NASA Contractor Report 159273-1
Users Manual

NASA-CR-159273-1
19800024182

Thermal Radiation Analysis System TRASYS II

R. G. Goble
C. L. Jensen

MARTIN MARIETTA CORPORATION
Denver, Colorado 80201

CONTRACT NAS1-15683
JUNE 1980

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

Contract NAS9-15304
MCR-73-105 (Revision 2)

THERMAL RADIATION ANALYSIS SYSTEM

T
R
A
S
Y
S

II

USER'S MANUAL

June 1979

Approved by:

Carl L. Jensen

Carl L. Jensen
Program Manager

MARTIN MARIETTA

Prepared for:

National Aeronautics and Space Administration
Lyndon B. Johnson Spacecraft Center
Contract NAS9-15304

Prepared by:

Carl L. Jensen

Carl L. Jensen

[illegible]

Richard G. Goble

FOREWORD

The Thermal Radiation Analyzer System (TRASYS) program marks the first instance that thermal radiation analysis has been put on the same basis as thermal analysis using program systems such as MITAS and SINDA. As with these thermal analyzer programs, the user is provided the powerful options of writing his own executive, or driver logic and choosing, among several available options, the most desirable solution technique(s) for the problem at hand. In addition, many features never before available in a single radiation analysis program are provided.
Among the more important are:

- Up to 1000 node problem size capability* with shadowing by intervening opaque or semi-transparent surfaces;
- Choice of diffuse, specular or diffuse/specular radiant interchange solutions;
- Capability for time variant geometry in orbit;
- Choice of analytically determined or externally supplied shadow data for environmental flux calculations;
- Form factors and. environmental fluxes computed using an internally-optimized number of surface grid elements, selected on the basis of user-supplied accuracy criteria;
- Choice of an element to element double integration technique and a precision Nusselt Sphere technique for computing form factors;
- A general edit capability for updating thermal radiation model data stored on tape.

*Depending on Computer application.

- A plot package that provides a pictorial representation of the user's geometry, orbital/orientation parameters, and heating rate output data as a function of time.

- Capability to automatically equate Form Factors for nodes that have been duplicated or imaged to the corresponding original node pair, to eliminate costly redundant computations.

TRASYS is indebted to a number of predecessor programs in the thermal radiation analysis field. The major contributors were HEATRATE, MTRAP version 2.0, and RADFAC.

This User's Manual represents a concerted effort to document the capabilities of TRASYS and will, hopefully, serve the twofold purpose of instructing the user in all applications and serve as a convenient reference book that presents the features and capabilities in a concise, easy-to-find manner.

This User's Manual was generated under a series of NASA Contracts. The technical monitoring was provided by Mr. Robert A. Vogt of the Thermal Technology Branch of the Structures and Mechanics Division, NASA Lyndon B. Johnson Space Center. His helpful suggestions during the development of TRASYS are gratefully acknowledged. TRASYS would not exist without the superb design and programming efforts of Messrs. R. E. Paulson and R. J. Connor, who were responsible for generating the majority of the TRASYS code. Their efforts are gratefully acknowledged. Extensive thanks are also due Mr. G. M. Holmstead for his efforts in developing the direct irradiation program segment and for the valuable consulting effort he performed during the course of program development. Mr. R. G. Goble is also recognized for his praiseworthy efforts in developing the specular-diffuse radiation interchange segment, the orbit plotter segment, and for his solutions of many knotty problems that cropped up during program checkout.

Revision Schedule

Revision 1 (Revision to TRASYS Users' Manual, May 1973)	August, 1977
Revision 2	June, 1979

CONTENTS

Contents (con't.)

Contents (Con't.)

APPENDICES

*In a separate volume.
Available as NASA CR-159273-2.

FIGURES

TABLES

1. INTRODUCTION

1.1 WHAT IS TRASYS?

The Thermal Radiation Analysis System is a digital computer software system with generalized capability to solve the radiation related aspects of thermal analysis problems. When used in conjunction with a generalized thermal analysis program such as the Systems Improved Numerical Differencing Analyzer (SINDA) program, any thermal problem that can be expressed in terms of a lumped parameter R-C thermal network can be solved. The function of TRASYS is twofold. It provides:

a. Internode radiation interchange data; and
b. Incident and absorbed heat rate data from environmental radiant heat sources.

Data of both types is provided in a format directly usable by the thermal analyzer programs.

One of the primary features of TRASYS is that it allows the user to write his own executive or driver program which organizes and directs the program library routines toward solution of each specific problem in the most expeditious manner. The user also may write his own output routines, thus the system data output can directly interface with any thermal analyzer using the R-C network concept.

Other outstanding features of TRASYS include:

a. A plot segment that provides pictorial plots of the problem geometry as well as output data;
b. Restart capability that prevents loss of output and is very convenient to use;
c. A generalized edit capability for conveniently changing data on the restart tape;
d. Geometry may vary with time;
e. Central processor memory required is adjusted dynamically according to problem size;
f. A choice of solution techniques in the calculation of geometric form factors, depending upon whether high precision or minimum computer run time is the primary objective.

Allowable problem size is limited by the amount of central memory available. At the Univac 1110 installation at Johnson Spaceflight Center, 65000 words of high speed core are available, which allows for problems up to approximately 650 nodes. At this installation, larger problems require a program modification to utilize extended core for some data arrays.

A definitive run time formula for TRASYS cannot be stated due to the strong dependence on the experience of the analyst, the number of shadowing surfaces in a given problem and the way the nodes are geometrically arranged relative to each other. Numerous shadowing nodes in close proximity to each other will add to run time. Also, heat rates for non-circular orbits generally require much more run time than for circular planet oriented attitude because form factors to the planet must be computed for each point in orbit.

For example, form factors for a relatively large 500 node model may require 2 to 20 hours of SUP time on a Univac 1110. In general, run time is proportional to the third or fourth power of the problem size. Thus, a 100 node problem may only require 6 to 11 minutes for form factors.

The TRASYS system consists of two major components: (1) the preprocessor, and (2) the processor library. The preprocessor has two major functions. First, it reads and converts the user's geometry input data into the form used by the processor library routines. Second, it accepts the users driving logic written in the TRASYS modified FORTRAN language that directs user-provided and/or library routines in the solution of the problem. The processor library consists of FORTRAN language routines that perform the functions commonly needed by the user. The user has, in some cases, a choice of solution techniques to perform the same function.

1.2 SYSTEM STRUCTURE

In the usual engineering environment, a programmer is commissioned to prepare an applications program which is subsequently made available to the engineer on a production basis. The engineer supplies input data and receives output data, as shown in Figure 1-1.

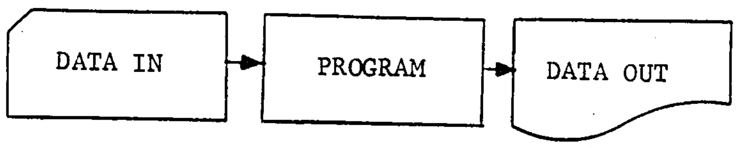


FIGURE 1.1: BASIC FLOW IN USING AN APPLICATIONS PROGRAM

Changes to the logic and equations are difficult for the program user to implement conveniently since they must be written in a computer-oriented language and submittal may be required through a formal programming organization. When TRASYS is used, however, the engineer need only call on the programmmer to supply a standard deck of computer oriented "control cards" which will call the various elements of the system into action in the proper sequence. The engineer then formulates his problem in the engineering-oriented TRASYS language, assembling both data and solution techniques (i.e., logic and equations) into this card deck, which then serves as the complete input to the TRASYS system. Programmer support has been minimized since the bulk of the programming effort is already built into the TRASYS preprocessor and processor library. The engineering user need only specify the data and the order and type of "program building blocks" which he deems necessary for the solution of his problem, as illustrated in Figure 1-2.

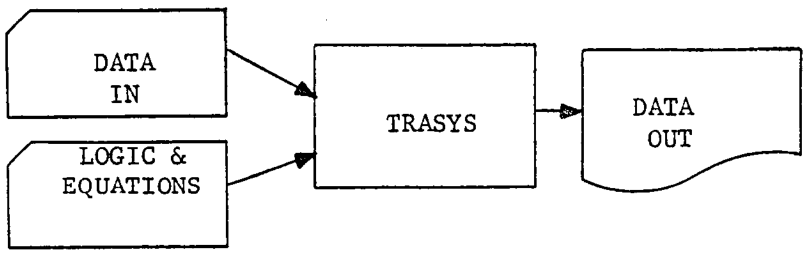


FIGURE 1-2: BASIC FLOW IN USING TRASYS

It should then be evident that TRASYS is much more than an applications program. It has, in fact, all of the functions and capabilities of a special purpose operating system. Since most computers in current use in engineering environments already have operating systems built around a FORTRAN compiler, TRASYS is designed to augment the existing FORTRAN system. Hence, the TRASYS library serves as an extension to the existing FORTRAN library, and the TRASYS program serves as a preprocessor to (i.e., it preceeds) the existing FORTRAN compiler. This augmentation arrangement is illustrated in Figure 1-3.

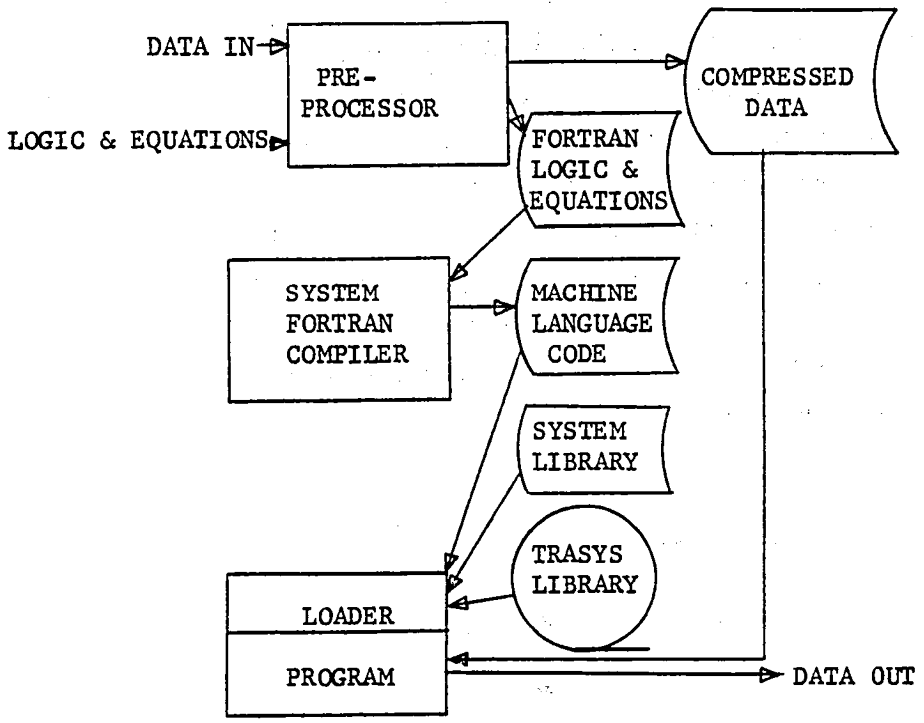


FIGURE 1-3: DETAILED INTERNAL FLOW OF TRASYS

When using the full capability of TRASYS, the engineer will be required to exert a programming effort of sorts, in a language consisting of FORTRAN statements and problem oriented TRASYS statements that are FORTRAN related. This, together with the wide variety of options and features offered by the system, suggests an appropriate word of caution: TRASYS is a comprehensive system which cannot be mastered overnight. The prospective user should not assume that a cursory review of the Instruction Manual will lead to immediate success, nor should he assume that this manual represents a "cookbook" which will eventually yield to a plodding and rigid adherence to each and every rule. In presenting instructions on the use of a computer program, it is not possible to completely avoid some "cookbook-like" sections; however, every effort has been made to explain the "why" and "how" behind each rule, option, and feature, with the intent of encouraging the reader to think about and understand TRASYS in depth. To help the novice user, an attempt has been made to default much of the required input to normally used values so that the user need not define them.

2. BACKGROUND INFORMATION

2.1 TYPOGRAPHICAL CONVENTIONS

2.1.1 Punched Cards

The reader is (or soon will be) familiar with the standard 80 column punched card. It is the user's primary means of conveying his input data and logic to TRASYS.

The program input format design is predicated on minimum dependence upon data/card column relationships. Most card input is covered by one column rule: card columns 1 thru 6 inclusive comprise the control field and columns 7 through 72 comprise the data field. Data in the control field are used by read routines to identify the type of data to expect in the cards' data field. In this manual, the typographical convention shown in Figure 2-1 will be used to indicate the card columns of interest. (Card columns 1,7 and 12 in this case).

```
CC1          CC7          CC7
                            2
```

FIGURE 2-1: SAMPLE CARD COLUMN DESIGNATIONS

Throughout the rest of the manual (in contrast to Figure 2-1) punched cards will not be identified as figures. Whenever material is presented with one or more indicators with the format CCX directly above, a punched card is indicated. The card data will always be presented as Gothic capitals and /or numerals. For example, a card format might be shown as follows:

```
CC1        CC7
  TITLE      THIS IS A SAMPLE TITLE CARD
```

In general, the character directly below the column number begins the relevant data field.

2.2 FILE AND TAPE CONVENTIONS

Since TRASYS can be implemented on a variety of computers, it is necessary to refer to data storage media by some nomenclature which will be independent of the particular system configuration. FORTRAN "logical unit numbers" are often used for this purpose, but were rejected for use in this manual because certain installations impose restrictions on the type of physical storage device which may be assigned to a given unit. Instead, each serial access storage device referenced by TRASYS is given a proper name as follows:

"PURPOSE tape"

Hence, for example, the Restart Output Tape referred to as the RSO tape, contains the results of processing the user's data, and Edit Input Tape CMERG contains input for merging, using the edit routine. "Tape" is used as part of the name only because a reel of magnetic tape is normally associated with computer storage. However, any "Tape" may, in fact, be a disk file, a drum file, a punched paper tape, or a magnetic tape, at the option of the user. Appendix G contains a list of the system-oriented unit designations for each of the "Tapes" mentioned in this manual, along with the recommended type of storage device to which these units should be assigned.

On the other hand, when speaking in general about saving or retrieving data on or from a serial access storage device, the generic term "file" will be used.

2.3 TERMS AND DATA CONVENTIONS

The words SUBROUTINE and ROUTINE are generally used interchangeably. Occasionally the word "LINK" will appear in this manual. It is synonomous with SEGMENT. A program SEGMENT is a specific collection of routines used to do a specific processing job, such as the calculation of radiation interchange factors. Generally, the routines comprising a segment are brought into core together, and in this sense, a segment can be thought of as an OVERLAY, where that concept is applicable to a particular computer operating system. INTEGER and FIXED POINT mean the same thing, as do REAL and FLOATING POINT.

The term HOLLERITH* is applied to strings of alphanumeric characters. The term DATA VALUE will be taken to mean one element of the set of all integers, floating point numbers, and 6-character** Hollerith strings. In the text, DV, DV1, DV2, etc. refer to floating point values. NDV, NDV1, NDV2, etc. are integer.

A data value may also be any arithmetic FORTRAN expression, which may contain variable names. For example:

```
        ANAME = 6.8*4.317/CON1
```

is allowed. On the other hand, function calls such as:

```
        DNAME = 4.7* SIN(1.73)
```

are not. The user is also cautioned to avoid mixed-mode expressions.

A data value should not be split between cards unless it is a subroutine CALL Argument in which case a continuation FLAG in Col 6 must exist as in the Standard FORTRAN continuation.

Integers will be shown in print as a sequence of digits preceded, optionally, by a plus or minus sign. Floating point numbers will appear in print as a sequence of digits with a leading, trailing, or imbedded decimal point, prefixed, optionally, by a plus or minus sign, and suffixed, optionally, by an exponent (to the base 10) denoted as the letter E followed by an integer. Hollerith strings of characters will be delineated in print by asterisks. These are necessary because blanks are valid characters and have a specific binary code (i.e., they do not appear on the printed page, but they do appear explicitly in the computer).

---

*The Hollerith code is actually a binary code for representing ALPHANUMERIC CHARACTERS ON PUNCHED CARDS. Other common binary codes for representing alphanumeric characters include BCD, ASCII, EBDIC, and FIELDATA. The use of Hollerith to denote character strings in general is purely arbitrary.

**A 6-character string may be stored in one UNIVAC computer word. TRASYS implementations on other computers may provide more or less characters per word. In the general case, a Hollerith data value would contain as many characters as will fit in one computer word.

In addition to DATA VALUES, another entity, called an IDENTIFIER, REFERENCE FORM, or VARIABLE, will be used (in a programming sense). For example, consider the following statement:

PI = 3.14

In this case, 3.14 is a floating point data value, and PI is an identifier. Note that PI is different from *PI* which is a Hollerith string.

The data field of any card may be terminated by the characer $. This terminates any further data read operations for that card, and allows the user to enter comment data to the right of the $. This may tempt the user to enter a comment to the right of it in an otherwise blank card. This results in an empty data field and a fatal error. Instead, comment cards are formatted in the classic FORTRAN manner, that is, with a C in card column 1. Such comment cards may be used in any of the data blocks.

The terms and data conventions stated thus far hold for all situations except one. In the operations data block, a portion of the input is in classic FORTRAN statements which are processed only by the FORTRAN compiler. Thus, Hollerith information must be supplied in a form compatible with the compiler. For example:

```
a)      CALL NDATAS (1, 3HALL,0,2HNO,3HYES)
        FFPNCH   =   3HPUN
```

are correct, while

```
b)      CALL NDATAS (1,*ALL*,0,*NO*,*YES*)
        FFPNCH   =   *PUN*
```

are not, and will result in FORTRAN errors in subroutine ODPROG, the routine generated from the users operations data block. This situation has resulted in a large number of errors over the years and it has been alleviated by putting all the Hollerith words commonly used by TRASYS in DATA statements in subroutine ODPROG. Thus, the following is allowed:

```
c)      CALL NDATAS (1,ALL,0,NO,YES)
        FFPNCH   =   PUN
```

The user is cautioned that this approach cannot possibly anticipate arbitrarily selected inputs such as configuration names. The cautious user has two options: check carefully the reserved word list (Appendix A) for the names in DATA statements, or always use the classic "H" format of example a) above. The examples supplied herein will always show the H format to emphasize this situation.

A TRASYS "model" is a name (up to 6 characters) used to identify an input deck that has been written to a restart tape. When encountered herein, the words "TRASYS model" refer to an input deck, less the options and edit data blocks.

A TRASYS configuration name is used to identify a particular geometric configuration defined within a run. If the variable geometry option is used, a TRASYS "model" may have reference to several configuration names.

2.4 DEFINITIONS

Albedo: The diffuse reflectivity of a planet in the solar waveband.

Direct irradiation (DI): The thermal energy, in flux units (energy per unit time per unit area) incident upon a node. In general, direct irradiation consists of solar, planetary and planetary albedo components.

Form factor (FF): The fraction of the total energy leaving a node i that reached a node j in a straight line path.

Gray body factor (GB): The fraction of the energy within a specified waveband emitted from a diffusely emitting, diffusely reflecting node i that reaches a similar node j by all possible paths. All nodes involved in reflections must also be diffusely emitting, diffusely reflecting.

Heat rate: The energy per unit time absorbed by a node. The components of heat rates are direct & reflected solar, direct & reflected albedo and direct & reflected planetary infrared.

Node: A finite, regular portion of a rectangle, polygon, disk, cylinder, sphere, core or paraboloid that is expected to be isothermal (everywhere the same temperature) in the user's RC (resistance-capacitance) thermal model (interchangable with nodal surface).

Optical properties: The physical properties of a surface that interacts with radiant energy. See Appendix I for definitions of the properties used in the TRASYS thermal radiation model.

Planetary IR or planetary infrared: The component of direct irradiation incident on a surface that is due to the thermal emission of the planet's surface.

RADK, or radiation conductor: A single card image ready for input into a thermal analyzer program that computes temperatures (e.g., SINDA). The information on the card consists of a conductor number, the numbers of the two nodes it connects and the value of the radiation conductor.

Radiant interchange factor: Exactly analagous to a gray body factor, except that specular and specular-diffuse reflections are allowed.

Shadow factor: In direct irradiation, the quotient of the energy incident upon a node and the energy incident upon it if there were no intervening surfaces blocking the incoming energy. In form factors, the quotient of a form factor and the analogous form factor that would exist if there were no intervening surfaces between the two surfaces involved.

Surface: A finite, regular portion of a rectangle, polygon, disc, cylinder, sphere, cone or paraboloid. May be subdivided into any number of nodes.

3.1 INPUT DECK

3.1 Introduction to the Input Deck

3.1.1 Basic Concepts

TRASYS input decks consist of two fundamental parts. Part I consists of the EDIT/CONTROL blocks. These blocks do not participate at all in the definition of the mathematical model of the thermal radiation problem. This part provides basic program control and provides the user with his edit capability. Part II is referred to hereinafter as the TRASYS MODEL. This part is made up of the data blocks that describe the user's problem in terms of geometry definition and drive logic. The options and source edit data blocks comprise the EDIT/CONTROL portion of the input data. Examples of options data are problem title information, restart tape identification, input data punch/no punch, list/no list flags and a documentation data list/no list flag. A one line (CC7-72) problem title is entered in the options data block. This title will appear on each page of output printed by the standard library output routines during execution. The Edit Data block allows the user to do a line by line edit on previously taped input data. The edit capability also allows the user to conveniently merge portions of one or more models into his master input model.

The MODEL portion of the input deck consists of the following blocks:

Documentation Data
Array Data
Quantities Data
Surface Data
Block Coordinate System (BCS) Data
Form Factor Data
Shadow Data

Flux Data
Correspondence Data
Operations Data
Subroutine Data

In general, the largest block is the surface data. This block is the user's means to describe the geometry of the surfaces that participate in the radiant interchange of his problem. Because of the surface data block's size, a number of input options, differing in format and concept, are provided. This allows the user to choose the most convenient means of defining the different parts of his geometry; thus easing his most laborious task.

Another type of data that may comprise a large portion of the user's input may consist of information normally considered to be program ouput or interim output. A user may have, for example, a large portion of the form factors needed for his solution available from some external source. Using the form factor data block, he may enter this data and save much processing time.

Another data block that allows the user to take advantage of large blocks of previously known data is the shadow data block. If DI shadow factor tables are known for a portion of this model, the user may enter them through this block and avoid computing them in a shadow factor generating run.

The remaining data blocks used for general alphanumeric input are the correspondence data, array data and quantities data blocks. The correspondence data provides the capability for the user to redesignate node numbers, and/or combine a number of nodes into single nodes. The array data block provides a convenient input point for any array data that the user may require. Array data may be integer or floating point data value strings, or Hollerith strings. The quantities data block performs the same function as the array data block except that single values are entered rather than strings. If the user desires, he may enter an extended written description of his problem in the documentation block. This will appear at the user's option at the beginning of his printed output and will be stored with the remainder of his input data on his RSO tape.

The user's driver logic is entered in the operations data and subroutines data blocks. The operations data block consists of a series of calls to user-addressable subroutines and computation segments arranged in a series of steps that are used for orderly handling of the output data in out-of-core storage. Operations block subroutine calls are primarily used to input and update appropriate problem parameters. The calls to the computation segments are what actually result in the generation of output data. The operations block subroutine calls are in classic FORTRAN format, and the user has at his disposal the FORTRAN V language for coding specialized operations block logic. In the operations block, the user has access to all variables he identified in his array and quantities data, plus an extensive list of program variables located in labeled common.

The subroutines data block contains FORTRAN language subroutines that are either user-called or called by the various computation segments. Routines found in the subroutines block bearing the same name as processor library routines will compile in place of the library routine, thus giving the user the capability to override any program function he desires.

3.1.2 Basic Structure

The basic structure of the TRASYS input deck is shown in Figure 3-1. This figure illustrates the two EDIT/CONTROL blocks and the 11 MODEL blocks in their correct input sequence. Format of the header cards that lead each block is defined in Figure 3-2. The data blocks must appear in the order shown in Figure 3-1. Any block may be omitted, along with its header card if it is not required for the problem at hand.

The EDIT/CONTROL blocks are discussed in Section 3.2; the model data blocks in Section 3.3.

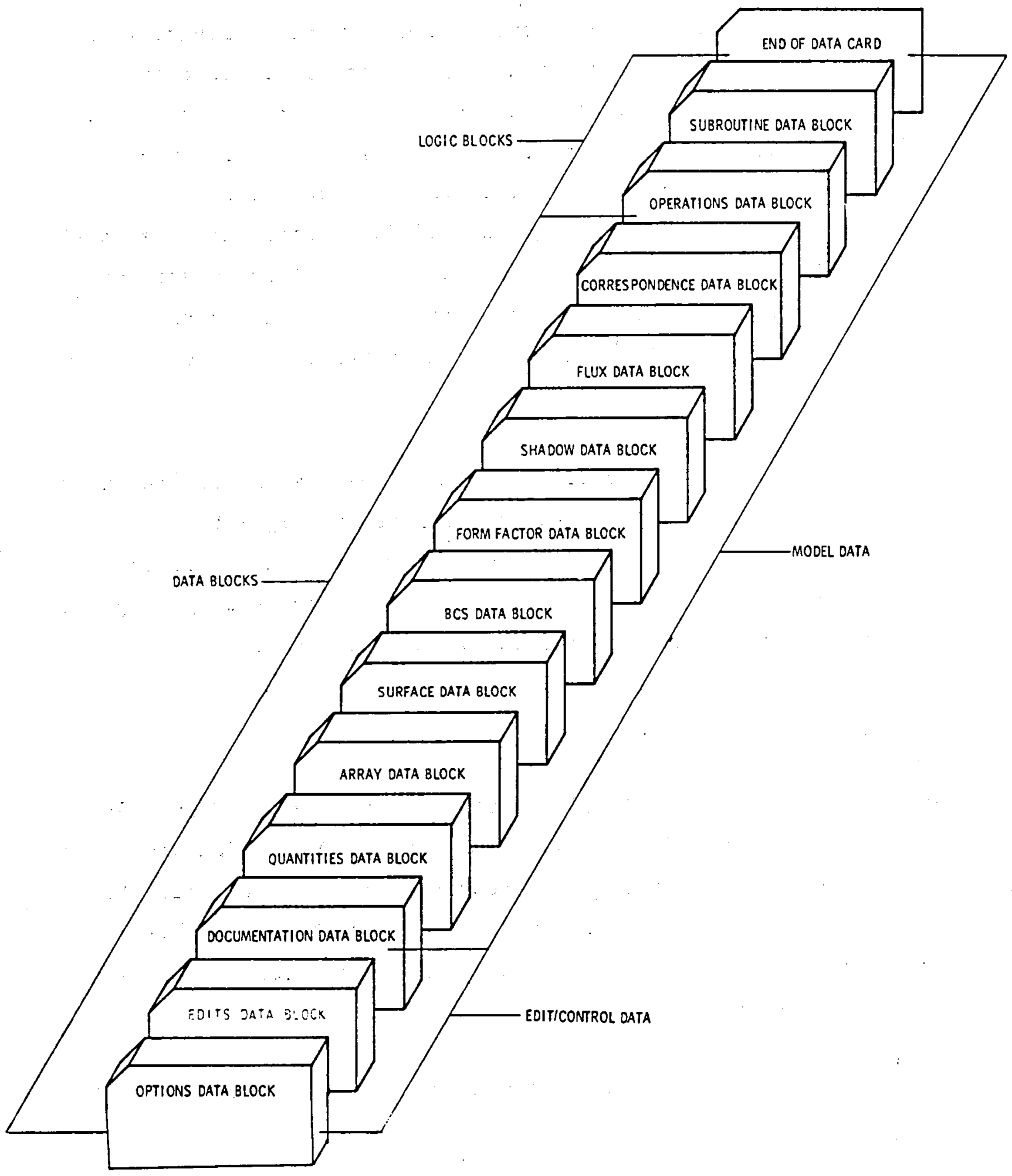


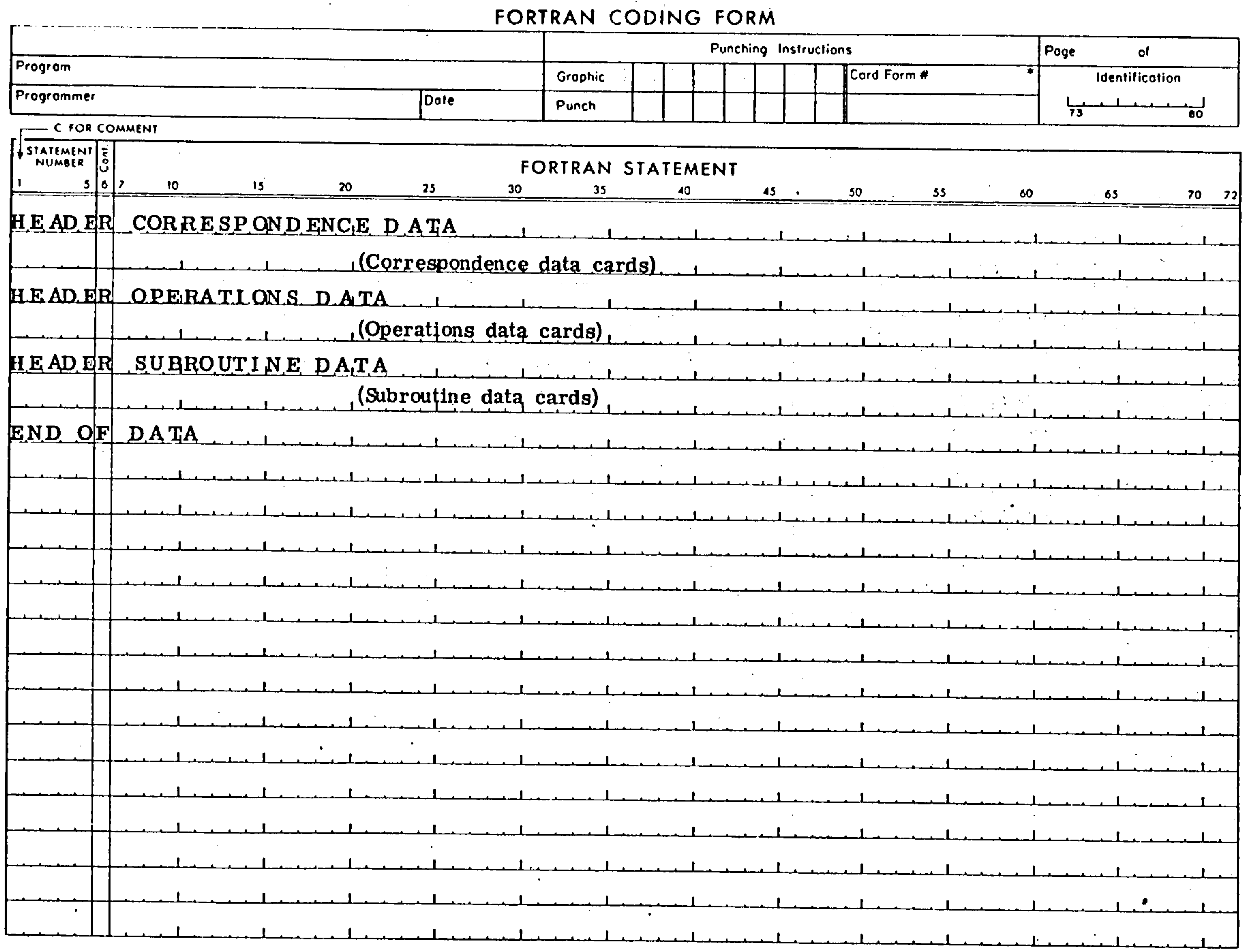
Figure 3-1 Input Deck Structure

	Punching Instructions		Page of
Program	Graphic	Card Form # *	Identification 73 80
Programmer / Date	Punch		

C FOR COMMENT

STATEMENT NUMBER 1 5	Cont. 6	FORTRAN STATEMENT 7 10 15 20 25 30 35 40 45 50 55 60 65 70 72
HEADE	R	OPTIONS DATA
		(Options data cards)
HEADE	R	EDIT DATA
		(Source edit cards)
HEADE	R	DOCUMENTATION DATA
		(Documentation data cards)
HEADE	R	QUANTITIES DATA
		(Quantities data cards)
HEADE	R	ARRAY DATA
		(Array data cards)
HEADE	R	SURFACE DATA
		(Surface data cards)
HEADE	R	BCS DATA
		(BCS data cards)
HEADE	R	FORM FACTOR DATA
		(Form factor data cards)
HEADE	R	SHADOW DATA
		(Shadow data cards)
HEADE	R	FLUX DATA

Figure 3-2 Header Card Formats



FORTRAN CODING FORM

Program		Punching Instructions	Page of
Programmer	Date	Graphic / Punch	Card Form #
			Identification 73 80

C FOR COMMENT

STATEMENT NUMBER 1 5 | Cont. 6 | FORTRAN STATEMENT 7 10 15 20 25 30 35 40 45 50 55 60 65 70 72

```
HEADER CORRESPONDENCE DATA
                  (Correspondence data cards)
HEADER OPERATIONS DATA
                  (Operations data cards)
HEADER SUBROUTINE DATA
                  (Subroutine data cards)
END OF DATA
```

Figure 3-2 (concl)

3.2 EDIT/CONTROL Data Blocks

3.2.1 Options Data Block

3.2.1.1 BASIC CONCEPTS

The Options data block provides the user with the following capabilities and operating options:

1) An entry point for his problem title and model name
2) Error plot option control
3) Source deck list/no list control
4) Source deck punch/no punch control
5) Go-No-Go option (No Go for edit only, no execution)
6) Print/no print for edit directives
7) Relabel edit directives
8) Print/no print of documentation data block
9) Read/write directions for all input and output tapes
10) Recomputation point in his operations data logic flow.

3.2.1.2 Options Data Block Variables

Table 3-I lists the options data block variables together with their options, default values, and descriptions.

3.2.1.3 Options Data Block Example

Figure 3-4 is an example of an options data block.

3.2.1.4 Automatic Node Plots Option

When surface data input rules are violated to the point where any surface is insufficiently defined to be usable, a fatal error flag is set and the run is terminated at the end of preprocessor execution. This is oftentimes undesirable because the primary objective of the first run on a newly defined model is usually to obtain plots of the problem geometry so that the user can visually verify his input. Time and effort can be saved if the surfaces that are usable to the node plotter are plotted in spite of the fatal errors.

The automatic node plot routines eliminate this problem. This capability functions as follows: when the word ERPLOT appears in the options data block and fatal errors result from the surface data, an operations data block is generated by the preprocessor. An example of such an operations data block is shown in Figure 3-3. This example is for a model that uses three block coordinate systems. The operations data block generated is then executed and the job terminates. Note that four automatically scaled plots are generated for each BCS. The views are from the x, y, and z axes, plus a 3-D. Also note that any operations data entered by the user is ignored.

```
HEADER OPERATIONS DATA

BUILD THING, ALLBLK
          CALL NDATAS(1,3HALL,0)
L         NPLOT
          CALL BUILDC(BCS2, 0)
L         NPLOT
          CALL BUILDC(BCS3, 0)
L         NPLOT
END OF DATA
```
Figure 3-3 Sample Operations Data Block Generated for Automatic Node Plots

Table 3-1 Options Data Input Detail

Options Data Input

CC1	CC7	OPTIONS	DEFAULT VALUE	DESCRIPTION
TITLE		PROBLEM TITLE IN CARD COLUMNS 7-72 INCLUSIVE	NONE	PROBLEM TITLE
	MODEL	= ANY 1-6 CHARACTER MODEL NAME	THING	PRIMARY MODEL NAME
		= NAME1 - NAME2	NONE	CHANGE MODEL NAME1 TO NAME2
	RSREC	POSITIVE INTEGER	0	DEFINES THE RSI TAPE RECORD NUMBER BEYOND WHICH RECOMPUTING SHOULD BEGIN IF THERE IS AN RSI TAPE READ ERROR (SEE SECTION 3.3.9.7: RESTART OPERATIONS
	INFO	A, N	A	A - PRINT THE INFORMATION DATA FILE N - DO NOT PRINT THE INFORMATION DATA FILE
	MAXFL	POSITIVE INTEGER	64000	DEFINES THE FIELD LENGTH (CORE) AVAILABLE IN THE COMPUTER (UNIVAC ONLY)
	LIST SOURCE	- ACTIVE	NONE	LIST ACTIVE CARDS IN MODEL
		- INACTIVE	NONE	LIST INACTIVE CARDS IN MODEL
		- ALL	NONE	LIST ALL ACTIVE AND INACTIVE CARDS
		(NOT INPUT)	NONE	NO LIST
	PUNCH SOURCE	- ACTIVE	NONE	PUNCH ACTIVE CARDS IN MODEL
		- INACTIVE	NONE	PUNCH INACTIVE CARDS IN MODEL
		- ALL	NONE	PUNCH ALL ACTIVE AND INACTIVE CARDS
		(NOT INPUT)	NONE	NO PUNCH
	NOGO	(INPUT)	NONE	EDIT, BUT DO NO PREPROCESS OR PROCESS
		(NOT INPUT)	NONE	EDIT, PREPROCESS, AND PROCESS
	NO PRINT	- EDIT	NONE	DO NOT PRINT EDIT DIRECTIVES
		(NOT INPUT)	NONE	PRINT EDIT DIRECTIVES

Table 3-1 (continued)

Options Data Input

CC1 CC7		OPTIONS	DEFAULT VALUE	DESCRIPTION
RELABEL		(INPUT) (NOT INPUT)	NONE NONE	CHANGE MODIFIER LABEL TO (AA) AND DELETE ALL INACTIVE CARDS
DMPDOC		(INPUT) (NOT INPUT)	NONE NONE	PRINT DOCUMENTATION DATA BLOCK NO PRINT
RSI	-	TXXXX[1]	SEE NOTE 2	PROGRAM WILL READ RSI TAPE TXXXX IF CONTROL CARD ASSIGNMENT IS MADE PRIOR TO EXECUTING PREPROCESSOR
RTI	-	TXXXX	NONE	PROGRAM WILL READ RTI TAPE TXXXX IF CONTROL CARD ASSIGNMENT IS MADE PRIOR TO EXECUTING PROCESSOR
RSO	-	TXXXX	SEE NOTE 3	PROGRAM WILL WRITE TO RSO TAPE TXXXX IF CONTROL CARD ASSIGNMENT IS MADE PRIOR TO EXECUTING PREPROCESSOR
RTO	-	TXXXX	NONE	PROGRAM WILL WRITE TO RTO TAPE TXXXX IF CONTROL CARD ASSIGNMENT IS MADE PRIOR TO EXECUTING PROCESSOR
BCDOU	-	TXXXX	BLANK	PROGRAM WILL WRITE TO BCDOU TAPE TXXXX WHETHER OR NOT BCDOU IS LISTED IN OPTIONS BLOCK IF CONTROL CARD ASSIGNMENT IS MADE PRIOR TO EXECUTING WRITE STATEMENTS CREATED BY THE USERS' HEADER OPERATIONS DATA BLOCK
CMERG		- TXXXX	BLANK	PROGRAM WILL READ CMERG TAPE TXXXX WHETHER OR NOT CMERG IS LISTED IN OPTIONS BLOCK IF CONTROL CARD ASSIGNMENT IS MADE PRIOR TO EXECUTING PREPROCESSOR AND CMERG EDIT DIRECTIVES ARE USED IN THE HEADER EDIT DATA BLOCK
EMERG		- TXXXX	BLANK	PROGRAM WILL READ EMERG TAPE TXXXX WHETHER OR NOT EMERG IS LISTED IN OPTIONS BLOCK IF CONTROL CARD ASSIGNMENT IS MADE PRIOR TO PREPROCESSOR AND EMERG EDIT DIRECTIVES ARE USED IN THE HEADER EDIT DATA BLOCK

Table 3-1 (continued)

Options Data Input CC1 CC7	OPTIONS	DEFAULT VALUE	DESCRIPTION
USER1	- TXXXX	BLANK	PROGRAM WILL WRITE TO USER1 TAPE TXXXX WHETHER OR NOT USER1 IS LISTED IN OPTIONS BLOCK IF CONTROL ASSIGNMENT IS MADE PRIOR TO EXECUTING WRITE STATEMENTS CREATED BY USERS' HEADER OPERATIONS DATA BLOCK
TRAJ	- TXXXX	NONE	PROGRAM WILL READ TRAJ TAPE TXXXX IF CONTROL CARD ASSIGNMENT IS MADE PRIOR TO EXECUTING READ STATEMENT CREATED BY USERS' HEADER OPERATIONS DATA BLOCK
TAPENAME*	- TXXXX	NONE	PROGRAM WILL READ OR WRITE TAPE TXXXX DEPENDENT UPON FUTURE AND AUXILIARY APPLICATIONS
NNMIN	INTEGER NO.	SEE NOTE 4	CHANGES NODE ARRAY DIMENSION

Table 3-1 (continued)

*See Appendix G for additional TAPE/FILE INFORMATION

NOTES:

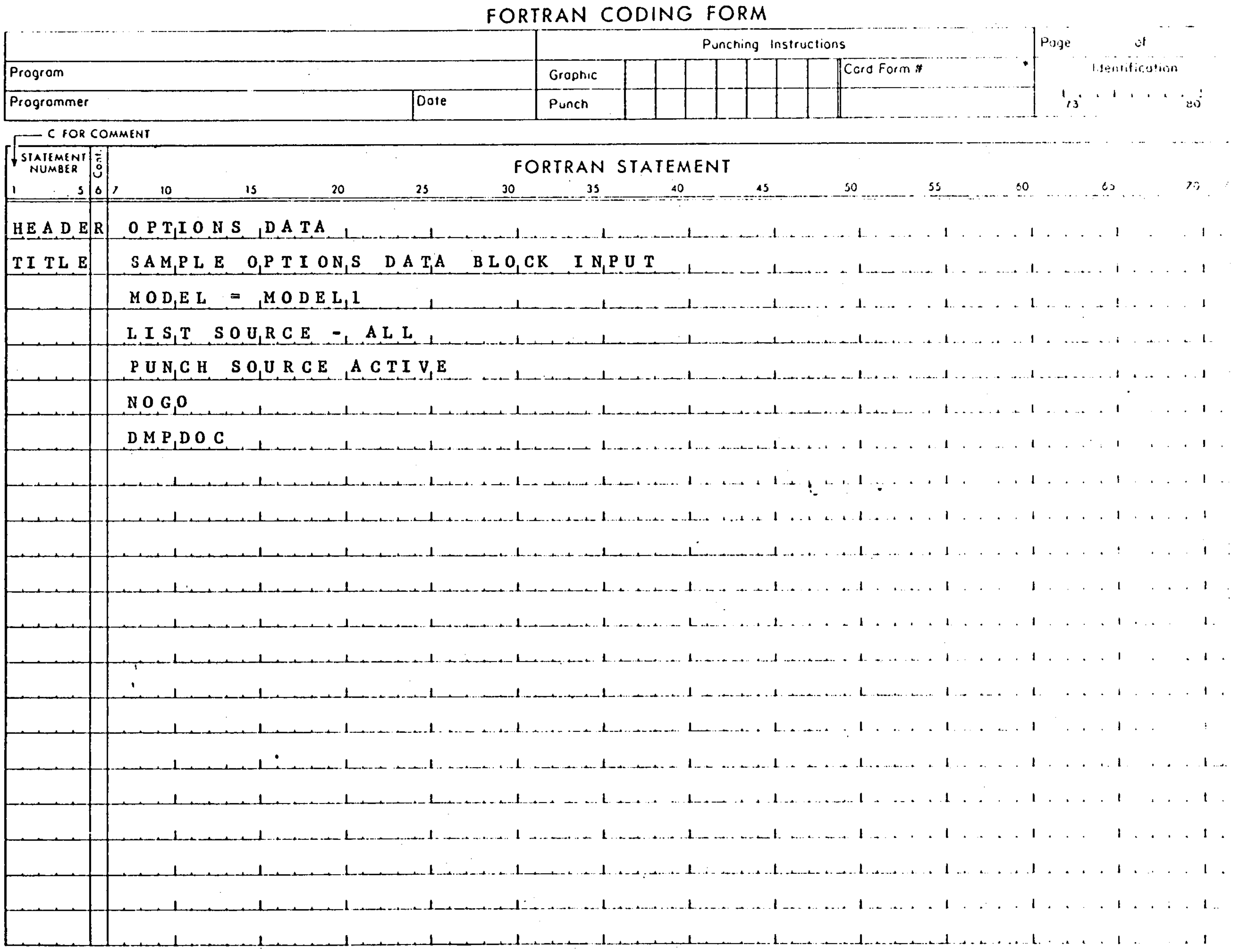
1 UTILIZING THE RSI TAPE AS AN EXAMPLE THE ALLOWABLE FORMS FOR ALL TAPES ARE:

COL 7	
RSI	NO LABEL
RSI - TXXXX	TXXXX IS ANY 6-CHARACTER USER LABEL (e.g., TAPE NUMBER). IT WILL BE PRINTED UNDER MODEL HISTORY FOR USER ONLY DOCUMENTATION PURPOSES
RSI - TXXXX	SAME AS ABOVE

2 RSI MUST ALWAYS BE LISTED IN OPTIONS DATA BLOCK IF IT IS TO BE USED, EXCEPT FOR A RESTART CASE IN WHICH NO OPTION DATA BLOCK IS INCLUDED IN DATA DECK BECAUSE THERE WERE NO OTHER REQUIREMENTS FOR IT AND THE ASSIGNED TAPE HAS THE MODEL DATA ON IT.

3 RSO MUST ALWAYS BE LISTED IN OPTIONS DATA BLOCK EXCEPT WHEN AN RSI TAPE HAS BEEN ASSUMED (SEE NOTE 2).

4 IF INPUT, THE NODE ARRAY WILL BE DIMENSIONED BY EITHER NNMIN OR THE TOTAL NUMBER OF NODES DEFINED IN THE SURFACE DATA, WHICHEVER IS LARGER. USED IN CONJUNCTION WITH FFNAC, (SEE SECTION 3.3.5: FORM FACTOR DATA) THE USER MAY UTILIZE FORM FACTORS ON A RSI TAPE CREATED WITH A LARGER MODEL THAN ON THE CURRENT RUN.



FORTRAN CODING FORM

Program	Punching Instructions	Graphic	Card Form #
Programmer	Date	Punch	
Page of	Identification 73 80		

C FOR COMMENT

STATEMENT NUMBER 1 5 | Cont. 6 | FORTRAN STATEMENT 7 10 15 20 25 30 35 40 45 50 55 60 65 70

```
HEADER R OPTIONS DATA
TITLE    SAMPLE OPTIONS DATA BLOCK INPUT
         MODEL = MODEL1
         LIST SOURCE - ALL
         PUNCH SOURCE ACTIVE
         NOGO
         DMPDOC
```

Figure 3-4 Options Data Block Example

3.2.2 EDIT DATA BLOCK

3.2.2.1 Basic Concepts

Figure 3-5 is a block diagram of the edit portion of the preprocessor. Edit logic flow is shown, together with its relationship with the remainder of the program. The edit portion consists of the SOURCE EDITOR which deletes, inserts, merges, and yanks records and blocks of records to form a primary model or input deck.

The source editor's function is to generate a complete input deck and pass it on to the data and logic preprocessors on the DATAI unit. If desired, the user may obtain a permanent copy of the DATAI model on an RSO tape. Source editing can be done in several ways. The simplest mode is to read the user's cards and pass them on as a complete deck on the DATAI unit. Alternatively, the user supplied CMERG tape can be passed along directly if the CMERG tape is nothing more than the user's cards previously transferred to tape. The CMERG unit also provides an interface between any user supplied input processing routine(s) and the program. In the general edit case, the source-edit cards are used to generate an executable model from input data found in card form and on the RSI, CMERG, or EMERG units.

The product of the TRASYS editor is a single TRASYS input deck. This is either data selected from an RSI tape, or data in the card input stream. Edits can be in the form of record deletions and record insertions. Records for insertions can be obtained from these sources: (1) the card input unit (edit data block); (2) the card merge unit - CMERG; and (3) the edit merge unit - EMERG.

The CMERG unit can be single or multifile tape, disk, or drum. The data contained on the CMERG unit must be in BCD mode, TRASYS card image form. This type of data is usually generated by: (1) card-to-tape by an off-line computer; (2) TRASYS processor output to USER1; or (3) TRASYS input data conversion programs.

The EMERG unit can be a single or multifile tape, disk, or drum. The data contained on the EMERG unit must be in the RSO tape format. The EMERG file is another RSO output tape from a previous run.

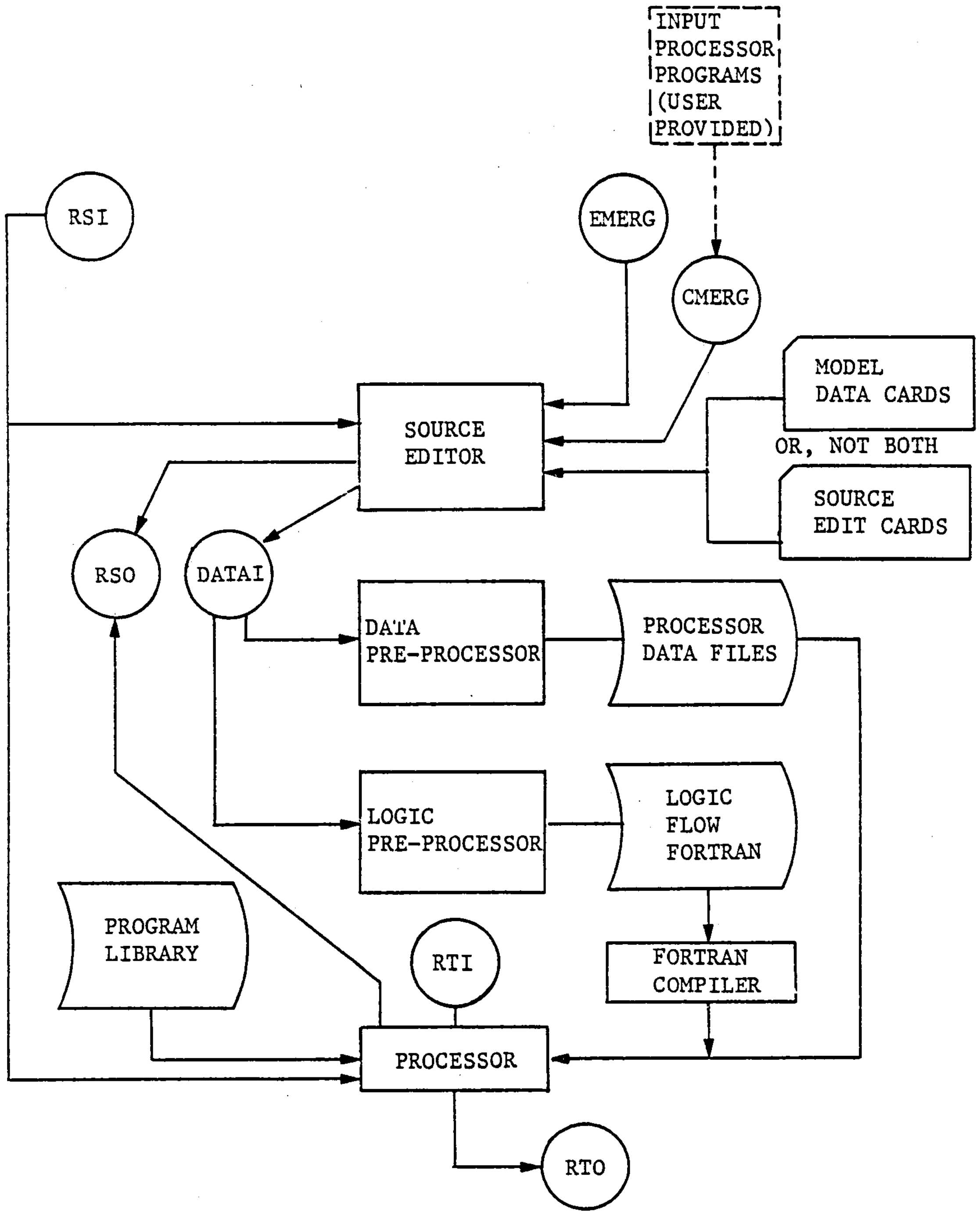


Figure 3-5 Edit Segment Logic Flow

Any record or group of records contained in any file on the CMERG or EMERG units can be merged into the primary input deck. Cards to be merged into the primary input deck can be in random order on the CMERG and EMERG units. Note that inserting data from the cards in the edit data block is possible only when the primary input comes from an RSI unit.

Besides deleting, inserting, and merging cards into the input deck, the source editor has the capability of yanking modifications. Each time a deck is edited, the inserted and deleted cards are tagged with a modifier label and deleted cards are maintained on an inactive status. A "yank" provides a simple means of returning a model to the condition it was before a given modification. For instance, yanking modification "A" will insert all cards deleted by "A" and delete all cards inserted by "A." More than one label can be yanked in one run. Cards deleted by a yank are not maintained on inactive status.

3.2.2.2 Edit Data Block

After an RSO tape has been generated, the user will have a source listing with edit numbers and edit labels. This tape may then become an RSI or an EMERG tape that can be edited according to the source listing using edit directives in the edit data block. Table 3-II presents details of the edit data block directives together with format information. Figure 3-6(a) is an example of an edit data block.

NOTE:

Rather than use the input deck on the RSI tape as is; or with modification via the Edit Data Block the user may input a complete Input Deck in card image form which will be used in place at the input deck on the RSI. This could be an alternate way to make changes to the input.

3.2.2.3 <u>Edit Operations</u>

Edit operations are illustrated by the following examples, see Fig. 3-6(b) and 3-6(c):

1) Input on cards, edits directly from cards and from EMERG and CMERG tapes.
2) Input on an RSI tape, edits from cards, EMERG and CMERG tapes.

Table 3-II Edit Data Block Input Details

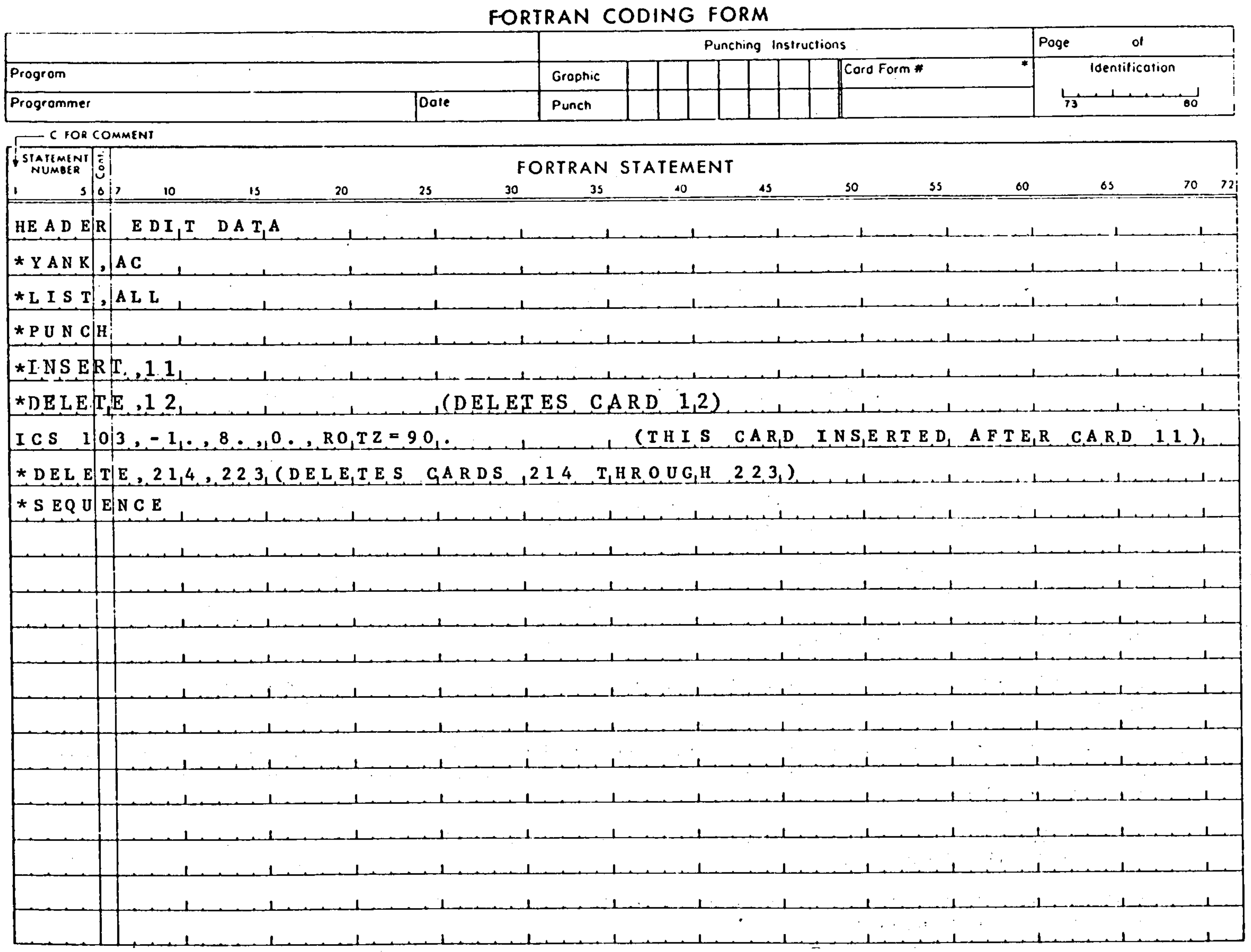
EDIT DATA INPUT (SEE NOTES AT END OF TABLE)	DESCRIPTION
*I, N1 OR *INSERT, N1	THE CARDS FOLLOWING THIS CARD WILL BE INSERTED AFTER EDIT NUMBER -N1-
*D, N1 OR *DELETE, N1	DELETE PRIMARY DECK CARD WITH EDIT NUMBER -N1-
*D, N1, N2 OR DELETE, N1, N2	DELETE PRIMARY DECK CARDS WITH EDIT NUMBERS -N1- THROUGH -N2-
*C, F1 OR *CMERG, F1	INSERT THE ENTIRE CARD FILE -F1- FROM UNIT -CMERG- AT THIS POINT IN THE INPUT DECK
*C, F1, N1, N2 OR *CMERG, F1, N1, N2	INSERT LINES N1 THROUGH N2 FROM CMERG FILE F1 AT THIS POINT IN THE INPUT DECK
*C, F1, N1, ALL OR *CMERG, F1 , N1, ALL	INSERT ALL LINES FROM N1 TO END OF CMERG FILE F1 AT THIS POINT IN THE INPUT DECK
*E, NAME OR *EMERG, NAME	INSERT THE ENTIRE EDIT DECK, - NAME- FROM UNIT -EMERG- AT THIS POINT IN THE INPUT DECK
*EMERG, NAME N1, N2	MERGE LINE N1 THROUGH LINE N2 OF EDIT DECK -NAME- AT THIS POINT IN THE INPUT DECK
*P OR *PUNCH	PUNCH ACTIVE CARDS FROM THE UPDATED DECK (SEE NOTE 3)
*P, INACTIVE OR *PUNCH, INACTIVE	PUNCH INACTIVE CARDS FROM THE UPDATED DECK (SEE NOTE 3)
*P, ALL OR PUNCH, ALL	PUNCH ALL ACTIVE AND INACTIVE CARDS FROM THE UPDATED DECK (SEE NOTE 3)
*L OR *LIST	LIST ACTIVE CARDS FROM THE UPDATED DECK

Table 3-II (concl)

EDIT DATA INPUT (SEE NOTES AT END OF TABLE)	Description
*L, INACTIVE OR *LIST, INACTIVE	LIST INACTIVE CARDS FROM THE UPDATED DECK
*L, ALL OR *LIST, ALL	LIST ALL ACTIVE AND INACTIVE CARDS FROM THE UPDATED DECK
*S OR *SEQUENCE	NUMERICALLY SEQUENCE PUNCHED CARDS. THIS CARD MAY APPEAR ANYWHERE IN THE EDIT DATA BLOCK
*Y, AB OR *YANK, AB	DELETE ALL CARDS FROM PRIMARY DECK THAT WERE INSERTED BY EDIT DIRECTIVE -AB- (SEE NOTE 4)
*Y, AB, AD OR *YANK, AB, AD	DELETE ALL CARDS FROM PRIMARY DECK THAT WERE INSERTED BY SOURCE EDIT DIRECTIVES -AB- THROUGH -AD- (SEE NOTE 4)

NOTES:

1. AN ASTERISK (*) IN CARD COLUMN 1 DESIGNATES AN EDIT CONTROL CARD.
2. EDIT CONTROL CARDS ARE FREE FIELD FORMATED IN CARD COLUMNS 2 THROUGH 72 WITH BLANK COLUMNS AND COLUMNS 73 THROUGH 80 IGNORED.
3. THIS CARD MAY BE USED IN PLACE OF THE -PUNCH SOURCE-OPTION IN THE OPTIONS DATA BLOCK. IF BOTH CARDS ARE INPUT, THIS CARD OVERRIDES THE OPTIONS DATA INPUT. THIS CARD MAY APPEAR ANYWHERE IN THE EDIT DATA BLOCK.
4. ONLY ONE YANK DIRECTIVE CAN APPEAR IN THE EDIT DATA BLOCK, AND THAT CARD MUST IMMEDIATELY FOLLOW THE -HEADER EDIT DATA- CARD.
5. A PRIMARY DECK IS THE INPUT DECK (ON A RSI TAPE) TO WHICH ALL INSERT & DELETE LINE NUMBERS REFER (*I and *D CARDS ONLY)
6. *C and *E CARDS ARE NOT RESTRICTED TO THE EDIT DATA BLOCK. THEY MAY ALSO APPEAR AT ANY POINT IN THE INPUT DECK WHERE THE MATERIAL THEY INSERT IS APPROPRIATE.



FORTRAN CODING FORM

Program	Punching Instructions: Graphic / Punch	Card Form #
Programmer	Date	Page of / Identification 73 80

C FOR COMMENT — STATEMENT NUMBER — CONT. — FORTRAN STATEMENT

```
HEADER EDIT DATA
*YANK,AC
*LIST,ALL
*PUNCH
*INSERT,11
*DELETE,12                    (DELETES CARD 12)
ICS 103,-1.,8.,0.,ROTZ=90.           (THIS CARD INSERTED AFTER CARD 11)
*DELETE,214,223(DELETES CARDS 214 THROUGH 223)
*SEQUENCE
```

Figure 3-6 (a) Edit Data Block Example

```
HEADER OPTIONS DATA
TITLE EDITOR CHECK OUT-EXAMPLE 1
      MODEL - DATA1 $ DATA1 = MODEL NAME THAT WILL APPEAR ON RSO TAPE
      EMERG - TXXXX
      CMERG - TXXXX
      RSO - TXXXX
HEADER SURFACE DATA
                              (SURFACE DATA CARDS)
C     INSERT CARDS 199 THRU 998 FROM EMERGE MODEL SURFD1 INTO SURFACE
C     DATA BLOCK
*EMERG,SURFD1,199,998
                        (ADDITIONAL SURFACE DATA CARDS)
HEADER FORM FACTOR DATA
C     INSERT FILE 4 FROM CMERG TAPE
*CMERG,4
HEADER FLUX DATA
                              (FLUX DATA CARDS)
HEADER OPERATIONS DATA
C     INSERT PART OF FILE 2 FROM CMERG TAPE
*C,2,1,39
END OF DATA
```

(b) Edit Operations Example - Model on Cards

Figure 3-6 (con't)

```
HEADER OPTIONS DATA
TITLE EDITOR CHECK OUT EXAMPLE 2
      MODEL = DOCK1 - DOCK2  $ DOCK1 = RSI MODEL NAME
C                             DOCK2 = RSO MODEL NAME
      EMERG - TXXXX
      CMERG - TXXXX
      RSI   - TXXXX
      RSO   - TXXXX

HEADER EDIT DATA
*D,2,6
                (CARDS TO BE INSERTED IN LIEU OF CARDS 2 THRU 6)
*I,11
                      (CARDS TO BE INSERTED AFTER CARD 11)
*I,340   $CARD FOLLOWING INSERTS CARD 312 THRU 450 FROM MODEL DOCKA
C     ON EMERG AFTER CARD 340
*E,DOCKA,312,450
```

(c) Edit Operations Example - Model on RSI Tape

Figure 3-6 (concl)

3.3 DATA BLOCKS

3.3.1 Documentation Data

Experience has shown that thermal mathematical models may have an extended useful life, especially if tape storage with convenient editing capability is available. The usual environment of sketchy and rarely updated documentation results in a waste of resources as these long-lived models are passed from analyst to analyst through project personnel changes.

