. (THETAI .LE. COLS) ) THEN
0242         IF ( (PHII .GE. 1) .AND. (PHII .LE. ROWS) ) THEN
0243           OUTBUF( THETAI, PHII ) = CHAR(7)
0244         ENDIF
0245       ENDIF
0246     C!!!
0247     C!!! Reduplicate a wrapped-around character.
0248     C!!!
0249       IF( (PHII .EQ. 1) .AND. ABS(PH2-PH1-TPI) .LE. RESPH) THEN
0250         OUTBUF( THETAI, ROWS ) = CHAR(7)
0251       ENDIF
0252     C!!!
0253     C!!! Put an upper bound on the increment for the case when the line
0254     C!!! segment is very short or the distance to the segment is great.
0255     C!!! In the degenerate case (on the Z-axis) prevent a potential infinite
0256     C!!! loop by putting a lower bound on delta-t (ie by always adding at
0257     C!!! least a very small number to I.)
0258     C!!!
0259       I = I + MIN( 0.99, (XPY*ALPH/MAGME + 1.E-7) )
0260     GOTO 50
0261   100   CONTINUE
0262   200   CONTINUE
0263     C!!!
0264     C!!! Now do an area fill on the object just outlined.
0265     C!!! Tell SCAN that this is a plate by using a "1".
0266     C!!!
0267       DO 300 PHII = 1, ROWS
0268         CALL SCAN( IP, OUTBUF(1,PHII), PHII, FILL, 1 )
0269   300   CONTINUE
0270
0271       RETURN
0272     END

```

SUBROUTINE GEOM

This routine computes necessary geometrical information needed by other routines. It is called before the main command loop.

```

0001 C-----
0002     SUBROUTINE GEOM
0003 C111
0004 C111     THIS ROUTINE COMPUTES ALL THE GEOMETRY ASSOCIATED
0005 C111 WITH FIXED PLATE STRUCTURE, SUCH AS EDGE UNIT VECTORS,
0006 C111 PLATE NORMALS, SHADOWED PLATES, ETC.
0007 C111
0008     INCLUDE 'SHACOM.FOR'
0174     DIMENSION IHIT(NEX), XII(3), XIN(3), VI(3), XC(3), XSI(3), XSII(3)
0175     DIMENSION XOB(3), XDC(3), VTCP(2), BTCP(4), VICN(2), BTGN(4), DS(3)
0176     DIMENSION VVO(3), VVN(3), VVB(3), VVH(3), XBI(3), PVTRN1(3)
0177     LOGICAL LSURF, LNPL, LTRN1
0178     LOGICAL LSHD, LSTD, LSTS, LCTD, LHCT, LHIT
0179     LOGICAL LGRND, LIHD, LDEBUG, LTEST, LWARN
0180     COMMON/TEST/LDEBUG, LTEST, LWARN
0181     COMMON/LIMIT/SML, SMLR, SMLT, BIG
0182     COMMON/WAVE/WK, WL
0183     COMMON/LSHDP/LSTS, LSTD(NEX)
0184     COMMON/GROUND/LGRND, MPXR
0185     IF(LDEBUG) WRITE(6,667)
0186 667   FORMAT(/, ' DEBUGGING GEOM SUBROUTINE')
0187 C111 DETERMINATION OF V, VN, AND VP UNIT VECTORS FOR EDGE-FIXED
0188 C111 COORDINATE SYSTEM
0189 C111 STEP THRU PLATES
0190     DO 100 MP=1, MPXR
0191     MEX=MEP(MP)
0192 C111 STEP THRU EDGES
0193     DO 15 ME=1, MEX
0194     MME=ME+1
0195     IF(MME.GT.MEX) MME=1
0196     VM=0.
0197 C111 CALCULATE EDGE UNIT VECTOR V AND EDGE LENGTH VMAG
0198     DO 10 N=1, 3
0199     V(N,ME,MP)=XI(N,MME,MP)-XI(N,ME,MP)
0200 10    VM=VM+V(N,ME,MP)*V(N,ME,MP)
0201     VMAG(ME,MP)=SQRT(VM)
0202     DO 11 N=1, 3
0203 11    V(N,ME,MP)=V(N,ME,MP)/VMAG(ME,MP)
0204 15    CONTINUE
0205     IF(.NOT.LDEBUG) GO TO 991
0206     DO 992 ME=1, MEX
0207     WRITE(6,*) (V(N,ME,MP), N=1, 3)
0208 992   CONTINUE
0209 991   CONTINUE
0210 C111 CALCULATE PLATE UNIT NORMAL VN
0211     VN(1,MP)=0.
0212     VN(2,MP)=0.
0213     VN(3,MP)=0.
0214     DO 22 ME=1, MEX
0215     MV=ME+1
0216     IF(MV.GT.MEX) MV=1
0217     VN(1,MP)=VN(1,MP)+V(2,ME,MP)*V(3,MV,MP)-V(2,MV,MP)*V(3,ME,MP)
0218     VN(2,MP)=VN(2,MP)+V(3,ME,MP)*V(1,MV,MP)-V(3,MV,MP)*V(1,ME,MP)
0219     VN(3,MP)=VN(3,MP)+V(1,ME,MP)*V(2,MV,MP)-V(1,MV,MP)*V(2,ME,MP)
0220 22    CONTINUE
0221     VNM=0.

```

```

0222      DD 20 N=1,3
0223 20   VNM=VNM+VN(N,MP)*VN(N,MP)
0224      VNM=SQRT(VNM)
0225      DD 21 N=1,3
0226 21   VN(N,MP)=VN(N,MP)/VNM
0227      IF(LDEBUG) WRITE(6,*) (VN(N,MP),N=1,3)
0228 C111  INSURE THAT ALL PLATES ARE FLAT. OTHERWISE ABORT!
0229 C111  TAKE DOT PRODUCT OF PLATE NORMAL AND EACH EDGE UNIT VECTOR
0230      DD 120 ME=1,MEX
0231      DOT=VN(1,MP)*V(1,ME,MP)+VN(2,MP)*V(2,ME,MP)+VN(3,MP)*V(3,ME,MP)
0232      ADOT=ABS(DOT)
0233      IF(ADOT.LT.0.01) GO TO 120
0234      MEE=ME+1
0235      IF(MEE.GT.MEX) MEE=1
0236      WRITE(6,121) MP,MEE,ADOT
0237 121  FORMAT(' PLATE # ',I2,' IS NOT FLAT! CORNER # ',I2,' HAS ',
0238 2'PROBLEM. '/' WARP= ',F7.3,' PROGRAM ABORTS IF THE WARP'
0239 3' IS GREATER THAN 0.03 *****')
0240      IF(ADOT.GT.0.03) STOP
0241 120  CONTINUE
0242 C111  CALCULATE UNIT BINORMAL VP WHICH IS IN PLATE PLANE
0243 C111  AND PERPENDICULAR TO PLATE EDGE
0244 C111  TAKE CROSS PRODUCT OF PLATE NORMAL AND EDGE VECTOR
0245      DD 30 ME=1,MEX
0246      VP(1,ME,MP)=VN(2,MP)*V(3,ME,MP)-VN(3,MP)*V(2,ME,MP)
0247      VP(2,ME,MP)=VN(3,MP)*V(1,ME,MP)-VN(1,MP)*V(3,ME,MP)
0248 30   VP(3,ME,MP)=VN(1,MP)*V(2,ME,MP)-VN(2,MP)*V(1,ME,MP)
0249      IF(.NOT.LDEBUG) GO TO 993
0250      DD 994 ME=1,MEX
0251 994  WRITE(6,*) (VP(N,ME,MP),N=1,3)
0252 993  CONTINUE
0253 100  CONTINUE
0254      RETURN
0255      END

```

SUBROUTINE INITGF

This routine is used to initialize graphics each time an output is desired. Here, it zeroes out the previous array information and recalculates parameters based on the user-specified desired resolution.