As an aid in alleviating this problem, TRASYS users may document their efforts in an easily edited and easily accessed form in the documentation data block.

The documentation data block has the following format:

```
CC1    CC7                                    CC72
  HEADER DOCUMENTATION DATA
         Documentation Data Card 1
         Documentation Data Card 2
              .        .      . .
              .        .      . .
              .        .      . .
         Documentation Data Card N
```

Documentation data cards have a data field from CC7 through 72 inclusive and have no restriction on their alphanumeric content. There is no practical limit on the number of documentation cards allowed.

The control field of documentation data cards is used for carriage control. The integer N appearing anywhere in CC1 through 6 of a documentation data card results in N lines being skipped before printout of that card. If less than N lines are available on a page, the card will begin a new page. If a new page is desired, the letter P is placed in CC1. A documentation data printout is available at the user's option. The flag DMPDOC appearing in the options block results in a printout of the documentation data prior to any preprocessor or processor operations.

3.3.2 Quantities and Array Data Blocks

3.3.2.1 Basic Concepts

The quantities and array data blocks have the primary function of providing the user with a convenient input point for any single variable and array data he plans to use during his execution. User constants are defined in the quantities data block and arrays in the array data block. A user variable or array may take any name not appearing in the program reserve name list or program control constant list (see Appendix A). Real, integer, or Hollerith data may be entered. Mode agreement is required for real and integer data names. Hollerith strings are limited to 6 characters in the quantities data block.

The pre-processor provides default values for all program control constants, so no control constant input is required in the quantities data. Further, all control constants can be redefined in the operations block. Thus, defining control constants at the quantities data block merely has the effect of redefining the default values. In general, this practice is not recommended because it is easy for the user to forget that he has a non-standard default value in his quantities data block when he is defining control constants in the operations data block.

3.3.2.2 <u>Rules for Input</u>

All quantities and array data are entered in the data field (Columns 7 through 72) of the cards following the appropriate header card. Specific rules for input are:

1) The general quantity data formats are:

   NAME = DV, (integer)

   ANAME = DV, (real)

   NAME (or ANAME) = DV where DV is a 1 to six character string. (Left justified, blank filled)

2) The general array data formats are:

   NAME = DV1, DV2 - - - DVN (integer)

   ANAME = DV1, DV2 - - - DVN (real)

   NAME (or ANAME) = *I AM A HOLLERITH ARRAY* (Hollerith)

3) Array data values may be operated as follows:

   NAME = DV1, REPEAT, DV2, N, DVN+2 (Repeats DV2 N times, continues with DVN+2)

   Real array repeat format is identical.

4) Variable names must consist of 3 to 6 alphanumeric characters, with an alphabetic character heading.

5) Any number of quantities or array data values and names may be entered in Card Columns 7 through 72 inclusive. Input may continue to following cards with no data in the control field, provided the fields between commas are complete on each card.

6) Commas are assumed at the end of each card. Commas may be entered at the beginning or end of cards at the user's option. The read routine ignores these.

7) May end but not begin with equal signs.

8) Empty fields (consecutive commas) are illegal.

9) Mode agreement between names and data values must be maintained.

3.3.2.3 Quantities and Array Data Accessing

Quantities and array data are placed in common and are thus accessible from any user or program-called execution routines. The following are the rules for accessing this data:

1) Quantities data are accessed by name only.
   For example, with

       ANAM = DV,

   in the quantities block, the statement

       VAL = ANAM

   in any execution routine results in DV being stored under the name VAL. Mode must be preserved according to the rules of FORTRAN.

2) Array data is accessed as illustrated by the following examples. Each example presumes that the array:

       ANAM = DV1, DV2 - - - DVN

   appears in the array data block.

   A. SUBROUTINE CALLS
      The statement:

      ```
      CALL SUBX (ARG1, ARG2, ANAM, ARG4 ---)
      ```

      in any execution routine will pass the entire ANAM array to subroutine SUBX.

B. INTEGER COUNT

The integer count for any array is accessed through the following function call:

```
IC = IACT (ANAM)
```

C. INDIVIDUAL DATA VALUES

Individual data values are accessed under the usual rules of FORTRAN. The statement:

```
VAL = ANAM(6)
```

results in DV6 being stored under the name VAL.

3) The array data block serves as the only means of reserving space for the operations data block. The following example illustrates this with:

```
ANAMA = DV1, DV2, - - - DVN,
XARRAY = REPEAT, O., N
```

in the array data block, the statements:

```
    DO 1 I = 1, IC
1   XARRAY (I) = ANAMA (I)
```

in the operations data block will locate the ANAMA array in the first IC words of XARRAY.

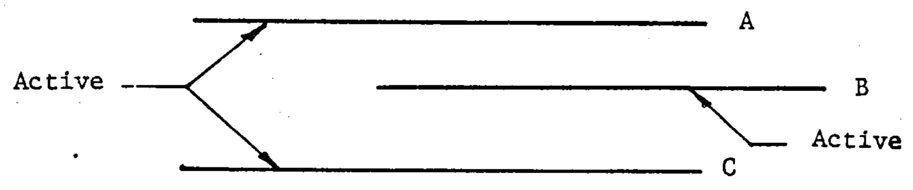
3.3.3 SURFACE DATA

3.3.3.1 BASIC CONCEPTS

In its present state of development, TRASYS allows the user's geometric configuration to be made up of the following geometric shapes, or portions thereof:

(1) Rectangles
(2) Discs
(3) Polygons
(4) Right Circular Cylinders
(5) Cones
(6) Spheres
(7) Paraboloids
(8) Rectangular Parallelepipeds with 5 or 6 faces

The surface areas of these shapes are what TRASYS is concerned with. The volume within a sphere, for instance, has no bearing on the thermal radiation problem. Either or both sides of any surface can be defined as "active." Also, any surface can be defined as a "shadower" or "non shadower" depending on whether or not it is desired that it be considered in shadowing (blockage) calculations.



The active side concept is illustrated in the sketch. If form factors were computed in this geometry, F_{AC} and F_{BC} would exist because the active sides involved are in view of each other. No F_{AB} form factor will be computed, because surface A cannot see the active side of surface B.

Again referring to the sketch, if surface B is defined as a shadower in form factor computations, it will affect the calculation of F_{AC}, reducing it accordingly. If not entered as a shadower, it would be totally "invisible" from surface A. Active side definition has no bearing on a surface's effect as a shadower. One further condition must exist for surface B to effect the value of F_{AC}, that is surfaces A and C must be flagged as "can be shaded" surfaces in form factor calculations.

A similar logic is used in computations of direct irradiation. Surfaces defined as shadowers may affect the direct irradiation computed depending on direction to the incident flux source.

Since all surfaces in nature can shade and be shaded, it may seem questionable to leave the shadowing definition up to the user. The reason for this is that significant amounts of computation time can be saved by flagging out surfaces that cannot enter into shadowing. It is also advantageous from the standpoint of computation time to minimize the number of surfaces to define a given configuration and nodal breakdown.

Any surface may be subdivided into nodal surfaces of equal or unequal size. In general, the nodal surfaces chosen should correspond with the isothermal nodes that appear in the user's thermal analyzer model. For various reasons, this may not be possible, so a convenient means for combining nodes is provided.

3.3.3.2 Coordinate System Definition

The surface data is associated with four different, right-handed cartesian coordinate systems:

Surface coordinate system
Intermediate coordinate system
Block coordinate system
Central coordinate system

Their definitions are as follows:

Central Coordinate System (CCS) - This is the single coordinate system to which all vehicle surfaces must be related. This coordinate system is also used to orient the spacecraft relative to the sun, planet, or a star. This coordinate system is analogous to the body coordinate system used in trajectory tapes.

Block Coordinate System (BCS) - Any "block" of surfaces that will be moved relative to other surfaces during execution must be related to a named block coordinate system. Similarly, if it is desired to activate and/or deactivate a block of surfaces during the course of the problem, these surfaces are related to a separate block coordinate system.

Intermediate Coordinate System (ICS) - An intermediate coordinate system is used when it is convenient to relate a group of surfaces to a coordinate system distinct from any BCS or the CCS.

Surface Coordinate System (SCS) - Each surface is related to its own SCS in a manner that provides a convenient means of input. The surface must then be related to the CCS by defining the rotations and translations necessary to make the SCS and CCS coincide.

3.3.3.3 Coordinate System Hierarchy

Each surface and hence nodal surface defined in the surface data block may undergo three transformations, as follows, before processing begins:

$$\text{SCS} \rightarrow \text{ICS} \rightarrow \text{BCS} \rightarrow \text{CCS}$$

where, for example the symbology SCS $\rightarrow$ ICS indicates a transform from SCS-defined 3-space to ICS defined 3-space. These transforms must be performed because all processing is done assuming surface definition in CCS-defined 3-space.

Depending on the complexity of each particular surface definition problem, the user may or may not concern himself with all the transforms. In the simplest case, the user defines a surface in terms of x, y, z coordinates in CCS 3-space. The program automatically generates an SCS for each surface and also generates the transforms necessary to describe the surface in CCS 3-space. In the most complex case, the user defines his surface in SCS 3-space, defines six rotation and translation variables for the SCS $\rightarrow$ ICS transform, defines six rotation and translation variables for the ICS $\rightarrow$ BCS transform, and finally six more variables for the BCS $\rightarrow$ CCS transform. For cases of intermediate complexity, for instance when an ICS is not needed, the ICS $\rightarrow$ BCS transform variables will default to zero and the user's SCS $\rightarrow$ ICS transform variables will, in reality define an SCS $\rightarrow$ BCS transform. Further, if neither an ICS or BCS is required, the SCS $\rightarrow$ ICS and ICS $\rightarrow$ BCS transforms will default to zero, and the user's SCS $\rightarrow$ ICS transform definition will, in reality, define an SCS $\rightarrow$ CCS transform.

3.3.3.4 Surface Data Input Philosophy

The user is provided with two distinct methods of defining his surface. He may define a surface relative to an SCS, then relate it to the remainder of the surfaces by defining the SCS-CCS translations and rotations; or he may locate the surface directly in relation to the CCS by entering the x, y, z coordinates of up to 19 points on his surface (point method). His choice of these methods depends on the particular surface being considered and its relationship to the remainder of the vehicle. In general, it is easy to define a surface relative to an SCS. This requires five numbers. It may or may not be convenient to determine the translation and rotation data (up to 6 numbers) needed for the SCS   ICS, SCS   BCS, or SCS   CCS relationship. When the computation of these rotation and translation parameters is laborious, the point method is usually a better choice. Except for polygons, this requires up to 5 point definitions per surface (15 numbers), but generally these points are on the surface involved and may be easily scaled from an engineering drawing.

3.3.3.5 Surface Data Variables

The variable names devoted to surface data definition are defined in Table 3-III. Also tabulated are their default values, and the allowable range of each variable, where applicable. Figure 3-7 illustrates the relationship of the dimension surface data variables to each geometric figure. Both the surface coordinate system methods and point methods of input are shown. A careful study of Table 3-III and Figure 3-7 will pay dividends to the new TRASYS user.

Table 3-III Surface Data Input Detail

VARIABLE NAME	RANGE OR OPTIONS	DEFAULT VALUE	DESCRIPTION
GENERAL DATA:			(REF. Fig 3-10)
SURFN	1-99999	NONE	a. INTEGER ARRAY OF NODE NUMBERS ASSOCIATED WITH SURFACE b. INITIAL NODE NO. ON SURFACE
NNX	1-999	1	NO. OF NODES IN X DIRECTION
UNNX	$0. <$ UNNX $\leq$ (XMAX-XMIN)	NONE	X-DIMENSION ARRAY FOR UNEQUAL NODE BOUNDARIES
NNY	1-999	1	SAFE AS NNX EXCEPT Y-DIRECTION
UNNY	$0. <$ UNNY $\leq$ (YMAX-YMIN)	NONE	SAME AS UNNX EXCEPT Y-DIRECTION
NNZ	1-999	1	SAME AS NNX EXCEPT Z-DIRECTION
UNNZ	$0. <$ UNNZ $\leq$ (ZMAX-ZMIN)	NONE	SAME AS UNNX EXCEPT Z-DIRECTION
NNAX	1-999	1	SAME AS NNX EXCEPT AX-DIRECTION
UNNAX	$0. <$ UNNAX $\leq$ (AXMAX-AXMIN)	NONE	SAME AS UNNX EXCEPT AX-DIRECTION
NNR	1-999	1	SAME AS NNX EXCEPT R-DIRECTION
UNNR	$0. <$ UNNR $\leq$ (RMAX-RMIN)	NONE	SAME AS UNNX EXCEPT R-DIRECTION
TYPE	RECT, TRAP DISK, CYL CONE, SPHER PARAB, BOX5 BOX6, POLY	NONE	SURFACE TYPE
IDUPSF	1-99999	NONE	NUMBER OF PREVIOUSLY INPUT SURFACE TO BE DUPLICATED
IMAGSF	1-99999	NONE	NUMBER OF PREVIOUSLY INPUT SURFACE TO BE IMAGED

Table 3-III (continued)

VARIABLE NAME	RANGE OR OPTIONS	DEFAULT VALUE	DESCRIPTION
GENERAL DATA:			
ACTIVE	TOP, BOTTOM BOTH (PLANAR SURFACES)		ACTIVE SIDE DEFINITION SCS METHOD: TOP = +Z FACE OF PLANAR SURFACES
	IN, OUT, BOTH (SURFACES OF REV- OLUTION, AND BOXES	NONE	POINT METHOD: REF. FIGURE 3-7
BCSN	1-6 CHARACTER NAME	ALLBLK	BLOCK COORDINATE SYSTEM NAME - IDENTIFIES SURFACE WITH A BCS
COM	N/A	BLANKS	30 CHARACTERS OF COMMENT TO DESCRIBE SURFACE
SHADE	FF, DI, BOTH, NO, ONLY	BOTH *FF	SURFACE CAN SHADE FLAG FF: SHADES IN FORM FACTOR CALCULATIONS ONLY DI: SHADES IN DIRECT IRRADIATION CALCULATIONS ONLY BOTH: SHADES IN BOTH FF AND DI CALCULATIONS NO: SURFACE CANNOT SHADE ONLY: SURFACE IS A SHADOWER ONLY (FF AND DI)
BSHADE	FF, DI, BOTH, NO	BOTH	SURFACE CAN BE SHADED FLAG FF: CAN BE SHADED IN FF CALCULATIONS ONLY DI: CAN BE SHADED IN DI CALCULA- TIONS ONLY BOTH: CAN BE SHADED IN BOTH FF AND DI CALCULATIONS NO: SURFACE CANNOT BE SHADED

Table 3-III (continued)

VARIABLE NAME	RANGE OR OPTIONS	DEFAULT VALUE	DESCRIPTION
DIMENSIONS DATA:			
AXMIN	$-270. \leq AXMIN \leq 450.$	NONE	MIN. X-ANGLE SURFACES OF REVOLUTION
AXMAX	$-270. \leq AXMAX \leq 450.$	NONE	MAX. X-ANGLE SURFACES OF REVOLUTION
ZMIN	N/A	NONE	MIN. DIMENSION - Z DIRECTION
ZMAX	N/A	NONE	MAX. DIMENSION - Z DIRECTION
RMIN	N/A	NONE	MINIMUM RADIUS - DISK SECTION
RMAX	N/A	NONE	MAXIMUM RADIUS - DISK SECTION
R	N/A	NONE	RADIAL DIMENSION
Z	N/A	NONE	Z DIMENSION
P1, P2 - ETC.	P1 - P15	NONE	CARTESIAN POINT INPUT (GENERAL FORM: PN = XN, YN, ZN)
PROPERTIES DATA:			
ALPHA	$0. \leq ALPHA \leq 1.0$	NONE	ABSORPTIVITY - SOLAR
EMISS	$0. \leq EMISS \leq 1.0$	NONE	EMISSIVITY - IR
TRANI	$-1.0 \leq TRANI \leq 1.0$	0.0	TRANSMISSIVITY - IR
TRANS	$-1.0 \leq TRANS \leq 1.0$	0.0	TRANSMISSIVITY - SOLAR
SPRI	$0. \leq SPRI \leq 1.0$	0.0	SPECULAR REFLECTIVITY - IR
SPRS	$0. \leq SPRS \leq 1.0$	0.0	SPECULAR REFLECTIVITY - SOLAR

Table 3-III (continued)

VARIABLE NAME	RANGE OR OPTIONS	DEFAULT VALUE	DESCRIPTION
POSITION DATA: (NOT APPLICABLE TO POINT METHOD INPUT)			
TX	N/A	0.0	TRANSLATION DISTANCE FROM ORIGIN OF CCS, BCS OR ICS TO ORIGIN OF SCS, MEASURED ALONG CCS, BCS, OR ICS X-AXIS.
TY	N/A	0.0	SAME AS TX, EXCEPT ALONG Y-AXIS
TZ	N/A	0.0	SAME AS TX, EXCEPT ALONG Z-AXIS
ROTX	$-360. < \text{ROTX} \leq 360.$	0.0	ROTATION ANGLE TO ROTATE CCS, BCS OR ICS INTO SCS; ROTATES ABOUT CCS, BCS OR ICS X-AXIS, Y TOWARD Z POSITIVE
ROTY	$-360. < \text{ROTY} \leq 360.$	0.0	SAME AS ROTX, EXCEPT ROTATES ABOUT Y, Z TOWARD X POSITIVE
ROTZ	$-360. < \text{ROTZ} \leq 360.$	0.0	SAME AS ROTX, EXCEPT ROTATES ABOUT Z, X TOWARD Y POSITIVE
ICS DEFINITION (I CARD) DATA: (MUST PRECEDE ALL S-CARDS)			
ICSN	1-99999	NONE	INTERMEDIATE COORDINATE SYSTEM NUMBER
TX	N/A	0.0	TRANSLATION DISTANCE FROM ORIGIN OF CCS OR BCS TO ORIGIN OF ICS, MEASURED ALONG CCS OR BCS X-AXIS
TY	N/A	0.0	SAME AS TX, EXCEPT ALONG Y-AXIS
TZ	N/A	0.0	SAME AS TX, EXCEPT ALONG Z-AXIS

Table 3-III (concluded)

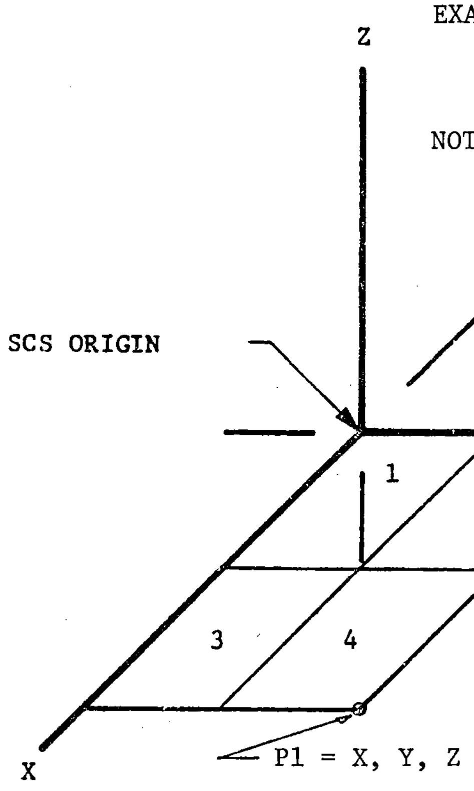
VARIABLE NAME	RANGE OR OPTIONS	DEFAULT NAME	DESCRIPTION
ICS DEFINITION (I CARD) DATA:			
ROTX	$-360. < ROTX \leq 360.$	0.0	ROTATION ANGLE TO ROTATE CCS OR BCS INTO ICS; ROTATES ABOUT CCS OR BCS X-AXIS, Y TOWARD Z POSITIVE
ROTY	$-360. < ROTY \leq 360.$	0.0	SAME AS ROTX, EXCEPT ROTATES ABOUT Y, Z TOWARD X IS POSITIVE
ROTZ	$-360. < ROTZ \leq 360.$	0.0	SAME AS ROTX, EXCEPT ROTATES ABOUT Z, X TOWARD Y POSITIVE
R-CARD DATA:			
REFNO	1-99999	NONE	NUMBER OF REFLECTING PLANE SURFACE
D-CARD DATA:			
DV	FLOATING POINT	1.0	LENGTH UNIT MULTIPLIER
N-CARD DATA:			
INC	INTEGER	0.0	SURFACE NUMBER CHANGE VALUE (SURFN = SURFN + INC).

---

*If the surface has a component of specular reflectance and the can-shade flag is either unspecified or set to "NO," the flag is reset to "FF." If the flag is set to "DI" it is reset to "BOTH." (Specular surfaces must be shadowers in the FF segment.

RECTANGLE

SCS
METHOD

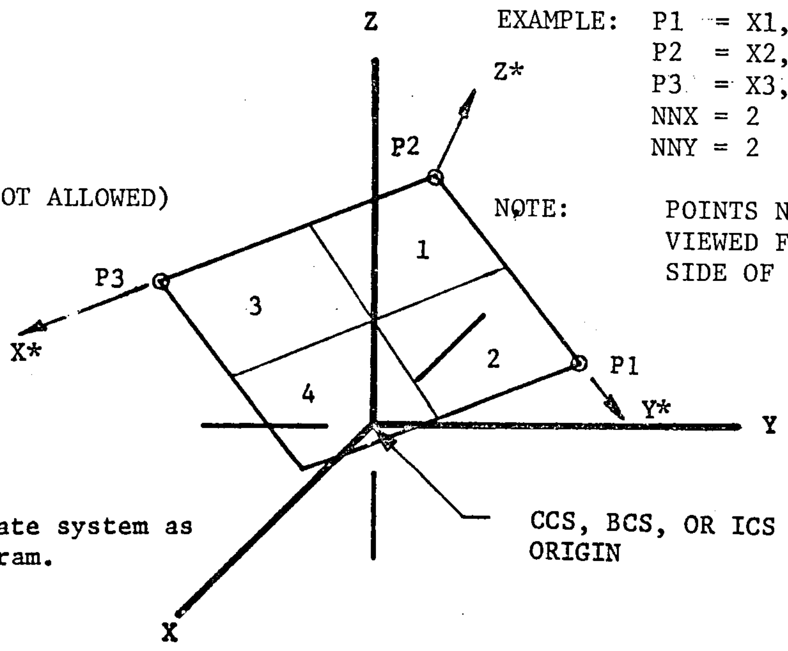


EXAMPLE: P1 = 2.0, 3.0, 0.0
NNX = 2
NNY = 2

NOTE: ONE CORNER MUST BE ON THE Z AXIS WITH RECTANGLE PARALLEL TO X-Y PLANE

THE FOLLOWING INPUTS ARE NOT ALLOWED (EITHER METHOD): NNZ, UNNZ, NNAX, UNNAX, NNR, UNNR, DIMENSIONS OR PN WITH N GREATER THAN 3.

POINT
METHOD
(POSITION DATA NOT ALLOWED)



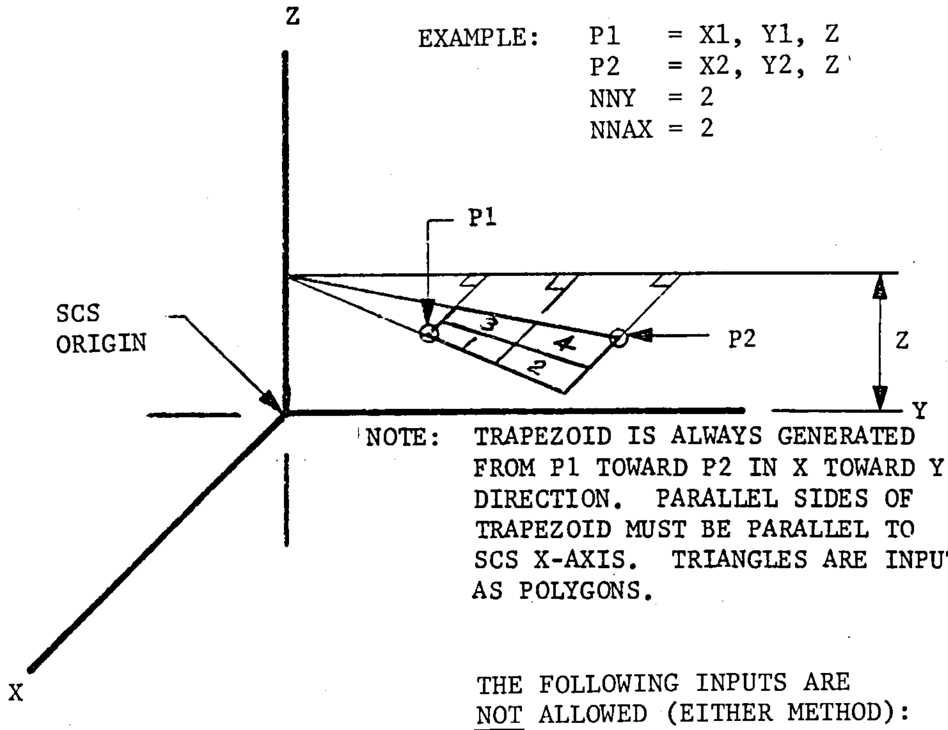
EXAMPLE: P1 = X1, Y1, Z1
P2 = X2, Y2, Z2
P3 = X3, Y3, Z3
NNX = 2
NNY = 2

NOTE: POINTS NUMBERED CCW AS VIEWED FROM THE "TOP" SIDE OF THE SURFACE

*"Surface" coordinate system as generated by program.

Figure 3-7 Surface Geometry Definition

SCS
METHOD

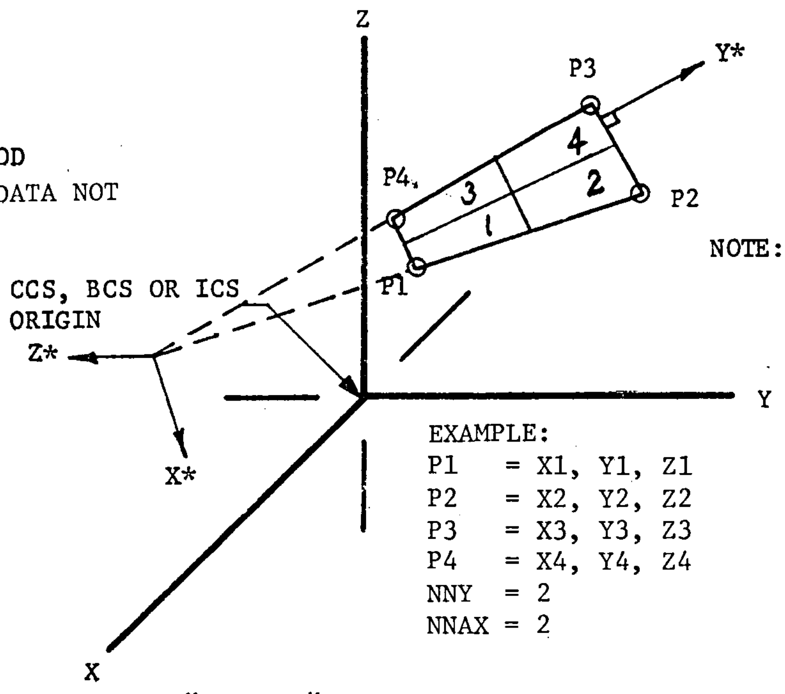


EXAMPLE: P1 = X1, Y1, Z
P2 = X2, Y2, Z
NNY = 2
NNAX = 2

NOTE: TRAPEZOID IS ALWAYS GENERATED FROM P1 TOWARD P2 IN X TOWARD Y DIRECTION. PARALLEL SIDES OF TRAPEZOID MUST BE PARALLEL TO SCS X-AXIS. TRIANGLES ARE INPUT AS POLYGONS.

THE FOLLOWING INPUTS ARE NOT ALLOWED (EITHER METHOD): NNX, UNNX, NNZ, UNNZ, NNR, UNNR, DIMENSIONS, OR PN WITH N GREATER THAN 4.

POINT METHOD
(POSITION DATA NOT ALLOWED)



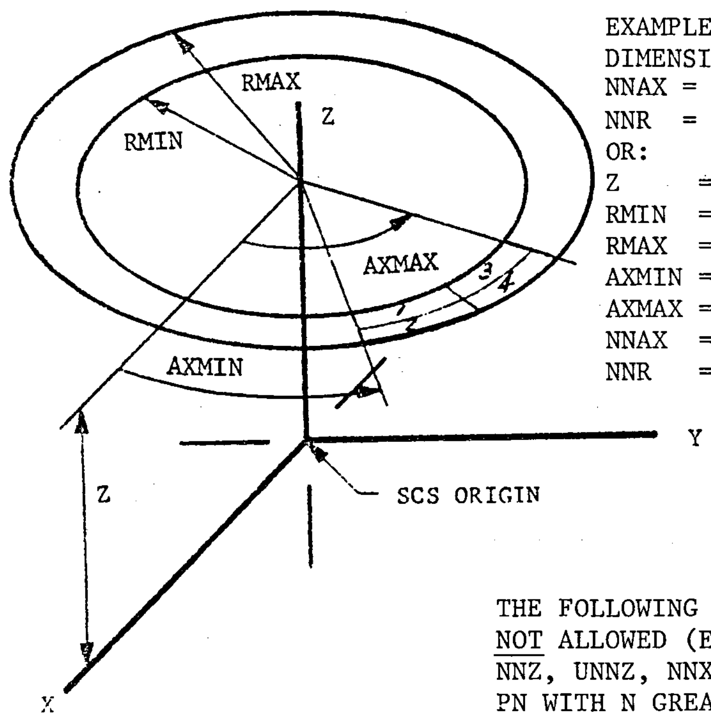
NOTE: POINTS ARE NUMBERED CCW AROUND FIGURE AS VIEWED FROM THE "TOP" SIDE OF THE SURFACE. P1-P4 MUST BE SHORTER THAN AND PARALLEL TO P2-P3. FOR A TRIANGLE, P1 AND P4 ARE ENTERED AS DUPLICATE POINTS

EXAMPLE:
P1 = X1, Y1, Z1
P2 = X2, Y2, Z2
P3 = X3, Y3, Z3
P4 = X4, Y4, Z4
NNY = 2
NNAX = 2

*"Surface" coordinate system as generated by program.

TRAPEZOID Figure 3-7 (continued)

SCS
METHOD

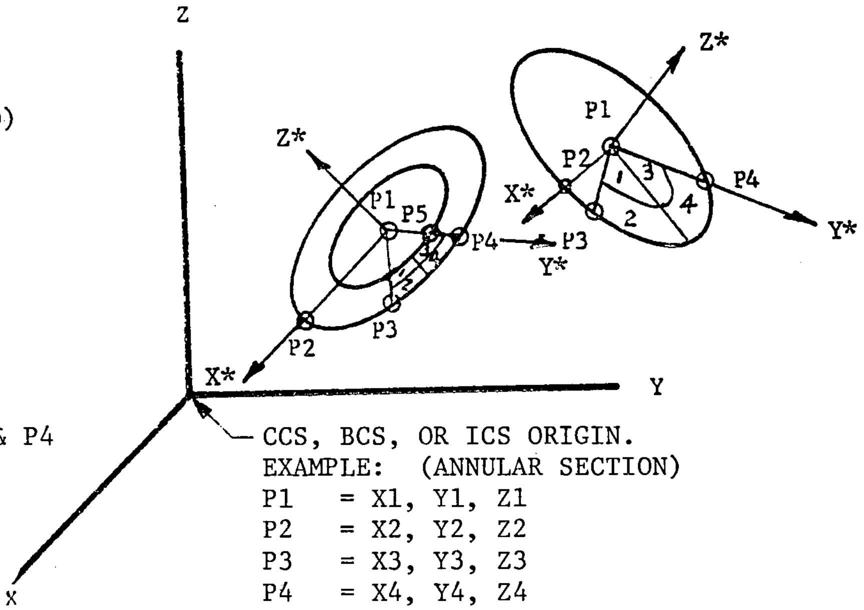


EXAMPLES:
DIMENSIONS = 12.,10.,15.,25.,45.
NNAX = 2
NNR = 2
OR:
Z = 12.
RMIN = 10.
RMAX = 15. ALTERNATE
AXMIN = 25.
AXMAX = 45.
NNAX = 2
NNR = 2

THE FOLLOWING INPUTS ARE
NOT ALLOWED (EITHER METHOD):
NNZ, UNNZ, NNX, UNNX, NNY, UNNY, OR
PN WITH N GREATER THAN 5.

POINT
METHOD
(POSITION DATA NOT ALLOWED)

EXAMPLE: (PIE-SECTION)
P1 = X1, Y1, Z1
P2 = X2, Y2, Z2
P3 = X3, Y3, Z3
P4 = X4, Y4, Z4
NNAX = 2
NNR = 2
NOTE:
P1 = DISC CENTER, P2, P3 & P4
ENTERED CCW, AS SEEN FROM
"TOP" SIDE



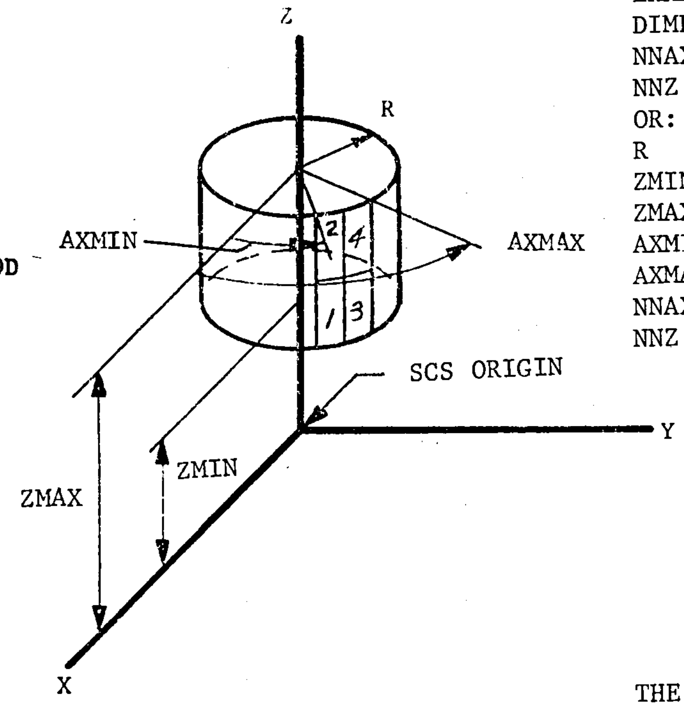
EXAMPLE: (ANNULAR SECTION)
P1 = X1, Y1, Z1
P2 = X2, Y2, Z2
P3 = X3, Y3, Z3
P4 = X4, Y4, Z4
P5 = X5, Y5, Z5
NNAX = 2
NNR = 2

*"Surface" coordinate system as generated by program.

NOTE: P1, P3, P4, AND P5 NUMBERED CCW AS VIEWED FROM THE "TOP" SIDE OF THE SURFACE. LINE P2-P1 MUST BE PERPENDICULAR TO LINE P4-P1 AND ALL OR PART OF THE ACTIVE SURFACE MUST LIE BETWEEN P2 AND P4.

DISC Figure 3-7 (con't)

SCS
METHOD

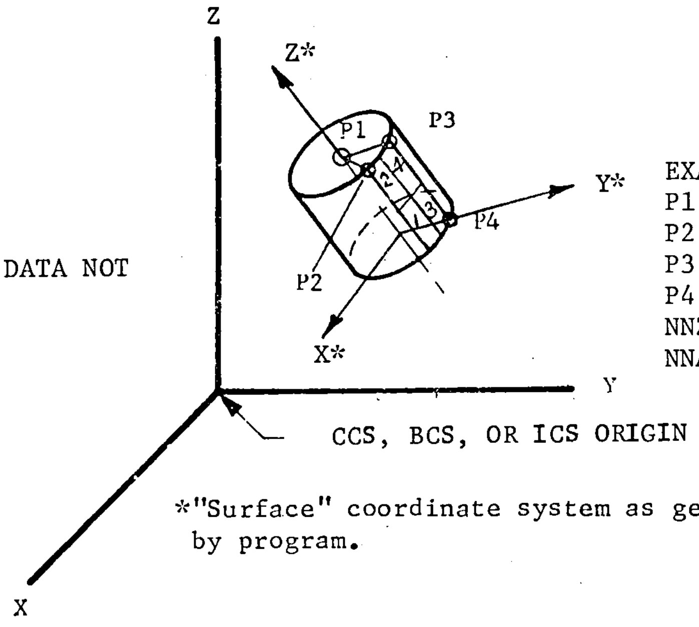


EXAMPLES:
DIMENSIONS = 11.5,10.,22.,25.,45.
NNAX = 2
NNZ = 2
OR:
R = 11.5
ZMIN = 10.
ZMAX = 22.
AXMIN = 25.
AXMAX = 45.
NNAX = 2
NNZ = 2

ALTERNATE

THE FOLLOWING INPUTS ARE NOT ALLOWED (EITHER METHOD):
NNX, UNNX, NNY, UNNY, NNR, UNNR OR PN WITH N GREATER THAN 4.

POINT
METHOD
(POSITION DATA NOT
ALLOWED)

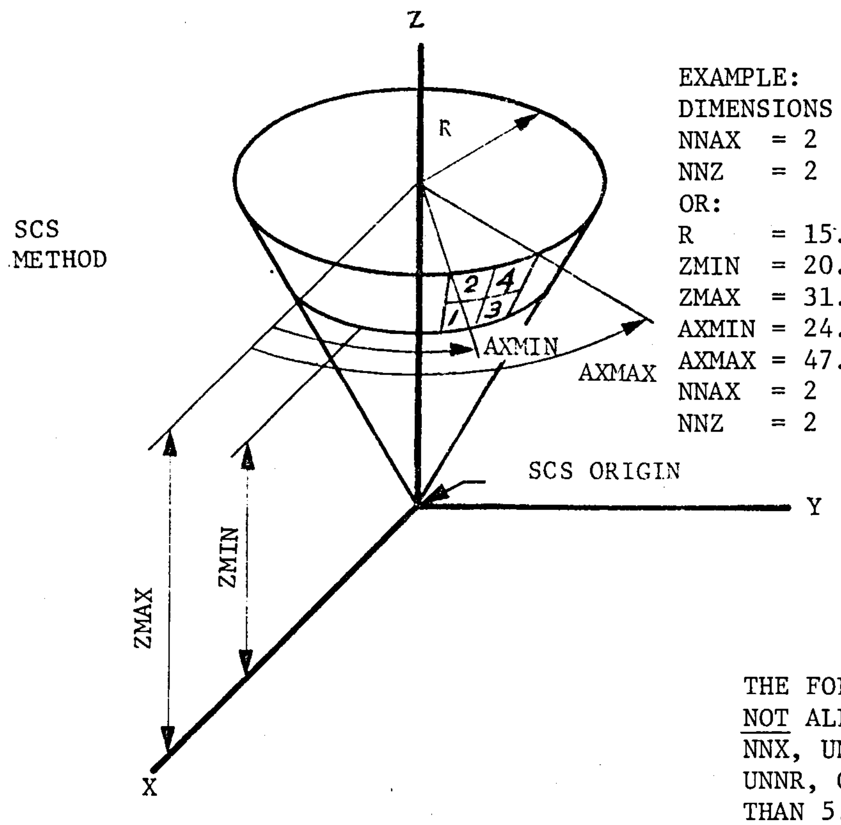


EXAMPLE:
P1 = X1, Y1, Z1
P2 = X2, Y2, Z2
P3 = X3, Y3, Z3
P4 = X4, Y4, Z4
NNZ = 2
NNAX = 2

*"Surface" coordinate system as generated by program.

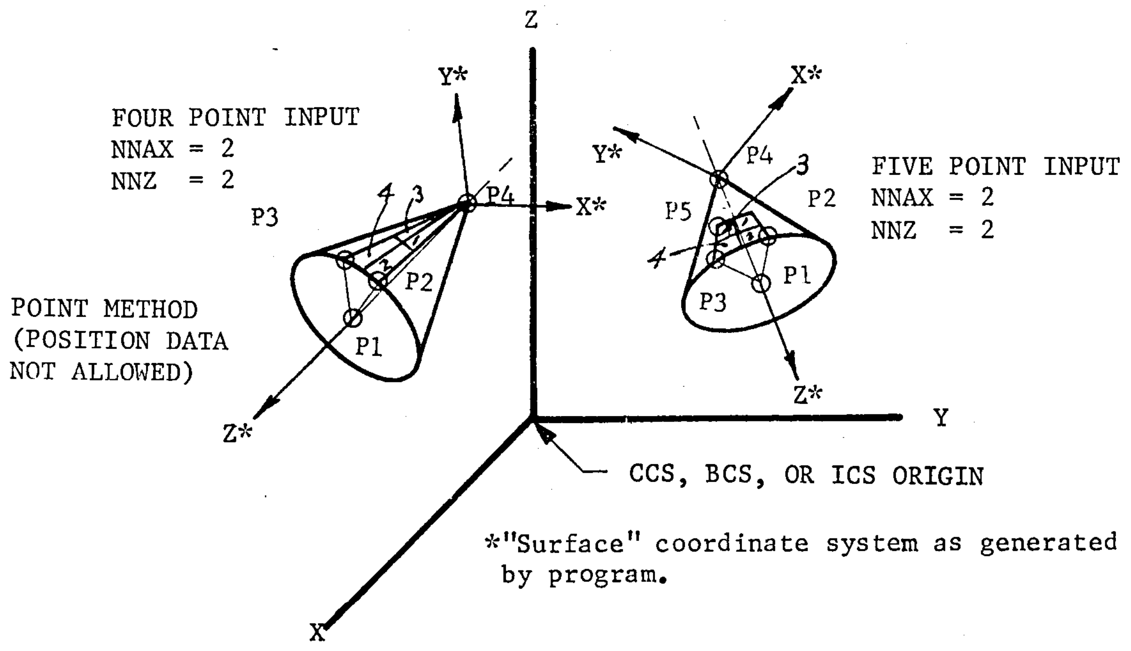
NOTE: SURFACE GENERATED FROM P2 TO P3, CCW ABOUT AXIS AS VIEWED FROM P1 END.

CYLINDER Figure 3-7



EXAMPLE:
DIMENSIONS = 15.,20.,31.5,24.2,47.
NNAX = 2
NNZ = 2
OR:
R = 15.
ZMIN = 20.
ZMAX = 31.5 ALTERNATE
AXMIN = 24.2
AXMAX = 47.
NNAX = 2
NNZ = 2

THE FOLLOWING INPUTS ARE NOT ALLOWED (EITHER METHOD): NNX, UNNX, NNY, UNNY, NNR, UNNR, OR PN WITH N GREATER THAN 5.

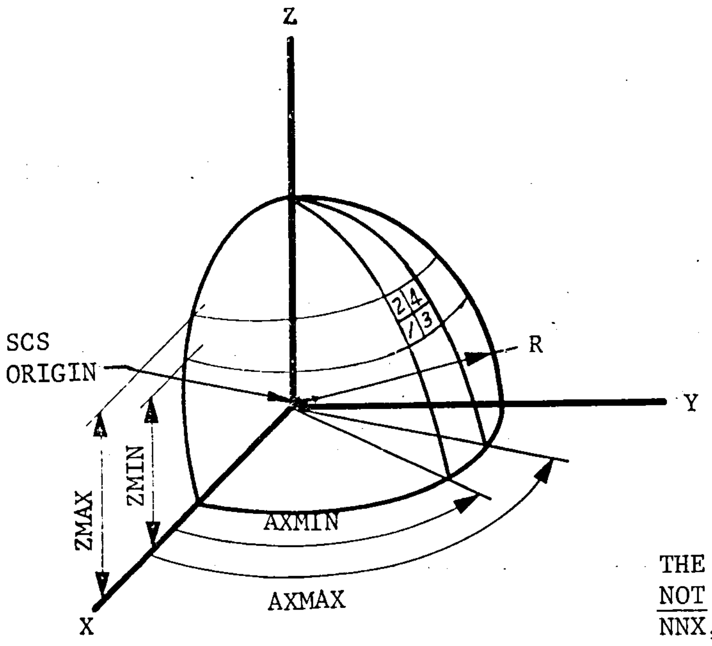


*"Surface" coordinate system as generated by program.

NOTE: SURFACE GENERATED FROM P2 TO P3, CCW ABOUT AXIS AS VIEWED FROM P1 TOWARD P4

CONE Figure 3-7 (con't)

SCS
METHOD

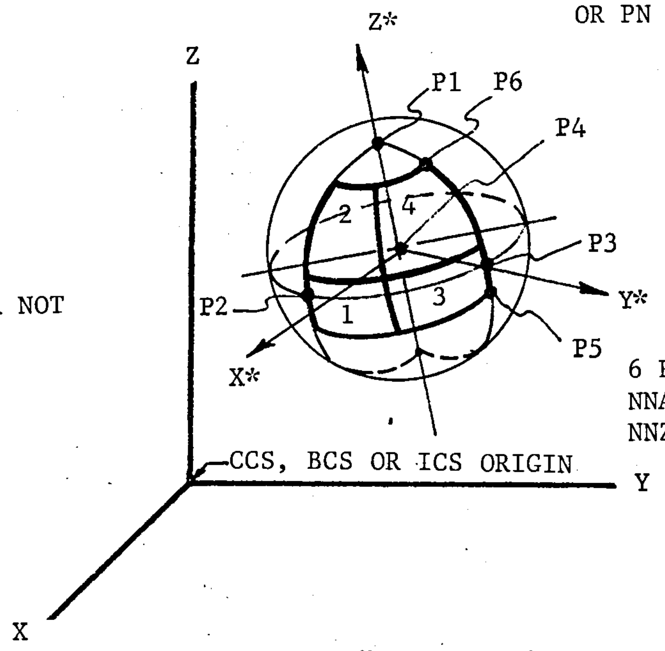


EXAMPLES:
DIMENSIONS = 20.,11.,15.,52.,60.
NNZ = 2
NNAX = 2
OR:

R	=	20.	
ZMIN	=	11.	
ZMAX	=	15.	ALTERNATE
AXMIN	=	52.	
AXMAX	=	60.	
NNZ	=	2	
NNAX	=	2	

THE FOLLOWING INPUTS ARE
NOT ALLOWED (EITHER METHOD):
NNX, UNNX, NNY, UNNY, NNR, UNNR,
OR PN WITH N GREATER THAN 6.

POINT
METHOD
(POSITION DATA NOT
ALLOWED)

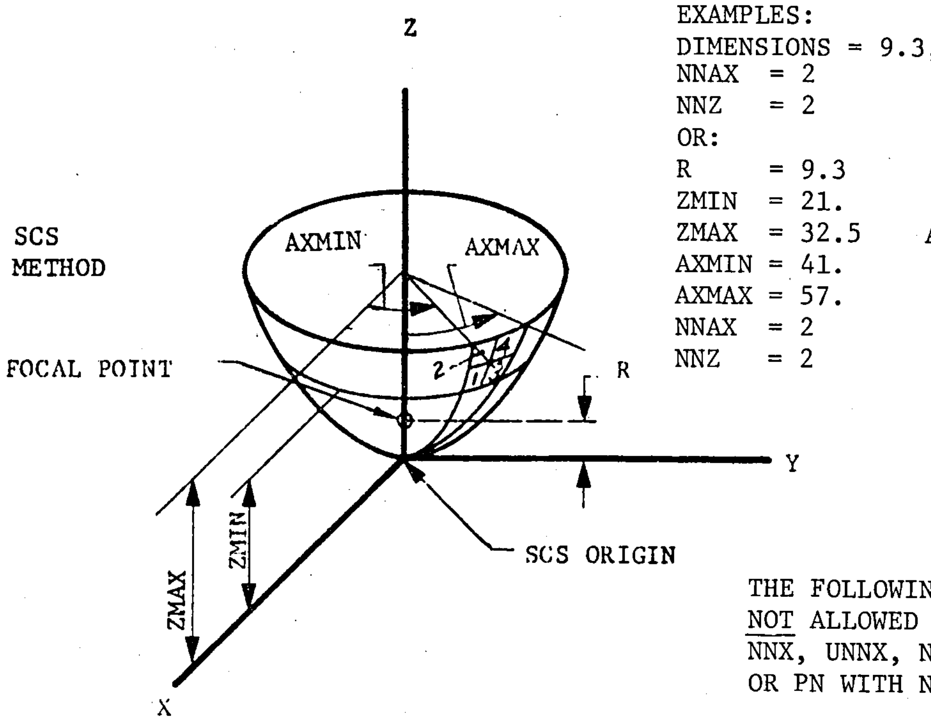


6 POINT INPUT
NNAX = 2
NNZ = 2

NOTES:
- o P1 defines the "North Pole" of the sphere.
- o P2 and P3 define the equatorial plane of the sphere and the extremities of the active portion of the sphere in the angular direction about the polar axis. Surface is generated from P2 to P3 CCW as viewed from P1 toward P4.
- o P4 defines the center of the sphere.
- o P5 and P6 must lie on the longitudinal line defined by P1 and P3. P5 and P6 define the extremities of the active portion of the sphere as measured along the polar axis.
- o All six points must be input for any partial definition of a spherical surface.
- o A complete sphere is generated from 3 points: P1 - North Pole, P2 - Center, P3 - Point on Equator where node generation begins.

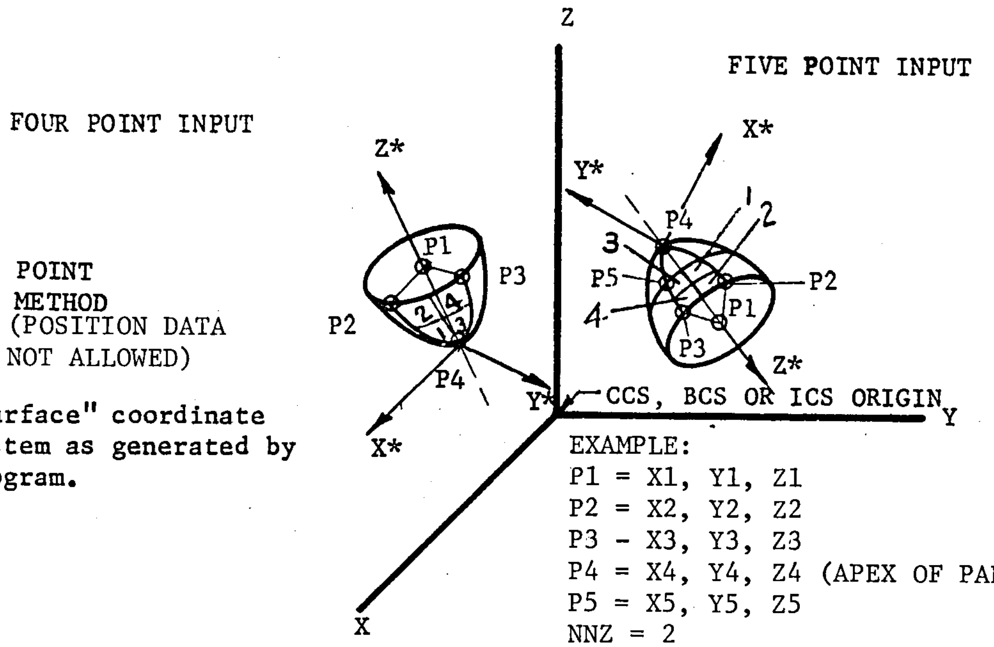
* "Surface" coordinate system as generated by program.

SPHERE Figure 3-7 (con't)



EXAMPLES:
DIMENSIONS = 9.3,21.,32.5,41.,57.
NNAX = 2
NNZ = 2
OR:
R = 9.3
ZMIN = 21.
ZMAX = 32.5 ALTERNATE
AXMIN = 41.
AXMAX = 57.
NNAX = 2
NNZ = 2

THE FOLLOWING INPUTS ARE
NOT ALLOWED (EITHER METHOD)
NNX, UNNX, NNY, UNNY, NNR, UNNR,
OR PN WITH N GREATER THAN 5.



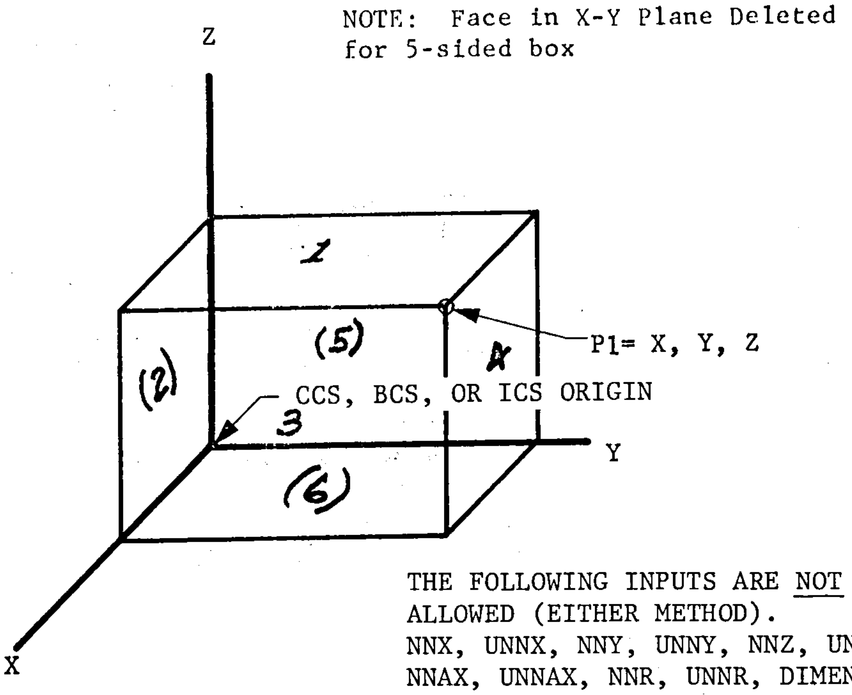
*"Surface" coordinate system as generated by program.

EXAMPLE:
P1 = X1, Y1, Z1
P2 = X2, Y2, Z2
P3 - X3, Y3, Z3
P4 = X4, Y4, Z4 (APEX OF PARABOLOID).
P5 = X5, Y5, Z5
NNZ = 2
NNAX = 2

NOTE: P1 AND P4 DEFINE AXIS OF REVOLUTION
SURFACE IS GENERATED FROM P2 TO P3
CCW AS VIEWED FROM P1 TOWARD P4.

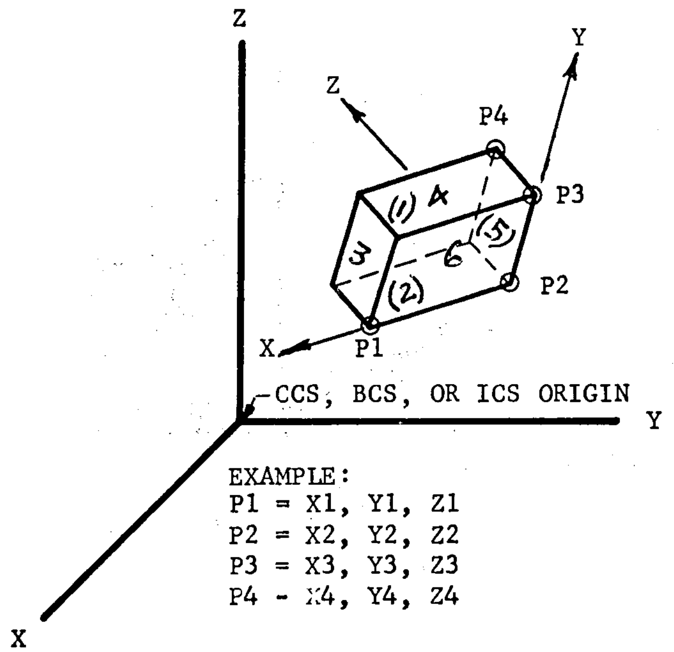
CIRCULAR PARABOLOID Figure 3-7 (con't)

SINGLE POINT METHOD



THE FOLLOWING INPUTS ARE <u>NOT</u> ALLOWED (EITHER METHOD).
NNX, UNNX, NNY, UNNY, NNZ, UNNZ, NNAX, UNNAX, NNR, UNNR, DIMENSIONS, POSITION, OR PN WHERE N IS GREATER THAN 4.

FOUR POINT METHOD



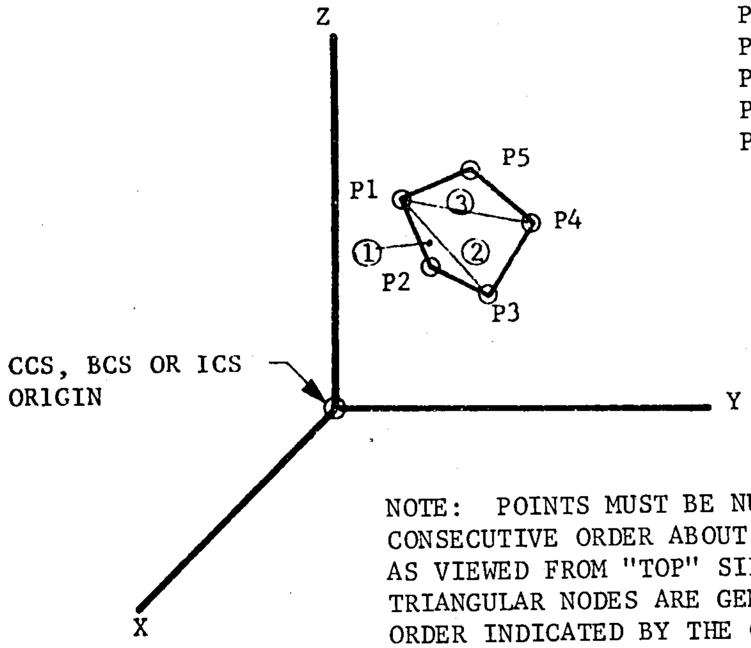
EXAMPLE:
P1 = X1, Y1, Z1
P2 = X2, Y2, Z2
P3 = X3, Y3, Z3
P4 - X4, Y4, Z4

NOTE: P1, P2 AND P3 MUST ALL LIE ON SAME FACE OF FIGURE, WITH P4 DIAGONALLY OPPOSITE P1. FACE CONTAINING P1, P2 and P3 DELETED FOR 5-SIDED BOX.

5 & 6 SIDED BOXES Figure 3-7 (Con't)

THE FOLLOWING INPUTS ARE NOT ALLOWED:
NNX, UNNX, NNY, UNNY, NNZ, UNNZ, NNR, UNNR, DIMENSIONS, POSITION, OR PN WITH N GREATER THAN 19.

EXAMPLE:
P1 = X1, Y1, Z1
P2 = X2, Y2, Z2
P3 = X3, Y3, Z3
P4 = X4, Y4, Z4
P5 = X5, Y5, Z5



NOTE: POINTS MUST BE NUMBERED IN CONSECUTIVE ORDER ABOUT FIGURE, CCW AS VIEWED FROM "TOP" SIDE OF SURFACE. TRIANGULAR NODES ARE GENERATED IN THE ORDER INDICATED BY THE CIRCLED NUMBERS.

N SIDED POLYGON Figure 3-7 (concl)

The variables found in the surface data block can be grouped as follows:

- general data
- dimensional data
- properties data
- position data
- ICS definition data

A summary of the function of each data group follows:

General data - this "catch all" data group is used to define:

1) node identification numbers
2) surface type (disk, sphere, etc.)
3) active side information
4) shadowing information, and
5) nodal breakdown and dimension information.

Dimensional data - This data group is used to define the desired boundaries of the geometric surfaces and portions of same.

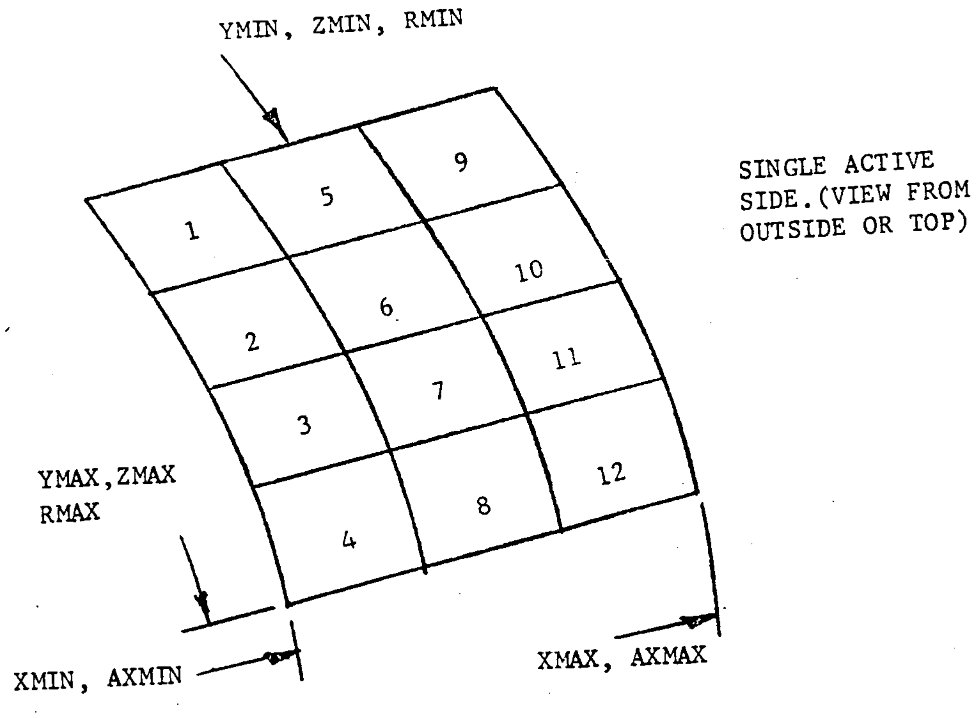
Properties data - This data group is used to define the optical properties of the surfaces. Properties allowed are diffuse solar absorptivity and transmissivity, diffuse infrared emissivity and transmissivity, specular solar reflectivity, and specular infrared reflectivity.

Position data - These data are the six rotation and translation variables necessary to locate an SCS relative to an ICS, BCS, OR CCS.

3.3.3.6 Nodal Surface Identification

When a surface is subdivided into nodal surfaces, the user has the option of numbering the nodal surface consecutively, beginning with the identification number he used for the surface, or arbitrarily, using a node number array. In either case, he must understand the scheme used by TRASYS to identify nodal surfaces. Figure 3-8 illustrates this process with examples of surfaces with single and dual active sides.

The user will no doubt quickly discover from Figure 3-7 that the node numbering schemes are related to the SCS-referenced method but have no relation to the CCS-referenced point method. The number scheme functions with point input, however, because the first step in processing a point-defined surface is to provide it with an internally generated SCS. Once this is done, the node numbering scheme can proceed. It is necessary, therefore, for the user to understand how the internally generated surface coordinate system relates to his point input. This is illustrated for each surface type in Figure 3-7.



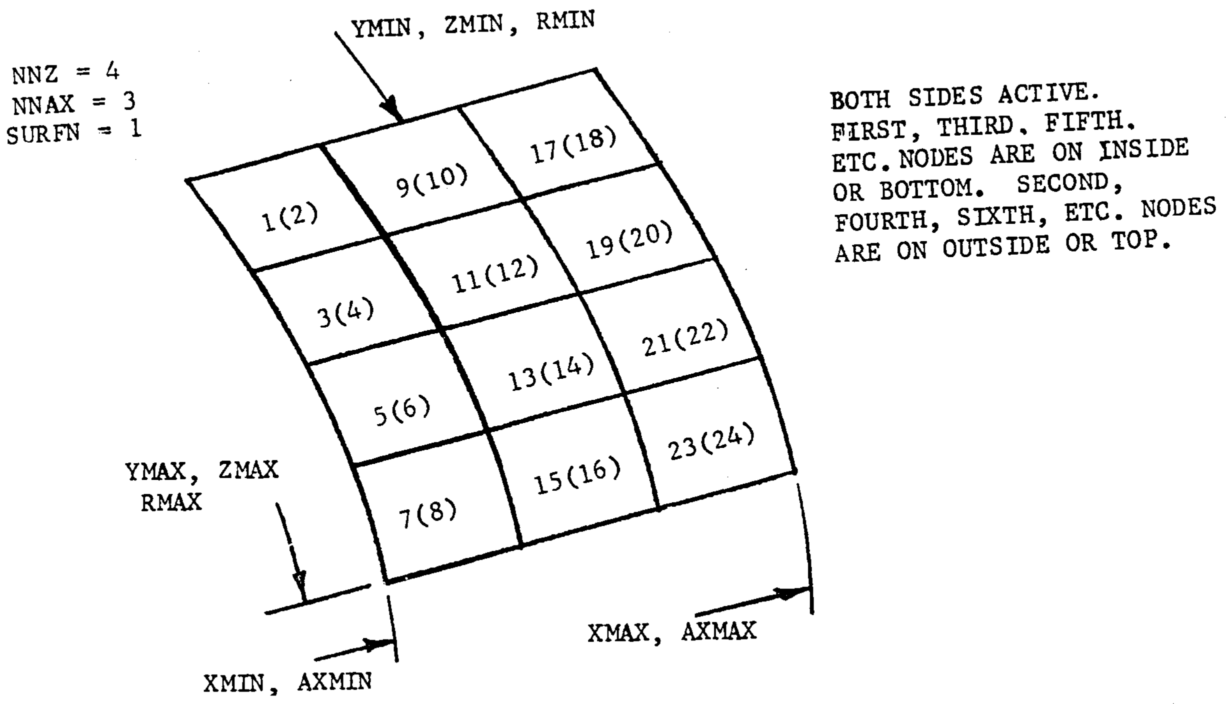


Figure 3-8 Node Generation Order

An understanding of Figures 3-7 and 3-8 should enable the user to properly number his nodal surfaces when using point input to define the surface.

The user should realize that the generalized node breakdown schemes involving NNX, NNY, UNNX, UNNY, etc., do not pertain to the BOX and POLYGON surface types where a fixed node generation scheme exists. If a user desires to subdivide the faces of a box, the faces must be input as rectangles. Subdividing a polygon requires entering the individual triangles desired. The user may wonder at the capricious-looking node breakdown that results from his polygon input. This occurs because shadowing solutions exist for triangles, but not polygons. After processing, the polygon's triangles are combined for output as one node, but the user should be aware of this subdivision process in order to avoid duplication of node numbers.

3.3.3.7 Dimensional Units

Nodal areas are carried in data storage for direct irradiation and radiation conductor calculations. For this reason, surface data length inputs must be in feet, the standard TRASYS length unit. Convenient means of units control are provided by D-cards (Ref. 3.3.3.9.1).

In regard to the surface data block, the user needs to remember that all the linear dimensions he uses in defining the surfaces of his model must be in feet (after D-card manipulations) and that all angular measurements must be in degrees (and decimal fractions of degrees) of arc.

3.3.3.8 Properties Data

In its present state of development, TRASYS is restricted to the assumptions that all surfaces are "gray", all surfaces emit diffusely, and all surfaces reflect with diffuse and specular components of reflectance.
That is:

$$\alpha_{IR} + \rho_{IR} + \rho^{s}_{IR} + \tau_{IR} = 1.0$$
$$\alpha_{s} + \rho_{s} + \rho^{s}_{s} + \tau_{s} = 1.0$$

where:

α = diffuse absorptivity
ρ = diffuse reflectivity component
ρ^{s} = specular reflectivity component
τ = transmissivity

subscripts:

IR = infrared waveband
s = solar waveband

It might be observed that since semitransparent materials and specular surfaces are allowed, the form factors are not, in general, surface property independent but a function of the transmissivities and specular components of reflectivity.

Material transmissivity plays a part in form factor calculations where blockage by a semitransparent surface is involved. The assumption made for these calculations is that any element to element configuration factor with an intervening semitransparent surface is multiplied by a shadow factor equal to the value of the blocking surface's transmissivity. This is a reasonable approach for thin intervening bodies.

Since only surfaces, rather than bodies, are used in TRASYS calculations, only one face of the semitransparent body will "count" as a shadower. If two surfaces are input for one body, the square root of the transmissivity can be used as the shadow factor to avoid having the shadowed configuration factors erroneously multiplied by the square of the transmissivity. In this case, the user must enter a negative transmissivity value. This is detected by the program and the absolute value of the square root of the transmissivity used as the shadow factor. Note: if two sides of a semitransparent body are generated using ACTIVE = BOTH, negative transmissivities are not required because only one shadowing surface is generated.

Specular reflectivity comes into play in the form factor calculations as a result of the imaging techniques used and the definition of an "image factor" as described in Appendix I.

It might be noted that the presence of semitransparent surfaces where $\tau_{IR} \neq \tau_{s}$ and/or the presence of specular surfaces where $\rho^{s}_{IR} \neq \rho^{s}_{s}$ results in different form factor matrices for the infrared and the solar wavebands. Both of these matrices are carried in program data storage and are printed in the standard output. It should be noted that even when there are no semi-transparent surfaces present both form factor matrices are carried in the program data storage, however they will be identical.

3.3.3.9 Surface Data Format

3.3.3.9.1 Control Field Formats

Eight different types of cards containing control field information are allowed in the surface data block. The card types are:

1. New Surface Card (S Card)
2. BCS Identifier Card (B Card)
3. ICS Definition Card (I Card)
4. Constant Definition Card (K Card)
5. Linear Dimension Units Card (D Card)
6. Node identification number increment card (N card)
7. Reference plane for imaging surfaces and/or BCSs (R Card) (Ref. para. 3.3.3.10).
8. Comment Cards

S Cards are used to signal the completion of the input for a surface and the beginning of a new surface. Their general format is:

CC1	CC7	CC7 3
S	Any surface Data	Card ID Information

Any data encountered beginning with an S card and ending with the card preceding the next S or R card is presumed to apply to a single surface and is defined as a surface description. If insufficient data to define a surface is found following an S card, either default will be supplied or an error message results. If redundant data is entered an error message results.

B cards are used to identify surfaces with the desired block coordinate system. Their general format is:

```
CC1          CC7                       CC7
                                        3

  BCS          Block Coordinate System Name    Card ID
```

All surface descriptions encountered between two B cards will be keyed to the BCS name found in the leading B card. Any surfaces not preceded by a B card will be automatically keyed to a block coordinate system named ALLBLK. BCS ALLBLK defaults to zero rotation and translation parameter values. That is, it coincides with the CCS. It may be redefined by the User with appropriate data in the Header BCS Data Block.

See Sections 3.3.3.10.3 and 3.3.3.10.4 for Alternate B Card formats for duplicating and imaging Blocks of Surface Data. In addition see Section 3.3.3.10 if the automatic generation of equivalent form factor for Duplicated and/or Imaged BCS feature is desired.

I Cards are used for definition of intermediate coordinate systems. Their general format is:

```
CC1            CC7                              CC7
                                                 3
  I              Intermediate Coordinate System Data    Card ID
```

Continuation cards are allowed for ICS definition. In other words, all information encountered between an I card and another card containing information in CC1 (except for comment cards) is presumed to pertain to a single ICS.

NOTE: A general surface data deck structure rule is that all I cards must precede all S cards.

K cards are used for definition of user constants referred to in surface data. Their general format is:

CC1	CC7	CC73
K	Constants Data	Card ID

Continuation cards are allowed for constants definition. All cards between a K card and the next card with data in CC1 (excepting comment cards) can be thought of as a data subblock that defines surface data constants.

NOTE: A general surface data deck structure rule is that all K cards must precede all S cards.

It should be noted that a basic difference exists between K-card constants and constants entered in the quantities data block. Unlike quantities data constants, K-card constants are used for surface data manipulation only, and are not available during processor execution.

D-cards are used for linear dimension units control. Since TRASYS computations cannot be made independent of dimensional units, it was necessary to choose a standard units system for compatibility between the various computation segments and subroutines. The TRASYS standard length unit is feet, which is oftentimes inconvenient when the user is working from engineering drawings in inches, or perhaps a metric unit. This problem has been eliminated by allowing for dimension change (D-cards) in the surface data input. These cards function as follows: when a D-card is encountered in the surface data, all linear dimensions in the surface (S-card) data following will be multiplied by the floating point data value on the D-card. This holds true until another D-card is encountered or until the end of the surface data block. All intermediate coordinate systems referenced by surfaces being modified by a D-card are also modified by the D-card. This means that the following rule must be observed carefully: the linear dimensions on any ICS referred to in a surface description must agree with the linear dimensions of the pertinent surface data, prior to modification by a D-card.

This D-Card format is:

```
CC1          CC7
  D            DV (floating point)
```

A surface data block using D-cards is shown in Figure 3-10.

N-cards are used for node number redefinition. The thermal analysis of large vehicles frequently involves combining several TRASYS models into one. The component models will generally have been generated independently, perhaps by different contractors, and node/surface number duplication in the various surface data blocks will be common. The laborious task of renumbering nodes to eliminate duplication is alleviated considerably by use of the N-card option. When an N-card is encountered in the surface data, all node and surface numbers in the surface (S-card) data following will be changed accordingly to:

SURFN = SURFN + NINC

where:

NINC is an integer value found on the N-card.

This holds true until another N-card is encountered or until the end of the surface data block. Changing SURFN for a surface means that all node numbers associated with that surface are changed, whether automatically generated or input as an integer array.

```
HEADER OPTION DATA
TITLE  CLASSES FOLLY
       MODEL=CLAS

HEADER SURFACE DATA
D       1./12.  $ FOLLOWING LINEAR DIMENSIONS ARE MULTIPLIED BY 1./12.
S       SURFN=201,
        TYPE=CYL,
        R=1.0,ZMIN=3.0,ZMAX=15.0,AXMIN=0.0,AXMAX=360.0,NNZ=1,NNAX=1,
        ACTIVE=OUT, ALPHA=0.3,EMISS=0.9,
        TY=5.0,
S       SURFN=301
        TYPE=SPHER
        R=3.0,ZMIN=0.0,ZMAX=3.0,AXMIN=0.0,AXMAX=180.0,NNZ=1,NNAX=1,
        TZ=20.,
        ACTIVE=OUT,ALPHA=0.2,EMISS=0.9,
S       SURFN=401,
        TYPE=CONE
        R=1.0,ZMIN=0.0,ZMAX=2.0,AXMIN=0.0,AXMAX=360.0,NNZ=1,NNAX=1,
        TZ=17.0,ROTY=180.,
        TY=5.0,
        ACTIVE=OUT,ALPHA=0.9,EMISS=0.9,
S       SURFN=501
        TYPE=CONE,
        R=3.0,ZMIN=1.0,ZMAX=3.0,AXMIN=0.0,AXMAX=180.0,NNZ=1,NNAX=1,
        TZ=2.0,
        ACTIVE=OUT,ALPHA=0.9,EMISS=0.9,
D       1.  $ TERMINATES EFFECT OF PREVIOUS D-CARD
N       10  $ REMAINING NODE NUMBERS ARE INCREASED BY 10
S       SURFN=701
        TYPE=CONE
        R=1.0,ZMIN=1.0,ZMAX=2.0,AXMIN=0.0,AXMAX=360.0,NNZ=1,NNAX=1,
        TZ=1.0,TY=5.0,
        ACTIVE=OUT,ALPHA=0.9,EMISS=0.9,
S       SURFACE=901
        TYPE=TRAP
        P1=0.707*4.0,0.707*5.0,4.0,
        P2=0.707*3.0,0.707*3.0,5.0,
        P3=0.707*3.0,0.707*3.0,8.0,
        P4=0.707*5.0,0.707*5.0,6.0,
        ACTIVE=BOTH,ALPHA=0.2,EMISS=0.9,
S       SURFN    =905,TYPE=POLY
        P1=-.707*5.,.707*5.,4.
        P2=-.707*3.,.707*3.,5.
        P3=0.707*3.0,.707*3.0,8.0
        P4=0.707*5.,.707*5.,6.
        ACTIVE=BOTH,PROP=.2,.9
HEADER OPERATIONS DATA
BUILD  CLAS,ALLBLK
L      NPLOT
END OF DATA
```

FIGURE 3-10 D-Card and N-Card Operations Example

The following N-card restriction must be observed: the variable NINC may take on any positive or negative integer value such that

```
1   SURFN + NINC   99999
```

is true for all values of SURFN involved.

The N-card format is:

```
CC1                     CC7 to CC 72
  N                       NINC
```

A surface data block using N-cards is shown in Figure 3-10.

Comment cards, with the following format:

```
CC1            CC7                              CC7
                                                  3
  C              Comment Information              Card ID
```

may appear anywhere in the surface data block. Another means of entering comment information is to delimit a data field (CC7-72) with a $ and enter comment information to the right of it, as follows:

```
CC7                                         CC7
                                              3
  Surface Data $ Comment Data                 Card ID
```

This may tempt the user to place a $ in the data field of an otherwise blank card and enter a comment. This is illegal. It results in a blank data field and an error message.

3.3.3.9.2 Single Variable Input Format

Any single variable recognized as surface data may be entered in a card data field according to the general format:

```
CC7                                         CC7
                                              2
  NAME1 = DV, NAME2 = DV, ---
```

NAME1 and NAME2 may be any variable name defined by the surface data variables list (Table 3-I), plus in the case of K cards, the names may be as defined by the user, limited only by the mode and word length limit of 6 characters.

Single variable input is the only means available for defining the following list of surface data variables.

TYPE		NNX	SURFN
ACTIVE	NOTE 1	NNY	IREFSF
SHADE		NNAX	IDUPSF
BSHADE		ICSN (in a surface	IMAGSF
SPRI	NOTE 2	description)	REFNO
SPRS			

Notes: 1. If ACTIVE = BOTH and SHADE = NO, the SHADE Flag applies only to the "bottom" or "inner" surface. SHADE = NO automatically for the "top" or "outer" surface.

2. Values of either or both of these variables greater than zero requires:

NNX = NNY = 1 (one node allowed per specular surface)
TYPE = RECT, DISC, TRAP, BOX5, BOX6, or POLY (specular surfaces must be planar)

SHADE = FF or BOTH (specular surfaces must be shadowers in FF segment. This flag is reset by the program to "FF" if unspecified or specified as "NO" and is reset to "BOTH" if specified as "DI.")

All other surface data variables may be defined in convenient "short form" array formats per subsections 3.3.3.9.3 through 3.3.3.9.9.

3.3.3.9.3 Intermediate Coordinate System Data Format

ICS data may be entered in array format as follows:

```
CC1        CC7                                        CC7
                                                        2
 I           ICSN, TX, TY, TZ, ROTX, ROTY, ROTZ
```

This array defines the translations and rotations necessary to transform a BCS (or CCS) into the ICS. The new user should refer to para. 3.3.3.9.11 until defining this transformation becomes automatic. The three rotations, ROTX, ROTY, and ROTZ are performed in that order. If it is desired to alter the order of the rotations, the following hybrid format is used:

```
CC1        CC7                                        CC7
                                                        2
 I           ICSN, TX, TY, TZ, ROTY = DV,
             ROTZ = DV, ROTX = DV
```

for rotation first about the BCS Y-axis, second about the BCS Z-axis and third about the BCS X-axis.

It should be noted that each ICSN value appears at least twice in the surface data block. Once in the I card defining the ICS, and again in each surface description where an SCS/ICS transform is desired.

3.3.3.9.4 Surface Identification Format

A single node surface is identified as follows, in single variable input format:

```
CC1          CC7                          CC72

 S              SURFN = DV (Integer)
```

If a surface is to be subdivided into several nodes, and they are not to be numbered consecutively, the node number array may be entered according to the formats:

```
CC1          CC7                          CC72

 S              SURFN = DV1, DV2, ---DVN
```

or for consecutively numbered nodes

```
CC1          CC7
 S              SURFN = DV1
                which generates node numbers DV1, DV1 + 1,
                DV1 + 2, DV1 + (N-1)
```

for an N-node surface.

3.3.3.9.5 Properties Data Format

The diffuse properties data may be defined using the following format:

```
CC7                                                  CC7
                                                       2
  PROP = ALPHA, EMISS, TRANS, TRANI
```

If values for TRANI and TRANS are not encountered, they will default to zero.

Specular properties data must be input in the single variable format (see 3.3.3.9.2).

3.3.3.9.6 Dimensions Data Format

The dimensions data may be defined using the following format:

```
CC7                                                CC7
                                                     2
  DIMEN = R, ZMIN, ZMAX, AXMIN, AXMAX
```

3.3.3.9.7 Point Data Format

The x, y, z coordinates of point data input are defined using the following format:

```
CC7                                              CC7
                                                   2
  PN = XN, YN, ZN
```

N values up to 19 are recognized, depending on the surface type (Ref. Figure 3-7).

This is the only format allowed for point data. Single variable definitions are not allowed.

3.3.3.9.8 Position Data Format

The position data may be defined using the following format:

```
CC7                                                 CC7
                                                      2
  POSIT = TX, TY, TZ, ROTX, ROTY, ROTZ
```

This array defines the translations and rotations necessary to transform an ICS, BCS, or CCS into the SCS. The new user should refer to para. 3.3.3.9.11 until defining this transformation becomes automatic. The three rotations ROTX, ROTY, and ROTZ are performed in that order. If it is desired to alter the order of the rotations, the following format is used:

```
CC7                                                    CC7
                                                         2
  POSIT = TX, TY, TZ, ROTY = DV, ROTZ = DV, ROTX = DV
```

The Position Data is only applicable to the "SCS method" of surface definition and not the point method.

3.3.3.9.9 Comment Data Format

A Hollerith string of up to thirty characters may be entered with each surface description according to the following format:

```
CC7                                                  CC7
                                                       2
  COM = * Any Alphameric Data *
```

These comments will be passed to the processor and printed with the surface description output that results from the BUILDC and ADD calls in the operations data.

3.3.3.9.10 Node Boundary Dimensions

When it is desired to generate an unequal node breakdown on a surface, it is necessary to define the node boundaries using one or more of the UNNX, UNNY, UNNZ, UNNAX, and UNNR arrays. Figure 3-11 illustrates this scheme for UNNZ and UNNAX.

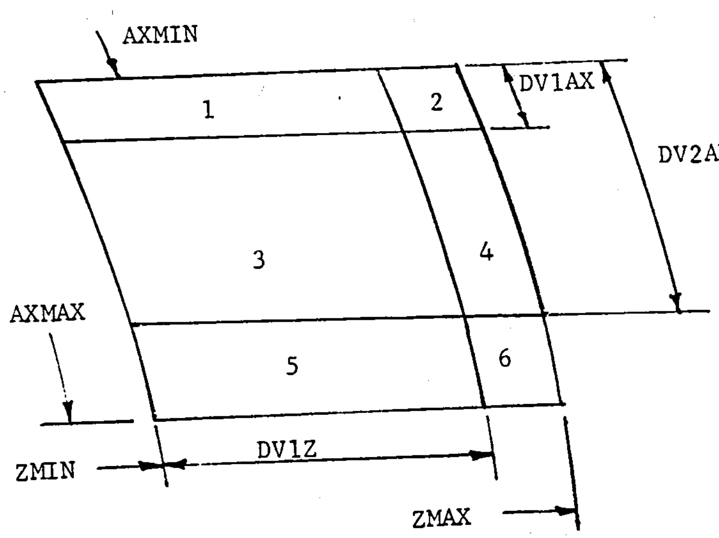


Figure 3-11 Example of Unequal Node Boundaries

This example required the following unequal boundary arrays:

UNNAX = DV1AX, DV2AX

UNNZ = DV1Z

These arrays are entered in the surface data block according to the following format:

```
          CC7                                                               CC7
                                                                              2
            UNNAX = DV1AX, DV2AX
            UNNZ  = DV1Z
```

The general format is:

```
          CC7                                                               CC7
                                                                              2
            UNNX  = DV1, DV2, . . . DVN
```

where: N = NNX -1.

3.3.3.9.11 Coordinate System Definition Process

The definition of any coordinate system relative to another requires the user to input the variables TX, TY, TZ, ROTX, ROTY, and ROTZ. Assuming that it is desired to define an SCS in the CCS, BCS, or ICS, the required variables can be evaluated by the following procedure:

1. Mentally locate the SCS origin in CCS, BCS or ICS 3-space.

2. The X, Y and Z coordinates, in CCS, BCS or ICS 3-space, of the SCS origin are TX, TY and TZ respectively.

3. Mentally locate the CCS, BCS or ICS origin at the SCS origin, with the CCS, BCS or ICS axes parallel to their actual directions.

4. Define up to three rotation angles ROTX, ROTY and ROTZ which can be performed in any order that will finish with the CCS, BCS, or ICS located per 3., coincident with the SCS.

   Enter the TX, TY, TZ, ROTX, ROTY and ROTZ values on the B-card in the HEADER BCS DATA block (to define the BCS relative to the CCS), and in the HEADER SURFACE DATA block on the I-card (to define the ICS relative to a BCS or the CCS), or in position data (to define the SCS relative to an ICS, a BCS, or the CCS), maintaining the order in which the rotations were performed.

3.3.3.10 DUP and IMAGE Options

Two options are available that enable the user to conveniently duplicate surfaces already defined in the surface data blocks or otherwise redefine them and add them to configuration or configurations in the surface data. Using the surface DUP option, single surfaces, consisting of one or more nodes can be created by referencing previously input surfaces. Using the BCS DUP option, entire blocks of surfaces previously defined under a particular BCS name can be created by simply referencing a previously input BCS name.

Similarly, images of single surfaces and blocks of surfaces can be created using the surface IMAGE option and the BCS IMAGE option. These options greatly reduce the user's effort if he is defining a bi-laterally symmetric configuration.

The BCS DUP and BCS IMAGE options are highly recommended whenever symmetry exists because the preprocessor logic that generates the symmetric surface descriptions also recognizes the equivalent form factors that exist due to symmetry. Equivalent form factor data is generated internally and passed on to the processor, thus eliminating redundant form factor calculations. Refer to para. 3.3.5.4 for guidance in using this feature.

3.3.3.10.1 Surface DUP Option

The following rules and restrictions apply when using the surface DUP option:

a) Surfaces to be duplicated must appear in the surface data block before the surfaces that are to be created by duping.

b) Any or all of the surface description variables of the surface being duplicated can be changed.

c) If the surface to be duplicated was input by the point method and any changes are to be made in the points, all points must be input for the surface being created.

d) Generated surfaces such as boxes and polygons will be duped in their entirety. That is, the individual nodes generated by "box" or "polygon" cannot be duped.

e) Surfaces created by the DUP option may later be imaged.

f) The surface to be duplicated is specified by setting the variable IDUPSF equal to the surface number.

g) Any correspondence data that applied to the nodes created by the Surface DUP option must be supplied by the user in the Correspondence data block.

A sample input deck using the surface DUP option can be found in Figure 3-12.

3.3.3.10.2 Surface IMAGE Option

The IMAGE option allows the user to create surfaces by imaging previously input surfaces in some specified reference plane. The following restrictions and rules apply when using the IMAGE option:

a) Surfaces to be imaged must appear in the surface data block before the surfaces that are to be created by imaging.

b) Reference planes (imaging planes) in which surfaces are to be imaged are special surfaces designated by R-cards. Each of these planes is assigned a unique identification number and is defined by specifying, in any order, any three non-colinear points lying on its surface. These points are defined with respect to the CCS.

c) Generated surfaces such as boxes and polygons will be imaged in their entirety. That is, individual nodes generated by "box" or "polygon" cannot be imaged.

d) Surfaces created by imaging cannot be duped.

e) For purposes of imaging, the image surface, the reflecting plane, and the imaged surface are treated internally to the program as if they were defined with respect to the central coordinate system.

```
HEADER OPTIONS DATA
TITLE  DUP OPTION SAMPLE PROBLEM
HEADER SURFACE DATA

S      SURFN          =10
       TYPE           =SPHERE
       R              =10
       ZMIN           =-9.99
       ZMAX           =9.99
       AXMIN          =0.
       AXMAX          =360.
       TX             =0.
       TY             =10.
       TZ             =20.
       ACTIVE         =OUT
       PROP           =0.2,0.9
       COM            =* SURFACE TO BE DUPED *
S      SURFN          =20
       IDUPSF         =10
       R              =20.
       ZMIN           =-19.99
       ZMAX           = 19.99
       TZ             =-20.
       PROP           =0.5,0.8
       COM            =* DUPLICATE OF SURFACE 10 *
HEADER OPERATIONS DATA
BUILD  FIG2, ALLBLK
L      NPLOT
END OF DATA
```

Figure 3-12 Sample Problem Using the Surface DUP Option

f) The surface to be imaged is specified by setting the variable IMAGSF equal to the imaged surface number. The reference plane in which IMAGSF is to be imaged is specified by setting IREFSF equal to the reference plane number.

g) When a surface is imaged, the nodes are also imaged resulting in a reversed order of node numbering. The active side of the surface also follows image rules. Figure 3-13 illustrates these phenomena.

h) Any correspondence data that applies to the nodes created by the surface IMAGE option must be supplied by the user in the correspondence data block.

A sample problem illustrating the surface IMAGE option can be found in Figure 3-14.

3.3.3.10.3 BCS DUP Option

The following rules and restrictions apply when using the BCS DUP option.

a) All the surfaces comprising the BCS to be duplicated must appear prior to the BCS that exercises the BCS DUP option.

b) All dimensional properties, surface properties and shadowing characteristics of each surface in the BCS generated by the BCS DUP option remain exactly the same as those in the BCS that was "DUPED". The new surfaces can be moved as a group by the parameters defining the new BCS in the BCS data block. The remaining properties may be altered in the operations data block by use of the "MOD" series of subroutine calls, (Reference Section 4.3.8).

c) Blocks of surfaces created by the BCS DUP option may be again DUPED by another BCS DUP operation.

d) Blocks of surfaces created by the BCS IMAGE option (see 3.3.3.10.4) cannot be DUPED.

e) Surfaces defined in the usual manner that follow the BCS card that performs BCS duping will simply appear together with the surfaces created under the same BCS name.

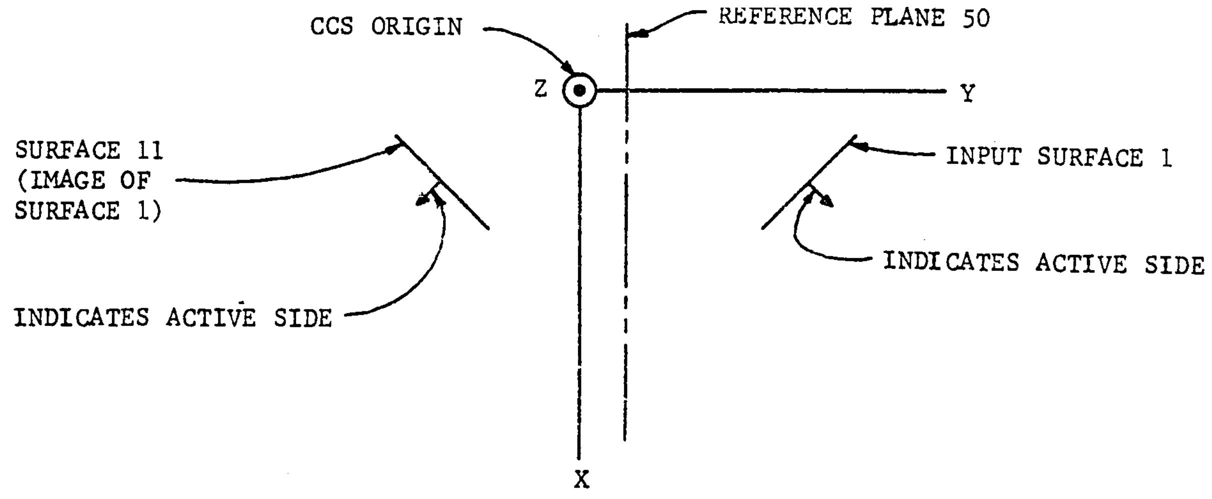
f) BCSs may not be DUPED under the same BCS name.

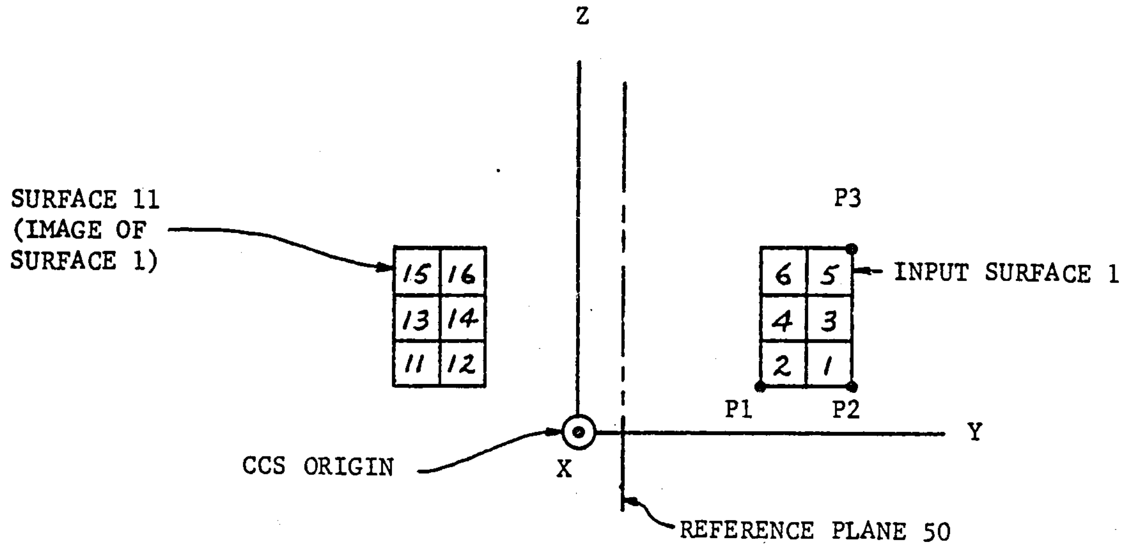
g) User correspondence data for the surfaces generated under the BCS DUP option need not be defined as such in the correspondence data block. Correspondence data applying to nodes within a BCS to be "duped" may be entered with a BCS name reference in the correspondence data block. See Section 3.3.8.6 for guidance in using this feature.

NOTE: Correspondence data for polygons created by BCS DUP is automatically generated.

BCS DUP card format:

```
CC1       CC7
  BCS        BCSNAM, DUPBCS = INAME, NINC = NNN
```





NOTE: NODE X+10 IS THE IMAGE OF NODE X

Figure 3-13 Imaging of Nodes and Active Side (Image Option Sample Problem)

```
HEADER OPTIONS DATA
TITLE  IMAGE OPTION SAMPLE PROBLEM
HEADER SURFACE DATA

S      SURFN          = 1
       TYPE           = RECT
       ACTIVE         = TOP
       NNX            = 3
       NNY            = 2
       PROP           = 0.5,0.9
       P1             = 3.0, 4.0, 1.0
       P2             = 1.0, 6.0, 1.0
       P3             = 1.0, 6.0, 4.0

       COM            = *SURFACE TO BE IMAGED*
S      SURFN          = 11
       IMAGSF         = 1
       IREFSF         = 50
       ICSN           = 100
       COM            = *IMAGE OF SURFACE 1*
R      REFNO          = 50
       P1             = 0.0, 1.0, 0.0
       P2             = 0.0, 1.0, 1.0
       P3             = 1.0, 1.0, 1.0

       COM            = *REFERENCE PLANE*
HEADER OPERATIONS DATA
BUILD  FIG1,ALLBLK
L      NPLOT
END OF DATA
```

Figure 3-14 Sample Problem Using the Surface Image Option

where:

BCSNAM =	Name of block coordinate system under which DUP-generated surfaces are placed.
INAME =	Name of block coordinate system under which surfaces to be "DUPED" are defined.
DUPBCS =	Required control word.
NINC =	Node number increment applied to all nodes under BCS INAME to generate the node numbers under BCS BCSNAM. (Integer Number)

A sample input deck using the BCS DUP option can be found in Figure 3-14A.

3.3.3.10.4 BCS IMAGE Option

The following rules and restrictions apply when using the BCS IMAGE option.

a) All the surfaces comprising the BCS to be imaged must appear prior to the BCS that exercised the BCS image option.

b) All surface properties and shadowing characteristics of each surface in the BCS generated by the BCS IMAGE option remains the same as those of the surfaces that were imaged. All dimensional properties appear as mirror images of the original surfaces, as seen in the "reflecting" plane. The surface properties and shadowing characteristics may be altered using the MOD subroutines in the operations data block. (Reference Section 4.3.8)

c) Blocks of surfaces created by the BCS IMAGE option may be again IMAGED by another BCS IMAGE operation, but they may not be DUPED.

d) Surfaces defined in the usual manner that follow the BCS card that performs BCS imaging will simply appear together with the surfaces created under the same BCS name.

e) Automatic generation of correspondence data applies to both the BCS DUP & BCS IMAGE features. Refer to para. 3.3.8.5 under correspondence data for guidance.

BCS cards that exercise the BCS IMAGE option have the following format and variable definitions:

```
 CC1         CC7
   BCS         BCSNAM,IMGBCS = INAME,NINC=NNN,IREFSF=NNN
```

BCSNAME, INAME and NINC are defined the same as on BCS cards that exercise duping. Imaged surfaces may become part of a previously defined BCS system (INAME) by making BCSNAM identical to INAME. IREFSF is the number of the reflecting plane surface. (Integer 1-99999) IMGBCS is a required control word.

```
HEADER OPTIONS DATA
TITLE  GOBLES FIRST FOLLY
       MODEL=FIG1
HEADER SURFACE DATA
BCS    RIGHT
S      SURFN = 201,
       TYPE=CYL,
       R=1.0,ZMIN=3.0,ZMAX=15.0,AXMIN=0.0,AXMAX=360.0,NNZ=1,NNAX=1,
       ACTIVE=OUT,ALPHA=0.3,EMISS=0.9,
       TY=5.0
S      SURFN=301
       TYPE=SPHER
       R=3.0,AMIN=0.0,ZMAX=3.0,AXMIN=0.0,AXMAX=180.0,NNZ=1,NNAX=1,
       TZ=20.,
       ACTIVE=OUT,ALPHA=0.2,EMISS=0.9,
S      SURFN=401,
       TYPE=CONE
       R=1.0,AMIN=0.0,ZMAX=2.0,AXMIN=0.0,AXMAX=360.0,NNZ=1,NNAX=1,
       TZ=17.0,ROTY=180.,
       TY=5.0,
       ACTIVE=OUT,ALPHA=0.9,EMISS=0.9,
S      SURFN=501
       TYPE=CONE,
       R=3.0,AMIN=1.0,ZMAX=3.0,AXMIN=0.0,AXMAX=180.0,NNZ=1,NNAX=1,
       TZ=2.0,
       ACTIVE=OUT,ALPHA=0.9,EMISS=0.9,
BCS    LEFT,DUPBCS=RIGHT,NINC=500
C      ABOVE CARD GENERATES SURFACES 701,801,901, AND 1001.
C      BY DUPLICATION OF SURFACES UNDER BCS RIGHT.
C
HEADER BCS DATA
BCS    RIGHT,0.,0.,0.,0.,0.,0.
BCS    LEFT,0.,0.,0.,0.,0.,0.
HEADER OPERATIONS DATA
       CALL CHGBLK(LEFT,0.,0.,0.,0,0,0,0.,0.,180.)
BUILD  FIG1,RIGHT,LEFT
L      NPLOT
END OF DATA
```

Figure 3-14A BCS Duping Operations Example

```
HEADER   OPTION DATA
TITLE    GOBLES SECOND FOLLY
         MODEL FIG1
HEADER SURFACE DATA
BCS      RIGHT
S        SURFN=201,
         TYPE=CYL,
         R=1.0,ZMIN=3.0,ZMAX=15.0,AXMIN=0.9,AXMAX=360.0,NNZ=1,NNAX=1,
         ACTIVE=OUT,ALPHA=0.3,EMISS=0.9,
         TY=5.0
S        SURFN=301
         TYPE=SPHER
         R=3.0,ZMIN=0.0,ZMAX=3.0,AXMIN=0.0.AXMAX=180.0,NNZ=1,NNAX=1,
         TZ=20.,
         ACTIVE=OUT,ALPHA=0.2,EMISS=0.9,
S        SURFN=401,
         TYPE=CONE
         R=1.0,ZMIN=0.0,ZMAX=2.0,AXMIN=0.0,AXMAX=360.0,NNZ=1,NNAX=1,
         TX=17.0,ROTY=180.,
         TY=5.0,
         ACTIVE=OUT,ALPHA=0.9,EMISS=0.9,
S        SURFN=501
         TYPE=CONE
         R=3.0,ZMIN=1.0,ZMAX=3.0,AXMIN=0.0,AXMAX=180.0,NNZ=1,NNAX=1,
         TZ=2.0,
         ACTIVE=OUT,ALPHA=0.9,EMISS=0.9,
BCS      LEFT,IMGBCS=RIGHT,NINC=500,IREFSF=999
C        ABOVE CARD GENERATES SURFACES 701,801,901, AND 1001.
C        THEY ARE IMAGES OF 201,301,401, AND 501 AS SEEN
C        IN REFLECTING SURFACE 999.
C
R        REFNO=999
         P1   =0.,0.,0.
         P2   =1.0,0.,0.
         P3   =0.,0.,1.0
HEADER BCS DATA
BCS      RIGHT,0.,0.,0.,0.,0.,0.
BCS      LEFT, 0.,0.,0.,0.,0.,0.
HEADER OPERATIONS DATA
BUILD  FIG1,RIGHT,LEFT
L        NPLOT
END OF DATA
```

Figure 3-14B BCS Imaging Operations Example.

A sample input deck using the BCS IMAGE option can be found in Figure 3-14B. Careful examination of Figures 3-14A and 3-14B will show that they define the same configuration in a different manner.

3.3.3.11 Automatic Generation of Equivalent Form Factors

When the BCS dup or BCS image feature is used in surface data input, a symmetry situation is usually created, wherein a good many form factors are equivalent to each other. This is illustrated as follows: In Figure 3-16a, nodes 11, 12, 13, and 14 in BCS2 were created by duplication of nodes 1, 2, 3, and 4 in BCS 1 plus offsetting BCS2 and BCS1 by BCS data block or CHGBLK input. In this situation, F11-12 = F1-2, F11-13 = F1-3, and so on. The tedious task of writing the necessary form factor equivalence data has been taken over by software within TRASYS. This feature is implemented by an additional parameter on the BCS dup card. For the situation in Figure 3-16a, the card:

```
CC1     CC7
  BCS     BCS2,  DUPBCS = BCS1, NINC = 10, IGEN = PART
```

will create nodes 11 thru 14 and also the equivalent form factor data needed to avoid calculating all of the form factors between nodes 11, 12, 13, and 14. Note that the form factors between 1 thru 4 and 11 thru 14 (e.g., F1-12, F11-2, etc.) must still be calculated because there is no predictable symmetry involved.

There are situations where this auto-generation of equivalent form factors is not appropriate. In figure 3-16b, the necessary symmetry is not present due to intervening surfaces. In this case, the card:

```
CC1    CC7
  BCS    BCS2, DUPBCS = BCS1, NINC = 10
```

is used to prevent auto-generation of equivalent form factors.

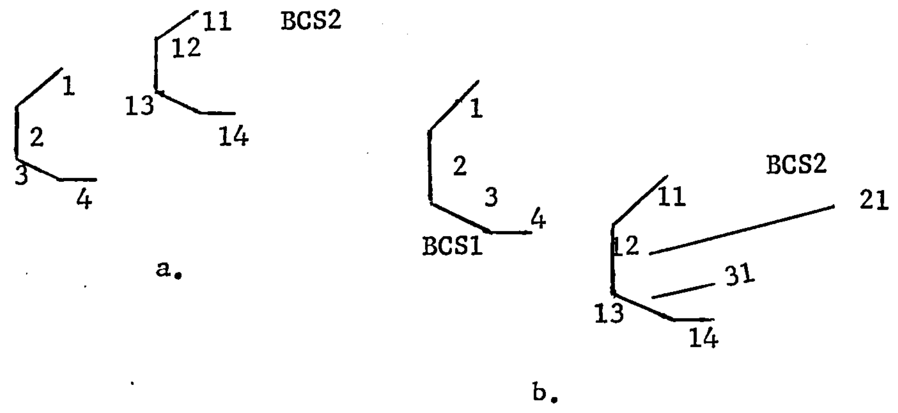


Figure 3-16. Equivalent Form Factors

When the BCS image feature is used, the symmetry shown in figures 3-16a and 3-16b exists. In addition, a second type of symmetry may exist because of the image relationship between the two groups of nodes. In figure 3-17a, the same equivalent form factors as in 3-16a exist. In addition, note that F1-12 = F11-2, F1-13 = F11-3, and so on for the form factors "looking" across the reflecting plane. In this case, the appropriate BCS card is:

```
CC1     CC7
  BCS     BCS1, IMGBCS = BCS1, NINC = 10, IGEN = ALL
```

Note that the image of BCS1 is placed in BCS1. This is the only way that the "ALL" equivalence situation can be guaranteed, so if BCS2 = BCS1 and IGEN = ALL, the program will nevertheless set IGEN = PART.

When surfaces intervene between the two imaged groups, the second type of symmetry is not present (figure 3-17b) and the appropriate card is:

```
CC1    CC7
  BCS     BCS2,IMGBCS = BCS1,NINC = 10, IGEN = PART
```

If BCS1 and BCS2 are not defined identically in the BCS data block, or one is moved relative to the other by a CHGBLK call, the second type of symmetry again does not exist (figure 3-17c) and the card appropriate for figure 3-17b should be used.

Intervening surfaces located within one or the other set of nodes destroys all symmetry (figure 3-17d) and the following card must be used:

```
CC1    CC7
  BCS     BCS2,  IMGBCS = BCS1, NINC = 10
```

Please note that if the IGEN variable does not appear on the BCS card, no equivalent form factor generation will be done; thus, the program is still compatible with input decks created before this feature was incorporated.

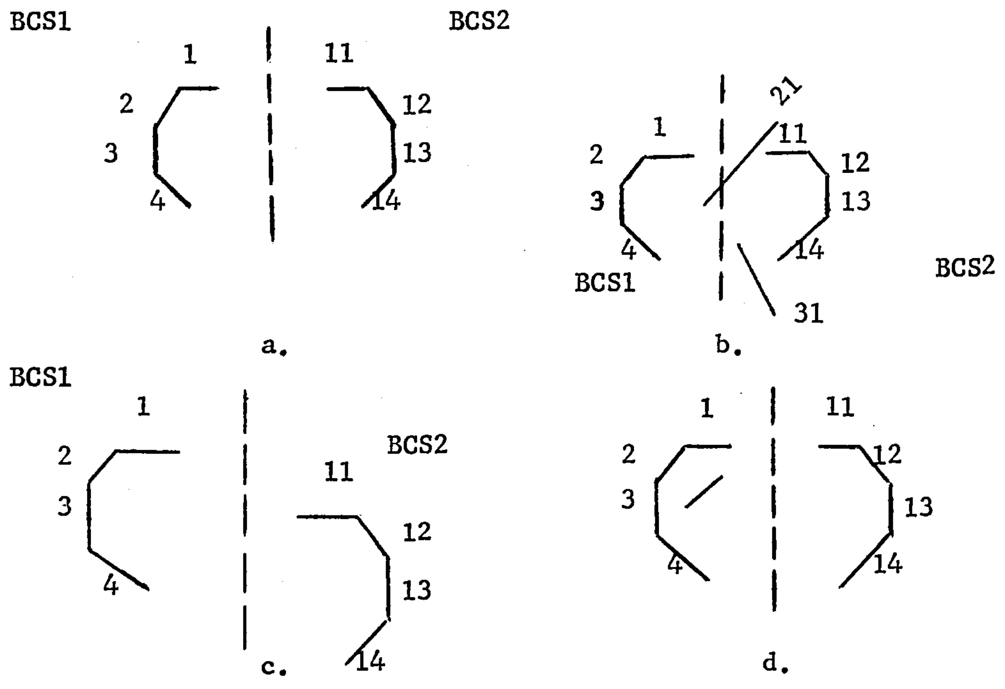


Figure 3-17. Equivalent Form Factors

3.3.3.12 Shadower-Only Surfaces

In many radiation-dominated thermal analysis problems, there are surfaces which are so remote from the region of interest that they do not actively enter into the radiation network. These surfaces, however, block incoming radiation and the view to space for the nodes of interest. The use of shadower-only surfaces permits the user to account for this blockage without increasing the complexity of his problem in the region of interest.

The following rules apply in the use of shadower-only surfaces:

a) Shadower-only surfaces provide blockage in both the FF and the DI segments. They appear no where in the program output of the User's problems, however.

b) Form factors from node i to both sides of shadower-only surfaces are summed and added to F_{ii} to conserve energy (infers $T_{shadowers} = T_i$).

Flag IFFSHO can be set equal to 2HNO in the Quantities Data Block or in the Operations Data Block to bypass the calculation of form factors to shadower-only surfaces. IFFSHO defaults to 3HYES.

c) Because these surfaces are not active in the problem, neither the active side nor the surface optical properties need be input.

d) Shadower-only surfaces are specified by setting the shade flag; SHADE = ONLY.

e) Shadower-only surfaces must be added after all active surfaces. It is recommended that shadower-only surfaces be input in separate BCS(s) and that these BCS(s) be added last in the BUILD-ADD sequence. (Reference Appendix D, Pg D-2, D-6.)

f) Shadower-only surfaces may appear or not appear in node plots, at the User's option. This is controlled by the ISHO argument in subroutine NDATA (reference Appendix D).

3.3.4 BCS Data

Each block coordinate system named in the surface data block must be defined in the BCS data block. An exception is the default BCS "ALLBLK" where all surfaces in the Surface Data block that are not preceeded by a "B" CARD (see para. 3.3.3.9.1) are assumed to be a part of BCS "ALLBLK". BCS "ALLBLK" is assumed always to be coincident with the CCS. If all surfaces defined in the surface data block are in BCS "ALLBLK" and the assumed coincidence with the BCS system is acceptable then No Header BCS Data block is required. If the user wishes to change the position and/or orientation of BCS "ALLBLK" it can be accomplished as if it was a BCS named by the user. Block coordinate systems are defined according to the following formats:

```
    CC1        CC7                                          CC72

     B           BCSNAM, TX, TY, TZ, ROTX, ROTY, ROTZ
```

If the rotations are not to be performed in the standard x, y, z, order, the hybrid format:

```
    CC1        CC7                                          CC72

     B           BCSNAM, TX, TY, TZ, ROTZ = DV,
                 ROTX = DV, ROTY, = DV
```

may be used. Rotations in this example are performed first about Z, then X, then Y. Arithmetic expressions can be used for data values (reference Section 2.3).

The variable names entered in the BCS data block are defined in Table 3-IV. Note that these definitions are almost identical to the ICS and position data of the surface data block, even to variable names. This creates no ambiguity, because the position and ICS variables are used in the preprocessor only, and unlike the BCS variables, are not addressable from the processor routines. In common with position and ICS data, these translation and rotation variables appear to translate the CCS into the BCS. The new user should refer to para. 3.3.3.9.11 until defining this transformation becomes automatic.

VARIABLE NAME	RANGE OR OPTIONS	DEFAULT VALUE	DESCRIPTION
BCSN	ANY 6 CHARACTER NAME	NONE	BLOCK COORDINATE SYSTEM NAME
TX	N/A	0.0	TRANSLATION DISTANCE FROM ORIGIN OF CCS TO ORIGIN OF BCS, MEASURED ALONG CCS X-AXIS.
TY	N/A	0.0	SAME AS TX, EXCEPT ALONG Y-AXIS
TZ	N/A	0.0	SAME AS TX, EXCEPT ALONG Z-AXIS
ROTX	$-360. < \text{ROTX} \leq 360.$	0.0	ROTATION ANGLE TO ROTATE CCS INTO BCS; ROTATES ABOUT CCS X-AXIS, Y TOWARD Z POSITIVE.
ROTY	$-360. < \text{ROTY} \leq 360.$	0.0	SAME AS ROTX, EXCEPT ROTATES ABOUT Y, Z TOWARD X IS POSITIVE.
ROTZ	$-360 < \text{ROTZ} \leq 360.$	0.0	SAME AS ROTX, EXCEPT ROTATES ABOUT Z, X TOWARD Y IS POSITIVE.

Table 3-IV BCS Data Input Detail

3.3.5 Form Factor Data

The Form Factor Data block provides a punched card and/or CMERG tape (see Section 3.2.2.3) BCD formatted entry point, so that the user can control the form factor data computations performed when either the FFCAL or NFCCAL links are called in the Operations Data Block.

A Form Factor Data block is required when:

1. With or without an RSI tape the user wants to by-pass the computations of some form factor * area (FA) products (or data which is stored on the RSI tape), and instead utilize hand input data values, or selected values on a CMERG tape from a previous run that the user has determined to be applicable.

2. The user wants to override some of the data values on the RSI tape and have the program recompute new values, perhaps because of a need to specify a tighter form factor accuracy criteria in subroutines FFDATA or NFDATA.

A Form Factor Data Block is not required for normal restart operations when all the form factor area (FA) products already stored on the RSI tape are applicable to the configuration being run.

The user may override the check the program normally makes to see if the active node array matches the node array for the FA matrix stored on the RSI tape. This is accomplished with the argument FFNAC in subroutines FFDATA or NFDATA. Overriding this check in the program will allow the user to:

1. Remove nodes/surfaces anywhere by changing the Surface Data block and/or Operations Data block. The program will delete the directly related FA data and will arrive at utilizing a reduced FA matrix. This reduction may be done with or without a Form Factor Data block. If in the surfaces removed there are surfaces which had affected the remaining stored values, i.e. blockers, then it is up to the user to utilize a Form Factor Data block to selectively set or recompute those data values significantly impacted by the model change.

2. Add nodes/surfaces with unique identification numbers by changing the Surface Data block and/or Operations Data block. This can be done with or without a Form Factor Data block. Without a Form Factor Data block the program will utilize all the values previously computed and stored on the RSI tape, and will set up a request matrix to compute the remaining FA data values. As was the case with deletions it is left up to the user to decide if a form factor data block is required to correct or recompute some of the data values read in from the RSI tape.

A caution must be observed when attempting to recover FA data on a restart tape when the total number of nodes is being reduced. This is illustrated by the following example. The original run computed form factors for a 300 node model, including a large cylinder subdivided into nodes. In the run now being prepared, 20 new nodes are being added, but the detail in the area of the cylinder is not required and the surface data defining the cylinder will be edited to reduce its number of nodes to one. Thus, the new configuration being defined will consist of 273 nodes. This is an acceptable procedure. The problem that crops up is that the maximum possible number of active nodes, as determined by analysis of the surface data block is 273 and the node data storage arrays will be sized accordingly. When the FA products from the 300 node model are read from the RSI tape, there will be insufficient core to receive them and the run will terminate on an error. The user must recognize this situation, if it is being created, and use the variable NNMIN in the Options Data block to provide for it. (See Table 3-1: Options data input detail). For the hypothetical case described, the card:

```
CC7
    NNMIN = 300
```

must appear in the Options Data Block.

A CMERG tape with the FA data in an acceptable input format for the form factor data block can be generated from a TRASYS run by specifying the tape option in subroutine FFDATA or NFDATA (see Sections 4.3.3.1 or 4.3.3.2). The correct implementation of this option will cause the FA products that are computed and/or read in (normally written only to the RSO tape) to be written to the BCD formatted USER1 tape.

From the standpoint of TRASYS operations, information in the form factor data block is used by the preprocessor to define two form factor request matrices. If a form factor data block is not encountered in the input stream, the form factor request matrices default everywhere to -1.0. The two request matrices provide for both the solar and infrared wavebands. They are identical unless semi-transparent surfaces with different infrared and solar transmissivities are present.

A form factor request matrix is a triangular matrix of the same form as a form factor matrix. It is used for detail direction of form factor computations, and finally for storage of the form factors. This is done as follows: prior to computing each form factor, the corresponding value in the request matrix is examined; if it is zero or greater than zero, it is presumed to be a valid form factor and is left in the matrix unchanged; if less than zero, the form factor is computed and stored in place of the negative number.

The user should never attempt to try and use FA data on a restart tape if it is necessary to change the node numbers from those associated with the FA data when it was originally generated. In the process of setting up the request matrices, the node array on the restart tape is compared, node by node, with the currently active node array. When a difference is encountered, the "new" number in the currently active node array will override and -1.0 values will be placed in the request matrices so that the FA products to/from the "new" node will be computed. The correct way to utilize the data on the restart tape is to retain all the old numbers for which valid FA data exists on the restart tape, then use the Correspondence Data block to change node numbers for the final output.

Prior to any form factor calculations, the following operations are performed on the request matrix, in the order indicated.

1. Matrix set everywhere to -1.0.
2. Matrix overwritten with all form factor * area (FA) products on RSI/RTI tapes.
3. Values of 0.0 set per ZERO cards.
4. Values of 0.0 set per ONLY cards.
5. Values of -1.0 set per RECOMP cards (overrides values on restart tape).
6. Set individual FA values per form factor data cards.

Operations 3 through 6 happen only if a form factor data block is present. Note that these operations enable the user to: a) Arbitrarily set all form factors from a given node to zero if it is known to be "out of sight" of the remaining nodes; b) Compute only the form factors from selected nodes, setting all other form factors to zero; c) Recompute FA values known to be in error on the restart tape(s); and d) Set individual FA values to any value desired. If no RSI tape is present, operation 2 of course does not occur and operation 5 is meaningless. The remaining operations occur as listed.

The above discussion applies to a single problem geometry. If more than one geometry exists in a given job, multiple sets of request matrices are involved. The form factor data block provides for definition of as many request matrices as required.

3.3.5.1 Variable Definitions

Form factor data block designators are defined in Table 3-V.

Table 3-V Form Factor Designators Definition

COL. 1 DESIGNATOR VARIABLE NAME	RANGE	DEFAULT VALUE	DESCRIPTION
FIG	Hollerith, 6 Characters, Max.	None	Configuration Name. Must follow Header Card*.
NODEA	1-99999 (Integer Array)	None	Node Identification No. Array. If required.
IR, SOL, BOTH or BLANK	N/A	Both	Indicates data for IR, Solar or Both Request Matrices.
ONLY	N/A	None	Indicates "ONLY" option.
ZERO	N/A	None	Zero's entire matrix.

*FIG Cards immediately follow the header card and also begin each set of form factor data in a multi-configuration run.

3.3.5.2 Form Factor Data Formats

1) FIG cards are entered according to the following format:

```
CC1           CC7                              CC7
                                                 2
  FIG           CNAME
```

2) The NODEA array is entered according to the following format

```
CC1           CC7                              CC7
                                                 2
  NODEA         NN1, NN2, . . . NNN, END
```

for an N-Node matrix. The NODEA array, if required, must follow the FIG card. If either the BCS dup or image option was used in the surface data block, the node numbers must be entered in the same order they appear in the surface data, provided they are under active BCS names. If BCS dup or image was not used, the node numbers must appear in the order that results from the BUILD cards or BUILDC/ADD sequences. For these reasons, the chances of a user-written node number matrix being valid for a large problem are remote. When a node array is needed, use Subroutine FFNDP (Ref. Section 3.3.5.5).

NOTE: The node array is required only when individual form factors are to be defined in the form factor data block, or when equivalent form factors (Section 3.3.5.4) appear. If all records in the FORM FACTOR DATA BLOCK utilize one of the condensed formats pertaining to an entire row and column, then a Node Array is not required.

3) The ZERO card is entered according to the following format:

```
CC1          CC7
 ZERO
```

The ZERO card will put 0.0 into the entire Form Factor * area matrix.

4) The ONLY option is implemented as follows:

```
CC1      CC7                                             CC72
 ONLY      NA, NB, NC, ND . . .
```

Results in <u>only</u> the form factor rows from each node in the list NA, NB, NC, etc. being computed. <u>All</u> other form factors in the matrices will be set to zero. Applies to both wavebands.

5) Single FA products are entered according to the following format:

```
CC1          CC7                                         CC72

 WBAND          NA, NB, DV
                where: NA, NB, and DV correspond
                respectively to I, J and the FA
                product in the expression:
```

$$F_{IJ} A_I = F_{JI} A_J = DV$$

6) Multiple-repeated FA products involving a single node are entered according to the following format:

```
CC1       CC7                                            CC72
 WBAND      NA, NB, NC, DV
```

This will result in an FA product equal to DV being entered in the row of the request matrix corresponding to node NA, for columns corresponding to nodes NB through NC, inclusive.

7) A request to recompute FA products is implemented as follows:

```
CC1      CC7                                          CC7
                                                        2
  WBAND    NA, NB, RECOMP
or
  WBAND    NA, NB, R
```

Recomputes FA product from NA to NB.

```
  WBAND    NA, RECOMP
or
  WBAND      NA, R
```

Recomputes FA products to/from NA from/to all other nodes.

8) To zero FA products by individual node pairs or for all node pairs for a specific node can be implemented as follows:

```
CC1      CC7                                          CC7
                                                        2
  WBAND    NA, NB, 0.0
```

Results in zero value for form factor from NA to NB.

```
CC1      CC7                                          CC7
                                                        2
  WBAND    NA, ZERO
or
  WBAND    NA, Z
```

Results in zero form factors from NA to all other nodes.

3.3.5.3 Form Factor Data Block Example

An example of a form factor data block is presented in Figure 3-17a.

3.3.5.4 Equivalent Form Factors

Many radiation enclosures involve geometry that is symmetric in some manner and may, therefore, have many form factors that are exactly equivalent to other form factors because the node pairs involved are the same size and shape and "see" each other in the same way. The analyst can identify these situations, and if he can conveniently enter this information, a considerable amount of computer time may be saved in form factor computation. This capability has been provided, and the following sections describe the required form factor data block input.

3.3.5.4.1 Equivalence Data Formats

Form factor equivalence data records appear in the form factor data block in the following format:

```
CC1         CC7

  WBAND        NNI, NNJ, NNK, NNL, DV
```

If the FA product DV is absent, the FA product from NNI to NNJ will be computed or obtained from a restart tape.

The first four variables are integer node numbers. The four numbers are interpreted to mean the form factor from NNI to NNJ is equal to the form factor from NNK to NNL. In common with the other form factor data, only one record of five numbers or less with the comma delineators may appear on a card (except for the node array).

In addition equivalent form factors can be generated automatically by the program for duplicated and imaged block coordinate systems. This can be implemented via the "B" card in the surface data block (see Sections 3.3.3.10: Dup and Image Options and Section 3.3.3.11: Equivalent Form Factors).

```
CC1     CC7

HEADER  FORM FACTOR DATA
FIG     AFTEND
NODEA   10,17,20,24,32,END
        10,20,12.4   $  DEFINES FA FROM NODE 10 TO NODE 20 (BOTH BANDS)
IR      10,32,6.2    $  DEFINES FA FROM NODE 10 TO NODE 32 (IR WAVEBAND)
        24,ZERO      $  ZEROS ALL FA'S FROM NODE 24 (BOTH WAVEBANDS)
        10,32,17,20     $  EQUIVALENT FORM FACTORS
ONLY    10,20,24,32     $  COMPUTE ALL FF'S FROM AND TO LIST OF NODES
SOL     10,RECOMP       $  RECOMPUTE SOLAR FORM FACTORS FROM NODE 10
        32,U         $ COMPUTE FFS FROM NODE 32 BY UNIT-SPHERE METHOD
        17,20,U      $ COMPUTE FF FROM 17 TO 20 BY UNIT-SPHERE METHOD
ZERO                 $ ZERO'S ENTIRE MATRIX
```

Figure 3-17a Form Factor Data Examples

3.3.5.5 <u>Selective Unit Sphere Form Factor Computations</u>

In the form factor data block the user has an option to specify that selected form factor computations be performed utilizing the Nusselt Unit Sphere method. This feature is only available for nodes that are planar and nonshadowed. Although its utility is limited it is a very accurate and fast solution. The user may specify this option as follows:

```
CC7
   NA, NB, U
   NA, U
```

In the first example if nodes NA, NB are planar, an unshadowed form factor between nodes NA and NB will be computed by the unit sphere method. In the second example, assuming all planar nodes, the form factors from/to node NA will be computed utilizing the unit sphere method.

These computations would override any corresponding RSI data and require the user to specify the FFCAL link in the Operations Data block.

3.3.5.6 Punching a Node Array--Subroutine FFNDP

Writing a node array for a large complex problem is exceedingly prone to error, yet a node array may be required (reference Section 3.3.5.2) in the form factor data block, shadow factor data block (reference Section 3.3.6) or flux data block (reference Section 3.3.7). Subroutine FFNDP may be used to obtain a node array punched in form factor data format. This may be done in a preliminary run, (when node plots are generated for instance) and the User will then have an error-free node array to use. The operations data calling sequence is:

```
CC7

 CALL FFNDP
```

3.3.6 Shadow Factor Data

3.3.6.1 Basic Concepts

Shadow factors are fractional numbers that describe the amount of shadowing (blockage) encountered by collimated energy incident on a nodal surface. A shadow factor of one indicates no blockage, zero indicates 100 percent blockage. Blockage results from other parts of the spacecraft or from the surface itself, if nonplanar.

Shadow data consists of tables of shadow factors, one table per node. These are 171-point bivariate tables. When the direction to an energy source is specified, using clock and cone angles, (see Figure C1 - Energy Source Direction for Shadow Tables) the clock and cone angles are used as arguments in a double-linear interpolation that returns a shadow factor to be used in computing Solar, Albedo and Planetary direct irradiation according to:

$DI_{shadowed} = SF * DI_{nonshadowed}$

The 171 points result from all combinations of 19 clock angles and nine cone angles, spaced as described in Appendix C.

Precomputing DI shadow factors for a given configuration is an approach that should be utilized with caution, for there are no straight forward ways to define its practical limitations. The concept is to save computer time by minimizing the repetitious or near repetitious computations that normally occur in the direct incident flux for any given node when one considers the look angles to the sun and to each node on the earth for every orbit point evaluated within a given execution, and from one execution to the next. Whether the approach is a practical one for the user, depends upon:

1. How sensitive the critical areas of the model are to accurate heat rate computations. Obviously, for example, if the model is enclosed with external insulation the external surfaces will be considerably

more sensitive than the internal parts where the heat will be reradiated, and stored in the structure and/or removed by an active thermal control system. In this situation if there are no critical external surfaces there is a reasonable chance the model is insensitive to the errors that would be introduced with the application of the shadow tables.

2. The complexity of the shadowing may require smaller look angle (clock and cone) increments than currently used in TRASYS. The linear interpolation between the discrete points in the table may introduce significant errors. The program has a built in safeguard to limit the error whenever the interpolation involves shadow table differences greater than 0.5. In this situation the program will not use the tables, but instead compute the shadowed flux values by using the usual shadow routines in the program's DI segment. This will take greater computer time, and some of the advantage in computational speed with the application of the shadow table will be lost.

3. The error associated with the interpolation also biases the accuracy towards those that would have large unshadowed data values. The accuracy of flux and temperature predictions will be greater as the magnitude of the unshadowed incident heat flux increases.

Note that although the shadow factor data originally was rounded off and carried with one significant decimal place the program no longer packs the data, and instead stores the shadow factors with no loss in accuracy.

The shadow factor data block functions to provide a punched card entry point for shadow factor data that is known in advance, and to direct the updating of existing shadow factor data on a restart tape (RSI).

3.3.6.2 Variable Definitions

VARIABLE NAME	DESCRIPTION
FIG	Configuration Name (Hollerith, 6 characters, max.).
NODEA	Node Number Array
RECOMP	Indicates RECOMP option
TABLE	Indicates a complete shadow table input
WBAND	Indicates energy waveband options: IR, SOL, BOTH. Defaults to BOTH. The shadow factors should be the same for both wavebands except when surfaces with transmissivities greater than zero are utilized.

3.3.6.3 Shadow Data Formats

MODEL and NODEA are entered according to the following formats:

CC1	CC7
FIG	CONF1 $ (any Hollerith name up to 6 characters)
NODEA	DV1, DV2, DV3 --- DVN, END $ (integer node numbers). The user may have the program regenerate the Node Array (see Section 3.3.5.5: Punching a Node Array -- Subroutine FFNDP) in a preliminary run.

Instructions to recompute shadow factor tables for a specified list of nodes are entered as follows:

```
CC1                CC7

  RECOMP             DV1 $ (integer node numbers)

  RECOMP             DV2

     .                 .

     .                 .

     .                 .
```

Note that this input only applies when an RSI tape with shadow data is present.

A complete shadow factor table for one node is entered according to:

```
CC1                CC7

   TABLE             WBAND, NN, C1, DV1, DV2, --- DV19
                                C2, DV1, DV2, --- DV19
                                 "    "    "       "
                                 "    "    "       "
                                 "    "    "       "
                                 "    "    "       "
                                C9, DV1, DV2, --- DV19
```

The mnemonics C1 through C9 refer to the 9 cone angles in a shadow factor table. The 19 data values following are for the 19 clock angles in a shadow factor table. If a cone number mnemonic is omitted, the 19 data values associated with it default to zero (100 percent shadowed). If less than 19 data values are entered following a CX mnemonics, the data values encountered are stored consecutively beginning with Clock 1. For example, if

```
CX, DV1, DV2, DV3 --- DVN (N 19 or less)
```

is encountered, the shadow factors for cone angle X, clock angles 1 through N will be DV1 through DVN. The shadow factors for clock angles N + 1 through 19 will default to zero.