```
0001 C-----
0002     SUBROUTINE INITGF
0003     INCLUDE 'SHACOM.FOR'
0169 C!!!
0170 C!!! This subroutine initializes some graphics stuff.
0171 C!!! Its function is to initialize things from one plot to the next,
0172 C!!! but within the context of a single session.
0173 C!!!
0174     INTEGER
0175     +     I, J, INT
0176 C!!!
0177 C!!! Clear the character buffer.
0178 C!!!
0179     DO 10 J=1, MAXROW
0180     DO 10 I=1, MAXCOL
0181     10     OUTBUF( I, J ) = ' '
0182 C!!!
0183 C!!! The number of rows & columns needed for internal representation is
0184 C!!! calculated from the user-selected (or defaulted) angular ranges of
0185 C!!! interest combined with the desired resolution in rads/pixel
0186 C!!!
0187     ROWS = INT( (PH2 - PH1) / RESPH + 0.5 ) + 1
0188     COLS = INT( (THET2 - THET1) / RESTH + 0.5 ) + 1
0189 C!!!
0190 C!!! Calculate some parameters needed by the dynamic I increment
0191 C!!! algorithms. The maximum allowable angular excursion is the
0192 C!!! smaller of the number of radians in a single pixel of either theta
0193 C!!! or phi.
0194 C!!!
0195     ALPH = MIN( RESTH, RESPH )
0196 C!!!
0197     RETURN
0198     END
```

SUBROUTINE PLAINT

This routines determines if a given ray strikes a plate.

```

0001  C
0002  C-----
0003      SUBROUTINE PLAINT(XIS,D,DHIT,MH,LHIT)
0004  C!!!
0005  C!!! DOES RAY HIT PLATE. IF MH=0 ALL PLATES ARE CHECKED.
0006  C!!! IF MH=-MP THEN ONLY MP CHECKED AND SOURCE POSITION
0007  C!!! MOVED TO HIT POSITION IF RAY HITS MP.
0008  C!!! IF MH=MP, THEN ALL PLATES OTHER THAN MP ARE CHECKED.
0009  C!!!
0010      INCLUDE 'SHACOM.FOR'
0176      DIMENSION XIS(3),D(3),XI(3),PVTRN(3)
0177      LOGICAL LHIT,LPLA,LCYL,LSTS,LSTD,LTRN
0178      LOGICAL LGRND,LDEBUG,LTEST,LWARN
0179      COMMON/TEST/LDEBUG,LTEST,LWARN
0180      COMMON/LIMIT/SML,SMLR,SMLT,BIG
0181      COMMON/LPLCY/LPLA,LCYL
0182      COMMON/HITPLT/MPH
0183      COMMON/GROUND/LGRND,MPXR
0184      LHIT=.FALSE.
0185      DHIT=0.
0186      IF(.NOT.LPLA) RETURN
0187  C!!! STEP THRU PLATES
0188      DO 60 MPP=1,MPXR
0189      MP=MPP
0190      IF(MP.EQ.MH) GO TO 50
0191      IF(MH.LT.0) MP=IABS(MH)
0192  C!!! IF TOTAL SHADOWING ALGORITHM IS BEING USED, HAS PLATE MP
0193  C!!! SHADOWED EVERY RAY TESTED?
0194  CXXXXX      IF(LSTS.AND..NOT.LSTD(MP)) GO TO 60
0195      MEX=MEP(MP)
0196      AN=0.
0197      DO 5 N=1,3
0198      5      AN=AN+(XIS(N)-XX(N,1,MP))*VN(N,MP)
0199      DN=D(1)*VN(1,MP)+D(2)*VN(2,MP)+D(3)*VN(3,MP)
0200  C!!! DOES RAY PASS THRU PLATE PLANE?
0201      IF(AN*DN.GE.0.) GO TO 50
0202      DO 10 N=1,3
0203  C!!! CALCULATE POINT WHERE RAY INTERSECTS PLATE PLANE
0204      10      XI(N)=XIS(N)-AN*D(N)/DN
0205      IF(MP.EQ.MPXR.AND.LGRND) GO TO 11
0206      DBT=0.
0207  C!!! IS HIT POINT ON PLATE?
0208      DO 30 ME=1,MEX
0209      MME=ME+1
0210      IF(MME.GT.MEX) MME=1
0211      RD=0.
0212      DO 20 N=1,3
0213      20      RD=RD+(XI(N,ME,MP)-XI(N))*(XX(N,MME,MP)-XI(N))
0214      CP=VN(1,MP)*((XX(2,ME,MP)-XI(2))*(XX(3,MME,MP)-XI(3))
0215      2-(XX(3,ME,MP)-XI(3))*(XX(2,MME,MP)-XI(2)))
0216      CP=CP+VN(2,MP)*((XX(3,ME,MP)-XI(3))*(XX(1,MME,MP)-XI(1))
0217      2-(XX(1,ME,MP)-XI(1))*(XX(3,MME,MP)-XI(3)))
0218      CP=CP+VN(3,MP)*((XX(1,ME,MP)-XI(1))*(XX(2,MME,MP)-XI(2))
0219      2-(XX(2,ME,MP)-XI(2))*(XX(1,MME,MP)-XI(1)))
0220      DBI=BTAN2(CP,RD)
0221      DBT=DBT+DBI
0222      30      CONTINUE

```



```

0223      IF(ABS(DBT).LT.PI) GO TO 50
0224      C111  CALCULATE DISTANCE TO HIT (DHIT=SHORTEST DHT)
0225      11    DHT=0.
0226      DO 40 N=1,3
0227      40    DHT=DHT+(IT(N)-IIS(N))*(IT(N)-IIS(N))
0228      DHT=SQRT(DHT)+SMLR
0229      IF(LHIT.AND.(DHT.GT.DHIT)) GO TO 60
0230      LHIT=.TRUE.
0231      DHIT=DHT
0232      MPH=MP
0233      IF(MH.GE.0) GO TO 60
0234      DO 45 N=1,3
0235      C111  MOVE HIT POSITION AN INCREMENT TOWARDS SIDE OF PLATE
0236      C111  WHICH SOURCE LIES ON
0237      45    IIS(N)=XT(N)-SIGN(SMLR,AN)+VN(N,MP)
0238      GO TO 61
0239      50    CONTINUE
0240      IF(MH.LT.0) GO TO 61
0241      C111  IF TOTAL SHADOWING ROUTINE IS BEING USED, INDICATE
0242      C111  THAT PLATE MP DOES NOT SHADOW SOURCE
0243      CXXXX  IF(LSTS) LSTD(MP)=.FALSE.
0244      60    CONTINUE
0245      61    IF(.NOT.LTEST) GO TO 62
0246      WRITE(6,63)
0247      63    FORMAT(/,' TESTING PLAINT SUBROUTINE')
0248      WRITE(6,*) IIS
0249      WRITE(6,*) D
0250      WRITE(6,*) DHIT,MH,LHIT
0251      62    RETURN
0252      END

```

SUBROUTINE SCAN

This subroutine rasterizes a line in the character buffer according to its shading requirements. It calls routines to determine if a given point is shadowed or not and uses this information to shadow the given geometry. The fill character is used to fill the line in.