Repeated data values may be entered using the repeat option for array data. For example, the card:

```
CC7

  C6, REPEAT, 0.5, 12, REPEAT, 0., 7
```

will enter shadow factors of 0.5 for clock angles 1 through 12 and 0. for clock angles 13 through 19 in the cone angle 6 array.

The user is referred to the description of the shadow factor table format (Appendix C) for an explanation of the way the clock and the cone angles relate to the energy source/vehicle orientations used in shadow factor generation. If a punched node array is desired to avoid error, subroutine FFNDP may be used (See Section 3.3.5.5: Punching a Node Array - Subroutine FFNDP).

3.3.6.4 Shadow Factor Operations Detail

Examples of shadow factor operations are shown in Figure 3-18.

(a) Shadow Factor Data, No Restart (-RSI-) Tape

```
HEADER SHADOW DATA
FIG AFTEND
NODEA      101, 102, 103, 104, 105, 110, 120, 130, 140, END
C          INFRARED SHADOW TABLE
TABLE      IR, 101, C2, REPEAT, .1, 5, .5, .4, .4, .3
                    C9, REPEAT, .4, 19
C          SHADOW TABLE FOR BOTH WAVEBANDS
TABLE      105, C1, REPEAT, 4, .5, REPEAT, 15, 0.
```

(b) Shadow Factor Data With Restart Tape

```
HEADER SHADOW DATA
FIG        PLOAD
NODEA      10, 21, 22, 23, 24, 25, 31, 32, 33, 70, END
C          RECOMPUTE DIRECTION
RECOMP 21
RECOMP 24
C          SHADOW DATA TO OVERRIDE DATA ON -RSI-
TABLE 31, C6, REPEAT, 19, 0.
TABLE 32, C6, REPEAT, 19, 0.
```

Figure 3-18 Shadow Factor Operations Detail

3.3.7 <u>Flux Data</u>

3.3.7.1 <u>Basic Concepts</u>

The Flux Data block provides a punched card and/or CMERG tape (see Section 3.2.2.3) entry point so that the user can selectively control the Direct Incident (DI's) Flux data computations. If an RSO tape is assigned while the DICAL segment or DIREAD subroutine is executed the DI's will be written to the RSO tape and can be read if that tape is assigned as an RSI tape on a subsequent run. No Flux Data block is required for a restart run if no changes have been made in the number of active nodes, their numbers and sequence and the user desires to:

1. Use the previously stored data on the RSI tape in its entirety or

2. With the utilization of CAll RSTOFF and CALL RSTON (see Section: 3.3.9.7) in the Operations Data block the Flux data stored in specific step(s) can be selectively read or by-passed as complete step(s).

A flux data block is required when:

1. The user wishes to recompute some of the data values stored on a valid RSI tape. The reason for wanting to do this might be that a better DI accuracy needs to be specified by the user in subroutine DIDT1 (see Section 4.3.5.4) or the geometry was changed some because one or more node(s) were moved.

2. The user wishes to set specific flux data values to override the associated data values read from a valid RSI tape and by-pass the program computions normally performed.

3. The node array stored on the RSI tape for a given configuration name will not match the node array of the active configuration created by the "BUILD" card (see Section 3.3.9.4). This would occur if after the user created the RSI tape, changes were made to the model that changed the number of nodes, or the node numbers, or the sequence.

If the user can determine that a significant portion of the data is still valid, the data can best be retrieved by utilizing the appropriate CMERG tape as a means of selective data entry in the Flux Data block. A suitable CMERG tape with the data in an acceptable input format for direct input to the Flux Data block can be created by specifying the tape option in subroutines DIDT1 or DIDT1S (see Section 4.3.5.4). The appropriate implementation of this option will cause the DI's computed and/or read in from the RSI tape to be written to the USER1 tape on a preliminary run. This former USER1 output tape can be used as a CMERG tape on subsequent runs, to allow the user to selectively input data in the flux data block (see Section 3.2.2.3). After the last write to the USER1 tape the user should include in the operations data block an END FILE NUSER1 and a CALL LIST with the appropriate user furnished arguments (see Appendix D). The LIST subroutine will list all the data stored on the USER1 tape and number each record (line) so the line counts will be known for CMERG edit commands.

From the standpoint of TRASYS operations, information in the flux data block is used to define the flux data request matrix. If a flux data block is not encountered in the input stream, the flux request matrix is set everywhere to -1.0. If it is desired to force the recomputation of fluxes, (overriding flux data on a restart tape, for instance), flux data values equal to -1.0 are entered for the appropriate nodes.

The variables NODEA and STEPN are required in addition to the flux data. These variables allow the input data to be stored according to the proper node numbers and points in orbit (step numbers). If a punched node array is desired to avoid errors, use subroutine FFNDP (Reference 3.3.5.5: Subroutine FFNDP).

3.3.7.2 Variable Definitions

VARIABLE NAME	DESCRIPTION
NODEA	Node identification number array
INITL	Value that fluxes may be initialized to (Optional) The input value for INITL will override the RSI data values unless set to -2.0
STEPN	Step number that following flux data applies to

A FIG card is not used in the flux data block because step numbers are the significant label for flux data storage.

3.3.7.3 Flux Data Formats

1) NODEA, STEPN, and INITL are entered for each step according to the following order and formats:

```
CC1       CC7                            CC7
                                           2
  NODEA     NN1, NN2, ---, NNN, END
  STEPN     NDV
  INITL     DV
```

The user may have the program generate the node array (see section 3.3.5.5) in a preliminary run.

2) Flux values may be entered in either of two formats. The quadruplet format is as follows:

```
CC7                                      CC7
                                           2
  NODID, DV1, DV2, DV3

  101,232., 114., 99.*.317 $ example
```

where:

NODID = node identification number

DV1 = incident solar flux

DV2 = incident albedo flux

DV3 = incident planetary infrared flux

Restrictions:

One quadruplet only per card.

All four data values are required. No default logic applies.

Flux values must be in TRASYS standard units (Btu/hr-ft^2).

The single value formats are as follows:

```
CC7                                                      CC7
                                                           2
  NODE = NDV1, SUN = DV2, ALB = DV3, PLAN = DV4
or
  NDV1, S = DV2, A = DV3, P = DV4
```

Where:

NDV1 = node identification number (integer)

DV2 = incident solar flux

DV3 = incident albedo flux

DV4 = planetary infrared flux

Restrictions:

All data values encountered between NODE values pertain to the preceding node number.

3.3.7.4 Flux Data Block Example

An example of a flux data block is shown in Figure 3-19.

```
HEADER      FLUX DATA
NODEA           10, 17, 20, 24, 32, END
STEPN           10
                10, 2.79 * 144., 0., 0. $ FLUX ON SUN-ORIENTED SURFACE
                17, 0., 25.74, 14.8
                NODE = 20, SUN = 0., ALB = 25.74, PLAN = 14.8

NODEA           10, 17, 20, 24, 32, END
STEPN           11
                32, -1., -1., -1. $ RECOMPUTE FLUXES FOR NODE 32, STEP 11
```

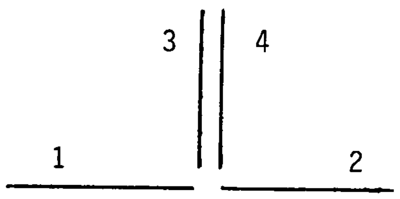
Figure 3-19 Flux Data Block Example

3.3.8 Correspondence Data

3.3.8.1 Basic Concepts

The correspondence data block performs the function of providing the user with an input point for the node numbering data necessary to make his thermal radiation model correspond on a one-to-one basis with his thermal analyzer RC (Resistance Capacitance) model.

The user has the choice of using the information in this block in the form factor segment to combine form factors or waiting to use it in the absorbed heat output (QOCAL) and radiation exchange output (RKCAL, RCCAL) processor segments. Two sources of error arise with form factor combining, however. If surfaces with differing optical properties are combined the radiation interchange computations are approximate because they proceed on the basis of area-weighted average optical properties. The second source of error is the so-called "fence" problem. Consider the 4-node configuration:



Clearly, form factors between nodes 4 and 1 and 3 and 2 do not exist. If nodes 1 and 2 were combined, however, form factors $F_{3-1,2}$ and $F_{4-1,2}$ would exist, and GBCAL calculations would produce a radiation interchange factor between 3 and 4, obviously incorrect. Obviously this can cause erroneous absorbed heating computations also.

The user may avoid these situations by splitting his correspondence data. One set of correspondence data, with no optical property combination or fence problems, is entered and used in form factor combining. The remaining correspondence data necessary to make the TRASYS model agree with the RC model is entered separately, and will be applied only in the absorbed heat and radiation interchange output segments. This assumes the user can identify correctly all of the potential errors inherent with form factor combining.

The from factor combining capability was developed to minimize the partitioning for the matrix inversion in the gray body (GB) segment, by reducing the matrix size, and thus the I/O cost associated with partitioning in the original version. Subsequent development and experience has shown that it has outlived its usefulness even for models as large as 800 nodes, especially when one considers the pitfalls associated with form factor combining. The program was originally designed to operate in 65000K decimal core. For the gray body segment the program dynamically assigns all of the available core to the matrix inversion solution. When the number of nodes exceeds between 200 and 250 for 65K core the matrix must be partitioned on a disc file with only a portion resident in core. Additional core can be utilized to eliminate or further reduce the degree of partitioning. Even when partitioning is necessary two improvements to the original program have eliminated, even for the 700 to 800 node models currently considered, most of the advantages of form factor combining. The most significant improvement to the Univac version was to replace the inefficient standard Fortran I/O operations involved with the partitioning with a system utility routine (NTRAN) that will transfer an entire array so that each operation requires one I/O transfer with no restrictions on block size. In addition the program was modified so that the partitioned matrix inversion can be restarted at intermediate stages if RTO/RTI tapes are used. This feature minimizes the losses if the matrix solution is not complete due to maximum time, etc. The form factor combining capability will be retained because it is feasible that as models grow, form factor combining may again become an attractive alternative.

Another set of correspondence data, not entered by the user, but implicit to the program, is the correspondence data file that automatically recombines polygons. The user has control of whether or not this is done in the CMCAL link (reference Section 5.11.). Another automatic application of correspondence is the optional capability to apply the equivalent correspondence generated for Auto BCS DUP and Auto BCS IMAGE Options (reference Section 3.3.3.10.3 and 3.3.3.10.4). Again the user has control of whether this is accomplished in the CMCAL link. To utilize this option the user should refer to Section 3.3.8.5: Automatic Generation of Correspondence Data.

3.3.8.2 Variable Definitions

Correspondence data block variables are defined in Table 3-VI.

VARIABLE NAME	RANGE OR OPTIONS	DEFAULT VALUE	DESCRIPTION
FIG	CNAME	NONE	CONFIGURATION NAME
CF	BLANK OR FF	BLANK	FLAG DESIGNATING TYPE OF CORRESPONDENCE DATA OPTIONS: FF: CORRESPONDENCE DATA WILL BE APPLIED TO FF's IN CM LINK BLANK: CORRESPONDENCE DATA WILL BE APPLIED IN THE QO LINKS AND RADIATION CONDUCTOR LINKS

Table 3-VI Correspondence Data Variable Definition

3.3.8.3 Correspondence Data Formats

1) FIG cards are entered according to the following format:

```
CC1             CC7                        CC7
                                             2
  FIG             CNAME, CF
```

2) Node correspondence data is entered according to the following format:

```
CC7
  NODID = DV1, DV2, --- DVN
```

NODID is an RC model node number (one to six digits, integer) and DV1 through DVN are the TRASYS node numbers that will be combined into node NODID.

3.3.8.4 Correspondence Data Block Structure

The correspondence data found between a card defining a configuration name and the next configuration name definition card can be thought of as a correspondence data sub-block. One of these sub-blocks is required for each unique geometry of the User's problem, assuming node combine operations are required for each geometry.

3.3.8.5 Automatic Generation of Correspondence Data

BCS duping and imaging frequently leads to a situation where a good deal of correspondence data can be automatically generated. Code to do this is now in TRASYS, and the feature is exercised by a new correspondence data block input. The expanded correspondence data input formats are:

NRC1 = N1, N2----NN ' NRC2, NINC2, NRC2, NINC3 ----

or:

NRC1 = N1, N2---NN ' NRC2, BCSN2 ' NRC3, BCSN3 ----

where:

NRC1 = the RC (resistance capacitance) model node number desired for the combination of TRASYS nodes N1 through NN.

NRC2 = the RC model node number desired for the combination of nodes N1 + NINC2 through NN + NINC2.

BCSN2 is the BCS name used the first time the BCS containing N1 through NN was duplicated or imaged. BCSN3 is the BCS name for the second dup or image, and so on.

NINC2 is the increment value specified on the BCS2 card; NINC3 is the increment on the BCS3 card, and so on.

Note that the two inputs are equivalent in function. The first format, using the increment value, is recommended. In the second format the BCS names stated are checked against the list of BCS names that appear in the surface data and the increments used on the appropriate BCS cards is then applied. If a BCS name match is not found, an error is generated. If the increment is specified, per the first option, no check can be made. The increment option must be used, however, when the BCS containing nodes N1 thru NN is not a "root" BCS, that is N1 thru NN were created by a BCS dup or image operation. The following are examples of implementing the correspondence data generation feature.

Referring to figure 3-20a, assume that the user wants to combine nodes 1, 2, and 3. Quite likely, nodes 11, 12, 13, and 21, 22, and 23 should also be combined. Normally this would require the following three cards in the correspondence data block:

```
CC7
  101 = 1, 2, 3
  111 = 11, 12, 13
  121 = 21, 22, 23
```

Either of the following single cards is now equivalent to the above three:

```
CC7
  101 = 1, 2, 3' 111, 10' 121, 20
  101 = 1, 2, 3' 111, BCS 2' 121, BCS3
```

where nodes 1, 2, and 3 were duped/imaged under BCS2 and again duped/imaged under BCS3. Note that each time nodes 1, 2, and 3 are duped and/or imaged, another apostrophe delineator is required.

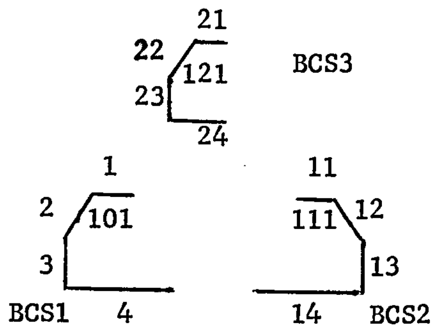


Figure 3-20a. Correspondence Data Generation

In figure 3-20b nodes 1, 2, and 3 were defined under BCS1 and duped within the same BCS as 11, 12, and 13. BCS1 was then imaged in its entirety, creating nodes 21, 22, 23, and 31, 32, and 33. In this case, automatic correspondence data must be generated by:

```
CC7
  101 = 1, 2, 3' 111, 10' 121, 20' 131, 30
```

in the correspondence data block. The more complex "family tree" involved precludes the use of the BCS name option.

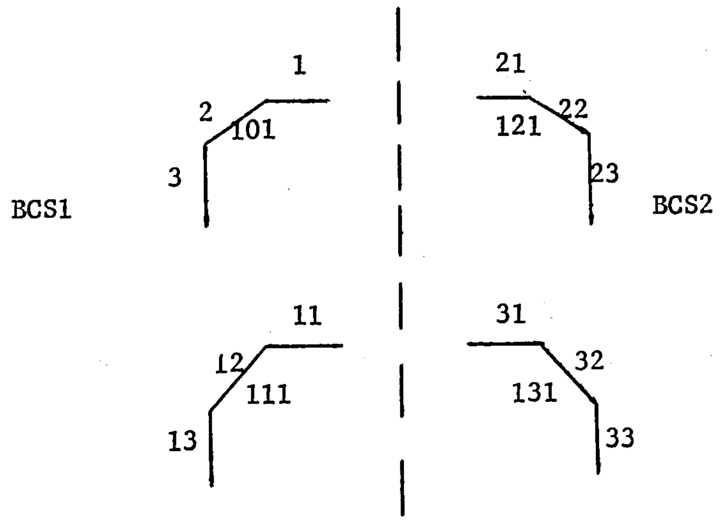


Figure 3-20b. Correspondence Data Generation

```
CC1     CC7 '

  HEADER   CORRESPONDENCE DATA

  FIG      YSIDE   $   (NOT TO BE COMBINED IN CMCAL)
           50 = 10,17
           32 = 40,41,42
            6 = 61,62,7 = 71,74
           18 = 1,11'1018,20  $  DUP OR IMAGE AUTO-CORRESPONDENCE
  FIG      YSIDE   $   FFS TO BE COMBINED IN CMCAL
           25 = 2,4,6
           35 = 3,5,7
```

Figure 3-21 Correspondence Data Block Example

3.3.9 Operations Data Block

3.3.9.1 Basic Concepts

The operations block can be thought of as a digital computer program coded in a somewhat modified FORTRAN language. The most powerful statements in the block are calls to processor library subroutines followed by "link" calls to primary processor program segments. Interspersed with these statements might be FORTRAN statements used to redefine any of the program variables in the reserve name list (see Appendix A) or control constant list, calls to user-supplied routines in the subroutines block, and any branching statements required for direction of problem solution logic. The operations data block is converted by the preprocessor to subroutine ODPROG. This routine serves as the driver for processor execution. In general, the conversion is a one-to-one passover of FORTRAN statments. The segment execution calls (L cards) however, result in the operating system dependent language necessary to define an overlay execution.

An operations data block for a problem involving one or more configurations of the model and/or more than one point in orbit consists of a series of modified FORTRAN statements which are divided into logical sections each of which begins with the definition of a new problem geometry (identified by a unique configuration name) and/or a new point in orbit (identified by a unique STEP number). Calculated data required for subsequent processor operations (with the exception of fluxes output by DICAL, DRCAL and AQCAL) is placed in out-of-core storage under the appropriate configuration name. (Reference Figure 3-22). Fluxes are placed in out-of-core storage under the STEP number that identifies the appropriate point in orbit.

The operations data block logic is processed in the order encountered. Each logical section (as defined above) must be serially executable, that is, no branching from one section to another is allowed.

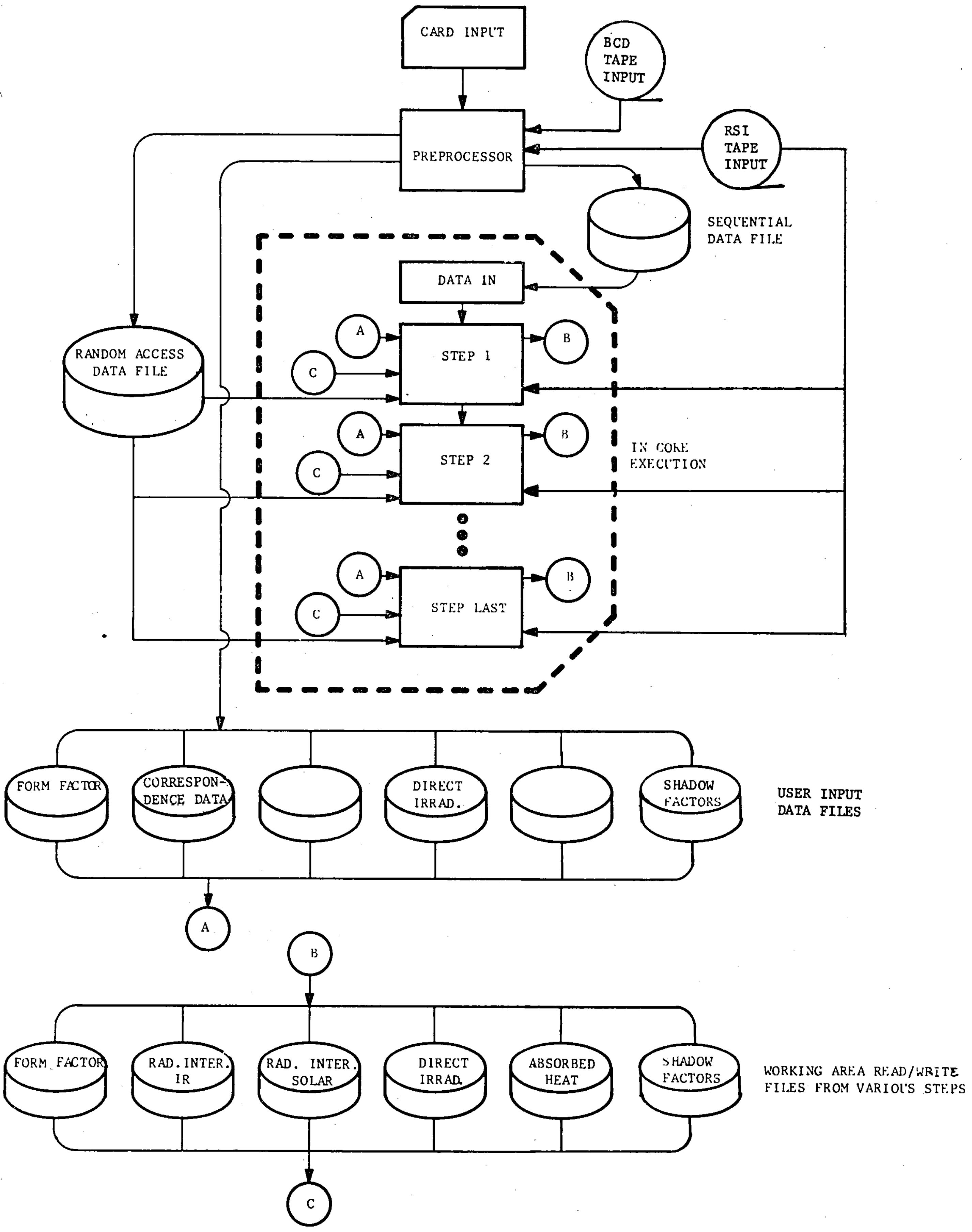


Figure 3-22 Program Data Storage Scheme

DO-loops may be used, but they must be located entirely within a section, that is, BUILDC/ADD cards and STEP cards must not be contained within a DO-loop. In addition, L-cards must not fall within a DO-loop because the indices are lost when the ODPROG segment is overlaid and removed from core. Also, multiple executions of any program segments other than NPLOT, OPLOT, or PLOT within a section will make later data retrieval impossible for that section. Statement numbers from 1 to 9999 may be used, and each statement number must be unique in the operations data block. All program control constants and variables in common at execution of the operations data block (subroutine ODPROG) may be found in Appendix A. This list is automatically extended by the program to contain any constants and arrays entered in the quantities and array data blocks.

3.3.9.2 ORBGEN Option

Writing an operations data block for the calculation of direct irradiation and absorbed heats for an extensive series of points in orbit can be a tedious, repetitive job. To alleviate this, the ORBGEN option is available. When an ORBGEN card is encountered in the operations data block, a package of preprocessor routines use the data on the card to generate the operations data code necessary to compute and/or read direct irradiation, absorbed heats, and print a set of heat rate vs time tables and integrated average heat rates in standard SINDA input formats. These tables may be thought of as a default output that prevents loss of computed data. All flux and absorbed heat data computed using the generated code is stored in the usual manner and may be retrieved, manipulated, and output in any way the user desires (see Section 3.3.7: Flux Data). More than one ORBGEN card can be used in the Operations Data Block. The initial Step Number is incremented by 500 each time an ORBGEN card is encountered.

The preprocessor generated card images created by the ORBGEN will be written to the first RSO file along with the rest of the input data, and given edit numbers. This will allow the user to edit the standard ORBGEN output to accomplish a non-standard operation. For example, through the Edit Data block specific steps of the direct incident data from a previous ORBGEN run can be selectively utilized via calls to subroutine RSTOFF and RSTON. Only the data not common to both runs would have to be computed. NOTE: Editing ORBGEN output is a feature not yet available on the CDC version of TRASYS.

ORBGEN cards are defined as follows:

Format

```
CC1               CC7
  ORBGEN            TYPE, TRUANI, TRUANF, NPT, IFO
```

Definitions

TYPE is a Hollerith variable defining the spacecraft orientation reference and pertinent orbit characteristics. Options are:

INER: Spacecraft is in planetary orbit, inertial (sun or star) oriented.

PLAN: Spacecraft is planet oriented.

CIRP: Spacecraft is planet oriented, in a circular orbit.

NOPL: Spacecraft is in a heliocentric orbit (no planet).

TRUANI is the true anomaly* at the first point in orbit 0 TRUANI 360.

TRUANF is the true anomaly at the final point in orbit. If TRUANF = TRUANI + 360, data for a complete orbit will be generated.

NPT is the number of equal true anomaly increments between the points for which fluxes and direct irradiation will be computed. If the planet shadow is not encountered by the orbit, NPT + 1 points will be computed. If the planet shadow is encountered, NPT + 5 points will be computed, thus describing the flux discontinuities at the planet shadow points.

*Reference Figure 4-7

IFO defines the Optional Input/Output Subroutines. Options for IFO are AQ, DI, DIR, ZEROI and ZEROS. If AQ, the incident and absorbed fluxes are computed. If DI, the incident fluxes only are computed. If ZEROI, all absorbed infrared flux values will be zero. If ZEROS, absorbed solar fluxes will be zero. If DIR, the incident fluxes will be read in directly from a RSI (restart) tape, and absorbed fluxes will be computed. The DIR option requires a complete set of flux values on the restart tape, because the DICAL segment is not loaded.

Options and Restrictions

1) Prior to entering an ORBGEN card, the orbit must be defined through a call to ORBIT1 or ORBIT2.

2) Orientation must be defined through a call to ORIENT.

3) Spin must be defined, if applicable, through subroutine SPIN. If spin is not zero, INER and CIRP are not allowable for TYPE.

4) Problem geometry must be defined prior to any ORBGEN card.

5) Punch/tape flags and accuracy parameters must be defined through subroutine DIDT1 or DIDT1S prior to an ORBGEN card.

6) The QO segment is limited to 100 time-points.

7) Multiple ORBGEN cards are permissible.

3.3.9.3 TRJGEN Option

TRJGEN cards provide a function similar to ORBGEN when computing fluxes and absorbed heats to match time points on a NASA/JSC MPAD trajectory tape. TRJGEN cards are used in the same way as ORBGEN cards and are defined as follows:

```
CC1           CC7
  TRJGEN        TSTART, ISTRRC, TSTOP, ISTPRC, ISKIP, IPLOT, IFD
```

Definitions

TSTART -	First time value on MPAD tape at which calculations are required.
ISTRRC -	Relative record flag. Some MPAD tapes leave more than one record per time point. If the first or only record per time point is required, enter 1, if the second, enter 2, and so on.
TSTOP -	Last time point for calculations.
ISTPRC -	Relative record flag for TSTOP. Enter relative record number desired for TSTOP time point.
ISKIP -	Number of time points on MPAD tape to skip between calculation time points. (Enter zero or any positive integer). A 0 will compute every time point.
IPLOT -	Orbit plot control flag. YES for plots <u>and</u> fluxes NO for fluxes only ONLY for plots only
IFD -	Calculation control flag. Same options and function as IFO on ORBGEN card.

Options and Restrictions

1) ORBIT1 or ORBIT2 must be called prior to TRJGEN cards with zeros in all but the first argument to define SOL, WSS & WDS.

2) ORIENT and SPIN calls are not applicable.

3) TRAJ must appear in the options data block.

4) Multiple TRJGEN cards are permissible.

It is recommended that the user obtain a dump of the trajectory tape contents for guidance in setting up TRJGEN cards. TRASYS is limited to 100 time points per TRJGEN card.

3.3.9.4 BUILD Option

This option provides the user the convenience of defining the geometry for a configuration (in terms of block coordinate system names) with a single card rather than through a series of user calls to subroutine BUILDC and ADD, as previously required by TRASYS I.

Format

```
CC1            CC7                                          CC7
                                                              2
  BUILD          FIG, BLK1, BLK2, BLK3
```

This is equivalent to the sequence:

```
               CC7
                 CALL BUILDC (BLK1, 3HFIG)
                 CALL ADD (BLK2)
                 CALL ADD (BLK3)
```

Note that the configuration name (FIG in the example) must begin to the right of card column 6. The BUILD option may continue for as many cards as required to list all of the Block Coordinate System (BCS) names that make up the desired configuration. A continuation flag is required when more than one card is needed. A BCS name should not be split between cards.

A BUILD card with columns 7-72 blank will automatically default the Configuration name to the Model name given in the Option Data Block and all BCSs specified in the Surface Data Block will automatically be listed to comprise the configuration. If there is no Model name in the Options Data Block, then the Model name will default to THING.

If the configuration name is the same as a BCS name and the BCS name does not exist in the configuration the preprocessor will flag the configuration name as an error. For example, if in the Surface Data Block there are BCSs with the names NEW, BLK1, BLK2, and it is desired to build a configuration in the Operations Data Block with only BLK1 and BLK2 this configuration cannot be called NEW.

3.3.9.5 Operations Block Formats

1) Step number cards are punched according to the following format

```
CC1          CC7                                   CC7
                                                     2
  STEP         DV (integer)
```

where DV is the integer step number, with the allowable range 1 to 9999. Positive step numbers are required in the Operations Data Block for only the DI and DR computation links. They must be specified by the user for each orbit point; or if ORBGEN or TRJGEN is used, they will be specified by the program. For other than DI and DR computations, the user may use positive step numbers as in TRASYS I. If they are omitted the program will insert negative step numbers only where required. This eliminates the confusion TRASYS I users previously had on what constitutes a Step. The elimination of user step numbers was possible by replacing it with the more meaningful label, the configuration name.

2) Subroutine calls are made in the classic FORTRAN format, with the word CALL beginning in CC7. Calling sequences for each user-accessible processor routine can be found in Appendix D.

3) Computation segment (link) calls are made using the following format:

```
CC1        CC7                              CC7
                                              2
 L            SEGNAM
```

Where SEGNAM is the name of one of the program segments contained in the processor library. Appendix E describes the processor segments and their functions. Currently allowable options for SEGNAM are: NPLOT, OPLOT, SFCAL, FFCAL, NFFCAL, RBCAL, CMCAL, DICAL, DRCAL, GBCAL, RKCAL, RCCAL, AQCAL, QOCAL, and PLOT.

3.3.9.6 Operations Block Examples

Operations block structure and function is illustrated by the listings of sample operations blocks in Figure 3-23.

Sample 1 of Figure 3-23 is a single step operations block that generates three node plots of a single geometry. Sample 2 is a two-step operations block that generates form factor matrices for two geometric configurations. Sample 3 is a 17-step operations block that generates direct irradiation data at 16 points in orbit, including the planet shadow in/out points. A geometry change at the shadow in/out points is involved. Sample 4 is a listing of the operations data block for a restart run that recalculates shadow factors, reads form factors, combined form factors, and gray body factors from the RSI tape, calculates radiation conductors (RADKs, AQ's and QO's are always calculated since they are not saved for restart), and recalculates direct fluxes using shadow factor tables.

SAMPLE 1 -- OPERATIONS BLOCK FOR PLOT OPERATIONS

```
HEADER OPERATIONS DATA

C       BUILD GEOMETRY

BUILD CNAME1,BNAME1,BNAME2,BNAME3

C       INITIALIZE FOR PLOT 1
        CALL NDATA(1,3HGEN,0,0,0,A20,1,3,2,0.,0.,37.,0)
C       INITIALIZE FOR PLOTS 2 AND 3
        CALL NDATAS (2,3H3-D,0,0,0)
        CALL NDATAS (3,1HX,0,0,0)
C         MAKE PLOTS 1,2 AND 3
L       NPLOT
```

SAMPLE 2 -- OPERATIONS BLOCK FOR FORM FACTOR OPERATIONS

```
HEADER OPERATIONS DATA

BUILD CNAME1,BNAME1,BNAME2,BNAME3

C       SET FF CALCULATION PARAMETERS (PRINTS FF'S, DOES NOT PUNCH)
        CALL FFDATA(0.,.2,0,0.,1.E-3,3HYES,0,0)
C       COMPUTE FORM FACTORS
L       FFCAL
C       MOVE BNAME2 SURFACES
        CALL CHGBLK(BNAME2,0.,0.,25.,1,2,3,0.45.,0.)
BUILD CNAME2,BNAME1, BNAME2

C       CALCULATE FORM FACTORS WITH SAME PARAMETERS AS FOR CNAME1
L       FFCAL
END OF DATA
```

Figure 3-23 Sample Operations Data Blocks

SAMPLE 3 -- OPERATIONS BLOCK FOR TWO GEOMETRY ABSORBED HEAT PROBLEM

```
HEADER OPERATIONS DATA
STEP  1
BUILD CNAME1,BNAME1,BNAME2,BNAME3

C         SET FF CALCULATION PARAMETERS, PUNCH FFS
      CALL FFDATA(0,.2,0,0,1.E-3,0,3HYES,0)
L     FFCAL
C       CALCULATE  GREY BODY FACTORS
      CALL GBDATA(4HBOTH,0,2HFF)
L     GBCAL
C       SET RADK CALCULATION PARAMETERS, PUNCH RADKS
      CALL RKDATA(0,0,0,1000,5HSPACE,999,0,0,0,0)
C       COMPUTE RADKS
L     RKCAL
C       DEFINE ORBIT AND LOCATE SUN
      CALL ORBIT2(3HEAR,0.,90.,0,0,0,120.*6080.,0.)
C       ORIENT VEHICLE (CCS Z-AXIS TOWARD SUN)
      CALL ORIENT(3HSUN,1,2,3,0.,90.,0.)
C       SET DI COMPUTATION DATA
      CALL DIDT1S(0.,0,0.,3HYES,0)
C       COMPUTE DIRECT IRRADIATION (DICOMP PARAMETERS DEFAULT TO COMPUTE ALL)
L     DICAL
L     AQCAL
STEP 2
C       UPDATE TRUE ANOMALY, SET UP TO COMPUTE PLANET AND ALBEDO FLUXES
      TRUEAN      = 30.
      CALL DICOMP(1,0,0)
L     DICAL
L     AQCAL
STEP 3
      TRUEAN      =60.
      CALL DICOMP(1,0,0)
L     DICAL
L     AQCAL
STEP 4
      TRUEAN      =90.
      CALL DICOMP(1,0,0)
L     DICAL
L     AQCAL
STEP 5
C       SKIP OVER PLANET SHADOW
      TRUEAN      =270.
      CALL DICOMP(1,0,0)
L     DICAL
L     AQCAL
STEP 6
      TRUEAN      =300.
      CALL DICOMP(1,0,0)
L     DICAL
L     AQCAL
STEP 7
      TRUEAN      = 330.
      CALL DICOMP(1,0,0)
L     DICAL
L     AQCAL
```

Figure 3-23 (con't)

```
STEP 8
C        STUFF TRUEAN = 0. DI and AQ VALUES (DUPLICATE POINT)
       CALL STFAQ(360.,0,1)
STEP 9
C        COMPUTE DATA AT SHADOW ENTRY POINT (DAYSIDE GEOMETRY)
       TRUEAN      =SHADIN - .1
       CALL DICOMP(1,0,0)
L      DICAL
L      AQCAL
STEP 10
C        COMPUTE DATA AT SHADOW OUT POINT (DAYSIDE GEOMETRY)
       TRUEAN      =SHAOUT + .1
       CALL DICOMP(1,0,0)
L      DICAL
L      AQCAL
C        PUNCH AQAVG,AQ VS. TIME TABLES - DAYSIDE
      CALL QODATA(3HALL,0,2HNO,3HYES,0,0,0,4HBOTH)
L      QOCAL
STEP 11
C        BUILD DARKSIDE CONFIGURATION

BUILD CNAME2,BNAME1,BNAME2,BNAME4

C        CALCULATE FFS (FFDATA PARAMETERS SET IN STEP 1)
L      FFCAL
C        CALCULATE GREY BODY MATRICES
       CALL GBDATA (4HBOTH,0,2HFF)
L      GBCAL
C        SET RADK CALCULATION PARAMETERS, COMPUTE RADKS
       CALL RKDATA(0,0,0,1000,5HSPACE,999,0,0,0,0)
L      RKCAL
C        REORIENT TO PLANET
       CALL ORIENT(4HPLAN,1,2,3,0.,90.,0.)
       TRUEAN      =120.
L      DICAL
C      COMPUTE ABSORBED HEATS
L      AQCAL
STEP 12
C        UPDATE TRUE ANOMALY, STUFF HEAT DATA FROM STEP 11 BECAUSE ORBIT
C        IS CIRCULAR, PLANET-ORIENTED
       CALL STFAQ(150.,0,11)
STEP 13
       CALL STFAQ(180.,0,11)
STEP 14
       CALL STFAQ(210.,0,11)
STEP 15
       CALL STFAQ(240.,0,11)
STEP 16
C        STUFF DATA FOR SHADOW ENTRY POINT (DARKSIDE CONFIGURATION)
       CALL STFAQ(SHADIN + .1,0,11)
STEP 17
C        STUFF DATA FOR SHADOW OUT POINT (DARKSIDE CONFIGURATION)
       CALL STFAQ(SHAOUT - .1,0,11)
C        PUNCH AQAVG,AQ VS TIME TABLES - DARKSIDE
       CALL QODATA(ISARY,0,2HNO,3HYES,0,0,0,4HBOTH)
L      QOCAL
END OF DATA.
```

Figure 3-23 (cont)

Sample 4 --- Orbit Generation from an ORBGEN Card

```
C       ORBIT GENERATION CARD FOLLOWS
ORBGEN       CIRP, 0.0, 360.0, 4, AQ

C * * * * * * * * * * * *     ORBIT GENERATION STARTS HERE * * * * * * * * * *
STEP  10000
      TRUEAN      =         0.
      TRUANF      =       360.000
      TRUANI      =         0.
      IAI         =        0
      IAS         =        0
      PLTYPE      =      6HPLSAVE
      CALL DICOMP(0,0,0)
L     DICAL
      NSPFF       =      10000
      PLTYPE      =      6HPLREAD
      CALL AQDATA(IAI,IAS,0,0,0)
L     AQCAL
STEP  10001
      CALL STFAQ (TRUANF,0,0,1000)
STEP  10002
      TRUEAN      =            90.000
      CALL DICOMP(0,0,10000)
L     DICAL
.     CALL AQDATA(IAI,IAS,0,0,0)
L     AQCAL
STEP  10003
      TRUEAN      =           180.000
      CALL DICOMP(0,0,1000)
L     DICAL
      CALL AQDATA(IAI,IAS,0,0,0)
L     AQCAL
STEP  10004
      TRUEAN      =           270.000
      CALL DICOMP(0,0,10000)
L     DICAL
      CALL AQDATA(IAI,IAS,0,0,0)
L     AQCAL
STEP  10005
      IF(SHADIN.LT.0.)               GO TO 90400
      TRUEAN      =   SHADIN-0.1
      IF(TRUEAN.LT.TRUANI.OR.
     1TRUEAN.GT.TRUANF)          GO TO 90000
      CALL DICOMP(0,4HZERO,10000)
L     DICAL
      CALL AQDATA(IAI,IAS,0,0,0)
L     AQCAL
90000 CONTINUE
STEP  10006
      TRUEAN      =   SHADIN+0.1
      IF(TRUEAN.LT.TRUANI.OR.
     1TRUEAN.GT.TRUANF)          GO TO 90100
```

Figure 3-23 (con't)

Sample 4 --- Orbit Generation from an ORBGEN Card (continued)

```
      CALL DICOMP(0,0,10000)
L     DICAL
      CALL AQDATA(IAI,IAS,0,0,0)
      AQCAL
90100 CONTINUE
STEP  10007
      TRUEAN     =    SHAOUT+0.1
      IF(TRUEAN.LT.TRUANI.OR.
     1TRUEAN.GT.TRUANF)       GO TO 90200
      CALL DICOMP(0,4HZERO,10000)
L     DICAL
      CALL AQDATA(IAI,IAS,0,0,0)
L     AQCAL
90200 CONTINUE
STEP  10008
      TRUEAN     =  SHAOUT-0.1
      IF(TRUEAN.LT.TRUANI.OR.
     1TRUEAN.GT.TRUANF)        GO TO 90300
      CALL DICOMP(0,0,10000)
L     DICAL
      CALL AQDATA(IAI,IAS,0,0,0)
L     AQCAL
90300 CONTINUE
90400 CONTINUE
      CALL QODATA(3HALL,0,0,0,0,0,0,0,)
L     QOCAL
C
C * * * * * * * * * * * * *    ORBIT GENERATION ENDS HERE * * * * * * * * * * *
```

Sample 5 --- Operations Block for Restart Run

```
HEADER OPERATIONS DATA
STEP  1
BUILD CNAME,BNAME
      CALL RSTOFF
L     SFCAL
      CALL RSTON
      CALL FFDATA(0,0,0,0,0,0,0,2HNO)
L     FFCAL
      CALL CMDATA(0,5HFFNEW,FF, 0, 0)
L     CMCAL
      CALL GBDATA(4HBOTH,0,2HCM)
L     GBCAL
      CALL RKDATA(0,2HNO,0,100,5HSPACE,999,0,0,2HNO,0)
L     RKCAL
      CALL ORBIT2 (3HEAR,0.,90.,0.,0.,0.,100.*6080.,0.)
      CALL DIDT2S(0,0.,180.,0.,0.,0.,100.*6080.,2HNO,0)
      CALL RSTOFF
L     DICAL
END OF DATA
```

Figure 3-23 (concluded)

3.3.9.7 Restart Operations

The simplest restart operation is picking up a run interrupted by a system abort (e.g., time limit) where there is no requirement to modify the input deck. This is accomplished by specifying RSI in the Options Data, using the previously generated RSO tape as an RSI tape and submitting the same deck used on the previous run. If the data from the new run is to be saved RSO must also be specified in the Options Data block. The same thing can be accomplished by submitting an input deck consisting only of an Options Data block which must, of course, specify an RSI tape. This is possible because the entire input deck resides on the RSI tape. If the interrupted run used an RTO tape, it should be specified and mounted as an RTI tape when making the restart run. This enables the user to reclaim interim data from the FF, NFF, GB, DI and SF segments.

All restart operations involving more than a simple resumption of an interrupted run are accomplished through edit commands. All input decks used in this type of restart run will consist only of an options data block and an edit data block. Any data (numerical or logic) may be inserted or deleted from the input deck using edit commands. The editing is conveniently accomplished using the usual edit commands (see Section 3.2.2.2: Edit Data Block) while referring to the output listing from the previous run for line number information. The user should keep in mind that the input deck, as edited is written to the RSO tape. Therefore, when restarting from a tape generated by a previous restart/edit run, the previously used edit commands are not required, and will in fact probably generate errors.

Calls to subroutines RSTOFF and RSTON are edited into the operations data block to give the user direct control over the reading of the processor phase data on the RSI tape. CALL RSTOFF says in essence, do not read the RSI tape until further notice. CALL RSTON says read it until further notice. The operations data begins as if an RSTON call was in effect. The use of RSTOFF can be illustrated using a common parametric study situation. Say that a user has generated radiation interchange factors for a particular model and has the pertinent form factors and gray body factors on a restart tape. He desires to change some surface properties and generate new radiation conductors. This, of course, involves editing in calls to subroutine MODPR prior to the L GBCAL card. However, if a call to RSTOFF is not inserted ahead of the L GBCAL card, the gray body matrix residing on the restart tape will be read in, and the radiant interchange factors obtained will be identical to those of the previous run. For further background on RSTOFF and RSTON the reader is referred to Section 4.3.9: Restart Control Subroutines.

The user should keep in mind that significant amounts of data may be lost when executing without an RTO tape. This is illustrated by the following explanations of the RSO/RTO write sequences:

a) The FFCAL and NFFCAL segments write to the RSO tape at the end of each row of form factor computations. It writes to the RTO tape after every 10 form factors are computed. The same is true of image factors in the RBCAL link.

b) The form factor combining link writes to the RSO after each row of form factors are combined. It does not write to the RTO tape.

c) The DICAL and DRCAL segments write to the RSO tape at the end of each orbital time point (step). They write to the RTO tape after each 10 flux calculations.

d) The SFCAL segment writes to the RSO tape after each nodal shadow factor table computation is complete. It does not write to the RTO tape.

e) The GBCAL segment writes to the RSO tape at the end of the matrix inversion process for each waveband. It writes to the RTO tape at the end of each of three steps in the process of inverting the matrix.

Once the data from the RTO is passed on to the RSO tape, the RTO tape is rewound, and the next set of computations are written to the RTO tape to be saved temporarily until the time the data is again passed on to the RSO tape. In reading a RTO created tape back in as a RTI tape, the program will first read what it can from the RSI tape and then read the RTI tape if available. After the one read it dynamically frees the RTI unit.

The following paragraph applies to restart operations using UNIVAC EXEC 8 operating systems only.

In all TRASYS processor printout, when an RSO tape is used, restart tape record numbers are printed whenever information is written to the RSO tape. When this run ends, whether from time limit, abort or normal exit, there will be some record number, say for instance 100, near the end of the printed output. This means that presumably 100 records of valid information exist on the RSO tape. It is a peculiarity of the Univac system, as applied to TRASYS, that it cannot reliably differentiate between a parity error and the end of information on an RSI tape. Thus, if this 100 record tape were used on a restart run, and a parity error was encountered at record 50, calculations would begin at record 50 and much of the data generated on the first run would be recomputed. This may be necessary if there is a true parity error on the tape, but if the error was generated by the tape drive, half of a perfectly good tape would be wasted. This potential problem is the reason for allowing the user to specify RSREC. Had the card RSREC=100 been in the Options Data block on the second run, the parity error would have resulted in an abort rather than recomputing. The abort would provide the option to try the run again to see if the parity error was spurious or really on the tape. If not specified, RSREC defaults to zero.

3.3.9.8 Description of Restart Files

3.3.9.8.1 Permanent Restart Output Tape - RSO

A complete RSO tape consists of two files. The first file contains images of all cards in the input data that were present in the original run. In addition, this file contains the edit information that allow editing of the input data in subsequent runs. After editing, this file is processed by the preprocessor in the usual manner in preparation for processor operations. The second file contains data that was output by the processor segments as the user-defined Operations Data logic was processed.

An alternate program and control card runstream for the Univac allows recovery from a system crash without loss of data, and bypasses the preprocessor when the run is re-initated after the crash. This version has nine files. For this version the first and last file (ninth file) are identical to the first and second files defined above. The second file contains the processor absolute element created in the previous run, and a copy of the first of eight temporary files used to pass data from the preprocessor to the processor. The third through eighth files are copies of the seven remaining temporary files.

3.3.9.8.2 Permanent Restart Input Tape - RSI

The RSI tape is an RSO tape from a previous run that is now to be used as input for a restart run. Input from this tape can be edited as required using edit statements. Data from tapes other than the RSI tape may be merged into the TRASYS model (first file of RSI tape) by adding CMERG or EMERG statements to the TRASYS model as required. As a reminder, CMERG files are TRASYS input data in BCD form. EMERG files are the first files of other RSI/RSO tapes.

3.3.9.8.3 Temporary Restart Output Tape - RTO

The RTO tape is used to save partially completed calculated data in the DICAL, DRCAL, FFCAL, RBCAL, and GBCAL segments to minimize the amount of data lost upon run abort due to time limit, equipment failure, etc.

3.3.9.8.4 Temporary Restart Input Tape - RTI

The RTI tape is an RTO tape from a previous run that is now to be used as input for a restart run.

3.3.10 Subroutine Data Block

3.3.10.1 Basic Concepts

The subroutine data block is a collection of FORTRAN language subroutines supplied by the user in order to extend or modify TRASYS capabilities for the problem at hand. These subroutines may be either user-addressable (from the operations block) or program-addressable, from the various computation segments.

Unless the user is creating what amounts to a major rewrite of a computation segment, the program subroutines in his subroutine block will bear the same name as processor library subroutines. The effect of his name duplication is that the user-supplied routine in the subroutine data block is compiled in lieu of the processor library subroutine prior to execution. Removal of such a routine reactivates the like-named library routine.

Three deviations from FORTRAN language are defined for the subroutine data block. L-cards are used to identify subroutines with particular processor segments, R-cards are used to alert the program to the beginning of a new subroutine (UNIVAC only) and the TRASYS COMMON cards are used to automatically supply program common to the subroutines.

3.3.10.2 Subroutine Block Formats

Subroutine data block format is illustrated in Figure 3-24. The segment names on the L-cards are strictly order-dependent. The L-cards with their associated subroutines need not all be present, but they must be encountered in the order shown below. Subroutines in the leading sub-block, with no L-card, are addressable only from the operations data block.

No L-card	
L	FFCAL
L	SFCAL
L	NPLOT
L	OPLOT
L	DICAL
L	GBCAL
L	AQCAL
L	QOCAL
L	RBCAL
L	PLOT
L	RCCAL (or RKCAL)
L	DRCAL
L	CMCAL
L	NFFCAL

Punching Instructions | Page of
F | Card Form # * | Identification 73 80
Programmer | Date | Punch

C FOR COMMENT
STATEMENT NUMBER 1 5 | Cont. 6 | FORTRAN STATEMENT 7 10 15 20 25 30 35 40 45 50 55 60 65 70 72

```
HEADER SUBROUTINE DATA
R      VALUE
       SUBROUTINE VALUE(X,Y,Z)
       Z           =SQRT(X**2+Y**2)
       RETURN
       END
L      FFCAL
R      FFROW
       SUBROUTINE FFROW
COMMON
       CALL FFRSUM
       IF(SUM(IN).GT.2.)             GO TO 100
       WRITE(NFF)NODE(IN),(FFVALS(I),I=IN,NNOD),(FFVALI(I),I=IN,NNOD)
       RETURN
100    WRITE(NOUT,1)IN
1      FORMAT(24H FORM FACTOR SUM FOR ROW,I6,34H IS GREATER THAN 2.,  ABO
     1RTING RUN)
       CALL ABT
       END
L     DICAL
R      DIPREP
```

Figure 3-24 Subroutine Data Block Example

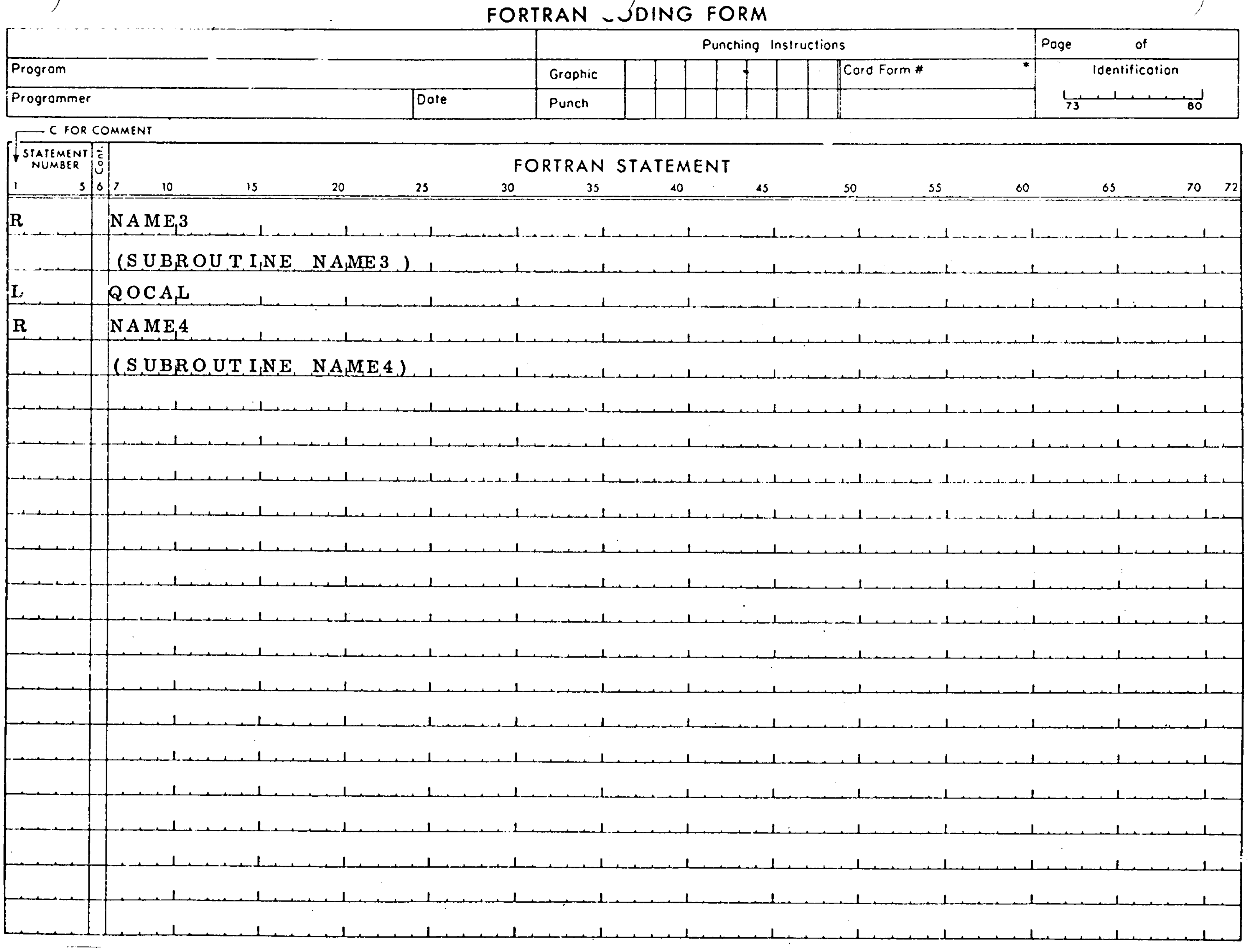
	Punching Instructions	Page of
Program	Graphic	Card Form #	Identification
Programmer	Date	Punch		73 80

C FOR COMMENT

STATEMENT NUMBER	Cont.	FORTRAN STATEMENT
1 5	6	7 10 15 20 25 30 35 40 45 50 55 60 65 70 72

```
      DO 100 I=31,NNOD
      QDP(I)      =0.
      QDR(I)      =0.
100   CONTINUE
      RETURN
      END
R     DIPRES
      SUBROUTINE DIPRES
COMMON
      DO 100 I=31,NNOD
      QDS(I)      =0.
100   CONTINUE
      RETURN
      END
L     GBCAL
R     NAME1
      (SUBROUTINE NAME1)
L     RKCAL
R     NAME2
      (SUBROUTINE NAME2)
L     AQCAL
```

Figure 3-24 (con't)



FORTRAN CODING FORM

	Punching Instructions		Page of
Program	Graphic	Card Form #	Identification 73 80
Programmer / Date	Punch		

C FOR COMMENT

STATEMENT NUMBER | Cont. | FORTRAN STATEMENT

```
R     NAME3
      (SUBROUTINE NAME3 )
L     QOCAL
R     NAME4
      (SUBROUTINE NAME4)
```

Figure 3-24 (concluded)

TRASYS COMMON cards are optional. When used, they serve to insert all labeled and blank common lists, associated with the segment named on the preceding L-card, into the subroutine. Appendix A-1 defines the variable names in common for subroutine ODPROG and the various segments.

NOTE: Subroutines immediately preceded by TRASYS COMMON cards will be compiled with both the operations data (ODPROG) common blocks and the common blocks associated with the preceding L-card. This means that all quantities data and array data, which are normally accessed only in ODPROG, are available to all subroutines in the subroutines data block.

Note that TRASYS COMMON cards preclude beginning a subroutine block comment card with the word COMMON.

R-cards must immediately precede each subroutine name card (UNIVAC only), with no intervening L-cards or TRASYS COMMON cards.

3.3.11 End of Data Card

The input deck should conclude with an END OF DATA card. If it is missing it will assume one and write a caution message.

4. USER CALLED ROUTINES

4.1 Basic Concepts

User called routines may be defined as those subroutines and computation segments callable from the operations block. Unless one or more subroutines of this type are entered in the user's subroutine data block, the user callable subroutines contained in the processor library comprise the entire list of user callable subroutines. Segments cannot be entered in the input stream.

4.2 Processor Library

The user callable processor library routines are listed below. In general, they are grouped according to their association with each of the processor computation segments. Page references for the subroutine descriptions in Appendix D are included.

4.2.1 Library Listing of Subroutines

	Name	Page	Name	Page
General Subroutines	BUILDC	D-7	ADD	D-2
	CHGBLK	D-9	FFNDP	D-25
	LIST	D-29	NODDAT	D-43
Plot Package Subroutines	NDATA	D-40	ODATA	D-44
	NDATAS	D-40	ODATAS	D-44
	PLDATA	D-51		

	Name	Page	Name	Page
Form Factor Subroutines	FFDATA	D-23	ADSURF	D-3
	CMDATA	D-10	RBDATA	D-57
	NFDATA	D-42		
Direct Irradiation Subroutines	ORBIT1	D-46	ORBIT2	D-48
	DIDT1	D-13	DIDT1S	D-13
	DIDT2	D-15	DIDT2S	D-15
	SPIN	D-65	ORIENT	D-50
	DICOMP	D-11	DITTP	D-20
	DITTPS	D-20	DRDATA	D-22
Restart	RSTON	D-63	RSTOFF	D-63
	FFREAD	D-26	DIREAD	D-19
Radiation Interchange Subroutines	GBDATA	D-28	RKDATA	D-61
	GBAPRX	D-27	RCDATA	D-58
Absorbed Heat Subroutines	AQDATA	D-5	STFAQ	D-67
Absorbed Heat Output Subroutine	QODATA	D-54	QOINIT	D-56
Data Modification Subroutines	MODAR	D-30	MODPR	D-32
	MODTR	D-38	MODPRS	D-33
	MODSHD	D-36		
Planet Surface Subroutines	DIDT3	D-17	DIDT3S	D-17
	SURFP	D-68		

4.2.2 Library Listing of Processor Segments

	Name	Page	Name	Page
Plot Package Segments	NPLOT	E-2	OPLOT	E-2
	PLOT	E-2		
Form Factor Segments	FFCAL	E-3	CMCAL	E-4
	NFFCAL	E-3	RBCAL	E-3
Direct Irradiation Segments	DICAL	E-5	DRCAL	E-5
Shadow Factor Generator Segment	SFCAL	E-6		
Radiation Interchange Segments	RKCAL	E-7	GBCAL	E-8
	RCCAL	E-7		
Absorbed Heat Segment	AQCAL	E-8		
Absorbed Heat Output Segment	QOCAL	E-9		

4.3 Subroutine Descriptions

4.3.1 Basic Concepts

The user-callable subroutines in the processor library fall into two functional groups. The most numerous group consists of subroutines used to define the program variables and set the logic flags that are required before a computational segment can be linked into the processor and executed. Variable definition in this manner, as opposed to definition from the data blocks, achieves two important goals. First, in any complex problem many segment calls are made, necessitating frequent redefinition of program variables. Under these conditions, data block input would be redundant. Second, the subroutine calls, in classic FORTRAN format, form natural groups of input variables, and as the calls are input serially in the user's operations block, he is provided a highly visible presentation of the variable definitions existing at each stage of his execution. Thus, if the user carefully proofreads a listing of his operations block, his logic and variable definitions should be error-free. If his geometry inputs have been verified using the plot package, the user may proceed with some confidence to consume a large block of computer time. It is the hope of the TRASYS designers that these features will ease somewhat the all to prevalent garbage in - garbage out syndrome.

The second group of processor library subroutines perform data handling tasks that eliminate redundant calculations. For example, direct irradiation may be required at 15 orbit points for a sun-oriented spacecraft. Armed with the knowledge that his solar flux is everywhere constant (outside the planet shadow), the user may compute the solar flux in Step 1, then use the STFAQ routine to retrieve the data from Step 1 and place it in data storage for any of the other 14 steps he desires.

Appendix D is composed of summary descriptions of each user subroutine. Definitions of each variable in the calling sequences are given, together with their default values, where applicable. The additional material necessary to use the subroutines is presented in the remainder of this section. After achieving a working knowledge, the user should find Appendix D sufficient for his quick-reference needs.

4.3.2 General Subroutines

Subroutines BUILDC and ADD and the BUILD card (see Sections 3.3.9.4: Build Option) are used to choose, from the various blocks of surfaces in che surface input data, what blocks are to be assembled to create (build) the problem geometry (active configuration). Also, the relative spatial positions of the surfaces and nodes in the active blocks may be changed using subroutine CHGBLK, prior to building the active configuration.

4.3.2.1 Subroutine BUILDC

Calling Sequence: CALL BUILDC (BCSNAM, CONFIG)

This subroutine begins the process of assembling the geometry desired from the blocks of surfaces in the surface data block. BCSNAM is any block coordinate system name found in the surface data block. If no BCS is named in the surface data block, CALL BUILDC (ALLBLK, NAME) will define a geometry consisting of the entire surface data block. CONFIG is any name (1 to 6 character Hollerith string) chosen by the user to identify his active model configuration. This call must be made for geometry definitions or after Block Coordinate System redefinition via subroutine CHGBLK.

Important Note: A BUILD card or a BUILDC call voids any previous BUILD/ADD calls and any previous Surface Property Modification Subroutines calls, (e.g., MODAR, MODPR).

4.3.2.2 Subroutine ADD

Calling Sequence: CALL ADD (BCSNAM)

This subroutine adds another block of surfaces to the geometry defined by previous BUILDC and ADD calls. One BUILDC call must precede any ADD call in the operations block.

Note: Direct calls to BUILDC and ADD have been made essentially obsolete by the BUILD card. See Section 3.3.9.3: Build Option, and Related Information for subroutine BUILDC in Appendix D.

4.3.2.3 Subroutine CHGBLK

Calling Sequence: CALL CHGBLK (BCSNAM, TX, TY, TZ, IROTX, IROTY, IROTZ, ROTX, ROTY, ROTZ)

This subroutine is used to spatially relocate blocks of surfaces by redefining a block coordinate system's location and orientation in central coordinate system 3-space. BCSNAM is the name identifying the block coordinate system being changed. Its spelling must be exactly as called out in the BCS data block.

The arguments TX, TY, TZ, ROTX, ROTY, ROTZ are the translation and rotation parameters necessary to transform the central coordinate system into the block coordinate system in its new position. These parameters are further discussed in Section 3.3.4.: BCS Data

The arguments IROTX, IROTY and IROTZ control the order in which the rotations ROTX, ROTY, and ROTZ are performed. They may take on the integer values 1, 2, or 3. For example, for IROTX = 3, IROTY = 1, and IROTZ = 2, the rotations will be performed in the order ROTY first, ROTZ second, and ROTX third. If zero is passed for all three arguments, the default values IROTX = 1, IROTY = 2, and IROTZ = 3 will result and the rotations will be performed ROTX first, ROTY second, and ROTZ third.

4.3.3 Form Factor Subroutines

The subroutine FFDATA is used to set the variables and control constants required before executing the FFCAL computation segment. An FFDATA call prior to all FFCAL executions is not mandatory because each FFDATA argument assumes a default value (ref. Appendix D). The variables defined by an FFDATA call will hold for any subsequent FFCAL executions in the operations block.

Subroutine NFDATA accomplishes the same function for the precision form factor segment, NFFCAL.

Subroutine CMDATA accomplishes a similar function prior to execution of the form-factor combining segment, CMCAL.

4.3.3.1 Subroutine FFDATA

Calling sequence: CALL FFDATA (FFACC, FFACCS, FFNOSH, FFRATL,
FFMIN, FFPRNT, FFPNCH, FFNAC)

FFACC is the variable that provides user control of the node surface elemental breakdown used for double integration form factor calculations. In general, the accuracy of a form factor calculation is proportional to the ratio of each elemental area divided by the square of the distance between each elemental area pair involved. Thus, if the element count for each node were chosen so that a given value of this ratio were never exceeded, then the error of each non-shadowed form factor calculation would similarly be limited. The form factor segment logic provides for this, and FFACC is the upper limit allowed for the area - distance squared ratio. The default value used, (FFACC = .05) provides form factor accuracy of approximately 2 percent for parallel flat plates. Background information for this accuracy relationship can be found in Appendix B. The user is cautioned that his problem run time is tied directly to his nodal element count, and indiscriminate reduction in the value of FFACC can be costly. The recommended approach to accuracy improvement is to selectively re-compute suspect form factors using a reduced FFACC value, or utilize the NFFCAL computation segment.