```
0001 C-----
0002 SUBROUTINE SCAN ( OBJ, LINE, PHII, FILL, TYPE )
0003 C!!!
0004 C!!! This subroutine "rasterizes" a line in the character buffer
0005 C!!! according to its shading requirements.
0006 C!!! The fill character is used to fill the
0007 C!!! line in. The line is declared larger character string in this
0008 C!!! subroutine than in the calling routine. This can be "hardwired"
0009 C!!! if it causes problems on other machines.
0010 C!!!
0011 INCLUDE 'SHACOM.FOR'
0177 CHARACTER*500 LINE
0178 CHARACTER*1 FILL
0179 INTEGER PTR, OBJ
0180 INTEGER LSTPTR
0181 INTEGER SCANC
0182 INTEGER SPANC
0183 INTEGER PHII
0184 INTEGER TYPE
0185 REAL DHII
0186 REAL D(3)
0187 REAL DP(3)
0188 REAL XIS(3)
0189 REAL THETA,PHI
0190 LOGICAL EOL
0191 LOGICAL LHII, LHIT1
0192 COMMON /SCNCOM/ PTR, EOL
0193 C!!!
0194 C!!! Initialize local variables.
0195 C!!!
0196 PTR = 1
0197 LSTPTR = 1
0198 EOL = PTR .GT. COLS
0199 C!!!
0200 C!!! Until the end of the line is scanned do ...
0201 C!!!
0202 100 IF (.NOT. EOL) THEN
0203 C!!!
0204 C!!! Locate the first occurrence of CHAR 7, 8, or EOL.
0205 C!!!
0206 PTR = SCANC( LINE )
0207 C!!!
0208 C!!! If plaint says it's a miss, update the last-pointer, span, scan,
0209 C!!! fill. Otherwise, fill in the characters between the pointers.
0210 C!!! Define the "source point" AS the location of the antenna,
0211 C!!! and see if our plate shadows the direction of the midpoint of the
0212 C!!! scan.
0213 C!!!
0214 THEIA = (0.5*FLOAT(PTR-LSTPTR)-1.0)*RESTH+THET1
0215 PHI = (PHII-1)*RESPH+PHI
0216 DP(1) = SIN(THETA)*COS(PHI)
0217 DP(2) = SIN(THETA)*SIN(PHI)
0218 DP(3) = COS(THETA)
0219 D(1) = DP(1)*VPC(1,1) + DP(2)*VPC(2,1) + DP(3)*VPC(3,1)
```

```

0220      D(2) = DP(1)*VPC(1,2) + DP(2)*VPC(2,2) + DP(3)*VPC(3,2)
0221      D(3) = DP(1)*VPC(1,3) + DP(2)*VPC(2,3) + DP(3)*VPC(3,3)
0222      C!!!
0223      C!!! This must be done due to the behavior of plaint modifying IIS.
0224      C!!!
0225      IIS(1) = ANTENN(1)
0226      IIS(2) = ANTENN(2)
0227      IIS(3) = ANTENN(3)
0228      C!!!
0229      C!!! Now do a case depending on what type of object we test for
0230      C!!! shadowing.
0231      C!!!
0232      C!!! 1 = plate
0233      C!!! 2 = elliptic cylinder
0234      C!!!
0235      GOTO (1,2) TYPE
0236      C!!!
0237      C!!! The object is a plate.
0238      C!!!
0239      1      CALL PLAINI( IIS, D, DHIT, -OBJ, LHIT )
0240      GOTO 999
0241      C!!!
0242      C!!! The object is a cylinder. Test endcaps and cylinder bodies.
0243      C!!!
0244      2      CALL CAPINT( IIS, D, DHIT, O, LHIT1, -OBJ )
0245      IF ( .NOT. LHIT1) CALL CYLINT(IIS,D,DHIT, LHIT, .FALSE., -OBJ)
0246      LHIT = LHIT .OR. LHIT1
0247      GOTO 999
0248      C!!!
0249      C!!! Take the appropriate action in the buffer.
0250      C!!!
0251      999    IF ( .NOT. LHIT ) THEN
0252      LSTPTR = PTR
0253      PTR = SPANC( LINE )
0254      DO 300 LSTPTR = LSTPTR, PTR-1, 1
0255      LINE( LSTPTR:LSTPTR ) = FILL
0256      300    CONTINUE
0257      LSTPTR = PTR
0258      ELSE
0259      PTR = SPANC( LINE )
0260      DO 400 LSTPTR = LSTPTR, PTR-1, 1
0261      LINE( LSTPTR:LSTPTR ) = FILL
0262      400    CONTINUE
0263      LSTPTR = PTR
0264      ENDDIF
0265      C!!!
0266      C!!! End UNTIL
0267      C!!!
0268      GOTO 100
0269      END IF
0270      C!!!
0271      RETURN
0272      END

```

FUNCTION SCANC/SPANC

These functions are used to scan through the character buffer (map array) and locate/skip certain characters. They return the positions of these characters as their result.

```
0001 C-----
0002 C!!!
0003 C!!! The following functions span/scan characters. That is, they
0004 C!!! return the position of next character in LINE which does or does
0005 C!!! not match the specified character. They also
0006 C!!! terminate the scan/span at the end of the line.
0007 C!!!
0008 INTEGER          FUNCTION SCANC( LINE )
0009 INCLUDE          'SHACOM.FOR'
0015 CHARACTER*(*)  LINE
0016 INTEGER          PTR
0017 LOGICAL          EOL
0018 COMMON /SCNCMN/ PTR, EOL
0019 C!!!
0020 C!!! Until a character matching CHARAC is found, advance the pointer.
0021 C!!!
0022 SCANC = PTR
0023 200 IF (.NOT. (EOL .OR.
0024 + ( LINE(SCANC:SCANC) .EQ. CHAR(7) ) )) THEN
0025     SCANC = SCANC + 1
0026     EOL = SCANC .GT. COLS
0027     GO TO 200
0028 ENDIF
0029 C!!!
0030 C!!! End UNITL
0031 C!!!
0032 RETURN
0033 END
0001 C!!!
0002 C-----
0003 C!!!
0004 INTEGER          FUNCTION SPANC( LINE )
0005 INCLUDE          'SHACOM.FOR'
0011 CHARACTER*(*)  LINE
0012 INTEGER          PTR
0013 LOGICAL          EOL
0014 COMMON /SCNCMN/ PTR, EOL
0015 C!!!
0016 C!!! Until a character NOT matching ASCII 7 is found, advance the
0017 C!!! pointer.
0018 C!!!
0019 SPANC = PTR
0020 200 IF (.NOT. (EOL .OR.
0021 + ( LINE(SPANC:SPANC) .NE. CHAR(7) ) )) THEN
0022     SPANC = SPANC + 1
0023     EOL = SPANC .GT. COLS
0024     GO TO 200
0025 ENDIF
0026 C!!!
0027 C!!! End UNITL
0028 C!!!
0029 RETURN
0030 END
```

SUBROUTINE WRTOUT

This subroutine produces formatted and binary output of the shadow map.

```
0001 C-----
0002 SUBROUTINE WRTOUT
0003 INCLUDE 'SHACOM.FOR'
0169 INTEGER I, J, COLI, COLF
0170 C!!!
0171 C!!! This subroutine writes the formatted output buffer to the output
0172 C!!! file. Start the output on a new page, and calculate a header
0173 C!!! based on the specified pixel resolution.
0174 C!!!
0175 C!!! Unit 7 is the main (ASCII) output file.
0176 C!!!
0177 C!!! Initialize the width of the map to be printed.
0178 C!!!
0179 COLI = 1
0180 COLF = 91
0181 IF(COLF .GT. COLS) COLF = COLS
0182 C!!!
0183 C!!! Print map.
0184 C!!!
0185 20 WRITE( 7, 100 ) ( ANTENN(I), I=1, 3, 1 ), INPFIL
0186 WRITE( 7, 200 ) ( (RESTH*(I-1) + THET1)*DPR , I= COLI, COLF, 10)
0187 WRITE( 7, 250 ) ( '+', I= COLI, COLF, 10)
0188 DO 50 J = 1, ROWS
0189 50 WRITE( 7, 300 ) ( RESPH*(J-1) + PH1 )*DPR,
0190 + ( OUTBUF(I,J), I= COLI, COLF )
0191 C!!!
0192 C!!! If the map does not fit on the line printer width,
0193 C!!! then split it onto another set of pages.
0194 C!!!
0195 IF(COLF .LT. COLS) THEN
0196 COLI = COLF
0197 COLF = COLF + 90
0198 IF(COLF .GT. COLS) COLF = COLS
0199 GO TO 20
0200 ENDIF
0201 C!!!
0202 C!!! Have internal parameters available in degrees.
0203 C!!!
0204 THET1D = THET1 *DPR
0205 THET2D = THET2 *DPR
0206 RESTHD = RESTH *DPR
0207 PH1D = PH1 *DPR
0208 PH2D = PH2 *DPR
0209 RESPHD = RESPH *DPR
0210 C!!!
0211 C!!! Unit 10 is a generic sort of binary output which can be plotted
0212 C!!! anywhere. Place a little header info at the front of the file.
0213 C!!!
0214 WRITE( 10 ) COLS, THET1D, THET2D, RESTHD
0215 WRITE( 10 ) ROWS, PH1D, PH2D, RESPHD
0216 C!!!
0217 C!!! Dump only that part of the buffer which pertains to this plot.
0218 C!!!
0219 DO 10 J = 1, ROWS
0220 DO 10 I = 1, COLS
0221 10 WRITE (10) OUTBUF( I, J )
0222 C!!!
```

```
0223 C!!! Output stuff is complete.
0224 C!!!
0225     RETURN
0226 C!!!
0227 C!!! Format statements.
0228 C!!!
0229 100  FORMAT( '1', 8X, 'ANTENNA (RCS) = ', '( ', 2(F8.4, ', '),
0230     +      F8.4, ' ) IN METERS', 5X, 'INPUT SET: ', A42, / )
0231 200  FORMAT( T60, 'THETA (DEGREES)',/, 9X, 11( 4X, F8.2) )
0232 250  FORMAT( 9X, 'PHI', 4X, A, 10( 9X, A) )
0233 300  FORMAT( 6X, F7.2, 3X, 101A )
0234     END
```

Include file

This is a listing of the common blocks and parameter statements contained in the single include file for SHADOW. Note that the include file appears in the compiler listing for the interactive service routines.

```
C111
C111 COMMON declerations...
C111
      COMMON /PIS/
      +      PI,
      +      TPI,
      +      DPR,
      +      RPD

C111
C+++ MAXIMUM DIMENSION FOR PLATES
      INTEGER      NPX
      PARAMETER (NPX=75)
C+++ MAXIMUM DIMENSION FOR PLATE EDGES
      INTEGER      NEX
      PARAMETER (NEX=12)
C+++ MAXIMUM DIMENSION FOR CYLINDERS
      INTEGER      NCX
      PARAMETER (NCX=5)
C+++ MAXIMUM DIMENSION FOR CYLINDER RIMS
      INTEGER      NNX
      PARAMETER (NNX=10)
C+++ MAXIMUM DIMENSION FOR ROWS (PHI)
      INTEGER      MAXROW
      PARAMETER (MAXROW=361)
C+++ MAXIMUM DIMENSION FOR COLUMNS (THETA)
      INTEGER      MAXCOL
      PARAMETER (MAXCOL=161)

C111
      COMMON /GEOPLA/
      +      XI      (3,NEX,NPX),
      +      V      (3,NEX,NPX),
      +      VP      (3,NEX,NPX),
      +      VN      (3,NPX),
      +      MEP      (NPX),
      +      MPX

C111
      COMMON /GEMEL/
      +      AC      (NNX,NCX),
      +      BC      (NNX,NCX),
      +      ZC      (NNX,NCX),
      +      ICR      (NNX,NCX),
      +      ICL      (3,NCX),
      +      VCL      (3,3,NCX),
      +      NEC      (NCX),
      +      NCX

C111
      COMMON /EDMAG/ VMAG(NEX,NPX)

C111
      COMMON /SHADWN/ COLS, ROWS, ANTENN(3),CTROID(3),
      +      MP,ME,NEXTME,MC,
      +      THET1,THET2,PH1,PH2,RESTH,RESPH,ALPH,
      +      UNIT(3),TRS(3),VRS(3,3),IUNIT,UNITF,UNITS,UN
      +      THZP,PHZP,THXP,PHXP,FILPNM,FILCNM
      COMMON /SHADWC/ INPFIL,OUTBUF(MAXCOL,MAXROW),
      +      FILCHC,FILCHP,FILCHR

C111
      COMMON /PATCUT/ VPC(3,3)

C111
```

```

C!!! The first set of declarations is the stuff in /SHADOW/ common bloc
C!!!
      INTEGER
      +      MP, ME, NEXTME, MC,
C! Plats#/edge#/cyl# variables.
      +      FILPM, FILCM,
C! Plate and cyl numbers for special filling
      +      COLS,
C! The size of the array subsection determined
      +      ROWS
C! by internal resolution requirements.