When node pairs are situated such that the interelement distances vary a great deal, it is sometimes necessary to temporarily subdivide node pairs in order to obtain sufficient accuracy. The parameter FFRATL controls this. The number of elements is dictated by a weighted average distance and compared to FFRATL. If this value exceeds FFRATL, the node pair is subdivided. When accuracy problems are encountered with node pairs having large interelement distance variation (nodes with congruent edges, for instance), the recommended procedure is to enter an FFRATL value lower than the default value (FFRATL = 15.), and selectively recompute the form factors. No change in FFACC should be required for this operation. Additional descriptive material on this technique can be found in Appendix B.

The elemental breakdown of node pairs also influences form factor accuracy when shadowing by intervening surfaces is involved. For large magnitude form factors, the node pair element breakdown required to satisfy shadowing considerations is computed and used if it exceeds that dictated by separation distance (see Appendix B). If the unshadowed Form Factor in both directions is less than FFMIN the computations associated with form factor shading are by-passed. The element count dictated by shadowing is inversely proportional to the parameter FFACCS. The default value for FFACCS (FFACCS = .1) was chosen based on experience. If the user knows that one or more of his significant form factors will be heavily influenced by shadowing, the recommended procedure is to selectively compute such form factors using a reduced value of FFACCS.

The parameter FFMIN is used to reduce the bulk of the BCD Card output that results when a large form factor matrix is punched in form factor data block input format for re-use. The form factor in both directions must be less than FFMIN before it is discarded, (results in a 0.0 data value). Significant preprocessor time can be saved if the insignificant form factors are eliminated from the form factor data block. Determining what is insignificant may be a problem though. The default value, 1.E-6 is sufficiently small that it does not materially affect the energy balance of most problems.

The FFNOSH parameter has two options 4HSHAD, 4HNOSH. The program will default FFNOSH = 4HSHAD. If by-passing the shadow calculation is acceptable (program won't allow it for specular surfaces - see Section 3.3.3.9.2: Single Variable Input Format) FFNOSH = 4HNOSH will overide the Surface Data SHADE and BSHADE flags and will probably save considerable computer time. Consideration of shadowing, even when none acutally exists, is the most expensive aspect of the form factor computations. For checkout of the proper geometry, and active side inputs etc., it may be beneficial to make a short run with FFNOSH = 4HNOSH, prior to a long run where shading is considered.

FFNAC is the flag to eliminate the usual check of the currently-defined node number array against the node array on the restart(RSI) tape. This allows a user to make use of form factors computed for a different geometry. This will be of benefit only when the user knows the geometry change will not have any significant effect on the form factors being reused. (See Section 3.3.5: Form Factor Data).

The remaining FFDATA parameters, FFPRNT, and FFPNCH are used to provide print and punch/tape output options. The tape option of FFPNCH will write the Form Factor * Area products to the USER1 tape with formatted writes acceptable to the Form Factor Data block (see Section 3.3.5: Form Factor Data). The USER1 tape must be referenced in the Option Data block and be actively assigned to the run.

An FFCAL or NFFCAL-related variable, IFFSHO, is used to control whether or not form factors to shadower-only nodes will be computed. The statement IFFSHO = NO prior to FFCAL or NFFCAL will bypass form factor calculations to the shadower-only nodes (see Section 3.3.3.12: Shadower-only Surfaces).

4.3.3.2 Subroutine NFDATA

Calling sequence: CALL NFDATA (NELCT, FFNOSH, FFMIN, FFPRNT, FFPNCH, FFNAC)

NELCT provides user control over computation accuracy in the NFFCAL segment through control over the number of elements used in form factor calculation. NELCT is the total number of elements allowed, divided between the i and j nodes according to their area. The allowable range is from 16 to 200, with a default of 50. Regardless of NELCT, elements on surfaces of revolution are forced to subtend no more than 15 degrees of arc. For example, a 360° cylindrical node will use at least 24 elements. The last five arguments are identical to the corresponding arguments in FFDATA. See Section 4.3.3.1.: Subroutine FFDATA.

An FFCAL or NFFCAL-related variable, IFFSHO, is used to control whether or not form factors to shadower-only nodes will be computed. The statement IFFSHO = NO prior to FFCAL or NFFCAL will bypass form factor calculations to the shadower-only nodes (see Section 3.3.3.12: Shadower-only Surfaces).

4.3.3.3 Subroutine CMDATA

Calling sequence: CALL CMDATA (NFIGFF, NFIGCO, NFFTYP, IAUTOC, FFPRNT)

Before utilizing the CMCAL Segment the user should refer to Section 3.3.8: Correspondence Data to read the words of caution on the proper application of form factor combining.

NFIGFF and NFIGCO are configuration names. The CMCAL segment will retrieve form factors (or image factors) stored under the name NFIGFF, combine them by applying CORRESPONDENCE DATA defined under configuration NFIGCO and store the resulting matrix under the current configuration name. Both these variables default to the current configuration name, as defined on the most recent BUILD Card. NFFTYP alerts CMCAL to the type of form factors to be found under NFIGFF. FF means ordinary form factors. RB means image factors, as generated by the RBCAL segment. If IAUTOC is entered as NO, polygons will remain uncombined. FFPRNT is the print/no print flag for the combined form factors. FFPRNT defaults to YES. If CMDATA is not called, CMCAL proceeds on the basis of default values.

4.3.3.4 Adiabatic "Closure" Surfaces

It is sometimes desirable to conserve energy in computing the IR radiation interchange factors in a thermal radiation problem that does not constitute a complete enclosure. The usual means of doing this is to complete the enclosure with an adiabatic reflector surface. This can be accomplished by entering a rudimentary closure surface in the surface data and using subroutine ADSURF to add the closure surface to the form factor matrix after form factors have been computed for the real surfaces in the problem. The ADSURF call should be followed by a GBDATA and a RKDATA or RCDATA call. An example of the application of this technique is shown in Appendix J.

Subroutine ADSURF

Calling sequence: CALL ADSURF (BCSN, NFIGFF, AREA)

BCSN is the block coordinate system name under which the closure surface appears in the surface data. Only one side of the closure surface can be active. NFIGFF is the configuration name under which the new modified form factor matrix is to be stored.

When ADSURF is called, the form factor matrix is read under the current configuration name, and form factors from each node to the closure surface are computed by subtracting the form-factor row sums from 1.0. This new row of form factors is added to the form factor matrix and the resulting matrix is stored under the name given for NFIGFF.

AREA is the area of the adiabatic closure surface. If desired, the area in square feet may be entered as this argument. If it is more convenient, the closure surface can be defined with the correct dimensions in the surface data block and the area will thus be computed. To use the computed area, enter 0 (zero) for this argument.

A subsequent GBDATA call must be made with the First and Last arguments always as shown. Call GBDATA (2HIR,6HNFIGFF,2HFF)

4.3.3.5 Subroutine RBDATA

Calling sequence: CALL RBDATA (NFIGFF, FFACC, FFACCS, FFRATL, FFPRNT)

Subroutine RBDATA is used to define the parameters necessary to execute the RBCAL program segment that computes form factors (also known as image factors) that include the one bounce effects of specular surfaces in the model. NFIGFF is the configuration name used by RBCAL to retrieve the ordinary form factors previously computed by FFCAL and/or NFFCAL. NFIGFF defaults to the current model name. The remaining RBDATA parameters are the same as defined under FFDATA and carry the same default values. If RBDATA is not called, RBCAL proceeds on the basis of default values. Note that currently even if the NFFCAL Segment is utilized to compute Form Factors, the imaged portion of the Form Factors will use the double summation method internally to the RB Segment. Thus, the image factors are computed with a combination of the two form factor methods.

4.3.4 Plot Package Subroutines

4.3.4.1 Subroutines NDATA, NDATAS

Calling sequences: CALL NDATA (NV, VU, SCL, NACT, ISHO, SELN, TIT, IROTX,
IROTY, IROTZ, ROTX, ROTY, ROTZ)

CALL NDATAS (NV, VU, SCL, NACT, ISHO)

These calls are used to define plot parameters prior to executing the NPLOT program segment. A call to one of these routines prior to an NPLOT execution is not mandatory, because all arguments have default values (ref. Appendix D). The variables defined by NDATA or NDATAS calls will hold for any subsequent NPLOT execution in the operations block.

The NV parameter allows the user to make up to 6 NDATA or NDATAS calls, thereby defining up to 6 plot operations, before executing NPLOT. One NPLOT execution will execute all the plot operations defined.

VU defines the type of plot desired. The options are 3H3-D, 1HX, 1HY, 1HZ, 3HALL, and 3HGEN. 3-D results in a 3-dimensional pictorial plot. X,Y, and Z produce orthographic projections of the geometry as seen from the X, Y, and Z axes of the CCS, respectively. ALL results in four frames, 3-D, X, Y, and Z. GEN is a general 3-D plot, where the user has control of the orientation of the CCS axes relative to his point of view.

SCL is the plot scale factor, defined by:

SCL = length on plot frame/length of surface where lengths of surfaces are as defined in the surface data block.

The user should keep in mind that his hardcopy plot frames are probably about 17.8 cm (7 inches) square.

NACT is the flag that controls whether or not active side indicating arrows appear on the plots. NACT defaults to NO. If YES is entered as this argument, arrows will be shown.

Shadow-only surfaces (Section 3.3.3.12: Shadower-Only Surfaces) are not included in the node plots unless argument ISHO is entered as YES.
SELN is the name of the array that contains a list of the integer node identification numbers the user desires to plot selectively. The selective node number array is entered in the array data block.

TIT is the name of the Hollerith array containing any title the user desires on his plot frame. Up to 66 characters are allowed.

IROTX, IROTY, IROTZ, ROTX, ROTY, and ROTZ are the group of six parameters defining point of view from which the user will "see" his problem geometry in a general view. For ROTX, ROTY, and ROTZ identically zero, the central coordinate system appears in plots as shown in Figure 4-1.

ROTX is the rotation angle about X, that rotates the plot reference coordinate system into the Central Coordinate System, Y toward Z positive. ROTY is the corresponding angle, about Y, Z toward X is positive. ROTZ is the corresponding angle, about Z, X toward Y positive.

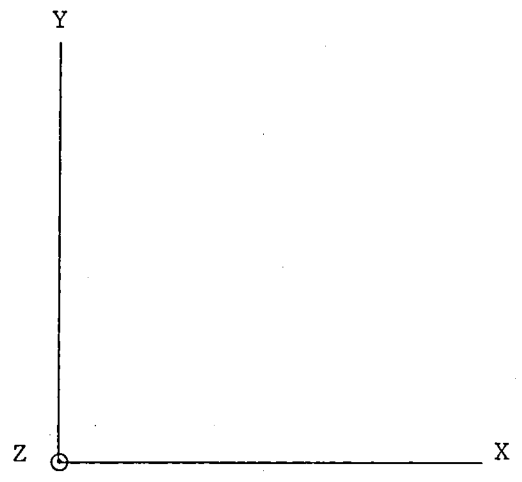


Figure 4-1 Node Plot Coordinate System Reference

Using Figure 4-1 as the reference position, the user may arbitrarily relocate the axes by defining ROTX, ROTY and ROTZ to relocate the reference system so that it coincides with the desired systems location. The order of the rotations is defined using IROTX, IROTY, and IROTZ.

4.3.4.2 Subroutines ODATA, ODATAS

Calling sequences: CALL ODATA (NV, VU, SCL, SCLR, RPLN, TRUEAN, TIMEST, TIME, SELN, TIT, IROTX, IROTY, IROTZ, ROTX, ROTY, ROTZ)

CALL ODATAS (NV, VU, SCL, SCLR, RPLN, TRUEAN, TIMEST, TIME)

These subroutines are functionally analogous to NDATA and NDATAS in relation to execution of the orbit plotter segment, OPLOT.

NV is defined and functions identically with the similarly named parameter in NDATA. VU defines the plot type. Options are 3H3-D, 4HBETA, 5HCIGMA, 3HSUN, 3HALL, and 3HGEN. 3H3-D results in a 3-dimensional pictorial plot of the planet and spacecraft. 4HBETA results in an edge-on view of the orbit plane, with the BETA angle shown true. The 5HCIGMA view places the orbit plane in the plot frame, as seen from north of the celestial equator. 3HALL produces four frames, 3-D, CIGMA, BETA, and SUN. GEN results in a plot with the orbit coordinate system axes rotated according to user definition. SCL relates the user's geometry dimensions to plot scale according to:

SFAC = SCL/OPMAX (NNS)

where

SFAC is the absolute surface scale factor (same as SCL in NDATA).

OPMAX (NNS) is the maximum extension of any surface point from the CCS origin (user surface data units).

The user should generally enter a value of SCL equal to about 1/2 the desired planet radius in inches of plot frame.

SCLR is the distance from the planet center where the user desires to see his CCS origin (in inches on plot frame).

RPLAN is the planet radius as plotted in inches. The planet radius default value used is 3.56 cm. (1.4 inches). The default values for SCL and SCLR are related to RPLAN according to the relationships:

SCLR = 8.*RPLAN/7.

SCL = (3.15 - SCLR)/2.

The user may note that his spacecraft's altitude, as it appears in the plots, is not related in any way to actual orbit altitude. This is because the primary reason for orbit plots is for visualization of orientation. Orbit radius is, however, available in common as the variable RTHET in the operations block. Therefore, if the user cares to consider the scaling involved, he may write operations block logic to relate SCLR to actual orbit radius.

TRUEAN is the true anomaly at the orbit point being defined for plotting (degrees from periapsis passage). Reference Figure 4-7: definition of True Anomaly, and Shadow Entry/Exit Points.

TIME is the time at which the orbit point plot is desired, in hours (required only if TRUEAN is not defined).

TIMEST is time of periapsis passage, in hours (required only if TRUEAN is not defined).

The remaining ODATA arguments, SELN, TIT, IROTX, IROTY, IROTZ, ROTX, ROTY, and ROTZ, are exactly analogous to the identically named NDATA arguments (ref. Section 4.3.4.1: Subroutines NDATA, NDATAS). Figure 4-2: Orbit Plot Coordinate System Reference depicts the orbit coordinate system as plotted for ROTX = ROTY = ROTZ = 0.

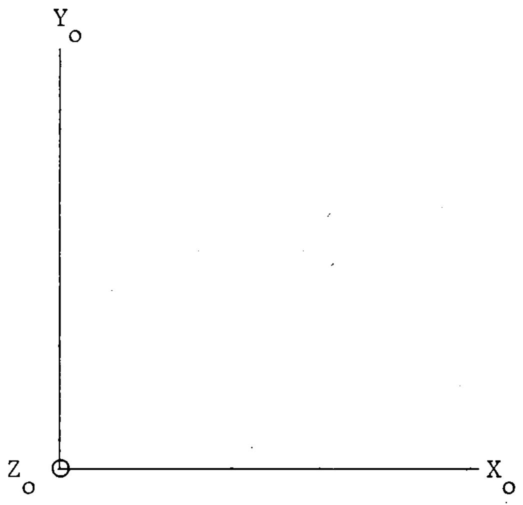


Figure 4-2 Orbit Plot Coordinate System Reference

4.3.4.3 Subroutine PLDATA

Calling sequence: CALL PLDATA (IPLUNT, IPLSN, IPLNA, PLCRVF, PLLABX, PLLABY, PLTIT1, PLTIT2, PLXMPF, PLYMPF, PLCMB)

This subroutine is used to define parameters necessary to execute the output data plotter segment PLOT. Refer to Appendix D: Subroutine PLDATA, for argument definitions.

4.3.5 Direct Irradiation Subroutines

The direct irradiation subroutines are used to spatially locate the spacecraft relative to energy sources. The various calls give the user the option of locating his spacecraft using classical orbit parameters, with a modified sun-referenced set of orbit parameters, with look angles, or with trajectory tape parameters. Subroutines are also available for defining spacecraft orientation and spin rate.

4.3.5.1 Subroutine ORBIT1

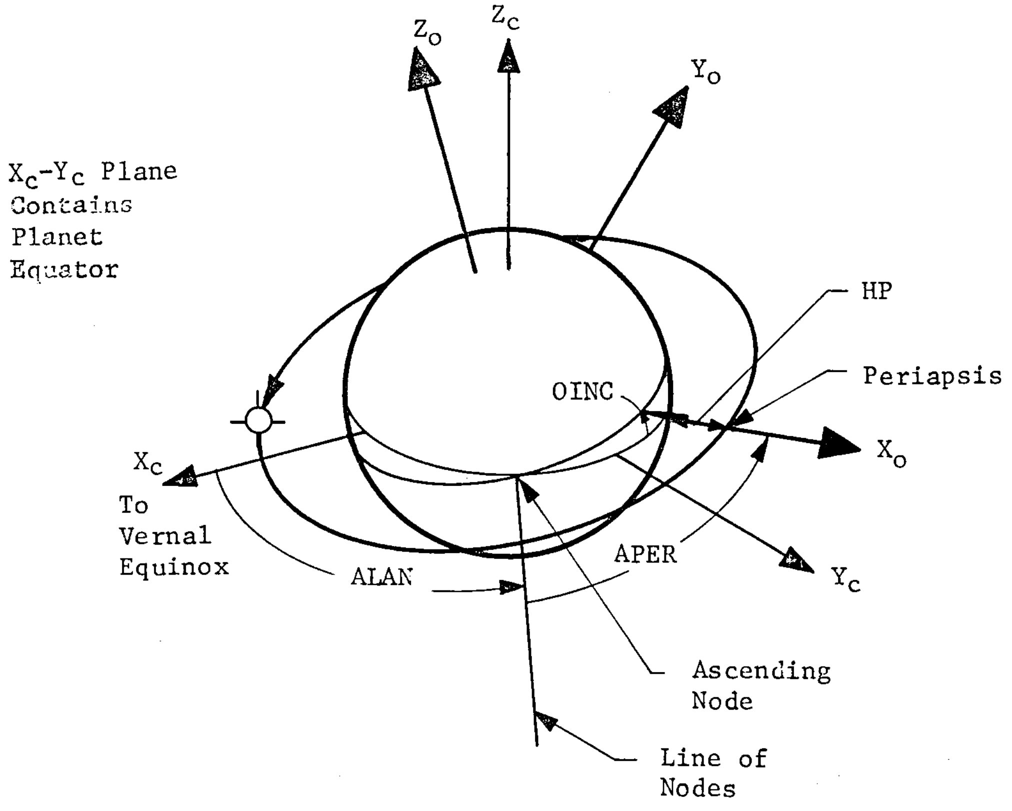
Calling sequence: CALL ORBIT1 (PNAME, ALAN, APER, OINC, TIMEST, HP, HA, SUNRA, SUNDEC, STRRA, STRDEC)

or:

CALL ORBIT1 (PNAME, ALAN, APER, OINC, TIMEST, HP, ECC, SUNRA, SUNDEC, STRRA, STRDEC)

This subroutine defines an orbit using classic orbit parameters and locates the sun in the same celestial coordinate system referenced to the Vernal Equinox (reference Figures 4-3 and 4-4).

In the case of a star-oriented spacecraft, the star is located in the celestial coordinate system, in the same manner as the sun.



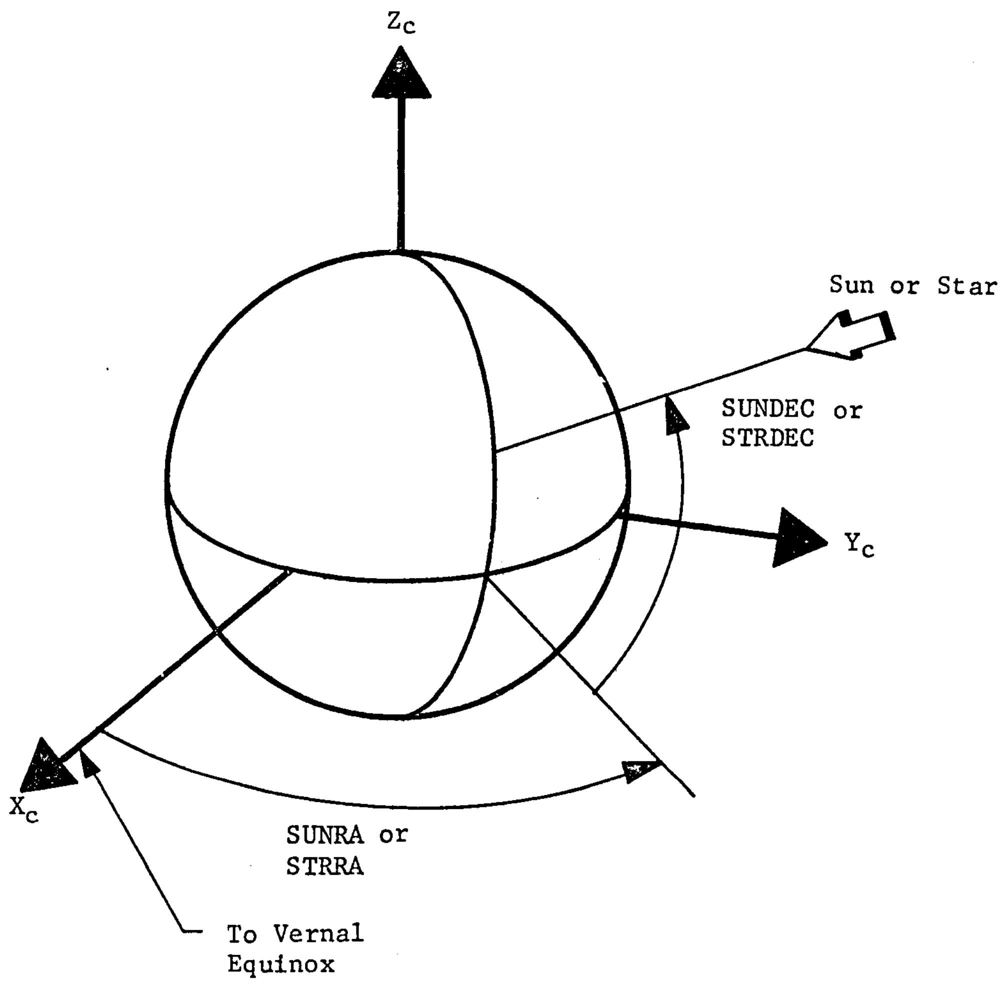
ALAN - Longitude of ascending node measured from X_c axis to line of nodes; positive toward Y_c

APER - Argument of perifocus. measured in orbit X_o-Y_o plane in direction of S/C motion from ascending node to periapsis

HP - Altitude at periapsis

OINC - Orbit inclination (angle between X_c-Y_c plane and orbit plane as seen from ascending node; $0. \leq \text{OINC} \leq 18$ ($\text{OINC} > 90^\circ$ for retrograde orbits)

Figure 4-3 Orbit Definition in a Celestial Coordinate System.



SUNRA, STRRA - Right ascension of sun/star; measured in X_c-Y_c plane from X_c axis; positive toward Y_c axis; $0. \leq RA \leq 360$

SUNDEC, STRDEC - Declination of sun/star; positive from X_c-Y_c plane toward Z_c; $-90 \leq DEC \leq 90$

Figure 4-4 Sun and Star Locations in Celestial Coordinate System

PNAME is a Hollerith name used as a flag to direct the definition of the planet-dependent parameters. The allowable PNAME options are: 3HMER, 3HVEN, 3HEAR, 3HMOO, 3HMAR, 3HJUP, 3HSAT, 3HNEP, 3HURA, and 3HSUN. These names serve to define the following variables which are set by the ORBIT1 or ORBIT2 call. If the User wants to change any of the variables they must be redefined to the new values after one of the ORBIT Subroutine Calls in the OPERATIONS DATA Block.

PRAD	-	planet radius
SOL	-	solar constant at the average planet-sun distance
PALB	-	planet albedo value (surface solar reflectance)
WDS	-	infrared emissive power at planet surface, dark side
WSS	-	infrared emissive power at planet surface, subsolar point
GRAV	-	acceleration of gravity at planet surface

The values obtained from the different planet name arguments are tabulated in Table 4-1. Note that the values tabulated are in metric units. The values stored in core, however, are in the TRASYS base units sytem, that is, length in feet, time in hours, energy in British thermal units. If the user desires to manipulate these quantities using his own operations block FORTRAN code, he would expect them to be in the ft - hour - Btu units.

Note that only Mercury and the Earth's moon are treated as bodies with nonuniform surface temperatures. This is correct for airless, slow-rotating planets. For these two bodies, the emissive power is considered everywhere constant on the dark side. On the sunlit side, the emissive power reduces from the subsolar value to the darkside value at the terminator, according to a cosine law.

The user is cautioned that his results using PNAME = 3HMER may be extremely misleading. This planet's eccentric orbit, plus its nearness to the sun, results in a solar constant variation of from approximately 6 to over 10 Earth "suns" during the Mercury year. Corresponding variations in the subsolar emissive power occur. The recommended procedure for PNAME = 3HMER is for the user to properly define SOL and WSS according to his knowledge of the planet-sun distance. This is done using two FORTRAN statements immediately following his ORBIT1 call. This same technique is available, of course, whenever the user desires to change WDS, WSS, or SOL to values other than the built-in nominals. The user's values will hold until another ORBIT call is encountered.

ALAN, APER, OINC, HP, and HA are the longitude of the ascending node, argument of perifocus, orbit inclination, and periapsis and apoapsis altitudes. These are the five parameters necessary to define an orbit in the celestial coordinate system. The alternate ORBIT1 call allows the input of eccentricity (ECC) in lieu of apoapsis altitude. TIMEST is the time of periapsis passage, in hours.

The angular measurement arguments are in decimal degrees of arc. The altitudes must be specified in feet. Note that FORTRAN allows arithmetic operations within argument lists; thus the following ORBIT1 call might be used where HP is known to be 150 nautical miles:

```
CALL ORBIT1 (3HEAR,32.,90.,22.5,0.,6080.*150.,.94,-41.,18.,0.,0.)
```

SUNRA and SUNDEC are the right ascension and declination of the sun, respectively, input in decimal degrees of arc (See Figure 4-4).

STRRA and STRDEC are the right ascension and declination, respectively, of a star for a star-oriented mission (see Figure 4-4). Zero or dummy arguments are passed for non-star-oriented missions.

For heliocentric orbits, (PNAME = 3HSUN) ALAN, APER, OINC, SUNRA, and SUNDEC have no meaning and are passed as zero or dummy arguments.

Table 4-I Stored Planet Property Values

Planet	Planet Radius (km)[a]	(ft)	Albedo	Solar Constant at Mean Planet Distance (w/m^2)[b]	(B/Ft^2-hr)	Darkside Emissive Power (w/m^2)[c]	(B/ft^2-hr)	Subsolar Emissive Power (w/m^2)[c]	(B/ft^2-hr)	Planet Gravitational Constant (m/s^2)[d]	(ft/s^2)
Mercury	2485.	(8.153 E06)	0.058	8920.	(2830)	0.	(0.)	8402.	(2666.)	3.513	(11.49)
Venus	6199.	(20.34 E06)	0.76	2570.	(815.5)	154.2	(48.93)	154.2	(48.93)	8.462	(24.68)
Earth	6370.	(20.90 E06)	0.30	1352.	(429)	236.6	(75.08)	236.6	(75.08)	9.844	(32.20)
Moon	1738.	(5.702 E06)	0.047	1352.	(429)	6.5	(2.060)	1288.	(408.7)	1.622	(5.306)
Mars	3314.	(10.87 E06)	0.148	577.3	(183.2)	123.0	(39.03)	123.0	(39.03)	3.921	(12.83)
Jupiter	69885.	(229.3 E06)	0.51	49.6	(15.74)	6.1	(1.936)	6.1	(1.936)	26.04	(85.18)
Saturn	57515.	(188.7 E06)	0.50	14.7	(4.66)	1.8	(.5711)	1.8	(.5711)	11.17	(36.54)
Uranus	25482.	(83.61 E06)	0.66	3.65	(1.16)	.31	(.0983)	.31	(.0983)	11.52	(37.68)
Neptune	24850.	(81.53 E06)	0.62	1.48	(.47)	.14	(.0444)	.14	(.0444)	8.977	(29.36)
Sun	698500.	(2291. E06)		--		6.262×10^7		6.262×10^7		273.8	(895.6)

[a]Values stored in program are in ft.

[b]Referenced to 1352 w/m^2 (429 Btu/hr-ft^2) at 1 AU. Values stored in program are in Btu/ft^2-hr.

[c]Values stored in program are in Btu/hr-ft^2.

[d]Values stored in program are in ft/hr.^{2}

4.3.5.2 Subroutine ORBIT2

Calling sequences: CALL ORBIT2 (PNAME, CIGMA, BETA, CIGMAS, BETAS, TIMEST, HP, HA)

or:

CALL ORBIT2 (PNAME, CIGMA, BETA, CIGMAS, BETAS, TIMEST, HP, ECC)

This subroutine defines an orbit using sun-referenced parameters in the orbit coordinate system. The orbit coordinate system has its X- and Y-axes in the orbit plane and its Z-axis completes a right-handed set. The spacecraft travels from X towards Y.

PNAME functions identically with ORBIT1.

CIGMA and BETA locate the solar vector in the orbit coordinate system (see Figure 4-5). CIGMAS and BETAS locate a star in the orbit coordinate system (see Figure 4-5). Again, these arguments are zero or dummy for non-star-oriented missions.

For heliocentric orbits, (PNAME = 3HSUN) ORBIT2 is not applicable.

4.3.5.3 Subroutine ORIENT

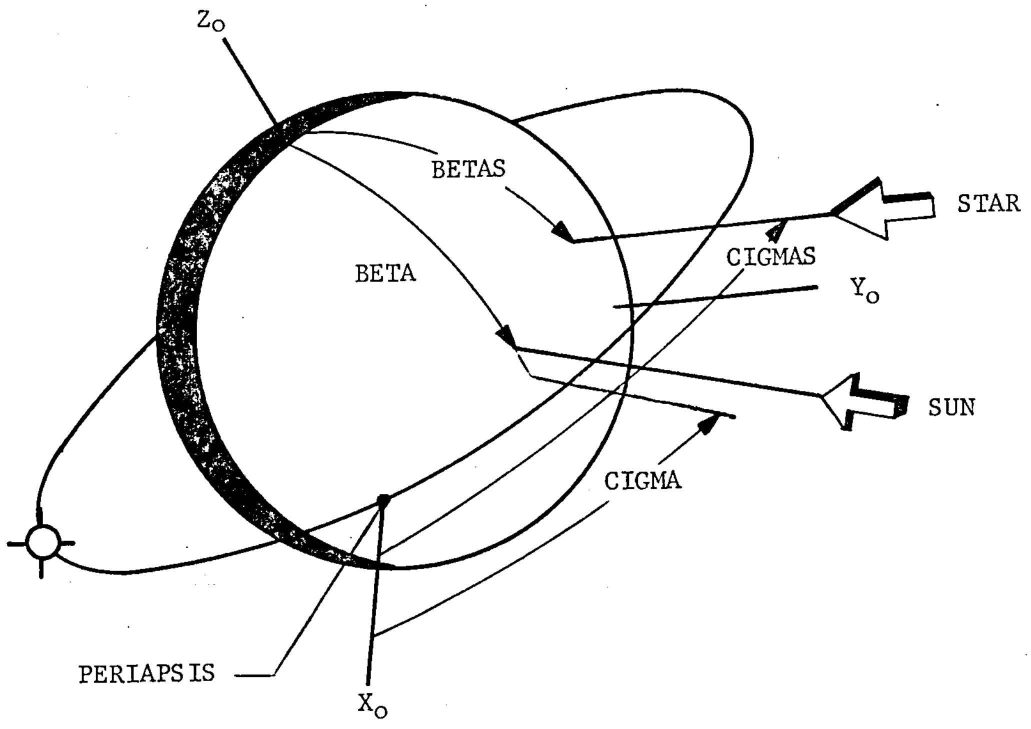
Calling sequence: CALL ORIENT (TYPE, IROTX, IROTY, IROTZ, ROTX, ROTY, ROTZ)

This subroutine is used to define the spacecraft orientation relative to space-environment heat sources. Orientation is accomplished by relating the spacecraft central coordinate system to a vehicle coordinate system (VCS) that remains fixed, relative to a heat source or a star reference.

TYPE is a Hollerith name used as a flag to define orientation of the VCS. Allowable options for TYPE are 4HPLAN, 3HSUN, 4HSTAR, or 4HTAPE. Figure 4-6 depicts the VCS relationship to the heat sources, star reference, and orbit coordinate system for the PLAN, SUN, and STAR option. The X_v-axis points to the planet, sun, or star and the Z_v-axis is in the same half-space as the Z_o-axis. The Y_v-axis lies in the orbital plane and completes the right-handed set. The TAPE option allows orientation to be defined from a trajectory tape. (See Section 3.3.9.3: TRJGEN Option or Section 4.3.5.8: Subroutine DITTP and DITTPS)

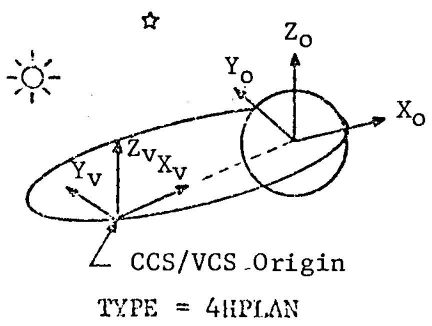
An ambiguity exists for the sun or star oriented options when the sun or star vector is parallel to the Z_o-axis. In this case, the Y_v-axis is defined to be in the direction of the velocity vector.

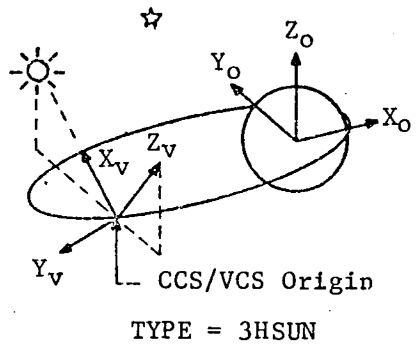
IROTX, IROTY, IROTZ, ROTX, ROTY, and ROTZ are the rotation parameters necessary to locate the spacecraft CCS relative to the VCS and, hence, the heat source(s). ROTX is the rotation angle to rotate the VCS <u>into</u> the CCS; it rotates about X_v-axis, Y_v toward Z_v positive. ROTY is the same as ROTX, except about the Y_v-axis, Z_v toward X_v positive; ROTZ is the same as ROTX, except about the Z_v-axis, X_v toward Y_v positive.

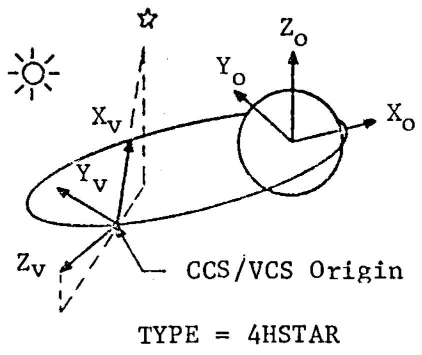


CIGMA	-	Angle from X_O axis to sun vector projection in X_O - Y_O plane. Measured CCW as seen from Z_O axis (in direction of S/C motion). $0. \leq$ CIGMA ≤ 360
CIGMAS	-	Same as CIGMA except to star vector projection
BETA	-	Angle from Z_O axis to sun vector $0 \leq$ BETA ≤ 180
BETAS	-	Same as BETA, except to star vector

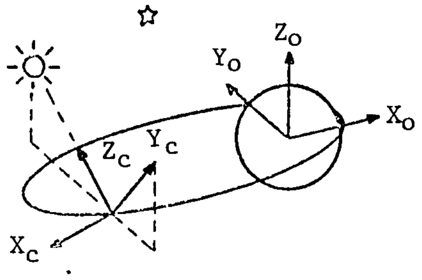
Figure 4-5 Orbit Definition in Orbit Coordinate System







Orientation Example
(CCS Z-axis locked to sun):
TYPE = 3HSUN



IROTX = 1	
IROTY = 2	
IROTZ = 3	Rotates VCS into CCS, -270° about Y axis
ROTX = 0.	
ROTY = -270.	
ROTZ = 90°	Rotates 90° about Z axis

Figure 4-6 Vehicle Orientation with Subroutine ORIENT

IROTX, IROTY, and IROTZ control the order in which the rotations are performed. Integers 1, 2, and 3 are the allowed options. For example, IROTX = 1, IROTZ = 2, and IROTY = 3 results in rotation first about X_v, then about Z_v, and then about Y_v.

4.3.5.4 Subroutines DIDT1 and DIT1S

Calling sequences: CALL DIDT1 (DINOSH, DIACC, DIACCS, TRUEAN, NSPFF, TIMEPR, DIPNCH, ISFAC)

or:

CALL DIDT1S (TRUEAN, NSPFF, TIMEPR, DIPNCH, ISFAC)

These subroutines allow the user to define the form factor and shadowing accuracy parameters used in his direct irradiation calculations. Additionally, these routines can be used to update the spacecraft position in orbit by defining true anomaly (reference Figure 4-7). True anomaly can be defined directly or by defining a current time.

DINOSH is a shadow/no shadow flag for direct irradiation calculations. DINOSH = 4HSHAD retains shadowing calculations. DINOSH = 4HNOSH bypasses shadowing calculations. If by-passing the shadowing calculations is acceptable to the user DINOSH = 4HNOSH will overide the Surface Data SHADE and BSHADE flags and may save considerable computer time. Determining the existance of shadowing even when none actually exists is the most expensive aspect of the DI computations. For checkout of the proper orbit/orientation and active side inputs etc., it may be beneficial at times to make a short run with DINOSH = 4HNOSH prior to a long run where shading is considered. For the user faced with the need for a more in-depth understanding of the shadowing as it applies to a specific model the program has the optional feature of printing out for each DI computation the list of surface numbers it considered as possible shadowers. This is accomplished by adding in the Operations Data block ITRC70 = 2HON prior to the DI segment call. (This feature is available on the UNIVAC version only).

DIACC is the element selection accuracy factor for node planet form factor calculations. Its function is similar to FFACC in form factor calculations and its default value is 0.25. (ref. Section 4.3.3.1: Subroutine FFDATA and Appendix B).

DIACCS is the element selection accuracy factor for shadowing calculations (not applicable when DINOSH = 4HNOSH). Its function is similar to FFACCS in form factor calculations and its default value is 0.1. (ref. Section 4.3.3.1: Subroutine FFDATA and Appendix B).

TRUEAN is the true anomaly of the spacecraft measured in decimal degrees of arc from periapsis passage in direction of spacecraft motion.

TIMEPR is used to define true anomaly in terms of time; TIMEPR is current time, in hours. If TIMEPR is defined, TRUEAN in decimal degrees is returned to the operations block in common. If TRUEAN is defined, current time is returned to the operations block under the variable name TIMEPR.

NSPFF specifies a step number within which a planetary flux calculation was made. If the FORTRAN statement: PLTYPE = 6HPLSAVE appears

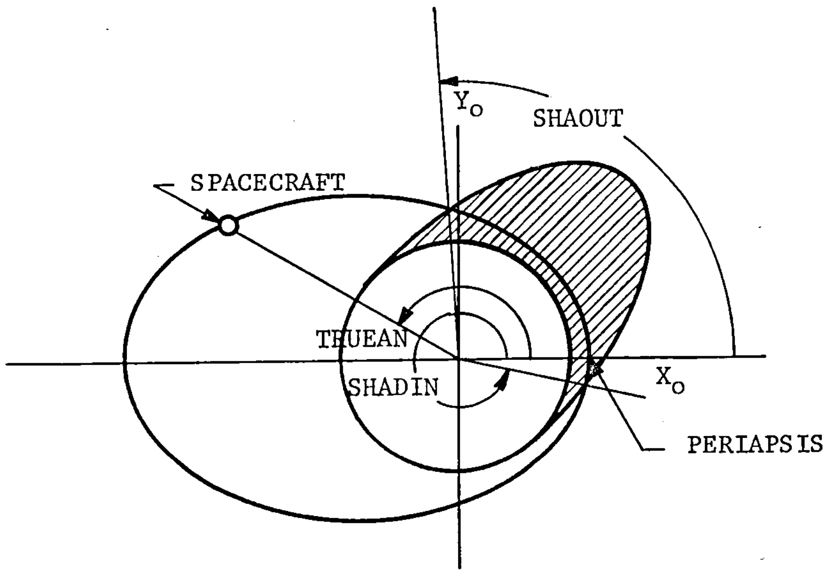


Figure 4-7 Definition of True Anomaly and Shadow Entry/Exit Points

prior to this calculation, the form factor matrix from the spacecraft to planet will be stored out of core under step NSPFF. If this form factor matrix is valid for additional orbit points (i.e., circular orbit, planet oriented), the FORTRAN statement PLTYPE = 6HPLREAD is made in the first step subsequent to NSPFF. This will result in the planet form factor calculations being bypassed for the subsequent steps, with a corresponding saving in computer time. To reduce computation time for Restarts the Form Factor Matrix from the Spacecraft to the Planet is also saved on the RSO tape for circular Orbit-Planet oriented cases, when the CIRP option of the ORBGEN card is used (see Section 3.3.9.2: ORBGEN Option).

DIPNCH is the flag for punching direct irradiation data in the flux data block format. Options are 3HPUN, 2HNO, and 4HTAPE. The tape option will write the DI's to the USER1 tape with formatted writes acceptable to the Flux Data block (see Section 3.3.7: Flux Data). USER1 must be referenced in Option Data block and actively assigned to the run.

ISFAC is the flag that controls whether or not flux shadow factors are written to the RSO tape for subsequent printing in the Direct Incident Link for restart runs. Enter NO to inhibit writing the shadow factors on the Univac version of TRASYS. Enter YES to write the shadow factors on the CDC version.

Subroutine DIDT1S is a short form version of DIDT1 to be used when the user does not desire to specify his accuracy and shadow/no shadow flag. See Appendix D for DIDT1 and DIDT1S argument default values.

4.3.5.5 Subroutines DIDT2 and DIDT2S

```
Calling sequences:  CALL DIDT2 (DINOSH, DIACC, DIACCS, NSPFF, SUNCL,
                    SUNCO, PLCL, PLCO, TIMPEPR, ALT, DIPNCH, ISFAC)
or:
                    CALL DIDT2S (NSPFF, SUNCL, SUNCO, PLCL, PLCO
                    TIMEPR, ALT, DIPNCH, ISFAC)
```

These subroutines are identical in function to DIDT1 and DIDT1S in that they define the shadowing and accuracy parameters to be used in the subsequent direct flux segment execution, as well as furnish the parameters necessary to define the spacecraft's spatial relation with the sun and planet heat sources.

The arguments DINOSH, DIACC, DIACCS, NSPFF and ISFAC are exactly as discussed in Section 4.3.5.4: Subroutines DIDT1 and DIDT1S, and tabulated in Appendix D.

SUNCL, SUNCO, PLCL, and PLCO are the clock and cone angles needed to define the direction of the sun and planet position vectors in vehicle coordinate system 3-space. Figure 4-8: Spacecraft Orientation with Subroutine DIDT2, shows how these parameters are defined. Their input units are decimal degrees of arc.

ALT is the spacecraft altitude, above the planet, this argument must be input in feet.

It should be noted that a DIDT2 call is not sufficient to define all the variables needed for a direct irradiation segment execution. In general, an ORBIT1 or ORBIT2 call must be made, or the variables PRAD, SOL, PALB, WDS, and WSS (ref. Section 4.3.5.1: Subroutine ORBIT1) must be defined individually in operations block FORTRAN statements. The call to ORBIT1 or ORBIT2 need only define PNAME. The remaining arguments may be dummys.

The user should also be aware that spacecraft spin as defined by subroutine SPIN, is not applicable when DIDT2 or DIDT2S are called, since DIDT2 and DIDT2S directly define spacecraft orientation as well as position in space.

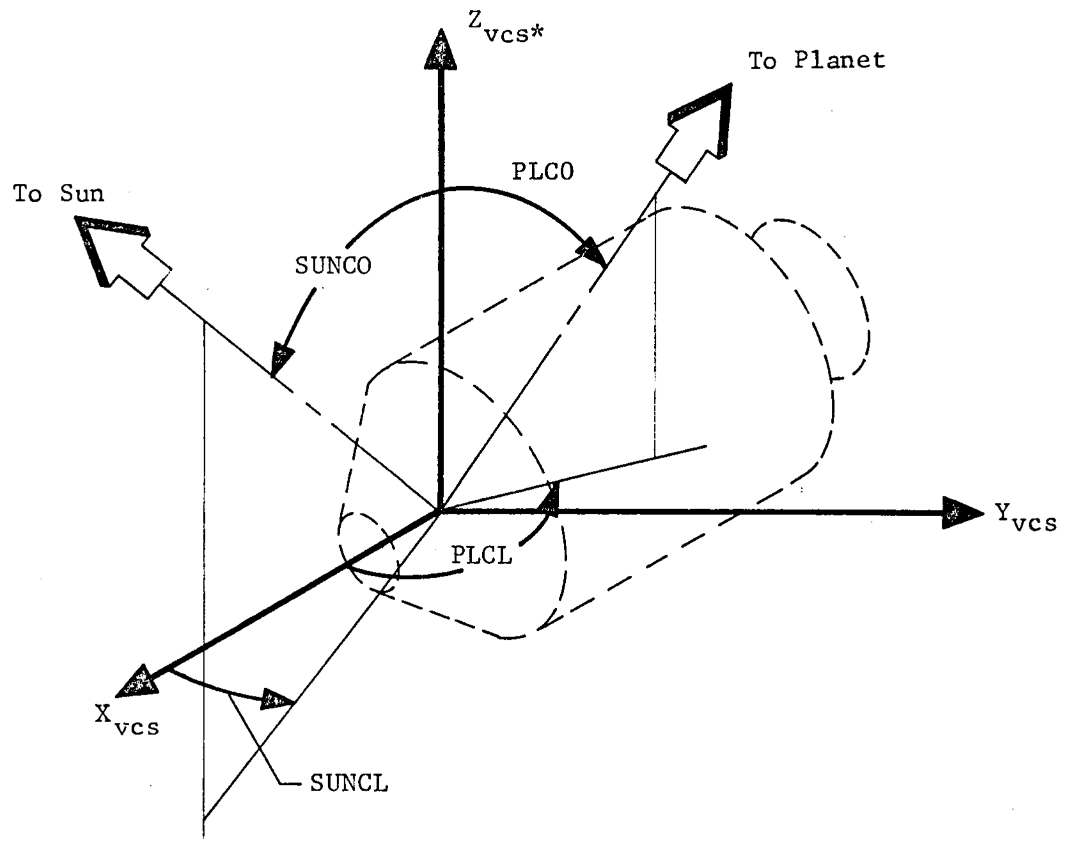


Figure 4-8 Spacecraft Orientation with Subroutine DIDT2

4.3.5.6 Subroutine SPIN

Calling sequence: CALL SPIN (CLOCK, CONE, RATE, TRUANS, SPNTM)

*If subroutine ORIENT is not called prior to DIDT2 or DIDT2S, the vcs and ccs coincide. This is the recommended mode of use. "STAR" is not allowed as an orient type when using DIDT2 or DIDT2S.

This subroutine is used to define spacecraft spin. The arguments CLOCK and CONE define the spin axis with reference to the central coordinate system. RATE defines the spacecraft spin rate about the spin axis in revolutions per hour. Figure 4-9: Spacecraft Spin Definition, illustrates the clock and cone angles, and the algebraic sign convention used with RATE.

The time spin begins is defined through TRUANS or SPNTM. If the user knows the time his spin begins, he specifies it directly as SPNTM. If he knows the true anomaly where it begins he specifies TRUANS and passes zero for SPNTM.

Spacecraft spin computations are done on the basis of the following: the spacecraft is assumed to be in the orientation defined by the last call to subroutine ORIENT at SPNTM. At any subsequent points in time, the spacecraft is reoriented, presuming a constant spin-rate, about the SPIN-defined spin axis, over the time elapsed since SPNTM.

The user should note a restriction when using subroutine SPIN in conjunction with the ORBGEN card: The only allowable ORBGEN options, after a call to SPIN, are PLAN and NOPL. This is because particular orientations are assumed by the CIRP and INER options that are obviously violated by spacecraft spin.

4.3.5.7 Subroutine DICOMP

Calling sequences: CALL DICOMP (ISOLFL, IALBFL, IPLAFL)

This subroutine allows the user to define the logic used in a subsequent DICAL execution. The choice of computing, stuffing from another step, or zeroing out individual solar, albedo, and planetary fluxes is available. See Appendix D: Subroutine DICOMP, for argument definitions.

4.3.5.8 Subroutine DITTP and DITTPS

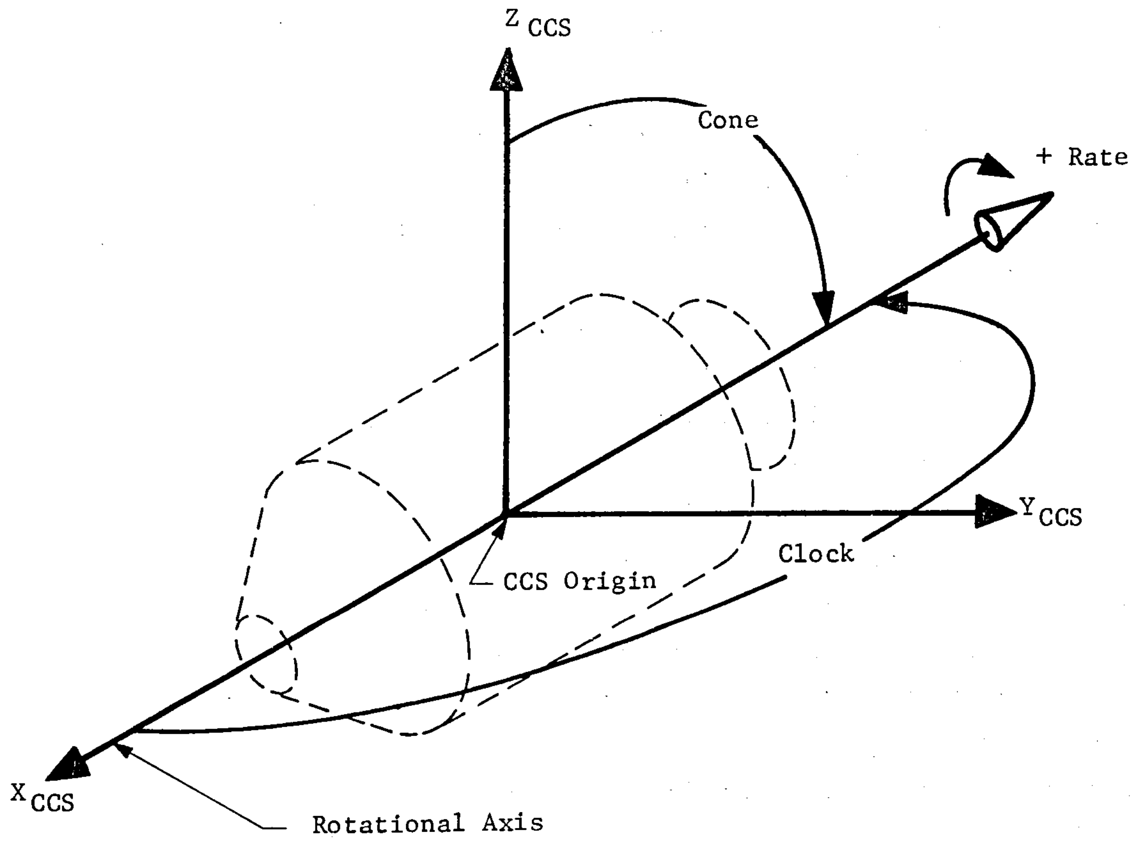
Calling sequences: CALL DITTP (TIME, ITYPE, PLANAM, IDWDN, FIDEN, NTIM, NTYPE, NCLPL, NCOPL, NCLS, NCOS, NRAD, NWOR, ALTMF, IBOD, DIPNCH)

or:

```
CALL DITTPS (TIME, ITYPE)
```

These subroutines allow the user to define his mission by reading trajectory tapes of the attitude timeline variety. The pertinent data are read from the tape and placed in storage for use by the DICAL segment through an internal call to DIDT2.

Subroutine DITTP allows the user to define the trajectory tape format, identify the proper file on multifile tapes, define the attitude parameters for his first compute point, and position the tape for reading subsequent points. Subsequent points are read using DITTPS, which presumes that the tape is previously positioned to the correct file, and a call to DITTPS results in repeated reads of trajectory tape records until a time value equal to TIME is encountered. If ITYPE is defined (as an integer data value), repeated reads are made until TIME is encountered, then reading continues until a special event identifier equal to ITYPE is encountered. Note that this tape reading method precludes calling for a time value less than the time argument used in a previous DITTP or DITTPS call.



<u>Note</u>: Spin axis coordinates illustrated for the case clock = 180, cone = 90

Figure 4-9 Spacecraft Spin Definition

Figure 4-10 is an operations data block segment that generates direct irradiation for three time points, using a trajectory tape. In Step 2, the trajectory tape is positioned to a file named ZLV1 and the data are read at TIME = 10.0 hours, which is not a special event point. The planet involved is Earth, flux output is punched, and the spacecraft altitude data on the tape are in nautical miles (ALTMF = 6080.). Trajectory tape format information is as follows:

a) Tape records are 58 words long (NWOR = 58).
b) Tape is for 1 body (IBOD = 0).
c) File identification is found in word 1 of tape records (IDWDN = 1).
d) Time is found in word 3 of tape records (NTIM = 3).
e) Special event identifier is found in word 5 of tape records (NTYPE = 5).
f) Planet center-to-spacecraft distance is found in word 13 of tape records (NRAD = 13).
g) Clock angle-to-planet vector is found in word 9, (NCLPL = 9).
h) Cone angle-to-planet vector is found in word 10, (NCOPL = 10).
i) Clock angle-to-sun vector is found in word 11, (NCLS = 11).
j) Cone angle-to-sun vector is found in word 12, (NCOS = 12).

Step 2 reads trajectory tape information at time = 10.5 hours, which is not a special event. Step 3 reads trajectory tape information at a special event of type 2, which occurs just subsequent to 11.0 hours.

This routine has not been tested because a compatible trajectory tape is not available. It is anticipated further updating and development would be required to make this subroutine a practical tool for analysis. At NASA JSC the TRJPRT option (see Section 3.3.9.3: TRJPRT Option, was developed since Subroutine DIDTP is not compatible with the Mission Planning and Analysis (MPAD) Common Format Trajectory Tapes utilized for the Space Shuttle project.

```
HEADER OPERATIONS DATA
STEP 1
        CALL BUILDC (ALLBLK,0)
L       FFCAL

L       GBCAL
STEP 2
        CALL DITTP(10.0,0,3HEAR,1,4HZLV1,3,5,9,10,11,12,13,58,6080.,0,
       13HPUN)
L       DICAL
STEP 3
        CALL DITTPS(10.5,0)
L       DICAL
STEP 4
        CALL DITTPS(11.0,2)
L       DICAL
```

Figure 4-10 Trajectory Tape Operations Example

4.3.5.9 Subroutine DRDATA

Calling sequences: CALL DRDATA (NSTPDI, DIACCS)

This subroutine defines the parameters necessary to execute the DRCAL program segment. NSTPDI is the step number that DRCAL will use for retrieving the direct solar fluxes computed by DICAL. DRCAL computes the solar irradiation resulting from one specular bounce. This specular reflection is added to the direct solar flux and the result is stored under the current step number. The diffuse component of the reflected energy is added to the total absorbed heat data later in an AQCAL execution, as in purely diffuse models. DIACCS is the shadowing accuracy parameter. (Reference DIDT1) NSTDPI defaults to the current step number and DIACCS defaults to 0.1. Default values are used by DRCAL if no call is made to DRDATA.

4.3.6 Radiation Interchange Subroutines

4.3.6.1 Subroutine GBDATA

Calling sequence: CALL GBDATA (GBWBND, 6HNFIGFF, NFFTYP)

This subroutine defines the parameters necessary prior to executing the GBCAL segment to obtain a gray-body factor matrix.

Argument GBWBND (Options: 3HSOL, 2HIR, 4HBOTH) defines the energy waveband--solar, infrared, or both--that will be used in gray-body factor calculation.

Argument NFIGFF is the model name under which the form factor matrix desired for gray body calculations is stored. Defaults to current model name.

Argument NFFTYP is the type of form factors stored under NFIGFF. FF for ordinary form factors, CM for combined form factors, and RB for image factors.

4.3.6.2 Approximate Radiant Interchange Factors - Subroutine GBAPRX

A routine is available in the processor library that computes diffuse-gray-body interchange factors according to the first-order approximation:

$$\mathfrak{F}_{ij} = \text{PROPI*PROPJ*}F_{ij}$$

where

PROPI and PROPJ are the diffuse surface properties, solar absorptivity or infrared emissivity; $\mathfrak{F}_{ij}$ is the approximate radiant interchange factor between surfaces i and j; F_{ij} is the form factor. This equation is, of course, exact for a black enclosure (PROPI = PROPJ = 1 for all i, j).

A call to subroutine GBAPRX in lieu of executing the GBCAL segment will generate the approximate gray-body factor data and store them in the same manner as GBCAL. This approach is theoretically correct when there are no reflections (e.g., surfaces are black). The greatest error is for a complete enclosure with very low emittances. The error will be reduced as a configuration becomes more "open" and the emittances become higher. In general the application of this routine will be for very special cases.

Calling sequence: CALL GBAPRX (GBWBND, 6HNFIGFF, NFFTYP)

This subroutine calculates gray-body radiant interchange factors using the approximate relationship described above and stores the results in data storage under the current configuration name.

Argument Name	Description	Options	Default
GBWBND	Waveband definition name	2HIR, 3HSOL 4HBOTH	4HBOTH
NFIGFF	Configuration name for form factor access		Current Config. Name
NFFTYP	Form factor type to be used in GB calculations	2HFF, 2HCM	Last type calculated under NFIGFF

Note: Input zero for default action.

4.3.6.3 Subroutine RKDATA

Calling sequence: CALL RKDATA (NFIGGB, RKPNCH, RKMIN, IRKCN, RKSP,
IRKNSP, SIGMA, RKAMPF, RKTAPE, NFIGCO)

This subroutine defines the parameters necessary prior to executing the RKCAL program segment to obtain radiation conductors (RADKs) in thermal analyzer format. The radiation conductors are not stored on the RSO tape. The computer charges for computing the radiation conductors when IR gray-body factors are known is minimal.

Argument NFIGGB identifies the configuration name under which the desired IR gray-body factor matrix can be accessed for computation of the desired radiation conductors.

Argument RKPNCH (Options: 3HPUN, 2HNO) is the punch/no punch flag for radiation conductors on BCD card format.

Argument RKMIN defines the lower limit of the radiation conductor values that will be punched or put on the BCDOU tape. The way the heat balance is written in most thermal analyzers is the energy exchange for those connections dropped out will be lumped back into node i. The radiation conductor to a space node is never dropped out unless 0.0. RKMIN is defined as follows for a valid radiation conductor:

$$\mathcal{F}_{ij}A_i/\epsilon_i A_i < \text{RKMIN}$$

where

$\mathcal{F}_{ij}$ is the gray-body factor from node i to j,
ϵ_i is the infrared emittance of node i.

Argument IRKCN is the initial radiation conductor identification number. The radiation conductors are numbered consecutively from IRKCN. They can range from 1 to 6 digits.

Arguments RKSP and IRKNSP provide the information to define radiation conductors to space for problems that do not form a complete enclosure. RKSP (Options: 5HSPACE, 2HNO) is the flag for calculation of radiation conductors to space. When RKSP = 5HSPACE, radiation conductors to space for node i are computed according to

$$\mathcal{F}\text{ i-space } {}^*A_i = A_i \epsilon_i - \sum_{J}^{N} (\mathcal{F}\text{ ij} * A_i)$$

for an N-node problem. IRKNSP is the user-defined identification number for his space node.

SIGMA and RKAMPF are available for the user to obtain unit agreement between his radiation model and thermal analyzer model. SIGMA is the Stefan-Boltzmann constant that will appear on the radiation conductor and RKAMPF is an arbitrary multiplication factor available to change from the TRASYS standard area units (square feet) to the area unit the user desires. If RKAMPF is 1.0, the area unit associated with SIGMA must be square feet.

Argument RKTAPE (Options: 4HTAPE, 2HNO) allows the user to write his radiation conductors to the BCDOU tape in a thermal analyzer format.

Argument NFIGCO is the configuration name for correspondence data access.

All RKDATA arguments have default values (see Appendix D) so that an RKDATA call before an RKCAL execution is not mandatory.

4.3.6.4 Radiation Condenser - Subroutine RCDATA

The radiation condenser segment provides the user with two methods of radiation model simplification.

The first of these methods, which is referred to as the Multiple Enclosure Simplification Shield (MESS) technique, allows a complex radiation enclosure to be modularized into discrete sub-enclosures by the assignment of imaginary interface shield nodes. Each of these smaller enclosures can be analyzed independently of the others, resulting in more efficient use of computers and manpower.

The second method, referred to as the Effective Radiation Node (ERN) technique, is used to reduce the number of radiation couplings required to thermally model an enclosure by replacing small conductors from each node with a single conductor coupled to the enclosure ERN.

The techniques and their application are described in more detail in Appendix F.

Calling sequence: CALL RCDATA (NFIGGB, RKPNCH, RKMIN, IRKCN, RKSP, IRKNSP, SIGMA, RKAMPF, RKTAPE, NFIGCO, RFRAC, RTOL, NERN, IPRIME, ISECND)

This is a user-called subroutine that defines the parameters used in RCCAL for the condensation and output of radiation conductors (RADKs).

Variable	Description	Default Value
NFIGGB	Configuration name for IR gray-body factor access	Current config. name
RKPNCH	Punch/No Punch Flag. Options: 3HPUN, 2HNO	3HPUN
RKMIN	Minimun Value of $\mathcal{F}_{ij}\ A_i/A_i\epsilon_i$ that will result in a valid RADK. For ERN couplings, if NERN is positive the RKMIN test is applied to the sum of those connections discarded after the RFRAC requirement is satisfied.	0.0001
IRKCN	Initial Radiation Conductor Number (Conductor numbers may range from 1 to 6 digits)	1
RKSP	Flag for Calculation of RADKs to Space See Section 4.3.5.3: Subroutine RKDATA Options: 5HSPACE, 2HNO	2HNO
IRKNSP	Space Node Number	32767
SIGMA	Stefan-Boltzmann Constant	1.713E-9
RKAMPF	Area Multiplying Factor	1.0
RKTAPE	Flag to write RADKs to BCDOU **Tape.** Options: 4HTAPE, 2HNO	**2HNO**

NFIGCO	Configuration name for Correspondence data access	Current Config. name
RFRAC	Significant Radiation Fraction: Radiation conductors of a node to be left intact divided by the sum of the node's conductors. Ref. Appendix F, Equation 6.	None
RTOL	Percentage of SLAST (last conductor saved to meet RFRAC criterion). Subsequent conductors are saved if their values are greater than RTOL*SLAST.	.99
NERN	Effective Radiation Node (ERN) Number. Any negative value will cause the program to print all ERN conductors but not punch or write them to tape.	None
IPRIME	Array Name for Array of Primary MESS Node Numbers and Special Node Numbers.	None
ISECND	Array Name for Array of Secondary MESS Node Numbers.	None

In the printout for both the RK and RC segment, after all the card images of the radiation conductors are listed, the program prints out two summary tables. The first table is a conservation check of each node in the model with any correspondence data applied before the number of radiation couplings are reduced in these segments. The value printed for each node is $\sum_{j=1}^{n} \mathcal{F}_{ij} A_i / A_i \epsilon_i$. For a complete enclosure theoretically it should be 1.0. If there is a space node it should also be 1.0. The second summary table is the conservation check made after the number of radiation couplings have been reduced. These two tables can be very helpful in finding input errors, and in evaluating the effects of radiation coupling reduction techniques.

Restrictions:

1. RCDATA must be called prior to RCCAL execution because all of the variables are not defaulted.

2. IPRIME and ISECND arrays must be input in the array data block (see Section 3.3.2: Quantities and Array Data Blocks) to specify MESS node pairs and special nodes. IPRIME contains a list of all primary MESS nodes and all special nodes in that order. ISECND contains a list of all secondary MESS nodes in a one-to-one correspondence with the primary MESS nodes in IPRIME.

4.3.7 Absorbed Heat Subroutines

4.3.7.1 Subroutine AQDATA

Calling sequence: CALL AQDATA (IAQGBI, IAQGBS, RSOLAR, RALB, RPLAN)

This subroutine defines the parameters necessary prior to executing the AQCAL program segment to compute absorbed heats.

IAQGBI (Options CONFN, FZEROI) CONFN is the Hollerith configuration name in effect when the appropriate infrared grey-body factors were computed. FZEROI is the zero flag for infrared absorbed heat. When IAQGBI = 4HZERO, all infrared absorbed fluxes will be zero, and no infrared grey-body factor matrix is needed for AQCAL execution.