      REAL
      +      CTROID,
C! A geometric center of the object in question.
      +      ANIENN,
C! The antenna location in Ref Coord. System.
      +      THET1,
C! The lower theta end of the range.
      +      THET2,
C! The higher theta end of the range.
      +      PH1,
C! The lower phi end of the range.
      +      PH2,
C! The higher phi end of the range.
      +      RESTH,
C! The desired theta/phi resolution
      +      RESPH,
C! in units of radians/pixel.
      +      ALPH
C! Maximum allowed angular excursion.

      CHARACTER
      +      OUTFIL*1,
C! The output buffer which is displayed.
      +      INPFIL*63,
C! The filename of the input set.
      +      FILCHC,
C! special fill character for cylinders
      +      FILCHP,
C! special fill character for everything else
      +      FILCHR
C! special fill character for plates
      DATA  FILCHC, FILCHP, FILCHR / 'C', 'P', 'X' /
C!!!
C!!! From the /PIS/ COMMON block...
C!!!
      REAL PI, TPI, DPR, RPD
C!!!
C!!! From the /GEOPLA/ COMMON block...
C!!!
      INTEGER
      +      MEP,
C! Number of edges per plate
      +      MPX
C! Total number of plates
      REAL
      +      XX,
C! The array of plate corners
      +      V,
C! Edge unit vectors
      +      VP,
C! Edge unit binormals
      +      VN

```



```

C! Unit normal for each plate
C!!!
C!!! From the /GEOMEL/ COMMON block...
C!!!
      INTEGER
      +       NEC,
C! Number of sections per cylinder
      +       MCI
C! Total number of cylinders
      REAL
      +       AC,
C! Elliptic parameter along x-axis
      +       BC,
C! Elliptic parameter along y-axis
      +       ZC,
C! Cylinder endcaps in cyl coord sys
      +       TCR,
C! Angle endcap makes with positive z axis
      +       ICL,
C! Cyl coord sys origin
      +       VCL
C! Definition of cyl coord sys
C!
      INTEGER
      +       IUNIT
      REAL
      +       UNITF,
      +       UNITS,
      +       UNITH,
      +       UNIT,
      +       TRS,
      +       THZP,PHZP,THXP,PHXP,
      +       VRS,
      +       VPC,
      +       VMAG
      DATA UNIT/1.,.3048,0.0254/
C!
C!!!+
C!!! The following common block is for VMS/SMG$ software only.
C!!!
      INTEGER                                KBDID, KEYTBL
      COMMON /IERCOM/                        KBDID, KEYTBL
C!!!-

```

10.4 Non-FORTRAN VAX/VMS source files

This section contains listings of the source files used by the interactive code which are not written in fortran. They are used by the interactive interface and are needed only by the VMS utilities.

CDU Source file

This file is the source input for the Command Language Definiton Utility (CDU) which defines the available interactive commands.

```
!++
!
! File: SHACMD.CLD Edit: AAA1001
!
MODULE COMMAND_TABLES
IDENT /SHACMD 01-001/
!+
! FACILITY: Shadow
!
! ABSTRACT:
!
! This is the command language definiton source for the SHADOW
! program. It defines the interactive command interface under
! the VAX/VMS operating system.
!
! AUTHOR: Laszlo Takacs
!
! CREATED: 1-NOV-1985
!
! MODIFIED BY:
! 1-000 - Original. AAA 1-NOV-1985
! 1-001 - Laszlo Takacs 20-DEC-1985
! Added support for the SET FILL command and rearranged
! the SET PLATE and SET CYLINDER commands.
!-
!
! Show syntax
!
Define syntax show_fil_syntax routine show_fil
Define syntax show_out_syntax routine show_out
Define syntax show_inp_syntax routine show_inp
Define syntax show_uni_syntax routine show_uni
Define syntax show_ant_syntax routine show_ant
Define syntax show_coo_syntax routine show_coo
Define syntax show_pat_syntax routine show_pat
Define syntax show_sca_syntax routine show_sca
Define syntax show_win_syntax routine show_win
Define syntax show_key_syntax routine show_key
!
! Set syntax
!
Define syntax set_ant_syntax routine set_ant
Define syntax set_coo_syntax routine set_coo
Define syntax set_pat_syntax routine set_pat
Define syntax set_sca_syntax routine set_sca
Define syntax set_win_syntax routine set_win
Define syntax set_key_syntax routine set_key
Define syntax set_out_syntax routine set_out
parameter p1 value( required )
parameter p2 value( type=$file, required ),
prompt="filename"
qualifier plottable, default
```

qualifier printable, batch
qualifier echoing, default

```
Define syntax set_inp_syntax routine set_inp
parameter p1 value( required )
parameter p2 value( type=$file, required ),
prompt="input set"
```

```
Define syntax set_fil_syntax routine set_fil
parameter p1 value(required)
parameter p2 value(default="X"),prompt="character"
qualifier plate value(required,list), nonnegatable
qualifier cylinder value(required,list), nonnegatable
qualifier sequential nonnegatable, syntax=sequential
disallow any2 ( plate, cylinder, sequential )
```

```
Define syntax sequential routine set_fil
parameter p1 value(required)
! noqualifiers
```

```
!Define syntax set_pla_syntax routine set_pla
! parameter p1 value( required )
! parameter p2 value( required ), prompt="plate number"
! parameter p3 value( default="P" ), prompt="character"
! qualifier all syntax=set_placyl_all
```

```
!Define syntax set_cyl_syntax routine set_cyl
! parameter p1 value( required )
! parameter p2 value( required ), prompt="cyl number"
! parameter p3 value( default="C" ), prompt="character"
! qualifier all syntax=set_placyl_all
```

```
Define syntax set_placyl_all
parameter p1
parameter p2 value( default="X" )
```

```
Define syntax set_uni_syntax
parameter p1 value( required )
parameter p2, value( required, type=units_types ),
prompt="inches, feet, or meters"
```

```
Define syntax set_uni_meters_syntax routine set_uni_meters
Define syntax set_uni_inches_syntax routine set_uni_inches
Define syntax set_uni_feet_syntax routine set_uni_feet
```

! Type definitions.

```
Define type units_types
keyword inches, syntax = set_uni_inches_syntax
keyword meters, syntax = set_uni_meters_syntax
keyword feet, syntax = set_uni_feet_syntax
```

```
Define type set_types
keyword fill_character, syntax = set_fil_syntax
! keyword plate, syntax = set_pla_syntax
! keyword cylinder, syntax = set_cyl_syntax
keyword output_device, syntax = set_out_syntax
keyword input_set, syntax = set_inp_syntax
keyword units, syntax = set_uni_syntax
keyword antenna_location, syntax = set_ant_syntax
keyword coordinates, syntax = set_coo_syntax
keyword pattern_cut, syntax = set_pat_syntax
keyword scale_factor, syntax = set_sca_syntax
keyword window, syntax = set_win_syntax
```

```

keyword keypad_mode,  syntax = set_key_syntax, negatable

Define type show_types
  keyword fill_character,  syntax = show_fil_syntax
  ! keyword plate,  syntax = show_fil_syntax
  ! keyword cylinder,  syntax = show_fil_syntax
  keyword output_device,  syntax = show_out_syntax
  keyword input_set,  syntax = show_inp_syntax
keyword units,  syntax = show_uni_syntax
  keyword antenna_location,  syntax = show_ant_syntax
keyword coordinates,  syntax = show_coo_syntax
keyword pattern_cut,  syntax = show_pat_syntax
keyword scale_factor,  syntax = show_sca_syntax
keyword window,  syntax = show_win_syntax
keyword keypad_mode,  syntax = show_key_syntax
!
! Verb definitions.
!
Define verb set
  parameter p1, value( required, type=set_types ),
  prompt = "Set what"

Define verb show
  parameter p1, value( required, type=show_types ),
  prompt = "Show what"

Define verb help routine help_command
  parameter p1, value( type=$rest_of_line )
  qualifier library, label = helplib, default,
  value( default="sys$disk:[]shadow" )

Define verb spawn synonym dcl
  synonym $ routine dcl_command
  parameter p1, value( type=$rest_of_line )

Define verb exit routine exit_command
Define verb shadow synonym s routine shadow_command

!
! End of file SHACMD.CLD.
!
!--

```

Keypad initialization file

This file defines the initial keypad assignments for the interactive program at run time. It may be modified to allow customizing of the keypad interface.

```
!+
! SHADOW.KPD -
!
! This file starts up the keypad definitions for the SHADOW
! program. This is a user-definable file and may be altered.
!
! Laszlo Takacs, 20-DEC-1985
!-
!+
!
! Set up the GOLD key.
!
Def/key/noecho PF1 "" /if=default /set=gold
Def/key/noecho PF1 "" /if=gold /set=default

!
! Help & Shadow.
!
Def/key/term/echo PF2 "Help"
Def/key/term/echo PF3 "Shadow"

!
! Set up the toggle keypad-mode key.
!
Def/key/term/echo PF4 "Set keypad" /if=default
Def/key/term/echo PF4 "Set Nokeypad" /if=gold

!
! Define miscellaneous keys.
!
Def/key/echo/if=default KP7 "Set output "
Def/key/echo/if=default KP8 "Set input "
Def/key/echo/if=default KP9 "Set antenna" /terminate
Def/key/echo/if=default MINUS "Set window" /terminate
Def/key/echo/if=default KP4 "Set scale_factor"/termina
Def/key/echo/if=default KP5 "Set units" /terminate
Def/key/echo/if=default KP6 "Set coordinate"/terminate
Def/key/echo/if=default COMMA "Set pattern" /terminate
Def/key/echo/if=default KP1 "Set fill "
Def/key/echo/if=default KP2 "Set fill /plate=(1,X) "
Def/key/echo/if=default KP3 "Set fill /Sequential"/ter
Def/key/echo/if=default KP0 "Spawn"

!
Def/key/echo/if=gold KP7 "Show output" /terminate
Def/key/echo/if=gold KP8 "Show input" /terminate
Def/key/echo/if=gold KP9 "Show antenna" /terminate
Def/key/echo/if=gold MINUS "Show window" /terminate
Def/key/echo/if=gold KP4 "Show scale_factor"/termin
Def/key/echo/if=gold KP5 "Show units" /terminate
Def/key/echo/if=gold KP6 "Show coordinate"/terminate
Def/key/echo/if=gold COMMA "Show pattern" /terminate
Def/key/echo/if=gold KP1 "Show fill" /terminate
Def/key/echo/if=gold KP2 "Set fill /cylinder=(1,X) "
Def/key/echo/if=gold KP3 "Show fill" /terminate.
Def/key/echo/if=gold KP0 "Spawn "