IAQGBS (Options CONFN, FZEROS) is analogous to IAQGBI for the solar waveband.

RSOLAR is a solar-heat rate multiplying factor (defaults to 1.0). NOTE: Albedo heat rates are multiplied by R solar also.

RALB is an albedo-heat-rate multiplying factor (defaults to 1.0).

RPLAN is a planetary-heat-rate multiplying factor (defaults to 1.0).

Configuration name arguments will default to the current configuration name, so that an AQDATA call is not required if all necessary data are in storage under the current configuration name.

Variable AQPRNT controls whether or not a detailed printout of absorbed heat data is obtained. Detailed prints consist of the direct and reflected components of the absorbed solar, albedo and planetary heat rates with applicable correspondence applied. To activate this print, enter the FORTRAN statement AQPRNT = YES prior to any ORBGEN card or L AQCAL card.

Without AQPRNT the absorbed Q values do not get printed. The normal printout only informs the user that they have been computed. The absorbed heat output segment (QOCAL) is the normal means by which the user obtains a printout of the total absorbed heating rate (see Section 4.3.7.3: Subroutine QODATA).

Although the AQ data values are written to the RSO tape, they are never utilized by a restart run. They are written to the RSO tape (as is other data in this category) to be used by other interface programs. The computational charges for the AQ and QO segments are very minimal once the direct incident and gray body factors are known.

4.3.7.2 Subroutine STFAQ

Calling sequence: CALL STFAQ (TRUEAN, TIMEPR, NSTP)

This subroutine stuffs values of absorbed heat and/or direct flux computed in a previously executed step into out-of-core storage for the current step. It also stores time for the current step, defined either directly or from true anomaly.

The argument NSTP is the step number from which the desired absorbed heat values will be obtained.

The geometry, as defined by BUILDC and ADD calls, in effect at the time any STFAQ call is made must agree exactly with that in effect when step NSTP was executed.

4.3.7.3 Subroutine QODATA

Calling sequence: CALL QODATA (NSARRY, NTMARY, QOTAPE, QOPNCH, QOAMPF, QOFMPF, QOTMPF, QOTYPE)

This subroutine defines the parameters necessary to allow absorbed heat data in thermal analyzer format to be generated in a subsequent QOCAL execution.

Argument NSARRY is the name of an array containing the previously executed step numbers where the desired absorbed-heat data can be found in storage. The Step Numbers in the array do not have to be in numerical or chronological order for the program to work correctly. Unless NSARRY = 3HALL, this array must be entered in the array data block. The user is referred to Section 3.3.2.2.: Quantities and Array Data Block, for examples of ways this array may be defined. If the 3HALL option is used, the step numbers do not have to be listed in an array by the user. All absorbed heat data computed since the last call to QOINIT will be output.

Argument NTMARY is the thermal analyzer array number the user desires for his time array when Q vs time tables are being generated. The Q arrays generated will be numbered consecutively from NTMARY+1.

Arguments QOTAPE and QOPNCH are flags to control the form of Q table output. Options are 4HTAPE, 2HNO for write/no write to the BCDOU tape, and 3HPUN, 2HNO for punch/no punch control.

Arguments QOAMPF, QOFMPF and QOTMPF are the multiplying factors for area, energy, and time, respectively. The default values of 1.0 result in time in hours, area in square feet, and energy in Btu/hr.

Argument QOTYPE controls the type of output obtained. 3HTAB results in Q vs time tables; 2HAV results in an integrated average Q for the time period defined by NSARRY.

4.3.7.4 Subroutine QOINIT

Calling sequence: CALL QOINIT

This subroutine rewinds the file containing the absorbed Q data, thus providing user control of the number of time points obtained with NSARRY = 3HALL.

4.3.8 Data Modification Routines

A series of routines are available that enable the user to change certain types of data from the operations data block. This provides a convenient way to perform many types of parametric studies without the necessity of making multiple runs and error-prone changes to the surface data. If "MOD" subroutines are used and then a new configuration is "built" the initial conditions passed from the preprocessor to the processor will override the previous MOD subroutine calls.

The series of routines allows the following node properties to be changed:

a) Area;
b) Diffuse infrared emissivity and/or solar absorptivity;
c) Specular infrared and/or solar reflectivity;
d) Infrared and/or solar transmissivities;
e) SHADE/BSHADE flags.

Calling sequences are designed so that the properties may be changed for one or all active nodes with one call. The use and function of these routines is explained in the following sections.

4.3.8.1 Subroutine MODAR

Calling sequence: CALL MODAR (ND, AR)

This subroutine changes the area of a designated node or the area of all currently active nodes by use of a multiplier, for utilization in the radiation conductor and QO segments.

If this subroutine is misused it may give erroneous results. The intended and only valid application of this subroutine is very limited. If a node always has the same number and distribution of elements, and the same elements are always shadowed in the FF and DI segments the MODAR approach is correct. In TRASYS, though the number of elements are seldom constant. If the number, however, is always large the error may be minimized. In general it is recommended that the MODAR subroutine be used only when a specific area of this node is always shadowed, and when there is always a sufficient element breakdown to closely define the boundaries of the shadow. The MODAR's do not affect the form factor * area, the script F * area matrices, or the DI's. MODAR does effect the radiation screening process, the radiation conductor value to space, and the absorbed heating rates in the QO segment.

Argument Name	Description	Options	Default
ND	Node Number Designator	a) Any Active Node Number (Integer).	None
		b) 3HALL	
AR	Desired Value for Area	a) Floating-Point Data Value	None
		b) Area Multiplier[1] (3HALL Option Only)	

Note: 1. When ND = 3HALL, all active node areas are modified according to AREA = AREA*AR.

Restriction: Call not valid prior to geometry definition through calls to BUILDC and ADD.

4.3.8.2 Subroutine MODPR

Calling sequence: CALL MODPR (ND, ALPHA, EMISS)

This subroutine modifies the diffuse infrared emissivity and/or the diffuse solar absorptivity of a designated node.

Argument Name	Description	Options	Default
ND	Node Number Designator	Any Active Node Number	None
ALPHA	Diffuse Solar Absorptivity	a) $0. \leq DV \leq 1.$ b) $DV < 0.$[1]	None
EMISS	Diffuse IR Emissivity	a) $0. \leq DV \leq 1.$ b) $DV < 0.$[1]	None

Note: 1. If ALPHA   0. or EMISS   0., current values are not changed.

Restriction: Call not valid prior to geometry definition through calls to BUILDC and ADD.

4.3.8.3 Subroutine MODTR

Calling sequence: CALL MODTR (ISR, TRANS, TRANI)

This subroutine modifies the solar and/or infrared transmissivity of a designated surface.

Argument Name	Description	Options	Default
ISR	Surface Number Designator	Any Active Surface Number	None
TRANS	Solar Transmissivity	a) $0. \leq DV \leq 1.$ b) $DV < 0.$	None
TRANI	IR Transmissivity	a) $0. \leq DV \leq 1.$ b) $DV < 0.$	None

Note: 1. TRANI and TRANS values less than zero are used to correctly account for the transmissivity of double-faced surfaces entered separately. A negative value is not required when the ACTIVE = BOTH option is used in the Surface Data block because only one surface is considered to shadow even if both sides are active (Reference Section 3.3.3.8: Properties Data).

2. Transmissivity changes affect the entire surface.

Restriction: Call not valid prior to geometry definition through calls to BUILDC and ADD.

4.3.8.4 Subroutine MODPRS

Calling sequence: CALL MODPRS (ND, SPRS, SPRI)

This subroutine modifies the solar and/or infrared specular reflectivity of a designated node.

Argument Name	Description	Options	Default
ND	Node Number Designator	Any Active Node Number	None
SPRS	Specular Reflectivity, Solar	a) $0. \leq DV \leq 1.$ b) $DV < 0.$[1]	None
SPRI	Specular Reflectivity, Infrared	a) $0. \leq DV \leq 1.$ b) $DV < 0.$[1]	None

Notes: 1. If SPRI $<$ 0. or SPRS $<$ 0., current values are not changed.

Restrictions:
1. This call is applicable only to nodes defined as specular reflectors in the surface data block.
2. Call not valid prior to geometry definition through calls to BUILDC and ADD.

4.3.8.5 Subroutine MODSHD

Calling sequence: CALL MODSHD (ISR, SHADE, BSHADE)

This subroutine modifies the SHADE/BSHADE flags for a designated surface.

Argument Name	Description	Option	Default
ISR	Surface Number Designator	Any Active Surface Number	None
SHADE	Can Shade Flag	FF, DI, BOTH, NO, 0[1]	None
BSHADE	Can Be Shaded Flag	FF, DI, BOTH, NO, 0[1]	None

NOTE:

1. If SHADE or BSHADE data values are zero, their values are not changed.

2. Shade flag changes affect the entire surface.

Restrictions:

1. Call not valid prior to geometry definition through calls to BUILDC and ADD.

2. Call not applicable to shadower-only surfaces.

4.3.8.6 Subroutine NODDAT

Calling sequence: CALL NODDAT

After a series of calls to the "MOD" routines, it might be desirable to obtain a printout of the nodal properties as a check. NODDAT provides this feature. A call to NODDAT anywhere in the operations data block will produce a printout of the optical properties assigned to each currently active node.

4.3.9 Restart Control Subroutines

Two routines are available in the processor library for use in stopping or resuming the reading of a Permanent Restart Input (RSI) tape during a restart run. The judicious use of RSTOFF and RSTON allows the user to insert new segment calls, delete segment calls or redo any part of the operations data. (Refer to Section 3.3.9.7: Restart Operations)

4.3.9.1 Subroutine RSTOFF

Calling sequence: CALL RSTOFF

A call to this routine stops the reading of data from the RSI file and initiates the processing of the operation data logic following the call.

4.3.9.2 Subroutine RSTON

A call to this routine stops processing of operations data logic and causes the resumption of the reading of data from the RSI file.

4.3.9.3 Application of Restart Control Subroutine

For TRASYS to read all of the data stored in a RSI tape, the user does not have to use subroutines RSTOFF and RSTON. For example, the situation where data is being computed and written to an RSO tape and aborts because of maximum time can be restarted and handled automatically by the program. With an RSI tape the program will always check, unless Subroutine RSTOFF has been called, to see first if there is valid data already available when executing those segments which take significant computer time. If the data can't be found, then the program will begin to compute the data. The data search on the RSI tape is always forward. If there has been a call to RSTOFF previously and then followed by a CALL RSTON just prior to the execution of the current segment of the program , the program will search forward through the pseudo files of each restartable processor segment (see Appendix C) until it comes to an end-of-file or finds the match it is looking for in the header record for the current segment. If a match is not found, then no further data can be read from the tape. The current and all subsequent segment executions will require the data to be computed. The extent of the data search on the RSI, if the current segment is not immediately preceeded by a call to RSTON, will be to advance only to the very next restartable processor pseudo file and look for a match. If one is not found, it will compute the data for the current segment. It will continue through the Operations Data block in sequence, executing each segment call in this manner.

In the case of a restart from an abort, if the Operations Data block is not changed, or at least the sequence of segments which retrieve the data remain the same, the program will read all of the data stored by the aborted run and bypass the computation of the restartable processor segments until it can't read anymore. At this point if it reads an end-of-file or the variable RSREC (see Section 3.2.1: Options Data Block) is less than the number of the last record read, the program will begin computing any remaining operations data computations. If the end-of-file is reached, the computation would have picked up right where it last wrote to the RSO tape. An end-of-file should always be at the end, even from an aborted run, because the program always puts a end-of-file after each write to the RSO. Just before the next write it will backspace over it and write the new data followed by a new end-of-file.

The RSTON and RSTOFF subroutines allow the user to skip over data on the RSO tape that is not wanted or is known to be bad. It will also allow the insertion of new segment executions in the Operations Data block while selectively retrieving the data available on the RSO tape. The user must be very cautious in retrieving data. What may be considered a match by the program may not really be a match. If the same model and configuration name is used, and the node array hasn't been changed, then for example, even though the form factors need to be recomputed because previous input errors have been corrected, or modeling changes have been incorporated, the program may still utilize the form factors data on the RSI that is known to be in error. In a like manner, a complete irrelevent set of DI data steps may be inadvertently retrieved from the run that created the RSI tape, even though the orientation and/or orbit definition may be different in the current run. An acceptable program match would occur if the same model name, configuration name, and master node array has been retained. In addition, the step number(s) in the DI segments would have to be consistent. These risks exist of course, because of the extreme flexibility of the program to conveniently retrieve all useful data. In the examples cited, the user must make a call to subroutine RSTOFF prior to the segment call where the existing RSI data should be bypassed.

The following is an example of a situation that may occur. First, assume an RSO tape is created by a run that went to completion with the following Operations Data block.

```
HEADER OPERATIONS DATA
BUILD
L      NPLOT
L      NFFCAL
       CALL GBDATA (2HIR,5HTHING,0)
L      GBCAL
       CALL RKDATA (5HTHING,0,0,0,3HYES,0,0,0,0,0)
L      RKCAL
END OF DATA
```

On a subsequent run, the RSO tape will be used as an RSI tape so the form factor and IR gray body matrices can be utilized to compute some orbital heating rates that are desired. If the following Operations Data block is used, there should be no problem.

```
HEADER OPERATIONS DATA
BUILD
L       NPLOT
L       NFFCAL
        CALL GBDATA(2HIR,5HTHING,0)
L       GBCAL
        CALL GBDATA(3HSOL,5HTHING,0)
L       GBCAL
        CALL ORBIT2(3HEAR,0.,90.,0,0,0.,150.*6080.,150.*6080.)
        CALL ORIENT(4HPLAN,1,2,3,0.,90.,90.)
ORBGEN PLAN,0.,360.,12, AQ
END OF DATA
```

The node plot segment would ignore the RSI tape plots since this is not a restartable processor segment. The form factor * area matrix and the IR gray body matrix will be obtained from the RSI file since they occur on the RSI tape in the same order as the current Operations Data block. In the original run, the IR gray bodies are the last segment which wrote to the RSO tape (RKCAL does not), so no more restartable data appears on the tape. The remaining segment calls (generated by the ORBGEN card) in the current Operation Data block will result in computations.

If the user however, reasoned that the program really doesn't need the form factors until it is ready to compute the solar gray bodies and, instead of the preceeding, used the following Operations Data block, a program abort would occur.

```
HEADER OPERATIONS DATA
BUILD
          CALL GBDATA(2HIR,5HTHING,0)
L         GBCAL
L         NFFCAL
          GBDATA(3HSOL,5HTHING,0)
L         GBCAL
          CALL ORBIT2(3HEAR,0.,90.,0,0,0.,150.*6080.,150.*6080.)
          CALL ORIENT(4HPLAN,1,2,3,0.,90.,90.)
ORBGEN    PLAN,0.,360.,12,AQ
END OF DATA
```

The abort would occur because the program would check the RSI file to see if IR gray bodies are available. Although IR gray bodies are present on the tape, it would not look past the form factor psuedo file so a mismatch occurs. It would then attempt to compute the IR gray bodies, but would quickly abort because the form factor * area matrix, which it must have if it is to compute the gray bodies, has not been loaded or computed at this point.

The following Operations Data block would not result in an aborted run, but also has a serious shortcoming:

```
HEADER OPERATIONS DATA
BUILD
          CALL RSTON
          CALL GBDATA(2HIR,5HTHING,0)
L         GBCAL
L         NFFCAL
          CALL GBDATA(3HSOL,5HTHING,0)
L         GBCAL
          CALL ORBIT2(3HEAR,0.,90.,0,0,0.,150.*6080.,150.*6080.)
          CALL ORIENT(4HPLAN,1,2,3,0.,90.,90.)
ORBGEN    PLAN,0.,360.,12,AQ
END OF DATA
```

In the first GB call, because of the RSTON subroutine call preceeding it, the program will not limit itself to look for a match in the header record of only the next available pseudo-file on the RSI tape. The program will search forward until it finds a match or the end-of-file. In this situation, it will find the IR gray bodies in the second psuedo file. When the form factor segment is executed, it will already be at the last psuedo file, so the program will recompute the form factors. The program would continue having to compute any remaining data. The form factor recomputation was unnecessary and, therefore, the first restart approach should be used.

4.3.9.4 Subroutine FFREAD

Calling sequence: CALL FFREAD

When a complete set of form factors are on the RSI tape, a call to FFREAD in lieu of an L FFCAL or an L NFFCAL card will read the form factors on RSI into program data storage. This subroutine eliminates the need for loading the form factor segments. The subroutine is primarily proved to save core in the instructional banks and may provide additional core for data if the loading the alternative FF segment would have been the largest instructional bank. (This feature is available on the UNIVAC version only).

Restrictions: The form factor matrices on the restart tape must be complete, and no form factor data block information will be used for the current configuration name.

4.3.9.5 Subroutine DIREAD

When a complete set of direct flux data for an orbit point is on the RSI tape, a call to DIREAD in lieu of an L DICAL card will read the flux information into current-step data storage. This routine is similar to FFREAD but is for flux data. The subroutine eliminates the need for loading the DI segment. The subroutine is provided to save core in the instructional banks and may provide additional core for data if loading the alternative DI segment would have been the largest instructional bank. If the DIR Option is used for the IFO variable in ORBGEN, DIREAD is called in each step by the program. (This feature is available on the UNIVAC version only).

Restrictions: Flux data for the step must be complete and no flux data block information will be used for the current configuration name and step number.

4.3.10 Planet Surface Subroutines

Subroutines SURFP, DIDT3, and DIDT3S provide a package that allows the user to locate his configuration on the surface of a rotating planet. Solar flux histories for a planetary day may then be computed by updating time of day and executing the DICAL segment. Planet rotation data and above atmosphere solar constants are provided internally for Earth, Mars, and the moon. The user controls atmospheric attenuation of the solar flux through a program variable called the atmospheric extinction factor. DICAL automatically avoids planetary albedo and infrared flux calculations for this option. A ground plane of arbitrary size and appropriate optical properties should be included in the configuration. Solar reflection from this plane is analogous to albedo flux, and infrared energy interchange calculations with this plane in the thermal analyzer will account for the infrared environment.

4.3.10.1 Subroutine SURFP

Calling sequence: CALL SURFP (PNAME, ALAT, SUNLAT, AEX)

This subroutine spatially locates the sun vector relative to the configuration on the rotating planet, defines planet rotation rate, and sets variables used to compute the solar "constant" as attenuated by the atmosphere as a function of the time of day. A change in any of the SURFP arguments must be accomplished through a call to SURFP.

PNAME (options 3HEAR, 3HMAR, 3HMOO) defines the following variables:

SOLO -	solar constant outside the atmosphere, BTU/hr ft^2 (429.0 for Earth and moon, 183.2 for Mars)
PER -	Rotation period of planet, hours

If a value other than the built in nominal for SOLO is desired, SOLO is redefined as desired in the operations data block subsequent to the SURFP call.

ALAT is the latitude of the configuration on the planet, degrees of arc.

SUNLAT is the solar declination in degrees of arc. If PNAME = 3HEAR, SUNLAT must be a day of year ($1. \leq$ SUNLAT $\leq 365.$) so that solar declination can be computed from an ephemeris equation.

AEX is the atmospheric extinction factor that appears in the equation:

SOL = SOLO/EXP (AEX/COS(PHI))

where: SOL = attenuated solar "constant"
SOLO = solar constant outside atmosphere
PHI = angle from local vertical to sun vector

Values for AEX may be found in the growing literature on solar energy conversion. Representative values are 0.25 for the southwest desert and 0.45 for southeastern coastal areas. An alternative to entering AEX is to enter a flux level, in Btu per hr-ft^2 that is desired at solar noon (300 Btu/hr-ft^2 is typical). The argument AEX is tested by subroutine SURFP. If greater than 1.0, noon solar flux is assumed, and a corresponding extinction coefficient is computed and stored for use.

Note: The TRASYS approach to the planet surface environment does not yet include diffuse sources of solar energy such as clouds and atmospheric backscatter.

A call to SURFP computes the two program variables DAWN and DUSK, the times of day for sunrise and sunset, respectively. These may be used as points in the heat flux tables in the same way as SHADIN and SHAOUT for orbiting configurations.

4.3.10.2 Subroutines DIDT3 and DIDT3S

Calling sequences: CALL DIDT3 (DINOSH, DIACCS, ITOD, DIPNCH, ISFAC)
CALL DIDT3S (ITOD, ISFAC)

These subroutines provide accuracy parameters and control flag definition for the DICAL segment when using the planet surface option.

DINOSH, DIACCS, DIPNCH and ISFAC are the usual direct irradiation accuracy parameters and control flags. (Ref. Section 4.3.5.4: Subroutines DIDT1, DIDT1S) ITOD is the time of day since midnight, hours/minutes, 4-digit integer (e.g., 1537 for 3:37 PM, 0820 for 8:20 AM). Time of day may be input in decimal hours through the operations block statement TIMEPR = DV. TIMEPR may also have "DUSK" and "DAWN" as data values. TIMEPR is the time of day since midnight, hours.

4.3.10.3 Orientation on Planet Surface

The planet surface option requires that the central coordinate system of the problem geometry be oriented as follows:

a) the ccs z-axis is the local vertical
b) the ccs y-axis points true north
c) the ccs x-axis points due east

No capability exists to orient the configuration through subroutine ORIENT. If re-orientation is required, the block coordinate system capability may be used.

5. PROCESSOR SEGMENTS

5.1 Pictorial Plot Segments

5.1.1 Node Plotter

Calling sequence: L NPLOT

This segment provides the user with 3-dimensional and/or orthographic projection pictorial plots of his problem geometry. Its primary use is to verify surface data input prior to proceeding with computations of radiation interchange or absorbed heat data. Examples of its ouput can be found in Appendix H.

This segment has no provision for user intervention in the form of program called subroutines that the user may modify. Control is provided through the NDATA or NDATAS subroutines.

5.1.2 Orbit Plotter

Calling sequence: L OPLOT

This segment provides the user with a pictorial representation of his spacecraft in relation to the body it orbits and the sun.

The planet and its shadow are depicted, together with a pictorial view of the spacecraft in orbit. The standard output enables the user to verify his orbit in relation to the sun, and spacecraft orientation relative to the sun, planet, or star. Examples of its output can be found in Appendix H.

This segment has no provision for user intervention beyond that provided by subroutines ODATA and ODATAS.

5.1.3 Data Plotter

Calling sequence: L PLOT

This segment provides the capability to plot any computed or input data as x versus y plots. The segment automatically writes a binary plot data unit (disc or drum) for producing plots of incident or absorbed heat rates or fluxes as a function of time.

The segment also provides a completely general plot capability if the user inputs operations data block FORTRAN to prepare the plot data unit prior to executing the PLOT segment. This type of plot operation is illustrated by the plot unit format described below.

The plot segment flow diagram is shown in Figure 5-1.

5.1.4 Binary Plot Unit Format

Write format: NAME, N, (DATA (I), I = 1, N)

where:

NAME:	type of record
N :	number of words in data array
DATA:	array of data

Record 1, TYPE = FRAME

word 1	5HFRAME
2	4
3	XMIN
4	XMAX
5	YMIN
6	YMAX

Record 2, TYPE = LABELX

word 1	6HLABELX	
2	MAXIMUM OF 5	(30 characters, maximum, per word)
3	LABEL ARRAY	(1)
4		(2)
5		(3)
6		(4)
7		(5)

Record 3, TYPE = LABELY

word 1	6HLABELY	
2	N(MAXIMUM OF 7, 30 characters, maximum, per word)	
3	LABEL ARRAY	(1)
4		(2)
5		(3)
6		(4)
7		(5)

Record 4, TYPE = NODENO

word 1	6HNODENO
2	1
3	INTEGER NODE #

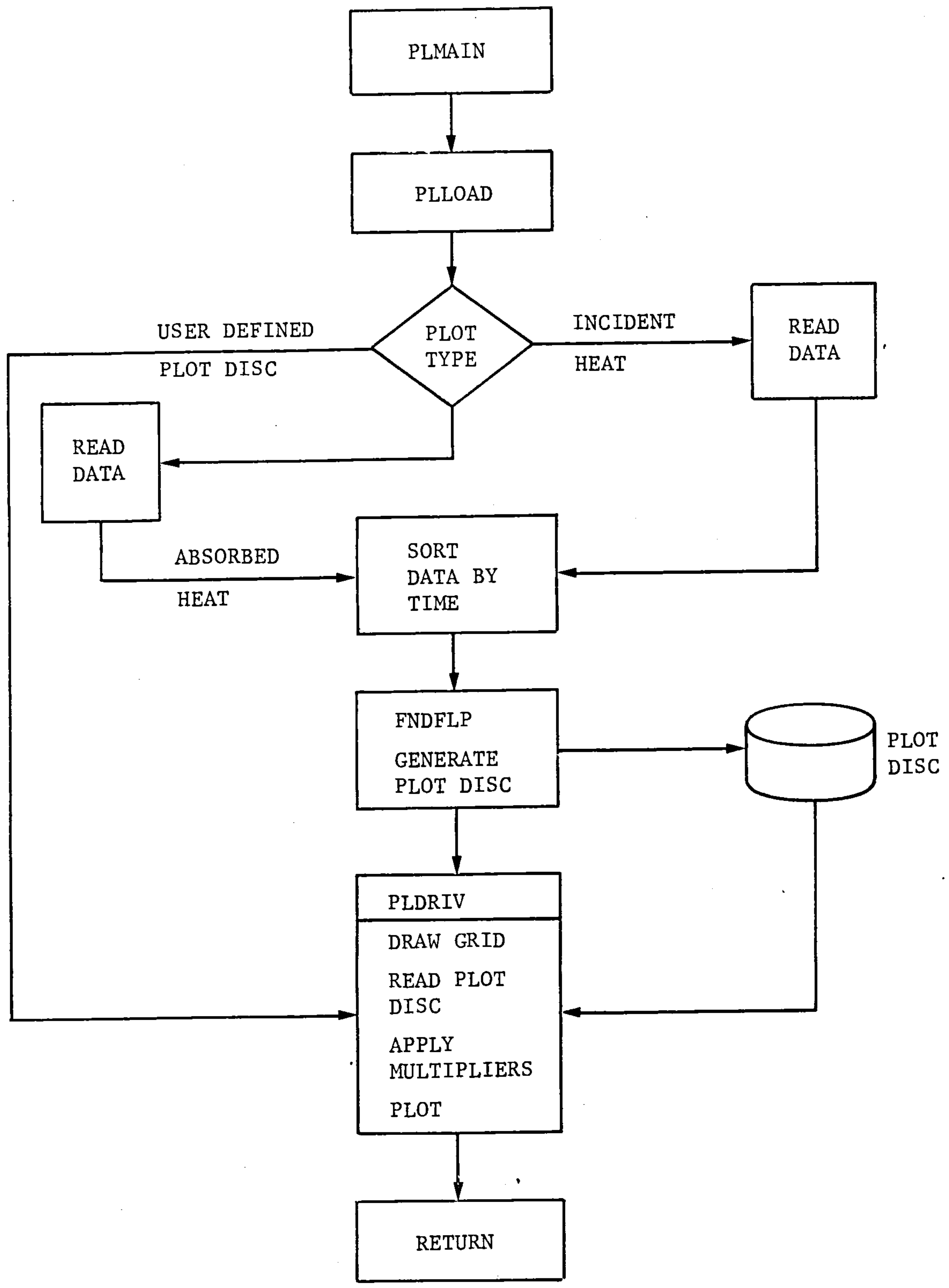


Figure 5-1 PLOT Segment Flow Diagram

```
Record 5,  TYPE = TITLE 1

        word 1  6HTITLE 1
             2  N(MAXIMUM OF 10, 60 characters, maximum, per word)

             3  TITLE ARRAY    1
             4                 2
             .                 .
             .                 .
             .                 .
            12                10

Record 6, TYPE = TITLE 2

        word 1  6HTITLE 2
             2  N(MAXIMUM OF 12, 72 character, maximum, per word)

             3  TITLE ARRAY          1
             4                       2
             .                       .
             .                       .
             .                       .
            14                      12

Record 7, TYPE = INDEP, N = user supplied

        word 1  5HINDEP
             2  N (1 ≤ N ≤ 1000)
             3  DATA (1)
             .
             .
             .
           N+2  DATA (N)
```

Record 8, TYPE = DEPEND, N = user supplied. (must agree with number of independent data values)

word 1 6HDEPEND
2 N $(1 \leq N \leq 1000)$

Note: a) TYPE DEPEND can occur as many times on a file as desired for multiple plots on a frame.

b) Any records but FRAME, INDEP and DEPEND types may be omitted.

c) Record 8, words 3 through N + 2 same as record 7.

5.2 Form Factor Segment

Calling sequence: L FFCAL

This segment computes form factor matrices for any geometric enclosure using a numerical integration method. Internode blockage is accounted for with differing solar and IR transmissivities and, because semitransparent and specular surfaces are allowed, two form factor matrices are computed: FFS for the solar waveband and FFI for the infrared waveband.

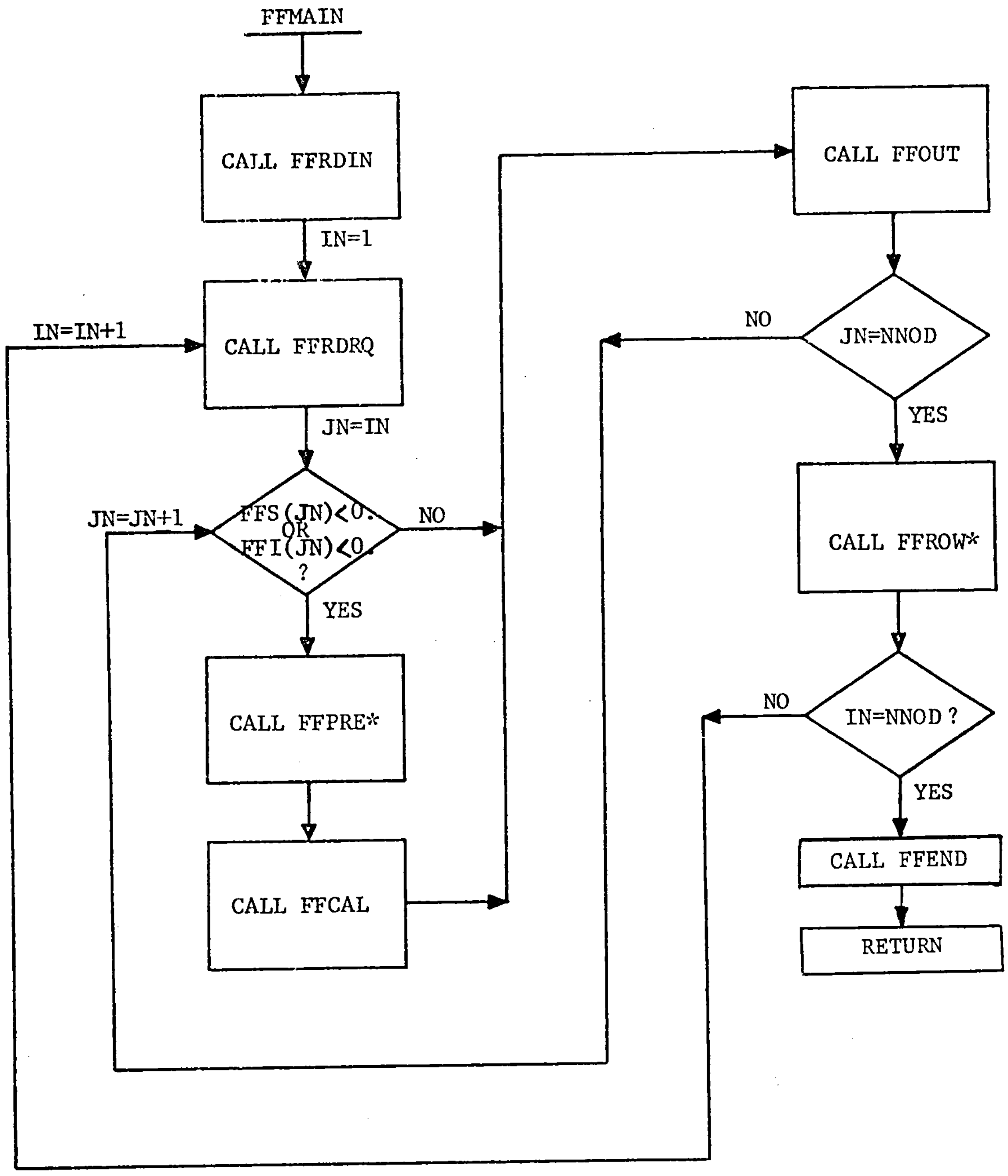
Blockage factors are also computed, printed and written to the restart tape. Blockage factors are defined as follows:

$$BFij = \frac{Fij\ (\text{shadowed})}{Fij\ (\text{unshadowed})}$$

Provision for user intervention via the subroutine data block is available through three program-called subroutines: (1) prior to computation of each form factor through subroutine FFPRE, (2) at the completion of a row of form factors through subroutine FFROW, and (3) at the completion of the entire matrix through subroutine FFEND. The logic flow of the FFCAL segment is shown in Figure 5-2.

FFCAL provides printed output, as well as punched cards or tape in FORM FACTOR DATA Block format, at the user's option.

FFCAL employs two techniques to avoid inaccuracies inherent to the double-integration form factor computation technique. First, it tests the node areas involved before beginning a computation, and makes certain that the factors are always computed from the smaller node to the larger node. Second, if an area-distance criteria is violated, nodes are temporarily subdivided into sub-nodes for form factor calculation. See Appendix B for more detail on this subject.



* USER ROUTINES

Figure 5-2 Segment FFCAL Flow Diagram

5.3 Precision Form Factor Segment

Calling sequence: L NFFCAL

This segment computes form factor matrices for any geometric enclosure using a modified double integration method wherein each elemental form factor is computed using the Nusselt-sphere calculation method. The Nusselt-sphere technique is generally implemented[1] as a single integration method. This leads to extreme computation difficulties when accounting for shadowing, so the technique used in FFCAL, wherein the j-node as well as the i-node are broken down into elements, is retained. With this approach, shadowing calculations proceed in the same way as FFCAL, that is, with a check to see if the element-to-element vectors are interrupted by a shadowing surface. The Nusselt-sphere method, however, eliminates the inaccuracies encountered with FFCAL when closely adjacent surfaces are involved.

The element selection technique used by NFFCAL is basically user controlled. A specified number of elements is divided among the i and j nodes according to their areas. Along with the improved accuracy some increase in run time, compared to FFCAL, may be noted.

The NFFCAL segment interfaces with other program segments and the restart tapes in exactly the same way as FFCAL. The printed output is also very much the same. At the level presented in Figure 5-2, the logic flow of the NFFCAL link is also the same as FFCAL.

---

[1] K. A. Toups, A General Computer Program for the Determination of Radiant Interchange Configuration and Form Factors, North American Aviation Inc., SID 65-1043-2, October, 1965.

5.4 Radiation Interchange Segment

Calling sequences: L GBCAL

Segment GBCAL computes a matrix of diffuse gray-body (GB) radiation interchange factors and places them in out-of-core storage for later use in computing absorbed heat or radiation conductors. Solutions for either the solar and/or infrared wave bands may be requested. No user intervention provisions are made beyond that of subroutine GBDATA. The IR gray body solution is dependent upon form factors and emittances and the solar gray body solution is dependent upon the form factors and solar absorptivity. To compute the gray bodies, the Operations Data Block should call the appropriate subroutines to compute or load the form factors prior to the first GB segment call.

5.5 Radiation Conductor Segment

Calling sequence: L RKCAL

Segment RKCAL computes radiation conductors for thermal analyzer models and provides output in punched card or BCD tape form for direct input to a thermal analyzer. A printout of the card/tape record images is also provided. Three program-called user routines are used to provide user intervention through his subroutines block. Subroutine RKPRE provides for any special initialization desired before computations begin. Subroutine RKPNCH performs the actual punch and tape write operations to the BCDOU file. The user may obtain data in any thermal analyzer program format by altering format statements in this routine. User routine RKEND provides for user intervention prior to return to operations block control. Figure 5-3 shows segment RKCAL logic flow. To compute the radiation conductors the IR gray bodies must be computed or loaded in the Operations Data block prior to the first call to the RK segment.

An example of RKCAL output can be found in Appendix H.

5.6 Radiation Condenser Segment

Calling sequence: L RCCAL

Segment RCCAL computes radiation conductors, simplifies and condenses these conductors using the ERN and MESS techniques, and provides output in punched card and/or BCD tape form for direct input to thermal analyzer. A printout of the card/tape record images, as well as the original (uncondensed) RADKs, is also provided. Three program-called user routines are used to provide user intervention through his subroutines block. Subroutine RCPRE provides for any special initialization desired before computations begin. Subroutine RCPNCH performs the actual punch and tape write operations to the BCDOU file. The user may obtain data in any thermal analyzer program format by altering format statements in this routine. User routine RCEND provides for user intervention prior to return to operations block control. Figure 5-4 shows segment RCCAL logic flow. RCCAL theory is presented in Appendix F. To compute the radiation conductors the IR gray bodies must be computed or loaded in the Operational Data block prior to the first call to the RC segment.

5.6.1 Sample Problem Using ERN/MESS Technique

The optics housing of the High Altitude Observatory (HAO) solar telescope, which is mounted on the Skylab Apollo Telescope Mount, is shown in Figure 5-5. Both the original enclosure and the modularized enclosure are shown along with the ERNs and the MESS nodes. Figure 5-6 shows the nodal breakdown for the enclosure.

TRASYS input for subenclosure 1 (see Figure 5-6) is shown in Figure 5-7.

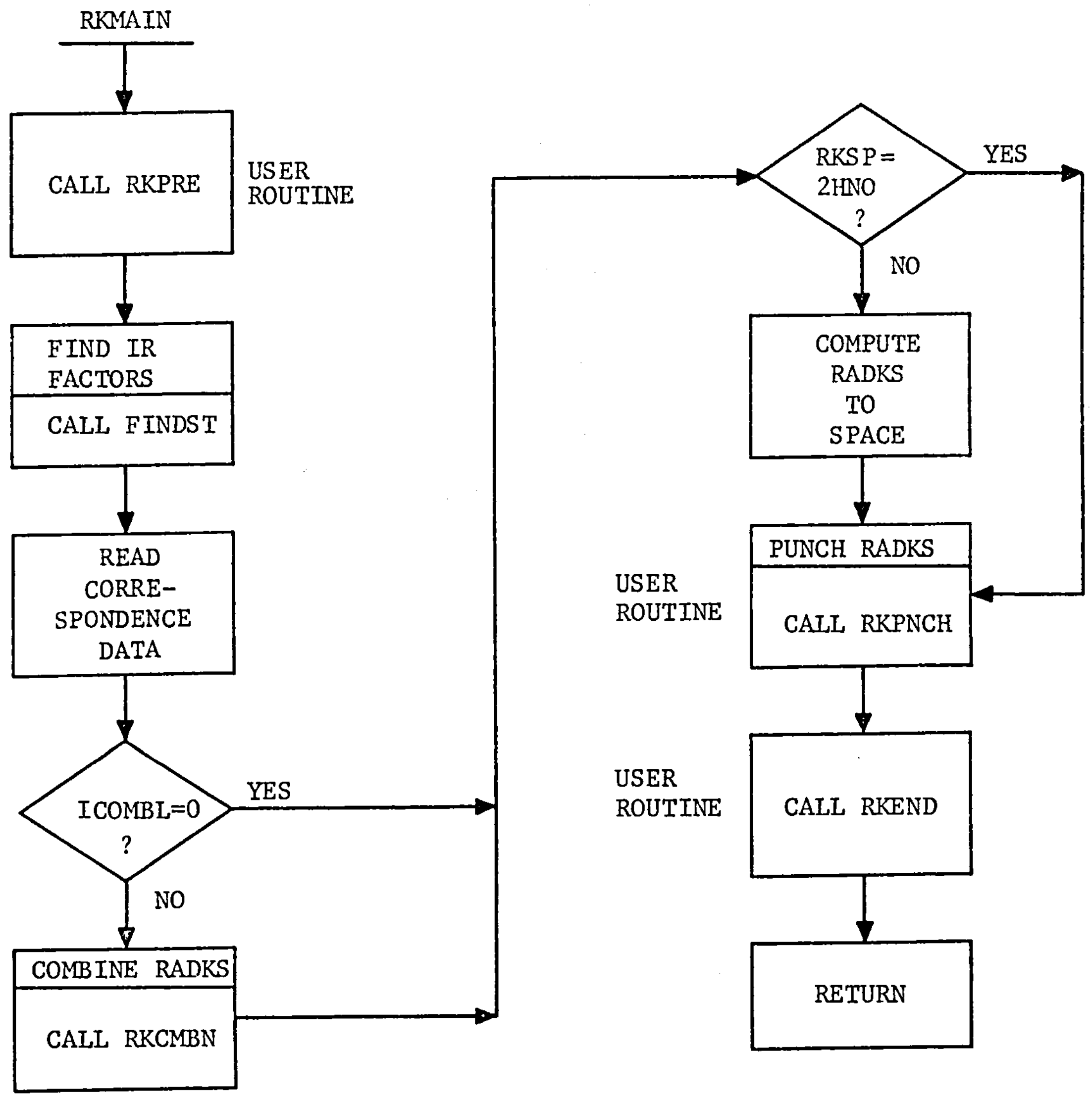
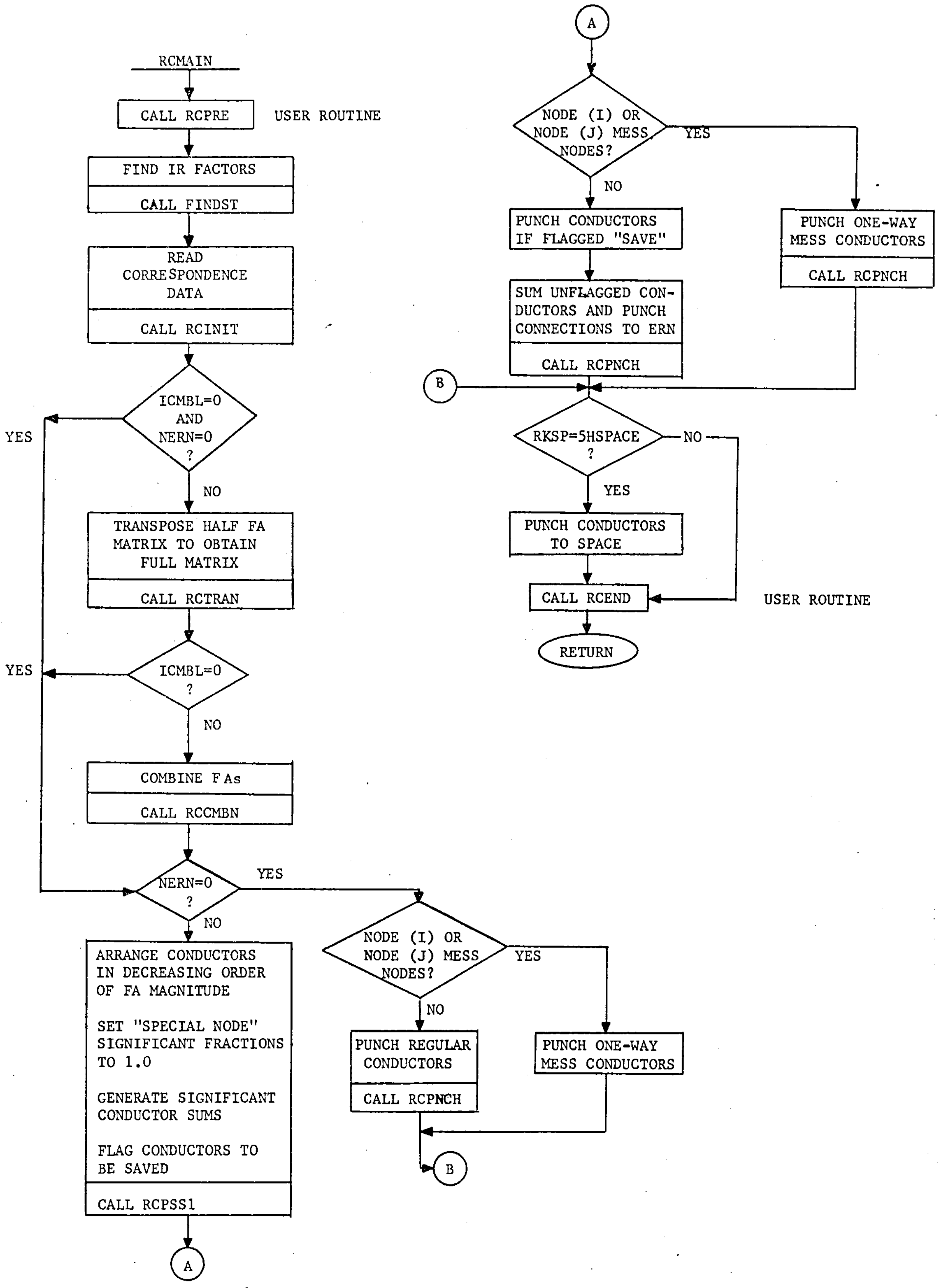


Figure 5-3 Segment RKCAL Flow Diagram



RCMAIN
CALL RCPRE
USER ROUTINE
FIND IR FACTORS
CALL FINDST
READ CORRESPONDENCE DATA
CALL RCINIT
ICMBL=0 AND NERN=0 ?
YES
NO
TRANSPOSE HALF FA MATRIX TO OBTAIN FULL MATRIX
CALL RCTRAN
ICMBL=0 ?
YES
NO
COMBINE FAs
CALL RCCMBN
NERN=0 ?
YES
NO
ARRANGE CONDUCTORS IN DECREASING ORDER OF FA MAGNITUDE
SET "SPECIAL NODE" SIGNIFICANT FRACTIONS TO 1.0
GENERATE SIGNIFICANT CONDUCTOR SUMS
FLAG CONDUCTORS TO BE SAVED
CALL RCPSS1
A
NODE (I) OR NODE (J) MESS NODES?
YES
NO
PUNCH REGULAR CONDUCTORS
CALL RCPNCH
PUNCH ONE-WAY MESS CONDUCTORS
B
A
NODE (I) OR NODE (J) MESS NODES?
YES
NO
PUNCH CONDUCTORS IF FLAGGED "SAVE"
SUM UNFLAGGED CONDUCTORS AND PUNCH CONNECTIONS TO ERN
CALL RCPNCH
PUNCH ONE-WAY MESS CONDUCTORS
CALL RCPNCH
B
RKSP=5HSPACE ?
NO
YES
PUNCH CONDUCTORS TO SPACE
CALL RCEND
USER ROUTINE
RETURN

Figure 5-4 Segment RCCAL Flow Diagram

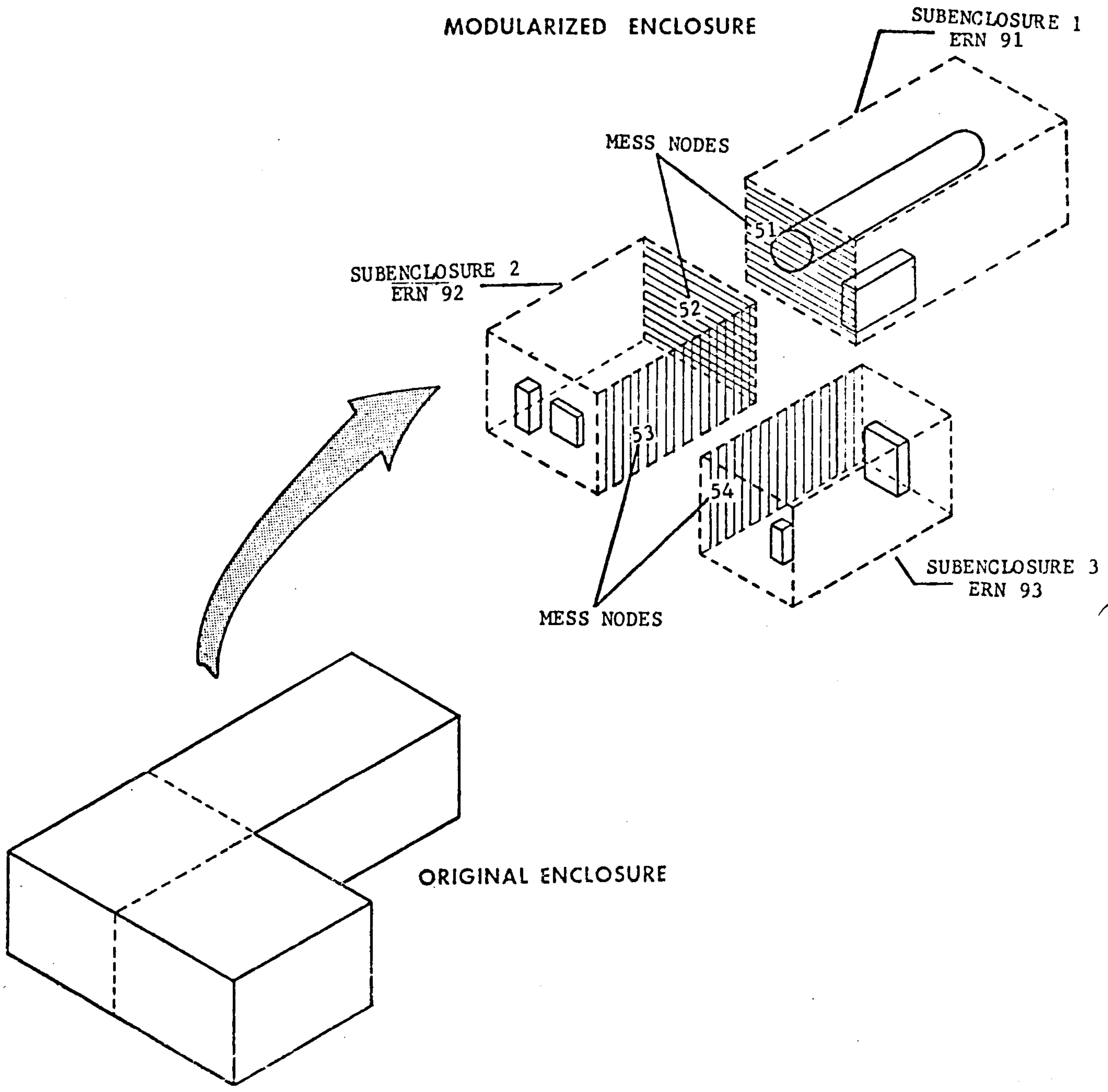


Figure 5-5 HAO Experiment Optics Housing Modularized Enclosures

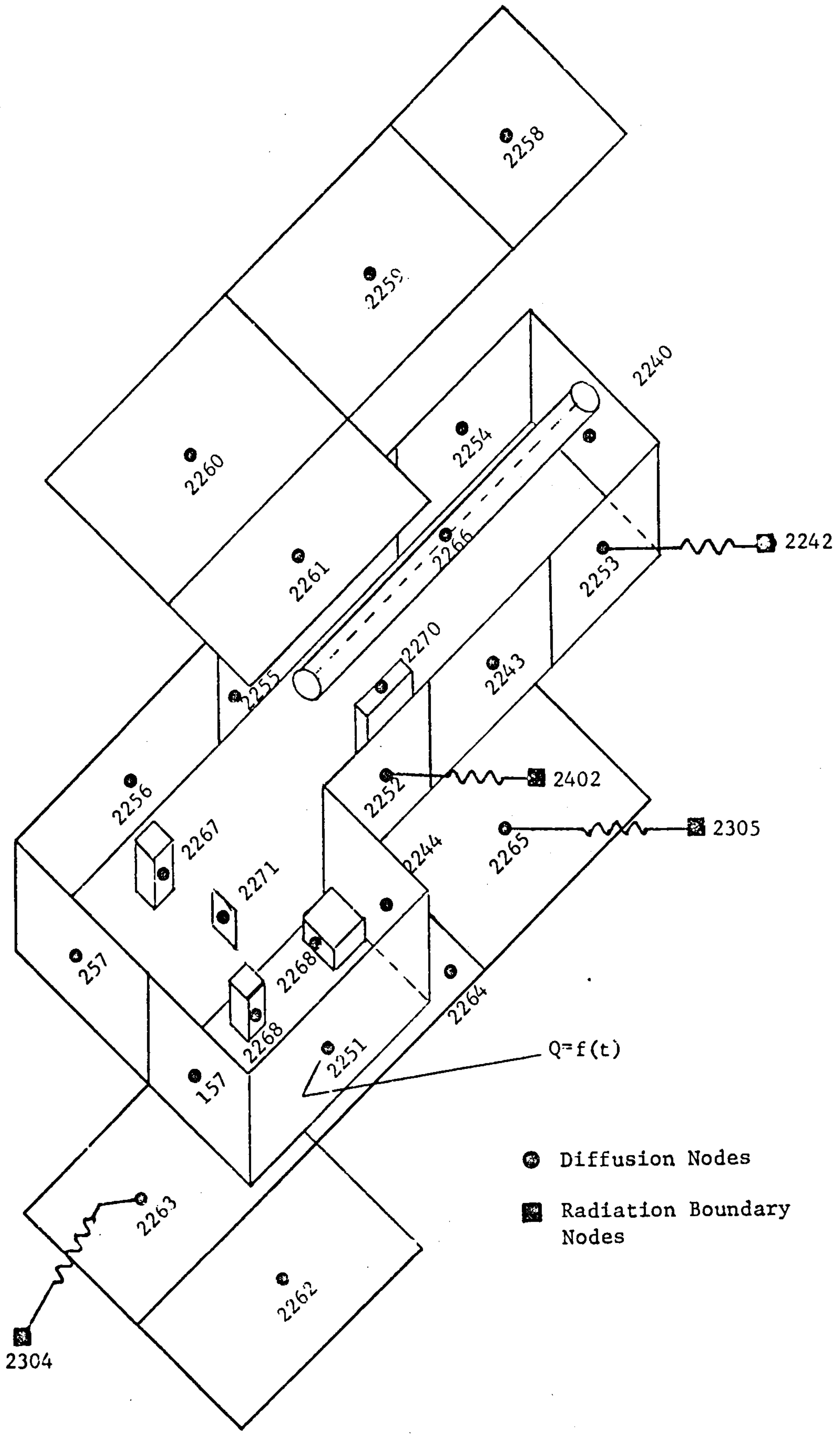


Figure 5-6 Apollo Telescope Mount HAO Experiment Optics Housing Sample Problem

```
HEADER  OPTIONS DATA
TITLE   RADIATION CONDENSER SAMPLE PROBLEM
        MODEL          =HAO
HEADER  ARRAY DATA
        IPPIME         =51
        ISECND         =52
HEADER  SURFACE DATA
S       SURFN          =1
        TYPE           =RECT
        ACTIVE         =TOP
        SHADE          =FF
        BSHADE         =FF
        P1             =10.,9.,0.
        PROP           =0.1,0.1
S       SURFN          =2
        TYPE           =RECT
        ACTIVE         =TOP
        SHADE          =FF
        BSHADE         =FF
        P1             =10.,77.92879,0.
        NNY            =15
        UNNY           =1.3125,2.2875,3.2625,4.1725,11.351,18.5575,25.5045,
                  28.6545,31.6295,37.6495,45.5588,54.4125,63.2318,72.0855
        PROP           =0.9,0.9
HEADER  FORM FACTOR DATA
FIG     HAO

NODEA   1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,END
        1,  2,  0.037798 *  90. $
        1,  3,  0.005768 *  90. $
        1,  4,  0.006289 *  90. $
        1,  6,  0.104575 *  90. $
        1,  7,  0.118291 *  90. $
        1,  8,  0.060140 *  90. $
        1,  9,  0.043028 *  90. $
        1, 10,  0.032649 *  90. $
        1, 11,  0.027951 *  90. $
        1, 12,  0.119425 *  90. $
        1, 13,  0.161426 *  90. $
        1, 14,  0.106845 *  90. $
        1, 15,  0.119568 *  90. $
        1, 16,  0.053506 *  90. $
        2,  6,  0.160944 * 13.125 $
        2,  7,  0.040225 * 13.125 $
        2, 11,  0.043852 * 13.125 $
        2, 12,  0.216057 * 13.125 $
        2, 13,  0.020062 * 13.125 $
        2, 14,  0.115054 * 13.125 $
        2, 15,  0.025691 * 13.125 $
        2, 16,  0.007440 * 13.125 $
        3,  6,  0.021628 *  9.75 $
        3,  8,  0.280861 *  9.75 $
        3, 11,  0.300885 *  9.75 $
        3, 12,  0.174021 *  9.75 $
        3, 14,  0.081130 *  9.75 $
        4,  5,  0.004979 *  9.75 $
        4,  7,  0.049930 *  9.75 $
        4,  8,  0.117956 *  9.75 $
        4,  9,  0.050425 *  9.75 $
        4, 10,  0.010519 *  9.75 $
        4, 12,  0.183472 *  9.75 $
```

Figure 5-7 RCCAL Sample Problem Input

```
 4,  13,   0.133690 *   9.75 $
 4,  14,   0.024245 *   9.75 $
 4,  15,   0.134895 *   9.75 $
 4,  16,   0.085010 *   9.75 $
 5,   6,   0.028614 *   9.10 $
 5,   7,   0.008978 *   9.10 $
 5,   8,   0.239230 *   9.10 $
 5,   9,   0.003409 *   9.10 $
 5,  10,   0.000480 *   9.10 $
 5,  11,   0.029433 *   9.10 $
 5,  14,   0.548228 *   9.10 $
 5,  15,   0.063204 *   9.10 $
 5,  16,   0.004413 *   9.10 $
 6,   8,   0.111438 *  71.785 $
 6,   9,   0.024525 *  71.785 $
 6,  10,   0.009124 *  71.785 $
 6,  11,   0.166134 *  71.785 $
 6,  12,   0.254095 *  71.785 $
 6,  13,   0.029230 *  71.785 $
 6,  14,   0.280099 *  71.785 $
 6,  15,   0.034727 *  71.785 $
 6,  16,   0.013945 *  71.785 $
 7,   8,   0.044039 *  72.065 $
 7,   9,   0.063644 *  72.065 $
 7,  10,   0.054888 *  72.065 $
 7,  11,   0.013817 *  72.065 $
 7,  12,   0.026117 *  72.065 $
 7,  13,   0.263905 *  72.065 $
 7,  14,   0.034592 *  72.065 $
 7,  15,   0.280429 *  72.065 $
 7,  16,   0.160389 *  72.065 $
 8,  11,   0.154383 *  69.470 $
 8,  12,   0.176175 *  69.470 $
 8,  13,   0.051308 *  69.470 $
 8,  14,   0.223080 *  69.470 $
 8,  15,   0.054380 *  69.470 $
 8,  16,   0.014259 *  69.470 $
 9,  11,   0.009565 *  31.500 $
 9,  12,   0.029146 *  31.500 $
 9,  13,   0.271163 *  31.500 $
 9,  14,   0.038298 *  31.500 $
 9,  15,   0.275746 *  31.500 $
 9,  16,   0.085091 *  31.500 $
10,  11,   0.006027 *  29.750 $
10,  12,   0.008478 *  29.750 $
10,  13,   0.247228 *  29.750 $
10,  14,   0.010768 *  29.750 $
10,  15,   0.251456 *  29.750 $
10,  16,   0.291733 *  29.750 $
11,  12,   0.220815 *  60.200 $
11,  13,   0.010974 *  60.200 $
11,  14,   0.257296 *  60.200 $
11,  15,   0.017966 *  60.200 $
11,  16,   0.028121 *  60.200 $
12,  14,   0.154333 *  79.093 $
12,  15,   0.044610 *  79.093 $
12,  16,   0.009757 *  79.093 $
13,  14,   0.052702 *  88.537 $
13,  15,   0.198592 *  88.537 $
13,  16,   0.158787 *  88.537 $
14,  16,   0.013765 *  88.193 $
15,  16,   0.170217 *  88.537 $
```

Figure 5-7 RCCAL Sample Problem Input (cont)

```
HEADER CORRESPONDENCE DATA
FIG    HAO

       2266        =1
       2270        =2,3,4,5
       2255        =6
       2254        =7
       2252        =8
       2243        =9
       2253        =10
       51          =11
       2264        =12
       2265        =13
       2259        =14
       2258        =15
       2240        =16
HEADER OPERATIONS DATA
BUILD  HAO, ALLBLK
       CALL FFDATA(0,0,0,0,0,0, PUN, NO)
L      FFCAL
       CALL GBDATA(2HIR,3HHAO,2HFF)
L      GBCAL
       CALL RCDATA(3HHAO,PUN,0,1000,0,999,0,1./144.,NO,0,0,
     1            0.9,91,IPRIME,ISECND)
L      RCCAL
END OF DATA
```

Figure 5-7 RCCAL Sample Problem Input (concl)

5.7 Direct Irradiation Segment

Calling sequence: L DICAL

This segment computes the thermal radiation directly incident on external spacecraft surfaces due to the presence of the sun or a nearby planet. Three components are computed: direct solar, reflected solar from the planetary surface (albedo) and infrared planetary emission. Shadowing effects due to internode blockage are accounted for. Normally, shadowing is computed analytically by determining if each vector from each node element to a heat source (sun or planet element) is blocked by an intervening surface. At the users option, shadowing may be computed by the use of shadow data on a restart (RSI) tape. This requires that the SFCAL segment be executed prior to any DICAL executions. If a restart tape is present, SFCAL will read the shadow data from it, make it available on a file used by DICAL, and set a DICAL flag that will bypass the analytical shadow calculations. If no RSI tape is present, SFCAL will compute the required shadow factor tables.

When using the externally - supplied shadow factor tables, DICAL will revert to an analytic calculation each time a shadow factor table must be interpolated over a shadow factor range greater than 0.5. For additional background and detail the reader is referred to Section 3.3.6: Shadow Factor Data.

Four program called subroutines are provided for user intervention through his subroutine data block. Subroutine DIPRES provides for special initialization prior to solar flux calculations. Similarly, DIPREP is called prior to planetary/albedo flux calculations. Subroutine DIENDS and DIENDP provide for user intervention subsequent to solar and planetary/albedo calculations, respectively. Figure 5-8 depicts the logic flow of segment DICAL.

DICAL output is placed in out-of-core storage for later use in absorbed flux calculations. In addition, direct irradiation data may be output to punched cards or the USER1 tape in HEADER FLUX DATA block format at the User's option. A printout of the direct irradiation data is provided also. An example of DICAL output obtained utilizing the ORBGEN option (Reference Section 3.3.9.2) can be found in Appendix H.

To compute fluxes, the Operations Data block must contain the orbit and orientation parameter definitions, if applicable, prior to any DI segment call.

If all of the flux values are already available on the RSI tape, the DIREAD subroutine (Reference Section 4.3.11: Subroutine DIREAD) may be used to load them in place of the DI segment call (UNIVAC only).