!
! Enter key is same as return. Period is EXII.
```

```
!  
Def/key/term/echo PERIOD "Exit"  
Def/key/term/echo ENTER ""  
!  
! End of SHADOW.KPD  
!-
```

Chapter 11

VAX Implementation

This chapter describes the VAX/VMS implementation of the shadow program. The program has been split into two parts which are not used together. When the computer environment is the VAX/VMS operating system, then the more flexible interactive mode described in this chapter should be used. Assuming that the required files have been properly restored from the distribution medium, there are procedures provided to accomplish assembly of the code with minimum user effort.

11.1 Assembling the Code

On a VAX/VMS computer system, the following files are required to build and use the code. Both the interactive and non-interactive versions of the code can be run in any of the standard VMS ways, that is interactively, in a batch queue mode, or in a DCL subprocess. The actual building of the program takes place by invoking the procedure SHABLD.COM. The resulting executable file SHADOW.EXE can then be run with the RUN command.

SHABLD.COM A DCL command procedure to compile and link the files. This is the main assembly command file.

SHACMD.CLD A VMS Command Language Definition file used to define the interactive commands available.

SHACOM.FOR The one include file for the code common blocks. The other include statements that appear in the code reference system libraries.

SHADNI.FOR This contains the alternate code that is to be used when a non-interactive code is desired.

SHADNW.FOR This contains code that is very much dependent on the facilities of VMS and has been separated as such. It is an essential part of the interactive program.

SHADOW.FOR This is the main body of the code and is common to both interactive and non-interactive versions. It is standard FORTRAN-77.

SHADOW.HLB This is the VMS-format help library containing descriptions and examples of interactive commands.

SHADOW.KPD This is an initialization file used by the interactive program to equivalence certain functions to keys of the user's choice.

SHAPLT.COM This is a DCL command procedure invoking the NCAR graphics plotting software.

SHAPLT.FOR This is the FORTRAN program which reads the output produced by the code and calls appropriate NCAR routines to make a plot.

LABEL.DAT This file is read by the SHAPLT program in order to label the NCAR plots.

11.2 Running the Code

In order to run the code on VMS, the executable file created by the SHABLD procedure is necessary. The program is then run with the dcl RUN command.

A typical interactive session with the program might consist of the following elements in their approximate order of execution.

OUTPUT FILES Establish a set of output files with the SET OUTPUT command. The output files are of three types. Using the qualifiers of the SET OUTPUT command, any desired combination of output files may be generated.

PROCESS AN INPUT Issue a SET INPUT command which reads the geometry from the specified file. In order for the program to process input sets, this command must be issued prior to any mapping commands. This command is usually executed once per session.

DEFINE A WINDOW Using the SET WINDOW command, establish the angular range of interest. When the program begins, the size of the window is set to the full angular extent of the far-zone sphere. By specifying a smaller angular range, the user examines portions of the geometry in greater detail.

DEFINE A SOURCE With the SET ANTENNA command, establish the location of the source. This command is one of the more frequently entered commands. It applies units and coordinate transformations that apply from the set units and set coordinates command.

HIGHLIGHT ITEMS With the SET FILL command, the user may optionally cause parts of the geometry to be marked. This very useful command may be executed at any time before a SHADOW command.

GENERATE A MAP Cause the generation of a shadow map by issuing a SHADOW command. The shadow command is used after the user has set all desired parameters including the window and the antenna location. Without executing this command, the code does not calculate any shadowing.

REPEAT ANY OF THE ABOVE Perform one or more of the above actions repeatedly to obtain several maps. Most of the commands above may be executed in any order provided that the SHADOW command is executed last.

EXIT Terminate the shadow session with an EXIT command. An acceptable alternate mode of exit is eof, or control-Z.

In order to make life easier by reducing the number of keystrokes required to enter interactive commands, a facility is provided with which the user may associate whole command strings with a single key. When the shadow program begins executing, it loads a set of predefined key definitions from a file. The user may edit this file to customize the keypad definitions to his/her liking. Since the file is loaded automatically, the only restriction on its use is that it must exist in the current process default directory and must be accessible at run time. The details about these interface routines and what they do may be found in the VAX/VMS Runtime Library Reference Manual.

11.3 Modifying the code

Modifications to the source code by the user can be performed, but of course the outcome cannot be predicted beforehand. One predictable user modification is changing the program's PARAMETER statements in the include file SHACOM.FOR. This would be necessary (and sufficient) to allow the program to deal with a greater number of plates or to construct a shadow map with greater resolution than the current maximum.

Chapter 12

Non-VAX Implementation

This chapter discusses how to implement the code on a different computer than a VAX. The obscuration code, SHADOW, has been separated into two main parts. The FORTRAN 77 part, is not VAX dependent and is contained in a file called SHADOW.FOR. Most of the rest of the files are VAX dependent and are used mostly for interactive features. Although, it is possible that other types of machines will have similar routines that will allow interactive manipulation, it is not possible here to suggest how this may be accomplished. It is assumed that the easiest way to use SHADOW on a non-VAX would be to run it in a non-interactive mode.

The main program in the default version of the file SHADOW.FOR is designed to be used with the non-FORTRAN 77 interactive version. A file called SHADNI.FOR contains a main program designed to be used in a non-interactive mode. It is listed in section 10.2. The main programs can be easily exchanged.

Note that the only other part of the code is this part that is non-FORTRAN 77 is the INCLUDE statement. This has been retained because many computer systems support this statement. It is used to include the lines of code in the named file in the spot that it is called as if the lines had been in that spot. It provides a powerful means of putting commonly defined parameters used throughout the code in one place. In this case, it is used to include the file SHACOM.FOR which contains COMMON blocks and PARAMETER statements that define the dimensions of arrays that store the geometry. If it is desired to increase the number of plates, edges per plate, cylinders, or rims per cylinder, etc; they can be changed in one spot. Please see the listing for this file elsewhere in this manual. The INCLUDE statement can be easily removed by hardwiring the contents of the file SHACOM.FOR into the text at the main program and the subroutines ABSCIN, CAPINT, CYLINT, CYLROT, DOCYLS, DOCYL, DOPLAS, DOPLA, GEOM, INITGF, PLAINT, SCAN, SCANC, SPANC, and WRTOUT.

The code can now be compiled, linked, and run. The user communicates with the code through the non-interactive commands. This allows almost the same capability. The only information that does not have a command to change its behavior is the fill options and the input and output file names. The fill options can be accessed through the main program. The listing below has comment lines referring to the place that the fill operations may be changed.

The input and output files can be named using assignments to the logical unit numbers for the given operation. The input file is read on logical unit #5. The echo file is written

on logical unit #6. The printable shadow map is written on logical unit #7. The plottable shadow map is written on logical unit #10. On a VAX the ASSIGN VMS command would be used.

Note that the user can specify more than one source. The non-interactive operation will run a shadow map for each source individually. The receiver will not be counted. If the user wants to look at the shadow map for a receiver, they should be treated in this code as if they are a transmitter (source).

Chapter 13

NCAR Plot Program

The shadow map can be plotted using graphical means. The SHADOW code will write a unformatted file that can be used for interfacing to special purpose plotting programs. It writes this file on logical unit #10. In the interactive mode the file name is specified by using the SET OUTPUT commands /PLOTTABLE option. In the non-interactive mode the file name is specified using an assign statement.

There are many ways to plot the resulting shadow map. Presently, there is little standardization between system for plotting. This may change with the advent of GKS, but for now, it can not be assumed that different organizations have compatible plotting capabilities. This chapter suggests one possible means to plot the output. It uses the National Center for Atmospheric Research (NCAR) graphics package [5]. It has been tried on The Ohio State University ElectroScience Laboratory's computer system and NASA Langley Research Center's computer system, both VAX 11/780s, with almost the same results. It is still not possible, however, to assume that it will run everywhere the same way.

The program is listed for the convenience of possible users, knowing that some conversion may be necessary. The code is written in basic NCAR subroutine calls. Consult your local system information on how to link to your systems NCAR graphics subroutines. In addition, it is not written completely in standard FORTRAN 77. There are a few VAX extensions used, such as some of the options in the OPEN subroutine and some comment lines use the non-standard exclamation point. These changes will be minor.

Note that the plot of the shadow map will have grid lines. There is another option given for a map without grid lines. This can be used by commenting out the call to subroutine GRIDL, and uncommenting the call to subroutine PERIML.

The file name containing the maps to be plotted are placed in the first line of a file named LABEL.DAT. The LABEL.DAT file also contains the header information to be placed at the top of the plot for future identification and reference. The code will loop through the specified shadow map file until all the shadow map contained in the file are plotted. A sample version of a LABEL.DAT file is given after the code listing. It shows a shadow map being read off of file FOR010.DAT which contains two shadow maps.

Listing of code to plot shadow map using NCAR:

```

0001      PROGRAM PLTOSU
0002      DIMENSION XDUM(2), YDUM(2), NC(5)
0003      INTEGER COLS, ROWS
0004      CHARACTER*80 LABELS(5), XLAB, YLAB, INF
0005      CHARACTER*(*)  XFORMA, YFORMA
0006      BYTE   BYTE
0007      C
0008      C These are character parameters for the plotting output.
0009      C
0010      PARAMETER          ( XFORMA = '(F6.1)' )
0011      PARAMETER          ( YFORMA = '(F6.1)' )
0012      C
0013      DATA XLAB /' PHI '/
0014      DATA YLAB /' THETA '/
0015      DATA NC / 5*72 /
0016      C
0017      C Read a header from F0R005.  Open the file readonly so that other users
0018      C can read it without needing write access to the file.
0019      C
0020      OPEN ( UNIT=5, TYPE='OLD', READONLY )
0021      READ ( 5, FMT='(A)' ) INF
0022      C
0023      C Read the header info from the data file.  Open it unformatted.
0024      C
0025      OPEN(UNIT=10,FILE=INF,TYPE='OLD',FORM='UNFORMATTED',READONLY)
0026
0027      13  READ(10,END=9999) COLS,THET1D,THET2D,RESTHD
0028      READ(10,END=9999) ROWS,PH1D,PH2D,RESPHD
0029
0030      ISCX = -2
0031      ISCY = -2
0032      XMIN = PH1D
0033      XMAX = PH2D
0034      YMIN = -THET2D
0035      YMAX = -THET1D
0036      NDX = 4
0037      NTX = 2
0038      NDY = 4
0039      NTY = 2
0040      C
0041      C Read the label info for this plot.
0042      C
0043      READ ( 5, * ) LABELS(1)
0044      READ ( 5, * ) SX, SY, SZ, SPRX, SPRY, TRX
0045      READ ( 5, * ) ZTHET, ZPHI, XTHET, XPHI
0046      C
0047      C Format the labels for the plot (via internal write statements.)
0048      C
0049      WRITE (LABELS(2), 1100) SX, SY, SZ
0050      WRITE (LABELS(3), 1200) ZTHET, ZPHI, XTHET, XPHI
0051      WRITE (LABELS(4), 1300) SPRX, SPRY
0052      WRITE (LABELS(5), 1400) TRX
0053      C
0054      C CALL INFOPLT(2,XDUM,YDUM,XMIN, XMAX,YMIN,YMAX,ISCX,
0055      *          NDX,NTX,ISCY,NDY,NTY,XLAB,5,YLAB,7,
0056      *          5,LABELS,NC,0,-1,1)
0057      C
0058      C Define a mapping window from data to plot
0059      C
0060      CALL SET (
0061      +          0.12,
0062      +          0.84,
0063      +          0.12,

```