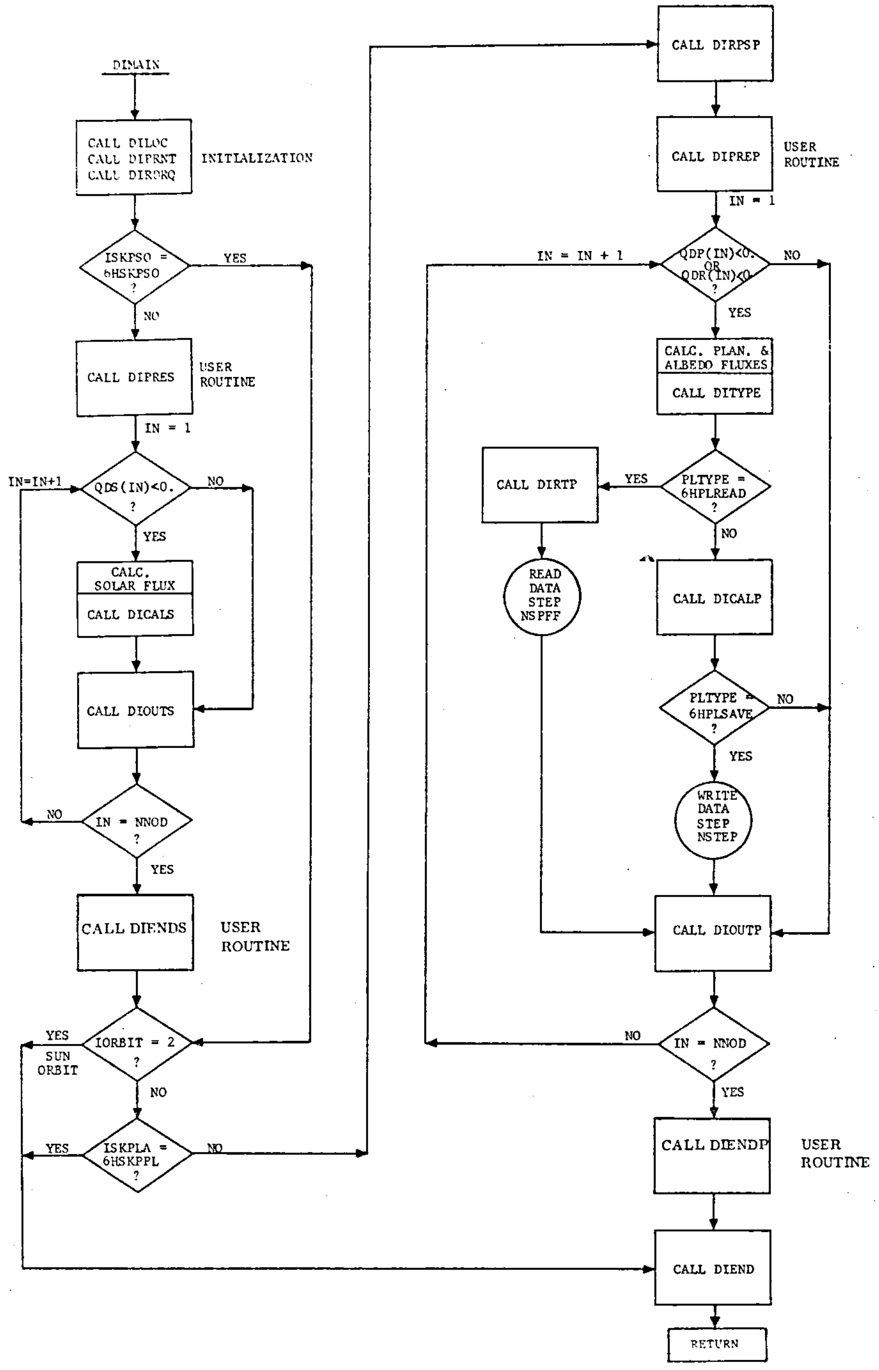


Figure 5-8 Segment DICAL Flow Diagram

5.8 Shadow Factor Segment

Calling sequence: L SFCAL

This segment computes shadow factor tables for each node of a spacecraft configuration to be used in direct irradiation calculations when it is desirable to save computation time at the expense of some accuracy.

By including a L SFCAL call prior to any DICAL call the program will compute shadow factor tables for the given configuration, and/or utilize precomputed shadow factor tables to obtain, by interpolation, the solar and planetary shadow factors for each node and at each orbit point. The entry points to the tables are the nodal clock and cone angles for the position vector to the sun or to a specific planetary element. Inaccuracies occur because the tables do not accurately reflect shadow entry and exit points for all nodes. The program minimizes this to a degree by computing a flux with shadowing entirely in the DICAL link whenever a shadow factor must be interpolated between points with data value differences greater than 0.5. Through utilization of the Shadow Factor Data Block (ref. Section 3.3.6) known shadow factors can be input and/or direction given for the computation or recomputation of selected portions of the shadow factor table.

Primary output for this segment is a file on the RSO tape containing shadow factor tables for each node. This file contains shadow data according to a format as presented in Appendix C. The shadow factor tables are also ouput in printed form as they are computed. Shadow factors for both the solar and infrared wavebands are computed, because semitransparent shadowing surfaces are allowed.

No user intervention is provided for this segment. Prior to an SFCAL call, the user may set a flag to obtain punched shadow factor data through the statement SFPNCH = 3HPUN. Punched output obtained will be in shadow factor data block format.

When shadow factors are read from an RSI tape the printout of the shadow factor tables are normally suppressed. They may be printed by including the statement SFPRNT = YES before the SFCAL call in the operations data. For additional background and detail, the reader is referred to Section 3.3.6: Shadow Factor Data.

5.9 Absorbed Heat Segment

Calling sequence: L AQCAL

This segment utilizes direct irradiation and radiant interchange data in out-of-core storage as input. From this, it computes absorbed heat values for each external spacecraft node. Internode reflections are accounted for in both the solar and infrared wavebands.

No user intervention through the subroutine data block is provided.

Segment AQCAL output is placed in out-of-core storage. Printed output of the Solar, Albedo and Planetary absorbed values with Correspondence Data applied is also provided as an option. An example of AQCAL output can be found in Appendix H.

5.10 Absorbed Heat Output Segment

Calling sequence: L QOCAL

This segment utilizes absorbed heat data in out-of-core storage to provide heat source tables in thermal analyzer format. At the user's option, heat versus time tables and/or orbital average heat data are provided for each external node.

Output is provided on punched cards or the BCDOU tape. Q versus time data is in thermal analyzer array data format, with a singlet time array and a corresponding singlet Q array for each node. Also punched are thermal analyzer interpolation subroutine cards for each node. Orbital average data is punched in source data block format.

Standard output is in SINDA thermal analyzer format. Subroutine DA11MDA is used for the interpolation subroutine. Output for other thermal analyzers may be obtained by altering the format statements in subroutine QOSAVE and entering the altered version in the subroutines data block. Card image printout of QOCAL output is provided. An example can be found in Appendix H.

Segment QOCAL logic flow is shown in Figure 5-9.

5.11 Form Factor Combining Segment

Calling sequence: L CMCAL

Segment CMCAL is used to apply correspondence data to a form factor or image factor matrix that was previously computed and placed in data storage. After applying the correspondence data, the resulting matrix of combined form factors is written to data storage for later use by the GBCAL program segment. CMCAL will apply the auto-correspondence data (generated by polygon input) and/or users-supplied correspondence data at the option of the user. User control is applied through subroutine CMDATA. CMCAL can be used to combine form factor matrices, as computed and/or stored by the FFCAL segment, or it can combine image factor matrices, as computed and/or stored by the RBCAL segment. Please note, however, that it is <u>not</u> possible to combine FFCAL output, and then attempt to compute combined image factors using RBCAL. The execution sequence must be FFCAL, RBCAL and CMCAL, in that order. The CMCAL flow diagram is shown in Figure 5-10. The reader should refer to Section 3.3.8: Correspondence Data for additional background and detail.

5.12 Image Factor Segment

Calling sequence: L RBCAL

Segment RBCAL is used to compute one aspect of the effects of specular-diffuse surfaces on radiant interchange factors. When specular-diffuse surfaces are present, the form factors may be considered to have two components: the direct or geometric form factor from node I to node J, plus

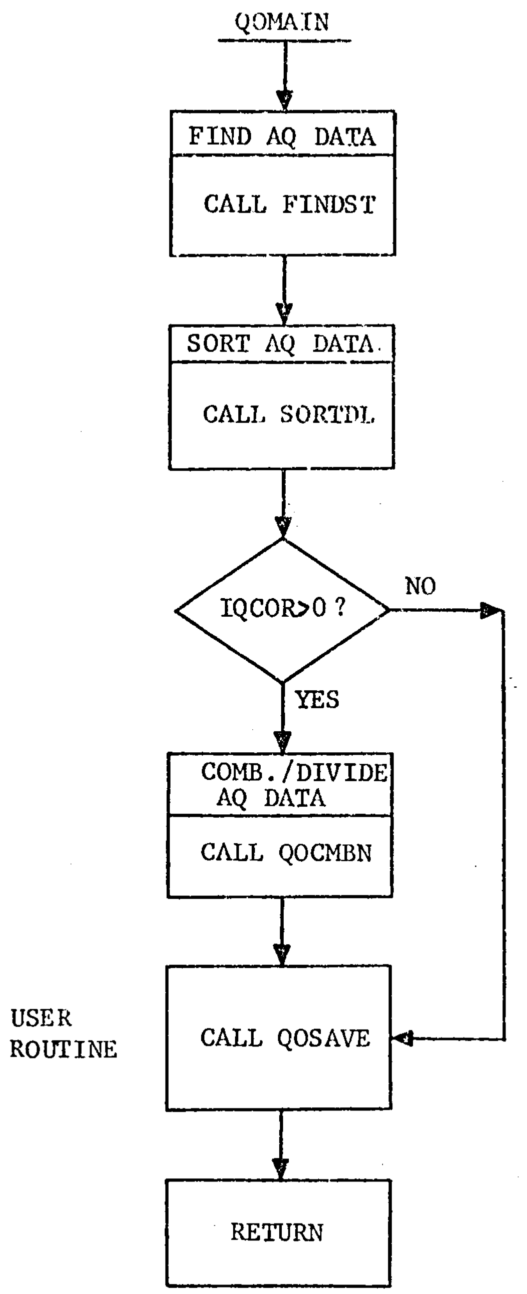


Figure 5-9 Segment QOCAL Flow Diagram

the "image" components resulting from the images of node J as seen by node I in all visible specular surfaces. In TRASYS, the direct form factors are computed by FFCAL or NFFCAL . RBCAL is then used to generate the nodal images in each specular surface, compute the various form factors to the images by the double summation method and add them to the direct form factors. The resulting modified form factors, dubbed image factors, are written to data storage. Segment GBCAL can then be used to generate radiant interchange factors from the image factor matrix and nodal surface properties.

Segment RBCAL considers first order specular bounces only, that is, images of images are not generated and considered. Appendix I presents the theory used in RBCAL. The segment RBCAL flow diagram is shown in Figure 5-11.

5.13 Direct Irradiation via Specular Surfaces - Segment DRCAL

Calling sequence: L DRCAL

The total direct irradiation that reaches a nodal surface consists of that reaching it directly from the sun or planet element plus that reaching it from images of the sun or planetary element as seen in specular surfaces. Segment DRCAL computes the irradiation resulting from the images in the same manner that segment DRCAL computes the specular-bounce components of the image factors.

In the present version of TRASYS, DRCAL does not compute the specular component of planetary irradiation because it is not felt that the compute time is justified. The flow diagram for segment DRCAL is shown in Figure 5-14.

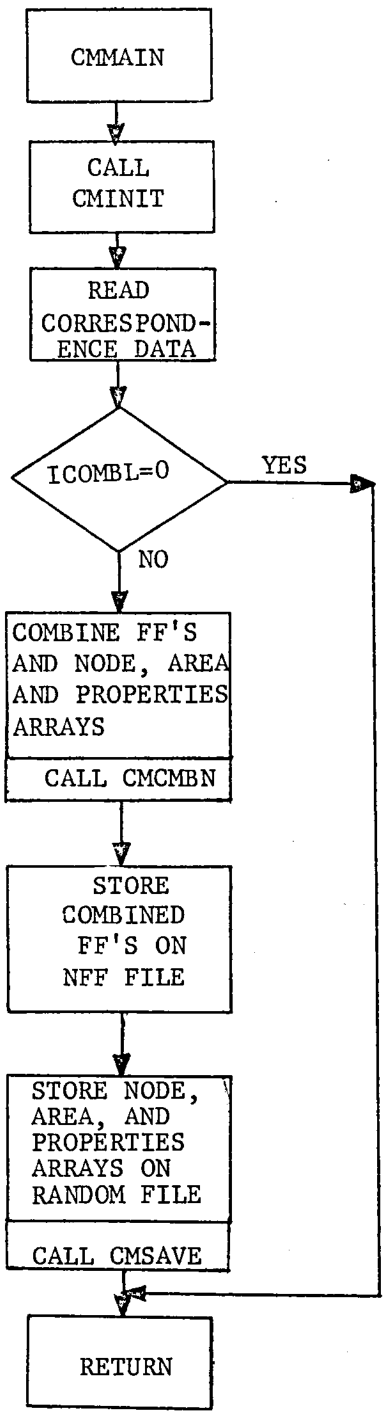

CMMAIN
CALL CMINIT
READ CORRESPOND-ENCE DATA
ICOMBL=0
YES
NO
COMBINE FF'S AND NODE, AREA AND PROPERTIES ARRAYS
CALL CMCMBN
STORE COMBINED FF'S ON NFF FILE
STORE NODE, AREA, AND PROPERTIES ARRAYS ON RANDOM FILE
CALL CMSAVE
RETURN

Figure 5-10 Segment CMCAL Flow Diagram

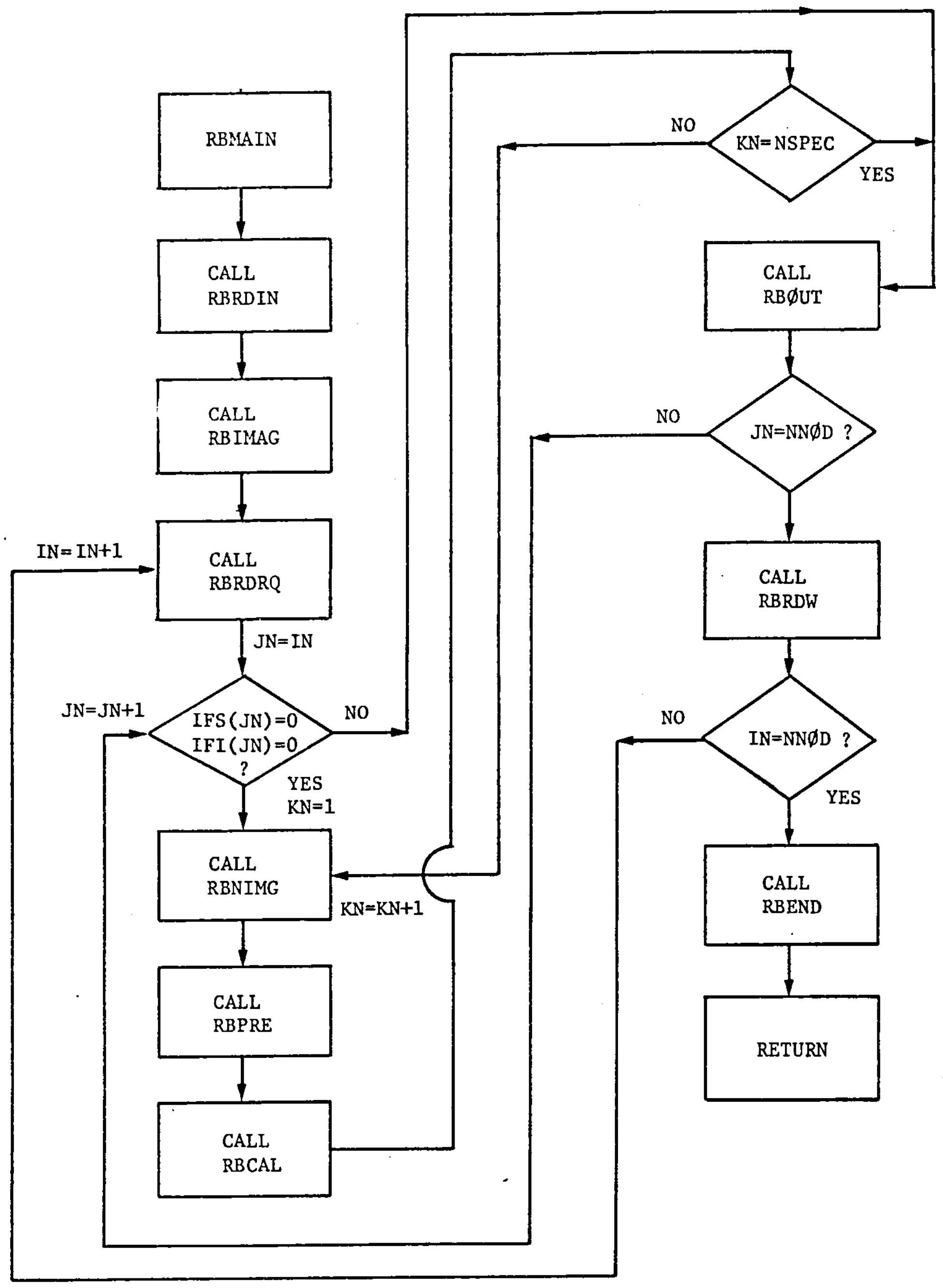


Figure 5-11 Segment RBCAL Flow Diagram

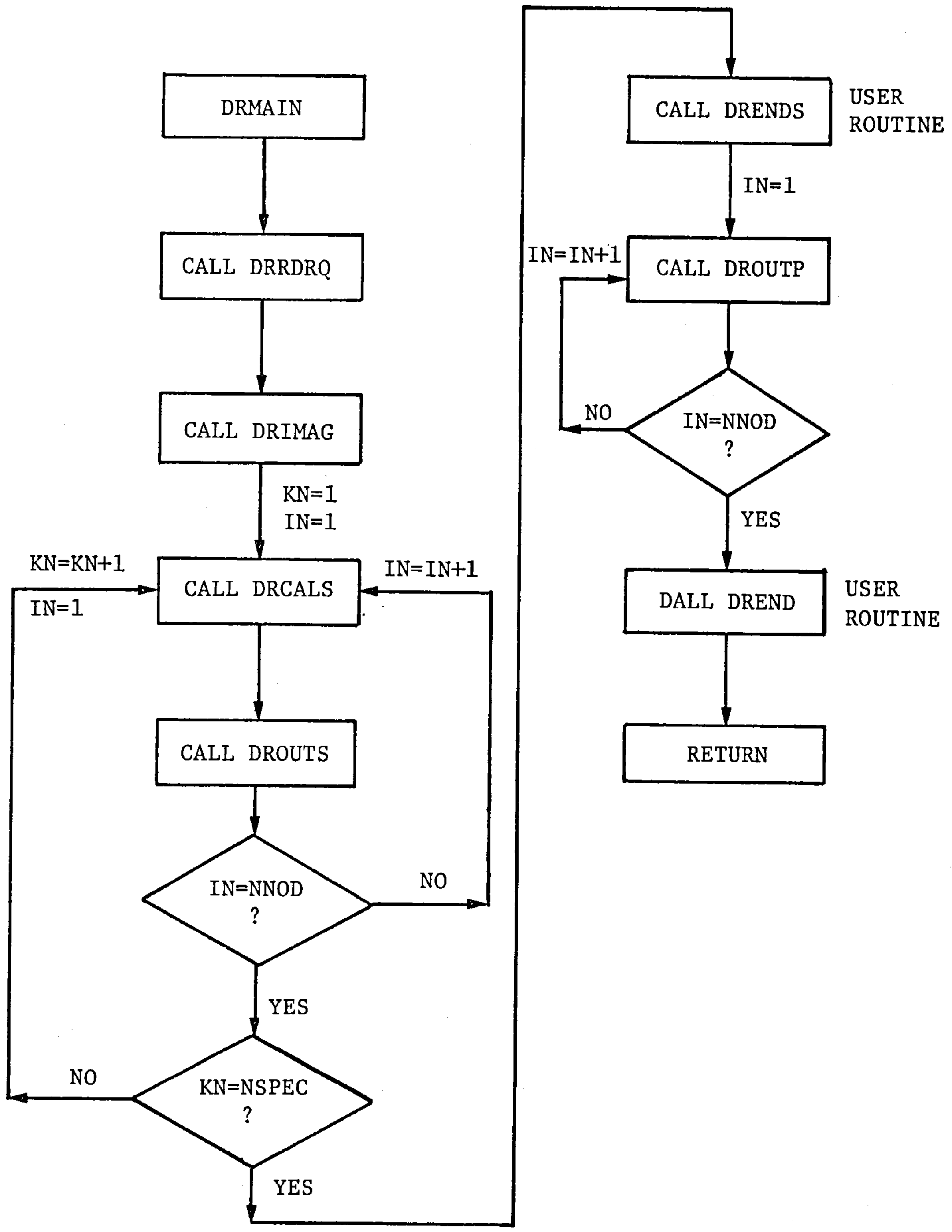


Figure 5-12 Segment DRCAL Flow Diagram

APPENDIX A

RESERVED WORD LIST - ALL SEGMENTS

Reserved Word List - All Segments

ALAN	ALPH			
APER	AQPRNT	ARAD	AREA	ASUN
ATMT	BETA	BETAS	BOXINL	BOXINR
BOXOUT	CIGMA	CIGMAS	CLOCK	CONE
DAWN	DELCT	DIACC	DIACCS	DIMS
DINOSH	DIPNCH	DLTLNE	DOY	DSTR
DTE	DTR	DUSK	DWP	ECC
ELPBEA	EMISS	FFACC	FFACCS	FFCMB
FFDISF	FFMIN	FFNAC	FFNOSH	FFPNCH
FFPRNT	FFRATL	FFZERO	FOG	GAUSS
GBWBND	GRAV	HA	HP	IAI
IALBFL	IAQGBI	IAQGBS	IAQSDA	IAQSDP
IAQSDS	IAS	IAUTOC	ICALFL	ICMBL
IDSTR	IEQFF	IFFSHO	IFS	IHSTEP
IKS	ILLUMN	IMESS	INCORE	INDXN
INDXS	INSHAD	INTMF	IOPNNP	IOPNV
IOPNVU	IOPTIT	IORBIT	IORNT	IOVL
IPAGE	IPLAFL	IPLNA	IPLSN	IPLUNT
IPRDMP	IQOARY	IQOCOR	IQOTAB	IQOTME
IRKCN	IRKNSP	IROTX	IROTY	IROTZ
IRSI	IRTI	ISFAC	ISFT	ISHO
ISKIP	ISKPSO	ISOLFL	ISPEC	ISPND
ISTPDR	ISTRT	ITRALL	ITRCAO	ITRCBO
ITRCCO	ITRCDO	ITRC10	ITRC20	ITRC30
ITRC40	ITRC50	ITRC60	ITRC70	ITRC80
ITRC90	IIPLOT	KBCDOU	KRSI	KRSO
KRTI	KRTO	KTRAJ	LAQSEG	LCMCOM
LCMSEG	LDISEG	LDRSEG	LFFSEG	LGBCOM
LGBSEG	LIDINR	LIDOUT	LIDSP	LINE
LMFSEG	LNPSEG	LODSEG	LOPSEG	LPLCOM
LPLSEG	LQOCOM	LQOSEG	LRBSEG	LRCCOM
LRCSEG	LRDSEG	LSFSEG	MAXBC	MAXFL
MB	MESSL	MESSR	MFCO	MFLUK
MG	MITSIN	MLINE	MNND	MODELN
MRSRC	MSRF	NACT	NBCDOU	NBCDSK
NBLKDR	NBLKLN	NCONT	NCURFL	NDI
NDIR	NELCT	NELN	NERN	NFF
NFFR	NFFTYP	NFIGCO	NFIGFF	NFIGGB
NFRMC	NGBIR	NGBIRR	NGBSO	NIBBLE
NJOB	NLRIO	NMESS	NMIR	NMODEL
NMODIR	NMODLS	NN	NNOD	NNODC
NNODU	NODE	NOUT	NPLS	NPLSR
NPNNP	NPTIT	NPUN	NPVU	NRAN
NRARR	NRMOD	NRSI	NRSO	NRSP
NRSRCB	NRSRCE	NRSRCI	NRSRCO	NRSRCT
NRTI	NRTO	NS	NSCRR	NSCR1
NSCR2	NSCR3	NSPEC	NSPFF	NSPND
NSQNTL	NSSTEP	NSTEP	NSTPDI	NSTPL
NSTSOL	NSURF	NTITLE	NTQ	NTQR
NTRAJ	NUSER1	NUSER2	ODTEMP	OINC

Reserved Word List - Continued

OPROT	OPRPLN	OPSCL	OPSCLR	OPTIMP
OPTIMS	OPTRUE	ORNT	PALB	PERIOD
PI	PLCL	PLCMB	PLCO	PLCRVF
PLLABX	PLLABY	PLTIT1	PLTIT2	PLTYPE
PLXMPF	PLYMPF	PNAME	PR	PRAD
PSD	PSH	QOAMPF	QOFMPF	QOPNCH
QORMPF	QOTAPE	QOTMPF	QOTYPE	RALB
RATE	RFRAC	RKAMPF	RKMIN	RKPNCH
RKSP	RKTAPE	ROTX	ROTY	ROTZ
RPLAN	RSOLAR	RSUN	RTD	RTHET
RTOL	SAOS	SFPRNT	SHADIN	SHAOUT
SIGMA	SOL	SOLO	SPINT	SREFLI
SREFLS	SRIR	SRSO	STRACK	STRDEC
STRRA	SUNCL	SUNCO	SUNDEC	SUNPVO
SUNRA	TDIAM	THGHT	TIMEPR	TIMEST
TIMSP	TITLE	TME	TRIR	TRSO
TRUANF	TRUANI	TRUEAN	TSTR	WDS
WSS	WSUN	ZNPROT	ZNPSCL	

In addition to the reserved word list above, the following reserved words must be preserved when working in the particular segments noted.

Segment AQCAL

QDS, QDR, QDP, QAS, QAR, QAP, GBSO, GBIR, AQTEMP, ICATEG, ICOMB

Segment CMCAL

DATA, FFVALI, FFVALS, ICOMB, ICATEG, IX, SCRIR, SCRSO, SUM

Segments DICAL, DRCAL

QDS, QDR, QDP, ISHAD, DATA

Segments FFCAL, NFFCAL

FFVALI, FFVALS, BFE, BFA, ISHAD, SUM, INDXF, DATA

Segment GBCAL

FA, SPACE, XSPACE, IX

Segment NPLOT

MNP

Segment OPLOT

MSP, JSURF

Segment PLOT

IX

Segment QOCAL

NODET, QAVERG, ICOMB, IFIRST, AREAT, IX

Segment RBCAL

DATA, FFVALI, FFVALS, ISHAD, NRMSS1, NRMSS2, RBVALI, RBVALS

Segment RCCAL or RKCAL

ISPN, MSND, NDS, SF, SPACNO, EMIT, AREAT, IX

Segment SFCAL

ISHAD, QDP, QDR, QDS

APPENDIX B

FORM FACTOR CALCULATION ACCURACY - DOUBLE SUMMATION METHOD

A. ELEMENTAL GRID VARIATIONS

The form factor for two finite areas, A_I and A_J (Fig. B-1), is defined as

$$F_{IJ} = \frac{1}{A_I} \int_{A_I} \int_{A_J} \frac{\cos\theta_i \cos\theta_j}{\pi r_{ij}^2} dA_J \, dA_I. \qquad [B-1]$$

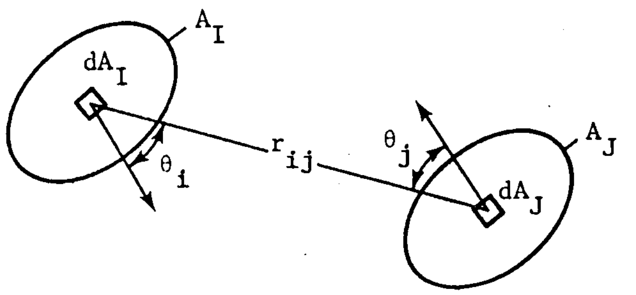


Figure B-1 Determination of Form Factors

A finite-difference approximation of Equation [B-1] is

$$F_{IJ} = \frac{1}{A_I} \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{\cos\theta_i \cos\theta_j}{\pi r_{ij}^2} A_j \, A_i. \qquad [B-2]$$

Equation [B-2] approaches an exact representation of Equation [B-1] as the size of the elemental areas, A_i and A_j, approach zero. For identical, parallel, directly opposed rectangles, the empirical relationship of elemental area size-to-separation distance versus form factor error shown in Figure B-2 is obtained.

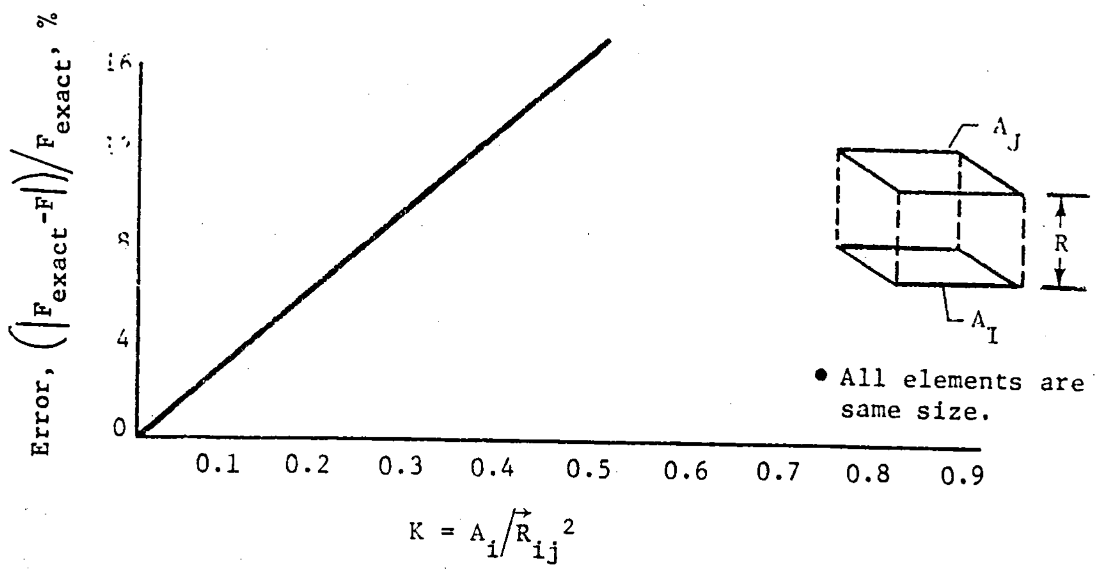


Figure B-2 Error Characteristics for Identical, Parallel, Directly Opposed Rectangles

For this case,

$$A_i = K\, r_{ij}2 \qquad \text{(B-3)}$$

where K is a proportionality constant.

A more general form of Equation (B-2), considering that

$$dF_{ij} = \frac{\cos\theta_i \ \cos\theta_j}{r_{ij}^{\,2}}\, dA_j \qquad \text{(B-4)}$$

or

$$\frac{dA_j}{dF_{ij}} = \frac{\pi r_{ij}2}{\cos\theta_i \ \cos\theta_j}, \qquad \text{(B-5)}$$

$$A_i \approx \frac{\text{FFACC}\ r_{ij}^{\,2}}{\cos\theta_i \ \cos\theta_j}. \qquad \text{(B-6)}$$

MTRAP version 1.0, LOHARP, and TRASYS/FFCAL use Equation B-2, modified with a shadowing constant, to compute form factors. The total number (i.e., size) and distribution of elemental areas is left to the user to define in MTRAP version 1.0 and LOHARP. The number of elements to be selected is defined by the closest node a given node "sees." The selection of elements, however, usually ends up being somewhat arbitrary or simply a matter of economics; i.e., the finer the grid, the more machine time required. In reality, the selection of elemental areas is an independent problem for each form factor.

The basic assumption for reasonably accurate form factors is, from Equation B-6, that the elemental area size is small compared to the separation distance between two elements.

TRASYS/FFCAL uses a technique using Equation B-6 to automatically select the element grid sizes of each node pair consistent with a user-defined accuracy parameter, FFACC. If all elemental areas on each of two nodes were the same size and had the same separation distance, r_{ij}, the apparent number of elements on a node to satisfy the accuracy value FFACC would be (from Equation B-6)

$$N_I = \frac{A_I}{A_i} = \frac{A_I}{(FFACC)} \frac{\cos\theta_i \cos\theta_j}{r_{ij}^2} . \qquad \text{B-7}$$

Since each element pair on the two nodes may have a different separation distance, a different apparent number of equal-sized elements will be required.

The approach used in TRASYS/FFCAL is a simple arithmetic average of element contributions, i.e.,

$$N_{opt_I} = \frac{A_I}{(FFACC) m_i\, m_j} \sum_{i=1}^{m_i} \sum_{j=1}^{m_j} \frac{\cos\theta_i \cos\theta_j}{r_{ij}^2} , \qquad \text{B-8}$$

where m_i and m_j are the initial number of elements arbitrarily chosen for nodes I and J.

The initial number of elements is chosen just large enough for a representative sample. A similar optimum number of elements for node J can be defined.

The total number of elements defined by Equation (B-8) is distributed uniformly over the node using a criterion that attempts to make the elements square. The arithmetic average technique assumes that the mean separation distance between nodes is large compared with the variation of separation distance over the two nodes. A check is made to see if this assumption is violated. The maximum number of elements defined by any element pair on the two nodes (Equation (B-7)) is compared with the arithmetic average value (Equation (B-8)). If the ratio of N_{max}/N_{opt} is greater than FFRATL, the two nodes are temporarily subdivided into subnodes. FFRATL is an input value defined by the user. The numbers of subnodes used are proportional to $\left(N_{max}/N_{opt}\right)_I$ and $\left(N_{max}/N_{opt}\right)_J$. The optimum grid elements are computed independently for each subnode using a separation-distance, weighted-average criterion, rather than an arithmetic one. The form factors resulting from the subnode pairs are then combined using form-factor algebra. Thus, elemental grids vary for each form factor and may be nonuniform over a node as required to satisfy input accuracy requirements.

B. NODAL PRELIMINARY SHADOWING CHECKS

Shadowing checks between elemental areas account for considerable machine time. Machine time could be saved if unnecessary checks were eliminated. The usual procedure in MTRAP version 1.0 is to process all the surfaces until either the form factor contribution is reduced to zero by shadowing surfaces, or all surfaces identified as shadowers have been investigated. The function of the nodal shadowing checks is to eliminate from the element-to-element shadowing checks all surfaces that cannot cause shadowing on any portion of the two nodes under consideration. The technique used in TRASYS/FFCAL is a significant modification of the technique used in LOHARP.

The nodal shadowing checks consist of constructing a sphere around each node for which form factors are being evaluated and for each shadowing surface. The radii of the spheres are such that the node or surface is completely enclosed. A test cylinder or cone frustum, depending on the relative sizes of the two spheres in question, is constructed as shown in Figure B-3. For the cylinder, the radius is equal to the larger of the two spheres. The cylinder or cone frustum's axial coordinate is a vector between the centers of the two spheres plus the sum of the two sphere radii. Next, the shadowing surfaces' enclosing spheres are checked to determine whether they intersect the test cylinder or cone frustum. Only surfaces whose sphere intersects the test cylinder or cone frustum will be considered in the actual element-to-element shadowing checks for these two nodes.

This technique of preliminary shadowing checks allows identification of any surface that shades the nodes in question. However, other marginal ones will also be identified.

In the detailed element-to-element shadowing checks, an element pair is either completely shadowed or not at all. The accuracy, then, of representing the shadow is proportional to the total number of elements on both nodes. The number of elements on the shadowing surface(s) is of no consideration. In the TRASYS program it is assumed that accurate shadowing is required only for large-magnitude form factors. If the preliminary shadowing checks identify shadowing surfaces for the form factor in question, the number of elements defined to represent the shadow is

$$\begin{aligned} N_{s_I} &= F_{IJ}\, B/(\mathrm{FFACCS}) \\ N_{s_J} &= F_{JI}\, B/(\mathrm{FFACCS}), \end{aligned} \qquad \text{(B-9)}$$

where
FFACCS is an input shadowing accuracy factor, and
B is a proportionality constant determined by trial and error.

The number of elements used for any given node for form factors is taken as the maximum of that defined in Equation (B-8) or Equation (B-9).

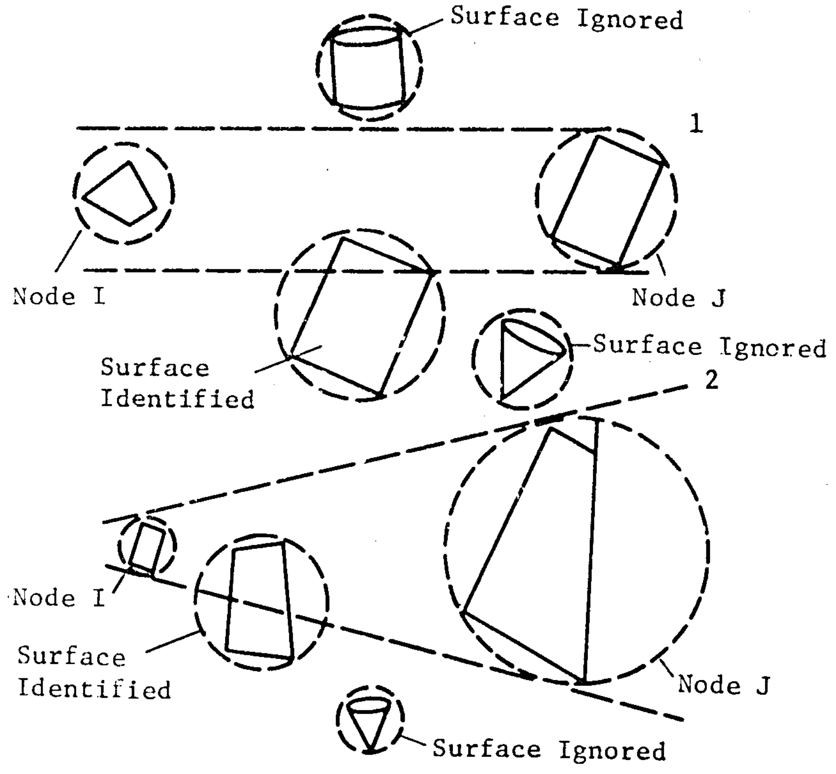


Figure B-3 Nodal Preliminary Shadowing Techniques

APPENDIX C

RESTART TAPE FORMAT

The master restart tape consists of two or nine data files, (see Section 3.3.9.8.1: Permanent Restart Output Tape - RSO). For Appendix C, two files will be assumed. The first file is written during preprocessor execution and the second during processor execution. The first file consists of edit history data and images of all active and inactive input data cards.

The second file consists of one or more "pseudo-files" begun with a standard header record and ending with a standard trailer record. Each restartable processor segment (FFCAL, GBCAL, SFCAL, DICAL, RBCAL, CMCAL and DRCAL) writes a pseudo-file containing the data necessary to restart an interrupted job with minimum repeated calculations. Also, there is a pseudo-file for each type of correspondence data, automatic, form-factor and GB. In addition, pseudo-files containing nodal property data are produced by CMCAL, from any BUILDC/ADD series, or from any series of calls to data modification routines (Ref. 4.3.8: Data Modification Routines). The correspondence data and property data pseudo-files are provided only for use by special TRASYS/thermal analyzer interface programs, and are not used in restart operations. Also, there may be a pseudo-file of direct-incident shadow factors for printout in restarted runs.

HEADER RECORD FORMAT

Word	
1	Record number (consecutive from beginning of file)
2	DATE
3	TIME
4	6HHEADER
5	CONFIGURATION NAME (6 characters, max.)
6	RESTART POINT (restartable), ACCESS NO. (node, area, property arrays) or STEP NO. (absorbed fluxes and incident fluxes)
7	One of the following LABEL words: FFCAL, GBIR, GBSO, SFCAL, CMCAL, RBCAL (with restart point); or PROPBD (from BUILDC, with access no.); or PROPCG (from mod routines with access no.); or PROPCM (from CMCAL, with access no.); or AQCAL, DICAL, DRCAL (with step no.); or CORRES, CORRFF, SHFAC.
8	TYPE OF COMBINING (CM,AU,AUCM) For CMCAL, GBIR, GBSO, AQCAL files; job number for other files.
9	NO. OF NODES for CMCAL, GBIR, GBSO, AQCAL files; date for other files.
10	thru 12 - SAME AS 1 THRU 3

TRAILER RECORD FORMAT

Words 1 through 3 contain record number, date, time

Words 4 through 9 contain 3HEND

Words 10 through 12 contain record number, date, time

CORRESPONDENCE DATA PSEUDO-FILE FORMAT

Record

1 HEADER (contains CORRES or CORRFF as label)

2 THRU N - Correspondence data records. Each record is 100 words or number of nodes words long, whichever is least.

N + 1 Trailer Record

PROPERTY ARRAY PSEUDO-FILE FORMAT

Record

1 HEADER (contains PROPBD or PROPCG as label).

2 4HNODE, NNOD, (NODE(I), I = 1,NNOD)*

3 4HAREA, NNOD, (AREA(I), I = 1,NNOD)*

4 5HEMISS, NNOD, (EMISS(I), I = 1,NNOD)*

5 5HALPHA, NNOD, (ALPH(I), I = 1,NNOD)*

6 4HTRIR, NNOD, (TRIR(I), I = 1,NNOD)*

7 4HTRSO, NNOD, (TRSO(I), I = 1,NNOD)*

8 4HSRIR, NNOD, (SRIR(I), I = 1,NNOD)*

9 4HSRSO, NNOD, (SRSO(I), I= 1,NNOD)*

10 Trailer record

If label is PROPCM, records 10 and 11 contain:

10 5HICOMB, ICMBL, (ICOMB(I), I = 1,ICMBL)*

11 Trailer record

where:

NODE = Node identification numbers

NNOD = Number of nodes in problem

*Note: All data records begin and end with record number, date and time

AREA	
EMISS	
ALPHA	
TRIR	Node surface properties (Ref. Table III-3)
TRSO	
SRIR	
SRSO	

ICMBL = Length of ICOMB array

ICOMB = Array of combined node identification numbers

DICAL PSEUDO-FILE FORMAT

Record	
1	Header (label = 5HDICAL)
2	NNOD, TIMEPR, TRUEAN, (NODE(I), I = 1,NNOD)*
3[1]	NEPT, SHADR, SHADP, ((PLAVT(I,J), I = 1,NEPT), J = 1,3)*
4[1]	(SUMR(I), SUMP(I), I = 1,NEPT)*
2*NNOD+2	NNOD, (QDS(I), I = 1,NNOD)*
2*NNOD+3	NNOD, (QDR(I), I = 1,NNOD)*
2*NNOD+4	NNOD, (QDP(I), I = 1,NNOD)*
2*NNOD+5	(SUNPV(I), I = 1,3), ((PLDC(I,J), J = 1,3), I = 1,3)
2*NNOD+6	NSURF, (IFS(K), IKS(K), (PR(I,K), I = 1,2), (DSTR(I,K), I = 1,5), (DIMS(I,K), I = 1,3), (PSH(I,K), I = 1,4), (((TSTR(I,J,K), J = 1,3), I = 1,3) K = 1,NSURF), NSPEC, (ISPEC(I), SREFLI(I), SREFLS(I), I = 1,NSPEC)
2*NNOD+7	Trailer record

where:

NNOD = Number of nodes

TIMEPR = Orbit time

TRUEAN = True anomaly

NODE = Node identification numbers

NEPT = Number of elements on planet

SHADR = Node - planet shadow factor (solar waveband)
SHADP = Node - planet shadow factor (IR waveband)

NOTE: These are redundant with SHFAC pseudo file.

PLAVT = Planet element area vectors (3 components)
SUMR = Form factors from node to planet elements (solar waveband)
SUMP = Form factors from node to planet elements (IR waveband)
QDS, QDR, QDP = Incident solar, albedo and planetary flux values

SUNPV = Sun position vector
PLDC = Matrix to transform vectors in the CCS to vectors in the planet-oriented VCS.
NSURF = Number of surfaces
IFS = Array of shadower suface input sequence numbers
IKS = Array of shadower surface type flags
PR = Array of shadower surface transmissivities in the infrared and solar wavebands
DSTR = Array of shadower surface dimensions
DIMS = Array of shadower surface SCS origin position vectors in CCS
PSH = Array of radii of shadower surface encompassing spheres and CCS position vectors to the centers of those spheres
TSTR = Array of matrices to transform vectors in the CCS to vectors in the shadower SCS's.
ISPEC = Array of input sequence numbers of specular surfaces
SREFLI = Array of surface specular reflectivity in infrared waveband
SREFLS = Array of surface specular reflectivity in solar waveband

NOTES: [1]Records 3 through 2*NNOD+1 exist only for circular planet oriented orbits. Otherwise this pseudo-file contains 8 records only.

*All data records begin and end with record number, date and time.

DRCAL PSEUDO-FILE FORMAT

Record	
1	Header (label = 5HDRCAL)
2	NNOD, TIMEPR, TRUEAN, (QDS(I), QDR(I), QDP(I), I = 1,NNOD)*
3	Trailer record

Reference DICAL format for variable definitions.

FFCAL or NFFCAL PSEUDO -FILE FORMAT

Record	
1	Header (label = 5HFFCAL)
2	LBNODA, NNOD, (NODE(I), I = 1,NNOD)
3	IROW, J, K, FFSHOI, FFSHOJ, (FFVALI(I), I = J,K), (FFVALS(I), I = J,K), BFE (I), I = J,K), (BFA(I), I = J,K)*
.	
.	
NNOD+2	Trailer record

where:

IROW = "Emitter" node number

J = Integer location (in node array) of emitter node (I.GE.J.LE.NNOD)

NNOD = Number of nodes

NODE = Node Numbers

FFVALI = Infrared form factors from emitter node to receiver node

FFVALS = Solar form factors from emitter node to receiver node

BFE = Blockage factors corresponding to each FFVALI

BFA = Blockage factors corresponding to each FFVALS

LBNODA = Node Array label (6HNODARY)

**NOTE: All data records begin and end with record number, date and time.

AQCAL PSEUDO-FILE FORMAT

Record	
1	Header (label = 5HAQCAL)
2	NNOD, TIMEPR, TRUEAN, (NODE(I), I =1,NNOD)
3	NNOD, (QAS(I), I = 1,NNOD)
4	NNOD, (QAR(I), I = 1,NNOD)
5	NNOD, (QAP(I), I = 1,NNOD)
6	Trailer record

where:

NNOD = Number of nodes

TIMEPR = Orbit time

TRUEAN = True Anomaly

NODE = Node identification numbers

QAS, QAR, QAP = Absorbed solar, albedo and planetary heat values

SHFAC PSEUDO-FILE FORMAT

This pseudo file follows each DICAL pseudo-file when ISFAC = YES and is used to printout the DI shadow factors for restart runs.

Record	
1	Header Record (label = 5HSHFAC)
2	(SHADS(I), I = 1,NNOD)*
3	(SHADR(I), I = 1,NNOD)*
4	(SHADP(I), I = 1,NNOD)*

where:

SHADS	=	Solar shadow factor
SHADR	=	Albedo shadow factor
SHADP	=	Planetary shadow factor

*All data records begin and end with record number, date and time.

CMCAL PSEUDO-FILE FORMAT

Record	
1	Header record (label = 5HCMCAL)
2	NNODC, ICMBL, (ICATEG(I), I = 1,NNOD), (ICOMB(I), I = 1,ICMBL), (NODEC(I), AREAC(I), EMISSC(I), ALPHC(I), TRIRC(I), TRSOC(I), SRIRC(I), SRSOC(I), I = 1,NNODC)*
.	
.	
.	
NNODC+2	Trailer record

where:

NNODC = Number of nodes after combining

ICMBL = Length of ICOMB array

ICATEG = Combining Category array

ICOMB = Correspondence data array

NODEC = Node no. array after combining

AREAC
EMISSC
ALPHC
TRIRC — Node property arrays after combining
TRSOC
SRIRC
SRSOC

GBCAL PSEUDO-FILE FORMAT

Record	
1	Header record (label = 4HGBIR or 4HGBSO)

*NOTE: All data records begin and end with record number, date and time.

2 IROW*, SPACE, (FA(I), I = J,NNOD*)

•

•

•

NNOD+2 Trailer record

where:

IROW = Number of emitter nodes

SPACE = Radiation interchange factor to space

FA = Array of radiant interchange factors from emitter node to receiver nodes

NNOD = Number of nodes

J = Integer locator number of node IROW

*NOTE: If CMCAL has been executed, NNOD = NNODC and IROW = NODEC(J)

RBCAL PSEUDO-FILE FORMAT

Record

1 Header Record (label = 5HRBCAL)

2 IROW, (RBVALI(I), I = J,NNOD), (RBVALS(I), I = J,NNOD)*

•

•

•

NNOD+2 Trailer record

where:

IROW = Node number of emitter node

RBVALI = Array of "total" form factors from emitter to all receivers (infrared waveband) "total" form factor includes direct form factor to receiver, plus form factors to all images of receiver as seen in specular surfaces.

RBVALS = Same, for solar waveband

*NOTE: All data records begin and end with record number, date and time.

J = Integer location for node IROW in NODE array

NNOD = Number of nodes in problem

SFCAL PSEUDO-FILE FORMAT

Record	
1	Header record (label = 5HSFCAL)
2	NNOD, (NODE(I), I = 1,NNOD)*
3	(TABSHA (ICO,ICL), ICL = 1,19), ICO = 1,9
4	(TABSHE (ICO,ICL), ICL = 1,19), ICO = 1,9
.	
.	
.	
2*NNOD+1	(TABSHA (ICO,ICL), ICL = 1,19), ICO = 1,9
2*NNOD+2	(TABSHE (ICO,ICL), ICL = 1,19), ICO = 1,9
2*NNOD+3	Trailer record

where:

NNOD = No. of nodes

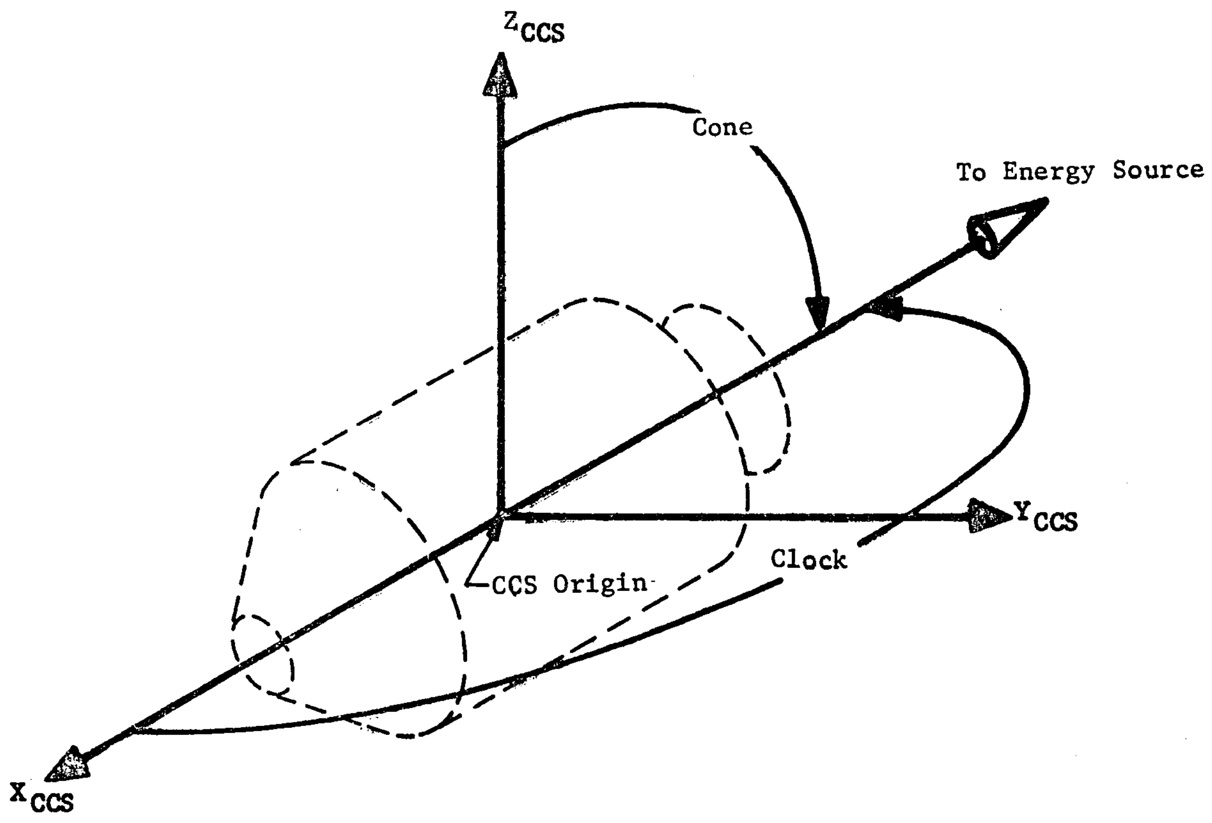
NODE = Node no. array

TABSHA = Shadow factor array - solar

TABSHE = Shadow factor array - infrared

*NOTE: All data records begin and end with record number, date and time.

Clock angles 1 through 19 range from 0° to 360° in 20°increments about the central coordinate system z-axis, (See Figure C-1). The 0° and 360° points are repeated to avoid wrap-around interpolation. Cone angles 1 through 9 are 180., 157.5, 135., 112.5 90., 67.5, 45, 22.5, and 0. degrees respectively.



Shadow table point illustrated for
Cone 5 (90^o), Clock 10 (180^o).

Figure C-1 Energy Source Direction for Shadow Factor Tables

APPENDIX D

SUBROUTINE DESCRIPTIONS

SUBROUTINE NAME: ADD

PURPOSE:

This subroutine adds to the problem geometry all nodes/surfaces contained in BCSNAM.

VARIABLE NAME:

BCSNAM is a Block Coordinate System name of up to 6 characters.

RESTRICTIONS:

Call valid only after previous calls to BUILDC or ADD within current step. BCSNAM must be a block coordinate system name as defined in surface data.

CALLING SEQUENCE:

```
CALL ADD (BCSNAM)
```

EXAMPLE:

```
CALL ADD (EXTANK)
```

RELATED INFORMATION:

See BUILD CARD information - Page D-8

SUBROUTINE NAME: ADSURF

PURPOSE:

This subroutine functions to add an adiabatic "closure" surface to the problem geometry and adds the pertinent form factors to the form factor matrix.

Variable Name:	Description	Default Values
BCSN	Name of a block coordinate system containing the "closure" surface	None
NFIGFF	Configuration name under which modified form factor matrix is to be stored	Current configuration name
AREA	Area of adiabatic "closure" surface	Computed Area based on data in Surface Data Block

RESTRICTIONS:

Block coordinate system BCSN must appear in the surface data block with one and only one node (and, therefore, one surface). This surface must be completely defined in the Surface Data Block, including the surface properties desired for the "closure" surface. The concept to simulate an adiabatic condition would require a very small IR emissivity.

In addition it requires for the third argument an area equivalent to the smallest possible area required to close out the configuration as a complete enclosure. Physically this area may consist of one or more parts, for example, the ends of a long cylinder. Only one surface is required since only the true enclosure area is what needs to be preserved. If the last argument is zero, the program will use the area computed in the Surface Data Block. The dimension of this surface regardless of surface type must be such that the computed area will be equivalent to the smallest possible area required to close out the configuration as a complete enclosure with one or more areas.

1. The call to GBDATA subsequent to an ADSURF call must have GBWBND = 2HIR
2. ACTIVE = Both is not allowed for the adiabatic "closure" surface.

<u>CALLING SEQUENCE:</u>

```
CALL ADSURF (BCSN, NFIGFF, AREA)
```

<u>RELATED INFORMATION</u>

See Section 4.3.3.4: Adiabatic "Closure" Surface and Appendix J. Use of Adiabatic Closure Surfaces.

SUBROUTINE NAME: AQDATA

PURPOSE:

This subroutine defines parameters used in AQCAL for calculation of absorbed heats. Direct fluxes for AQ calculations are obtained from current step data storage.

Variable Names	Default Names
IAQGBS - configuration name for solar grey-body matrix*	Current Config. Name
IAQGBI - configuration name for IR grey-body matrix*	Current Config. Name
RSOLAR - multiplying factor for solar absorbed heat	1.0
RALB - multiplying factor for albedo absorbed heat	1.0
RPLAN - multiplying factor for planetary absorbed heat	1.0

RESTRICTIONS:

Must be called subsequent to a DICAL execution within same step.

NOTES: If not called prior to an AQCAL execution (within same step), default values assumed. Individual default values obtained by passing zero arguments.

CALLING SEQUENCE:

```
CALL AQDATA (IAQGBI, IAQGBS, RSOLAR, RALB, RPLAN)
```

*NOTE: If IAQGBS or IAQGBI is input as 4HZERO, the absorbed solar or the absorbed infrared fluxes, respectively, will be set to 0.0.

The reading of solar or infrared gray body factors will also be bypassed so that unused gray body factors need not be calculated.

RELATED INFORMATION:

If a comprehensive printout of absorbed fluxes is desired, the FORTRAN statement AQPRNT = YES should appear prior to any L AQCAL card or any ORBGEN card (reference Section 4.3.7.1: Subroutine AQDATA).

SUBROUTINE NAME: BUILDC

PURPOSE:

This subroutine is used to define as problem geometry all nodes and surfaces identified with a Block Coordinate System. BUILDC is the first call to define a new configuration.

VARIABLE NAME:

BCSNAM is a block coordinate system name consisting of up to 6 characters. (alphanumeric, beginning with an alphabetic character)

CONFIG is a Hollerith name identifying the current active configuration.

RESTRICTIONS:

Must be called prior to any Subroutine ADD calls within a step. BCSNAM must be ALLBLK, or a block coordinate systems name as defined in surface data. CONFIG must be input as a Hollerith string of up to six characters. For example, 5HSHUTL.

NOTE: If BCSNAM = ALLBLK, all surfaces in the surface data block become problem geometry. If CONFIG = 0 in the first call to BUILDC, the configuration name defaults to the run model name input in the options data block. CONFIG must be input in subsequent calls to BUILDC.

CALLING SEQUENCE:

```
CALL BUILDC (BCSNAM, CONFIG)
```

RELATED INFORMATION:

Sequences of calls to BUILDC and ADD may be accomplished with one card (reference Section 3.3.9.4: Build Option), formatted as follows:

```
CC1           CC7
   BUILD        FIG,BLK1,BLK2,BLK3
```

This is equivalent to the sequence:

```
              CC7
                CALL BUILDC(BLK1,3HFIG)
                CALL ADD(BLK2)
                CALL ADD(BLK3)
```

Note that the configuration name, FIG, must begin to the right of card column 6. If a continuation card is required to list all the BCS names involved, some character is required in CC6 of each continuation card. A BCS name may not be split between cards.

SUBROUTINE NAME: CHGBLK

PURPOSE:

This subroutine allows the user to change block coordinate system parameters where:

BCSNAM	- block coordinate system to be changed
TX	- translation along CCS X-axis
TY	- translation along CCS Y-axis
TZ	- translation along CCS Z-axis
IROTX	- order X rotation is to be performed (1,2,3)
IROTY	- order Y rotation is to be performed (1,2,3)
IROTZ	- order Z rotation is to be performed (1,2,3)
ROTX	- angle of rotation about CCS X-axis
ROTY	- angle of rotation about CCS Y-axis
ROTZ	- angle of rotation about CCS Z-axis

RESTRICTIONS:

1. TX, TY, TZ, ROTX, ROTY, ROTZ must be floating-point numbers.
   IROTX, IROTY, IROTZ must be integers, 1, 2, or 3
2. Must be called prior to the applicable BUILDC/ADD sequence.

CALLING SEQUENCE:

```
CALL CHGBLK (BCSNAM, TX, TY, TZ, IROTX, IROTY, IROTZ, ROTX, ROTY, ROTZ)
```

SUBROUTINE NAME: CMDATA

PURPOSE:

This subroutine is used to define parameters used by CMCAL in combining form factors according to correspondence data.

Variable Names		Options	Default Value
NFIGFF	- Configuration name for uncombined form factor access	N/A	Current Config. Name
NFIGCO	- Configuration name for form factor correspondence data access	N/A	Current Config. Name
NFFTYP	- Form factor type flag	2HFF, 2HRB	2HFF
IAUTOC	- Flag to apply auto combining data	3HYES, 2HNO	3HYES
FFPRNT	- Print flag for combined form factors	3HYES,2HNO	3HYES

NOTES

If not called prior to a CMCAL execution, default values are used.

CALLING SEQUENCE:

```
CALL CMDATA (NFIGFF,NFIGCO,NFFTYP,IAUTOC,FFPRNT)
```

RELATED INFORMATION:

See Section 3.3.8: Correspondence Data

SUBROUTINE NAME: DICOMP

PURPOSE:

This subroutine is used to define logic used in subsequent DICAL execution.

DEFINITIONS:

Variable Names	Options	Default
ISOLFL - solar flux	a. 4HZERO - zeros out solar flux for all nodes b. 0 (integer) - results in computation of solar fluxes c. STEPN (integer step number) - stuffs solar fluxes from STEPN into current step storage	0 (compute)
IALBFL - albedo flux compute/stuff flag	Same as for ISOLFL	0
IPLAFL - planetary flux compute/stuff flag	Same as for ISOLFL	0

RESTRICTIONS:

Cannot zero albedo flux if planetary is calculated (and vice versa).

NOTES:

1. Compute/stuff flags are overridden by the planet shadow. Nonzero solar or albedo fluxes will never be stuffed into storage for a point within the planet shadow.

2. Failure to call DICOMP prior to a DICAL execution results in default values for all three flags.

CALLING SEQUENCE:

```
CALL DICOMP (ISOLFL, IALBFL, IPLAFL)
```

SUBROUTINE NAMES: DIDT1, DIDT1S

PURPOSE:

Calls to define direct irradiation shadowing and accuracy parameters and to compute heat source position vectors from true anomaly or time.

DEFINITIONS:

Variable Names	Default Values
DINOSH - shadow/no shadow flag (Options: 4HNOSH, 4HSHAD)	4HSHAD (shadow calculations not bypassed)
DIACC - element selection accuracy factor for node/planet form factors	0.25
DIACCS - element selection accuracy factor for shadowing 40 calculations	0.10
TRUEAN - true anomaly. If TIMEPR is entered TRUEAN will be computed	None
NSPFF - step number reference to obtain node-planet form factors previously saved in reference step number	0 (new form factors computed)

Variable Names	Default Values
TIMEPR - time	None
DIPNCH - flux punch flag (Options: 3HYES, 2HNO, 4HTAPE*)	2HNO
ISFAC - flag to write shadow factor on RSO for printout on subsequent runs (Options: 3HYES, 2HNO)	2HNO(CDC) 3HYES(UNIVAC)

RESTRICTIONS:

Either TRUEAN or TIMEPR must be defined in call.

CALLING SEQUENCE:

```
CALL DIDT1 (DINOSH, DIACC, DIACCS, TRUEAN, NSPFF, TIMEPR, DIPNCH, ISFAC)

CALL DIDT1S (TRUEAN, NSPFF, TIMEPR, DIPNCH, ISFAC)
```

RELATED INFORMATION:

The statement ITRC70 = 2HON prior to a call to the DICAL segment will print for each node the surface numbers considered as possible shadowers.

NOTE:

*Writes BCD output to USER1 file in Flux Data Block input Format.

SUBROUTINE NAMES: DIDT2, DIDT2S

PURPOSE:

Calls to define direct irradiation shadowing and accuracy parameters and to compute heat source position vectors from look angles.

DEFINITIONS:

$$\left.\begin{array}{l}\text{DINOSH}\\ \text{DIACC}\\ \text{DIACCS}\\ \text{NSPFF}\\ \text{DIPNCH}\\ \text{ISFAC}\end{array}\right\} \text{Reference DIDT1}$$

SUNCL, SUNCO - look angles to sun (clock, cone) in the VCS.*

PLCL, PLCO - look angles to planet (clock, cone) in the VCS.**

TIMEPR - present time

ALT - spacecraft altitude

NOTE:

*Allowable ranges of SUNCL, PLCL are 0 to 360 degrees.

**Allowable ranges of SUNCO, PLCO are 0 to 180 degrees.

RESTRICTIONS:

These calls must be preceded by a call to ORBIT1 or ORBIT2. The purpose is to define the orbit-centered body and set the variables PRAD, SOL, PALB, WDS and WSS. A call to ORIENT is required if the CCS and the VCS are not coincident.

Subroutine SPIN should not be used with DIDT2 or DIDT2S.

CALLING SEQUENCE:

```
CALL DIDT2 (DINOSH, DIACC, DIACCS, NSPFF, SUNCL, SUNCO,
PLCL, PLCO, TIMEPR, ALT, DIPNCH, ISFAC)

CALL DIDT2S (NSPFF, SUNCL, SUNCO, PLCL, PLCO, TIMEPR, ALT, DIPNCH,
ISFAC)
```

SUBROUTINE NAME: DIDT3, DIDT3S

PURPOSE:

These subroutines are used to define and update direct irradiation parameters when using the planet surface option.

DEFINITIONS:

Variable Names	Default Value
DINOSH - shadow/no shadow flag (Ref. DIDT1)	4HSHAD
DIACCS - element selection accuracy factor for shadowing	0.1
ITOD - time of day (military time), integer (e.g., 1435)	None
DIPNCH - flux punch/no punch flag (Ref. DIDT1)	2HNO
ISFAC - flag to write shadow factors on RSO (Ref. DIDT1)	2HNO(CDC) 3HYES(UNIVAC)

RESTRICTIONS:

Must be preceded by a call to subroutine SURFP.

CALLING SEQUENCE:

```
CALL DIDT3(DINOSH,DIACCS,ITOD,DIPNCH,ISFAC)
CALL DIDT3S(ITOD, ISFAC)
```

NOTES:

1. Use the statements TIMEPR = DAWN and TIMEPR = DUSK in the operation data to set current time to sunrise time and sunset time, respectively.