```

0064      +                0.84,
0065      +                XMIN,
0066      +                XMAX,
0067      +                YMIN,
0068      +                YMAX,
0069      +                1 )      ! Do a linear-linear plot.
0070      C
0071      C A call to labmod might help the output look nicer.
0072      C
0073      CALL LABMOD (
0074      +                %REF( XFORMA ),
0075      +                %REF( YFORMA ),
0076      +                LEN( XFORMA ),
0077      +                LEN( YFORMA ),
0078      +                1,
0079      +                1,
0080      +                0,
0081      +                0,
0082      +                0                )
0083      C
0084      C Put labels on plot
0085      C
0086      X MID=0.5*(XMIN+XMAX)
0087      Y MID=0.5*(YMIN+YMAX)
0088      XDEL=(XMAX-XMIN)/36.
0089      YDEL=(YMAX-YMIN)/36.
0090      XL=XMIN+0.5*XDEL
0091      DO 100 IL=1,5
0092      YL=YMAX+(6-IL)*YDEL
0093      100 CALL PWRIT(XL,YL,%REF(LABELS(IL)),NC(IL),1,0,-1)
0094      C
0095      C Define the perimeter of the plot with a grid.
0096      C
0097      CALL GRIDL (
0098      +                NDX,                ! Number of MAJOR
0099      +                NTX,                ! Number of MINOR
0100      +                HDY,                ! Number of MAJOR
0101      +                NTY )                ! Number of MINOR
0102      C
0103      C Theta and Phi Axis Labels
0104      C
0105      YBOT=YMIN-2.5*YDEL
0106      CALL PWRIT(XMID,YBOT,%REF(XLAB),5,1,0,0)
0107      XSID=XMIN-5.0*XDEL
0108      CALL PWRIT(XSID,YMID,%REF(YLAB),7,1,90,0)
0109      !C
0110      !C Use this call if you don't want grid lines.
0111      !C Define the perimeter of the plot.
0112      !C
0113      ! CALL PERINL (
0114      ! +                NDX,                ! Number of MAJOR
0115      ! +                NTX,                ! Number of MINOR
0116      ! +                HDY,                ! Number of MAJOR
0117      ! +                NTY )                ! Number of MINOR
0118      C
0119      XINC = 1.8
0120      YINC = 0.9
0121      ISYM = 1
0122      C
0123      C Loop on rows then on columns.
0124      C
0125      DO 10 J = 1, ROWS
0126      X = RESPHD*(J-1)+PH1D
0127

```

```

0128      DO 20 I = 1, COLS
0129      READ ( 10, END=999 ) BYTE
0130      IF ( BYTE .NE. 32 ) THEN
0131      Y = -(RESTHD*(I-1)+THET1D)
0132      C
0133      C      CALL PLTSYM( X, Y, XINC, YINC, ISYM )
0134      C
0135      C Plot the symbol on the page.
0136      C
0137      CALL PWRIT(
0138      +
0139      +
0140      +
0141      +
0142      +
0143      +
0144      +
0145      END IF
0146      20  CONTINUE
0147      10  CONTINUE
0148      C
0149      C "Frame" the NCAR output.
0150      C
0151      999  CALL FRAME
0152      GOTO 13
0153      C
0154      C Close the input file and stop.
0155      C
0156      9999 CLOSE ( UNIT=10 )
0157      CLOSE ( UNIT=5 )
0158      STOP 'NCAR/Shadow plot completed.'
0159      C
0160      C Formats go down here.
0161      C
0162      1100 FORMAT (' ANTENNA LOCATED AT ',2(F7.1,', '),F7.1)
0163      1200 FORMAT (' ANTENNA ORIENTATION: ',3(F7.1,', '),F7.1)
0164      1300 FORMAT (' SOLAR PANELS ROTATED ',F7.1,', ',F7.1)
0165      1400 FORMAT (' THERMAL RADIATORS ROTATED ',F7.1)
0166      END

```

Listing of sample LABEL.DAT file:

```
FORO10.DAT;  
' SHADOW TEST1 FOR CASE AN6S1'  
25. 15. 256.5 0. -52. 0.  
0. 0. 90. 0.  
' SHADOW TEST2 FOR CASE AN6S1'  
25. 15. 256.5 0. -52. 0.  
0. 0. 90. 0.
```


Standard Bibliographic Page

1. Report No. NASA CR-178099		2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle MANUAL FOR OBSCURATION CODE WITH SPACE STATION APPLICATIONS				5. Report Date May 1986	
				6. Performing Organization Code	
7. Author(s) R. J. Marhefka and L. Takacs				8. Performing Organization Report No. 716199-7	
9. Performing Organization Name and Address The Ohio State University ElectroScience Laboratory Department of Electrical Engineering 1320 Kinnear Road Columbus, Ohio 43212				10. Work Unit No.	
				11. Contract or Grant No. NSG-1498	
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546				13. Type of Report and Period Covered Contractor Report	
				14. Sponsoring Agency Code 482-54-23-15	
15. Supplementary Notes Langley Technical Monitor: E. M. Bracalente					
16. Abstract The Obscuration Code, referred to as SHADOW, is a user-oriented computer code to determine the cast shadow of an antenna in a complex environment onto the far zone sphere. The surrounding structure can be composed of multiple composite cone frustrums and multiple sided flat plates. These structural pieces are ideal for modeling space station configurations. The means of describing the geometry input is compatible with the NEC - Basic Scattering Code. In addition, an interactive mode of operation has been provided for DEC VAX computers. The first part of this document is a User's Manual designed to give a description of the method used to obtain the shadow map, to provide an overall view of the operation of the computer code, to instruct a user in how to model structures, and to give examples of inputs and outputs. The second part is a Code Manual that details how to set up the interactive and non-interactive modes of the code and provides a listing and brief description of each of the subroutines.					
17. Key Words (Suggested by Authors(s)) Obscuration computer code Shadow map Plate models Cone frustrum models Space Station applications			18. Distribution Statement RESTRICTED Subject Category 32		
19. Security Classif.(of this report) Unclassified		20. Security Classif.(of this page) Unclassified		21. No. of Pages 237	22. Price



3 1176 00190 5323