DIDT3 and DIDT3S CANNOT be used with DAWN or DUSK as the first argument. The sequence:

```
DIDT3 (4HNOSH, .05,0,0,0)
TIMEPR = DAWN
```

is required to set DINOSH, DIACCS and TIMEPR is the time desired is DAWN.

SUBROUTINE NAME: DIREAD

PURPOSE:

This subroutine is used to expedite read-in of the DI information on RSI tapes.

RESTRICTIONS:

May be called in lieu of L DICAL on restart runs only when the DI pseudo-file is known to be complete and if no overrides from the flux data block are desired.

CALLING SEQUENCE:

```
CALL DIREAD
```

RELATED INFORMATION:

When IFO (last argument) is set to DIR on the ORBGEN card, all L DICAL cards normally generated by the ORBGEN call will be replaced by calls to DIREAD. (Ref. Sec. 3.3.9.2: ORBGEN option)

SUBROUTINE NAMES: DITTP, DITTPS

PURPOSE:

These subroutines read data from a trajectory tape and define spacecraft/heat source parameters through subroutine DIDT2. DITTP is called initially in order to define planetary parameters and position the tape for subsequent time points. DITTPS is used to update time and attitude/position data.

DEFINITIONS:

Variable Names		Options
TIME	- mission time	Real no.
ITYPE	- identifier for special event record	Integer
PLANAM	- name of orbit-centered planet (if applicable) (ref ORBIT1)	Hollerith
IDWDN	- number of word FIDEN in identification record	Integer
FIDEN	- file identification word Hollerith	
NTIM	- number of time word in information record	Integer
NTYPE	- number of word ITYPE in information record	Integer
NCLPL	- number of word containing clock angle-to-planet vector	Integer
NCOPL	- number of word containing cone angle-to-planet vector	Integer
NCLS	- number of word containing clock angle-to-sun vector	Integer
NCOS	- number of word containing cone angle-to-sun vector	Integer
NRAD	- number of word containing planet center-to-spacecraft distance	Integer
NWOR	- number of words in tape record Integer	
ALTMF	- multiplying factor to convert units of NRAD word to feet	Real no.

Variable Names		Options
IBOD -	one-body/two-body flag 0 - One-body tape 1 - Two-body tape, use body 1 2 - Two-body tape, use body 2	integer
DIPNCH -	Punch/no punch flag for orbital flux output	Hollerith

RESTRICTIONS:

a. Calls to DITTPS to update time and type can be made only after a call to DITTP is in effect.

b. The TIME argument in DITTPS calls must be greater than any previously defined TIME argument until the tape is repositioned through a call to DITTP.

RELATED INFORMATION:

See Section 4.3.5.8: Subroutine DITTP and DITTPS

CALLING SEQUENCES:

```
CALL DITTP (TIME, ITYPE, PLANAM, IDWDN, FIDEN, NTIM, NTYPE, NCLPL,
NCOPL, NCLS, NCOS, NRAD, NWOR, ALTMF, IBOD, DIPNCH)

CALL DITTPS (TIME, ITYPE)
```

SUBROUTINE NAME: DRDATA

PURPOSE:

This subroutine is used to define parameters used by DRCAL in computing direct irradiation with real body effects.

Variable Names		Options	Default Values
NSTPDI	- step number for flux data access	Integer	Current step no.
DIACCS	- accuracy parameter for flux shadowing	Reference DIDT1	0.1

NOTES:

If not called prior to DRCAL execution, default values are used.

Calling Sequence:

```
CALL DRDATA (NSTPDI, DIACCS)
```

SUBROUTINE NAME: FFDATA

PURPOSE:

This subroutine will define parameters used in FFCAL if other than default values are used.

DEFINITIONS:

Variable Names			Default Values
FFACC	-	orientation accuracy factor	0.05
FFACCS	-	shadowing accuracy factor	0.1
FFNOSH	-	shadowing override flag (4HNOSH, 4HSHAD)	4HSHAD
FFRATL	-	distance/area ratio factor	15.0
FFMIN	-	eliminate small form factors	1.E-6
FFPRNT	-	flag to print form factors (3HYES,2HNO)	3HYES
FFPNCH	-	flag to punch form factors (3HYES,2HNO,4HPALL*, 4HTAPE**)	2HNO
FFNAC	-	node array check flag (3HYES,2HNO)	3HYES

* 4HPALL will punch all form factors (UNIVAC version)

** Writes form factor output to the USER1 file in form factor data block format.

RESTRICTIONS:

None

NOTES:

Example: CALL FFDATA (0., 0., 4HNOSH, 0, 1.E-3, 0, 3HYES, 3HYES) Results in no shadowing computations, form factors below 0.001 ignored, form factors printed and default values used elsewhere. If value passed is zero, default value assumed.

RELATED INFORMATION:

1. The statement IFFSHO = 2HNO prior to a call to the FFCAL link will bypass Form Factor computations to shadower only nodes. IFFSHO defaults to 3HYES.

2. FFPNCH defaults to punch calculated form factors if RSO tape is not specified.

3. The statement FFZERO = DV, prior to a call to the FFCAL or NFFCAL links will set the entire matrix to the Specified Data Valve (DV) and over-ride the form factor * Area matrix on the RSI file. It will not, however, override the Form Factor Data block.

CALLING SEQUENCE:

```
CALL FFDATA (FFACC, FFACCS, FFNOSH, FFRATL, FFMIN, FFPRNT, FFPNCH,
FFNAC)
```

SUBROUTINE NAME: FFNDP

PURPOSE:

This subroutine is used to obtain a node number array, punched on cards in format used in form factor, flux data, and Shadow Factor data blocks.

RESTRICTIONS:

None

CALLING SEQUENCE:

```
CALL FFNDP
```

SUBROUTINE NAME: FFREAD

PURPOSE

This subroutine is used to expedite read-in of form factor information on restart tapes.

RESTRICTIONS:

May be called in lieu of L FFCAL or L NFFCAL on restart runs only when the form-factor pseudo-file is known to be complete and if no over-ride/recomp information is desired from the form factor data block.

CALLING SEQUENCE:

```
CALL FFREAD
```

SUBROUTINE NAME: GBAPRX

PURPOSE:

This subroutine calculates gray-body radiant interchange factors using an approximate relationship and stores the results in data storage. Uses form factors and optical properties stored under current configuration name.

Argument Name	Description	Options	Default
GBWBND	Waveband definition name	2HIR, 3HSOL 4HBOTH	4HBOTH
NFIGFF	Configuration name for form factor access		Current Config. Name
NFFTYP	Form factor type to be used in GB calculations	2HFF, 2HCM	Last type calculated under CFIGFF

NOTE:

1. Input zero for default action.
2. Gray body factor computed according to:

$$\mathcal{F}_{ij} = F_{ij}\epsilon_i\epsilon_j$$

where: F_{ij} = form factor from i to j

ϵ_i = infrared or solar absorptivity, i.

ϵ_j = infrared or solar absorptivity, j

RELATED INFORMATION:

See Section 4.3.6.2: Approximate Radiant Interchange Factors - Subroutine GBAPRX

CALLING SEQUENCE:

```
CALL GBAPRX (GBWBND, 6HNFIGFF, NFFTYP)
```

SUBROUTINE NAME: GBDATA

PURPOSE

Defines parameters used by segment GBCAL in computing a grey-body factor matrix. Uses form factor * area matrix stored under configuration name, and the specified optical properties.

Variable Names	Default Value
GBWBND - Waveband definition name (2HIR, 3HSOL, 4HBOTH)	4HBOTH
NFIGFF - Configuration name for form factor access	Current Config. name
NFFTYP - Form factor type to be used in GB calculations Options: 2HFF, 2HRB, 2HCM	Last type calculated under NFIGFF

CALLING SEQUENCE:

```
CALL GBDATA (GBWBND, 6HNFIGFF, NFFTYP)
```

RESTRICTIONS:

1. GBDATA must be called prior to calculating gray body factors.

2. Do not use a GBWBND of 3HSOL or 4HBOTH when used in conjunction with a call to ADSURF.

3. Cannot use SOL in place of 3HSOL for Variable GBWBND. SOLAR is permissable.

SUBROUTINE NAME: LIST

PURPOSE:

This subroutine is used to obtain a printed listing of data on BCDOU and/or USER1 tapes.

Variable Names	Options	Default
NAMEF - Name of tape to be listed	BOTH, USER1, BCDOU	BOTH
N - Number of files to be listed	Integer, 3HALL	None

NOTE:

1. Call valid after writing data to BCDOU from RCCAL, RKCAL and/or QOCAL, or after writing data to USER1 from FFCAL and/or DICAL.

CALLING SEQUENCE:

```
CALL LIST (NAMEF, N)
```

RESTRICTIONS:

Do not call Subroutine LIST until all writing to NAMEF Tape/file has been completed.

SUBROUTINE NAME: MODAR

PURPOSE:

This subroutine changes the area of a designated node, or changes the area of all currently active nodes by use of a multiplier.

Argument Name	Description	Options	Default
ND	Node Number designator	a. Any active node number (integer) b. 3HALL	None
AR	Desired value for area	a. Floating-point data value b. Area multiplier[1] (3HALL option only)	None

NOTE:

1. When ND = 3HALL, all active node areas are modified according to: AREA = AREA*AR.

RESTRICTIONS:

1. Call not valid prior to geometry definition through calls to BUILDC and ADD.

2. MODAR calls are cancelled by BUILD card or a subsequent BUILDC/ADD Sequence. Areas revert to those surface data.

RELATED INFORMATION:

See Appendix D - Subroutine NODDAT. Also see Section 4.3.8.1: Subroutine MODAR.

CALLING SEQUENCE:

```
CALL MODAR (ND, AR)
```

SUBROUTINE NAME: MODPR

PURPOSE:

This subroutine modifies the diffuse infrared emmissivity and/or the diffuse solar absorptivity of a designated node.

Argument Name	Description	Options	Default
ND	Node number designator	Any active node number	None
ALPHA	Diffuse solar absorptivity	a. $0. \leq DV \leq 1.$	None
		b. $DV < 0.$	
EMISS	Diffuse IR emissivity	a. $0. \leq DV \leq 1.$	None
		b. $DV < 0.$	

NOTE:

1. If ALPHA   0. or EMISS   0., current values are not changed.

RESTRICTIONS:

1. Call not valid prior to geometry definition through calls to BUILDC and ADD.

2. MODPR calls are cancelled by BUILD card, or a subsequent BUILDC/ADD sequence. Properties revert to those in surface data.

__CALLING SEQUENCE:__

```
CALL MODPR (ND, ALPHA, EMISS)
```

__RELATED INFORMATION:__

See Subroutine NODDAT - Page D - 42

SUBROUTINE NAME: MODPRS

PURPOSE:

This subroutine modifies the solar and/or infrared specular reflectivity of a designated node.

Argument Name	Description	Options	Default
ND	Node number designator	Any active node number	None
SPRS	Specular reflectivity, solar	a. $0. \leq DV \leq 1.$ b. $DV < 0.$[1]	None
SPRI	Specular reflectivity, infrared	a. $0. \leq DV \leq 1.$ b. $DV < 0.$[1]	None

NOTES:

1. If SPRI < 0. or SPRS < 0., current values are not changed.

RESTRICTIONS:

1. This call applicable only to nodes defined as specular reflectors in the surface data block.

2. Call not valid prior to geometry definition through calls to BUILDC and ADD.

3. MODPRS calls are cancelled by a subsequent BUILD card or BUILDC/ADD sequence. Properties revert to those in surface data.

CALLING SEQUENCE:

```
CALL MODPRS (ND, SPRS, SPRI)
```

RELATED INFORMATION:

See subroutine NODDAT - Page D-42

SUBROUTINE NAME: MODSHD

PURPOSE:

This subroutine modifies the SHADE/BSHADE flags for a designated surface.

Argument Name	Description	Options	Default
ISR	Surface number designator	Any active surface number	None
SHADE	"Can shade" flag	FF, DI, BOTH, NO, 0[1]	None
BSHADE	"Can be shaded" flag	FF, DI, BOTH, NO, 0[1]	None

NOTES:

1. If SHADE or BSHADE data values are zero, their values are not changed.

2. Shade flag changes affect entire surface.

3. MODSHD calls are cancelled by a subsequent BUILD card or BUILDC/ADD sequence.

RESTRICTIONS:

Call not valid prior to geometry definition through calls to BUILDC and ADD.

Call not applicable to shadower-only surfaces.

Calls to MODSHD are cancelled by a subsequent BUILD card or BUILDC/ADD sequence. Properties revert to those in surface data.

CALLING SEQUENCE:

```
CALL MODSHD (ISR, SHADE, BSHADE)
```

SUBROUTINE NAME: MODTR

PURPOSE:

This subroutine modifies the solar and/or infrared transmissivity of a designated surface.

Argument Name	Description	Options	Default
ISR	Surface number designator	Any active surface number	None
TRANS	Solar transmissivity	a. $0. \leq DV \leq 1.$ b. $DV < 0.$[1]	None
TRANI	IR transmissivity	a. $0. \leq DV \leq 1.$ b. $DV < 0.$[1]	None

NOTES:

1. A negative TRANS or TRANI should be used if two TRASYS surfaces are input for one semitransparent body. This avoids having the shadow factors multiplied by the SQUARE of the Transmissivity. The program will use the SQUARE ROOT of the Transmissivity if the user enters the transmissivities with a negative sign (Ref. Section 4 3.3.3.8: Properties Data).

2. Transmissivity changes affect entire surface.

RESTRICTIONS:

Call not valid prior to geometry definition through calls to BUILDC and ADD.

Calls to MODTR are cancelled by a subsequent BUILD Card or BUILDC/ADD sequence. Properties revert to those in surface data.

CALLING SEQUENCE:

```
CALL MODTR (ISR, TRANS, TRANI)
```

RELATED INFORMATION:

See Appendix D: Subroutine NODDAT

SUBROUTINE NAMES: NDATA, NDATAS

PURPOSE:

These subroutines may be called prior to a call to the node plotter segment to define optional views and miscellaneous parameters where:

Parameter	Description	Options*	Default
NV	View number	1-6	1
VU	View	3HALL, 3H3-D 1HX, 1HY, 1HZ, 3HGEN	3HALL
SCL	Scale	Floating-point no.	Automatic scale
NACT	Flag for plotting active side of arrows	2HNO, 3HYES	2HNO
ISHO	Flag to plot shadower-only surfaces	2HNO, 3HYES	2HNO
SELN	Name of array contain-identification numbers of nodes to be selectively plotted	Array name	Plot all nodes
TIT	Array name of plot title	Array name (array length 66 characters max.)	Uses job title
IROTX,IROTY IROTZ	Order of rotations (for VU = 3HGEN)	1,2,3 (any order)	1,2,3
ROTX, ROTY ROTZ	View rotations (for VU = 3HGEN)	Real no.	0.0,0.0, 0.0

*Input zero for default action

NOTE:

The NV parameter allows the user to define up to 6 plot operations that will be executed with one NPLOT segment call. Later in execution, (after a geometry change, for instance), he can execute the same 6 operations or change one or more by reference to the appropriate NV before his NPLOT call.

RESTRICTIONS:

None

CALLING SEQUENCE:

```
CALL NDATA (NV, VU, SCL, NACT, ISHO, SELN, TIT, IROTX, IROTY, IROTZ,
ROTX, ROTY, ROTZ)

CALL NDATAS (NV, VU, SCL, NACT, ISHO)
```

SUBROUTINE NAME: NFDATA

PURPOSE:

This subroutine defines the parameters for the NFFCAL segment if other than default values are desired.

DEFINITIONS:

Variable Names		Description	Default
NELCT	-	total element count on a node (16 through 200 allowed)	50
FFNOSH	-	Reference subroutine FFDATA	4HSHAD
FFMIN	-	Reference subroutine FFDATA	1.E-6
FFPRNT	-	Reference subroutine FFDATA	3HYES
FFPNCH	-	Reference subroutine FFDATA	2HNO
FFNAC	-	Reference subroutine FFDATA	3HYES

CALLING SEQUENCE:

```
CALL NFDATA (NELCT, FFNOSH, FFMIN, FFPRNT, FFPNCH, FFNAC)
```

RELATED INFORMATION:

See Appendix D: subroutine FFDATA

SUBROUTINE NAME: NODDAT

PURPOSE:

This routine may be called at any time from the operations data block to print nodal areas and surface optical properties. This routine is particularly useful following calls to the "MOD" routines.

RESTRICTIONS:

May be called at any time after a model has been defined by calls to "BUILDC" and "ADD", or using the "BUILD" card.

CALLING SEQUENCE:

```
CALL NODDAT
```

SUBROUTINE NAMES: ODATA, ODATAS

PURPOSE:

These subroutines may be called prior to a call to the orbit plotter to define optional views and miscellaneous parameters where:

Parameter	Description	Options*	Default
NV	View number	1-6	1
VU	View	3HALL, 3H3-D, 4HBETA, 5HCIGMA, 3HSUN, 3HGEN	3HALL
SCL	Spacecraft size measured from CCS origin in plot frame dimensions	Real no.	Computed automatically
SCLR	Orbit radius in plot frame dimensions	Real no.	Computed automatically
RPLN	Planet radius in plot frame dimension	Real no.	1.4 inches
TRUEAN	True anomaly	Real no.	None
TIMEST	Time of periapsis passage	Real no.	None
TIME	Present time	Real no.	
SELN	Name of array containing surface numbers to be selectively plotted	Array name (array length 66 characters, max.)	Plots all surfaces defined as shadowers

Parameter	Description	Options*	Default
TIT	Array name of plot title	Array name	Uses job title
IROTX, IROTY, IROTZ	Order of rotations (for view = 3HGEN)	1,2,3 (any order)	1,2,3
ROTX, ROTY, ROTZ	View rotations to rotate plotter reference coordinate system (see Figure 4-2 on P. 4-16) into user's desired view.	Real no.	0.0, 0.0, 0.0

*Input zero for default action

The NV parameter allows the user to define up to 6 plot operations that will be executed with one OPLOT call. Later in execution (after a geometry change, for instance), he can execute the same 6 operations or change one or more by reference to the appropriate NV before this OPLOT call.

RESTRICTIONS:

Calls valid only after orbit has been defined.

CALLING SEQUENCE:

```
CALL ODATA (NV, VU, SCL, SCLR, RPLN, TRUEAN, TIMEST, TIME, SELN, TIT,
IROTX, IROTY, IROTZ, ROTX, ROTY, ROTZ)

CALL ODATAS (NV, VU, SCL, SCLR, RPLN, TRUEAN, TIMEST, TIME)
```

SUBROUTINE NAME: ORBIT1

PURPOSE:

This subroutine defines spacecraft orbits in terms of classic orbital mechanics parameters and a celestial coordinate system.

DEFINITIONS:

Variable Names			Default Values
PNAME	-	name of orbit-centered body	None
ALAN	-	longitude of ascending node	None
APER	-	argument of perifocus	None
OINC	-	orbit inclination	None
TIMEST	-	time of periapsis passage, hours	0.0
HP	-	altitude at periapsis	None
HA	-	altitude at apoapsis	None
ECC	-	orbit eccentricity	None
SUNRA	-	right ascension of sun	None
SUNDEC	-	declination of sun	None
STRRA	-	right ascension of star	None
STRDEC	-	declination of star	None

RESTRICTIONS:

None

CALLING SEQUENCE:

```
CALL ORBIT1 (PNAME, ALAN, APER, OINC, TIMEST, HP, HA SUNRA, SUNDEC,
STRRA, STRDEC)

CALL ORBIT1 (PNAME, ALAN, APER, OINC, TIMEST, HP, ECC, SUNRA, SUNDEC,
STRRA, STRDEC)
```

NOTES:

PNAME options are as follows: 3HMER, 3HVEN, 3HEAR, 3HMOO, 3HMAR, 3HJUP, 3HSAT, 3HNEP, 3HURA, and 3HSUN. These names are used to key the following program variables:

WDS	-	darkside infrared emissive power at planet surface
PALB	-	planet albedo value (solar reflectance)
PRAD	-	planet radius
WSS	-	infrared emissive power at subsolar point
SOL	-	solar "constant" at planet-sun distance
GRAV	-	planet gravitational constant at surface

The Sixth argument is tested for magnitude. If it is $\leq$ 1.0, ECC is assumed.

If it is $>$ 1.0, HA assumed.

Execution of this subroutine defines the planetary shadow entry and exit points (Ref. Figure 4-7).

RELATED INFORMATION:

Refer to Figure 4.3 and Figure 4.4 for definition of terms.

SUBROUTINE NAME: ORBIT2

PURPOSE:

This subroutine defines spacecraft orbits and sun/star position-orbit relationship in the orbit coordinate system.

DEFINITIONS:

Variable Names			Default Values
PNAME	-	name of orbit-centered body	None
CIGMA	-	clock angle - X_o axis to solar vector projection	None
BETA	-	cone angle - Z_o axis to solar vector	None
CIGMAS	-	clock angle - X_o axis to star vector projection	None
BETAS	-	cone angle - Z_o axis to star vector projection	None
TIMEST	-	time of periapsis passage, hours	0.0
HP	-	altitude of periapsis	None
HA	-	altitude of apoapsis	None
ECC	-	orbit eccentricity	None

RESTRICTIONS:

None

CALLING SEQUENCE:

```
CALL ORBIT2 (PNAME, CIGMA, BETA, CIGMAS, BETAS, TIMEST, HP, HA)
CALL ORBIT2 (PNAME, CIGMA, BETA, CIGMAS, BETAS, TIMEST, HP, ECC)
```

NOTES:

See subroutine ORBIT1. This call not applicable to heliocentric orbits.

RELATED INFORMATION:

Refer to Figures 4.3 and 4.5 for definition of terms.

SUBROUTINE NAME: ORIENT

PURPOSE:

To define spacecraft orientation relative to orbital heat sources.

DEFINITIONS:

Variable Names		Default Values
TYPE	- orientation type	None
IROTX	- order of rotation about x-axis	1
IROTY	- order of rotation about y-axis	2
IROTZ	- order of rotation about z-axis	3
ROTX	- rotation about VCS x-axis to rotate VCS into CCS	0.
ROTY	- rotation about VCS y-axis to rotate VCS into CCS	0.
ROTZ	- rotation about VCS z-axis to rotate VCS into CCS	0.

NOTES:

TYPE options are as follows: 4HPLAN, 3HSUN, 4HSTAR, 4HTAPE.
Individual default values obtained by passing zero.

RESTRICTIONS:

Not recommended for use with DIDT2, DIDT2S. A call to ORIENT must precede a call to DIDT1 or DIDT1S.

RELATED INFORMATION:

Refer to Figure 4.6 for definition of terms

CALLING SEQUENCE:

```
CALL ORIENT (TYPE, IROTX, IROTY, IROTZ, ROTX, ROTY, ROTZ)
```

SUBROUTINE NAME: PLDATA

PURPOSE:

This subroutine defines parameters necessary to execute the data plotter.

DEFINITIONS:

Variable Name	Description	Options	Default
IPLUNT	Plot data flag (a composite Hollerith word)	Letter 1: A - Absorbed I - Incident Letter 2: F - Fluxes R - Rates Letters 3, 4, 5, & 6 (as required) S - Solar A - Albedo P - Planetary T - Total (Sum of SAP) ALL - ALL	None
IPLSN	Identifies steps to be plotted	A. 3HALL B. Name of Array of step numbers. Steps do not have to be in any order	3HALL
IPLNA	Identifies nodes to be plotted	A. 3HALL B. Name of Array of node numbers	3HALL
PLCRVF	Flag for curve-fitting	3HYES, 2HNO	3HYES

PLLABX	Plot label X	Array name (array length 28 characters max.)	Blanks
PLLABY	Plot label Y	Array name (array length 28 characters max.)	Blanks
PLTIT1	Plot label title line 1	Array name (array length 58 characters max.)	Blanks
PLTIT2	Plot label title line 2	Array name (array length 70 characters max.)	Blanks
PLXMPF	X-axis multiplying factor	Real no.	1.0
PLYMPF	Y-axis multiplying factor	Real no.	1.0
PLCMB	Plots output with correspondence applied for current configuration name.	3HYES, 2HNO	2HNO

RESTRICTIONS:

None

NOTE:

a. Examples of IPLUNT:

3HARP; plots absorbed rates, planetary
5HIFALL; plots all incident fluxes
5HAFSAP; plots solar, albedo and planetary absorbed fluxes.

b. Any set of dependent-and independent-variable data pairs may be plotted if IPLUNT = 1 and the data are written to disc/drum unit 1 in advance (reference Section 5.1.4).

CALLING SEQUENCE:

```
CALL PLDATA (IPLUNT, IPLSN, IPLNA, PLCRUF, PLLABX, PLLABY, PLTIT1,
PLTIT2, PLXMPF, PLYMPF, PLCMB)
```

SUBROUTINE NAME: QODATA

PURPOSE:

This subroutine used to define the absorbed heat output format to be obtained from the subsequent QOCAL execution.

DEFINITIONS:

Variable Names			Devault Value
NSARRY	-	array of step numbers where absorbed Q data is stored. Any order allowed. Options: array name, 3HALL.	3HALL*
NTMARY	-	thermal analyzer time array number (Q arrays numbered consecutively from NTMARY + 1)	1
QOTAPE	-	BCDOU tape output flag. Options: 4HTAPE, 2HNO	2HNO
QOPNCH	-	punch output flag. Options: 3HPUN, 2HNO	3HPUN(CDC) 2HNO(UNIVAC)
QOAMPF	-	area multiplication factor	1.0
QOFMPF	-	energy multiplication factor	1.0
QOTMPF	-	time multiplication factor	1.0
QOTYPE	-	type of output flag. Options: 3HTAB for Q vs time tables, 2HAV for orbtial average Q data, 4HBOTH for both	4HBOTH

*See Subroutine QOINIT

RESTRICTIONS:

Current geometry definition must agree with geometry of all steps in NSARRY.

NOTES:

Sort is made to obtain monotonically increasing time array. Trapezoidal-rule average made for orbital average heat tables.

QOAMPF applies only to areas on thermal analyzer subroutine call output.

QOFMPF applies only to heat flux array output.

QOTMPF applies to time array output and to value of period on subroutine call output.

CALLING SEQUENCE:

```
CALL QODATA (NSARRY, NTMARY, QOTAPE, QOPNCH, QOAMPF, QOFMPF, QOTMPF,
QOTYPE)
```

SUBROUTINE NAME: QOINIT

PURPOSE:

This subroutine rewinds the absorbed heat (NTQ) data storage file, thus providing user control of the number of time points obtained with NSARRY = 3HALL in subroutine QODATA. This will make previously stored absorbed heat data inaccessible to the absorbed heat output (QOCAL) segment.

CALLING SEQUENCE:

```
CALL QOINIT
```

RELATED INFORMATION:

See Subroutine QODATA.

SUBROUTINE NAME: RBDATA

PURPOSE:

This subroutine is used to define parameters used by RBCAL in computing form factors with real-body radiation effects.

Variable Names		Default Value
NFIGFF	- Configuration name for form factor access	Current config. name
FFACC	Reference FFDATA	
FFACCS	Reference FFDATA	
FFRATL	Reference FFDATA	
FFPRNT	Reference FFDATA	

NOTES:

1. If not called prior to RBCAL execution, default values are used.
2. Must utilize uncombined Form Factors.

CALLING SEQUENCE:

```
CALL RBDATA (NFIGFF, FFACC, FFACCS, FFRATL, FFPRNT)
```

SUBROUTINE NAME: RCDATA

PURPOSE:

This is a user-called subroutine that defines the parameters used in RCCAL for the condensation and output of radiation conductors (RADKs).

VARIABLE DESCRIPTION AND DEFAULT VALUES:

Variable	Description	Default Value
NFIGGB	Configuration name for gray body factor access	Current Config. Name
RKPNCH	Punch/no punch flag. Options: 3HPUN, 2HNO	3HPUN
RKMIN	Minimum value of $\mathcal{F}A/\epsilon A$ that will result in a valid RADK. RKMIN test is not made on conductors to the space node. If NERN is POSITIVE the RKMIN test is applied to the sum of those connections discarded after the RFRAC requirement is satisfied.	0.0001
IRKCN	Initial radiation conductor number	1
RKSP	Flag for calculation of RADKs to space. Options: 5HSPACE, 2HNO	2HNO
IRKNSP	Space node number	32767
SIGMA	Stefan-Boltzmann constant	1.713E-9
RKAMPF	Area multiplying factor	1.0

RKTAPE	Flag to write RADKs to BCDOU tape. Options: 4HTAPE, 2HNO. See Subroutine LIST for Related Information.	2HNO
NFIGCO	Configuration name for correspondence data access	Current config. name
RFRAC	Significant radiation fraction: radiation conductors of a node to be left intact divided by the sum of the node conductors	0.7
RTOL	Percentage of SLAST (last conductor value saved to meet RFRAC criterion). Subsequent conductors will be saved if their values are greater than RTOL * SLAST.	0.99
NERN	Effective radiation node (ERN) number. If NERN is negative, all ERN conductors will be printed but not punched or written to tape.	None
IPRIME	Array name for array of primary MESS node numbers and special node numbers	None
ISECND	Array name for array of secondary MESS node numbers	None

RESTRICTIONS:

RCDATA must be called prior to RCCAL execution since all of the variables are not defaulted.

IPRIME and ISECND arrays must be input in the array data block to specify MESS node pairs and special nodes. IPRIME ontains a list of all primary MESS nodes and all special nodes in that order. ISECND contains a list of all secondary MESS nodes in IPRIME.

CALLING SEQUENCE:

```
CALL RCDATA (NFIGGB, RKPNCH, RKMIN, IRKCN, RKSP, IRKNSP, SIGMA,
RKAMPF, RKTAPE, NFIGCO, RFRAC, RTOL, NERN, IPRIME, ISECND)
```

SUBROUTINE NAME: RKDATA

PURPOSE:

This subroutine defines parameters used in RKCAL for output of radiation conductors (RADKs).

Variable Names		Default Value
NFIGGB	Configuration name for gray body factor access	Current Config. name
RKPNCH	Punch/no punch flag. Options: 3HYES, 2HNO	3HYES
RKMIN	Minimum value of $\mathcal{F}_{ij}A_i/\ \epsilon_i A_i$ that will result in a valid RADK. Test not applied to conductors to space nodes.	0.0001
IRKCN	Initial radiation conductor number	1
RKSP	Flag for calculation of RADKs to space Options: 5HSPACE, 2HNO	2HNO
IRKNSP	Space node number	32767
SIGMA	Stefan-Boltzmann constant	1.713E-9
RKAMPF	Area multiplying factor	1.0
RKTAPE	Flag to write RADKs to BCDOU tape. Options: 4HTAPE, 2HNO. See Subroutine LIST for Related Information	2HNO
NFIGCO	Configuration name for correspondence data access	Current Config. name

NOTES:

If not called prior to RKCAL execution, default values will be assumed. Individual default values obtained by passing zero arguments.

CALLING SEQUENCE:

```
CALL RKDATA (NFIGGB, RKPNCH, RKMIN, IRKCN, RKSP, IRKNSP, SIGMA,
RKAMPF, RKTAPE, NFIGCO)
```

SUBROUTINE NAME: RSTOFF

PURPOSE:

This subroutine is used to discontinue the reading of data from an RSI tape during a restart run. All operations following a call to this routine are performed as though it was not a restart run, unless a call to subroutine RSTON is made later.

RESTRICTIONS:

Call valid only during a restart run from an RSI tape.

RELATED INFORMATION:

See Section 4.3.9: Restart Control Subroutines

CALLING SEQUENCE:

```
CALL RSTOFF
```

SUBROUTINE NAME: RSTON

PURPOSE:

This subroutine is used to re-establish the reading of data from an RSI tape following a call to RSTOFF.

RESTRICTIONS:

Call valid only during a restart run from an RSI tape.

RELATED INFORMATION:

See Section 4.3.9: Restart Control Subroutines.

CALLING SEQUENCE:

```
        CALL RSTON
```

NOTE:

The judicious use of RSTOFF in conjunctin with RSTON allows the user to insert, delete, and/or recalculate any operations in his Operations Data Block.

SUBROUTINE NAME: SPIN

PURPOSE:

Subroutine to define spacecraft spin rate, spin axis, and time of beginning of spin.

DEFINITIONS:

Variable Names			Default Values
CLOCK	-	clock angle - CCS x-axis to spin axis projection	0.
CONE	-	cone angle - CCS z-axis to spin axis	0.
RATE	-	spin rotation rate, revolutions/hour (positive clock-wise as viewed along spin axis from origin)	0.
TRUANS	-	true anomaly where spin begins	0.
SPNTM	-	time corresponding to TRUANS	0.

RESTRICTIONS:

Must be called subsequent to orbit definition through subroutines ORBIT1 or ORBIT2.

Subroutine SPIN cannot be used in conjunction with ORBGEN options INER and CIRP, and RATE not equal to zero.

NOTES:

a. The time at which spin begins may be defined either directly through SPNTM or through TRUANS. If SPNTM = 0, SPNTM is computed from TRUANS.

b. Spinning may be stopped only by a call to subroutine SPIN with RATE = 0. and spin stop time or true anomaly defined.

CALLING SEQUENCE:

```
CALL SPIN (CLOCK, CONE, RATE, TRUANS, SPNTM)
```

SUBROUTINE NAME: STFAQ

PURPOSE:

This subroutine stuffs known values of direct flux and absorbed heat from a previously executed step into current step data storage. It also defines time of current step, either directly or from true anomaly.

Variable Names		Default Values
TRUEAN	- current true anomaly, degrees	None
TIMEPR	- current time, hours	None
NSTP	- step number reference for known DI and AQ values	None

RESTRICTIONS:

Current geometry must agree with that of NSTP.

CALLING SEQUENCE:

```
CALL STFAQ (TRUEAN, TIMEPR, NSTP)
```

NOTE:

If TRUEAN.GT.0., time is computed from TRUEAN; otherwise TIMEPR is passed directly to current step data storage.

SUBROUTINE NAME: SURFP

PURPOSE:

This subroutine defines parameters for computing solar fluxes on a configuration located on a planet's surface.

DEFINITIONS:

Variable Names			Default Values
PNAME	-	name of planet (3HEAR, 3HMOO, 3HMAR)	None
ALAT	-	location on planet surface, degrees of latitude	None
SUNLAT*	-	latitude of subsolar point, degrees	None
AEX	-	Atmospheric extinction factor(or peak flux at solar noon, Btu/hr-ft^2)31	None

RESTRICTIONS:

None

CALLING SEQUENCE:

```
CALL SURFP (PNAME,ALAT,SUNLAT,AEX)
```

NOTE:

*For PNAME = 3HEAR, SUNLAT is input as day of year (1.   SUNLAT   365.)

APPENDIX E

SEGMENT DESCRIPTIONS

SEGMENT NAME: NPLOT

PURPOSE:

This segment generates pictorial plots of nodal surfaces.

RESTRICTIONS:

None

CALLING SEQUENCE: L NPLOT

SEGMENT NAME: OPLOT

PURPOSE:

This segment generates pictoral plots of the spacecraft in orbit.

RESTRICTIONS:

None

CALLING SEQUENCE: L OPLOT

SEGMENT NAME: PLOT

PURPOSE:

This segment generates function vs time plots of absorbed and incident heat rates and fluxes. When used in conjunction with operations block FORTRAN that writes data to a plot data unit, this segment provides general x vs y plot capability.

RESTRICTIONS:

Reference Subroutine PLDATA

CALLING SEQUENCE: L PLOT

SEGMENT NAME: FFCAL

PURPOSE:

This segment calculates all form factors for the active configuration.

CALLING SEQUENCE: L FFCAL

SEGMENT NAME: NFFCAL

PURPOSE:

This segment calculates high precision form factors using the Nusselt-sphere technique.

CALLING SEQUENCE: L NFFCAL

SEGMENT NAME: RBCAL

PURPOSE:

This segment computes all "image factors" for configurations containing one or more specular surfaces. It computes form factors from all nodes to images of all nodes, as seen in the specular surfaces, and adds these form factors to the "direct" form factors computed by FFCAL or NFFCAL to create "total" form factors or image factors.

SEGMENT NAME: CMCAL

PURPOSE:

This segment combines form factor matrices according to user-input correspondence data and auto-combine correspondence data for polygons.

CALLING SEQUENCE: L CMCAL

RESTRICTION: Call is meaningful only after an FFCAL or NFFCAL execution under the configuration name defined in CMDATA.

SEGMENT NAME: DICAL

PURPOSE:

This segment computes solar, planetary, and albedo irradiation incident on spacecraft nodes.

RESTRICTIONS:

Call valid only after previous calls have been made to define spacecraft geometry, location in space, characteristics and distances of heat source bodies, and computation accuracy parameters.

CALLING SEQUENCE: L DICAL

SEGMENT NAME: DRCAL

PURPOSE:

This segment computes the component of solar flux resulting from the image of the sun as seen in the specular-diffuse surfaces by each node. These components are added to the direct flux values computed by DICAL to obtain total direct flux.

RESTRICTION:

Call valid only after a previous call to DICAL, using the same configuration.

CALLING SEQUENCE: L DRCAL

SEGMENT NAME: SFCAL

PURPOSE:

a. Segment computes analytically and stores on tape tables of inter-node blockage (shadow) factors for use in direct irradiation calculations.

b. When a complete shadow factor file is supplied on the RSI tape, segment is executed in order to pass shadow tables into program storage and initialize DICAL to compute irradiation using shadow tables.

RESTRICTIONS:

None

CALLING SEQUENCE: L SFCAL

SEGMENT NAME RKCAL

PURPOSE:

This segment computes radiation conductor values and punches (at user's option) output data in thermal analyzer format. Output card images are printed.

RESTRICTIONS:

Call valid after spacecraft geometry is defined and matching form factor matrix is computed.

CALLING SEQUENCE: L RKCAL

SEGMENT NAME: RCCAL

PURPOSE:

This segment computes radiation conductors, simplifies and condenses these conductors using the ERN and MESS techniques, and provides output in punched card and/or BCD tape form.

RESTRICTIONS:

Same as RKCAL

CALLING SEQUENCE: L RCCAL

SEGMENT NAME: GBCAL

PURPOSE:

Segment computes and stores gray-body factor matrix.

RESTRICTIONS:

Same as RKCAL

CALLING SEQUENCE: L GBCAL

SEGMENT NAME: AQCAL

PURPOSE:

This segment computes absorbed heat rates in two wavebands, accounting for diffuse reflection.

RESTRICTIONS:

Appropriate direct irradiation, gray-body factors, and surface properties must be in system storage.

CALLING SEQUENCE: L AQCAL

SEGMENT NAME: QOCAL

PURPOSE:

This segment accesses absorbed flux data and generates orbital average and absorbed flux vs time arrays. Arrays are output in thermal analyzer format on cards or BCDOU tape.

RESTRICTIONS:

None

CALLING SEQUENCE: L QOCAL

APPENDIX F

RADIATION CONDENSER SEGMENT THEORY

A. BASIC CONCEPTS

The Multiple Enclosure Simplification Shield (MESS) technique and the Effective Radiation Node (ERN) technique are independent and can be discussed separately. Consider an N-node radiative enclosure that forms a section of a complex thermal model. The temperature of node i is a function of thermal radiation coupling and the applied heat load, Q_1(assume that heat loads resulting from conduction and convection are included in Q_1). The steady-state temperature of node i is then given by

$$T_i = \left[\left(\sum_{j=1}^{N} \sigma A_i F^*_{ij} T_j^4 + Q_i \right) / \sum_{j=1}^{N} \sigma A_i F^*_{ij} \right]^{\frac{1}{4}} \tag{1}$$

B. ERN TECHNIQUE

In applying the ERN technique, the enclosure radiation conductors for the ith node are divided into P_i primary and $N-P_i$ secondary couplings. The summation term in the numerator of Equation (1) can then be written as follows:

$$\sum_{j=1}^{N} \sigma A_i F_{ij} T_j^4 = \sum_{k=1}^{P_i} \sigma A_i F_{ik} T_k^4 + \sum_{\ell=P_i+1}^{N} A_i F_{i\ell} T_\ell^4 \tag{2}$$

The number of radiation conductors can be reduced by arranging the conductors in decreasing order of the conductor value ($A_i F_{ij}$) and replacing the secondary coupling summation of Equation (2) with a single conductor coupled to an ERN. That is,

$$\sum_{\ell=P_i+1}^{N} \sigma A_i F_{i\ell} T_\ell^4 = \left[\sum_{\ell=P_i+1}^{N} \sigma A_i F_{i\ell} \right] T_{ERN}^4 \tag{3}$$

---

*In Appendix F, the letter F shall denote the gray-body factor, $\mathfrak{F}$.

The ERN temperature is calculated by the thermal analyzer program as a steady-state node temperature based on a fourth-power, conductor-weighted average of the enclosure node temperatures using the secondary conductors.

$$T_{ERN} = \left[\sum_{i=1}^{N} \sum_{\ell=P_i+1}^{N} \sigma A_i F_{i\ell} T_i^4 \Big/ \sum_{i=1}^{N} \sum_{\ell=P_i+1}^{N} \sigma A_i F_{i\ell} \right]^{\frac{1}{4}} \qquad (4)$$

Using the relationships of Equatins (2) and (3), the approximate ith node temperature can be written from Equation (1) as a function of the ERN temperature.

$$T_i' = \left\{ \left[\sum_{k=1}^{P_i} \sigma A_i F_{ik} T_k^4 + \left(\sum_{\ell=P_{i+1}}^{N} \sigma A_i F_{i\ell} \right) T_{ERN}^4 + Q_i \right] \Big/ \sum_{j=1}^{N} \sigma A_i F_{ij} \right\}^{\frac{1}{4}} \qquad (5)$$

C. APPLICATION OF THE ERN TECHNIQUE

The significant radiation fraction defined by the relationship

$$\text{RFRAC} = \sum_{k=1}^{P_i} \sigma A_i F_{ik} \Big/ \sum_{j=1}^{N} \sigma A_i F_{ij} = \sum_{k=1}^{P_i} F_{ik} / \varepsilon_i \qquad (6)$$

is specified by the user. The number of primary conductors, P_i, is determined by summing conductor values for a given node until the sum is greater than the fraction RFRAC of the sum of all conductors to the node. That is,

$$\sum_{k=1}^{P_i} \sigma A_i F_{ik} > \text{RFRAC}* \sum_{j=1}^{N} \sigma A_i F_{ij} \qquad (7)$$

All primary and reverse direction conductors are flagged to be used intact. The secondary conductors for each node are summed to determine the conductor value for the node-to-ERN coupling.

Since the error in the approximate temperature is a function of the enclosure temperature band, the ERN technique results can be improved if nodes that deviate significantly from the average temperature of the enclosure are not coupled to the ERN. These analyst-defined nodes are referred to as special nodes.

The percentage reduction in enclosure conductors and subsequent network error as a result of applying the ERN technique are controlled by the analyst's selection of an RFRAC value consistent with the known accuracy of problem parameters (enclosure geometry, surface optical properties, etc).

Experience has shown that the greatest percentage reduction in conductors results for enclosures with more than 75 nodes, significant shadowing and low-emittance surfaces. An RFRAC value of 0.7 has been found to result in a significant reduction in conductors with acceptable error for typical radiation enclosures.

D. MESS TECHNIQUE

The MESS technique provides the analyst with a means of dividing a radiation enclosure into an arbitrary number of subenclosures.

MESS node pairs are defined by the analyst at the interface between subenclosures as two planar surfaces with the property of absorbing and emitting all energy incident upon them (black surfaces). Consider an N-node subenclosure, n, as shown in Figure F-1, where the subscripts r and r' refer to the MESS node pair of the nth and jth subenclosures, respectively. Temperatures in n are affected by $T_{MESS_{r'}}$, which represents the average thermal effect of the j subenclosure nodes on the nodes of n. The primary conductors of Equation (2) include conductors to MESS nodes. For a general subenclosure, n, with R_n interface MESS nodes, the primary radiation coupling summation for node i is:

$$\sum_{k=1}^{P_i} \sigma A_i F_{ik} T_k^4 = \sum_{r=1}^{R_n} \sigma A_i F_{ir} T_{MESS_{r'}}^4 + \sum_{\ell=R_n+1}^{P_i} \sigma A_i F_{i\ell} T_\ell^4 \qquad (8)$$

An energy balance on MESS node r' gives

$$T_{MESS_{r'}} = \Bigg[\Big(\sum_{\substack{m=1 \\ m \neq r'}}^{R_j} \sigma A_{r'} F_{r'm} T_{MESS_{m'}}^{4} + \sum_{k=R_j+1}^{P_{r'}} \sigma A_{r'} F_{r'k} T_k^{4} + \sigma A_{r'} F_{r'r} T_{MESS_r}^{4} \Big) \Big/ \Big(\sum_{\substack{m=1 \\ m \neq r'}}^{R_j} \sigma A_{r'} F_{r'm} + \sum_{k=R_j+1}^{P_{r'}} \sigma A_{r'} F_{r'k} + \sigma A_{r'} F_{r'r} \Big) \Bigg]^{\frac{1}{4}} \quad (9)$$

$F_{r'r}$ represents the reflections between n and j due to nonblack subenclosure surfaces and is obtained from the radiation interchange matrix for each subenclosure.

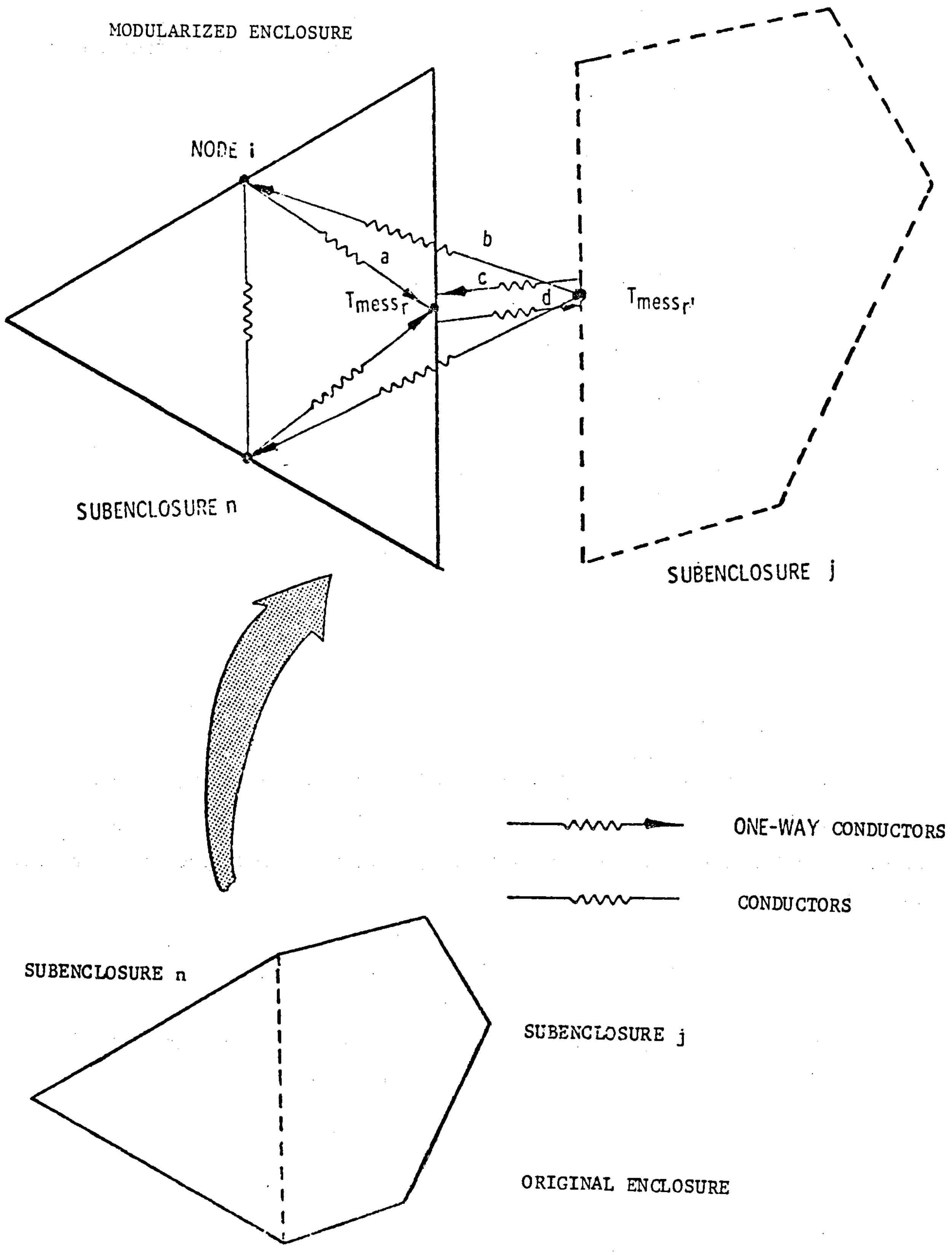


Figure F-1 MESS Technique One-Way Conductors

The approximate temperature of the ith node is obtained from Equation (5), using Equation (8), as

$$T_i' = \left\{ \left[\sum_{r=1}^{R_n} \sigma A_i F_{ir} T_{MESS_{r'}}^4 + \sum_{\ell=R_n+1}^{P_i} \sigma A_i F_{i\ell} T_\ell^4 + \left(\sum_{h=P_i+1}^{N} \sigma A_i F_{ih} \right) T_{ERN}^4 + Q_i \right] / \left(\sum_{r=1}^{R_n} \sigma A_i F_{ir} + \sum_{\ell=R_n+1}^{P_i} \sigma A_i F_{i\ell} + \sum_{h=P_i+1}^{N} \sigma A_i F_{ih} \right) \right\}^{\frac{1}{4}} \qquad (10)$$

The error in T_i' is a complex function of the percentage of ERN secondary conductors, temperature band of the subenclosure nodes, and the number os subenclosures. In a variety of problems studied, the error has been found to be negligible.

E. APPLICATION OF THE MESS TECHNIQUE

Generation of MESS one-way conductors from the subenclosure radiant interchange matrix requires that the analyst specify the interface MESS node pairs. As node conductors are generated, MESS nodes are flagged and appropriate one-way conductors are generated for use in the thermal analyzer program.

The location of MESS Node pairs in an enclosure is influenced by:

a. the number of subenclosure surfaces;

b. geometric considerations;

c. expected thermal gradients;

d. the number of analysts available to work on the enclosure.

Optimum reduction in form factors and conductors occurs in large enclosures divided such that the subenclosures contain approximately equal numbers of nodes. For enclosures divided into two approximately equal subenclosures, up to 50% reduction in the number of form factors and conductors can be expected.

F. ERN/MESS APPLICATION

The ERN and MESS techniques can be applied separately or simultaneously as the particular problem dictates. Whey they are applied simultaneously, an ERN is defined for each subenclosure and the MESS nodes are considered to be special nodes; that is, MESS nodes are not coupled to the ERN.

G. SUMMARY

The ERN/MESS technique reduces the number of form factors and radiation conductors necessary for enclosure radiation analysis and extends the analysis to include enclosures of arbitrary complexity. The use of the ERN/MESS technique can result in significant savings in time, both for the analyst and the computer.

APPENDIX G - TAPE NAME DESIGNATIONS AND SUMMARY INFORMATION

TAPE NAME	AVAILABILITY**	DESCRIPTION
NOUT	PP/P	Print Output File
DI	PP	Preprocessor Data Input File
RIO	PP/P	Random Access Data File
CMERG	PP	CMERGE Tape
EMERG	PP	EMERGE Tape
RSI	PP/P	Permanent Restart Input Tape
RSO	PP/P	Permanent Restart Output Tape
PNCH	PP	Punch Output File
SC1	PP	Scratch File
SC2	PP	Scratch File
SC3	PP	Scratch File
CMPL	PP	Logic Preprocessor Output File
SQNTL	PP/P	Sequential Data File
DIR	PP/P	Flux Data Block File
FFR	PP/P	Form Factor Data Block File

GBIRR	PP/P	Correspondence Data Input File
PLSR	PP/P	Shadow Factor Data File
RTI	P	Temporary Restart Input Tape
RTO	P	Temporary Restart Output Tape
BCDOU	P	BCD Data Output Tape
FILMPL	P	Plot Output File
TRAJ	PP/P	Trajectory Tape
USER1	P	User File
USER2	P	User File
TQR	P	Equivalent form factor information file.
FF	PP/P	Form Factor Data Storage File
GBIR	P	Infrared Grey-Body Storage File
GBSO	P	Solar Grey-Body Storage File
PLS	P	Planetary Form Factor Save File
TQ	P	Total Heat Rates Storage
SCR1	P	Scratch File

SCR2	P	Scratch File
SCR3	P	Scratch File
SCRR	P	Random Scratch File
RARR	P	Random I/O Data File
PUN	P	Punch Output File
DI	P	Direct Irradiation Data Storage File

**PP: Preprocessor
P: Processor

II. SUMMARY USER TAPE/FILE INFORMATION

A. Input Tapes/Files

1) RSI - Typical data stored are: model, FAs, GBs, RBs, SFs, DIs, and DRs. Source would be TRASYS II RSO tape.

2) RTI - Temporary storage of FA, GB, RB, DI, DR interim calculations. Source would be a TRASYS II RTO tape.

3) CMERG - Any BCD data source could be TRASYS I RTO or TRASYS II USER1 tape.

4) EMERG - Could consist of segment(s) of TRASYS input model. Source would be TRASYS II RSI/RSO tape.

5) TRAJ - Trajectory tape. Also used for FA and GB data. Source would be TRASYS I USER1 tape or MPAD trajectory tape.

B. Output Tapes/Files

1) RSO - Typical data stored are model, FAs, GBs, RBs, SFs, DIs, and DRs. Used as a TRASYS II RSI input tape.

2) RTO - Used for temporary storage of FA, GB, RB, DI, and DR interim calculations. Used as a TRASYS II RTI input tape.

3) USER1 - Typical BCD data stored are: FFs and DIs. Can be used as TRASYS II CMERG input tape. See Section II-c, Appendix G.

4) BCDOU - BCD data stored for thermal analyzer (SINDA) interface includes RADKs, cyclic and averaged heating rates.

C. Preprocessor Tapes/Files	Processor Tapes/Files
RSI	RSI[2]
RSO	RSO
CMERG[1]	---
EMERG[1]	---
	RTI[2]
	RTO
	BCDOU
	USER1
	TRAJ

[1]Freed internally within TRASYS runstream at end of preprocessor execution.

[2]Freed internally within TRASYS runstream when an end of file or parity error is encountered..

APPENDIX I

DEVELOPMENT OF EQUATIONS

FOR

DIFFUSE PLUS SPECULAR RADIATION ANALYSIS

I. ANALYTICAL DEVELOPMENT

The assumptions and groundrules and the development of the mathematical equations for diffuse-plus-specular radiation analysis are presented in this section.

1. Assumptions and Groundrules

The following assumptions and groundrules were used in the analytical development present herein for diffuse-plus-specular radiation analysis techniques.

a. All surfaces are considered to be semi-gray (accounts for absorptions and reflection, but no emission in the ultra-violet portion of the spectrum; accounts for absorption and reflection as well as emission in the infrared portion of the spectrum).

b. Equations are developed for use in analyzing radiation enclosures consisting of diffuse, specular, and/or diffuse-plus-specular surfaces using an imaging technique.

c. All surfaces are considered to emit diffusely and to reflect with diffuse and specular components such that the relationship

$$\epsilon_i + \rho_i^d + \rho_i^s + \tau_i = 1.$$

is satisfied.

d. All surfaces with specular components of reflectance are restricted to planar surfaces to simplify imaging.

e. Only first-order images are considered (that is, no images of images or images in images are generated).

2. Development of Equations

The development of equations for radiation interchange factors follows the same procedure for both the infrared and ultra-violet portions of the spectrum with the only differences being in notation. Therefore, only those equations applicable to the infrared portion of the spectrum are developed here.

Consider a radiation enclosure consisting of N surfaces. The net heat flux from any one of these surfaces can be represented by

$$q_{i,net} = \sigma \sum_{j=1}^{N} \mathcal{F}_{ij} \left(T_i^4 - T_j^4\right) \tag{1}$$

where $\mathcal{F}_{ij}$ is the radiation interchange factor that couples surface i to surface j.

The method of approach that is applied here in the development of radiation interchange factor ($\mathcal{F}_{ij}$) equations is an extension of the method set forth by Gebhart for purely diffuse enclosures (references 1, 2, and 3. The special utility in this formulation is that it yields coefficients which represent the fraction of energy emitted by a surface that is absorbed by another surface after reaching the absorbing surface by all possible paths.

Considering first-order images only, the general equation for the Gebhart-type absorption factors for a diffuse-plus-specular enclosure can be written

$$\beta_{ij} = \epsilon_j F_{ij} + \epsilon_j \sum_{k=1}^{N} \rho_k^s F_{ij(k)} + \sum_{m=1}^{N} \rho_m^d F_{im} \beta_{mj} + \sum_{k=1}^{N} \sum_{m=1}^{N} \rho_m^d \rho_k^s F_{im(k)} \beta_{mj}; \quad i=1, 2, \ldots, N; \; j=1,2,\ldots N \tag{2}$$

By means of a term by term examination, equation (2) can be interpreted as follows:

The fraction of the energy leaving surface i that is finally absorbed by surface j equals the sum of

a. the energy that goes directly from surface i to surface j and is absorbed,

b. the energy that goes from surface i to surface j by all possible first-order specular reflections and is absorbed,

c. that fraction of the energy that goes directly from surface i to each of the surfaces in the enclosure, finally arrives at surface j by all possible paths due to diffuse reflections, and is absorbed.

d. that fraction of the energy that goes from surface i to each of the surfaces in the enclosure by all possible first-order specular reflections, thence to surface j by all possible paths due to diffuse reflections, and is absorbed.

Rearranging the terms in equation (2) yields

$$\beta_{ij} = \epsilon_j \left[F_{ij} + \sum_{k=1}^{N} \rho_k^s \; F_{ij(k)}\right] + \sum_{m=1}^{N} \rho_m^d \left[F_{im} + \sum_{k=1}^{N} \rho_k^s F_{im(k)}\right] \beta_{mj} \; ;$$

$$i = 1,2,\ldots, N \quad , \quad j = 1,2,\ldots,N \qquad (3)$$

Equation (3) can be further simplified by defining an "image factor" (ϕ_{ij}) as that fraction of the energy that leaves surface i and arrives at surface j both directly and by all possible first-order specular reflections, such that

$$\phi_{ij} = F_{ij} + \sum_{k=1}^{N} \rho_k^s \quad F_{ij(k)};$$

$$i = 1,2,\ldots,N \quad ; \; j = 1,2,\ldots,N \qquad (4)$$

Substitution of equation (4) into equation (3) yields

$$\beta_{ij} = \epsilon_j \, \phi_{ij} + \sum_{m=1}^{N} \rho_m^d \; \phi_{im} \, \beta_{mj} \qquad (5)$$

Rearrangement of the terms in equation (5) yields

$$\sum_{m=1}^{N} (\delta_{im} - \rho_m^d \, \phi_{im}) \, \beta_{mj} = \epsilon_j \, \phi_{ij} \qquad (6)$$

Equation (6) can be represented in matrix form as

$$[D_{ij}] \, [\beta_{ij}] = [\epsilon_j \phi_{ij}] \qquad (7)$$

where D is an N X N coefficient matrix with general element

$$D_{ij} = \delta_{ij} - \rho_j^d \phi_{ij} \qquad (8)$$

The systems of equations represented by (7) can be solved by matrix inversion to obtain the absorption factors (β_{ij})

$$\beta_{ij} = \sum_{m=1}^{N} D_{im}^{-1} \; \epsilon_j \phi_{mj} \qquad (9)$$

The radiation interchange factors ($\mathcal{F}_{ij}$) are related to the absorption factors reference 1) by the expression

$$\mathcal{F}_{ij} = \epsilon_i \, \beta_{ij} \tag{10}$$

and, using the usual arguments for the conservation of energy, the reciprocity relation for the N^2 values of $\mathcal{F}_{ij}$ is

$$A_i \mathcal{F}_{ij} = A_j \mathcal{F}_{ji} \tag{11}$$

The foregoing equations apply to radiation enclosures consisting of any combination of diffuse and specular surfaces ranging from totally diffuse to totally specular enclosures.

When the problem consists of an "incomplete" enclosure and a space node is present, the radiant interchange factor to space is computed from:

$$A_i \mathcal{F}_{is} = A_i \;\; \epsilon_i - \sum_{j=1}^{N} A_i \mathcal{F}_{ij} \tag{12}$$

for an N-node incomplete enclosure.

II. REFERENCES

1. Gebhart, B., "Unified Treatment for Thermal Radiation Transfer Processes--Gray, Diffuse Radiators and Absorbers", Paper No. 57-A-34, ASME, December 1957.

2. Gebhart, B., Heat Transfer, McGraw-Hill Book Co., Inc., 1961, pp. 117-122.

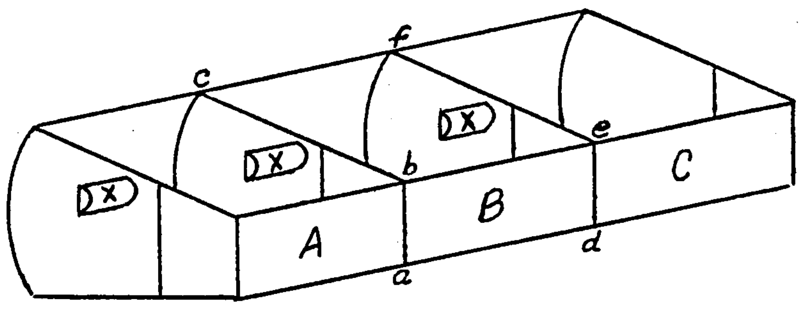
3. Gebhart, B., "Surface Temperature Calculations in Radiant Surroundings of Arbitrary Complexity--for Gray, Diffuse Radiation," International Journal of Heat Mass Transfer, Vol. 3, No. 4, 1961, pp. 341-346.

APPENDIX J

USE OF ADIABATIC CLOSURE SURFACES

A thermal analyst is sometimes confronted with the requirement to determine temperatures and heat flows in an enclosure that would require a great number of nodes to furnish adequate simulation, but also shows a great deal of symmetry and duplication of geometry.

An example of such an enclosure is shown below.



Subenclosures A, B and C each contain an identical heat source/sink node (X), and each subenclosure can "see" into the adjacent subenclosure with a significant view. If the thermal boundary conditions for each of the subenclosures are approximately the same, a condition of thermal radiation symmetry exists, that is, energy leaving B across plane abc is equivalent to the energy entering B across plane abc. The same can be said about the situation at plane def. Thus, planes abc and def are adiabatic surfaces in the sense that the net heat flow across them is zero. It is, therefore, correct thermal simulation to create a TRASYS model of subenclosure B with adiabatic "mirrors" at planes abc and def to simulate the presence of subenclosures A and C. Notice that this takes into account only the effects of subenclosures A and C. If a longer string of identical subenclosures exist and it is desired to account for them, two or more enclosures can be constructed and closed with adiabatic mirrors. A two enclosure model would thus account for the thermal affects of two enclosures on each end, a three enclosure model would account for three on each end, and so on.

Implementation of this procedure is quite simple using subroutine ADSURF. There is no need to define the shape or location of the two reflecting planes, although the true enclosure area of the reflecting planes is a required input. When the user needs one or more adiabatic reflector planes, he merely adds an extra BCS to his surface data and defines within it a single rudimentary surface with very high reflectivity. This surface definition might be as follows:

```
CC1     CC7
  BCS     ADSUR1
  S       SURFN = 10000, TYPE = RECT, PROP = .0001, .0001, ACTIVE = TOP
          P1 = 1.0, 1.0, 0.0
```

NOTE: BCS ADSUR1 can contain one and only one surface.

With the adiabatic surface defined thus, the area must be determined outside TRASYS and entered as an argument to subroutine ADSURF. If desired the adiabatic surface may be defined in the usual manner and its area will be determined by the program. A zero would then be used as the area argument in ADSURF.

With this surface in the surface data block, the user may apply the adiabatic "mirror" technique to an enclosure by a call to subroutine ADSURF subsequent to his form factor execution. If desired, the effect of the adiabatic surface can be determined in a single run by executing FFCAL or NFFCAL, GBCAL and RKCAL in the usual manner, followed by a call to ADSURF, followed by GBCAL and RKCAL execution to produce another set of radiation conductors, this time with the "mirrors" in place.

End of Document